博士論文

特徴量変換に基づく
物体識別に関する研究

Dissertation
Doctor of Philosophy

A Study on an Object Classification Employing Features Transformation

中島　佑樹
学籍番号　12584203

指導教員　　タン　ジュークイ　准教授

九州工業大学　大学院　工学府
機械知能工学専攻　知能制御工学コース

平成 30 年度

摘要

近年，ディジタルカメラと計算機は高性能化，低価格化が進んでおり，膨大な画像を用いた研究開発は世界中で盛んに行われている．低解像度画像からの高解像度画像化，画像の部分欠損の補間，画像中から対象物体を検出する物体検出，人の姿勢及び行動認識，ITS(Intelligence Transport System)に利用される道路標識検出，自動車検出，車道検出，歩行者の検出，ロボットビジョンに欠かせない 3 次元計測など，画像を入力として様々な出力を実現する画像処理システムは実際の現場に応用されている．

最も注目されている画像処理技術の一つに画像識別がある．これは画像に付与されたクラスを識別する技術である．従来，人が目で見て観測し，観測された物体が何であるかを識別することの代替となる技術であるため，画像処理の中でも応用の幅が広い．この技術は機械学習の中でも，クラス分類と呼ばれる問題を解くことにより実現され，多くの研究者が様々な画像のクラス分類問題を研究してきた．画像処理におけるクラス分類問題は画素値もしくは画素値から計算される特徴量を入力として機械学習を行い，得られる識別器を用いて，クラスを推定するという処理により実現される．

特徴量は手動で設計する手法と自動で特徴量の計算式を最適化する手法がある．本研究では手動で設計する手法から始めて，特徴量の計算式を自動で最適化する手法までを提案している．

提案法では特徴量の計算式を最適化することにより識別率向上を図る場合，特徴量計算式を，処理を表すノード，入出力を表すエッジを用いたネットワーク FTN(Feature Transform Network)として定義する．ここで，最適化時に用いられる評価関数と評価サンプルに対する識別率との間には相関があるという仮説を立て，これを検証する．検証した結果，両者には相関が見られたため，この評価関数を用いた最適化問題 FTOP(Feature Transform Optimization Problem)を定義し，これを解くことにより識別率の向上を目指す．

本論文では，まず，手動での特徴量設計として HOG の拡張法を述べる．次に，特徴量の計算式の最適化手法として FTN 及び FTOP を提案する．さらに，FTOP を歩行者識別に適用する方法について述べる．ベンチマークデータセットを用いて歩行者識別(2 クラス分類問題)実験を実施し，その結果について述べ，考察を与える．最後に結論を述べる．

目 次

変数リスト

変数名	変数の説明	初出の式
$\boldsymbol{R}$	実数集合	式(3.1)
$\mathbf{Z}$	整数集合	式(4.30)
w	画像横位置	式(2.1)
h	画像縦位置	式(2.1)
$\boldsymbol{I}$	画像	式(3.1)
$\underline{\boldsymbol{I}}$	ブロック正規化後の HOG のヒストグラムの要素	式(3.2)
magnitude	勾配の大きさ	式(2.1)
orientation	勾配の方向	式(2.1)
$\boldsymbol{a}$	HOG のヒストグラムベクトル	式(2.2)
$\bar{b}$	ブロック正規化後の HOG のヒストグラムの要素	式(2.2)
κ	HOG のブロックのインデックス	式(2.2)
ε	分母が 0 にならないための正の実数	式(2.2)
feature	提案法の拡張された HOG 特徴量ベクトル	式(2.3)
block	HOG の正規化ブロックサイズ	式(2.3)
cell	HOG のセルサイズ	式(2.3)
bin	HOG のビン数	式(2.3)
w'	対象矩形領域の横位置を表す一時変数	式(2.3)
h'	対象矩形領域の縦位置を表す一時変数	式(2.3)
ξ	HOG のビンのインデックス	式(2.3)
π	円周率	式(2.3)
Bhattacharyya	*Bhattacharyya* 距離	式(2.4)
d	次元のインデックス	式(2.4)
binId	確率密度関数のビンのインデックス	式(2.4)
W^{+}	Positive クラスの確率密度関数	式(2.4)
W^{-}	Negative クラスの確率密度関数	式(2.4)
m	学習サンプルのインデックス	式(2.5)
$\boldsymbol{M}'$	学習サンプル集合	式(2.7)
$\tilde{\boldsymbol{x}}$	特徴量ベクトル	式(2.6)
label	クラス分類のラベル	式(2.5)
t	AdaBoost, Real-AdaBoost の学習ステップ数	式(2.6)
Dist	AdaBoost, Reak-AdaBoost におけるサンプルの重み分布	式(2.5)
$\underline{\boldsymbol{Y}}$	変換後のテンソル	式(3.1)

W	画像横サイズ	式(3.1)
H	画像縦サイズ	式(3.1)
C	チャンネル数	式(3.1)
W'	変換後のテンソルの横サイズ	式(3.1)
H'	変換後のテンソルの縦サイズ	式(3.1)
C'	変換後のテンソルのチャンネル数	式(3.1)
$width$	畳み込みフィルタの矩形領域の横サイズ	式(3.2)
$height$	畳み込みフィルタの矩形領域の縦サイズ	式(3.2)
c	チャンネル	式(3.2)
$\overline{w}$	対象矩形領域の横位置を表す一時変数	式(3.2)
$\overline{h}$	対象矩形領域の縦位置を表す一時変数	式(3.2)
$width'$	Max フィルタの矩形領域の横サイズ	式(3.2)
$height'$	Max フィルタの矩形領域の縦サイズ	式(3.2)
$\boldsymbol{A}$	畳み込みフィルタの重み行列	式(3.2)
$\boldsymbol{\lambda}$	関数のパラメータ集合	式(3.3)
$\boldsymbol{\Lambda}$	ある関数の出力から関数の入力へのつながりを定義する変数集合	式(3.6)
i, j	ある関数を表すノードのインデックス	式(3.8)
ι^j	ノード j へと接続されるノードの集合	式(3.8)
J	FTN の終端ノード	式(3.8)
D	特徴量の次元数	式(3.9)
$\underline{\boldsymbol{S}}$	関数への入力テンソル	式(3.8)
W_S	関数への入力テンソルの横サイズ	式(3.8)
H_S	関数への入力テンソルの縦サイズ	式(3.8)
C_S	関数への入力テンソルのチャンネル数	式(3.8)
W_J	FTN の終端ノードの入力テンソルの横サイズ	式(3.10)
H_J	FTN の終端ノードの入力テンソルの縦サイズ	式(3.10)
C_J	FTN の終端ノードの入力テンソルのチャンネル数	式(3.10)
G	FTN を表すグラフ	式(3.11)
Φ	エッジの接続関数	式(3.11)
E	FTN のエッジ集合	式(3.11)
V, V'	関数ノードの集合	式(3.11)
m_v	最適化に用いる評価サンプルのインデックス	式(3.12)
M_v	最適化に用いる評価サンプル集合	式(3.13)
m_l	最適化に用いる学習サンプルのインデックス	式(3.12)説明
$\boldsymbol{M}_l$	最適化に用いる学習サンプル集合	式(3.12)説明

$\boldsymbol{M}$	最適化に用いる全サンプル集合	式(3.12)説明
$\boldsymbol{P}$	処理の種類の集合	式(3.16)
p	処理の種類のインデックス	式(3.16)
$Cost_p$	処理の種類 p 毎のコストの上限	式(3.16)
$Cost_{max}$	FTN 全体のコストの上限	式(3.16)
$\boldsymbol{Q}$	ノード間の接続行列（ノード間の接続の可，不可を表す）	式(3.14)
$cost$	関数のコスト	式(3.15)
D_{max}	特徴量の次元数の上限	式(3.17)
$\boldsymbol{z}^{(m_v)}$	サンプル m_v に対する識別器の出力ベクトル	式(3.13)
$z^{(m_v)}$	AdaBoost, Real-AdaBoost のサンプル m_v に対する識別器の出力	式(4.8)
th_d	特徴量ベクトルの d 次元目の要素に対する決定木の分割ルールに用いられる閾値	式(4.5)
n_{label}	決定木のリーフに含まれるラベル $label$ のサンプル数	式(4.6)
M_{label}	ラベル $label$ の全サンプル数	式(4.6)
M_{left}	決定木における左ノードに含まれるサンプル数	式(4.7)
M_{right}	決定木における右ノードに含まれるサンプル数	式(4.7)
$M_{\text{left positive}}$	決定木における左ノードに含まれる Positive サンプル数	式(4.7)
$M_{\text{right positive}}$	決定木における右ノードに含まれる Positive サンプル数	式(4.7)
$M_{\text{left negative}}$	決定木における左ノードに含まれる Negative サンプル数	式(4.7)
$M_{\text{right negative}}$	決定木における右ノードに含まれる Negative サンプル数	式(4.7)
M_{before}	決定木における分割前のノードに含まれるサンプル数	式(4.7)
$Gini$	Gini 係数	式(4.7)
AUC	DET 曲線により得られる面積	式(4.9)
FPR	False Positive per Rate	式(4.9)
MR	Miss Rate	式(4.9)
k	DET 曲線を得る際に用いる閾値のインデックス	式(4.9)
K	DET 曲線を得る際に用いる閾値の個数	式(4.9)
$\boldsymbol{\lambda}_{\text{convertcolor}}$	表色系変換のパラメータの集合	式(4.10)
$\boldsymbol{\lambda}_{\text{arithmetic}}$	四則演算のパラメータの集合	式(4.17)
$\boldsymbol{\lambda}_{\text{norm}}$	ノルムのパラメータの集合	式(4.21)
$\boldsymbol{\lambda}_{\text{orientation}}$	方向のパラメータの集合	式(4.26)

$\boldsymbol{\lambda}_{\text{gradient}}$	勾配のパラメータの集合	式(4.29)
$\boldsymbol{\lambda}_{\text{convolution}}$	畳み込みフィルタのパラメータの集合	式(4.30)
$\boldsymbol{\lambda}_{\text{convolution type}}$	畳み込みフィルタの重みを決定するアルゴリズムの種類を表すパラメータの集合	式(4.30)
λ_{type}	畳み込みフィルタの重みを決定するアルゴリズムの種類を表すパラメータ	式(4.30)
$\boldsymbol{\lambda}_{\text{param}}$	畳み込みフィルタの重みを決定するアルゴリズムが持つパラメータ集合	式(4.30)
$\boldsymbol{\lambda}_{\text{Gaussianparam}}$	Gaussian フィルタのパラメータ集合	式(4.30)
$\boldsymbol{\lambda}_{\text{DoGparam}}$	DoG フィルタのパラメータ集合	式(4.30)
$\boldsymbol{\lambda}_{\text{Gaborparam}}$	Gabor フィルタのパラメータ集合	式(4.30)
$\boldsymbol{\lambda}_{\text{Derivativeparam}}$	微分フィルタのパラメータ集合	式(4.30)
Sigma	Gaussian フィルタ，Gabor フィルタに用いるパラメータ集合	式(4.30)
sigma	Gaussian フィルタ，Gabor フィルタに用いるパラメータ	式(4.30)
***Sigma*1, *Sigma*2**	DoG フィルタに用いるパラメータ集合	式(4.30)
*sigma*1, *sigma*2	DoG フィルタに用いるパラメータ	式(4.30)
τ	DoG フィルタの *sigma*1 と *sigma*2 の比率	式(4.30)
Theta*, *Gamma	Gabor フィルタに用いるパラメータ集合	式(4.30)
theta, gamma	Gabor フィルタに用いるパラメータ	式(4.30)
ii	Integral Image の行列	式(4.37)
SUM	Integral Image 計算のための一時的な行列	式(4.38)
Bin	HOG に用いるビン数の集合	式(4.30)
CellSize	HOG に用いるセルサイズの集合	式(4.42)
BlockSize	HOG に用いるブロックサイズの集合	式(4.42)
Neighbor	周辺画素集合の種類集合	式(4.43)
neighbor	周辺画素集合	式(4.43)
pattern	LBP のパターンのインデックス	式(4.44)
$\underline{\boldsymbol{\Gamma}}$	LBP のヒストグラム計算の入力となるパターン毎の 0 か 1 が格納される一時的なテンソル	式(4.44)
$N_{pattern}$	LBP のパターン数	式(4.44)
E	平均値	式(4.46)

省略語リスト

省略語	正式名称
ITS	Intelligence Transport System
FTN	Feature Transform Network
FTOP	Feature Transform Optimization Problem
DoG	Difference of Gaussian
LBP	Local Binary Patterns
SIFT	Scale Invariant Feature Transform
EOH	Edge of Orientation Histograms
HOG	Histograms of Oriented Gradients
HOF	Histograms of Flow
STPatch	Space-Time Patch
ICA	Independent Component Analysis
CSS	Color Self Similarity
RDSF	Relational Depth Similarity Features
PCA	Principle Component Analysis
ICF	Integrated Channel Features
ULBP	Uniformed LBP
BOK	Bag of Keypoints
GED	Graph Edit Distance
SURF	Speed Up Robust Features
A-KAZE	Accelerated KAZE
E-HOG	Extend HOG
M-HOG	Multiple HOG
P-HOG	Pyramid HOG
CoHOG	Co-occurrence HOG
B-HOG	Binarized HOG
R-HOG	Relational HOG
sp-Cov	Spatial Covariance
NN	Neural Network
MNIST	Mixed National Institute of Standards and Technology database
ILSVRC	ImageNet Large Scale Visual Recognition Competition
CNN	Convolutional Neural Network
R-CNN	Region-CNN

ROI	Region of Interest
YOLO	You Only Look Once
PCN	Part and Context Network
SSD	Single Shot MultiBox Detector
LSTM	Long Short Term Memory
RNN	Recurrent Neural Network
ONNX	Open Neural Network Exchange
NNEF	Neural Network Exchange Format
PDF	Probability Density Function
AUC	Area Under Curve
OpenCV	Open Source Computer Vision Library
DET	Detection Error Trade-off
GPGPU	General Purpose for Graphic Processing Unit
FPGA	Field-Programmable Gate Array
OSS	Open Source Software

図リスト

表リスト

第 1 章　序論

1.1　本研究の学術的背景

人は視覚から様々な情報を得ている．文字を読む．写真の人物が誰かを判断する．映画を見てそれがどんなシーンかを理解する．工場では製品に異常がないかを目視で確認する．スポーツでは上手な選手を見て真似ようとする．車を運転するとき，信号や道路標識，前方の障害物や歩行者が飛び出さないかを見て確認する．これらは，視覚から得られる膨大な光刺激による信号情報を脳が処理し意味のある情報に変換することによって実現される．このプロセスを工学的に実現しようとする分野がコンピュータビジョンである．人が見て理解できることと同等のことが工学的に実現されれば，その応用は広い．

視覚に関して工学的にアプローチする場合，入力センサとしてはカメラが代表的である．ディジタルカメラから取得された画像を計算機に入力し，処理させる技術をディジタル画像処理と呼ぶ．以降，単にディジタル画像処理のことを画像処理と呼ぶ．画像処理の研究において用いられるカメラは多岐にわたる．一般的なカメラに加えて, 赤外線カメラ[1,2], 距離カメラ[59,60,63,64], 全方位カメラ[3,4]が用いられる．また，カメラだけではなくレーザレンジファインダなどの異なるセンサと併用(センサフュージョン)する研究もある[5,6].

