長岡技術科学大学大学院
工科大学修士論文

題　　目

Transformer ベースを用いた ビデオフレーム補間に関する研究

指導教員
准教授　　杉田　泰則

著　　者
電気電子情報工学専攻
信号処理応用研究室
19315487 CHAU TUAN KIEN

令和 5 年 2 月 10 日

ABSTRACT

Transformer ベースを用いた ビデオフレーム補間に関する研究

Author: 19315487 TUAN KIEN CHAU
Supervisor: Associate Professor Yasunori Sugita

Video frame interpolation aims to generate new intermediate frames from successive frames of existing video. This is a fundamental problem in computer vision involving understanding motion, structure, and natural image distribution, and has many practical applications such as video reconstruction, slow-motion generation, and video compression.

Conventional frame interpolation can be broadly divided into flow-based, kernel-based, and attention-based methods. Flow-based methods estimate the optical flow between frames and synthesize intermediate frames according to the predicted optical flow. However, these methods rely on the accuracy of the optical flow and are difficult to estimate in areas subject to blurring and brightness variations. Kernel-based methods use convolutional neural networks (CNNs) to estimate spatially adaptive convolution kernels and convolve with input images to synthesize new frames. However, since a large kernel size is used to deal with large motions, there is a problem of high computational cost. The attention-based method extracts motion information by using an attention mechanism and a channel attention module that distributes the information of the input image to multiple channels through the Pixel Shuffle Layer and processes the channels. However, it is believed that the lack of processing of the time dimension information required for video frame interpolation can affect the accuracy of the resulting image.

In this paper, we propose a transformer-based method that learns the long-term dependencies of input image pixels in both space and time, with the aim of improving the accuracy of generated images. n the conventional method, the input image was 3D (C $\times$ H $\times$ W), but in order to consider the time dimension, the input image is changed to 4D (C $\times$ T $\times$ H $\times$ W) in the proposed method. Also, in order to learn spatial information in the input images from locally to globally, we used an encoder-decoder structure that scales down and scales up the image, unlike conventional methods that keep the size of the feature map as it is. In addition, we also applied an intermediate frame synthesis module for features with four dimensions.

In the experiment, a comparative evaluation was performed using the same learning data for the conventional CAIN method and the proposed method. Comparing the images

generated by the conventional method and the proposed method with PSNR and SSIM, which are evaluation indices, confirmed the usefulness of the proposed method. In addition, comparing the average values of PSNR and SSIM for all test data, in the case of the Vimeo90K dataset, the proposed method outperformed by 0.601 [dB] in PSNR and by 0.017 in SSIM. For UCF101 dataset, the outperformed values of PSNR and SSIM are 0.586 [dB] and 0.015. From the above, it was confirmed that the interpolation accuracy is improved by using a transformer that learns the long-term dependencies of the pixels of the input image in both space and time.

目次

第 1 章

はじめに

1.1 研究背景

ビデオフレーム補間は、既存ビデオの連続するフレームから新たな中間フレームを生成することを目的としている。これは、動き、構造、および自然な画像分布の理解を含むコンピュータビジョンにおいて重要な技術の一つであり、ビデオ復元、スローモーション生成、およびビデオ圧縮などの多数の実用的なアプリケーションがある。

既存のビデオフレーム補間方法では、フローベース、カーネルベース、アテションベースの方法に大別できる。フローベース方法 [1][2][3][4] は、フレーム間のオプティカルフローを推定し、推定されたオプティカルフローに従って入力画像をワープすることによって中間フレームを合成する。これらの方法には、有望的な結果が得られるが、次のような様々な問題もある。まず、オプティカルフローの精度に大きく依存しており、ぼやけや輝度変化を従う領域では推定が困難である。次に、ほとんどの方法は、フレーム間の線形モーションの想定に基づいており、仮定に違反する多くの現実世界のシナリオでパフォーマンスが制限される。

カーネルベース方法 [5][6][7] は、フローベースのアプローチと異なり、規定された仮定に依存しない。その代わりに、畳み込みニューラルネットワーク（CNNs）を使用して空間適応畳み込みカーネルを推定し、入力画像と畳み込むことで新たなフーレムを合成する。これらの方法では、大きなモーションに対応するために、大きなカーネルサイズが用いられる。その結果、大量なメモリが必要になる。また、事前に定義されたカーネルサイズより大きな動きの場合、うまく処理できないという問題点もある。

アテンションベース方法 [8] では、モデルの複雑さと計算コストを抑えるために、カーネルやオプティカルフローの推定を行わずに、PixelShuffle[9] とチャネル注意モジュール [10] を利用してビデオを補間する CAIN[8] と呼ばれる残差ネットワークが提案された。その方法の主なアイデアは、PixelShuffle [9] を介して入力画像の情報を

複数のチャネルに分散し、チャネルを処理するチャネルアテンションモジュールを用いることでモーション情報を抽出する。しかし、CAIN は入力画像の空間及びチャネル情報だけ処理してビデオフレーム補間に重要である時間次元の情報の処理が欠如するために、生成画像の精度に影響を与える可能性がると考えられる。

1.2　研究目的

本論文では、生成画像の精度を向上することを目的として、空間と時間の両方で入力画像のピクセルの長期的な関連度を学習するトランスフォーマー [11] を用いる手法を提案する。CAIN という従来法では、入力画像は 3 次元 ($C \times H \times W$) として実験を行ったが、時間次元を考慮するために、提案法で、入力画像は 4 次元 ($C \times T \times H \times W$) に変更した。また、入力画像のピクセルの長期的な関連度をうまく学習するために、入力画像のサイズをそのままにする従来法と異なり、画像を縮小・拡大するエンコーダー デコーダー構造を使用した。更に、4 次元をもつ特徴量に対応する中間フレーム合成モジュール [12] も適用した。

1.3　本論文の構成

本論文の構成は次の通りである。第 1 章では、本論文の研究背景及び目的を述べた。第 2 章では、関連研究について述べる。第 3 章では、本研究の基礎となる知識について述べる。第 4 章では、提案手法である Transformer ベースを用いたビデオフレーム補間について述べる。第 5 章では、提案法と従来法の比較実験を行い、提案法の有効性を示す。第 6 章では、本論文を通してのまとめと今後の課題を述べる。

第 2 章

関連研究

2.1 CAIN

2.1.1 全体モデル

ビデオフレーム補間に関する研究として、PixelShuffle[9] とチャネル注意モジュール [10] を利用し、中間フレーム合成を行う残差ネットワークがあり、CAIN[8] と呼ばれている。CAIN は、ビデオフレーム補間のトップ パフォーマーの 1 つであり、フローの推定、適応畳み込みカーネル、またはワーピングを必要がない。ここでは、この手法について説明する。

図 2.1 に CAIN のモデル構造を示す。まず、2 つの入力フレーム $\{\mathbf{I}_1, \mathbf{I}_2\} \in \mathbb{R}^{3\times H\times W}$ をそれぞれ $\{\tilde{\mathbf{I}}_1, \tilde{\mathbf{I}}_2\} \in \mathbb{R}^{192\times H/8\times W/8}$ にダウシャッフルし (係数 s=8)、チャネル次元に連結して結合フレーム $\tilde{\mathbf{I}} \in \mathbb{R}^{384\times H/8\times W/8}$ を構築する。(3×3) 畳み込みを適用し、チャネルの次元を 192 に減らす。変換された入力特徴マップは、チャネル注意を持つ 5 つの残差グレープ (ResGroup) を通過して、$\tilde{\mathbf{I}}_{1,2}$ 示されるダウンシャッフルされたターゲットフレームを学習する。残差グレープに通る特徴マップのサイズ $(192 \times H/8 \times W/8)$ はそのままであり、最終的な中間フレームは、アップシャッフル (係数 s=8) で $\hat{\mathbf{I}}_{1,2} \in \mathbb{R}^{3\times H\times W}$ の元のサイズ を合成する。

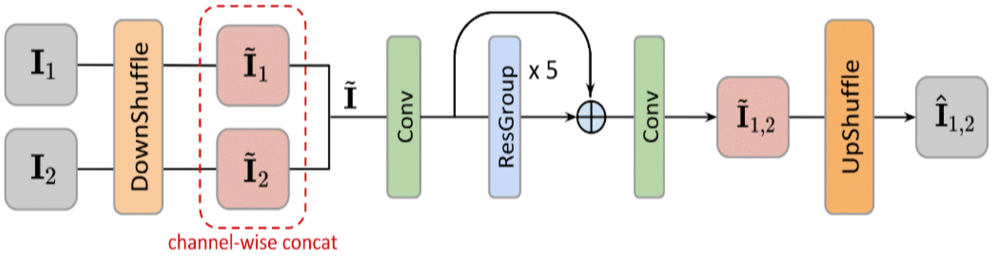

図 2.1　The overall architecture of CAIN ([8] より引用)

2.1.2　Pixel Shuffle

Pixel Shuffle[9] は、パラメータなしで、情報を失うことなく、画像（または特徴マップ）のレイアウトを再配置する操作である。ダウンシャッフルでは、画像 $\mathbf{I} \in \mathbb{R}^{C\times H\times W}$ の空間次元を s の係数で縮小し、チャネル次元を s^2 の係数を増加することで、結果として $\mathbb{R}^{s^2C\times H/s\times W/s}$ のテンソルが得られる。アップシャッフルは逆の操作を指す。

図 2.2 は係数 s=2 の時の両方の操作を視覚化したものを示す。左側の画像 $\mathbf{I} \in \mathbb{R}^{1\times 4\times 6}$ とすると、$s \times s = 2 \times 2 = 4$ つのピクセルを含む N 個のウィンドウに分けられる（ここで、$N = \frac{(H\times W)}{(s\times s)} = \frac{(4\times 6)}{(2\times 2)} = 6$）。各ウィンドウにあるピクセルをチャネル次元に沿って並べると、右側のテンソル $\in \mathbb{R}^{4\times 2\times 3}$ が得られる。

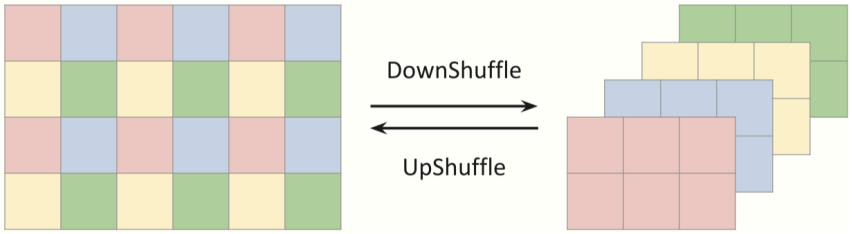

図 2.2　The visualization of Pixel Shuffle operation ([8] より引用)

2.1.3　残差グループ (ResGroup) の構造

これから、残差グループ (ResGroup) の構造を詳しく説明する。図 2.3(a) に ResGroup の構造を示す。ダウンシャッフルされた特徴マップの後に、５つの残差ブロックが続く。それぞれが１２個残差チャネルアテンションブロック（RCAB）で構成され、合計で６０個の RCAB がある。各 RCAB には２つの（３ x ３）の畳み込み層が含まれており、間に整流線形ユニット（ReLU）アクティベーションがあり、Skip Connection の前にチャネル注意（CA) モジュールが配置される。

CA モジュールのアーキテクチャは図 2.3(b) に示す。入力特徴マップを $\mathbf{F} \in \mathbb{R}^{H'\times W'\times C'}$ とすると、グローバル平均プーリングにより、入力特徴マップのチャネルごとのグローバル情報 $\mathbf{F}_c \in \mathbb{R}^{1\times 1\times C'}$ を取得する。次の２つの（１ x １）の畳み込み層は、非線形のチャネル間の関係を捉えることである。

正式に、チャネル注意メカニズムの重みは次のように計算される。

$$\text{att}\left(\mathbf{F}^c\right) = \sigma\left(\mathbf{W}_1 * \left(\text{ReLU}\left(\mathbf{W}_0 * \mathbf{F}^c\right)\right)\right) \tag{2.1}$$

