2022年度　修士論文

Cycle GANを用いたSemantic Segmentationの教師無し学習

指導教員　　石川 博 教授

研究指導名　コンピュータービジョン研究

早稲田大学大学院 基幹理工学研究科 情報理工・情報通信専攻
5121F036-9　川本 剛士

2023年 1月 23日　提出

論文概要

コンピュータービジョン研究において重要な画像認識問題の一つに，Semantic Segmentation がある．このタスクは画像に写っている対象物体が画像のどこに写っているかをピクセルレベルで識別する問題である．2012 年以降，Convolutional Neural Network の発展により画像認識の精度が飛躍的に向上している．しかし，それらの手法は基本的に学習用画像と，それに対する正解ラベル画像がセットとなっているデータセットを用いている事がほとんどである．データセットとして用いられる数千・数万枚の画像全てにピクセルレベルの正解ラベルを付与することは多大な時間と労力を要するため，正解ラベルラベルを不要とする手法が重要となる．本研究では，画像生成モデルである Cycle GAN [20] を用いて完全教師無し学習を行うことを目的としている．Cycle GAN に入力する画像に対し，同時に入力画像の Affine 変換を施したものを入力することで追加の損失関数を定義して学習を行う手法を提案した．Cycle GAN は入力画像と正解ラベル画像がペアである必要がないため，既存の正解ラベル画像を流用することが可能である．また，Cycle GAN は本来画像変換モデルとしての役割があるが，Semantic Segmentation として評価を行うことで，精度を向上する手法を提案した．

謝辞

本研究を進めるにあたり、指導教員の石川博教授には日々の研究や論文執筆に関して手厚い助言やご指摘を頂き、指導いただきました。また、日々の助言をいただきました石川研究室の先輩方にも大変お世話になりました。ここに感謝の意を表します。

2023 年 1 月 23 日

川本 剛士

目次

図目次

表目次

第 1 章

はじめに

1.1 背景

近年，画像認識や画像生成といった分野において機械学習 (AI) を用いることが一般的となっている．なかでもセマンティックセグメンテーションは広い分野で用いられている．例えば，自動運転技術や医療用画像識別，工業用品検査等である．画像認識技術の発展には ImageNet [5] などの大規模な画像データセットの発展や，GPU を用いた並列計算が可能になったことなど，様々な要因がある．それ以外にも，ニューラルネットワークに対して畳み込み層を導入した Convolutional Neural Network(CNN) [12, 17, 8] の導入も画像認識技術の発展に寄与している．近年 CNN を用いた大規模なネットワークにより人間に並ぶほどの性能を持つ画像認識技術が次々と発表されている．

セマンティックセグメンテーションとは，画像内に写っている物体をピクセル単位で認識して領域分割する技術である．機械学習を用いた認識モデルを生成する際には膨大な量の学習用データとその教師データが必要とされる．学習用データを集めることは容易であっても，その教師データを用意するためには多くの場合手作業でラベル付けを行う必要があり，多大な労力がかかってしまう．

本研究が対象とする画像群は車載カメラ画像であり，ドライブレコーダーの設置数の増加によってこれらの画像は大量に得ることができるようになっている．また，自動運転技術の進歩によってこれらの車載カメラ画像に対する認識技術の発展は不可欠となっている．これらの車載カメラ画像や動画像のデータは大量に集まっていてもそれに対する教師ラベル画像を用意することが追いついていないため，教師ラベル画像を必要としない手法が必要となる．

1.2 目的

これらの背景を踏まえて本研究では，新規のデータに対して教師データが付与されていなくても学習を行うことができる手法を確立することを目的とする．現状公開されているデータセットの教師ラベル画像を流用することで，教師ラベルが付与されていない画像群に対する Semantic Segmentation を行う事を目的とする．

1.3 本稿の構成

本稿の構成を述べる．まず第 1 章では本研究の背景と目的を述べる．次に第 2 章では本研究と関連のある研究について述べる．続いて第 3 章では提案手法に関して述べる．第 4 章では第 3 章に述べた提案手法に沿って実験を行った結果を述べる．最後に第 5 章では結論を述べる．

第 2 章

関連研究

2.1 Convolutional Neural Network

Convolutional Neural Network(CNN) は近年，画像認識技術において幅広く利用されている．ここでは，CNN を構築する上で用いられている技術をまとめる．

2.1.1 パーセプトロン

Neural Network を構成する最小単位をパーセプトロンと呼ぶ．これは生物の脳のニューロンと呼ばれる神経細胞をモデル化したもので，複数の入力から一つの出力を行う．それぞれの入力には重みを持っており，その重みは出力に対する入力の重要性を表す．入力を $\boldsymbol{x} = (x_1, x_2, \ldots, x_N)$, 重みを $\boldsymbol{w} = (w_1, w_2, \ldots, w_N)^{\mathsf{T}}$, バイアスを b とする．このとき，入力に対して重み付き和を求めたものにバイアスを加え，関数 f で変換したものがパーセプトロンの出力となる．図 2.1 にパーセプトロンの構造を示す．

$$a = b + \sum_{i=1}^{N} w_i \cdot x_i, \tag{2.1}$$

$$y = f(a). \tag{2.2}$$

式 (2.2) における関数 f は活性化関数と呼ばれるもので，多くの場合非線形関数が用いられる．

2.1.2 Neural Network

Neural Network とは，多数のパーセプトロンを階層化して連結したものである．図 2.2 に Neural Network の構造を示す．パーセプトロンは線型分離可能な問題しか解くことができないため，パーセプトロンを連結することにより非線形分離が必要な問題を解くことを

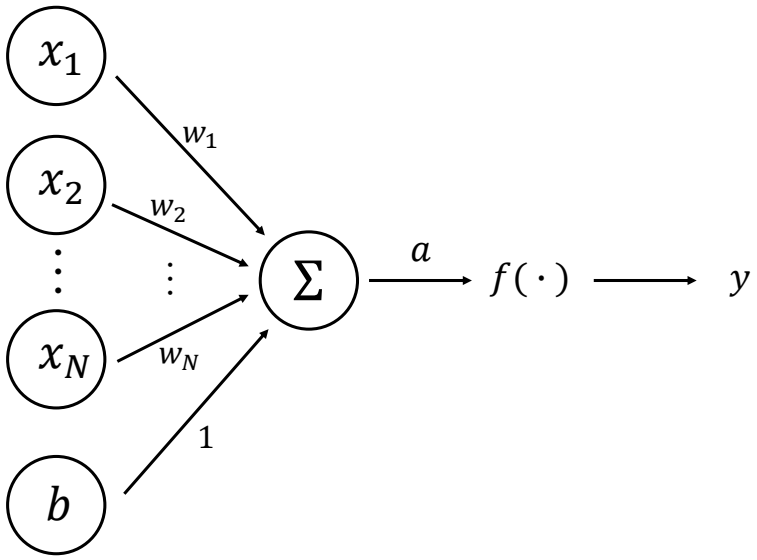

図 2.1: パーセプトロンの構造

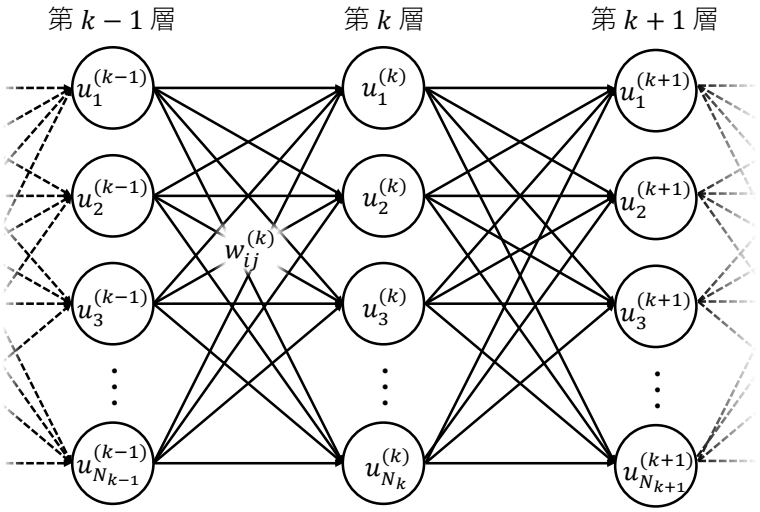

図 2.2: Neural Network の構造．$u_i^{(k)}$ は k 層目の i 番目のノードを表し，$w_{ij}^{(k)}$ は $k-1$ 層目の i 番目のノードから k 層目の j 番目のノードへの重みを表す．

可能にした．そのため，パーセプトロンの数を増やす，すなわち層を増やせば増やすほど複雑な非線型分離が可能となる．しかし，非線形分離が可能となったからといって，簡単に重みやバイアスといったパラメータを求めることができるわけではない．目的の非線形関数に近い関数をネットワークで近似できるようにパラメータを調節する必要がある．Neural Network ではこのパラメータの調節は学習することによって自動的に行っている．

2.1.3 Convolutional Neural Network

前小節で述べた Neural Network では，隣接するニューロン同士が全て結合している全結合層と呼ばれる層を用いて計算を行う．しかし，全結合層には入力データを 1 次元に変形してから入力する必要があるため，データの形状を無視してしまうという欠点がある．例えば，入力データが画像の場合には縦，横，深さ (チャンネル) といった 3 次元の形状のデータとなるため，全結合層を用いると空間的な位置情報を失ってしまう．そこで，画像のような空間的な関係性を保持したまま計算を行えるように提案されたのが畳み込み層である．畳み込み層を用いた Neural Network を Convolutional Neural Network(CNN) と呼ぶ．

畳み込み層 (Convolutional Layer) では，一定のサイズのフィルタ (カーネル) をスライドさせる．CNN では，フィルタの各要素が Neural Network での重みに相当するため，学習によってフィルタの各要素を更新する．フィルタをスライドさせる幅のことをストライドと呼ぶ．また，通常の畳み込み演算を行うとデータのサイズが小さくなってしまう．これを防ぐために入力データの周囲を固定値で埋めることがある．これをパディングと呼ぶ．多くの場合，周囲を 0 でうめるゼロパディングを行う．図 2.3 に畳み込み演算の例を示す．また，畳み込み演算によって変換された各層の出力結果を特徴マップと呼ぶ．

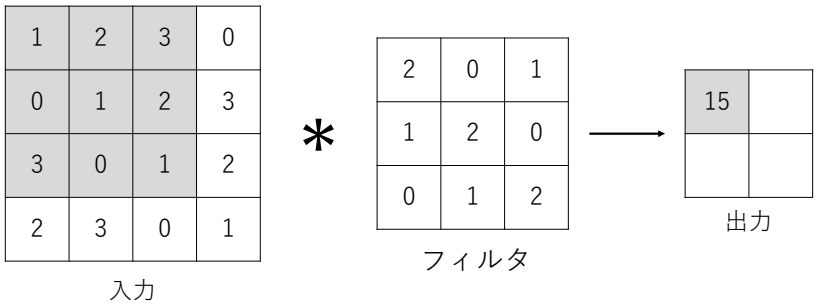

図 2.3: フィルタサイズ 3×3 ，ストライド 1，パディング無しの場合における畳み込み演算の例

チャンネル方向にもデータを持つ特徴マップに対して畳み込み演算を行う場合には，フィルタをチャンネル数だけ用意し，チャンネルごとに畳み込み演算を行った結果を加算して出力を得る．さらに，フィルタを複数用意し，特徴マップに対して畳み込み演算を複数回行うことでチャンネル方向に深い特徴マップを得ることができる．図 2.4 にチャンネル方向にも特徴量を持つ場合の畳み込み演算の概要図を示す．

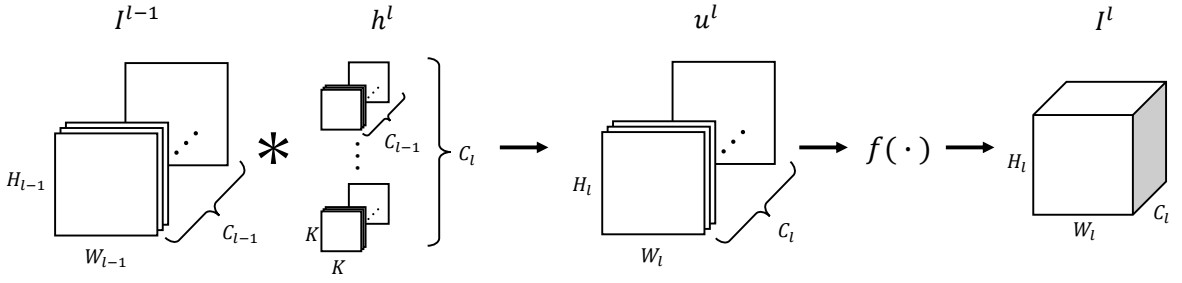

図 2.4: チャンネル方向にも特徴量を持つ場合の畳み込み演算の概要図

第 l 層における畳み込み演算の計算式を示す. 第 l 層は第 $l-1$ 層から $H_{l-1}\times W_{l-1}\times C_{l-1}$ の特徴マップを受け取り，$K\times K\times C_{l-1}$ のサイズを持つフィルタを C_l 種類用意することで畳み込みを行うことにより $H_l\times W_l\times C_l$ のサイズを持つ特徴マップを出力する．第 $l-1$ 層，第 l 層の特徴マップをそれぞれ I^{l-1}, I^l とし，m 番目のチャンネルを計算するフィルタを h^l_m とする．その時，第 l 層における m 番目の特徴マップの (x,y) の要素 I^l_{xym} は

$$u^l_{xym} = \sum_{c=0}^{C_{l-1}} \sum_{i=0}^{K-1} \sum_{j=0}^{K-1} I^{l-1}_{(x+i)(y+j)c} \cdot h_{ijcm} + b^l_m \tag{2.3}$$

$$I^l_{yxm} = f(u^l_{xym}) \tag{2.4}$$

と表される．ただし，b^l_m はバイアス，関数 f は活性化関数を表す.