このように研究開発が盛んである背景として，ディジタルカメラと計算機の高度化，低価格化が挙げられる．その他にも，画像処理の研究開発に役立つプログラミング言語，プログラミングライブラリ，ソフトウェア，ハードウェア，クラウドサービスの登場が挙げられる(図 1.1).

画像処理の研究の中でも，近年もっとも研究が盛んな分野の一つが画像識別である．画像識別は画像に付与されたラベルを自動で識別する技術である．画像識別は画像処理の中でも応用の幅が広い．なぜなら，人が目で見た物体が何であるかを識別することの代替となる技術であるためである．

特に，ITS(Intelligent Transport System)や自動運転車は画像識別の応用として身近なものになりつつある．警察庁交通局[7]の調べによれば，平成 21 年度年間の死亡交通事故件数は 4914 件，歩行中の死亡者数は 1717 人にのぼった．このような背景を受けて，車両にセンサを搭載し，道路交通環境を安全かつ円滑にするためのシステムを構築しようとする動きが活発化してきたのである．中島ら[111-113]は，ITS の開発を目的として研究を行ってきた．ITS に導入される画像処理技術としては，車両検出[54]，道路レーン検出[17,45]，道路標識認識[33,44]など数多くの技術が開発されている．特に自動で人物や歩行者を検出する技術は，歩行者や運転者の安全を確保ための技術として重要視されて入る．そのため，その

図 1.1　画像処理の研究開発を支える技術.

開発と実用化は，画像識別の研究に携わるものにとって喫緊の課題である.

1.2　従来研究の動向と位置づけ

歩行者の検出に限らず, 画像から対象物体の位置と大きさを検出する技術では, 対象領域が属するラベルを識別する画像識別と呼ばれる技術が必要となる．画像識別は特徴量変換と識別器とで構成される．特徴量変換では，各画素に格納され

た値を変換する．機械学習では，特徴量を入力として識別器を構築する．

これまで，画像識別の研究では様々な特徴量の計算式が提案されてきた（表 1.1）. 特徴量の設計方針は手動による特徴量設計と NN による設計に分けられる．まず，手動による特徴量設計について述べる．

手動による特徴量設計の方針はさらに 3 つに分けることができる．

(1) 新規特徴量設計 : 自然現象や現実のアナロジーから識別の対象に有効と考えられる特徴量を設計する方法．また，既存の問題点を解決する新しいアルゴリズムを考案する方法

(2) 既存特徴量拡張：既存の特徴量の計算式を拡張(ハイパーパラメータを複数使用，式の一部を変更)する方法

(3) 特徴量組合せ：既存の特徴量を組合せる(異なる特徴量の併用，特徴量計算の前処理及び後処理を入れる)方法

(1)は，表 1.1 において，次の方法が当てはまる．

- Texton[9]：Gaussian フィルタや DoG(Difference of Gaussian)などの複数の畳み込みフィルタの出力値を用いる方法
- エッジの抽出[17,45]：エッジを抽出し，その結果をルールに基づいて判別する方法
- 平均値[12]：画素の平均値を特徴量に用いる方法
- 肌色[13]：肌色領域を特徴として切り出す方法
- LBP(Local Binary Patterns)[14]：注目画素と周辺画素の大小によりパターンを計算する方法
- 前処理としての画像補正[15]：ヒストグラム均一化を画像に施す手法．Haar-Wavelet[83]による畳み込み[18] : Haar-Wavelet を用いた畳み込み計算の出力値を特徴量とする方法
- Gabor フィルタ[84]による表情識別[20]：視覚の神経細胞が特定の角度に反応する現象をモデル化した Gabor フィルタを用いる方法
- Haar-like[22]：顔の画像が眉毛，目，唇と肌の部分とで明暗が異なる場合が多いという事実を用いた方法
- Motion Filter[24]：人の動きを捉える方法
- SIFT(Scale Invariant Feature Transform)[25,34,26,28,62]：回転やスケールに不変な特徴を得る方法
- EOH(Edge of Orientation Histograms)[27,37] : 方向付けした勾配方向別のヒストグラムを 2 つの方向の比率として表す方法
- HOG(Histograms of Oriented Gradients) [29,47,54,55,56]：輝度勾配の方向毎に輝度勾配の強度を足し込んで得られるヒストグラムを計算する方法
- HOF(Histogram of Flow)[32]：局所領域の動きをヒストグラム化する方法

- Edgelet[33]：連続するエッジを特徴量とする方法
- 前景抽出[38,40,64]：前景と背景を分離することにより得られる特徴量を用いる方法
- STPatch(Space-Time Patch)[39]：画素毎の動きによる状態遷移を用いる方法
- Covariance[41]：あるパッチにおける特徴量の共分散を用いる方法
- ICA(Independent Component Analysis)[57]: 画像の独立成分を特徴量とする方法
- CSS(Color Self Similarity)[58]：色の自己相関を表す方法
- RDSF(Relational Depth Similarity Features)[59]: パッチ毎の距離のヒストグラムの自己相関を表す方法
- Sketch Tokens[71]：手書きエッジとの対応を学習させる方法

(2)は，同表において，次の方法が当てはまる．

- 複数の方向と光の当たり方をした画像に対して Texton を適用[21]
- 45° 回転を加えた Haar-like[23]
- HOG の拡張[35,36,49,52,60,68,42,51]（詳細は後述）
- SIFT の拡張[43,70]（詳細は後述）
- Wavelet の結果を PCA(Principle Component Analysis)により次元削減[19]
- Box フィルタによる Texton の近似による高速化[61]
- Hough Forests に共起性を導入した Joint-Hough Forests[66]
- Haar-like の共起性を導入する Joint-Haar-like[31]
- LBP に共起性を導入[65]

これらの拡張には 4 つの特徴が挙げられる. Integral Image[85]を用いた高速化，従来の特徴量の計算のパラメータをひとつに固定するのではなく複数用いることによる高精度化，前処理を施した後に従来の特徴量を計算することによる高精度化，特徴量の共起性を導入することによる高精度化である．

(3)は，同表において，次の方法が当てはまる．

- HOG と微分フィルタを用いた Hough Forests[50]
- HOG，輝度勾配，L*u*v*の線形和を特徴量とする ICF(Integrated Channel Features)[53]
- HOG と LBP を用いる手法[48]
- Gabor と LBP を用いる手法[69]
- sp-Cov(Spatial Covariance)と ULBP(Uniformed LBP)を用いる手法[74]

これらは，特性の異なる特徴量を複数用いることにより，互いの特徴量の欠点を補間しあうことが期待される手法である．

表 **1.1**　画像識別の従来研究.（続く）

	手動による特徴量設計			NN による畳み込みフィルタの重み最適化	NN の自動設計
西暦	新規(1)	拡張(2)	組合せ(3)		
1980				Neocognitron（文字）[8]	
1981	Texton（テクスチャ）[9]				
1990				CNN（文字）[10]	
1994	平均値, マハラノビス（顔）[12]				
1996	肌色モデル（顔）[13]				
	LBP（テクスチャ）[14]				
1997	画像補正（顔）[15]				
1998	edge に基づくルール（交通）[17]			LeNet（文字）[16]	
1999	Haar-Wavelet（人）[18]				
2000		PCA-Wavelet（一般物体）[19]			
2001	Gabor（顔）[20]	Texton（テクスチャ）[21]			
	Haar-like（顔）[22]				
2002		回転を考慮した Haar-like（顔）[23]			
2003	Motion Filter（人）[24]				
2004	SIFT（一般物体）[25]				
	SIFT，BOK（一般物体）[26]				
	EOH（顔）[27]				

表 **1.1**　画像識別の従来研究（続く）.

	手動による特徴量設計			NN による畳み込みフィルタの重み最適化	NN の自動設計
西暦	新規(1)	拡張(2)	組合せ(3)		
2005	SIFT，BOK（一般物体）[28]	Joint-Haar-like（顔）[31]			
	HOG（人）[29]				
	積分ヒストグラム（人）[30]				
2006	HOF（人）[32]				
	Edgelet（人）[33]				
2007	SIFT（交通）[34]	E-HOG（人）[35]			
	EOH（人）[37]	P-HOG（人）[36]			
	背景差分（人）[38]				
	STPatch（人）[39]				
	シルエットテンプレート（人）[40]				
	Covariance（人）[41]				
2008		SURF（一般物体）[43] PCA，HOG（人）[42]			
			前景尤度，Covariance（人）[44]		
2009	HOG（人）[47]		PCA-SIFT（交通）[46]		
	CoHOG（人）[49]		HOG，LBP（人）[48]		
	HOG（人）[54]		HOG,微分，HoughForests（人）[50]		
	HOG（人）[55]	Joint-HOG（人）[51]			
	HOG（交通）[56]	Color-HOG（人）[52]			
	edge に基づくルール（交通）[45]		HOG，ICF（人）[53]		

表 1.1　画像識別の従来研究.

	手動による特徴量設計			NN による畳み込みフィルタの重み最適化	NN の自動設計
西暦	新規(1)	拡張(2)	組合せ(3)		
2010	Saliency，ICA（一般物体）[57]	距離，HOG（人）[60]			
	CSS（人）[58]	人マスク，HOG（人）[111]			
	RDSF（人）[59]				
2011	SIFT，GED（一般物体）[62]	近似 Texton（人）[61] 共起 LBP（テクスチャ）[65]			
	距離差分（人）[63]	Joint-Hough Forests（交通）[66]			
	距離，Chamfer（人）[64]		M-HOG，Hue（人）[113]		
2012		B-HOG，R-HOG（人）[68]		AlexNet（一般物体）[67]	
2013	SketchTokens（人）[71]	A-Kaze（一般物体）[70]	Gabor，LBP（顔）[69]		
2014			Spatial-Covariance，ULBP（人）[74]		
2015				VGG（一般物体）[72]	
				R-CNN（一般物体）[73]	
				GoogLeNet（一般物体）[75]	
				YOLO（一般物体）[76]	
2016				Fused-CNN，SSD（人）[77]	強化学習（一般物体）[79]
				ResNet（一般物体）[78]	
2017				PCN（人）[80]	
2018				YOLO v3（一般物体）[81]	強化学習（一般物体）[82]

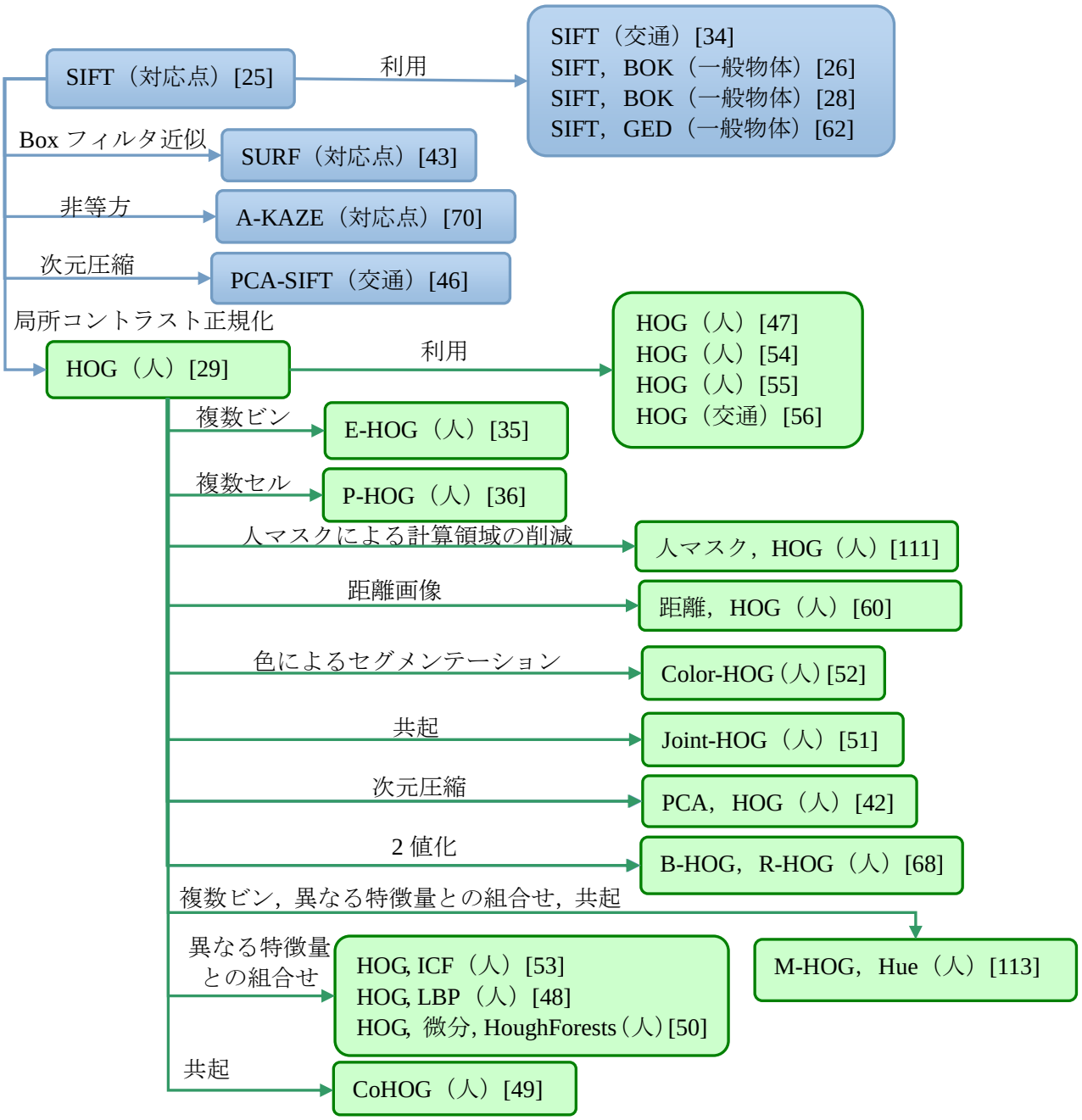


図 1.2　SIFT と HOG の利用と拡張.