ただし、$\sigma(\cdot)$ はシグモイド関数、$\mathbf{W}_0$ と $\mathbf{W}_1$ 2 つの 1×1 の畳み込み層の重みである。チャネル注意モジュールの最終的な出力は、入力特徴マップ $\mathbf{F}$ と取得した注意モジュールの重み att $(\mathbf{F}^c)$ の要素ごとの積として計算される。

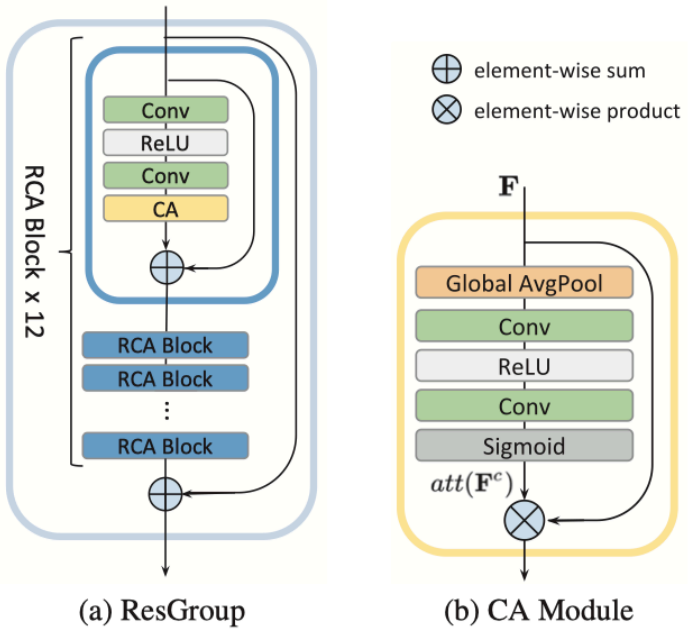

図 2.3　Illustration of ResGroup and CA module ([8] より引用)

2.2　問題点

従来法では、ダウンシャフルされた 2 枚の入力画像をチャネル次元に沿って連結され、1 つの特徴量としてモデルに入力した。それは、入力画像間の時間情報をキャプチャしなかった。それによって、生成画像の結果に影響を与えると考えられる。また、CAIN では、入力画像の空間情報を学習するために、畳み込み層を使用した。畳み込み層は、同じ畳み込みカーネルを使用、入力の異なる位置で畳み込み演算を行う。それにより、画像内のすべてのピクセルが同等に扱われるために、空間的に変化する複雑な動きを含むビデオ補間に適してない。

第 3 章

本研究の基礎となる知識

時間次元の情報に対応するために、モデルに入力する画像のディメンションを変更した。従来法では、２枚の入力画像を $\{\mathbf{I}_1, \mathbf{I}_2\} \in \mathbb{R}^{C\times H\times W}$（$C, H, W$ はそれぞれ画像のチャネル、高さ、幅の次元である）としてチャネル次元に沿って連結し、モデルに入力した。つまり、結合フレーム $\tilde{\mathbf{I}}$ のディメンションは変わらず、３次元 $(2C \times H' \times W')$ である。時間次元に対応するために、本論文で、２枚の入力画像を新しい次元に沿って連結することにした。つまり、$\tilde{\mathbf{I}}$ のディメンションは４次元 $(C \times T \times H' \times W')$ になった。ここで、T は入力画像の数であり、本研究で２に設定した。これから、モデルに入力する画像のディメンションは４次元として実験を行う。

3.1　Pixel Shuffle の変更

４次元の入力画像に対応するために、Pixel Shuffle を変更した。画像 $\mathbf{I} \in \mathbb{R}^{C\times T\times H\times W}$ とすると、ダウンシャッフルでは、s の係数で結果として $\mathbb{R}^{s^2C\times T\times H/s\times W/s}$ のテンソルが得られる。アップシャッフルは逆の操作である。

3.2　トランスフォーマーの挿入

3.2.1　トランスフォーマーの概念

トランスフォーマー [13] は、入力と出力の間の長期的な依存関係（または関連度）を効率的に捉える自己注意メカニズムを使用することにより、自然言語処理（NLP）用に最初に用いられた [13]。NLP の成功に動機つけられ、最近、トランスフォーマーをコンピュータービションに対応させた手法が提案された [14]。この手法のトランフォーマー [14] は、入力要素（ピクセルなど）をグローバルに相互作用する自己注意層に基づいている。しかし、このグローバル操作は画像サイズの二次複雑度を持って

いるために、ビジョンアプリケーションに適していない。この問題を克服するために、SWIN Transformer[15] が提案された。画像を分割する重複しないウィンドウ内で局所的に自己注意を計算し、画像サイズに対して線形の複雑度を持つ。また、シフト ウィンドウを使用することで、長期的な依存関係をモデル化する機能を保持しながら、複雑さの問題に対処できる。しかし、SWIN Transformer は画像入力にのみ適しており、時間次元が関係するビデオフレーム補間タスクには簡単に使用できない。そのために、SWIN Transformer[15] に基づいて、空間と時間の両方で長期的な依存関係をキャプチャする自己注意トランスフォーマー (Sep-STS)[11] を提案した。

本論文で、提案された Sep-STS を用いて、CAIN を拡張する。

3.2.2　Sep-STS

時間次元の関係に対応させるために、Sep-STS は入力画像の時空間計算を空間と時間の処理に分離する。

まず、空間での計算のために、入力画像のサイズ ($C \times T \times H \times W$) が与える場合、$C, T, H, W$ はそれぞれチャネル、時間、高さ及び幅の次元を表し、最初に、図 3.1(a) の左に示すように、$C \times \frac{THW}{M^2} \times M \times M$ サイズの重複していない 2D サブウィンドウに分割する（$\frac{THW}{M^2}$ はウィンドウの総数である）。次に、各サブウィンドウに対して、自己注意を実行する。サブウィンドウ特徴量 $\mathbf{X} \in \mathbb{R}^{C \times M \times M}$ の場合、クエリ Q、キー K 及び、値 V は次のように計算される

$$Q = XP_Q, K = XP_K, V = XP_V \tag{3.1}$$

ただし、P_Q, P_K, P_V は異なるウィンドウ間で共有される射影行列である。一般的に、$Q, K, V \in \mathbb{R}^{d \times M \times M}$。したがって、アテンションマトリックスは、サブウィンドウ内の自己アテンションメカニズムによって次のように計算される。

$$\text{Attention}(Q, K, V) = \text{SoftMax}(QK^T/\sqrt{d} + B)V \tag{3.2}$$

ここで、B は学習可能な相対一エンコーディングである。実際には、[13] に従って、注意関数を h 回並列に実行し、マルチヘッド自己注意 (MSA) の結果を連結する。

異なるローカルリージョンの情報を接続するために、図 3.1(a) の右に示すように、通常のシフトウィンドウパーティションを使用し、長期的な依存関係モデリングを可能にする。

次に、時間での計算のために、図 3.1(b) に、入力特徴マップを長さ T の HW 時間ベクトルに再形成し、フレーム間の依存関係をモデル化できるように、各ベクトル内で MSA を実行する。この 2 つの操作は一緒に用い、ビデオを処理する必要がある。

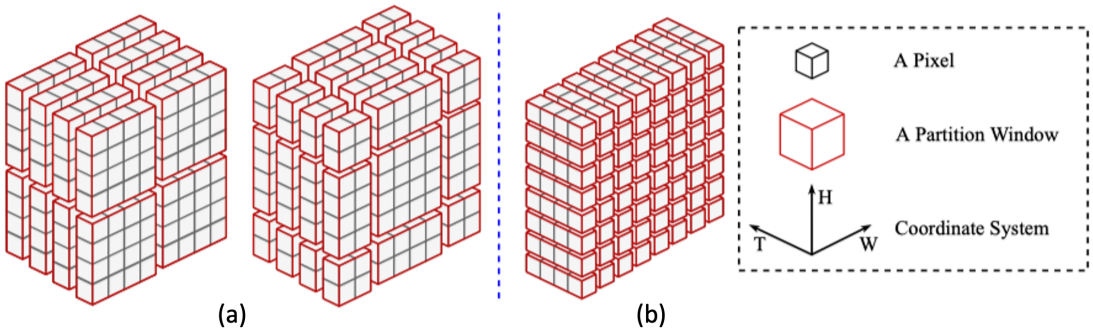

図 3.1　Illustation of Sep-STS([11] より引用)

3.3　転置畳み込み層 (Transposed Convolution Layer)

転置畳み込み層は、学習可能なパラメータがあり、入力画像の空間次元より大きな空間次元を持つ出力画像を生成する（またはアップサンプリングする）のに用いられる。

転置畳み込み層の計算は次のようになる。

1. (2×2) のインプットに (2×2) のカーネルを使って (3×3) のアウトプットにアップサンプリングする場合を考える。

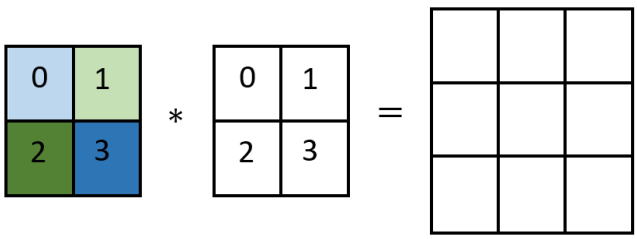

図 3.2　Transpose Convolution Layer Operation - Step 1

2. 次に、インプットからまず左上の一つだけを取り出す。そして、カーネルの各要素と掛け算をし、それをアウトプットの左端に並べる。

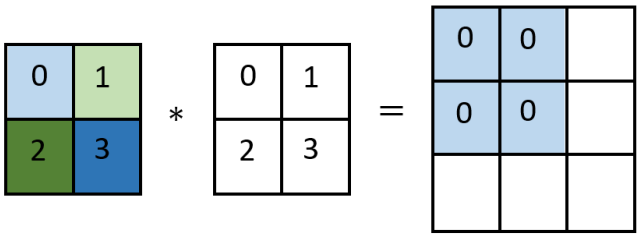

図 3.3　Transpose Convolution Layer Operation - Step 2

3. つづいてインプットを一つ右にずらす。同様にその部分とカーネルを掛け算し、その結果をアウトプットの右に一つずらした場所に設定する。同様に、左下、右下と計算する。

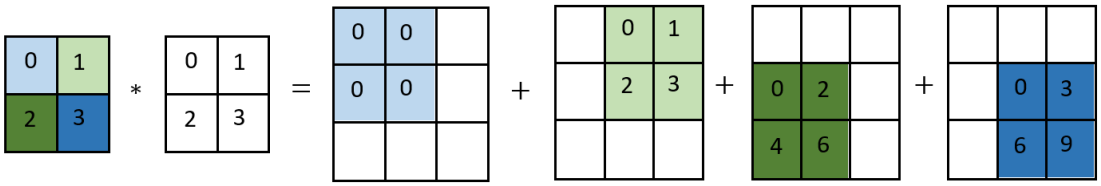

図 3.4　Transpose Convolution Layer Operation - Step 3

4. 最後に上記で計算された各数値を位置ごとに合計する。

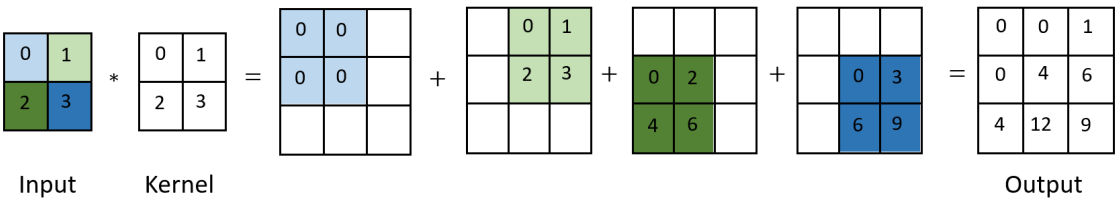

図 3.5　Transpose Convolution Layer Operation - Step 4

ここでは、1 つのチャネルが入力（$1 \times H \times W$）でアウトプットも 1 チャネル（$1 \times H' \times W'$）生成されるが、C 個のチャネルがある場合、（$C \times H' \times W'$）のアウトプットが得られる。