畳み込み層と組み合わせて用いられるのがプーリング層である．画像認識においては主に Max プーリングが用いられ，フィルタ内の最大値のみを出力するといった演算を行う．図 2.5 にプーリングの例を示す．これにより，特徴マップの微小な平行移動に対しての出力に大きな変化が発生しにくくなるため，入力画像の平行移動に対する頑健性を得ることができる．また，入力画像の解像度を下げることができるため，畳み込み層における計算量を削減することもできる.

CNN では全結合層の代わりに畳み込み層とプーリング層を用いることによってネットワークを構成する．CNN に対する出力が回帰問題や分類問題の場合，最終的な出力の層には全結合層が用いられる．また，パラメータの更新を行う手法は Neural Network と同様に損失関数を用いて勾配降下法で学習を行う.

2012 年に Krizhevsky らによって提案された，考案者の名前を用いて Alex Net [12] と呼ばれるモデルが ImageNet Large Scale Visual Recognition Competition (ILSVRC) において圧倒的な精度でトップとなった．このモデルの提案によって画像認識における CNN

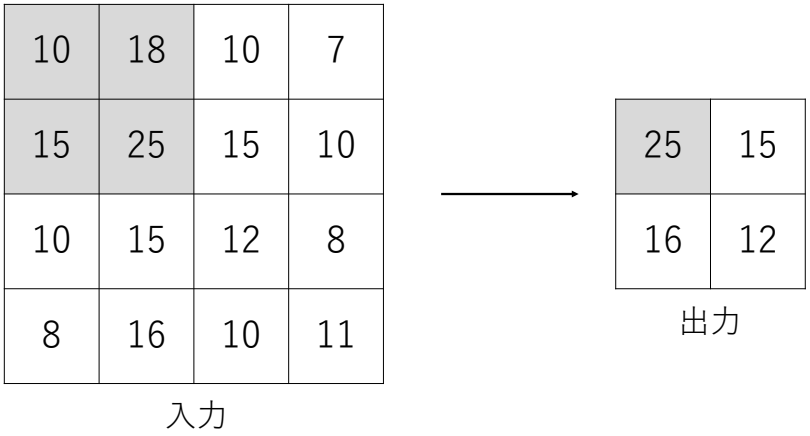

図 2.5: フィルタサイズ 2×2，ストライド 2，の場合におけるプーリングの例

の有効性が認知され，それ以降より深い層の CNN の構造を持つモデルの研究が進んだ．代表的なものとしては，Simonyan らの VGG [17] や，Zeng らの GoogLeNet [19]，He らの ResNet [8] がある．これらのモデルは現在においてもベースラインとして用いられることも多い．

2.2 活性化関数

活性化関数とは，Neural Network で用いられる非線形関数である．Neural Network は各層が線形変換の役割を持っており，多層に重ねただけでは一つの線形関数に置き換えることができてしまう．そこで各層の間に非線形関数である活性化関数を適用することで Neural Network 全体を非線形な変換になり，表現力の高い関数が生成可能となる．

2.2.1 Sigmoid 関数

Sigmoid 関数は入力を x，定数 a とすると，以下のように表される．

$$f(x) = \frac{1}{1 + e^{-ax}} \tag{2.5}$$

$(-\infty, +\infty)$ の定義域に対し，出力は $(0, 1)$ の範囲となる．Neural Network で一般的に用いられる Sigmoid 関数というと $a = 1$ とした標準シグモイド関数を用いる．

2.2.2 双曲線正接関数

双曲線正接関数は Hyperbolic Tangent 関数や Tanh 関数とも呼ばれる関数である．双曲線正接関数は入力を x とすると，以下のように表される．

$$f(x) = \tanh x \tag{2.6}$$

$$= \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2.7}$$

$(-\infty, +\infty)$ の定義域に対し，出力は $(-1, 1)$ の範囲となる．

2.2.3 ReLU 関数

ReLU 関数 (Rectified Linear Unit 関数) [6] は Glorot らによって有効性が発表された関数である．ReLU 関数は入力が 0 を超えていればそのまま出力し，0 を超えていなければ 0 を出力する関数であり，入力を x とすると，以下のように表される．

$$f(x) = \begin{cases} x & (x > 0) \\ 0 & (x \leq 0) \end{cases} \tag{2.8}$$

$(-\infty, +\infty)$ の定義域に対し，出力は $[0, +\infty)$ の範囲となる．

2.2.4 Leaky ReLU 関数

Leaky ReLU 関数 (Leaky Rectified Linear Unit 関数) [13] は Maas らによって ReLU 関数を発展させる形で発表された関数である．Leaky ReLU 関数は入力が 0 を超えていればそのまま出力し，0 を超えていなければ入力を a 倍した数値を出力する関数であり，入力を x とすると，以下のように表される．ここで事前に決定される定数 a は一般に 0.01 が用いられる．

$$f(x) = \begin{cases} x & (x > 0) \\ ax & (x \leq 0) \end{cases} \tag{2.9}$$

$(-\infty, +\infty)$ の定義域に対し，出力は $(-\infty, +\infty)$ の範囲となる．

2.3 誤差関数

機械学習における誤差関数とは，モデルからの予測値と正解値の誤差を測るための関数である．Semantic Segmentation では，出力画像と正解画像との画像間距離を計算する関数を誤差関数として用いる．本研究では以下の L1 誤差，平方二乗誤差を用いた．

2.3.1 L1 誤差

L1 誤差は，画像間の L1 ノルムによって計算される．高さ H ，幅 W ，チャンネル数 C を持つ 2 つの画像 I_1，I_2 に対して，L1 誤差 L_{mae} は

$$L_{mae} = ||I_1 - I_2||_1 \tag{2.10}$$

$$= \frac{1}{H \times W \times C} \sum_{h=0}^{H-1} \sum_{w=0}^{W-1} \sum_{c=0}^{C-1} |I_1 - I_2| \tag{2.11}$$

と表される．

2.3.2 平均平方二乗誤差

平均平方二乗誤差は MSE と略されることが多い．平均平方二乗誤差 L_{mse} は

$$L_{mse} = ||I_1 - I_2||_2 \tag{2.12}$$

$$= \frac{1}{H \times W \times C} \sum_{h=0}^{H-1} \sum_{w=0}^{W-1} \sum_{c=0}^{C-1} (I_1 - I_2)^2 \tag{2.13}$$

と表される．

2.4 最適化手法

誤差関数を用いて出力画像と教師データの差を小さくする方法には勾配降下法と呼ばれる最適化手法が用いられる．勾配降下法とは，損失関数に対しパラメータの勾配を計算し，損失関数が小さくなるようにパラメータを勾配方向に微小変化を行っていくことにより最小値をとるようなパラメータの値を見つける方法である．以下に勾配法の代表的な手法である確率的勾配降下法 (SGD)，Momentum SGD，RMSprop，Adam [11] について紹介する．ただし，t 回目の更新後のパラメータを $w^{(t)}$ ，損失関数をパラメータで微分したものを $g(w^{(t)})$ とする．各手法において，更新式には事前に決めておく必要があるパラメータが含まれており，推奨値は決まっているものの，学習時には実験的に決定する必要がある．そのようなパラメータをハイパーパラメータと呼ぶ．

2.4.1 確率的勾配降下法 (SDG)

SGD は勾配法の中でも最も基本的な手法である．SGD では，パラメータの更新時にデータ集合の中から一部分のみを取り出して，パラメータの更新を行う．これにより，全ての学

習データに対し更新を行う勾配降下法に比べて効率よく学習することが可能となる．ハイパーパラメータは η である．

$$\boldsymbol{w^{(t+1)}} = \boldsymbol{w^{(t)}} - \eta \cdot g(\boldsymbol{w}^{(t)}) \tag{2.14}$$

2.4.2 Momentum SGD

Momentum SGD は，SGD の更新式に慣性項を付与した手法である．更新量の決定の際に，前回の更新量の定数倍を加算することでパラメータの更新を緩やかにする手法である．ハイパーパラメータは η, μ である．

$$\boldsymbol{\Delta w^{(t)}} = \boldsymbol{w^{(t)}} - \boldsymbol{w^{(t-1)}} \tag{2.15}$$

$$\boldsymbol{w^{(t+1)}} = \boldsymbol{w^{(t)}} - \eta \cdot g(\boldsymbol{w}^{(t)}) + \mu \cdot \boldsymbol{\Delta w^{(t)}} \tag{2.16}$$

2.4.3 RMSprop

RMSprop は，初期学習率を設定したのちは学習係数を自動で調節することが可能な手法である．また，更新式の重み計算部分 (式 (2.17), 式 (2.18)) で勾配部分を二乗することにより，勾配の振動が大きい際には重みを小さくし，振動を抑えるように重みを調節することが可能となった．ハイパーパラメータは $\alpha, \eta_0, h_0, \epsilon$ である．

$$h_t = \alpha \cdot h_{t-1} + (1-\alpha) g(\boldsymbol{w}^{(t)})^2 \tag{2.17}$$

$$\eta_t = \frac{\eta_0}{\sqrt{h_t} + \epsilon} \tag{2.18}$$

$$\boldsymbol{w^{(t+1)}} = \boldsymbol{w^{(t)}} - \eta_t \cdot g(\boldsymbol{w}^{(t)}) \tag{2.19}$$

2.4.4 Adam

Adam [11] は Kingma らによって提案された手法である．Momentum SGD における慣性項のアイデアを 式 (2.20) に導入し，RMSprop における勾配の振動を抑えるアイデアを式 (2.21) に導入している．それにより Momentum SGD と RMSprop を組み合わせたような手法となっている．ハイパーパラメータは $\alpha, \beta_1, \beta_2, \epsilon$ である．また，初期値が決まっ

ている変数もあり，それぞれ $m_{(0)} = 0, v_{(0)} = 0$ である．

$$m_{(t+1)} = \beta_1 \cdot m_t + (1 - \beta_1) \cdot g(\boldsymbol{w}^{(t)}) \tag{2.20}$$

$$v_{(t+1)} = \beta_2 \cdot v_t + (1 - \beta_2) \cdot g(\boldsymbol{w}^{(t)})^2 \tag{2.21}$$

$$\hat{m} = \frac{m_{(t+1)}}{1 - \beta_1^t} \tag{2.22}$$

$$\hat{v} = \frac{v_{(t+1)}}{1 - \beta_2^t} \tag{2.23}$$

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \alpha \cdot \frac{\hat{m}}{\sqrt{\hat{v}} + \epsilon} \tag{2.24}$$

2.5 セマンティックセグメンテーション

セマンティックセグメンテーションとは，画像中に含まれる物体をピクセルレベルで識別するタスクである．セマンティックセグメンテーションには通常 CNN が用いられる．学習の際には，学習用の入力画像のほかに各ピクセルに対しラベル付けが行われた正解データが必要となる．CNN では，画像内の空間的な特徴などを学習し，セグメンテーションの出力を得る．その出力と正解データの差を最小化することで学習を行う．以下にセマンティックセグメンテーションの手法の例を示す．

2.5.1 U-Net

U-Net [15] は，2015 年に Ronneberger らによって提案された手法である．U-Net の構造を 図 2.6 に示す．主に医療画像を対象として発表されたが，一般画像に対しても広く用いられている．U-Net は全ての層が畳み込み層で構成されており，入力画像サイズに制限はない．U-Net では，Contracting path と呼ばれる特徴的な構造が導入された．エンコーダから出力される特徴マップの解像度は小さく，局所的な特徴を失っている．この点を改善するために Contracting path は導入された．Contracting path では，デコーダ部分でアップサンプリングを行った後，サイズが対応するエンコーダの特徴マップを連結する．これにより，アップサンプリングを行う際に局所的な特徴を持つ特徴マップを混ぜることができるため，全体的な特徴と局所的な特徴を両方ともを混ぜてセグメンテーションを行うことができる．これによりモデルの性能向上に成功した．

2.5.2 DeepLab-v3+

DeepLab-v3+ [3] は, 2018 年に Google Inc. の Chen らによって提案された手法で, 2014 年に同じく Chen らによって発表された DeepLab [1] を改善したものである．DeepLab に対し，Atrous Spatial Pyramid Pooling(ASPP) や Atrous Convolution を導入することで

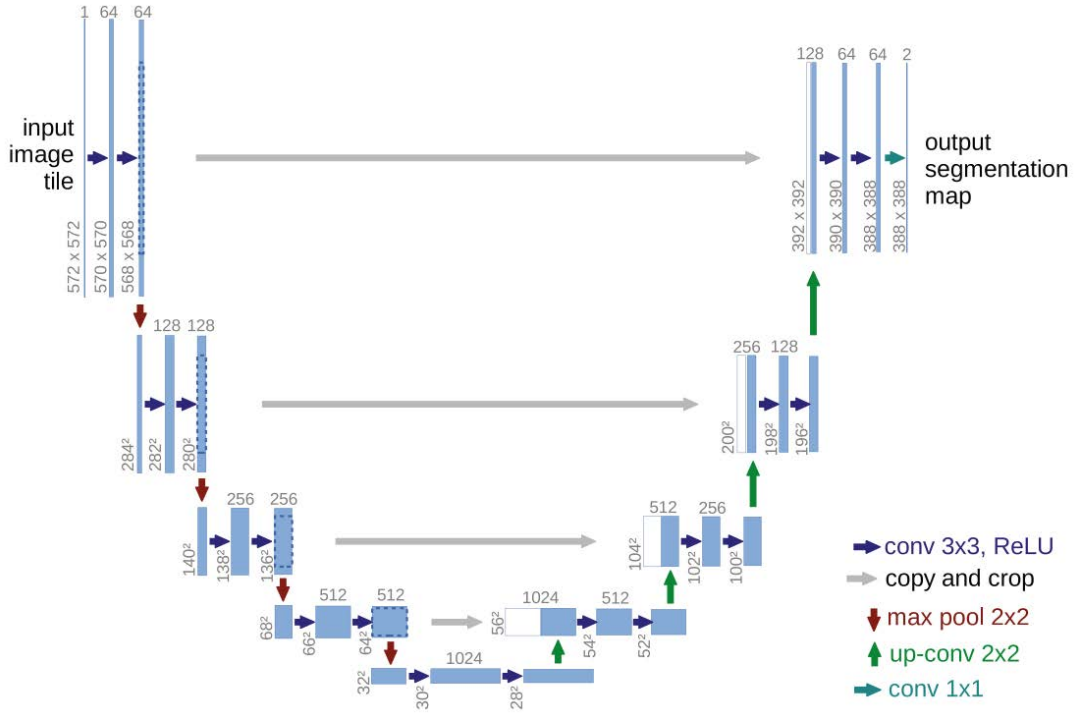

図 2.6: U-Net の構造 ([15] より引用)