ここで，SIFT と HOG を例に(1)～(3)について詳述する．図 1.2 に SIFT と HOG の拡張について整理した図を示す．SIFT では対応点を取得するために特徴点の抽出と特徴量（記述子）の算出法が提案されている．SIFT では対応点における課題である回転と，スケール変化による特徴量の変化を抑制するために工夫されている．キーとなる方向を計算し，キー方向を基準として相対的に特徴量を記述することにより回転不変を実現する．また，DoG の出力値の極値が取得されるまでス

ケールを変化させることにより，スケール不変性のある特徴量を得ることができる．SIFT は課題に対する解を新規に設計した(1)の好例である．この特徴により[26,28,34,62]のように SIFT は様々な目的に用いられた．[34]では道路標識の識別に，[26,28]では BOK(Bag of Keypoints，Bag of Features，Bag of Visual Words とも)を用いた一般物体認識に，また[62]では GED(Graph Edit Distance)を用いた一般物体認識に SIFT が用いられている．SIFT が計算時間に課題があるため，DoG の計算を Box フィルタの近似により高速化した SURF(Speed Up Robust Features)[43]が提案されている．また，SIFT に用いられる Gaussian の等方性が特徴点の類似度を計算するために必要なエッジをぼかしてしまうという課題を，非等方性を持つフィルタを利用して解決した A-KAZE(Accelerated KAZE)[70]が提案されている．これらは，新規に提案された特徴量が持つ課題を克服する設計方針(2)にあたる．

SIFT の特徴量では明暗に対する頑健性のために輝度勾配方向に輝度勾配強度を足し込むヒストグラムが提案されている．SIFT におけるヒストグラム計算に着目し提案された特徴量が HOG[29]である．HOG では，SIFT のヒストグラムに対して，局所領域のコントラストの変化に頑健性をもたせるためにブロックによる正規化が用いられている．HOG は対象の形を捉える特徴として人検出や車両の検出に用いられている[47,54-56]．また，HOG は積分ヒストグラム[30]により高速化することができる．

HOG においても(2)の方針に従って，様々な HOG の拡張が提案されている[35,36,42,49,52,60,68,111]．HOG[29]は単一の輝度勾配方向のビン数を用いるが，E-HOG(Extend-HOG)[35]は複数のビン数を用いている．M-HOG[113]では E-HOG に対して高速に計算可能な複数ビンの条件を明らかにしている．P-HOG(Pyramid-HOG)[36]は複数のセルサイズに対して HOG を計算している．CoHOG(Co-occurrence HOG)[49]は複数の輝度勾配方向に対する共起性を導入している．CoHOG は計算の考え方は HOG の拡張といえるが特徴量の計算式は HOG とは異なるため新規の特徴量とも捉えられる．前処理を工夫した後に HOG を計算する方法には以下がある．距離の計算後に HOG を計算する手法[60]，画素に対して色によるクラスタリングを施した後に HOG を計算する Color-HOG[52]，人マスクを計算した後に HOG を計算する手法[111]である．HOG に対する後処理としては以下がある．HOG 特徴量を計算した後の Real-AdaBoost[110]の出力に対して共起性を与える Joint-HOG[51]，HOG に対して PCA により次元圧縮する手法[42]，B-HOG(Binarized HOG)では閾値により，R-HOG(Relational HOG)は 2 つの方向の比較により HOG を 2 値化する手法[68]である．さらに HOG は(3)の方針においても様々に利用されている[48,50,51,53]．

(3)の方針に従って提案された HOG の拡張は以下の通りである[48,50,53,113]．HOG と LBP を用いる手法[48]，HOG と微分フィルタを Hough Forests に用いる手

法[50]，HOG，輝度勾配，L*u*v*の線形和を特徴量とする ICF[53]，M-HOG と HSV の Hue を用いる手法[113]が該当する．このように特徴量の計算式は手動により設計され改善されてきた．

一方，特徴量を自動で最適化する試みとして，ANN(Artificial Neural Network，以降単に Neural Network と記述)を用いた研究が報告されている．Neocognitron[8]は S 細胞（画像の局所領域における畳み込みの結果を出力する中間層）と C 細胞（出力層）により成る．S 細胞の出力は次の S 細胞に入力され階層化されるが，このとき複数の S 細胞の出力のうち最大値のみが次の S 細胞に入力される．これにより，Texton とは異なり，畳み込みの重みが手動によらずに最適化される．この考え方を応用したものが CNN(Convolutional NN)[10]である．Neocognitron の S 細胞を実現するために畳み込み層を用いる．また，Neocognitrion の S 細胞の出力のうち最大値のみを通過させる機構にはプーリング層を用いる．そして，畳み込み層とプーリング層を繰り返す．最後に C 細胞を，出力層への全結合を用いて実現している．CNN は LeNet[16]に用いられ手書き数字(MNIST[86])の識別に適用されている．しかし，2012 年になるまで，画像識別の主流は前述したような手動により特徴量を設計する手法であった．

2012 年，ILSVRC(ImageNet Large Scale Visual Recognition Competition)[87]という一般物体認識のベンチマークで好成績を収めた CNN(Convolutional NN)を用いた手法 AlexNet[67]が発表された．AlexNet では，Dropout，プーリングのオーバーラップ，Group Convolution と呼ばれる畳み込みにより，得られる複数のチャネルを分割する手法が採用されている．これらの工夫によって従来より層が多い（深いと表現される）NN の学習を成功させている．ILSVRC の一般物体認識のデータセット ImageNet[88]では，CNN の拡張として AlexNet，VGG[72]，GoogLeNet[75]，ResNet[78]が提案され好成績を収めている．これらは NN のネットワーク構造（畳み込み層のフィルタサイズ，プーリング，活性化関数）が異なっている．それが識別率に寄与している．2012 年以降，一般物体認識では，手動での特徴量設計から NN のネットワーク構造の設計へと，特徴量の設計方針が遷移している．

CNN は層が深くなるほど 1 回の画像識別に処理時間がかかる．従来の物体検出はスライディングウィンドウと呼ばれる矩形領域を走査しながら，矩形領域に対して画像識別を実行する．そのため CNN を用いた手法では物体検出に処理時間がかかる傾向にある．そこで，CNN を用いた R-CNN(Region-CNN)[73]は，ROI(Region of Interest)を検出する Selective Search[89]と画像識別を行う NN を用いた．走査を行わずに Selective Search により ROI を抽出することにより，画像識別の実行回数を減らしている．さらに，CNN を用いた手法では，画像識別というタスクのみではなく，対象物体の矩形領域も同時に推定する手法が提案されている[76,77,80,82]．YOLO(You Only Look Once)[76]は矩形領域の推定と画像識別とを同

時に行う NN である．R-CNN に対して，画像の入力から矩形領域の抽出とラベルの推定までをまとめてで実行する End-to-End の学習を YOLO では実現している．これにより物体検出の高精度化と高速化が図られている．YOLO は 2018 年現在 ver.3[81]となり，さらなる高精度化と高速化を実現している．

一般物体認識だけではなく歩行者検出にも CNN の手法が提案されている[77,80]．歩行者検出のベンチマークの一つである INRIA Person Dataset[90]では，2014 年に sp-Cov と ULBP を用いる手法[74]が最も精度が良かった．しかし，2016 年に Fused-CNN[77]が最も良い精度を得ている．さらに，2017 年に PCN(Part and Context Network)[80]と呼ばれる手法が同等の精度を実現している．Fused-CNN では SSD(Single Shot MultiBox Detector)と呼ばれる手法が用いられている．これは YOLO[76]では固定幅のグリッドに対して最大 2 つのラベルと矩形領域を推定する．そして，NN をピラミッド化することにより異なる幅のグリッドに対して複数のラベルと矩形領域を推定する手法である．また PCN は，CNN と LSTM(Long Short Term Memory)[91]と呼ばれる構造を用いた RNN(Recurrent NN)[92]を併用することにより，隠れ部位を考慮した NN を実現している．

手動による特徴量設計から NN のネットワーク構造の設計へと設計方針が遷移してきた．さらに，近年では NN のネットワーク構造を自動で最適化する手法が提案されている[79,82]．これらは強化学習[93]を用いて RNN を学習することにより最適な CNN を獲得する手法[79]である．[79]により得られたネットワーク構造を大規模なデータセット CIFER-10 に用いても有効であることが報告されている[82]．[82]で用いられたネットワーク構造は NASNet と呼ばれている．

1.3 本研究の位置づけと意義

本研究は，画像処理による歩行者識別の識別精度向上のため，特徴量の計算式の設計に関する研究と位置づける．本研究ではまず，特徴量を手動で設計する手法を提案する．次に特徴量を自動設計する手法を提案する．

1.3.1 特徴量の手動設計

手動による特徴量設計の方針の(2)に従い，既存特徴量の拡張による識別精度向上を試みる．1.2 にて述べたように，HOG[29]は様々な手法に拡張されている．しかし, 特徴量のハイパーパラメータに制限がある. そこで, 本研究では複数の HOG のハイパーパラメータに自由度を与えることを提案している. HOG の拡張に関する提案は 2 章で詳述する．

1.3.2 特徴量の自動設計

歩行者識別は手動での特徴量設計による手法[74]が歩行者識別のベンチマーク[90]において最も良い精度を実現していた．しかし，Fused-CNN[77]の提案による歩行者識別においても NN を用いた特徴量設計が最も良い結果を残している．AlexNet[67]の発表以降，NN のネットワーク構成に関する研究が進められていく中，NN の構造自体を自動構成する手法が期待された．そして 2016 年に強化学習による NN の設計[79]が発表された．

しかし，NN を用いた特徴量の自動設計には 2 つの問題がある．

- [問題 1] NN に用いることができる関数は微分可能でなければならない
- [問題 2] CNN を用いた画像識別の結果を故意に操作する手法が確立されている

[問題 1]について述べる．NN を用いる手法では勾配法を用いたバックプロパゲーションによる重みの最適化が可能であることを前提にしている．勾配法を用いたバックプロパゲーションは対象の関数が微分可能でなければならない．したがって，NN を対象とした特徴量設計の自動化手法[79]では sp-Cov や ULBP のように，従来研究において有効であることが示されている特徴量の計算式が，微分可能でないため最適化の対象にすることができない．畳み込みフィルタである Texton のみを用いる手法[61]や一部の CNN の手法[98]よりも sp-Cov や ULBP を用いる手法[74]のほうが識別精度が良いことが報告されている．このことから，特徴量の計算式の表現として，CNN で用いられる畳み込みフィルタと Max フィルタ以外の特徴量の計算式も含めて最適化できる枠組みがあれば，識別精度を高められることが期待できる．しかし，現状ではこれらを扱うための枠組みがない．この問題は本研究における重要問題であるため，3.1 問題提起にて詳述する．

[問題 2]について述べる．CNN を用いること自体がセキュリティの面から問題になるという報告[94,95]がある．One Pixel Attack[94]と呼ばれる手法は，1 ピクセルの画素値を変化させるだけで故意に誤った識別結果を導くことができる．One Pixel Attack[90]は 2 次元の画像に対する修正である．しかし，[95]によれば，人が見れば一見亀に見える 3 次元物体であるにも関わらず，ライフルであると故意に識別させることが可能である．これを個人認証に悪用した場合，周りから見て不信な行動を見せずに他人として認証させることができる．現在，CNN を含む NN を用いたシステム開発では OSS(Open Source Software)が用いられている．また，NN のモデルファイルが共有されている．従って，上記の NN は誰でも利用できる. さらに NN を記述するフォーマット ONNX(Open Neural Network Exchange)[96], NNEF (Neural Network Exchange Format)[97]の整備が進められている．このように開発技術を公にし整備していくことはその技術の普及には欠かせない．他方で悪用に対しても容易に広めてしまう危険性をはらんでいる．これらの点からも NN

以外の手法の研究を進めることは意義があると考えられる．

1.4　本研究の学術的な特色と独創点

本研究では，NN を用いずに特徴量の計算式を最適化する手法を提案する．特徴量の計算式をグラフ理論のネットワークとして捉える．ネットワークは処理を実行するノードとノードに対する入出力関係を表すエッジからなる．提案法ではこのネットワークを FTN(Feature Transform Network)と呼ぶ．FTN に対してある種の操作を実行することにより特徴量の構造を変化させる．本研究では，最適化する対象を FTN とし，画像識別の汎化性能を向上させるための組合せ最適化問題 FTOP(Feature Transform Optimization Problem)を提案する．FTOP は特徴量の計算式を最適化するための手法であり，歩行者識別や画像識別に限定されない．

本研究の独創点は『Neural Network を用いない特徴量の計算式の最適化手法である』ことである．これにより微分可能でない関数を特徴量の計算式に含めながら最適化が可能である．また，実際のアプリケーションに合わせて FTOP を解くのに以下３点の特異な機能を有している．

- [機能 1] 機械学習の計算環境によって学習可能な特徴量ベクトルの次元数の上限は決まる．これには特徴量ベクトルの次元数を指定することにより対応できる
- [機能 2] 画像識別が用いられるアプリケーションによって許容される特徴量の計算時間が決まる．これにはノードに使用する関数の処理時間をコストとして，FTN 全体のコストに対する上限を指定することにより対応できる
- [機能 3] FTN 全体を最適化することも可能であるが，最適化処理時間の短縮のために，従来研究のノウハウを利用して，関数と関数の接続関係を制限できる

本研究の対象である歩行者識別に FTOP を適用させる手法を提案する．歩行者識別に FTOP を適用するために，ノードに用いる処理の種類の選定，汎化性能を向上させるための評価関数の設計を示す．また，最適化の処理時間の問題により最適化問題を分割するという工夫が必要となる場合の分割方法について示す．

提案法の有効性を示すための実験として，FTOP を求解することが歩行者識別の汎化性能を向上させることに寄与するかを確認する実験と従来法との比較実験を実施する．FTOP の求解における評価関数と汎化性能との相関を確認したところ正の相関が見られた．また，従来法との比較実験においても良好な結果が得られている．

1.5　本論文の構成

本論文の構成は以下の通りである．２章にて研究初期に実施した手動による特徴量設計の研究である HOG 特徴量の拡張法について述べる．３章にて特徴量の計算式の最適化手法である FTN と FTOP について述べ，FTOP を求解することが歩行者識別に有効であることを確認する仮説検証実験を行った結果，有効であることを示す．２章から３章にかけて，手動で特徴量の設計する研究から自動化する手法へとシフトしている．４章では FTOP を歩行者識別に適用する手法について述べ，従来法との比較実験を行った結果，従来法よりも良い結果が得られたことを示す．最後に５章にて結論を述べる．また，付録に４章の実験にて得られた FTN と FTN により計算される特徴量の可視化画像を図 A.2，図 A.3 にそれぞれ示す．

第２章 HOG の拡張法

本章では HOG[29]の拡張法について述べる．手動で特徴量を設計する手法として HOG[29]が持つパラメータに自由度を与える．ここで，自由度とはあるパラメータに対して固定１つの値のみ許していたものを，複数の値を許すことをさす．拡張による効果を実験にて示す．

2.1 HOG 拡張法の提案

提案法に用いる HOG は Dalal らの HOG を拡張している．まず, Dalal らの HOG について述べ，その後提案法における HOG の拡張について述べる．

Dalal ら[29]によって提案された HOG 特徴量は輝度勾配方向に輝度勾配強度を加算して得られるヒストグラム特徴量(図 2.1)である．まず，輝度勾配の強度 *magnitude* と方向 *orientation* は式(2.1)により算出される．

$$\begin{aligned} &\Delta w = \boldsymbol{I}_{w-1h} - \boldsymbol{I}_{w+1h},\ \Delta h = \boldsymbol{I}_{wh-1} - \boldsymbol{I}_{wh+1}, \\ &magnitude_{wh} = \sqrt{(\Delta w)^2 + (\Delta h)^2}, \\ &orientation_{wh} = \arctan\left(\frac{\Delta h}{\Delta w}\right). \end{aligned} \tag{2.1}$$