3.4　Adaptive Collaboration of Flows (AdaCoF)

AdaCoF [12] はオプティカルフローワープと学習した適応ローカルカーネルを組む合わせることで、フローベースとカーネルベースの両方の方法を統合するモデルである。このモデルのアイディアは、中間フレームの各ピクセル値を推定するための情報を含む入力フレーム内の任意の場所にある任意の数のピクセルを参照できる操作を作成することである。ターゲットピクセルを合成するには、オフセットベクトルと呼ばれる、参照位置を指す複数のフローを推定し、それらをサンプリングする。次に、サンプリングされた値をバイリニア補間により、ターゲット ピクセルを取得する。更に、オクルージョン マップも追加して、参照ピクセルの 1 つオクルードされたときに、2 つの入力画像の 1 つだけ使用する。

本論文で、AdaCoF を利用して、提案モデルに応用する。

第 4 章

提案手法

4.1 RSTSCAGroup の提案

今回、従来法の残差グループに注目して変更した。[16] のモデルを参考にし、Sep-STS を適用して、Residual Spatial Temporal Swin transformer Channel Attention Group (RSTSCAGroup) を提案した。図 4.1(a) に RSTSCAGroup を示す。RSTSCAGroup は N 個の RSTSCABlock から構成される。RCAB と異なり、２つの畳み込み層の代わりに、２つの Sep-STS ブロックを使用する。連続する２つの Sep-STS ブロックごとにチャネルアテンションモジュールは２つの Sep-STS ブロックの入力でチャネル統計を生成し、生成されたアテンションに２つの Sep-STS ブロックの出力と乗算する。RSTSCABlock に Skip Connection もあり、RSTSCABlock が入力と出力特徴量の違いに注目することが保証される。各 RSTSCABlock のチャネルアテンションモジュールはパラメータを共有する。

チャネルアテンションモジュールの構造は、時空間情報より適切にエンコードするために、2D 畳み込み層ではなく、3D 畳み込み層を使用した。（図 4.1(b) に示す）

Sep-STS に基づいて、[11] で Sep-STS ブロックを提案した。これは、分離された空間的及び時間的 Sep-STS と多層パーセプトロン（MLP）で構成されている（図 4.1(c)）。MLP は 2 層構造を採用し、活性化には GELU 関数 [17] を使用する。SWIN Transformer[15] と同様に、トレーニングを安定させるために、Layer Normalization (LayerNorm)[18] と Residual Connections[19] を適用する。また、連続する Sep-STS ブロックに通常のパーティションとシフトされたパーティションを交互に使用して、長距離の時空間依存関係をモデル化する。

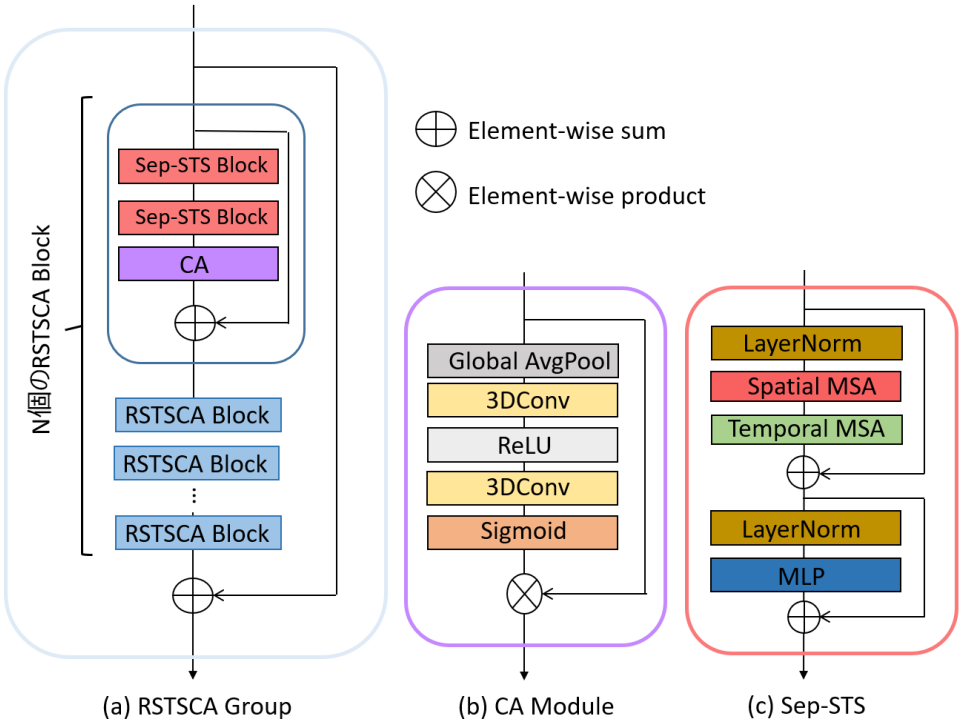

図 4.1　Illustration of RSTSCAGroup, CA Module, and Sep-STS Block

4.2　残差ブロック (Residual Block - ResBlock)

ResNet モデル [19] に基づいて、時間次元に対応する ResBlock を作成した。ここで、入力フレームの時空間特徴を学習するために、3D 畳み込み層を採用した。ResBlock は、ReLU を間に挟んだ 2 つにの 3D 畳み込み層及び、残差結合も適用され、図 4.2 に示す。

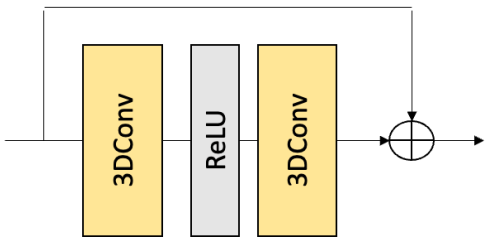

図 4.2　ResBlock architecture

4.3　提案モデル

2つの入力フレーム $\mathbf{I}_1$ と $\mathbf{I}_2$ が与えられる場合、ビデオフレーム補間は中間フレーム $\hat{\mathbf{I}}_{1.5}$ を合成することである。図 4.3 に提案法のモデル構造を示す。提案モデルは、浅層特徴量抽出、深層特徴量抽出及び、最終的なフレーム合成の 3 つのモジュールで構成される。まず、浅い特徴量抽出モジュールは残差ブロック（ResBlock）から構成され入力フレームを受け取り、任意のチャネル数（C と表記）を持つ浅い特徴量を生成する。次に、浅い特徴量を深層特徴量抽出モジュールに入力し、特徴表現を抽出してモーション情報を取得する。最後、フレーム合成モジュールは深い特徴量を使用して中間フレーム $\hat{\mathbf{I}}_{1.5}$ を生成する。

深層特徴量抽出モジュールでは、特徴量を固定なサイズを通る従来法と異なり、提案モデルでは、エンコーダーデコーダー　アーキテクチャに変更した（詳細は付録 A.1 に説明）。従来法では RCAB で畳み込み層の連続を使用することで入力に対して広い領域の特徴を学習可能となる。しかし、トランスフォーマーの連続にすると、深い特徴量の学習に役立つが、学習範囲は前者より小さい。したがって、エンコーダーデコーダーに変更することで入力画像ピクセルの長期的な関連度を計算し入力画像に対してローカルからグローバルの情報を両方学習可能となる。また、Pixel Shuffle を利用することにより、ダウンサンプリングの際、学習情報を失うことがない。

深層特徴量抽出モジュールの構造として、エンコーダーとデコーダーがある。エンコーダーでは、4 つのステージで構成される。ステージ 1 では、係数 $s = 2$ を持つダウンシャッフルの Pixel Shuffle から始まり、入力特徴量を $(\frac{H}{2} \times \frac{W}{2})$ にダウンサンプリングし、$4C$ のチャネル数の特徴量に (1×1) の 3D 畳み込み層を適用して、$2C$ のチャネル数に減らす。ダウンサンプリング層の後に、ネットワークの主要コンポーネントである RSTSCAGroup が続き、解像度は $(\frac{H}{2} \times \frac{W}{2})$ に保たれる。ステージ 2，3，4 はステージ 1 と同じ、出力解像度をそれぞれ $(\frac{H}{4} \times \frac{W}{4})$、$(\frac{H}{8} \times \frac{W}{8})$、$(\frac{H}{16} \times \frac{W}{16})$ にする。

デコーダーでは、3 つのステージがある。ステージ 1 は、係数 $s = 2$ を持つアップシャッフルの Pixel Shuffle 層を用い、低解像度の特徴量を $(\frac{H}{16} \times \frac{W}{16})$ から $(\frac{H}{8} \times \frac{W}{8})$ にアップサンプリングする。また、Skip Connection も採用され、特徴量をエンコーダーの前のレイヤーからデコーダーの後のレイヤーに直接送る。PixelShuffle は、解像度を向上させ、チャネル数を減らすことができるが、各チャネルの特徴量値間の接続を破壊するため、残差ブロック（ResBlock）を使用して特徴量値間を再構築しながら、$16C$ から $8C$ のチャネル数に減らす。ステージ 2，3 はステージ 1 と同じ、出力解像度をそれぞれ $(\frac{H}{4} \times \frac{W}{4})$、$(\frac{H}{2} \times \frac{W}{2})$ になる。

フレーム合成モジュールでは、AdaCoF のフレーム合成モジュールを用いた。ただ

し、特徴量を最適にアップサンプリングするために、バイリニア補間ではなく、転置畳み込み層 (Transposed Convolution Layer) を利用した。

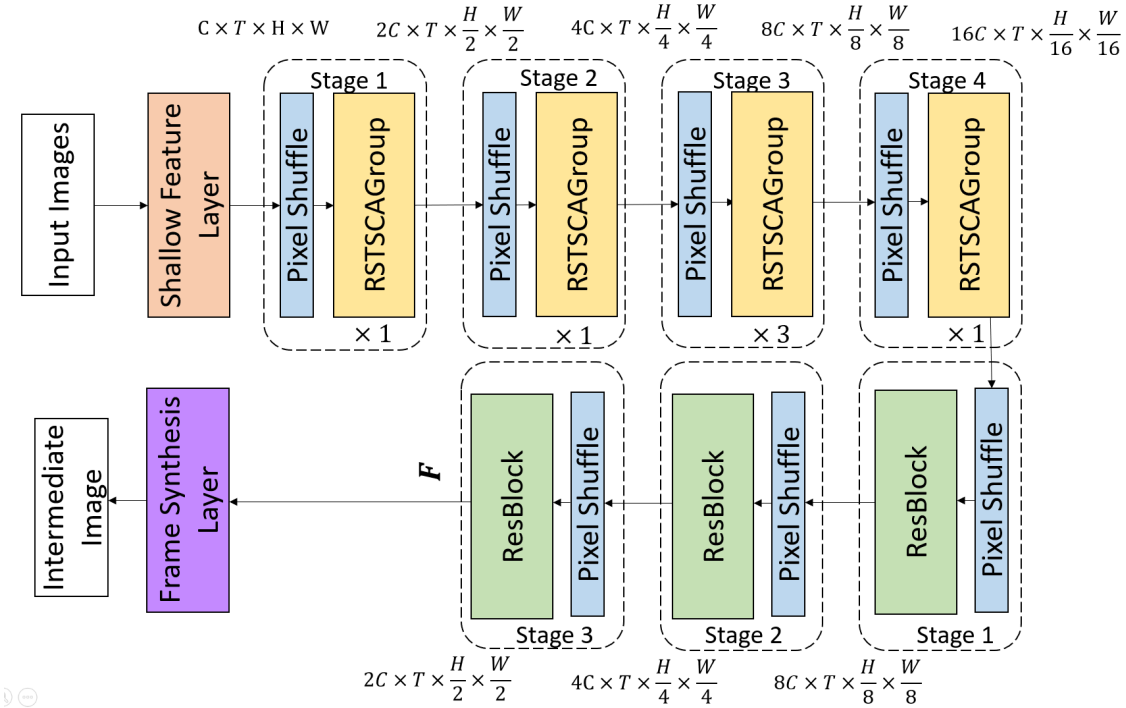

図 4.3　Propose model architecture

4.4　フレーム合成モジュール

本論文で用いられたフレーム合成モジュールの構造は図 4.4 に示す。深い特徴量 $F \in \mathbb{R}^{2C\times T\times \frac{H}{2}\times \frac{W}{2}}$ が与えられる場合、フレーム合成モジュールは、オフセットベクトルのセットを推定し、入力画像からの情報を集約することにより、中間フレームを生成する。