特徴マップを小さくせずに画像全体の特徴を得ることが可能となり，DeepLab-v3 [2] における Atrous Convolution を深さ方向とチャンネル方向の畳み込みに分割することによって計算量を削減した．これらの改善により高精度な SemanticSegmentation を実現したモデルである．DeepLab-v3+ の構造を 図 2.7 に示す．

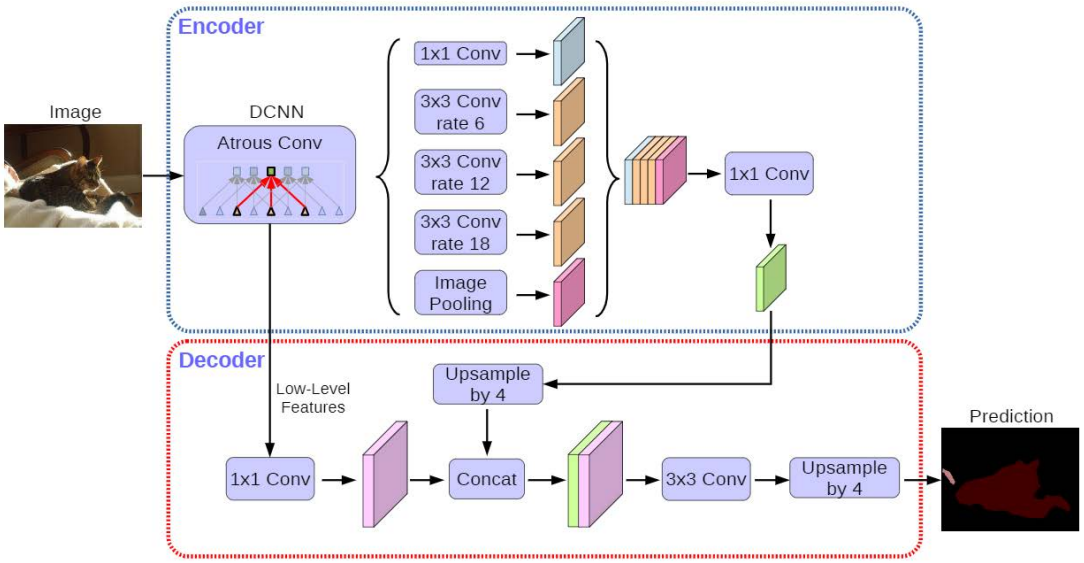

図 2.7: DeepLab-v3+ の構造 ([3] より引用)

2.6 敵対的生成ネットワーク (GAN)

敵対的生成ネットワーク (GAN, Generative Adversarial Networks) [7] は，2014 年に Goodfellow らによって提案された手法である．GAN を用いることで教師無し学習で画像生成モデルを作成することができる．GAN の基本構造を 図 2.8 に示す．

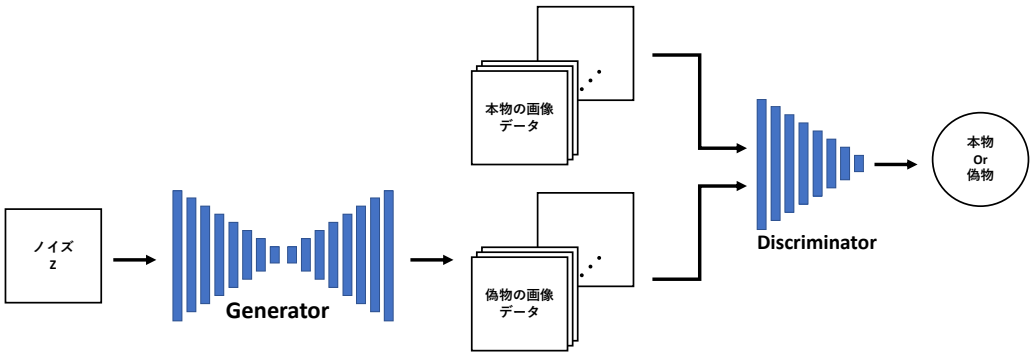

図 2.8: GAN の基本構造

GAN は 2 つの CNN を持っており，片方を Generator，もう片方を Discriminator と呼ぶ．Generator ではノイズ変数 z を入力として $G(z)$ を出力し，Discriminator は画像 I を入力としてスカラー $D(I)$ を出力する．Generator は生成したい画像に似ている画像を出力できるように学習し，Discriminator は生成したい正解画像群と Generator からの出力画像を正しく識別できるように学習を行う．

正解画像を x とすると最適化対象の損失関数 $V(D,G)$ は以下の 式 (2.25) のようになり，識別器 D と生成器 G のミニマックスゲームとして最適化を行う．ここで $\boldsymbol{x} \sim p_{\mathrm{data}}(\boldsymbol{x})$ は確率密度関数 $p_{\mathrm{data}}(\boldsymbol{x})$ で表現される正解画像群から得られるデータ x を表し，$\boldsymbol{z} \sim p_{\mathrm{z}}(\boldsymbol{z})$ は確率密度関数 $p_{\mathrm{z}}(\boldsymbol{z})$ で表現される分布から得られるノイズ z を表す．

$$\min_{G}\max_{D} V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{data}}(\boldsymbol{x})}[\log D(x)] + \mathbb{E}_{\boldsymbol{z}\sim p_{\mathrm{z}}(\boldsymbol{z})}[\log(1 - D(G(z)))] \tag{2.25}$$

識別器 D は $V(D,G)$ を最大化するため，$D(x)$ を大きく，$D(G(x))$ を小さくするように学習する．また，生成器 G は $V(D,G)$ を最小化するため，$D(G(x))$ を大きくするように学習する．これらの学習は識別器 D と生成器 G の片方のパラメータを固定し，識別器と生成器に対して交互に行う．

2.6.1 GAN の学習時のテクニック

GAN は画像生成モデルとして発展してきたが，モデルの学習時にいくつかの問題が起きやすいことが知られている．一つはモード崩壊と呼ばれる問題で，もう一つが損失関数の不安定さである．一つ目のモード崩壊とは，Generator の学習時に Discriminator が騙されやすい画像の生成に成功した場合，Generator にどのようなノイズを入力しても Discriminator を騙すことができる，常に同じような画像しか出力しないようになってしまう現象である．これを改善するためには，学習率を低く設定し直してもう一度学習を行い直すことが有効とされているが，必ずしもそれで改善するわけではない．二つ目の損失関数の不安定さとは，GAN の学習時に計算する Generator と Discriminator の損失関数が安定せずに，片方の損失関数が学習開始後すぐに 0 に収束してしまう現象である．片方の損失関数が圧倒的に早く学習を行ってしまうともう片方の損失関数が学習できなくなってしまう．一般的に，Discriminator の損失関数にこの現象が発生することが多い．これを改善するためには，学習時のハイパーパラメータを調整することや，Discriminator の学習回数と Generator の学習回数をそれぞれ調整するなど，実験的に調整する方法が多く取られている．

以下に，GAN の学習時に使用されるテクニックをまとめる．

Least squares GAN(LS GAN)

Least squares GAN(LS GAN) [14] は，2017 年に Mao らによって提案された手法で，損失関数を工夫することで勾配消失問題を軽減することができる．式 (2.25) に示されるオリジナルの GAN の損失関数から対数関数を排し，二乗誤差を求める関数に変更している．LS GAN で用いられている損失関数を以下に示す．

$$\min_D V_{LSGAN}(D) = \frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\mathrm{data}}(\boldsymbol{x})}[(D(x)-b)^2] \tag{2.26}$$

$$+ \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[(D(G(z))-a)^2] \tag{2.27}$$

$$\min_G V_{LSGAN}(G) = \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[(D(G(z))-c)^2] \tag{2.28}$$

損失関数に含まれる定数である a, b, c は，学習時に事前に決定する．原論文では $a = -1, b = 1, c = 0$ あるいは $a = 0, b = 1, c = 1$ が推奨されている．

Patch GAN

Patch GAN [9] は 2017 年に Isola らによって提案された手法である．GAN における Discriminator は画像を入力として，その画像が正しい入力画像であり Generator から生成された偽画像ではない確率を出力する．しかしその方法では，学習を進める間に出力が

入力画像の一部分のみに大きく影響されてしまう事がわかっている．そこで Patch GAN では，入力画像をある大きさの Patch に分割し，それぞれを Discriminator に通して予測値を計算する．それによって得られた Patch 毎の予測値を平均することで入力画像に対する Discriminator の出力として扱う．この方法をとることで，入力画像の一部分のみに Discriminator の出力への影響が偏ることを回避できる．実装する際には，入力画像を Patch に分割してから Discriminator に入力するのではなく，Discriminator の出力を 1×1 ではなく，ある大きさの特徴マップとすることで入力画像の真偽を出力している．

Label Smoothing

Label Smoothing とは，GAN に限らず画像分類系の機械学習モデルを作成する際に使用される手法である．画像を入力とする CNN から出力される確率値を評価する場合，正解の場合は 1，不正解の場合は 0 とするが，Label Smoothing では，不正解の場合にも 0.1 等の定数を振り分けることで過学習などを抑制する働きがある．GAN における Label Smoothing では，正解画像と偽物画像それぞれを 1,0 から 0.9,0.1 に変更するのでは無く，0.9 と 0 に設定する．この手法は One-sided label smoothing [16] と呼ばれており，GAN の安定化に効果的とされている．

2.6.2 Cycle GAN

Cycle GAN [20] は，2017 年に Zhu らによって提案された手法で，画像生成を目標とする GAN の一種である．Yi らの DualGAN [18] ，Kim らの DiscoGAN [10] も同年に同様の手法を発表した．Cycle GAN は 2 つの画像ドメイン X, Y の間で双方向の変換を行うことができる画像生成ネットワークであり，学習には画像ドメイン X, Y の間でペアとなる画像は不要である点が他の教師あり学習と比べて優れている．Cycle GAN の構造を 図 2.9 に示す．

G, F は画像を入力として画像を出力するオートエンコーダの構造を持つ CNN であり，D_X, D_Y は Discriminator と呼ばれ，画像を入力として確率値を出力する CNN である．G はドメイン X の画像群からドメイン Y の画像を生成できるように学習し，Y についても同様にドメイン Y の画像群からドメイン X の画像を生成できるように学習を行う．D_X はドメイン X の正解画像とドメイン Y の画像から生成された画像 $F(Y)$ を正しく識別できるように学習し，D_Y についても同様にドメイン Y の正解画像とドメイン X の画像から生成された画像 $G(X)$ を正しく識別できるように学習を行う．Cycle GAN を学習させる際に必要な損失関数は敵対性損失 $\mathcal{L}_{GAN}$ とサイクル一貫性損失 $\mathcal{L}_{cyc}$ の二つで，それぞれ

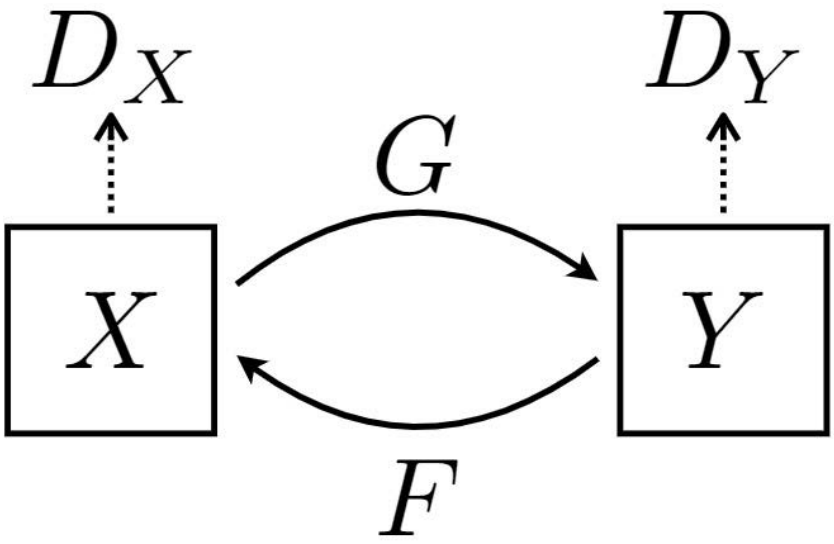

図 2.9: Cycle GAN の構造 ([20] より引用)

以下のように表される．

$$\begin{aligned}\mathcal{L}_{GAN}(G, D_Y, X, Y) &= \mathbb{E}_{y \sim p_{\mathrm{data(y)}}}[\log D_Y(y)] \\ &+ \mathbb{E}_{x \sim p_{\mathrm{data(x)}}}[\log(1 - D_Y(G(x)))] \qquad (2.29) \\ \mathcal{L}_{cyc}(G, F) &= \mathbb{E}_{x \sim p_{\mathrm{data(x)}}}[||F(G(x)) - x||_1] \\ &+ \mathbb{E}_{y \sim p_{\mathrm{data(y)}}}[||G(F(x)) - y||_1] \qquad (2.30)\end{aligned}$$