ここで，w は画像横位置，h は画像縦位置，$\boldsymbol{I}$ は輝度値を表す．

次に，画像上にセルを定義し，セル内の輝度勾配の強度 *magnitude* を方向 *orientation* 別に足し込むことにより得られるヒストグラムを計算する．ヒストグラム計算のために *orientation* は分割される．例えば図 2.1 のように 20° 刻みで分割する場合，9 分割のヒストグラムが計算される．これをヒストグラム特徴量ベクトル $\boldsymbol{a} = (a_1, a_2, ..., a_9)$ とする．Dalal ら[29]の提案したブロックによるヒストグラムの正規化法は以下の通りである．例えば図 2.1 に示すように 3×3 セルを 1 ブロックとして 1 セルずつ移動させながら正規化を行うと，セル位置(u,v)における特徴ベクトルを $\boldsymbol{a}_{uv}$ とすれば，κ 番目のブロック $\boldsymbol{b}_\kappa$ は $\boldsymbol{b}_\kappa = (\boldsymbol{a}_{uv}, \boldsymbol{a}_{u+1v}, \boldsymbol{a}_{u+2v}, \boldsymbol{a}_{uv+1}, \boldsymbol{a}_{u+1v+1}, \boldsymbol{a}_{u+2v+1}, \boldsymbol{a}_{uv+2}, \boldsymbol{a}_{u+1v+2}, \boldsymbol{a}_{u+2v+2})_\kappa$ と表すことができ，81 次元の特徴量ベクトルとなる．ここで $\boldsymbol{b}_\kappa$ の要素を b, 正規化後の要素を $\bar{b}$ とすれば,

$$\bar{b} = \frac{a}{\sqrt{\|\boldsymbol{b}_\kappa\|^2 + \varepsilon^2}}. \tag{2.2}$$

となる．ε は分母が 0 にならないための正の実数である．

入力画像を 30×60 ピクセル，1 セルを 5×5 ピクセルとすれば，横方向に 4 ブロ

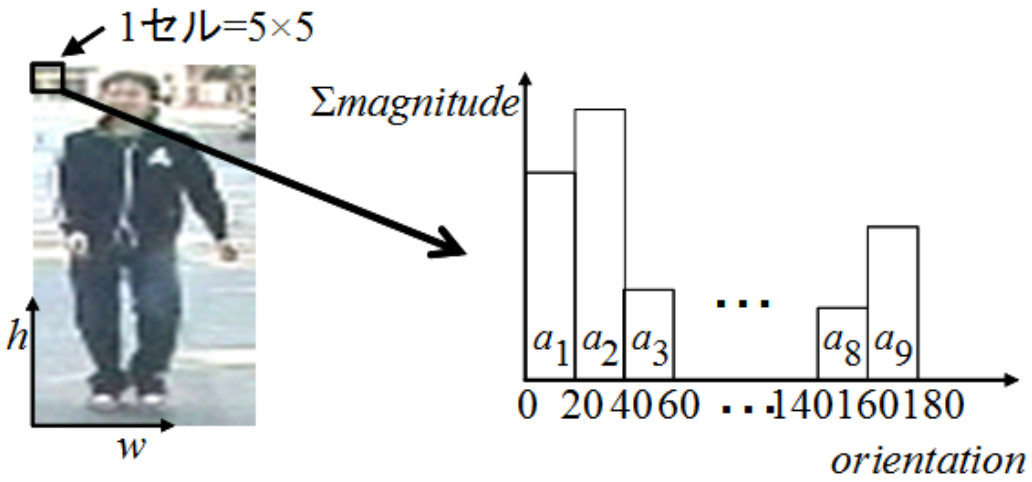


図 2.1　輝度勾配ヒストグラム.

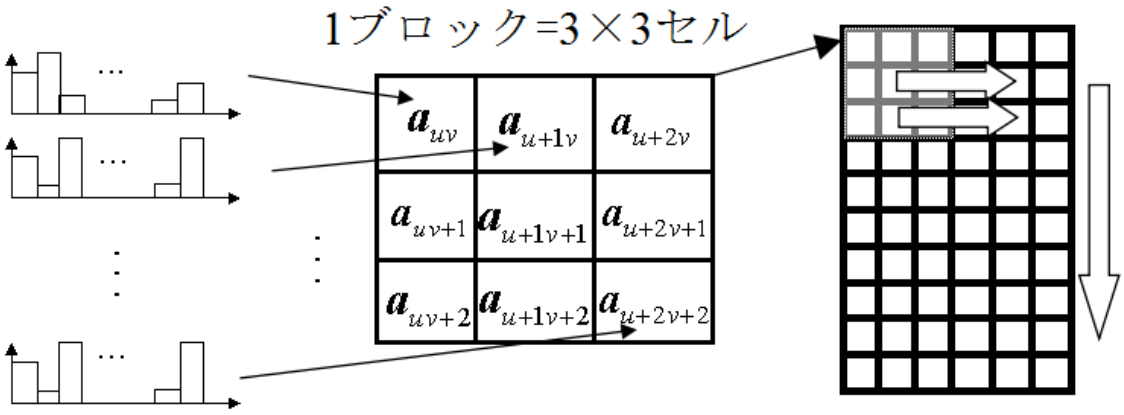


図 2.2　ブロックによる正規化.

ック，縦方向に 10 ブロック，計 40 ブロックに対して正規化を行う．1 ブロックあたり 81 次元を有するため，40 ブロック×81 次元=3240 次元となる．

ここで，従来の HOG 拡張について述べる．E-HOG[35]では勾配方向の分割数を複数許容している．P-HOG[36]は複数のセルサイズおいて HOG を計算している．これらは HOG の計算に用いるパラメータを複数許容する手法である．HOG の前処理として色によるクラスタリングを施す Color-HOG[52]，HOG の後処理として PCA することにより次元削減する手法[42]，HOG 特徴量を計算後に共起性を計算する Joint-HOG[51]は HOG の計算自体には拡張は無い．共起性という考え方は CoHOG[49]にも当てはまるが，HOG の名前が付与されているものの，ヒストグラムの計算式が HOG とは異なる特徴量である．また，B-HOG，R-HOG[68]は精度向上ではなくメモリ削減のための 2 値化手法である．

このように，精度向上のための HOG の拡張はパラメータを複数許容する手法

と HOG の前後処理を工夫する手法に分けられる．ここで，提案法では HOG の拡張においてパラメータを複数許容する手法を用いる．この拡張により識別精度が向上すれば前後処理[42,51,52]による更なる精度向上が期待されるためである．

HOG のパラメータにはセルサイズ(*cellSize*)，ブロックサイズ(*blockSize*)，輝度勾配の方向を分割するビン数(*bin*)が含まれる．ここで，E-HOG と P-HOG はそれぞれ複数のビン数とセルサイズを許容している．しかし，従来法では複数の正規化ブロックのサイズを許容する手法が提案されていない．正規化ブロックは局所領域におけるコントラストの変化に対する頑健性を与えるための工夫であるが，コントラストの違いは局所領域によって異なることが予想される．この場合，単一の正規化ブロックサイズではコントラストの変化を捉えきれないと考えられる．この点において，正規化ブロックサイズを複数許容することに意味があると考えられる．

そこで，E-HOG のビン拡張，P-HOG のセルサイズ拡張に加えて，正規化ブロックのサイズ拡張を加えた HOG の拡張法を提案する．$cell_{wh}$ を(w, h)を中心とした *cellSize* の矩形領域セル，$block_{wh}$ を(w, h)を中心とした *blockSize* の矩形領域ブロック，*bin* をビンの分割数，ξ をビンのインデックスとすれば提案法の HOG は以下の式により計算される．

$$\boldsymbol{feature}_{wh}^{(block,cell,bin)} = \frac{\sum_{(w',h')\in cell_{wh}} magnitude(w',h')}{\sum_{(w',h')\in block_{wh}} magnitude(w',h')}, \quad if\ (\xi-1)\pi/bin < orientation(w',h') < \xi\pi/bin. \tag{2.3}$$

ここで，複数の(*block, cell, bin*)を組合せることにより，提案法の HOG は計算される．Dalal らの HOG[29]と異なり，提案法では HOG のセルをオーバーラップさせる．これにより，位置ずれに対して頑健になると考えられる．図 2.3 に入力画像の R 成分に対する HOG の結果を示す．HOG の bin 数を 5 にした結果である．画像中の各方向に沿って，特徴量が抽出されていることが分かる．

HOG は Integral Histogram[30]を用いることにより高速に計算できる．Integral

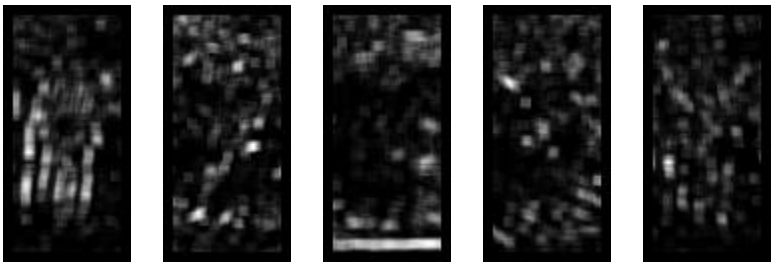


図 2.3　HOG.

Histogram[30]は Integral Image[85]を分割ビン毎に計算することにより実現される．Integral Image は後述する Box フィルタ計算の高速化に用いられるため，4.2.7 Box フィルタにて詳述する．

2.2 RealAdaBoost

提案する HOG 拡張法を歩行者識別に適用するために機械学習である RealAdaBoost[110] を用いる．RealAdaBoost では弱識別器の識別精度を *Bhattacharyya* 距離により評価する．*Bhattacharyya* 距離は式(2.4)により算出される．positive，negative サンプルの PDF をそれぞれ W^+，W^-，BIN(・)を bin 番号への変換関数，*binId* は算出する PDF(Probability Density Function)の bin 番号である．

$$Bhattacharyya_d = \sum_{\forall binId} \sqrt{W^+(binId)W^-(binId)} \tag{2.4}$$

図 2.4 の左図のように識別精度が高い場合は *Bhattacharyya* 距離が小さくなる．従って，*Bhattacharyya* 距離が小さい弱識別器を選べばよい．

サンプル $m \in \boldsymbol{M}'$($\boldsymbol{M}'$は学習サンプル集合)に対して HOG 拡張法により算出された特徴量 ***feature*** を並べたベクトル $\tilde{\boldsymbol{x}}^{(m)}$ を RealAdaBoost に入力する．RealAdaBoost は AdaBoost を弱識別器の出力が実数値となるように改良したアルゴリズムである．AdaBoost では弱識別器を設計する必要があるが，RealAdaBoost では弱識別器は確率密度関数(PDF: Probability Density Function)の導入によって，その出力が実数値となるように設計されるため，特徴量を学習器に入力すれば学習することができる．RealAdaBoost のアルゴリズムを図 2.5 に示す．

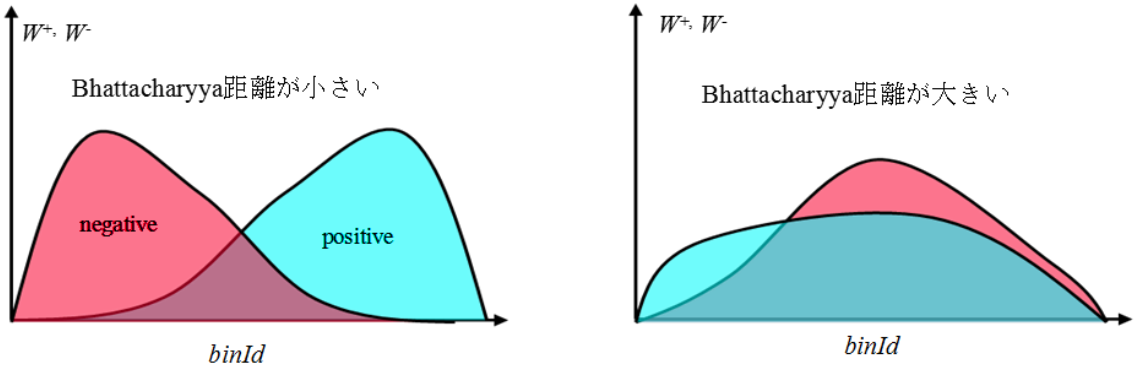


図 2.4　確率密度関数と評価値の関係．

弱識別器の学習

Input: サンプル重み分布 $Dist_t$

Output: 弱識別器 $weak_t$

1: positive，negative サンプルの PDF をそれぞれ W^+，W^-，BIN(・)を bin 番号への変換関数，*binId* を算出する PDF の bin 番号，$\tilde{x}_d^{(m)}$ を特徴量ベクトル $\tilde{\boldsymbol{x}}^{(m)}$ から選択された d 次元目の特徴量，サンプル m のラベルを $label_m$，サンプルの重み分布を *Dist* とすれば，PDF は以下の式により計算される．

$$W^+(binId)=\sum_{BIN(\tilde{x}_d^{(m)})=binId\wedge label_m=1} Dist(m), W^-(binId)=\sum_{BIN(\tilde{x}_d^{(m)})=binId\wedge label_m=-1} Dist(m). \quad (2.5)$$

2: 算出された各次元 d の PDF である W^+，W^-を用いて式(2.4)により評価値 $Bhattacharyya_d$ を算出する．

3: $Bhattacharyya_d$ が最小となる d の特徴量 $\tilde{x}_d$ を用いた弱識別器 $weak_t$ は 0 除算防止の限りなく 0 に近い正の値 ζ を用いて，以下の式により定義される．*BIN* は特徴量から *binId* に変換する関数である．

$$weak_t(\tilde{x}_d)=\frac{1}{2}\ln\frac{W^+(BIN(\tilde{x}_d))+\zeta}{W^-(BIN(\tilde{x}_d))+\zeta} \quad (2.6)$$

4: Return $weak_t$

RealAdaBoost

Input: 弱識別器数 T，確率密度関数の分割数

Output: *strong*

1: サンプル重み $Dist_0(m) \leftarrow 1/|\boldsymbol{M}'|$，$m\in\boldsymbol{M}'$.

2: For t=0; t<弱識別器数 T; t++

3: $weak_t \leftarrow$ 弱識別器の学習($Dist_t$)

4: 次式よりサンプル重み分布 D_t を次式により更新する．

$$Dist_{t+1}(m)\leftarrow\frac{Dist_{t+1}(m)}{\sum_{m\in M'}Dist_{t+1}(m)}=\frac{Dist_t(m)\exp\left(-label_m weak_t\left(\tilde{\boldsymbol{x}}^{(m)}\right)\right)}{\sum_{m\in M'}Dist_t(m)\exp\left(-label_m weak_t\left(\tilde{\boldsymbol{x}}^{(m)}\right)\right)}, \forall m\in M' \quad (2.7)$$

5: End for

6: 強識別器 $strong(\boldsymbol{x})$を以下の式により生成する．

$$strong(\tilde{\boldsymbol{x}})=\mathrm{sign}\left(\sum_{t=1}^{T} weak_t(\tilde{\boldsymbol{x}})\right) \quad (2.8)$$

7: Return *strong*

図 2.5　RealAdaBoost の学習アルゴリズム．

2.3 実験

提案する HOG 拡張法の有効性を確認するために歩行者識別において従来法との比較実験を行った．比較実験には INRIA Person Dataset[90]を用いた．付録の図 A.1 に INRIA Person Dataset のサンプルを示す．実験に用いたデータセットのサンプル数を表 2.1 に示す．INRIA Person Dataset[90]では 64x128[pixel]の画像が提供されているが，実験環境のメモリ上限のため，画像を 30x60[pixel]に縮小して実験している．また，人が含まれていない Negative サンプルは切り取り済の画像が提

表 2.1 INRIA Person Dataset のサンプル数.