図 4.4 に示すように、まず、時間次元で F をアンバインドして、入力フレームの T 個の個別の特徴量を取得する。各フレーム t に対して $F_t \in \mathbb{R}^{2C\times \frac{H}{2}\times \frac{W}{2}}$ $(t = 1, 2)$。次に、F_t を 3 つの小さな CNNs に通して入力フレーム I_t に対するカーネル重み $W_t \in \mathbb{R}^{K^2\times H\times W}$、水平オフセットベクトル $\alpha_t \in \mathbb{R}^{K^2\times H\times W}$ 及び、垂直オフセットベクトル $\beta_t \in \mathbb{R}^{K^2\times H\times W}$ を推定する（K は各カーネルのサンプリング位置の数である）。フレーム t の位置 (x, y) でフレーム合成モジュールの出力は式 4.1 のように計算される。

$$O_t(x, y) = \sum_{k=1}^{K} W_t(k, x, y) I_t(x + \alpha_t(k, x, y), y + \beta_t(k, x, y)) \tag{4.1}$$

更に、オクルージョンも考慮する。オクルージョンの場合、入力画像の 1 つでター

ゲット ピクセルが見えなくなる。したがって、オクルージョン マップ $V \in [0,1]^{H\times W}$ を定義し、最終的な中間フレームは次の式に生成される。

$$\hat{I}_{1.5} = V \odot O_1 + (J - V) \odot O_2 \tag{4.2}$$

ここで、$\odot$ はアダマール積、J は 1 の $H \times W$ 行列。ターゲット ピクセル (x, y) の場合、$V(x, y) = 1$ はピクセルが I_1 でのみ表示されることを意味し、$V(x, y) = 0$ は I_2 でのみ表示されることを意味する。V にシグモイド活性化を使用して $V \in [0,1]^{H\times W}$ を満たす。

オクルージョン マップ V を取得するために、チャネル次元で特徴量 F_t を連結し、連結された特徴量を小さな CNNs に送信して V を生成する。用いられた CNNs は 2 つの 2D 畳み込み層と 1 つのアップサンプリング層から構成される。本論文で、アップサンプリング層はバイリニア補間ではなく、転置畳み込み層を使用した。バイリニア補間は、入力画像の周囲の 4 つ画素の距離に基づいて存在しない出力画素を計算し、画像を拡大することである。しかし、この方法は学習可能なパラメータがなく、そして、決まった距離で画素を計算するので、ぼやけな画像を生成してしまうことがある。そのために、学習可能なパラメータを持つ転置畳み込み層に変更した。

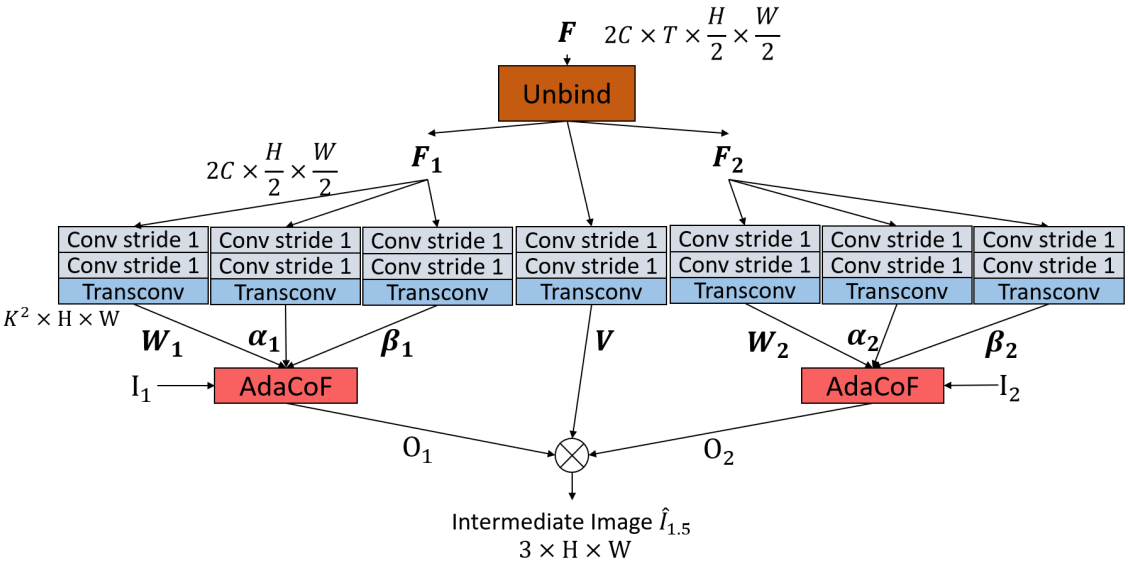

図 4.4　Frame Synthesis Module

4.5　モデルの設定

RSTSCAGroup の設定に関してウィンドウサイズ M は 8、RSTSCABlock 数は各ステージに対して 1 – 1 – 3 – 1、C も 32 に設定した。また、フレーム合成モジュールの

カーネルサイズ K は (5×5) にした。

ネットワークの各ステージの詳細情報は表 4.1 に示す。（B：バッチサイズ）

表 4.1　Model architecture specifications

Propose model Architecture		
Stage	Output Feature Size	
	Encoder	Decoder
1	$(B \times 64 \times 2 \times 128 \times 128)$ RSTSCABlock 数=1	$(B \times 256 \times 2 \times 32 \times 32)$
2	$(B \times 128 \times 2 \times 64 \times 64)$ RSTSCABlock 数=1	$(B \times 128 \times 2 \times 64 \times 64)$
3	$(B \times 256 \times 2 \times 32 \times 32)$ RSTSCABlock 数=3	$(B \times 64 \times 2 \times 128 \times 128)$
4	$(B \times 512 \times 2 \times 16 \times 16)$ RSTSCABlock 数=1	

4.6　モデルの学習

提案モデルでは、生成された画像 $\hat{I}_{1.5}$ と正解画像 $I_{1.5}$ の差の平均絶対誤差（Mean Absolute Error:MAE）という損失関数として使用する。損失関数は以下の式で定義される。

$$L = \|I_{1.5} - \hat{I}_{1.5}\| \tag{4.3}$$

最適化手法として $\beta_1 = 0.9, \beta_2 = 0.99$ の Adam を用いて学習される。学習率は最初に $1e-4$ 設定され、徐徐に $1e-6$ に減衰する。

第 5 章

実験

5.1 実験条件

5.1.1 データセット

モデルは Vimeo90K トレーニングセットで学習され、Vimeo90K 及び UCF101 の 2 つのデータセットで評価される。トレーニングデータセットを拡張するために、元の画像から 256×256 パッチをランダムにトリミングする。また、水平方向、垂直方向に反転し、フレームの順序を確率 0.5 で入れ替えることにより、事前確率によるバイアスを排除する。

Vimeo90K[20]：Vimeo90K トレーニングセットには 51,312 個のトリプレットが含まれており、各トリプレットは解像度 448x256 の 3 つの連続したビデオフレームで構成されている。Vimeo90K テストセットは、448 × 256 の 3782 個のトリプレットが含まれている。

UCF101[21]：様々な人間の行動を含むビデオが含まれている。テストセットには、256x256 の解像度が 379 個のトリプレットがある。

5.1.2 評価指標

評価指標には、ピーク信号対雑音比 (Peak signal-to-noise ratio: PSNR)、構造的類似性 (Structural Similarity: SSIM)[22] を用いる。

PSNR は、信号の最大パワーと元画像に対する評価画像のノイズの比率を示す指標であり、式 (5.1) で求められる。

$$PSNR = 10 \cdot \log_{10} \frac{MAX^2}{MSE} \tag{5.1}$$

ここで、MAX は画像の最大画素値であり、MSE (Mean Squared Error) は元画像と

評価画像から算出される。評価目安として、値が大きいほど良好である（40[dB] 以上で目視による見分けがつかない程度の再現度、30[dB] 以下で明らかな劣化が見られる再現度とされている）。

SSIM は、元画像と評価画像間での構造的類似度を示す指標であり、画像内の局所領域ごとに類似度を計算し、その平均値を出力値とする。SSIM は式 (5.2) で求められ、本論文ではウィンドサイズを (7 x 7) としている。

$$SSIM = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{5.2}$$

ここで、μ_x, μ_y は各領域の画素値の平均値、σ_x, σ_y は標準偏差、σ_{xy} は共分散、C_1, C_2 は発散防止定数である。評価目安として、値が 1 に近いほど良好であり、0.98 以上で目視による見分けがつかない程度の再現度、0.90 以下で明らかな劣化が見られる再現度とされている。

5.1.3　実験条件

本実験条件は表 5.1 に示す。

表 5.1　Experiment Conditions

Model	CAIN	提案法
Train dataset	Vimeo90K	
Test dataset	Vimeo90K and UCF101	
Batch size	32	16
Epoch	90	
Learning Rate	1e-4	
Loss Function	MAE	
Optimization Method	ADAM	

5.2　実験結果

Vimeo90K と UCF101 の全てのテストデータに対する PSNR、SSIM の平均値を表 5.2 に示す。提案法では、CAIN に対して Vimeo90K データセットの場合、PSNR で 0.601 [dB]、SSIM で 0.017、UCF101 データセットの場合、PSNR で 0.586 [dB]、SSIM で 0.015 上回る結果が得られた。また、提案モデルのパラメーター数は従来法の半分程度であることが確認できた。

次に、実験条件に対する CAIN（従来法）と提案法の生成結果例を図 5.1〜5.5 に示す。図 5.1〜5.5 において、(a) 及び (b) はネットワークの入力フレーム、(c) はフレーム間の動き領域を簡易的に示すための入力フレーム合成画像、(d) は正解画像である中間フレーム、(e) は提案法による生成結果、(f) は CAIN による生成結果である。

図 5.1 は入力フレーム間の動きが大きい画像に対する生成結果例を示している。CAIN では、緑の画像は大きく乱れているのに対し、提案法ではある程度を生成できていることがわかる。図 5.2 は入力フレーム間の動きが中程度の画像に対する生成結果例を示す。CAIN でぼやけな車が見えるが提案法はその車をうまく生成できていることがわかる。図 5.3 は入力フレーム間の動きが小さい画像に対する生成結果例を示している。図 5.3(c) に示すように、フレーム間の動きがかなり小さいため、CAIN と提案法は共に同じ程度の結果を生成した。図 5.4 は入力フレーム間の動きが中程度及び文字が含む画像に対する生成結果例を示す。CAIN では文字が大きく乱れている。提案法では、生成された文字がよく読める結果となっている。図 5.5 は入力フレーム間全体に動きのある画像に対する生成結果例を示している。CAIN では、女性の手が大きくぼやけとなっているのに対して提案法では、ある程度の外形を生成できることがわかる。

以上の結果から、提案法は、従来法に対し同党以上の結果を達成したと言える。

表 5.2　Quantitative comparisons on the Vimeo90K and UCK101 datasets

Model	Parameters (M)	Vimeo90K		UCF101	
		PSNR	SSIM	PSNR	SSIM
CAIN	42.8	34.831	0.961	34.452	0.954
提案法	21.3	**35.432**	**0.978**	**35.038**	**0.969**