敵対性損失 $\mathcal{L}_{GAN}$ は一般的な GAN の損失関数と同様で，ドメイン Y の画像とドメイン X からの変換画像 $G(X)$ を正しく識別する D_Y と，それを騙そうとする G のミニマックスゲームとなっている．サイクル一貫性損失 $\mathcal{L}_{cyc}$ は Cycle GAN 特有の損失関数となっており，ドメイン X からの変換画像 $G(X)$ を再変換してドメイン X に戻した画像 $F(G(X))$ は変換を行う前の画像と同じになる事を利用した損失関数である．これらの損失関数を利用して最終的な最適化を行う損失関数 $\mathcal{L}$ は以下のように表される．

$$\begin{aligned}\mathcal{L}(G, F, D_X, D_Y) &= \mathcal{L}_{GAN}(G, D_Y, X, Y) \\ &+ \mathcal{L}_{GAN}(F, D_X, Y, X) \\ &+ \lambda \mathcal{L}_{cyc}(G, F) \qquad (2.31)\end{aligned}$$

この損失関数 $\mathcal{L}$ を利用して

$$G^*, F^* = \arg\min_{G,F} \max_{D_X, D_Y} \mathcal{L}(G, F, D_X, D_Y) \qquad (2.32)$$

とすることで最適化を行う．$\mathcal{L}_{cyc}(G, F)$ の係数 λ は事前に設定される定数であり，$\lambda = 10$ が推奨されている．

実際に Cycle GAN を実装する際にはいくつかの工夫が施されている．一つ目は，Discriminator に対して Patch GAN [9] で提案されている手法を用いている点である．二

つ目は学習を安定化させるために損失関数から対数関数を除外した点である．これには least-squares GAN [14] に示される手法を用いている．具体的には，式 (2.29) に示した $\mathcal{L}_{GAN}$ を，Segmenter，Desegmenter の学習時それぞれについて以下のように最小化するように置き換える．

$$G^* = \arg\min_G \mathbb{E}_{x\sim p_{\mathrm{data(x)}}}[(D_Y(G(x))-1)^2] \tag{2.33}$$

$$D_Y^* = \arg\min_{D_Y} \mathbb{E}_{y\sim p_{\mathrm{data(y)}}}[(D_Y(y)-1)^2] + \mathbb{E}_{x\sim p_{\mathrm{data(x)}}}[(D_Y(G(x)))^2] \tag{2.34}$$

これによって，GAN の学習時に多く発生してしまう勾配消失問題を避ける事を期待することができる．

Cycle GAN を Semantic Segmentation に対して用いる際には画像ドメイン X, Y のそれぞれに学習用画像 x と教師ラベル画像 y を設定する．ここで用いる学習用画像と教師ラベル画像の間には対応関係が必要でないため，現状存在する学習用画像 x とその教師ラベル画像 y に対して学習用画像と同じドメインの別の画像群 x' に置き換えて学習を行うことが可能となる．これは画像群 x' に対する教師無し学習と考えることができる．

第 3 章

提案手法

3.1 概要

本研究では教師ラベルが付与されていない車載カメラ画像の Semantic Segmentation を行うことを最終的な目標とする．本研究では，Cycle GAN をベースとした手法を提案する．提案手法では入力画像とペアになっていない教師ラベルを用いて学習することで教師無し学習とし，Cycle GAN をそのまま用いるよりも高い精度を得ることができる手法を提案する．

本研究が対象とする画像群は車載カメラ画像であり，ドライブレコーダーの設置数の増加や自動運転技術の進歩によってこれらの車載カメラ画像に対する認識技術の発展は不可欠となっている．本研究では，現状公開されているデータセットを用いて，意図的に入力画像とラベル画像をペアにせずに学習する．そうすることで入力画像に対応する教師ラベルを与えずに学習を行うため，教師無し学習の状況を構築する．これにより，教師ラベルが付与されていない画像群に対する教師無し学習の新しい手法を提案する事を目標とする．

3.2 提案手法

3.2.1 ネットワークの呼称

本研究では車載カメラ画像群を教師ラベル画像群に変換する CNN と，その逆変換を行う CNN の二つを用いる．前者を Segmenter，後者を Desegmenter と呼称する．また，本物の車載カメラ画像か否か を識別する CNN を D_X ，本物の教師ラベル画像か否かを識別する CNN を D_Y と呼称する．モデルの呼称について 図 3.1 に示す．

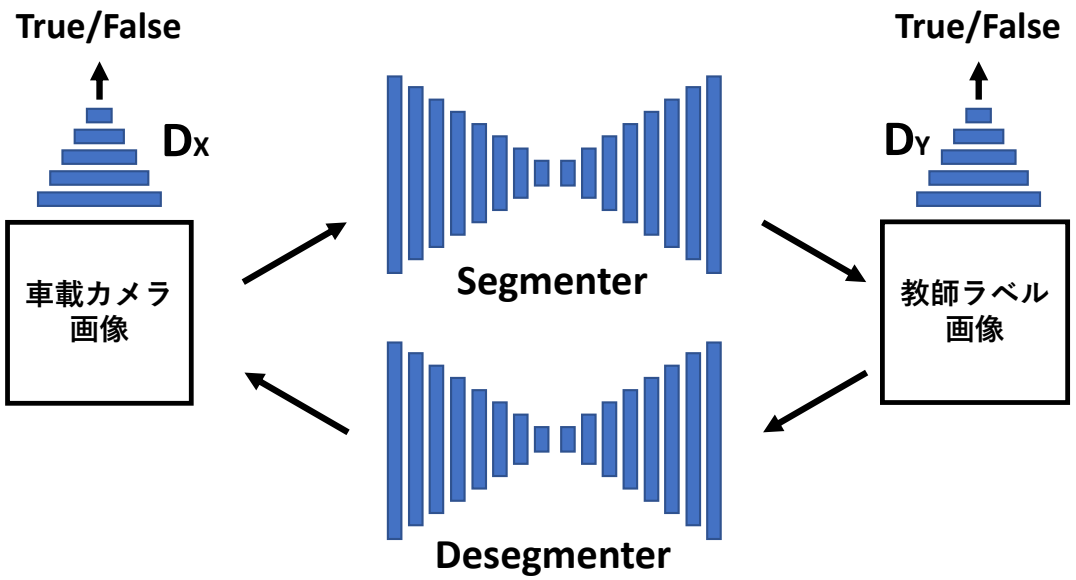

図 3.1: 提案手法における CNN の呼称

3.2.2 提案モデル

本研究の提案モデルを 図 3.2 に示す．提案モデルは Cycle GAN のモデル (図 2.9) をベースに構築した．入力画像と教師ラベル画像に対してそれぞれ適当な Affine 変換を行うことで，新たな損失関数を追加した．提案手法では，Cycle GAN に新たな CNN を追加しない事から，全体のモデルサイズやパラメータ数は増加しない．

Segmenter のモデルには DeepLab-v3+ [3] を用い，Desegmenter には 表 3.1 に示す構造を持つ CNN を用いた．また，Discriminator は 表 3.3 に示す構造を持つ CNN を用いた．Discriminator に用いた Leaky ReLU 関数の定数 a は 0.2 を用いた．Discriminator には Patch GAN [9] で提案されている手法を用い，Discriminator の出力をある大きさの特徴マップとすることで入力画像全体に対して真偽を判断することができるようにした．

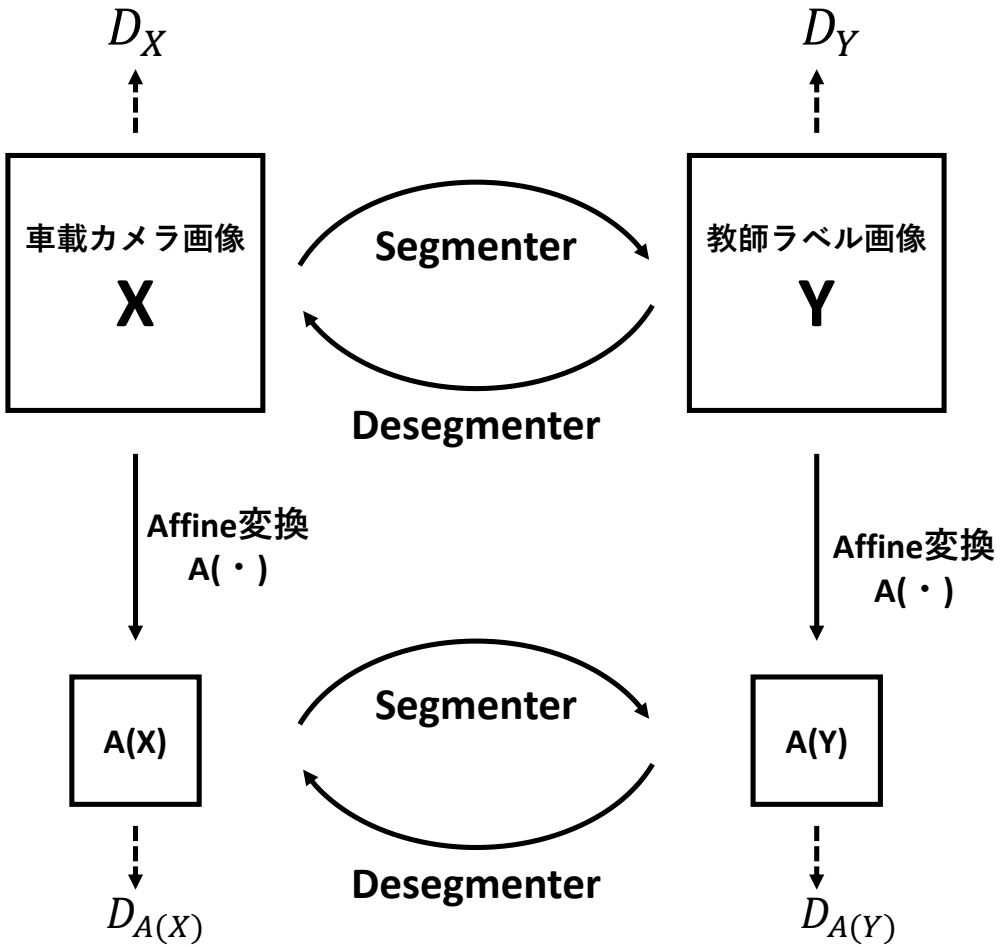

図 3.2: 提案モデル

提案手法において使用した損失関数を以下に示す.

$$\begin{aligned}
\mathcal{L}_{GAN}(Seg, D_Y, X, Y) &= \mathbb{E}_{y\sim p_{\mathrm{data(y)}}}[\log D_Y(y)] \\
&\quad + \mathbb{E}_{x\sim p_{\mathrm{data(x)}}}[\log(1 - D_Y(Seg(x)))] \qquad (3.1) \\
\mathcal{L}_{Aff}(Seg, D_Y, X) &= \mathbb{E}_{y\sim p_{\mathrm{data(y)}}}[\log D_Y(A(y))] \\
&\quad + \mathbb{E}_{x\sim p_{\mathrm{data(x)}}}[\log(1 - D_Y(Seg(A(x))))] \qquad (3.2) \\
\mathcal{L}_{cyc}(Seg, Deseg) &= \mathbb{E}_{x\sim p_{\mathrm{data(x)}}}[||Deseg(Seg(x)) - x||_1] \\
&\quad + \mathbb{E}_{y\sim p_{\mathrm{data(y)}}}[||Seg(Deseg(x)) - y||_1] \qquad (3.3) \\
\mathcal{L}_{cycAff}(Seg, Deseg) &= \mathbb{E}_{x\sim p_{\mathrm{data(x)}}}[||Seg(A(x)) - A(Seg(x))||_1] \\
&\quad + \mathbb{E}_{y\sim p_{\mathrm{data(y)}}}[||Deseg(A(y)) - A(Deseg(y))||_1] \qquad (3.4)
\end{aligned}$$

式 (3.1)，式 (3.3) に示したものは Cycle GAN と同様の損失関数である．また，式 (3.2) に示した関数は，それぞれのドメインの画像に対し事前に Affine 変換を施した画像を Segmenter，Desegmenter で変換し，それらを Discriminator で識別する事で，Segmenter と Desegmenter は Discriminator を騙すように学習し，Discriminator はそれを見抜くよ

表 3.1: Desegmenter の構造

層種	出力特徴マップサイズ
Input	$N \times H \times W$
ReflectionPad.	$N \times (H+6) \times (W+6)$
Convolution	$64 \times H \times W$
Instance Norm.	$64 \times H \times W$
ReLU	$64 \times H \times W$
Convolution	$128 \times H/2 \times W/2$
Instance Norm.	$128 \times H/2 \times W/2$
ReLU	$128 \times H/2 \times W/2$
Convolution	$256 \times H/4 \times W/4$
Instance Norm.	$256 \times H/4 \times W/4$
ReLU	$256 \times H/4 \times W/4$
Residual Block (表 3.2) $\times 6$	$256 \times H/4 \times W/4$
Trans Conv.	$128 \times H/2 \times W/2$
Instance Norm.	$128 \times H/2 \times W/2$
ReLU	$128 \times H/2 \times W/2$
Trans Conv.	$64 \times H \times W$
Instance Norm.	$64 \times H \times W$
ReLU	$64 \times H \times W$
ReflectionPad.	$64 \times (H+6) \times (W+6)$
Convolution	$3 \times H \times W$
Tanh	$3 \times H \times W$

うに学習を行うための損失関数である．最後に，式 (3.4) に示した関数は，それぞれのドメインの画像を Segmenter，Desegmenter で変換したのちに Affine 変換を施したものと，それぞれのドメインの画像を先に Affine 変換を施した後に Segmenter，Desegmenter で変換したものが一致することを利用した損失関数である．