学習 Positive サンプル数	2416
学習 Negative サンプル数	6000
評価 Positive サンプル数	1126
評価 Negative サンプル数	3000
解像度 横×縦 [pixel]	30x60

表 2.2 HOG 拡張パラメータ.

手法	拡張パラメータ		
	ビン数	セルサイズ [pixel x pixel]	ブロックサイズ [セル x セル]
拡張なし	9	5x5	3x3
ビン数拡張	1,3,9	5x5	1x1
ビン数+ セルサイズ拡張	1,3,9	5x5,5x10,10x5	1x1
提案法	1,3,9	5x5,5x10,10x5	1x1,2x2,3x3

表 2.3 ソフトウェアの構成.

ランタイム(コンパイラ)	VC++ 11 64bit
HOG	リファレンス実装
RealAdaBoost	リファレンス実装

表 2.4　計算機環境.

OS	Windows7 Pro 64bit
CPU	Intel(R) Core(TM) i7-3770K CPU 3.50GHz
メモリ	32GB

供されていないため，ランダムに切り取ることにより作成している．本実験では表 2.2 に示すように HOG の各パラメータを変更し, 拡張の効果を計測する. なお, RealAdaBoost に用いたビンの分割数は 64，弱識別器数は 1000 である．表 2.3 と表 2.4 に実験に用いた開発環境を示す.

上記の設定にて実験した結果を図 2.6 に示す. 図 2.6 は横軸に弱識別器数を縦軸に誤識別率をとる．図 2.6 から最も誤識別率が小さい弱識別器数 750 のときの識別にかかる処理時間は 34[micro sec]であった．図 2.7(a), (b), (c)には RealAdaBoost 学習における弱識別器追加の繰り返しに対して，それぞれ，選択されたビン数，セルサイズ，ブロックサイズを縦軸に表している.

2.4　考察

図 2.6 からビン数，ビン数+セルサイズ，提案法と拡張するに従って，いずれの弱識別器数の場合にも誤識別率が低下しており，提案法の効果が現れている．また，ビン数の拡張とセルサイズの拡張の実験結果よりも，ブロックサイズの拡張を追加したほうが誤識別率が低下していることから，2.1　HOG 拡張法の提案に

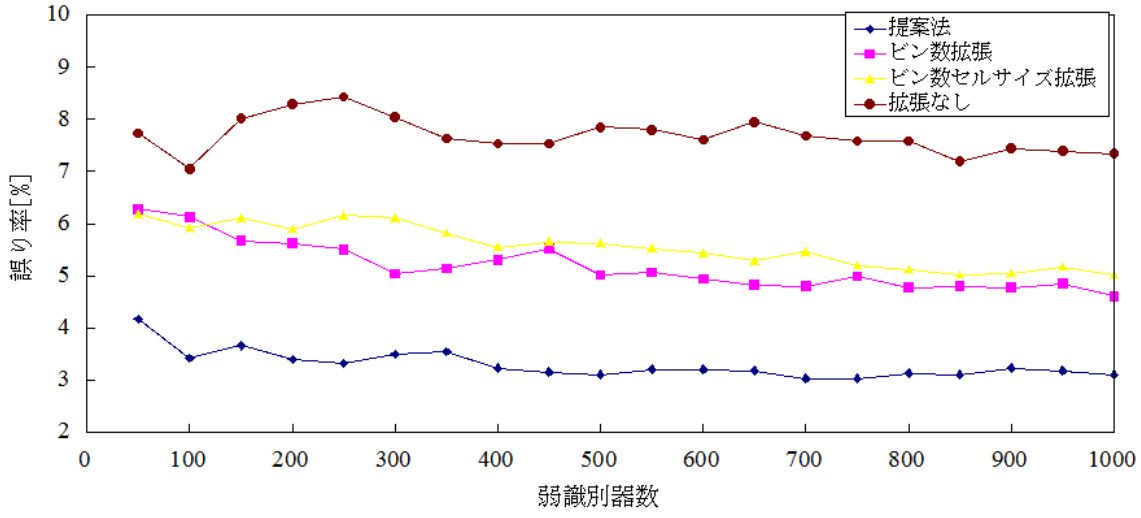
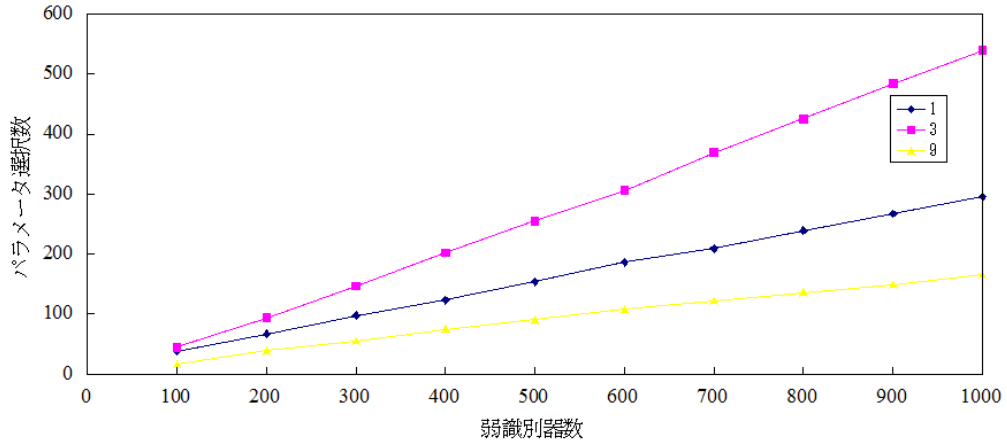
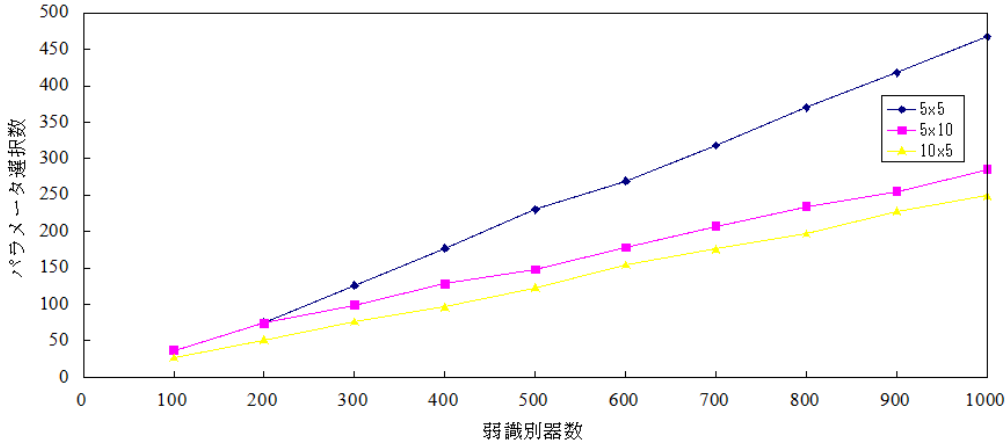


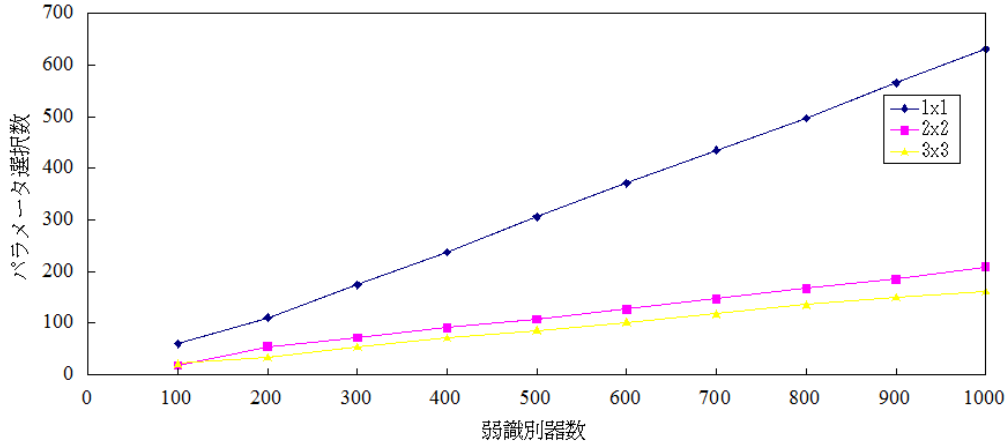
図 2.6　弱識別器数に対する誤識別率.



(a)



(b)



(c)

図 2.7　学習過程における選択されたパラメータ数. (a)ビン数, (b)セルサイズ数, (c)ブロックサイズ数.

て述べたブロックサイズの拡張も追加した方が良いとする主張が正しいことが確認できる．

図 2.7 から弱識別器数の増加に従ってビン数，セルサイズ，ブロックサイズいずれのパラメータも増加していることから，複数のパラメータを用いることにより，パラメータの異なる特徴量が互いに識別精度を補間していると考えられる．もし，補間されない特徴量であるならば，はじめから選ばれない，もしくは，弱識別器数が増えるに従って選択数が変わらなくなるはずであるが，図 2.7 ではいずれのパラメータも増加している．

2.5 まとめ

提案する HOG 拡張法により歩行者識別精度が向上することが判明した．HOG 拡張法では従来法のようにビン数とセルサイズのみではなく，ブロックサイズも複数パラメータを許容することを提案している．実験では、拡張なし、ビン数拡張，ビン数/セルサイズ拡張，ビン数/セルサイズ/ブロックサイズ拡張(提案法)のように拡張されるパラメータの種類を増やす方向に HOG 特徴量のパラメータを設定し，それぞれ比較した．この実験結果によれば，拡張されるパラメータの種類が増えるにつれ歩行者識別精度は向上した．特に，従来提案されていなかったブロックサイズのパラメータを複数許容する拡張を施すことにより，ビン数，セルサイズの拡張に比べて大きく性能改選することが分かった．この知見は後述の FTOP による特徴量の計算式の最適化における HOG においても使用される．

第 3 章 FTOP(Feature Transform Optimization Problem)

本章では提案法である FTOP(Feature Transform Optimization Problem:特徴量変換最適化問題)の定義と求解法について述べる．まず，従来法を考察し，問題提起する．次に，問題体に対する提案法として FTOP を定義する．最後に，FTOP を解くための手法を提案する．

3.1 問題提起

画像識別とは画像に付与されたラベルを推定することである．ラベルは目的に合わせて設定される．例えば歩行者識別であれば，歩行者ラベルと歩行者以外ラベルを付与する．画像識別を実現する枠組みを図 3.1 に示す．識別器の構築過程は以下のとおりである．まず，入力画像から特徴量を計算する．次に，特徴量と画像のラベルを機械学習の学習に入力する．その結果，画像を識別するための識別器が得られる．画像を識別する過程は以下のとおりである．識別器の構築と同様に入力画像から特徴量を計算する．特徴量を構築済の識別器に入力すると，識別器が画像ラベルを出力する．

画像識別の研究では特徴量の計算式と機械学習を工夫することにより識別精度を向上させる試みが報告されてきた. 近年, 特に特徴量の計算式の改善が著しい. ここで，提案法と従来法の違いを明らかにするために特徴量の計算式について考

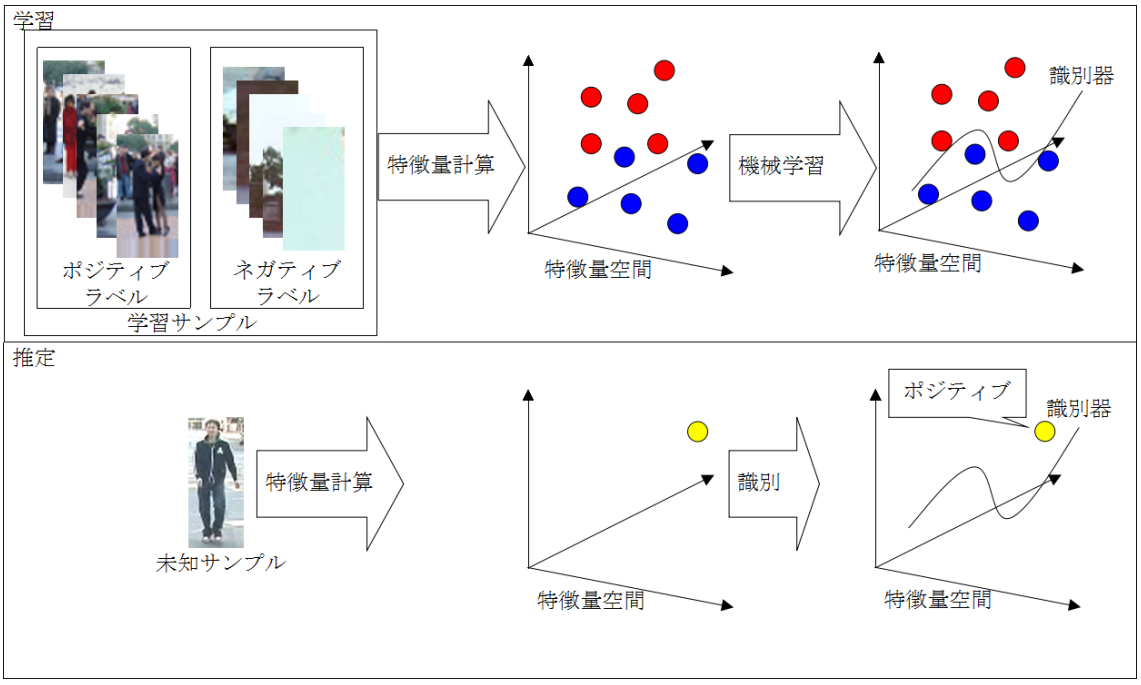


図 3.1　画像識別の流れ.