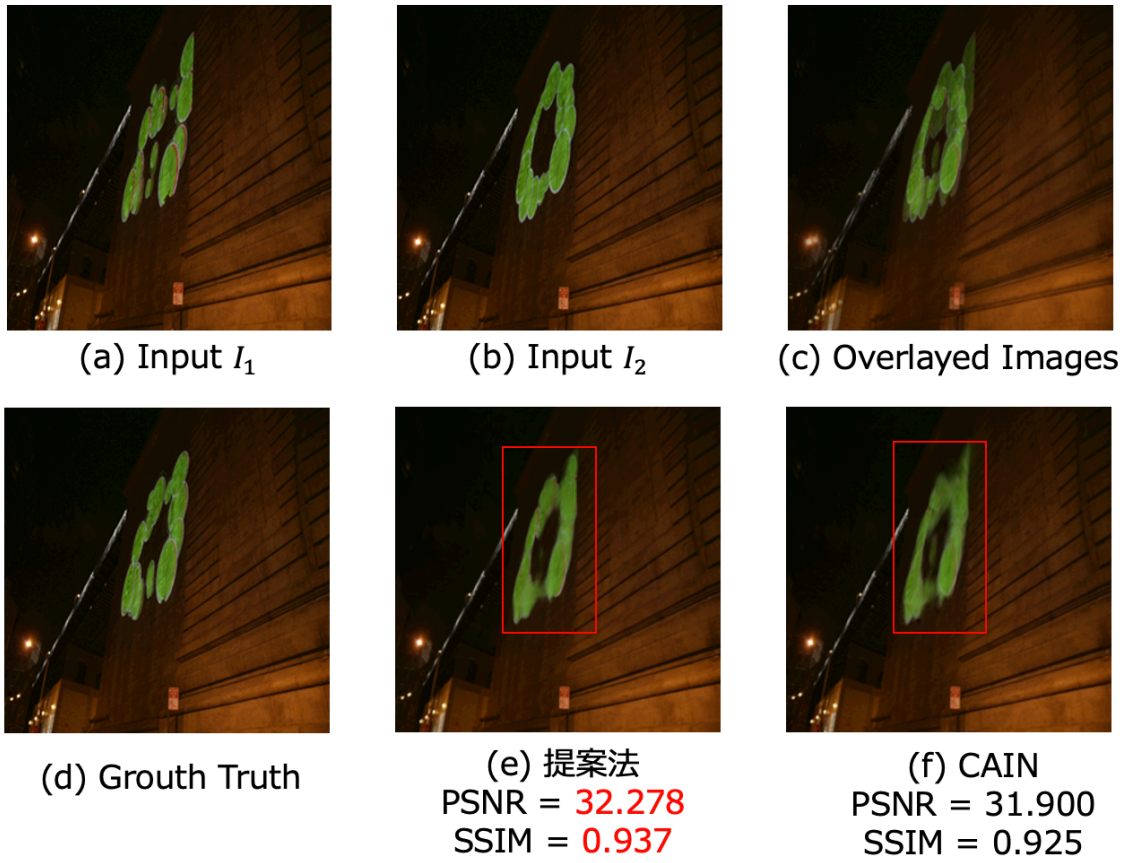

図 5.1　Generated Image Example 1

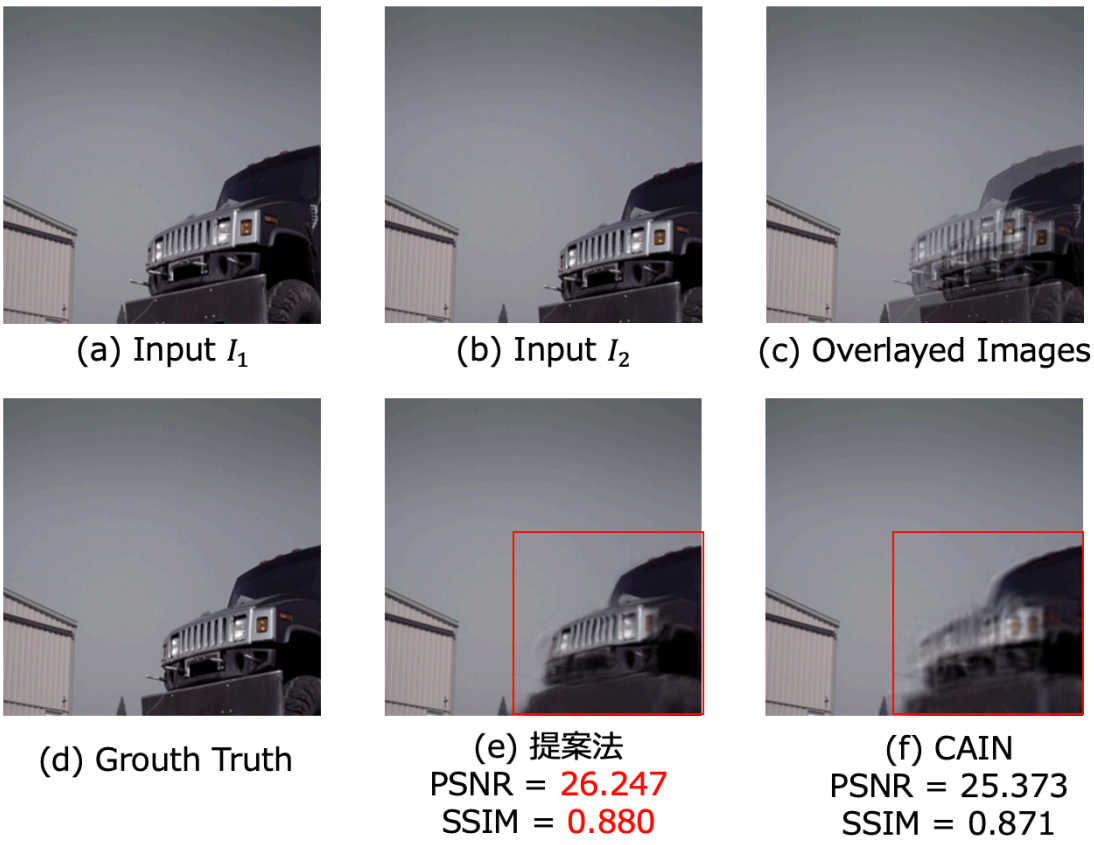

図 5.2　Generated Image Example 2

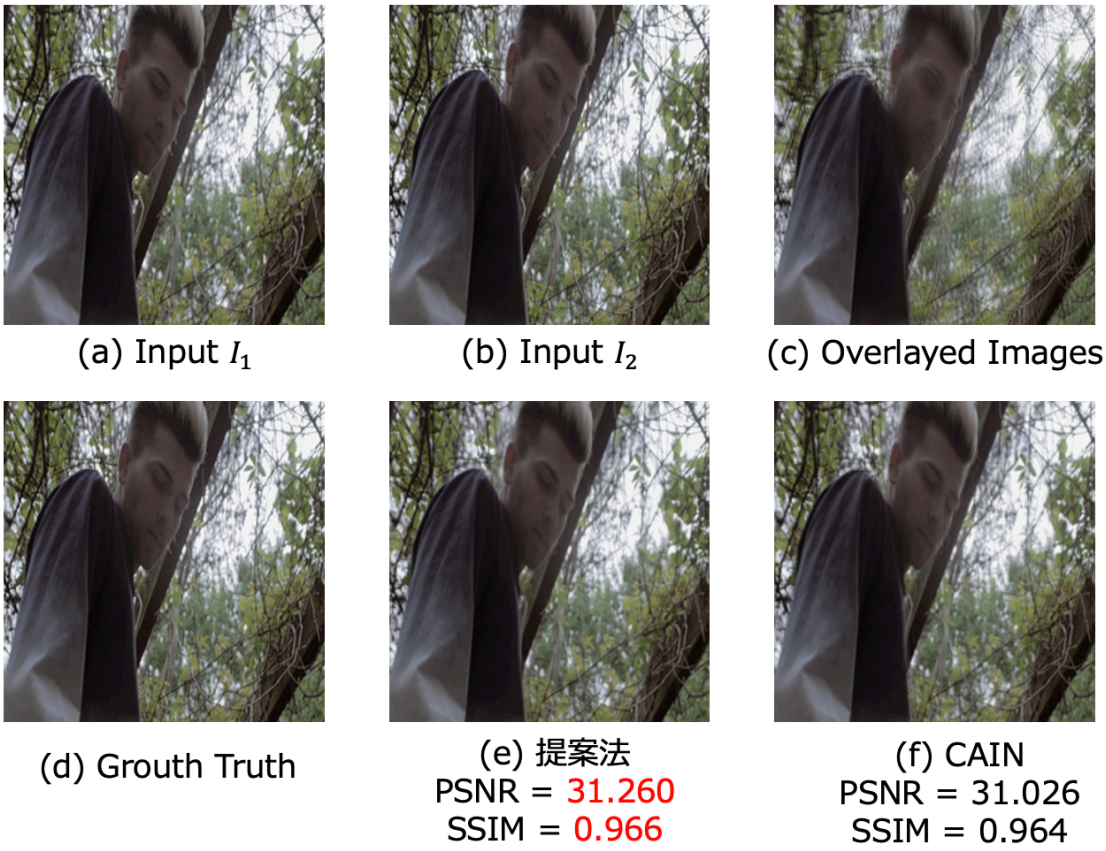

図 5.3　Generated Image Example 3

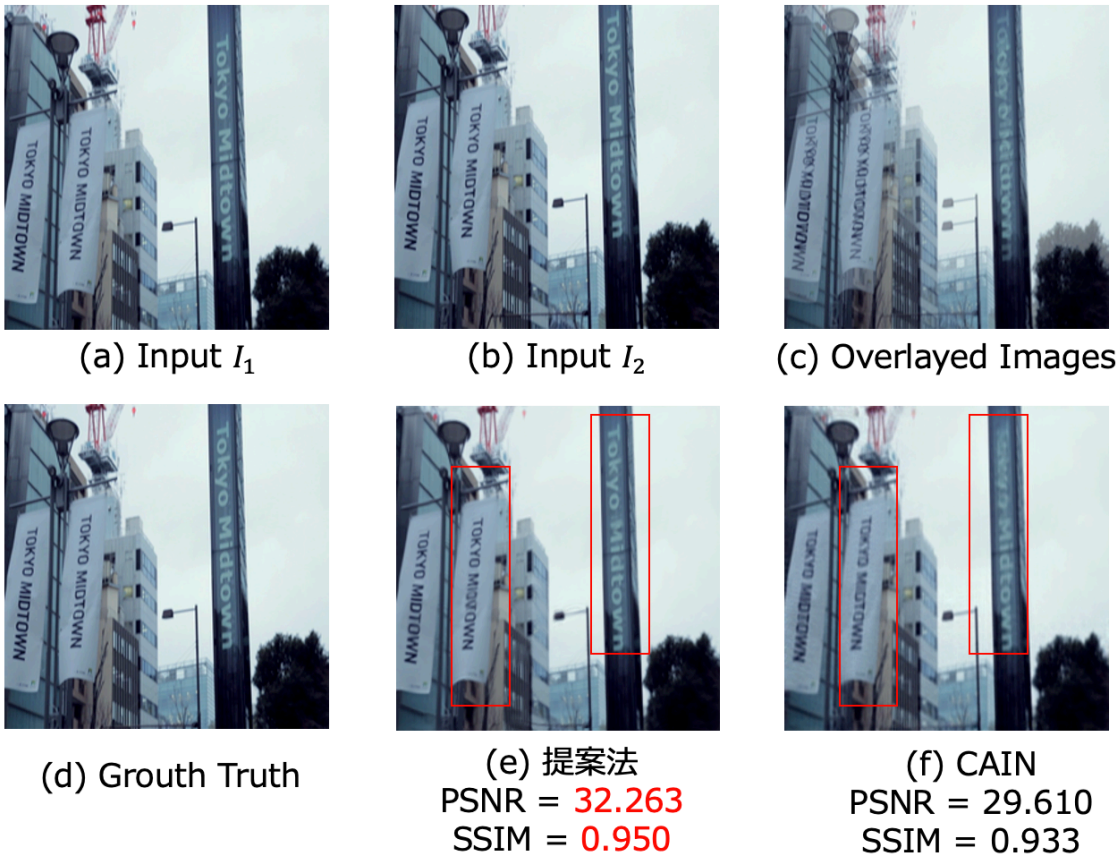

図 5.4　Generated Image Example 4

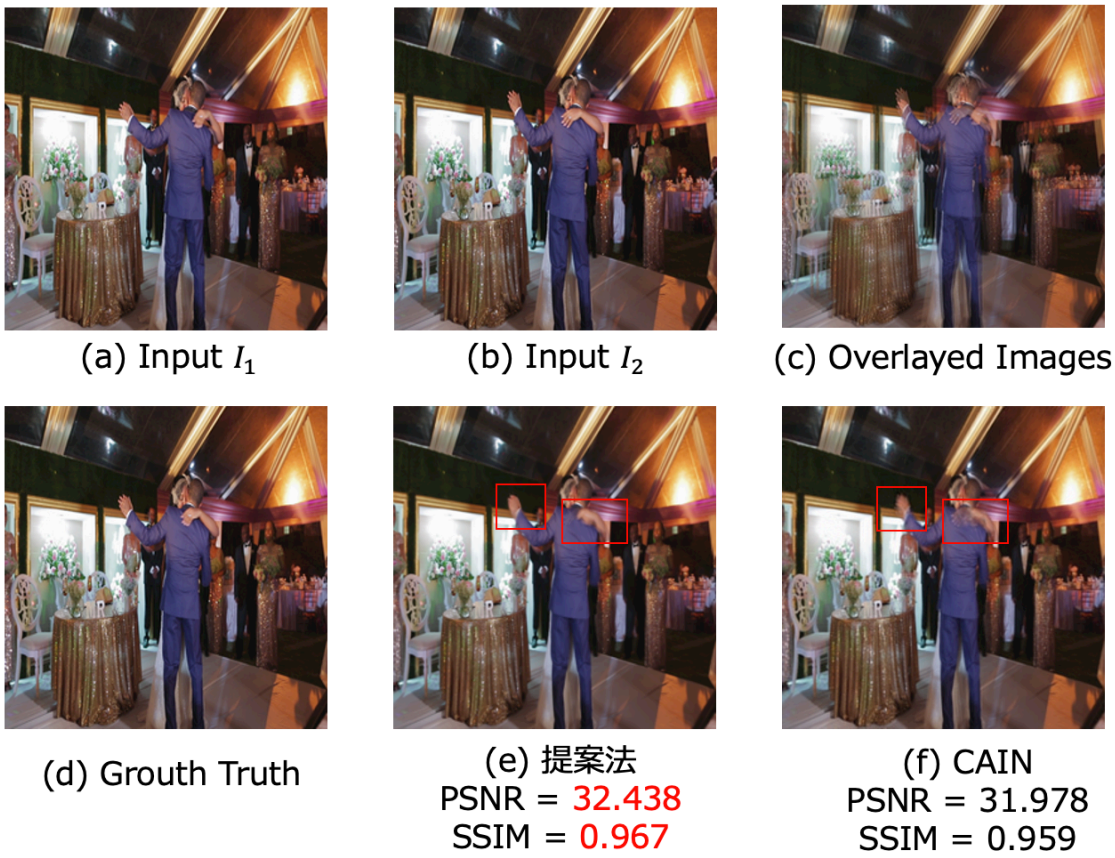

図 5.5　Generated Image Example 5

第 6 章

おわりに

6.1　まとめ

本論文では、生成画像の精度を向上することを目的に、空間と時間の両方で入力画像のピクセルの長期的な依存関係を学習するトランスフォーマーベースを用いる手法を提案した。第 1 章では、ビデオフレーム補間に関する最近の手法と問題点、本論文の研究目的について述べた。第 2 章では、従来法として CAIN を取り上げ、その手法の問題点について述べた。第 3 章では、本論文の提案法で用いられる RSTSCAGroup 及び、AdaCoF 等について説明した。第 4 章では、提案モデルの構造について説明した。第 5 章では、提案法の有用性を検討するために、同一の学習データを用いて CAIN と提案法の評価実験を行った。CAIN と提案法の生成画像を評価指標である PSNR と SSIM で比較したところ、様々な入力画像例において CAIN と同等かそれ以上の結果が得られ、提案法の有用性が確認できた。また、Vimeo90K と UCF101 の全てのテストデータに対する PSNR、SSIM の平均値を比較したところ、提案法では、CAIN に対して Vimeo90K データセットの場合、PSNR で 0.601 [dB]、SSIM で 0.017、UCF101 データセットの場合、PSNR で 0.586 [dB]、SSIM で 0.015 上回る結果が得られた。以上から、ネットワークにトランスフォーマーベースを用いることで大きいや中程度の動き等といった様々な例のフレーム補間の精度向上が実現できたと言える。

6.2　今後の課題

本論文では、従来法である CAIN に基づきトランスフォーマーベースを用いる手法を提案した。しかし、従来法と提案法では入力フレーム間の単一の中間フレーム補間にのみ対応しており、マルチフレーム補間に対応させることも今後の課題と言える。

謝辞

本研究を進めるにあたり、ご指導・ご助言を頂いた杉田泰則准教授に厚くお礼申し上げます。また、論文の審査において多くのご指示を頂きました、本学電気系岩橋政宏教授、圓道知博教授ならびに原川良介助教に厚く御礼申し上げます。最後に、本研究に関して多くの指摘をくださいました信号処理応用研究の皆様に深く感謝いたします。

参考文献

[1] Z. Liu, R. A. Yeh, X. Tang, Y. Liu, and A. Agarwala, "Video frame synthesis using deep voxel flow," *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 4473–4481, 2017.

[2] H. Jiang, D. Sun, V. Jampani, M.-H. Yang, E. Learned-Miller, and J. Kautz, "Super slomo: High quality estimation of multiple intermediate frames for video interpolation," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9000–9008, 2017.

[3] W. Bao, W.-S. Lai, C. Ma, X. Zhang, Z. Gao, and M.-H. Yang, "Depth-aware video frame interpolation," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3698–3707, 2019.

[4] J. Park, C. Lee, and C.-S. Kim, "Asymmetric bilateral motion estimation for video frame interpolation," *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 14519–14528, 2021.

[5] S. Niklaus, L. Mai, and F. Liu, "Video frame interpolation via adaptive convolution," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2270–2279, 2017.

[6] S. Niklaus, L. Mai, and F. Liu, "Video frame interpolation via adaptive separable convolution," *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 261–270, 2017.

[7] S. Niklaus, L. Mai, and O. Wang, "Revisiting adaptive convolutions for video frame interpolation," *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1098–1108, 2020.

[8] M. Choi, H. Kim, B. Han, N. Xu, and K. M. Lee, "Channel attention is all you need for video frame interpolation," *AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, pp. 10663–10671, 2020.

[9] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient

sub-pixel convolutional neural network," *IEEE Transactions on Image Processing*, pp. 1874–1883, 2016.

[10] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7132–7141, 2018.

[11] Z. Shi, X. Xu, X. Liu, J. Chen, and M.-H. Yang, "Video frame interpolation transformer," *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17461–17470, 2022.