表 3.2: Residual Block の構造

層種	出力特徴マップサイズ
Input	$N \times H \times W$
ReflectionPad.	$N \times (H+2) \times (W+2)$
Convolution	$N \times H \times W$
Instance Norm.	$N \times H \times W$
ReLU	$N \times H \times W$
ReflectionPad.	$N \times (H+2) \times (W+2)$
Convolution	$N \times H \times W$
Instance Norm.	$N \times H \times W$

表 3.3: Discriminator の構造

層種	出力特徴マップサイズ
Input	$N \times H \times W$
Convolution	$64 \times H/2 \times W/2$
LeakyReLU	$64 \times H/2 \times W/2$
Convolution	$128 \times H/4 \times W/4$
Instance Norm.	$128 \times H/4 \times W/4$
LeakyReLU	$128 \times H/4 \times W/4$
Convolution	$256 \times H/4 \times W/4$
Instance Norm.	$256 \times H/4 \times W/4$
LeakyReLU	$256 \times H/4 \times W/4$
Convolution	$512 \times H/4 \times W/4$
Instance Norm.	$512 \times H/4 \times W/4$
LeakyReLU	$512 \times H/4 \times W/4$
Convolution	$1 \times H/4 \times W/4$

以上の損失関数を定数で重みづけすることで全体としての損失関数 $\mathcal{L}$ を設計した.

$$\begin{aligned}\mathcal{L}(Seg, Deseg, D_X, D_Y) = &\lambda_{seg} \cdot \mathcal{L}_{GAN}(Seg, D_Y, X, Y) \\ &+ \lambda_{deseg} \cdot \mathcal{L}_{GAN}(Deseg, D_X, Y, X) \\ &+ \lambda_{aff} \cdot (\mathcal{L}_{Aff}(Seg, D_Y, X) + \mathcal{L}_{Aff}(Deseg, D_X, Y)) \\ &+ \lambda_{cyc} \cdot \mathcal{L}_{cyc}(Seg, Deseg) \qquad (3.5) \\ &+ \lambda_{cycAff} \cdot \mathcal{L}_{cycAff}(Seg, Deseg) \qquad (3.6)\end{aligned}$$

式 (3.5) において各損失関数に付加されている係数である $\lambda_{seg}, \lambda_{deseg}, \lambda_{aff}, \lambda_{cyc}, \lambda_{cycAff}$ は実験的に決定する定数とした.

さらに，損失関数に対数関数が含まれる事を避けるために Least squares GAN [14] に示される手法を用いた．具体的には，式 (3.1) ，式 (3.2) に示した $\mathcal{L}_{GAN}$ と $\mathcal{L}_{Aff}$ を，Segmenter，Desegmenter，D_X，D_Y の学習時それぞれについて以下のように最小化するように置き換える.

$$\begin{aligned}Seg^* = \arg\min_{Seg}(&\mathbb{E}_{x\sim p_{\mathrm{data(x)}}}[(D_Y(Seg(x)) - 1)^2] \\ &+ \mathbb{E}_{x\sim p_{\mathrm{data(x)}}}[(D_Y(Seg(A(x))) - 1)^2]) \qquad (3.7) \\ Deseg^* = \arg\min_{Deseg}(&\mathbb{E}_{y\sim p_{\mathrm{data(y)}}}[(D_X(Deseg(y)) - 1)^2] \\ &+ \mathbb{E}_{y\sim p_{\mathrm{data(y)}}}[(D_X(Deseg(A(y))) - 1)^2]) \qquad (3.8) \\ D_X^* = \arg\min_{D_X}(&\mathbb{E}_{x\sim p_{\mathrm{data(x)}}}[(D_X(x) - 1)^2] \\ &+ \mathbb{E}_{y\sim p_{\mathrm{data(y)}}}[(D_X(Deseg(y)))^2] \\ &+ \mathbb{E}_{y\sim p_{\mathrm{data(y)}}}[(D_X(Deseg(A(y))))^2]) \qquad (3.9) \\ D_Y^* = \arg\min_{D_Y}(&\mathbb{E}_{y\sim p_{\mathrm{data(y)}}}[(D_Y(y) - 1)^2] \\ &+ \mathbb{E}_{x\sim p_{\mathrm{data(x)}}}[(D_Y(Seg(x)))^2] \\ &+ \mathbb{E}_{x\sim p_{\mathrm{data(x)}}}[(D_Y(Seg(A(x))))^2]) \qquad (3.10)\end{aligned}$$

これによって，Cycle GAN の学習を安定化することが期待することができる.

第 4 章

実験

4.1 データセット

本研究の実験では，Cityscapes [4] を用いた．Cityscapes は車載カメラ画像のデータセットであり，ドイツ 19 都市，フランス 1 都市，スイス 1 都市の計 21 都市において撮影された車載カメラの動画と，そこから得られた幅 2048× 高さ 1024 ピクセルの静止画像，さらにそれらに手動で付与された 20 クラスのラベルが含まれる．学習に用いた画像のいくつかを 図 4.1 に示す．また，撮影場所の一覧と画像枚数を以下の 表 4.1 に示し，付与されているクラス一覧を 表 4.2 に示す．

4.2 評価指標

セマンティックセグメンテーションで一般的に用いられる評価指標のうち，本研究では mean IoU，mean accuracy，pixel accuracy を用いる．これらの計算方法を以下に示す．

$$\text{mean IoU} : \frac{1}{K}\sum_i \frac{N_{ij}}{\sum_j N_{ij} + \sum_j N_{ji} - N_{ij}} \tag{4.1}$$

$$\text{mean accuracy} : \sum_i \frac{N_{ij}}{\sum_j N_{ij}} \tag{4.2}$$

$$\text{pixel accuracy} : \frac{\sum_i N_{ij}}{\sum_i \sum_j N_{ij}} \tag{4.3}$$

但し，ラベルは K 種類とし，正解ラベルが i であるピクセルをラベル j に分類した総数を N_{ij} とする．

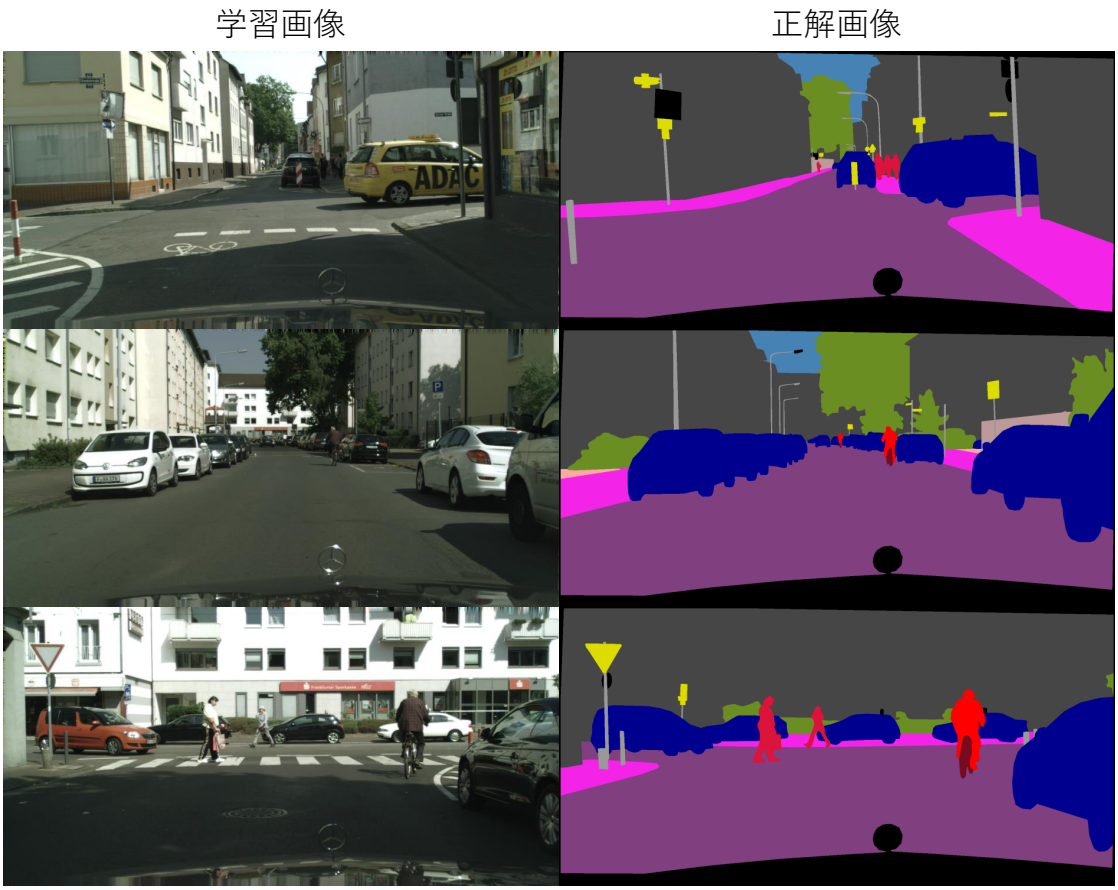

図 4.1: 使用したデータセットの例

4.3 実験方法

提案手法の学習方法について説明する．表 4.1 に示した都市のうち，項番 19 のフランクフルト，項番 20 のリンダウ，項番 21 のミュンスターという 3 都市における計 500 枚の画像を入力画像として使用した．学習の際にペアで入力する教師ラベル画像は，入力画像として使用した 3 都市以外の 18 都市からランダムに使用した．Segmenter には DeepLab-v3+ [3] を用い，Desegmenter には 表 3.1 に示す構造を持つ CNN を用いた．D_X ，D_Y には表 3.3 に示す構造を持つ CNN を用いた．Affine 変換として，画像の高さと幅をそれぞれ半分にすることで解像度を半分にする操作を用いた．学習は 500Epoch 行い，損失関数の最適化手法としては 式 (2.24) に示す Adam [11] を用いた．また，学習率は 0.0002 で固定とした．学習中は mean IoU 式 (4.1) を毎 Epoch 算出し，学習データに対して mean IoU が最大となる Epoch を最終的なモデルとして使用する．比較対象としたモデルを以下の 表 4.3 に示す．ただし，LS は Label Smoothing を表す．提案手法の損失関数 (式 (3.5)) におい

表 4.1: Cityscapes データセットの撮影都市と画像枚数

	都市名	所在国	画像枚数
1	アーヘン	ドイツ	174
2	ボーフム	ドイツ	96
3	ブレーメン	ドイツ	316
4	ケルン	ドイツ	154
5	ダルムシュタット	ドイツ	85
6	デュッセルドルフ	ドイツ	221
7	エアフルト	ドイツ	109
8	ハンブルク	ドイツ	248
9	ハノーファー	ドイツ	198
10	イエーナ	ドイツ	119
11	クレーフェルト	ドイツ	99
12	メンヒェングラートバッハ	ドイツ	94
13	ストラスブール	フランス	365
14	シュツットガルト	ドイツ	196
15	テュービンゲン	ドイツ	144
16	ウルム	ドイツ	95
17	ヴァイマール	ドイツ	142
18	チューリッヒ	スイス	122
19	フランクフルト	ドイツ	267
20	リンダウ	ドイツ	59
21	ミュンスター	ドイツ	174

て，$\lambda_{aff}, \lambda_{cycAff}$ を 0 とすることで，既存手法である Cycle GAN [20] と同様の手法となる．また，提案手法では $\lambda_{aff}, \lambda_{cycAff}$ の組み合わせを複数種類実験した．

表 4.2: Cityscapes データセットの撮影都市と画像枚数

	クラス名	クラスグループ	RGB 値
1	void	void	(0, 0, 0)
2	road	flat	(128, 64,128)
3	sidewalk	flat	(244, 35,232)
4	building	construction	(70, 70, 70)
5	wall	construction	(102,102,156)
6	fence	construction	(190,153,153)
7	pole	object	(153,153,153)
8	trafic light	object	(250,170, 30)
9	trafic sign	object	(220,220, 0)
10	vegetation	nature	(107,142, 35)
11	terrain	nature	(152,251,152)
12	sky	sky	(70,130,180)
13	person	human	(220, 20, 60)
14	rider	human	(255, 0, 0)
15	car	vehicle	(0, 0,142)
16	truck	vehicle	(0, 0, 70)
17	bus	vehicle	(0, 60,100)
18	train	vehicle	(0, 80,100)
19	motercycle	vehicle	(0, 0,230)
20	bicycle	vehicle	(119, 11, 32)

表 4.3: 比較を行ったモデルの概要

Model	λ_{seg}	λ_{deseg}	λ_{aff}	λ_{cyc}	λ_{cycAff}	LS
Cycle GAN	1	1	0	10	0	
提案手法①	1	1	1	10	0	
提案手法②	1	1	1	10	10	
提案手法③	1	1	1	5	0	
提案手法④	1	1	1	5	5	
提案手法⑤	1	1	1	10	0	✓
提案手法⑥	1	1	1	10	10	✓

4.4 実験結果

4.4.1 定量的評価

今回実験を行った各手法に関して，4.2 章 に示した各評価指標を算出した．まとめたものを 表 4.4 に示す．

表 4.4: 定量的評価

Model	mean IoU	mean acc.	pixel acc.
Cycle GAN	0.1080	0.1844	0.3956
提案手法①	0.1145	0.1925	0.3938
提案手法②	0.0976	0.1878	0.4512
提案手法③	0.0962	0.1886	0.4446
提案手法④	0.1078	0.1844	0.4362
提案手法⑤	**0.1156**	**0.2015**	**0.4103**
提案手法⑥	0.0974	0.1810	0.4142

以上の結果より，提案手法⑤の結果が最も優れている事がわかった．

4.4.2 定性的評価

ここでは，実験によって得られたそれぞれのモデルによるセグメンテーション結果を示す．本章では完全に教師無し学習として実験を行っているため，評価の際の画像群も学習に使用したもので行う．結果のいくつかを以下の 図 4.2 と 図 4.3 に示す．