察する．従来法において，特徴量の計算式が改善されてきた順にその課題と解決の流れを示し，提案法が解決しようとする問題を提起する．

特徴量の計算式 f は以下のように用いられる．

$$\underline{\boldsymbol{Y}} = f\left(\underline{\boldsymbol{I}}\right),\ \underline{\boldsymbol{I}} \in \boldsymbol{R}^{W\times H\times C},\ \underline{\boldsymbol{Y}} \in \boldsymbol{R}^{W'\times H'\times C'} \tag{3.1}$$

ここで，$\boldsymbol{R}$ は実数集合，入力される画像 $\underline{\boldsymbol{I}}$ は画像横サイズ W，画像縦サイズ H，チャンネル C のテンソルとして表現される．例えば 64x128[pixel]の RGB 画像があった場合は W=64，H=128，C=3 である．$\underline{\boldsymbol{I}}$ が特徴量の計算式 f に入力されると，特徴量 $\underline{\boldsymbol{Y}}$ は横サイズ W'，縦サイズ H'，チャンネル C'のテンソルとして出力される．ここで，W と W'，H と H'，C と C'は異なる場合もある．例えば上記の例の $\underline{\boldsymbol{I}}$ に対して，3x3 の畳み込み積分を各チャンネル毎にパディングなしで施す場合は W'=W-2，H'=H-2，C'=3 となる．パディングとは計算に用いられる値が存在しない場合に，代わりの値を用いて計算するための埋め込みである．3x3 の畳み込み積分を $\underline{\boldsymbol{I}}_{c00}$ に対して施す場合，例えば $\underline{\boldsymbol{I}}_{c-1-1}$ の値が必要になるが，これは存在しない．そこで例えば $\underline{\boldsymbol{I}}_{c-1-1}=0$ として畳み込み積分を計算する．$\underline{\boldsymbol{I}}$ のあるチャンネル c=0,1,...,C の横位置 w，縦位置 h における畳み込み積分は以下の式により計算される．

$$\underline{\boldsymbol{Y}}_{cwh} = f\left(\underline{\boldsymbol{I}}_{cwh}\right),\ \forall w \in \{1,2,...,W\}, h \in \{1,2,...,H\},\quad \boldsymbol{A} \in \boldsymbol{R}^{width\times height},$$

$$f_{\text{convolution}}\left(\underline{\boldsymbol{I}}_{cwh}\right) = \sum_{\bar{h}=-height/2}^{height/2} \sum_{\bar{w}=-width/2}^{width/2} \underline{\boldsymbol{I}}_{cw-\bar{w}h-\bar{h}} \boldsymbol{A}_{\bar{w}\bar{h}}\ . \tag{3.2}$$

ただし $width$，$height$，$\boldsymbol{A}$ はそれぞれ畳み込み積分フィルタの横サイズ，縦サイズ，重みである．

ここで，重み $\boldsymbol{A}$ のように特徴量の計算式に含まれるパラメータを $\boldsymbol{\lambda}$ とする．式(3.1)は以下となる．

$$\underline{\boldsymbol{Y}} = f\left(\underline{\boldsymbol{I}}, \boldsymbol{\lambda}\right),\ \boldsymbol{A} \in \boldsymbol{\lambda}\ . \tag{3.3}$$

従来，特徴量の計算式が持つ $\boldsymbol{\lambda}$ は Texton[9]のように事前に値が指定されてきた．Texton[9]は Gaussian フィルタや Box フィルタなどの畳み込み積分によって計算されるフィルタ(以降，畳み込みフィルタ)により構成されている．これらの畳み込みフィルタを用いる時は事前に各重みを指定する必要がある．このように，従来法では特徴量の計算式のパラメータは事前に手動で与えられてきた．しかし，この方法では画像識別の対象に合わせて事前に畳み込みフィルタの重みを設定しなければならない．畳み込みフィルタの種類は無限にあり，それを事前に設定することが困難であった．この考え方は，畳み込みフィルタだけではなく，パラメト

リックな特徴量の計算式全てに当てはまる問題である．

これに対し, CNN[10]のように λ(畳み込みフィルタにおける $\boldsymbol{A}$)を最適化する手法が提案されている．CNN[10]に用いられる畳み込みとプーリングを用いた特徴量の計算式の例を示す．CNN に用いられるプーリング処理は平均プーリング, Max プーリングがあるが, ここでは Max プーリングを例に用いる．Max プーリングは Max フィルタによって実現される．以下に Max フィルタを示す.

$$
\begin{aligned}
&f_{\max}\left(\underline{\boldsymbol{I}}_{cwh},\, width',\, height'\right) = \max\left\{\underline{\boldsymbol{I}}_{cw-\bar{w}h-\bar{h}},\, \forall \bar{w}, \bar{h}\right\},\\
&\bar{w} \in \left\{-width'/2,\, -width'/2+1,...,\, width'/2\right\},\\
&\bar{h} \in \left\{-height'/2,\, -height'/2+1,...,\, height'/2\right\}.
\end{aligned}
\tag{3.4}
$$

ここで $width'$, $height'$はそれぞれ Max フィルタの横サイズ, 縦サイズである．Max フィルタを用いて CNN における畳み込みとプーリングの関係を表すと以下の式のようになる.

$$
\begin{aligned}
&\underline{\boldsymbol{Y}}_{\text{convolution}} = f_{\text{convolution}}\left(\boldsymbol{I}_{c(3)},\, \boldsymbol{A}\right),\\
&\underline{\boldsymbol{Y}}_{\text{CNN}} = f_{\max}\left(\underline{\boldsymbol{Y}}_{\text{convolution}}, width',\, height'\right).
\end{aligned}
\tag{3.5}
$$

ここで, $\underline{\boldsymbol{I}}_{c(3)} \in \boldsymbol{R}^{W\times H}$ は第 3 モード, すなわちチャンネル方向に対して第 c 成分にてスライスした行列である.

CNN では畳み込みフィルタの出力を Max フィルタに入力すること, 及び $width'$, $height'$は事前に与えられる．ここで, 畳み込みの重み $\boldsymbol{A}$ は NN におけるバックプロパゲーションによって最適化される．これにより, 識別対象に最適化された畳み込み処理が実現可能となった．実際の CNN の例では式(3.5)の畳み込みフィルタと Max フィルタを多く組み合わせることにより識別精度の向上が図られている. CNN を用いた従来法では畳み込みフィルタと Max フィルタの入出力の接続, すなわち NN のネットワーク構造 (AlexNet[67], VGG[72], GoogLeNet[75], ResNet[78])が提案されている．このように, 畳み込みの重みを最適化することにより画像識別の対象に合った畳み込みフィルタが自動で取得できるようになった. しかし, この手法ではパラメータの最適化は畳み込みの重みのみであり, 畳み込みフィルタ及び Max フィルタのフィルタサイズは事前に与える必要がある. また, 畳み込みフィルタと Max フィルタの入出力の接続すなわちネットワーク構造も事前に与える必要がある. すなわち, CNN の提案以前の特徴量の計算式におけるパラメータの設定から, CNN におけるネットワーク構造の設定へと事前に設定するパラメータがシフトしたのである.

これに対し, 近年では Learning2Learn(学習のための学習)と呼ばれる考え方が提案されている. 強化学習と RNN を用いた CNN のネットワーク構造を最適化する手法[79]は画像識別のための Learning2Learn である． λに畳み込みフィルタの重

み及びフィルタサイズ，Max フィルタのフィルタサイズを含める．畳み込みフィルタと Max フィルタの入出力を定義するパラメータ $\boldsymbol{\Lambda}$ とすれば，式(3.3)の複数の f の合成関数を一つの関数 F を用いて以下の式により拡張される．

$$\underline{\boldsymbol{Y}} = F\left(\underline{\boldsymbol{I}}, \boldsymbol{\lambda}, \boldsymbol{\Lambda}\right). \tag{3.6}$$

強化学習と RNN を用いた CNN のネットワーク構造を最適化する手法[79]では RNN にて畳み込みフィルタ及び最大値フィルタのフィルタサイズ，畳み込みフィルタと最大値フィルタの適用順を表現し，畳み込みフィルタの重みをバックプロパゲーションにより最適化することを強化学習にて繰り返す．これにより畳み込みの重み $\boldsymbol{A}$ だけではなく，NN のネットワーク構造が識別対象に合わせて最適化される．

しかし，CNN では畳み込みフィルタ，Max フィルタ，活性化関数を最適化の対象にしている．したがって，CNN では sp-Cov や ULBP のように，従来研究において有効であることが示されている特徴量の計算式を全て最適化の対象にすることができない．畳み込みフィルタである Texton のみを用いる手法[61]や一部の CNN の手法[98]よりも sp-Cov や ULBP を用いる手法のほうが識別精度が良いことが報告されていることから，特徴量の計算式の表現として畳み込みフィルタと Max フィルタ以外の特徴量の計算式も含めて最適化できる枠組みがあれば識別精度を高められることが期待される．しかし，現状ではこれらを扱うための枠組みがない．

本論では上記を扱うための枠組みを提案する．提案する枠組みを図 3.2 に示す．提案する枠組みは式(3.6)の関数 F を最適化することが目的である．提案する枠組みでは，特徴量の計算式の候補を生成する．候補の特徴量計算式を用いて計算された特徴量を入力に機械学習する．機械学習により得られた識別器を用いて評価値を計算する．候補を生成しては評価値を計算することを繰り返し，最も良い評価値が得られた特徴量の計算式の候補を最終的な特徴量計算式として採用する．採用した特徴量計算式を用いて計算した特徴量ベクトルと教師ラベルを入力に機械学習することにより最終的な識別器を構築する．以降ではまず，上記の枠組みを扱うためのデータ構造となる FTN(Feature Transform Network)を提案する．これは特徴量の計算式をグラフ理論のネットワークにて表現するものである．次に，FTN を用いた最適化問題を提案する．これを FTOP と呼ぶ．最後に，FTOP を解くためのアルゴリズムを提案する．

3.2 FTN(Feature Transform Network)

ここでは，特徴量の計算式をネットワークにより表現した FTN について述べ

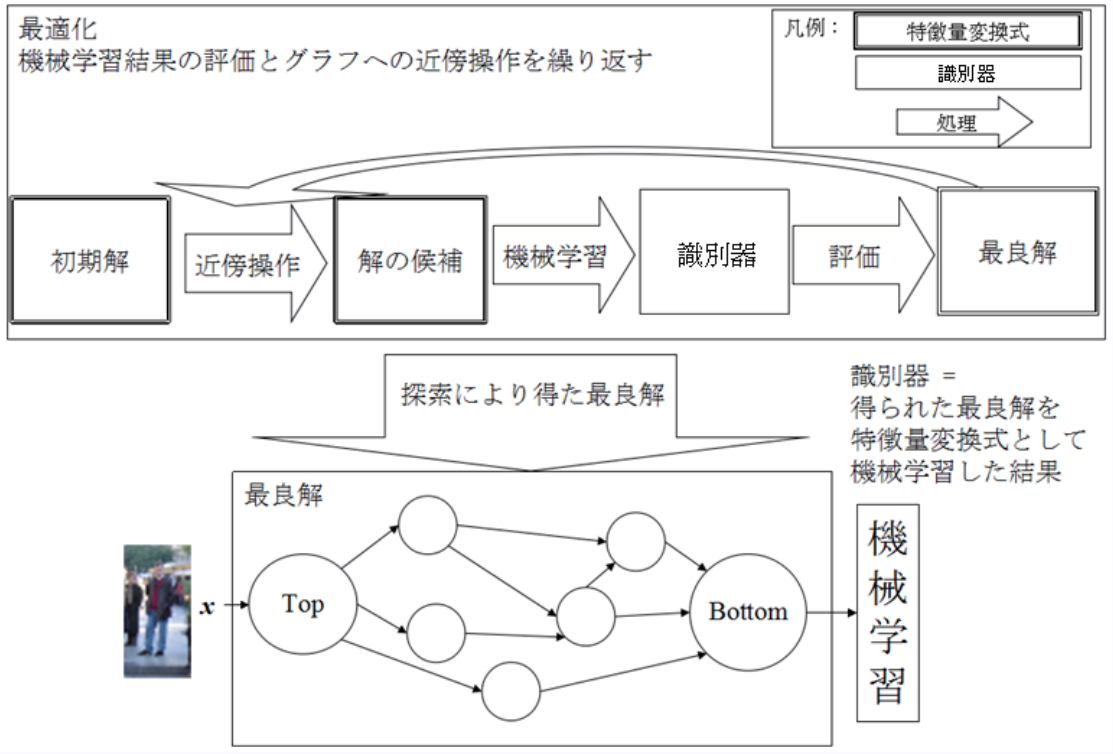


図 3.2　提案法の枠組みの概要.

る．FTN を導出するために，まず，入力画像 $\underline{\boldsymbol{I}}$ から特徴量ベクトル $\tilde{\boldsymbol{x}}$ を計算する式を示す．式(3.7)～(3.8)では入出力はテンソルであり，ある出力のテンソルの第 3 モードのスライスによって生成される行列をある関数の入力とする連鎖を繰り返すことを表している．そして，この連鎖は最終的に式(3.9)の f_J まで到達すると特徴量ベクトル $\tilde{\boldsymbol{x}}$ を出力する．なお，式(3.7)～(3.8)では $\boldsymbol{\lambda}$ を省略しているが，式(3.7)～(3.9)は式(3.3)のように関数はそのパラメータ $\boldsymbol{\lambda}$ を持ち，式(3.6)を要素で書き下した式であることに注意する．

$$\underline{\boldsymbol{Y}}^1 = f_1\left(\underline{\boldsymbol{I}}\right),\ \underline{\boldsymbol{I}} \in \boldsymbol{R}^{C\times W\times H}. \tag{3.7}$$

$$\begin{aligned}
&\underline{\boldsymbol{Y}}^j = f_j\left(\underline{\boldsymbol{S}}^j\right),\ \underline{\boldsymbol{Y}}^j \in \boldsymbol{R}^{C_j\times W_j\times H_j},\\
&\underline{\boldsymbol{S}}^j = \left(\underline{\boldsymbol{Y}}^{j_1}_{c_1^{j_1}(3)}, \underline{\boldsymbol{Y}}^{j_1}_{c_2^{j_1}(3)}, \ldots, \underline{\boldsymbol{Y}}^{j_1}_{C^{j_1}(3)}, \underline{\boldsymbol{Y}}^{j_2}_{c_1^{j_2}(3)}, \ldots, \underline{\boldsymbol{Y}}^{j_{J'}}_{C^{j_{J'}}(3)}\right),\ \underline{\boldsymbol{S}}^j \in \boldsymbol{R}^{C_S\times W_S\times H_S},\\
&j \in \{2,3,\ldots,J-1\},\ \iota^j \subset \{1,2,\ldots,J-2\},\ j \notin \iota^j,\\
&\iota^j = \{j_1, j_2, \ldots, j_{J'}\},\\
&c^{\iota^j} = \{c_1^{j_1}, c_2^{j_1}, \ldots, C^{j_1}, c_1^{j_2}, \ldots, C^{j_{J'}}\}.
\end{aligned} \tag{3.8}$$

$$\tilde{\boldsymbol{x}} = f_J\left(\underline{\boldsymbol{S}}^J\right),\ \tilde{\boldsymbol{x}} \in \boldsymbol{R}^D. \tag{3.9}$$

ここで，入力画像 $\underline{\boldsymbol{I}}$ に対して，C はチャンネル，W は画像横サイズ，H は画像縦サイズである．関数 f_j により計算された $\underline{\boldsymbol{Y}}^j$ のチャンネル，横サイズ，縦サイズをそれぞれ C_j，W_j，H_j とする．関数 f_j に入力される関数 f_{j_1} の出力の第 3 モードの第 c_1 成分のスライス，即ち行列を $\underline{\boldsymbol{Y}}^{j_1}_{c_1^{j_1}(3)}$ で表す．式(3.8)は関数 f_j には複数の関数の出力の複数のチャンネル成分が入力される場合があることを表している．入力される複数の行列を並べた一つのテンソル $\underline{\boldsymbol{S}}^j$ を構築し，それが関数 f_j に入力される．このとき，入力に用いられる複数の行列は縦サイズ，横サイズが異なる場合があるため，それらの最小値をテンソル $\underline{\boldsymbol{S}}^j$ の横サイズ $W_S = \min\left(W_{j_1}, W_{j_2}, ..., W_{j_{J'}}\right)$，縦サイズ $H_S = \min\left(H_{j_1}, H_{j_2}, ..., H_{j_{J'}}\right)$ とする．テンソル $\underline{\boldsymbol{S}}^j$ のチャンネルサイズ C_S は入力に用いた行列の個数である．D は特徴量ベクトル $\tilde{\boldsymbol{x}}$ の次元数である．関数 f_J は以下の式のように入力されたテンソルを機械学習のためにベクトルに並べる関数である．

$$\tilde{\boldsymbol{x}} = f_J\left(\underline{\boldsymbol{S}}^J\right) = \left(\underline{\boldsymbol{Y}}_{111}, \underline{\boldsymbol{Y}}_{112}, \ldots \underline{\boldsymbol{Y}}_{11W_J}, \underline{\boldsymbol{Y}}_{121}, \ldots \underline{\boldsymbol{Y}}_{1W_J H_J 1}, \ldots \underline{\boldsymbol{Y}}_{211}, \ldots \underline{\boldsymbol{Y}}_{C_J W_J H_J}\right). \tag{3.10}$$