[12] H. Lee, T. Kim, T.-y. Chung, D. Pak, Y. Ban, and S. Lee, "Adacof: Adaptive collaboration of flows for video frame interpolation," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5315–5324, 2019.

[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.

[14] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," *ICLR*, 2021.

[15] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9992–10002, 2021.

[16] W. Xing and K. Egiazarian, "Residual swin transformer channel attention network for image demosaicing," *2022 10th European Workshop on Visual Information Processing (EUVIP)*, pp. 1–6, 2022.

[17] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," *arXiv*, vol. abs/1606.08415, 2016.

[18] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv*, vol. abs/1607.06450, 2016.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2015.

[20] T. Xue, B. Chen, J. Wu, D. Wei, and W. T. Freeman, "Video enhancement with task-oriented flow," *International Journal of Computer Vision*, vol. 127, no. 8, pp. 1106–1125, 2019.

[21] K. Soomro, A. R. Zamir, and M. Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," *arXiv*, vol. abs/1212.0402, 2012.

[22] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

付録 A

A.1　提案法の特徴量のサイズに関する実験

A.1.1　実験設定

第 4 章の 4.3 提案モデルで深層特徴量抽出モジュールは、特徴量を固定なサイズを通る従来法と異なり、エンコーダーデコーダー アーキテクチャに変更するということを述べた。今回の実験は、提案法の特徴量を固定なサイズに設定し、結果比較を行う。

２つのモデルを作成し、図 A.1 に示す。図 A.1(a) に従来法である CAIN モデルを示す。２つの入力画像 $\mathbf{I}_1, \mathbf{I}_2 \in \mathbb{R}^{C\times H\times W}$ とし、チャネル次元に沿って連結し、ダウンシャフル層でサイズ $(128C \times \frac{H}{8} \times \frac{W}{8})$ に変更して、$(64C \times \frac{H}{8} \times \frac{W}{8})$ にチャネル数を減少した後に、５個の残差グループ (ResGroup) を通過し、特徴量 $\mathbf{F}$ が得られる。最終的な中間フレームは、$\mathbf{F}$ をアップシャッフルして中間フレーム $\mathbf{I}_{1.5} \in \mathbb{R}^{C\times H\times W}$ の元のサイズ を生成する。

図 A.1(b) は CAIN の ResGroup に注目して変更した。ResGroup の代わりに第 4 章の 4.1 で提案した RSTSCAGroup に変更した。5 個の RSTSCAGroup を用い、各 RSTSCAGroup は 3 個の RSTSCA Block から構成される。また、２つの入力画像 $\mathbf{I}_1, \mathbf{I}_2 \in \mathbb{R}^{C\times H\times W}$ とし、時間次元に沿って連結し、ダウンシャフル層で特徴量のサイズ $(64C \times T \times \frac{H}{8} \times \frac{W}{8})$ に変更し、5 個の RSTSCAGroup に通過した後に特徴量 $\mathbf{F}$ が得られる。最終的な中間フレームは、$\mathbf{F}$ を時間次元でアンバインドして、入力フレームの T 個の個別の 3 次元特徴量 $(64C \times \frac{H}{8} \times \frac{W}{8})$ を取得する。2D 畳み込み層を用いて T 個の特徴量を合成し、アップシャッフル層で中間フレーム $\mathbf{I}_{1.5} \in \mathbb{R}^{C\times H\times W}$ の元のサイズに戻す。

図 A.1(c) は図 A.1(b) の最後の段階に注目して変更する。得られた特徴量 $\mathbf{F}$ を第 4 章の 4.4 で説明したフレーム合成モジュールに入力して中間フレーム $\mathbf{I}_{1.5} \in \mathbb{R}^{C\times H\times W}$ を合成する。ただし、フレーム合成モジュールの転置畳み込み層は 8 倍で特徴量 $\mathbf{F}$ の拡大を行った。

A.1.2 実験結果

表 A.1 に提案法を含み、４つのモデルの詳細情報の比較を示す。Vimeo90K トレーニングセットで学習して Vimeo90K テストデータで評価する 4 つのモデルの比較結果は図 A.2 に示す。図 A.2 を見ると、 作成した２つのモデルの結果が大きく下がることが分かった。

モデル１の結果が下がる理由として、時間次元に検討する時、単純な畳み込み層を用いて T 個の特徴量を合成するのが良くないからと考える。中間フレーム合成モジュールを入れるモデル２では、モデル１より結果が上がることがわかるが、$(\frac{H}{8} \times \frac{W}{8})$ の特徴量 $\mathbf{F}$ を転置畳み込み層により８倍で入力画像の元のサイズに戻すと、正しくないピクセルを生成してしまうことがあると考える。そのため、ンコーダーデコーダー アーキテクチャを利用してデコーダーで毎ステージは、２倍で特徴量のサイズを拡大する。そうすると、中間フレーム合成モジュールに入る前に $(\frac{H}{2} \times \frac{W}{2})$ のサイズになり、転置畳み込み層は２倍だけサイズを拡大する。また、ンコーダーデコーダーで毎回縮小すると、入力画像ピクセルの長期的な関連度を計算してトランスフォーマーの能力が向上できる。

表 A.1 Four Models Comparison

Model	Input Dimension	Parameters (M)
CAIN	$(C \times H \times W)$	42.8
Model 1	$(C \times T \times H \times W)$	25.5
Model 2	$(C \times T \times H \times W)$	25.8
提案法	$(C \times T \times H \times W)$	21.3