Cycle GAN は，二つの画像ドメイン間の双方向の変換を行う CNN を同時に学習するため，一方の画像を変換したのち，再変換を行うことで元の画像ドメインに戻すことができる．いくつかの例を 図 4.4 と 図 4.5 に示す．図 4.4 より，入力画像を Segmenter に入力し，その出力結果を Desegmenter に入力すれば元に戻る，といった事が確認できる．図 4.5 より，Cycle GAN では再構成画像において色情報が消失してしまっているが，提案手法①・⑤においては色情報を保持することができている事がわかる．比較的 mIoU が高かった提案手法①・⑤は Desegmenter による再構成の能力も高いことがわかった．

また，提案手法②・④では，モード崩壊と考えられる現象が発生してしまっているが，Label Smoothing を使用した提案手法⑤・⑥ではモード崩壊は発生していない．このことからも通常の GAN だけでなく Cycle GAN の構造を持つ GAN にも Label Smoothing が有効であることが確認できた．

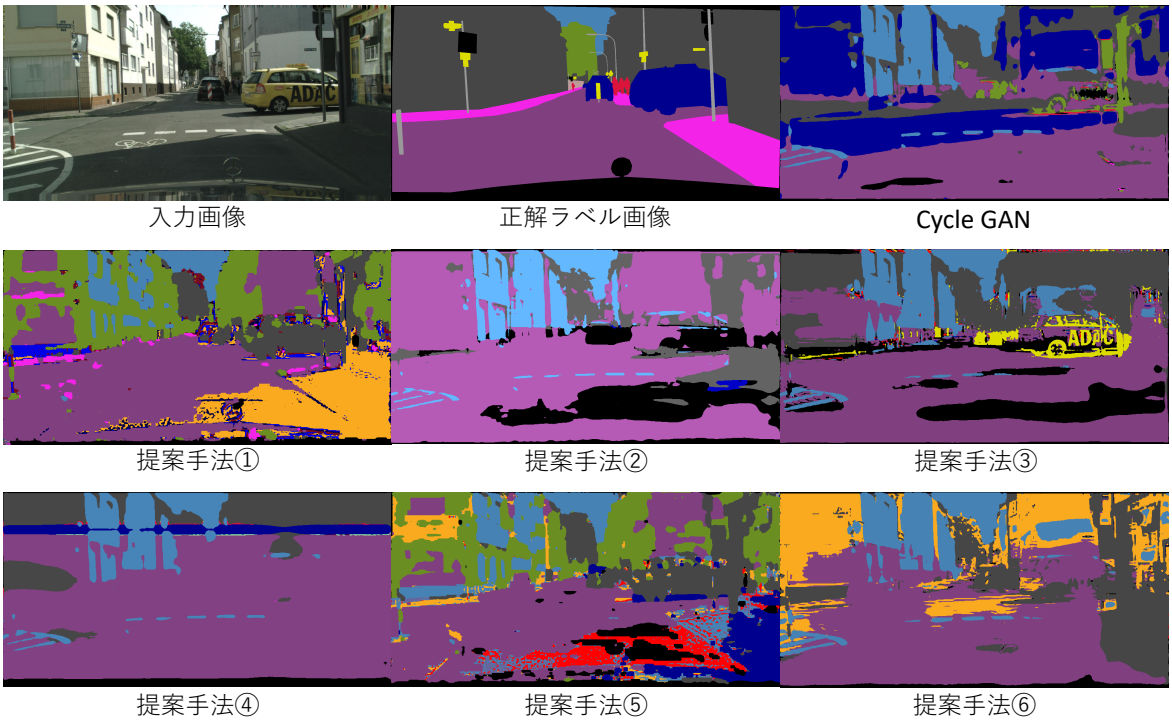

図 4.2: 各手法のセグメンテーション結果の例①

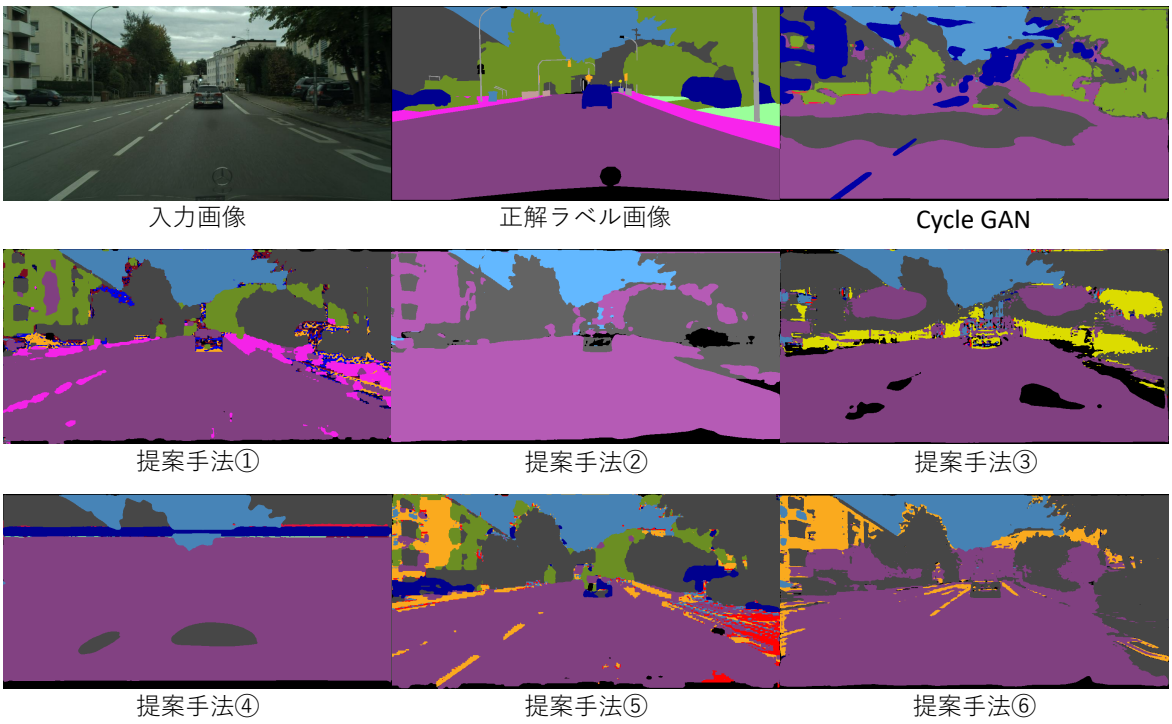

図 4.3: 各手法のセグメンテーション結果の例②

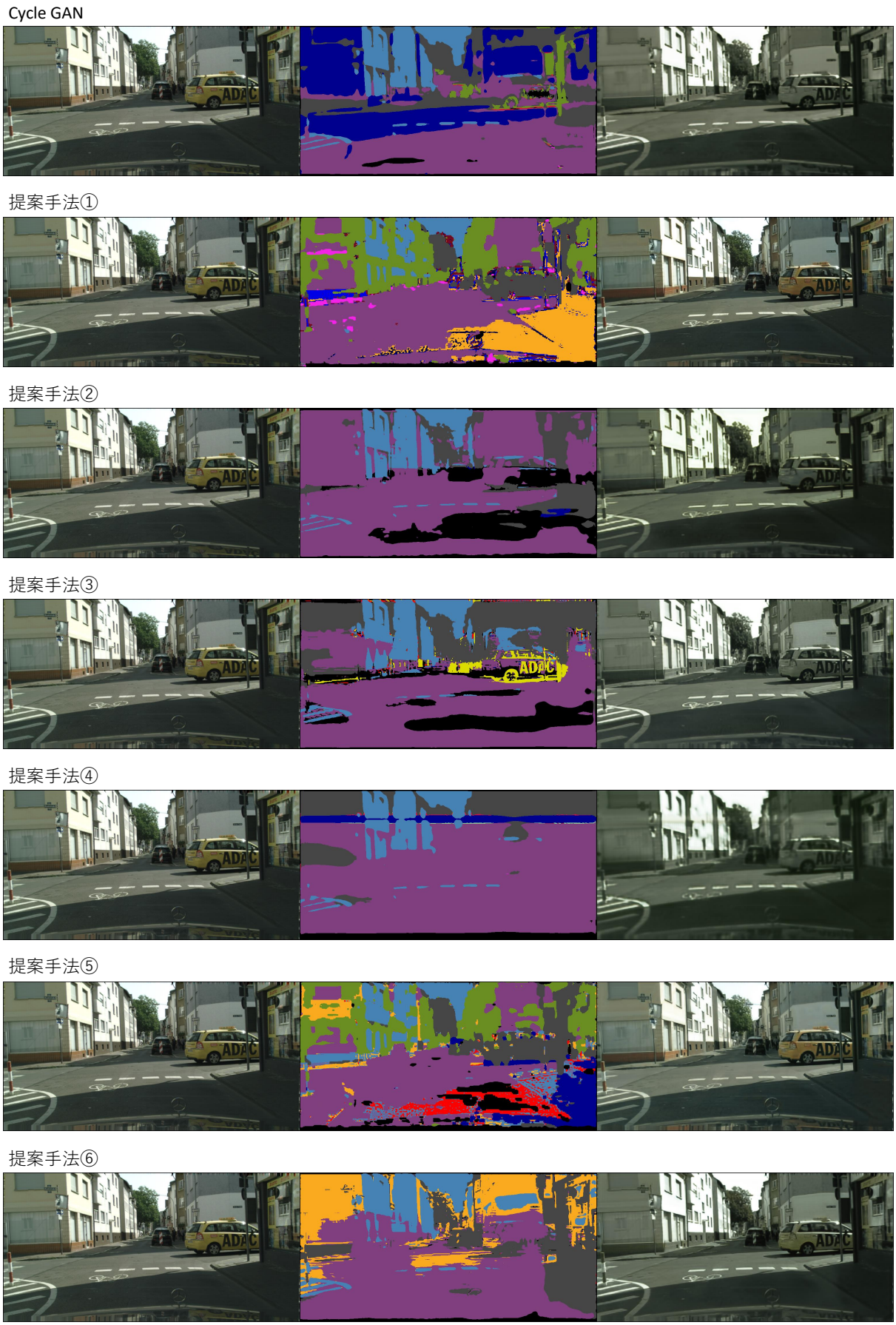

図 4.4: 入力画像とラベル画像の双方向変換① (左列 (a): 入力画像，中列 (b):Segmentetion 画像，右列 (c): 再構成画像)

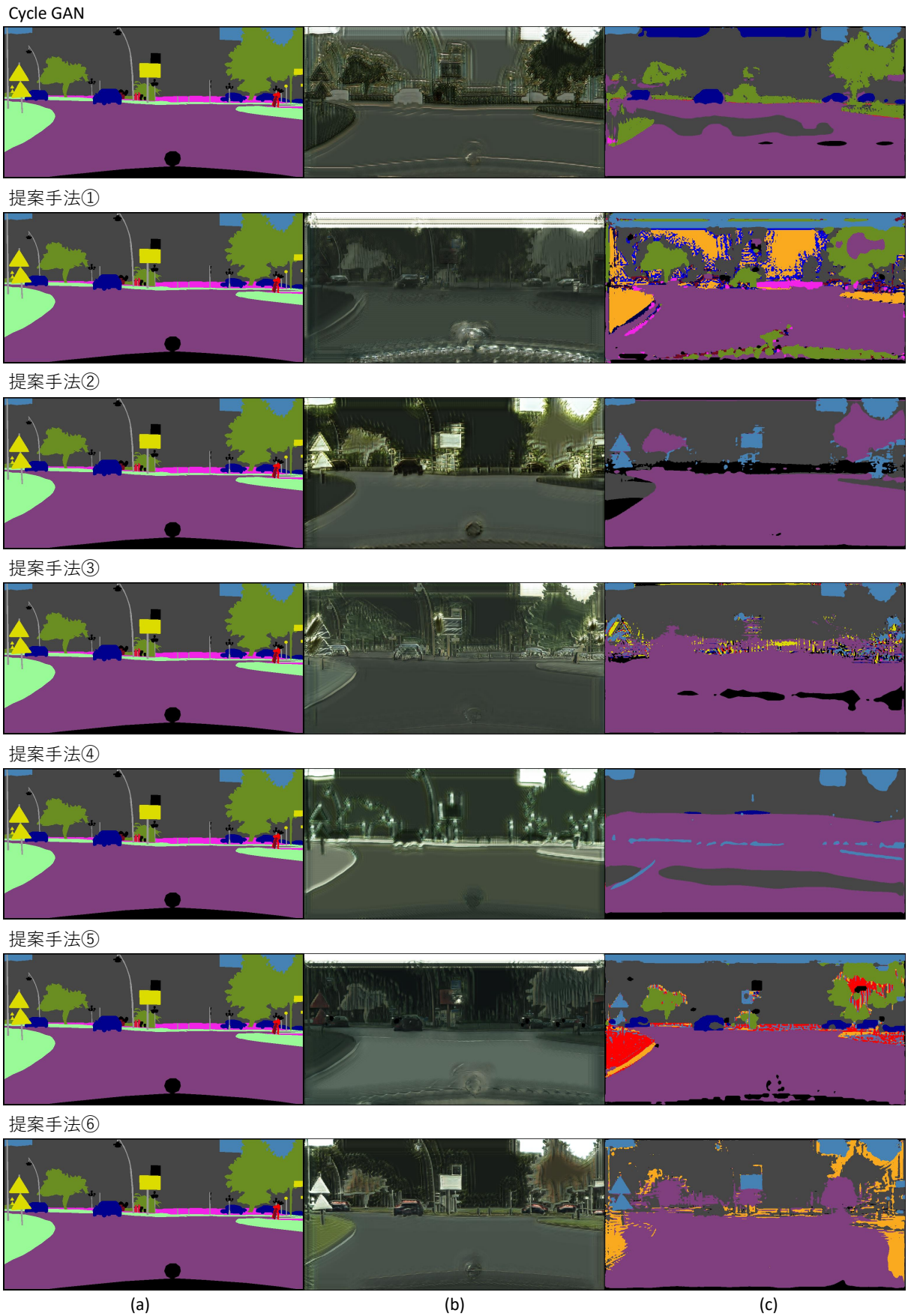

図 4.5: 入力画像とラベル画像の双方向変換② (左列 (a): ラベル画像，中列 (b):Desegment 画像，右列 (c): 再構成画像)

第 5 章

結論

本研究では，Cycle GAN を用いて，教師無し学習による Semantic Segmentation を行う手法を提案した．入力画像とラベル画像をそのまま Segmenter と Desegmenter に入力する事とは別に，それぞれの画像を Affine 変換した画像も入力することで追加の損失関数を設計した．これによって，全体のモデルサイズやパラメータ数を増加させることなく，性能を向上すると考えられる．

実験結果から，Cycle GAN をそのまま用いた Segmentation 結果よりも mIoU を向上することを示した．また，入力画像を変換し，再構成した画像についても，Cycle GAN よりも定性的に優れていることを示した．

参考文献

[1] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. 12 2014.