式(3.7)～(3.9)を用いて CNN を表すには，f に畳み込みフィルタ，活性化関数，Max フィルタを与えることにより，HOG では f に勾配強度，勾配方向，ヒストグラム，正規化処理を与える．ここで，式(3.6)の関数が持つパラメータ $\boldsymbol{\lambda}$ には例えば畳み込みフィルタであれば畳み込みの重み，ヒストグラムの算出であれば分割数が含まれる．f_J はテンソルの要素をチャンネル方向，横，縦の順に並べたベクトル $\tilde{\boldsymbol{x}}$ への変換である．これにより，機械学習に入力されるベクトル $\tilde{\boldsymbol{x}}$ へ変換される．

さて，提案法の目的は式(3.6)の最適化である．ここで，式(3.8)における j^j と c^{j^j} が式(3.6)のパラメータ $\boldsymbol{\Lambda}$ であり，式(3.8)の関数が持つパラメータが式(3.6)の $\boldsymbol{\lambda}$ となる．提案法では $\boldsymbol{\Lambda}$ と $\boldsymbol{\lambda}$ を最適化するための表現としてグラフを用いる．グラフを用いれば最適化の時に $\boldsymbol{\Lambda}$ と $\boldsymbol{\lambda}$ に対する操作が簡潔に表現できるためである．式(3.7)～(3.9)をグラフのネットワーク G により表現した式は以下の通りである．

$$\begin{aligned} &G := (\Phi, V, E),\ \Phi : E \to V' \times V, \\ &V' = \left\{ j'c',\ \forall j' \in \iota^j,\ \forall c' \in c^{\iota^j} \right\},\ V = \{ j = 2, \ldots J \}, \\ &e_{j'c'j} \in E\ . \end{aligned} \tag{3.11}$$

ここで G は j=1 を始点，j=J を終点とした有向グラフである．以降始点を TOP，終点を Bottom と呼ぶ．

関数 $f_{j'}$の出力のうちチャンネル c'成分を関数 f_j に入力することは，ノード $j'c'$か

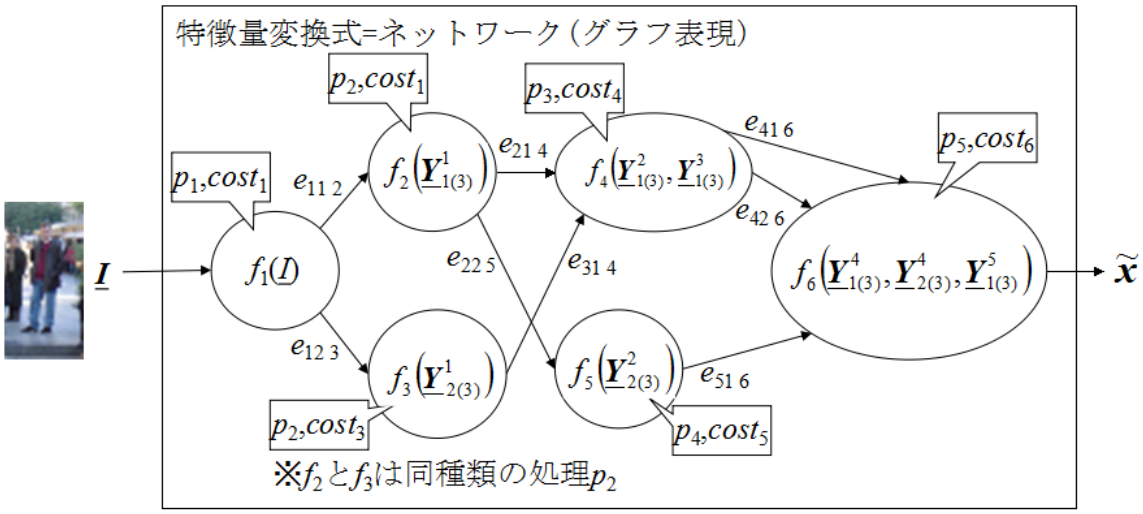


図 3.3　提案法の枠組み.

らノード j への接続を表すエッジ $e_{j'c'\,j}$ により表される．例えば図 3.3 における f_4 では，$e_{21\,4}$，$e_{31\,4}$ と表せる．ここで，$e_{j'c'\,j}$ を変えることにより，入力画像 $\underline{\boldsymbol{I}}$ から特徴量ベクトル $\tilde{\boldsymbol{x}}$ へ変換する特徴量の計算式における入出力を決めるパラメータ $\boldsymbol{\Lambda}$ を操作することが可能となる．この変換ネットワークには f_j に対応するコスト c_j，f_j に対応する処理の種類 $p^j \in P=\{p_1, p_2,...,p_{|P|}\}$ が定義される．例えば，図 3.3 において処理の種類を当てはめてみると，$\{p_1, p_2, p_3, p_4, p_5\}$:={表色系変換，畳み込みフィルタ，Max フィルタ，Box フィルタ，ベクトル化}と設定できる(各処理の種類の説明は後述)．ここで，j=2,3 では同じ処理の種類 p_2=畳み込みフィルタが設定されている．このように同じ処理の種類が複数の関数に指定される場合がある．ただし，f_2 と f_3 では畳み込みフィルタの重みは異なってもよい．それぞれの畳み込みフィルタの重みは関数が持つパラメータ $\boldsymbol{\lambda}$ に含まれており，最適化の対象となる．本論では式(3.7)～(3.11)を FTN と呼ぶ.

3.3　FTOP(Feature Transform Optimization Problem)の定義

本節では FTN を最適化するための組合せ最適化問題を定義する．図 3.2 のように提案法では特徴量変換式を FTN を用いて生成した後, 機械学習により識別器を構成し，識別器に最適化用の評価サンプルを識別させることにより，特徴量変換式の評価値を計算する．そこで, 最適化問題を定義するために, 特徴量変換式 $F(\underline{\boldsymbol{I}})$ と機械学習の関係を以下の式に示す．$\boldsymbol{M}_v=\{m_v|m_v$ は最適化用の評価サンプル$\}$ とし $\boldsymbol{z}^{(m_v)}$ を識別器 *Classifier* の出力とする．

$$\tilde{\boldsymbol{x}}^{(m_v)} = F\left(\boldsymbol{I}^{(m_v)}\right),$$
$$\boldsymbol{z}^{(m_v)} = Classfier\left(\tilde{\boldsymbol{x}}^{(m_v)}\right),\ \boldsymbol{z}^{(m_v)} \in \boldsymbol{R}^{D'}. \tag{3.12}$$

ここで，D'は識別器の出力ベクトルの次元数，*Classifier* は $\boldsymbol{M}_l=\{m_l|m_l$ は最適化用の学習サンプル$\}$により学習された識別器である. 提案法では評価関数として汎化性能を計算する．$\boldsymbol{M}_v$ と $\boldsymbol{M}_l$ は一つも重複させない．従って，最適化に用いるサンプルを $\boldsymbol{M}$ とすれば，$\boldsymbol{M}=\boldsymbol{M}_v \cup \boldsymbol{M}_l$ である．ここまで定義してきた式を用いて組合せ最適化問題を定義し，その最適化問題を解く手法を提案する.

提案する最適化問題の評価関数及び制約は以下の通りである．最適化の目的は f_j に用いる処理の種類 p^j，p^j が持つ関数のパラメータ $\boldsymbol{\lambda}_j$，f_j の入力を決める $e_{j'c'j}$ を変数として以下を満たすネットワーク G を得ることである．ここで，$|\boldsymbol{P}|\times|\boldsymbol{P}|$ の接続行列 $\boldsymbol{Q}$，ネットワーク全体の合計コストの上限 $Cost_{\max}$，処理の種類 p 毎の合計コストの上限 $Cost_p$，特徴量次元数の上限 $D_{\max}$ とすれば，最適化問題は以下の式により定義される.

Objective:

$$\text{Minimize } L\left(\left\{z^{(m_v)} \mid \forall m_v \in M_v\right\}\right). \tag{3.13}$$

Subject to:

$$\forall e \in E',\ E' = \left\{e' \mid \boldsymbol{Q}\text{を満たす}e'\right\}. \tag{3.14}$$

$$\sum_{j=1}^{J} cost_j \le Cost_{\max}\ . \tag{3.15}$$

$$\sum_{j\in\{p\text{を持つノード}j\text{の集合}\}} cost_j \le Cost_p,\ \forall p \in \boldsymbol{P}\ . \tag{3.16}$$

$$\dim \tilde{\boldsymbol{x}} \le D_{\max}\ . \tag{3.17}$$

式(3.13)の目的は汎化性能の向上である. 学習に用いない評価サンプルを識別した結果を評価関数に用いることにより汎化性能の向上が期待される．なお，評価関数 L は最適化の適用先により定義を変えることを想定しているため，4.1 にて使用する L を具体的に記述している.

式(3.14)は $e_{j'c'j}$ を制限する制約である．すなわち，接続関数 $\boldsymbol{\Phi}$ に制限をかけるための制約になる．計算機資源が制限されている場合には処理の接続を予め制限することにより効率よく解を探索できる可能性がある．この制約は処理順を固定するものではない．この制約を用いれば処理順に制約がある中で探索させることができる．接続行列 $\boldsymbol{Q}$ の具体例を示す．接続行列 $\boldsymbol{Q}$ は処理の種類 p_1 から p_2 への接続が可能で，p_2 から p_1 への接続が禁止の場合，$\boldsymbol{Q}_{p1p2}=1$，$\boldsymbol{Q}_{p2p1}=0$ となる.この制約を使えば，例えば畳み込み積分の後には必ず最大値フィルタを施すということを実現できる．これは必ず 1 対 1 である必要はなく，例えば畳み込みフィルタの後には Max フィルタもしくは畳み込みフィルタを再度施すという設定も可能であ

る．逆に $\boldsymbol{Q}_{p1p2}=1$，$\forall p_1$，$p_2 \in \mathrm{P}$ とすれば，処理順の制限なく探索させることができる．目的に合わせて制約を選択可能である．

式(3.15)はネットワーク全体の処理のコストの総和が $Cost_{\max}$ 以下となる制約である．各処理の処理時間を計測しておき，それを *cost* に設定すれば，処理時間の制限を与えた状態で最適化することができる．*cost* には必ずしも処理時間を設定する必要はなく，$cost_j=1$，$\forall j \in \{1,2,...,J\}$ とすれば，ネットワーク全体のノードの個数を制限できる．

式(3.16)はネットワーク G に含まれる処理の種類 p 毎のコストの総和を $Cost_p$ 以下に制限する制約である．例えば，ある処理の種類は処理時間が大きいことが分かっているため予め 1 回しか使わない，という設定が必要な場合に使用する．他には，ある処理の種類 p_1 は多く，ある処理の種類 p_2 の処理は少なくネットワーク G に含めたい場合，$Cost_{p1}<Cost_{p2}$ として設定することにより，そのような傾向を与えることができる．傾向と述べている理由は，この制約を満たしつつコストの総和の大小関係が逆になる場合もあるからである．

式(3.17)は機械学習に入力する特徴量ベクトル $\tilde{\boldsymbol{x}}$ の次元数を $D_{\max}$ 以下にする制約である．最終的な識別器を構成するための機械学習を実行する時に，機械学習に掛けられる時間や計算機資源の制限により，特徴量次元数を制限する場合に設定する．以上に定義した最適化問題を特徴量変換最適化問題 FTOP(Feature Transform Optimization Problem)と呼ぶ．

3.4 FTOP(Feature Transform Optimization Problem)の求解アルゴリズム

本節では FTOP の解法を提案する．FTOP にて扱う関数は微分可能でなくてよい．従って，勾配法のように連続値の最適化問題を解く解法を用いることができない．そこで，FTOP を組合せ最適化問題の解法を用いて解く．組合せ最適化問題の解法は探索アルゴリズムと近傍操作からなる．探索アルゴリズムの基本となる処理の流れは，解を生成し，解に対して評価値を計算し，よい評価値が得られた解を最良解とすることを繰り返す．メタヒューリスティクスを用いる場合は，探索の過程で解を生成する時に近傍操作が用いられる．近傍操作は組合せ最適化問題毎に設計する．

3.4.1 多点局所探索

メタヒューリスティクスには局所探索，シミュレーテッドアニーリング，タブーサーチ，遺伝的アルゴリズムが含まれる．FTOP を解くうえで，機能的には上記のどのアルゴリズムも採用可能であるが，本論では局所探索を採用する．理由は上記のどのアルゴリズムも絶対的な優位性がなく，局所探索法は他の手法に比

べ探索のためのパラメータが少なく実装が容易であるためである．例えばタブーサーチと多点局所探索法とを比較した実験[99]では，従来タブーサーチがよいと考えられてきたが, 多点局所探索の方が高速に求解されたことが報告されている．局所探索には局所解に陥った場合に抜け出せないという問題があるが，これは図 3.4 に示す多点局所探索を採用することにより対応できる．図 3.5 に示すように異なる初期解を複数生成し，それぞれの初期解から局所探索することにより，局所解に陥ることを防ぐ(図 3.5)．多点局所探索はそれぞれの局所探索を独立して実行することができるため，複数の局所探索を同時並行に実行することができる．こ

局所探索

Input:収束条件，ランダムシード
Output:最良解，評価値

```
1:  ランダムシードを用いて初期解の生成
2:  最良解 ← 初期解
3:  Best Obj. ← 最良解の評価値を計算
4:  While 収束条件を満たさない
5:      解の候補 ← 制約を満たすように
                    最良解をランダムシードを用いて近傍操作
6:      Current Obj. ← 解の候補の評価値を計算
7:      If Current Obj < Best Obj.
8:          最良解 ← 解の候補
9:          Best Obj. ← Current Obj
10:     End if
12:  End while
11:  Return 最良解，Best Obj.
```

多点局所探索

Input:収束条件，初期解の生成数
Output:最良解

```
1   :For i=0; i<初期解の生成数; i++
2:      最良解[i]，Best Obj.[i] ← 局所探索(収束条件，ランダムシード[i])
3:  End for
4:  最良解 ← Best Obj.[i]が最小となる最良解[i]
5:  Return 最良解
```

図 3.4　多点局所探索アルゴリズム．

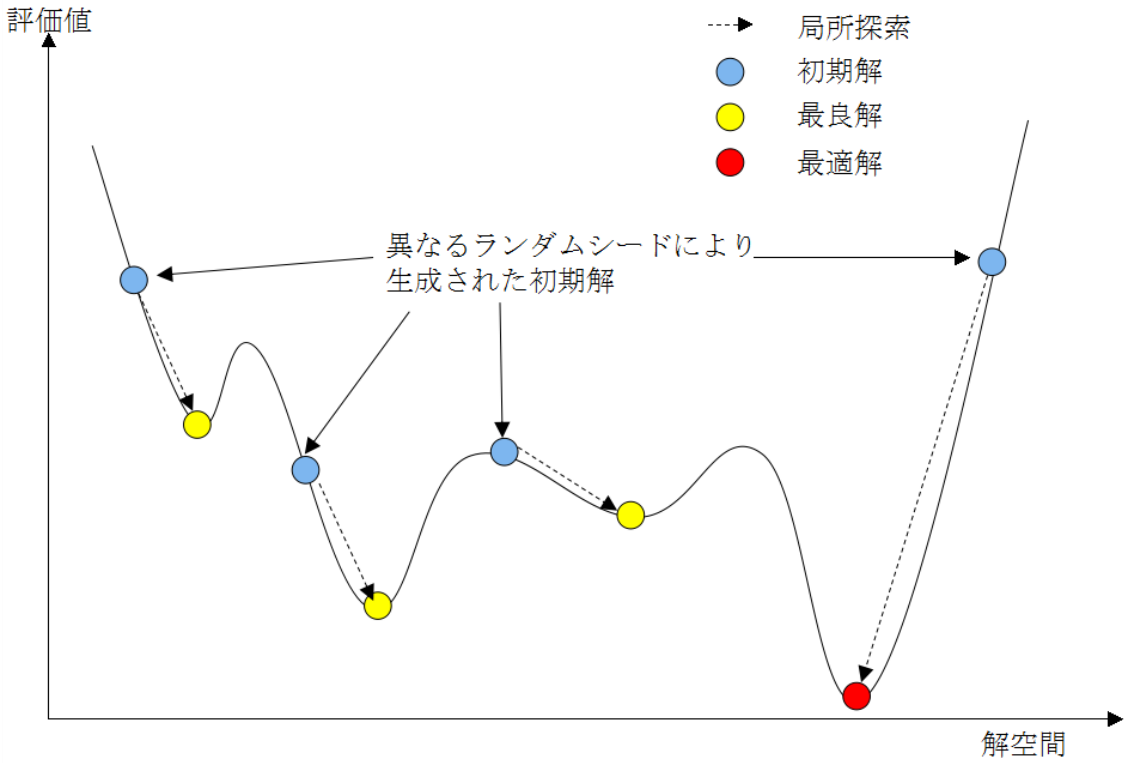


図 3.5　多点局所探索アルゴリズムの求解例.