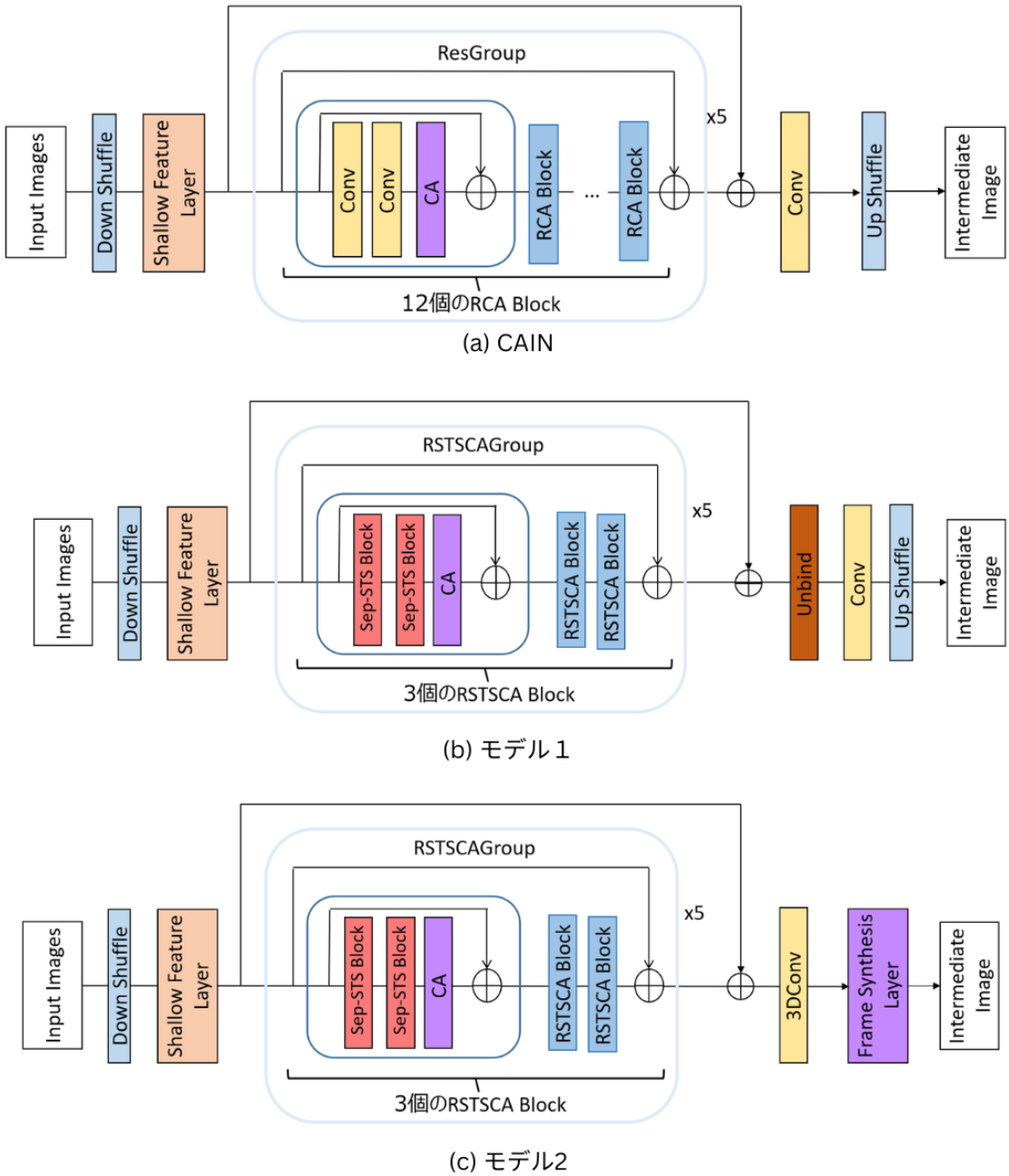

図 A.1　CAIN and two new models

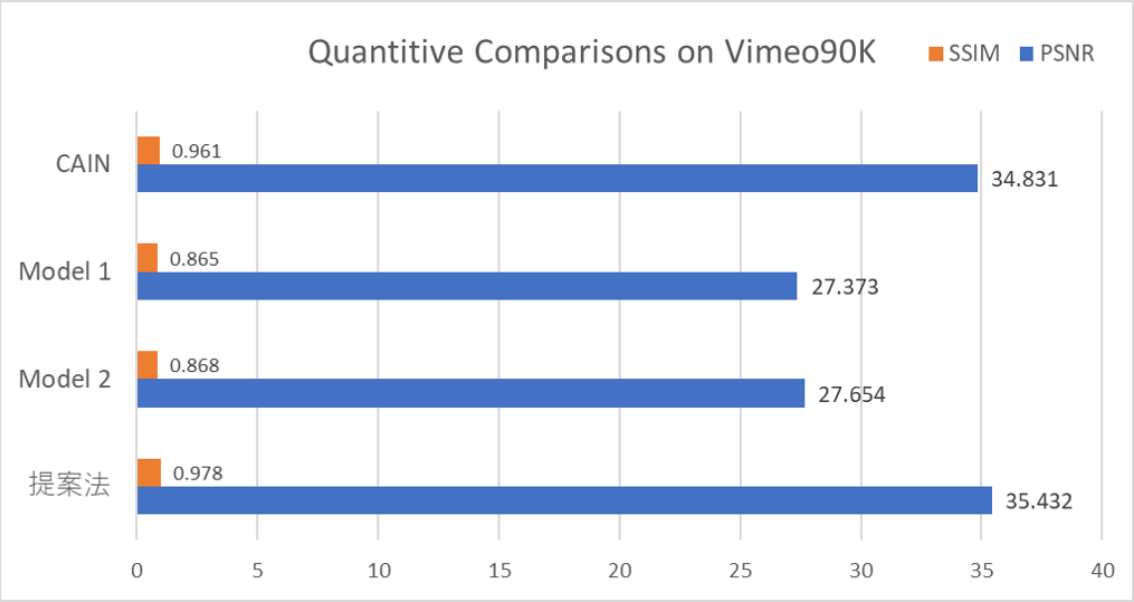

図 A.2　Quantitive Comparison of four models on Vimeo90K

A.2 提案法の RSTSCABlock 数の変更に関する実験

A.2.1 実験設定

第 4 章の 4.5 モデルの設定で RSTSCABlock 数は各ステージに対して 1 – 1 – 3 – 1 に設定した。今回の実験は、RSTSCABlock 数は各ステージに対して同じ 2 – 2 – 2 – 2 に設定し、結果比較を行う。

A.2.2 実験結果

Vimeo90K と UCF101 の全てのテストデータに対する PSNR、SSIM の平均値を表 A.2 に示す。表 A.2 を見ると、提案法 （RSTSCABlock 数=2-2-2-2）では、提案法（RSTSCABlock 数=1-1-3-1）に対して Vimeo90K データセットの場合、PSNR で 0.065 [dB]・UCF101 データセットの場合、PSNR で 0.053 [dB] 上回るが、Vimeo90 及び UCF101 の SSIM はそれぞれ同じ程度及び 0.001 下回ることが確認できた。

次に、実験条件に対する 2 つのモデルの生成結果例を図 A.3〜A.7 に示す。図において、(a) 及び (b) はネットワーク入力フレーム、(c) はフレーム間の動き領域を簡易的に示すための入力フレーム合成画像、(d) は正解画像である中間フレーム、(e) は提案法 （RSTSCABlock 数=1-1-3-1）による生成結果、(f) は（RSTSCABlock 数=2-2-2-2）による生成結果である。提案法 （RSTSCABlock 数=2-2-2-2）の生成画像は数値的に結果のほうが良いことがわかるが、ある場合提案法 （RSTSCABlock 数=1-1-3-1）のほうが良い。例として、図 A.6 では、提案法 （RSTSCABlock 数=2-2-2-2）の指標評価の結果のほうが高いが、提案法 （RSTSCABlock 数=1-1-3-1）で生成された画像の文字のほうがよくみられることが分かった。図 A.7 では、提案法 （RSTSCABlock 数=1-1-3-1）は提案法（RSTSCABlock 数=2-2-2-2）より生成画像及び指標評価の結果が良いことも見えた。また、モデルのパラメータ数に関して提案法 （RSTSCABlock 数=1-1-3-1）のほうが少ないため、提案法（RSTSCABlock 数=1-1-3-1）が実用的な効果があると言える。

表 A.2 Comparisons of number of RSTSCABlock on the Vimeo90K and UCK101 datasets

Model	Parameters (M)	Vimeo90K		UCF101	
		PSNR	SSIM	PSNR	SSIM
提案法（RSTSCABlock 数=1-1-3-1）	21.3	35.432	**0.978**	35.038	**0.969**
提案法 （RSTSCABlock 数=2-2-2-2）	28.3	**35.497**	**0.978**	**35.091**	0.968