[2] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. 6 2017.

[3] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. 2 2018.

[4] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

[6] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks, 2011.

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, Vol. 27. Curran Associates, Inc., 2014.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 2016-Decem, 2016.

[9] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE*

Conference on Computer Vision and Pattern Recognition (CVPR), July 2017.

[10] Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jung Kwon Lee, and Jiwon Kim. Learning to discover cross-domain relations with generative adversarial networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, Vol. 70 of *Proceedings of Machine Learning Research*, pp. 1857–1865. PMLR, 06–11 Aug 2017.

[11] Diederik P. Kingma and Jimmy Lei Ba. Adam: A method for stochastic optimization. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, pp. 1–15, 2015.

[12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. 2012.

[13] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models, 2013.

[14] Xudong Mao, Qing Li, Haoran Xie, Raymond Y.K. Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 9351, pp. 234–241. Springer Verlag, 2015.

[16] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. Improved techniques for training gans. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, Vol. 29. Curran Associates, Inc., 2016.

[17] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, pp. 1–14, 2015.

[18] Zili Yi, Hao Zhang, Ping Tan, and Minglun Gong. Dualgan: Unsupervised dual learning for image-to-image translation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[19] Guangyong Zeng, Yi He, Zongxue Yu, Xi Yang, Ranran Yang, and Lei Zhang. Preparation of novel high copper ions removal membranes by embedding organosilane-functionalized multi-walled carbon nanotube. *Journal of Chemical Technology and Biotechnology*, Vol. 91, No. 8, pp. 2322–2330, 2016.

[20] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-

to-image translation using cycle-consistent adversarial networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2242–2251, 2017.

付録 A

実装の詳細

A.1 使用した PC

本研究において使用した PC(NORISEN・KAWAUSO) のスペックを以下の 表 A.1 と表 A.2 にそれぞれ示す.

表 A.1: 使用した PC(NORISEN) の詳細スペック

	詳細
CPU	Intel ® Core™ i7-6700K @ 4.00 GHz
Ram	64GB
GPU 1.	NVIDIA ® GeForce GTX 1080 8GB
GPU 2.	NVIDIA ® GeForce GTX 1080 8GB
OS	Ubuntu 20.04 LTS

表 A.2: 使用した PC(KAWAUSO) の詳細スペック

	詳細
CPU	Intel ® Xeon ® ES-2637 v4 @ 3.50 GHz
Ram	320GB (32GB ×10)
GPU 1.	NVIDIA ® Tesla V100 32GB
GPU 2.	NVIDIA ® Tesla V100 32GB
GPU 3.	NVIDIA ® Tesla P100 16GB
OS	Ubuntu 18.04 LTS

A.2　使用した言語と主要ライブラリ

本研究において使用したプログラム言語と，併せて使用した主要ライブラリのバージョンを以下の 表 A.3 にそれぞれ示す.

表 A.3: 使用した言語とライブラリのバージョン

言語・ライブラリ	バージョン
Python	3.6
PyTorch	1.4.0
NumPy	1.16
cuDNN	7.6.5
CUDA	10.1

付録 B

セグメンテーション結果

教師無し学習の研究における学習画像群に対するセグメンテーション結果のいくつかを図 B.1，図 B.2 に示す．また，Segmenter による変換後に Desegmenter による再変換を行った結果のいくつかを図 B.3，図 B.4 に示し，その逆の操作を行った結果を図 B.5，図 B.6 に示す．

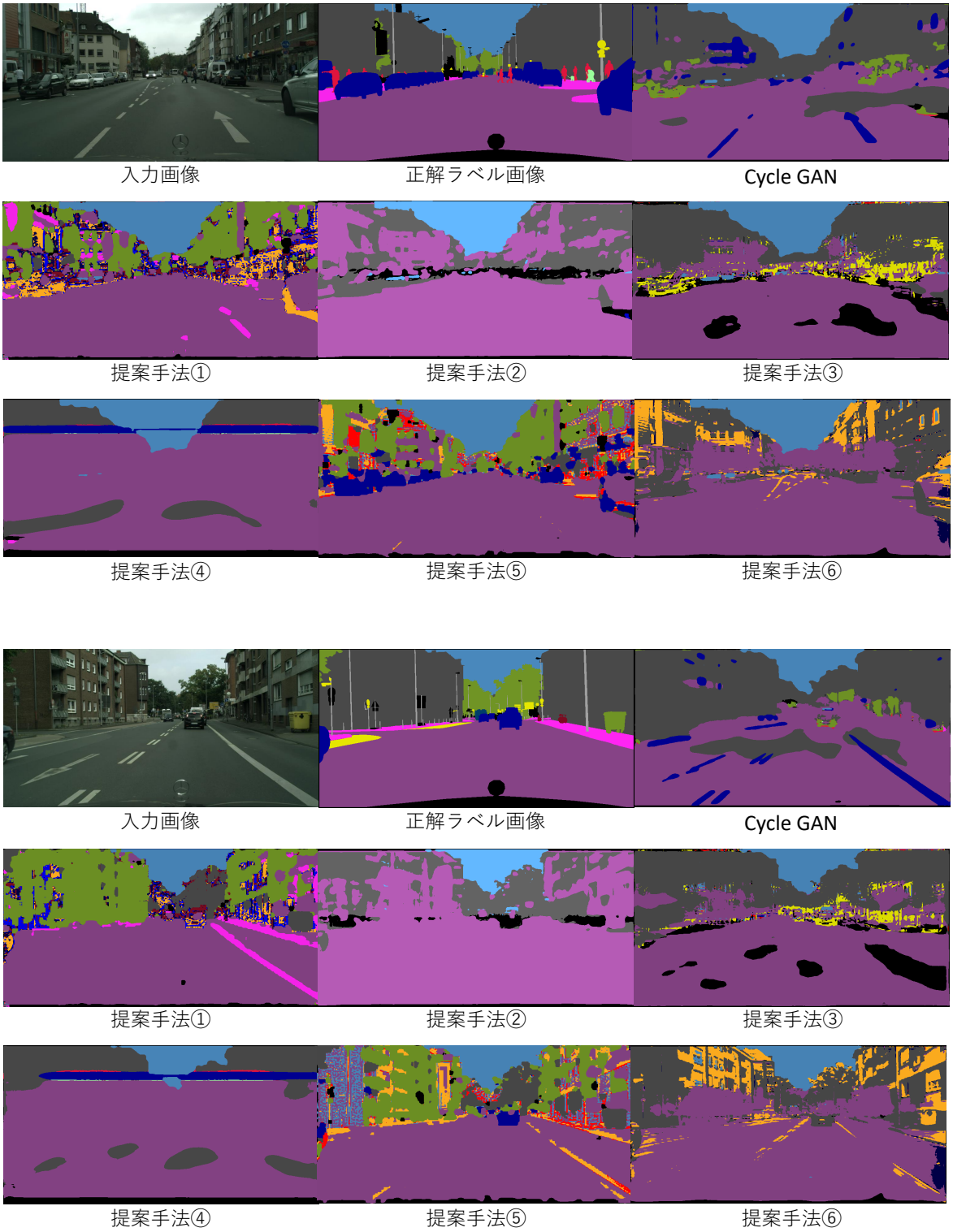

図 B.1: セグメンテーション結果①

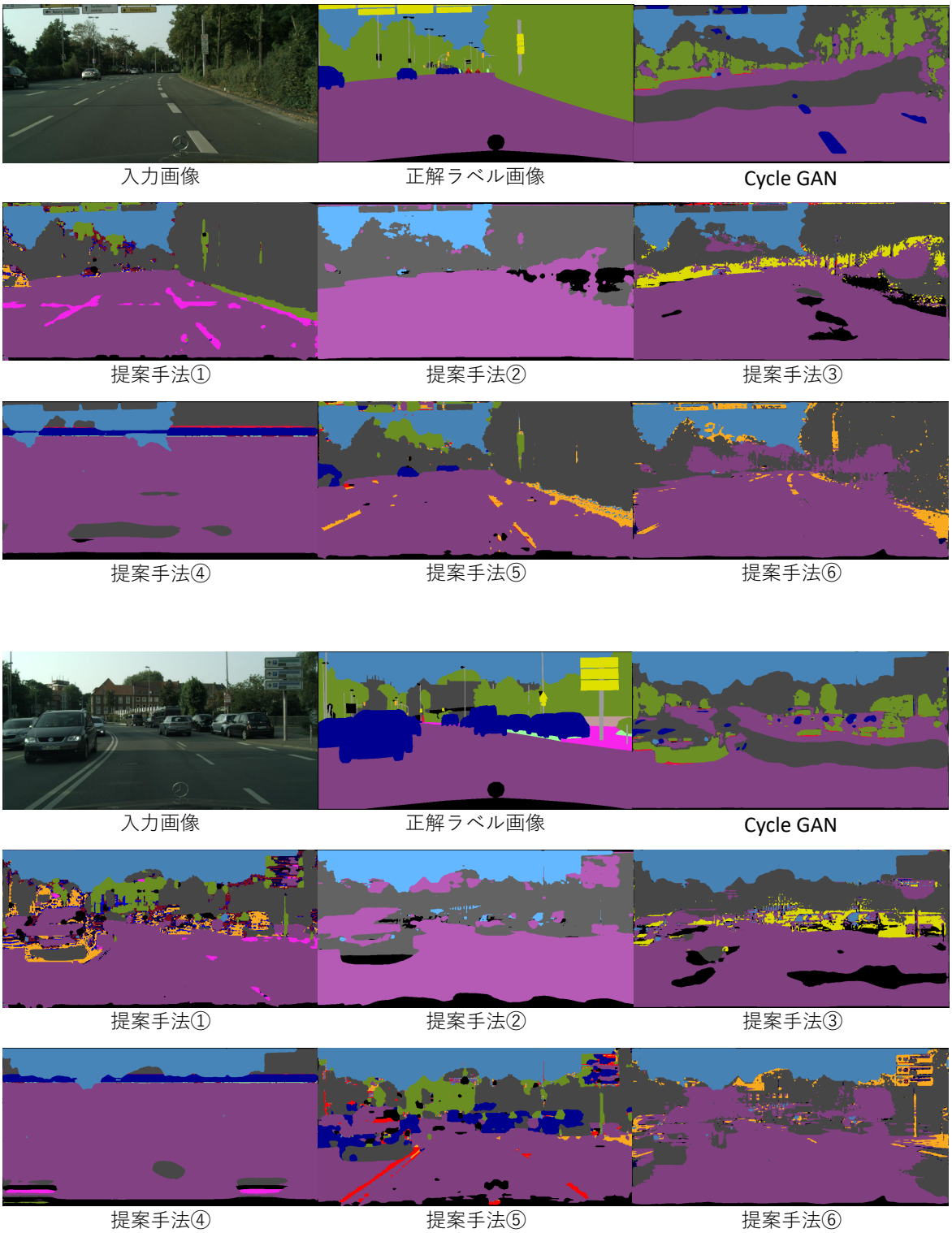

図 B.2: セグメンテーション結果②

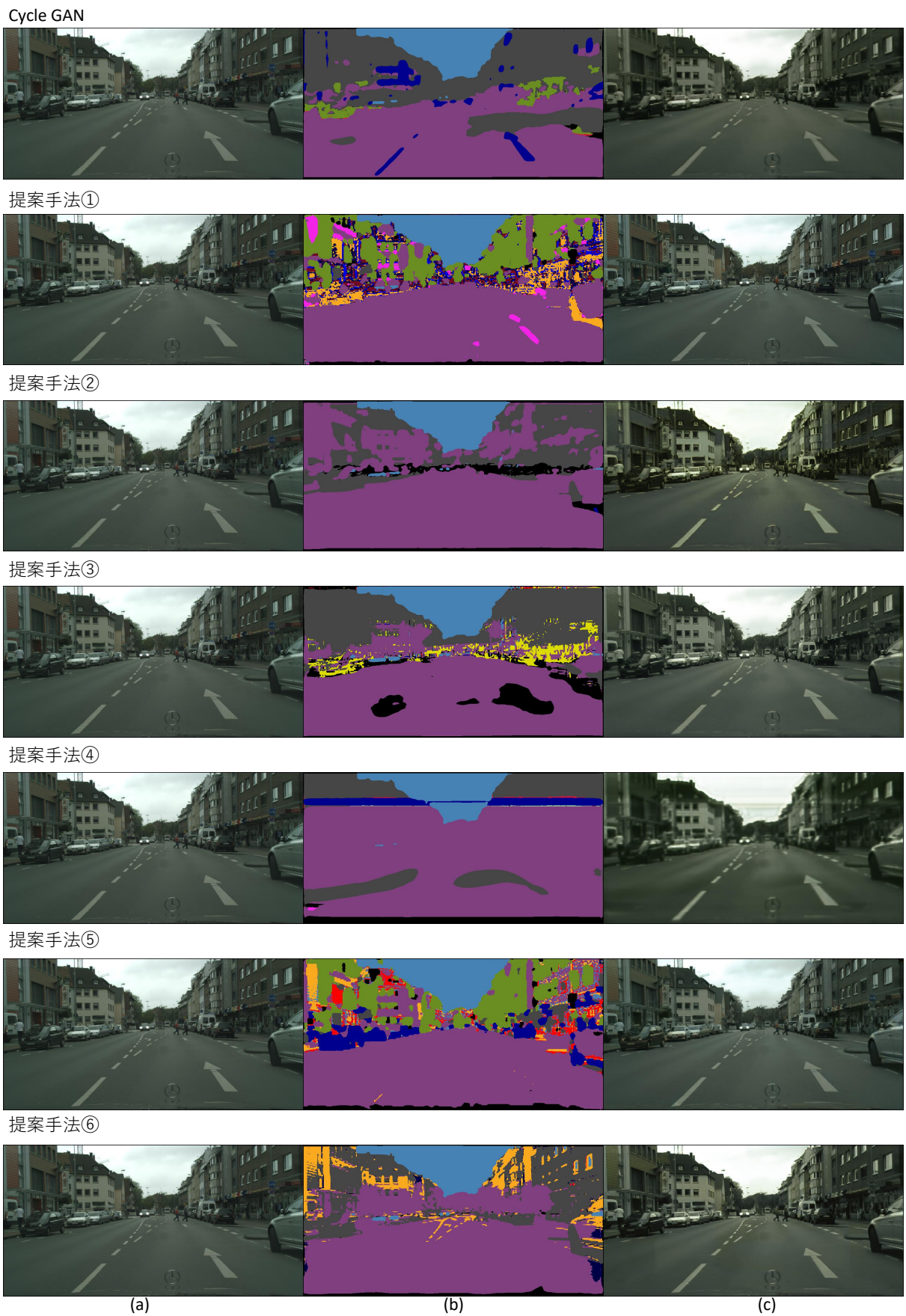

図 B.3: 相互変換結果① (左列 (a): 入力画像，中列 (b):Segmentetion 画像，右列 (c): 再構成画像)

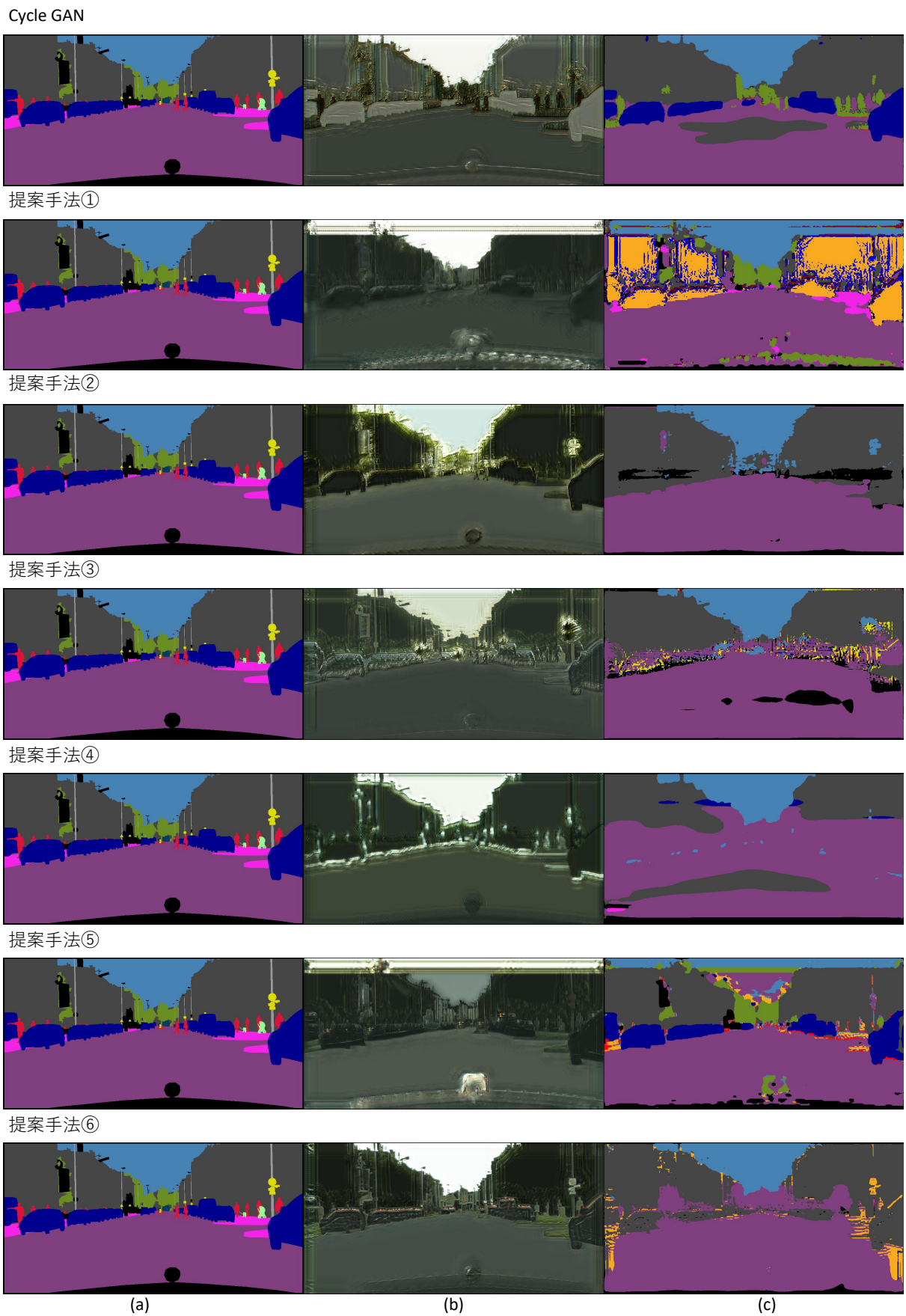

図 B.4: 相互変換結果② (左列 (a): 入力画像，中列 (b):Segmentetion 画像，右列 (c): 再構成画像)

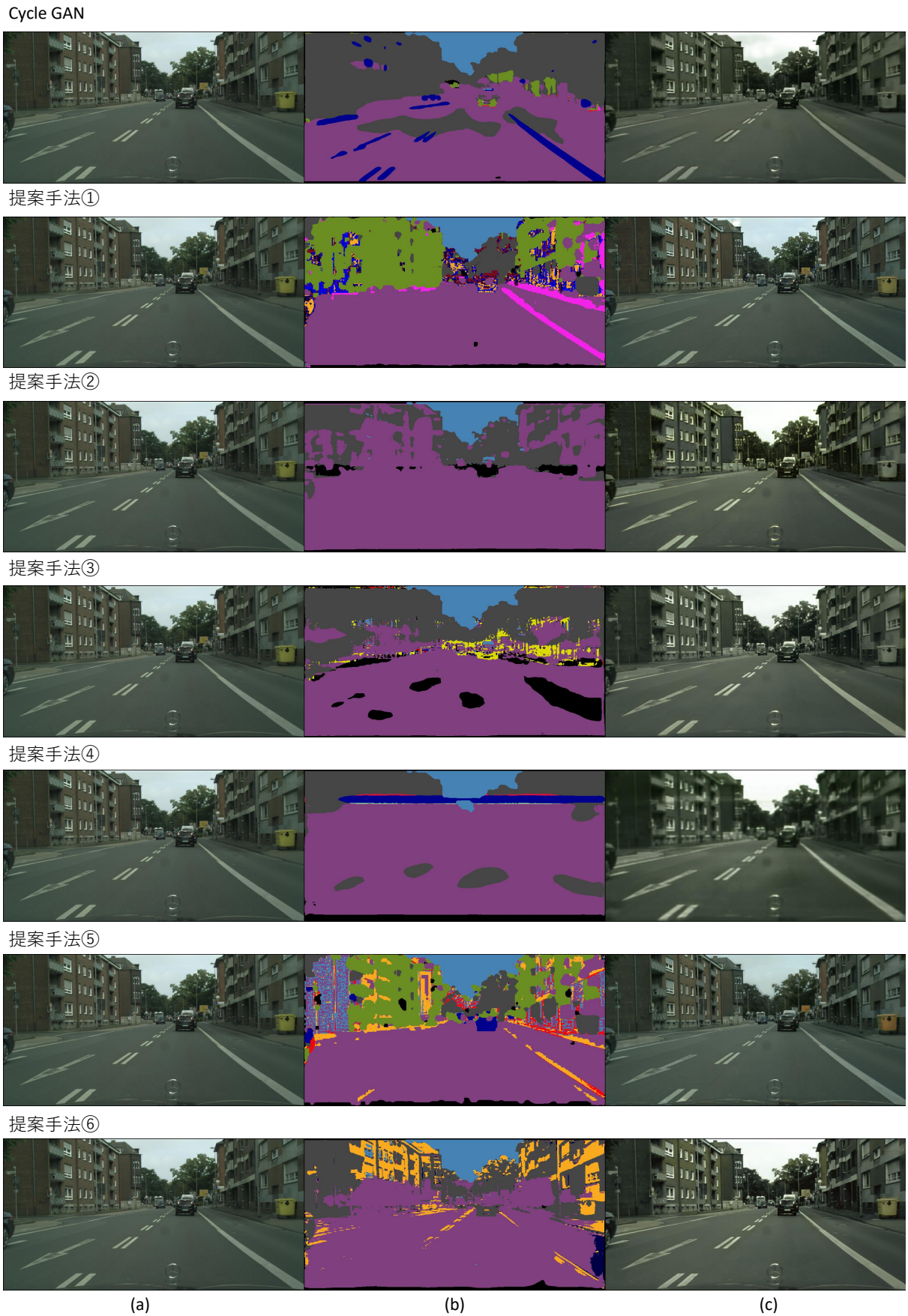

図 B.5: 相互変換結果① (左列 (a): ラベル画像，中列 (b):Desegment 画像，右列 (c): 再構成画像)

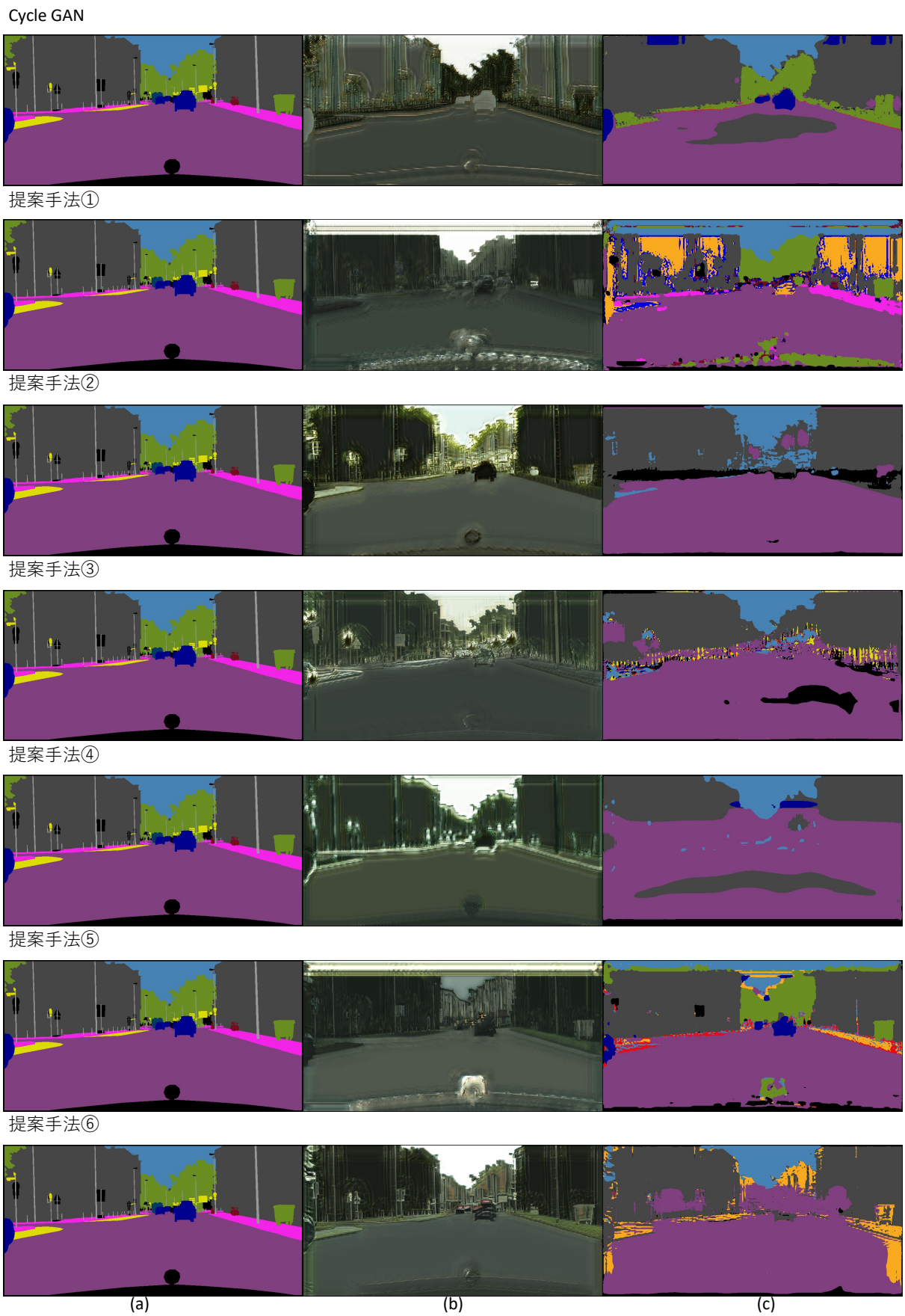

図 B.6: 相互変換結果② (左列 (a): ラベル画像，中列 (b):Desegment 画像，右列 (c): 再構成画像)