図 A.3 Comparison between Number of RSTSCABlock Example 1

図 A.4　Comparison between Number of RSTSCABlock Example 2

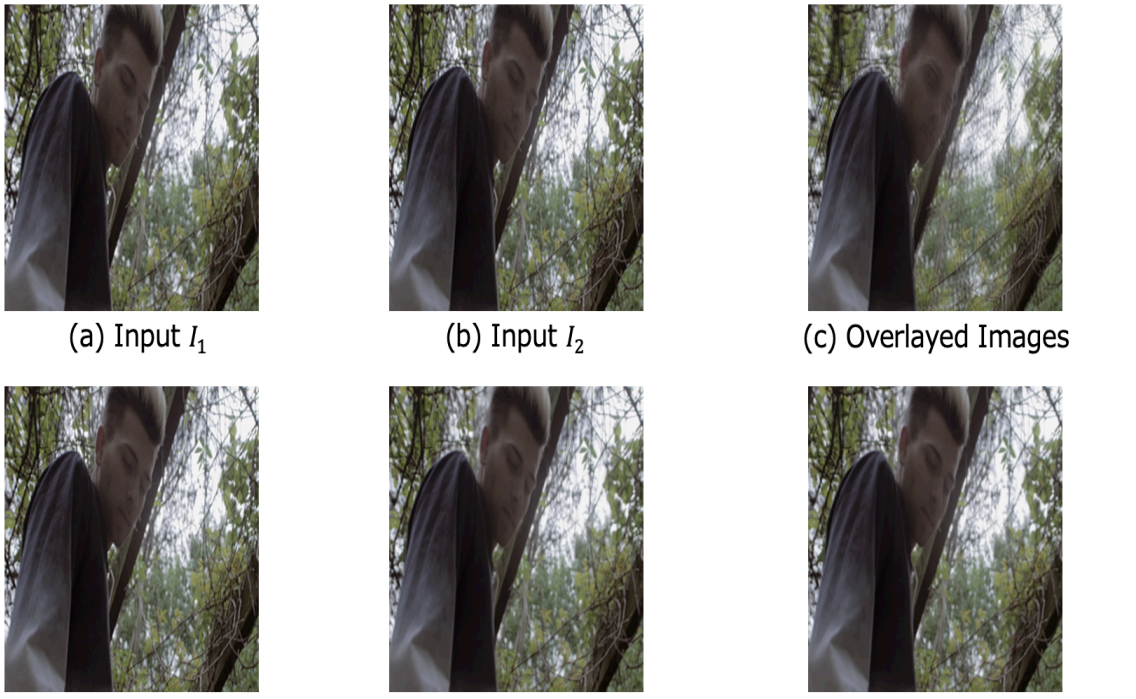

図 A.5　Comparison between Number of RSTSCABlock Example 3

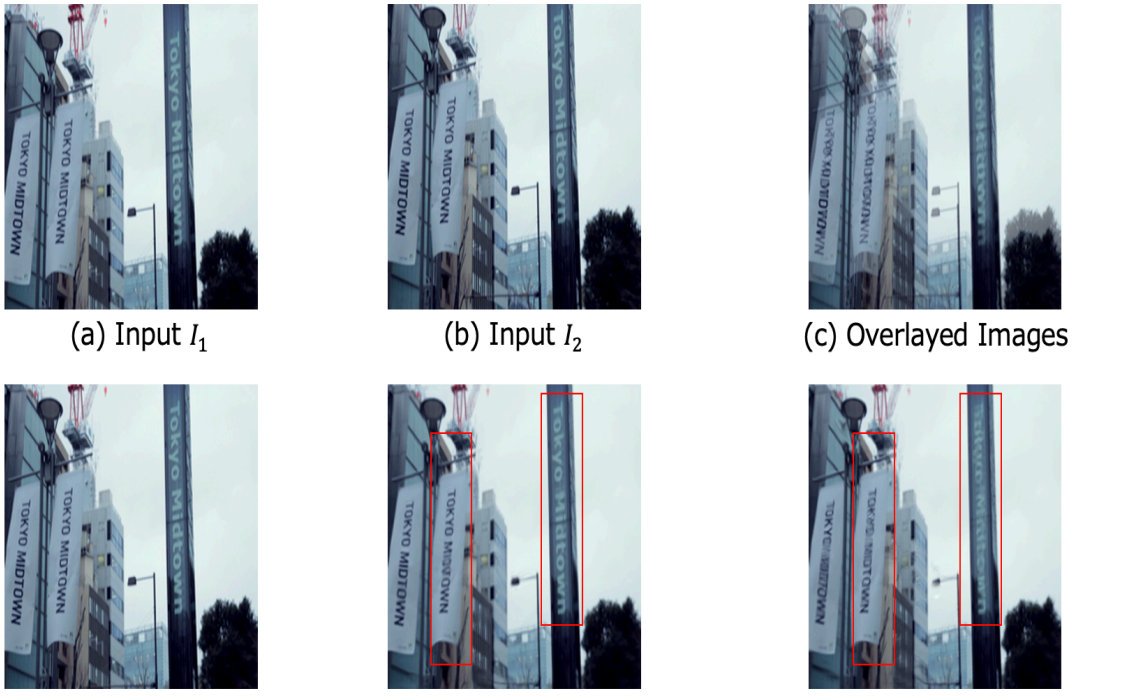

図 A.6　Comparison between Number of RSTSCABlock Example 4

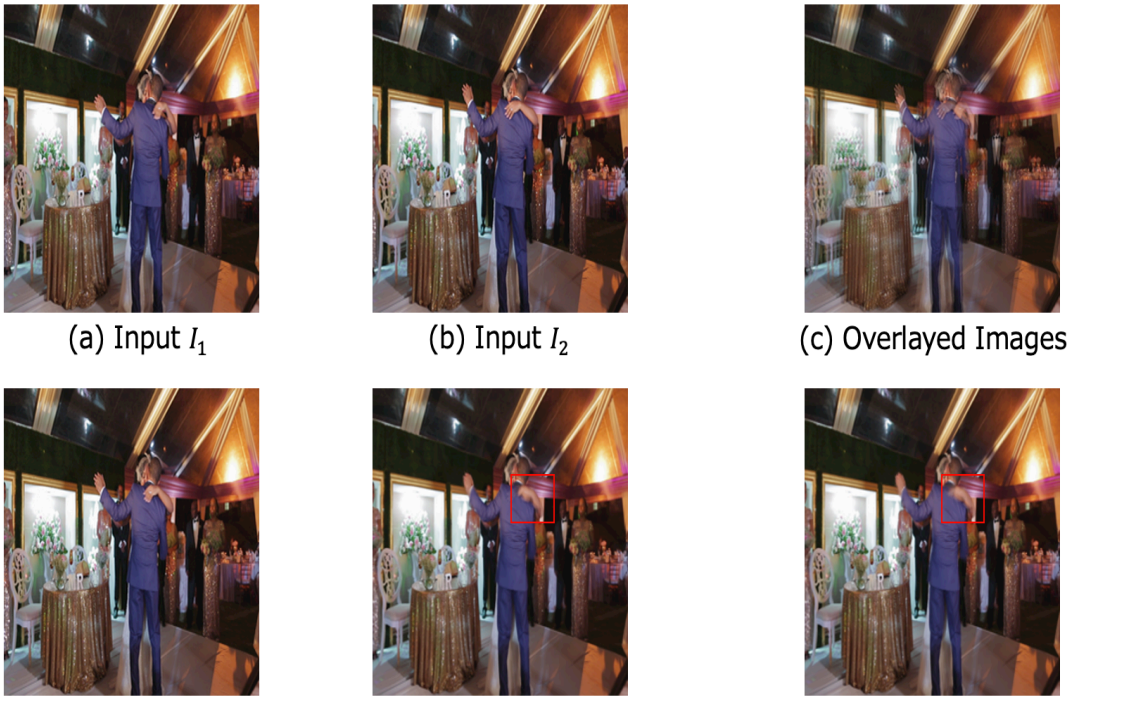

図 A.7　Comparison between Number of RSTSCABlock Example 5

A.3 提案法のフレーム合成モジュールに関する実験

A.3.1 実験設定

第 4 章の 4.4 フレーム合成モジュールでフレーム合成モジュールアッのプサンプリング層はバイリニア補間ではなく、転置畳み込み層を使用した。今回の実験では、バイリニア補間及び転置畳み込み層という 2 つの生成結果の比較を行う。

実験条件は第 5 章の 5.1.3 実験条件と同様に設定し、表 5.1 に示す。

A.3.2 実験結果

Vimeo90K と UCF101 の全てのテストデータに対する PSNR、SSIM の平均値を表 A.3 に示す。モデルのパラメータ数はあまり変わらないが、提案法では、CAIN に対して Vimeo90K データセットの場合、PSNR で 0.212 [dB]、SSIM で 0.004、UCF101 データセットの場合、PSNR で 0.044 [dB]、SSIM で 0.001 上回る結果が得られた。

次に、実験条件に対する CAIN（従来法）と提案法の生成結果例を図 5.1〜5.5 に示す。図において、(a) 及び (b) はネットワーク入力フレーム、(c) はフレーム間の動き領域を簡易的に示すための入力フレーム合成画像、(d) は正解画像である中間フレーム、(e) は提案法（転置畳み込み層）による生成結果、(f) は提案法（バイリニア補間）による生成結果である。

表 A.3 Upsampling layer comparisons on the Vimeo90K and UCK101 datasets

Model	Parameters (M)	Vimeo90K		UCF101	
		PSNR	SSIM	PSNR	SSIM
提案法（転置畳み込み層）	21.3	**35.432**	**0.978**	**35.038**	**0.969**
提案法 （バイリニア補間）	21.2	35.220	0.974	34.994	0.968

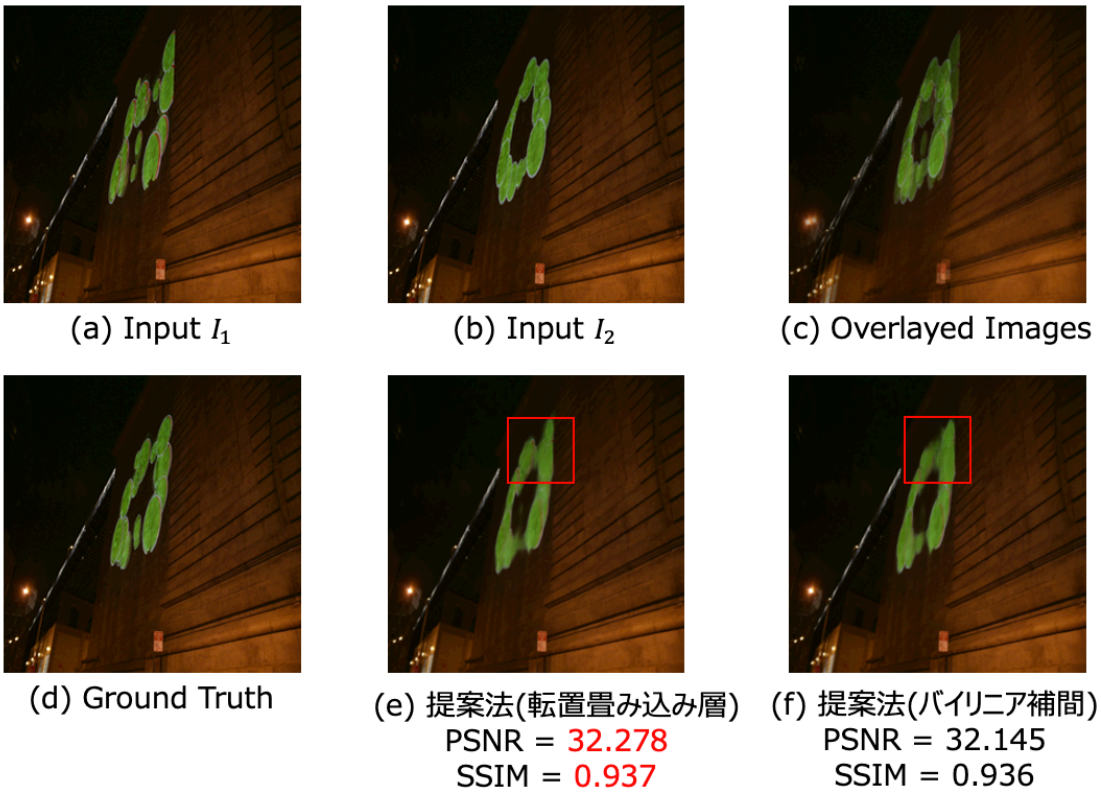

図 A.8　Comparison between Bilinear and Transconvolution Example 1

図 A.9　Comparison between Bilinear and Transconvolution Example 2

図 A.10　Comparison between Bilinear and Transconvolution Example 3

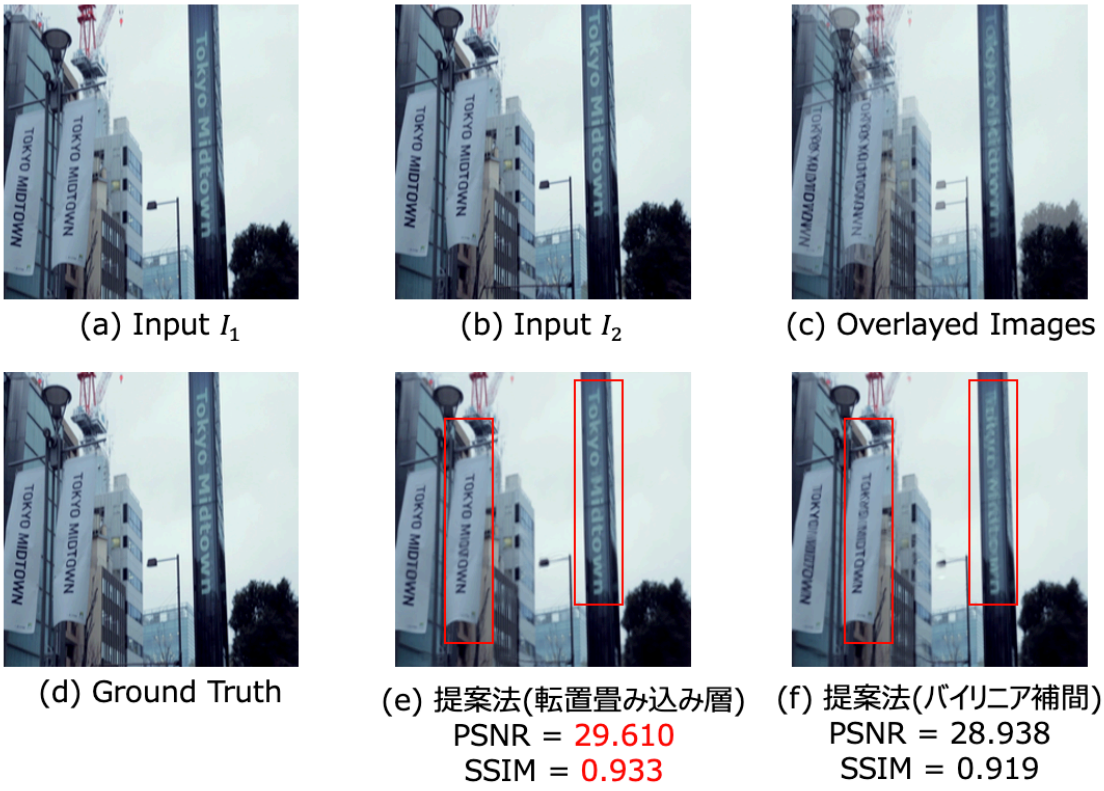

図 A.11　Comparison between Bilinear and Transconvolution Example 4

図 A.12　Comparison between Bilinear and Transconvolution Example 5