れにより，探索時間の削減を図ることができる.

3.4.2　近傍操作

近傍操作とは解を探索する過程で候補となる解を生成するための操作である. 提案法での近傍操作は 2 つの目的のために実行される. ひとつは特徴量の計算式 F に含まれる関数 f_j が持つパラメータ λ，もう一つは特徴量の計算式 F が持つ入出力の接続を表すパラメータ Λ の最適化である. それぞれ具体例を示す. パラメータ λ の近傍操作は用いる処理の種類によって異なる. そのため，3.1 にて詳細を示す. ここでは，パラメータ Λ の近傍操作について説明する. パラメータ Λ に対する近傍操作はネットワーク G に対して実行される. 例えば, G に含まれる $e_{j'1c'1j}$ を $e_{j'2c'2j}$ に変更することはパラメータ Λ に対する近傍操作にあたる. この場合は, f_j への入力が，関数 $f_{j'1}$ の出力のチャンネル c'_1 から関数 $f_{j'2}$ の出力のチャンネル c'_2 に変更されることを表す. 以下に，提案法における近傍操作を示す. 以下の近傍操作を図示すれば図 3.6 のようになる.

①G にノード j を追加

ネットワーク G にノード j を追加することは，新しい関数 f_j をネットワーク G

に追加することを表す．ノード j は②の近傍操作により G に含まれる他のノードと接続されることにより，特徴量の計算式の一部となる．

②G にエッジ e を追加

ネットワーク G にエッジ e を追加する．

この操作はある関数の出力と，その出力を入力とするある関数とを接続するための操作である．①のように G に新たなノードが追加された場合や既存のノード間に新たな入出力関係を追加する時に用いられる．

③G に含まれるノード j の削除

ネットワーク G に含まれるノード j を削除する．これは特徴量の計算式において不要になったノードを削除する時や⑤の処理の一部として利用される．特徴量の計算式において不要になるとは，そのノードからの接続を直接または間接的に持つ全てのノードが Bottom と接続されていないことを検査することにより判定される．Bottom は機械学習に用いる特徴量ベクトルを生成する関数である．そのノードに接続されていないということは特徴量ベクトルに反映されないということであるため，ネットワーク G から削除することができる．

④G に含まれるエッジ e の削除

Bottom に接続するエッジを削除することによりある関数の出力を特徴量ベクトルに含むかを調整することができる．③により不要になったエッジの削除にも用いられる．③によりノードが削除されると，そのノードを接続先としていたエッジは不要であると判定される．

⑤G に含まれるノードと新規のノードを交換

ネットワーク G に含まれていたノードと新規のノードを交換する．

このとき，交換元のノード j_{old} と交換先のノード j_{new} の入次数，出次数が同じである必要がある．例えば，j_{old} が 2 チャンネルを入力に 1 チャンネルを出力する場合，j_{new} も同じく 2 チャンネルを入力に 1 チャンネルを出力する関数でなければならない．もし，j_{old} と j_{new} の関数の種類が等しい場合，パラメータ λ を近傍操作することに等しく，j_{old} と j_{new} の関数の種類が異なる場合はパラメータ Λ を近傍操作することに等しい．

上記の①～⑤を繰り返し実行することにより，FTOP を満たすネットワーク G を探索する．

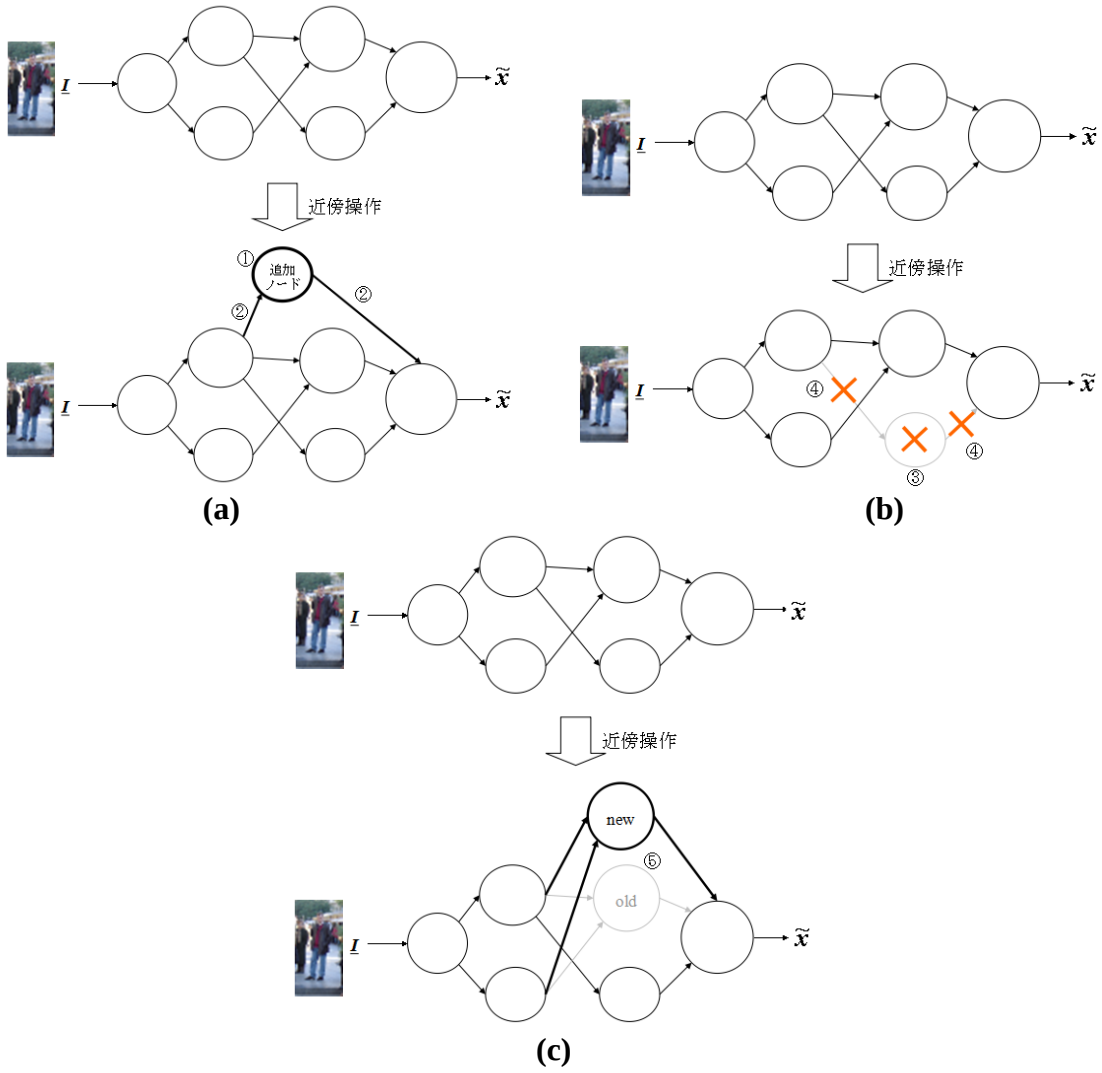


図 3.6　近傍操作．(a)①②によるノードとエッジの追加，(b)③④によるノードとエッジの削除，(c)⑤によるノードの入れ替え．

3.4.3 求解アルゴリズム

3.4.1 及び 3.4.2 にて述べた多点局所探索と近傍操作を用いて FTOP を解く．求解アルゴリズムを図 3.7 に示す．求解アルゴリズムの概要は以下の通りである．近傍操作①～⑤を用いて解の候補を生成する時には制約式(3.14)～(3.17)を違反させない．生成した解の候補に対して評価値を L により計算する．評価値が最良解の評価値よりも小さければ，最良解を解の候補に，最良解の評価値を解の候補の評価値に更新する．これらの操作を多点局所探索により実行する．収束判定は最良解の更新が停止した後の解の候補の生成回数とする．

ノード及びエッジの追加操作

INPUT: G

OUTPUT: G

While
　ランダムに変換ノード j_{add} を生成
　If　G と j_{add} から算出した式(3.15)or(3.16)が制約違反
　　Break;
　End if
　近傍操作①により j_{add} を G に追加
　G からランダムに $j'c'$を選択
　If　$e_{j'c'\,jadd}$ が式(3.14)を満たす
　　近傍操作②により $j'\,c'$を j_{add} と接続する $e_{j'c'\,jadd}$ を G に追加
　End if
End while
Return G

Bottom へのエッジの追加操作

INPUT: G

OUTPUT: G

While
　G からランダムに $j'c'$を選択
　If $j'c'$を Bottom と接続した場合に式(3.14)or(3.15)が制約違反
　　Break
　End if
　近傍操作②により $j'c'$と Bottom を接続し，$e_{j'c'\,jbottom}$ を G に追加
End while
Return G

初期解の生成

INPUT: -

OUTPUT: G

G に Top を追加
G ← ノード及びエッジの追加操作(G)
G に Bottom を追加

図 3.7　FTOP の求解アルゴリズム（続く）

4: G ← Bottom へのエッジ追加操作(*G*)
5: Return *G*

求解アルゴリズム

INPUT: 初期解生成回数, 近傍探索収束回数,
⑤操作回数, エッジ削除率
OUTPUT: *G**

1: For 初期解生成数
2: *G* ← 初期解の生成
3: *G* の評価値を式(3.13)により算出
4: If 3:の評価値 < 最良解の評価値
5: 最良解 *G** ← *G*
6: 最良解の評価値 ← 3:の評価値
7: End if
8: End for
9: *G* ← copy(*G**)
10: While 近傍探索収束回数 > カウント
11: ランダムにノード j_{new} を生成
12: *G* からランダムにノード j_{old} を選択(j_{old}≠Top かつ j_{old}≠Bottom)
13: For ⑤操作回数
14: If 近傍操作⑤により j_{old} と j_{new} を交換した場合に
式(3.14)～(3.16)を満たす
15: 近傍操作⑤により j_{old} と j_{new} を交換
16: End if
17: End for
18: 近傍操作④により終端ノード Bottom に接続されたエッジを
エッジ削除率に従いランダムに削除
19: 18:の結果，出力先の接続無しとなったノードを近傍操作③，④により
再帰的に削除
20: ノード及びエッジの追加操作(*G*)
21: Bottom へのエッジの追加操作(*G*)
22: *G* の評価値を式(3.13)により算出
23: If 22:の評価値 < 最良解の評価値
24: *G** ← copy(*G*)
25: 最良解の評価値 ← 22:の評価値

図 3.7 FTOP の求解アルゴリズム（続く）

26: カウント ← 0
27: Else
28: カウントをインクリメント
29: G ← copy(G*)
30: End if
31: End while
32: Return G*

図 3.7 FTOP の求解アルゴリズム.

3.5 仮説検証実験

仮説を検証するための予備実験を実施した．FTOP を求解することが画像識別に有効であるためには，最適化問題における評価関数と未知画像に対する識別精度に相関がなければならない．この仮説を確認する実験が仮説検証実験である．

歩行者識別のために FTOP を以下のように設定する．FTOP の評価関数には AUC(Area Under the Curve)を用いる．AUC は誤った識別が少ないほど小さい値を示す．評価値計算のために用いる機械学習及び最終的な識別器の構築には弱識別器に決定木を用いた AdaBoost を用いる. ノードに用いる処理の種類には表色系変換，畳み込みフィルタ，Max フィルタを用いる．なお，本研究の主眼となる実験は 4 章で述べるため，4 章にて一覧性を持って FTOP を歩行者に適用する手法を詳述する．（AdaBoost，決定木，AUC は 4.1 節に詳述する．ノードに用いる処理の種類である表色系変換, 畳み込みフィルタ, Max フィルタは 4.2 節に詳述する.）

実験には歩行者検出のベンチマークデータセットである INRIA Person Dataset[90]を用いる．本実験に用いた画像サンプル数を表 3.1 に示す．表 3.1 の学習サンプルは最適化及び最終的な識別器の構築に用いられる．従って，最適化の時には学習サンプルをさらに最適化用の学習サンプルと評価サンプルに分ける．

付録の図 A.1 に INRIA Person Dataset の例を示す．この Dataset はトリミングされた歩行者の画像とトリミングされていない歩行者の存在しない画像が用意されている．歩行者が存在しないサンプルを収集するため，歩行者の存在しない画像からランダムにトリミングした画像を用いた．歩行者検出では歩行者(Positive)とそれ以外(Negative)の 2 クラス分類として識別器を構成する．

本実験の実験に用いたソフトウェアの実装について述べる．本実験に用いたライブラリを表 3.2 に，実験した計算機環境を表 3.3 に示す．本実験では実験の再現性を高めるため OSS のみを用いて開発した. OpenCV(Open Source Computer Vision Library)[101]は 1999 年に開発が始まり，現在 Intel 社により開発が管理されている

画像処理，機械学習の OSS である．実装が提供されていない画像処理についてはリファレンス実装した．提案法の FTOP の求解はスクラッチ開発した．

本実験では $\boldsymbol{\lambda}$ の最適化のために各パラメータを探索する．これらのパラメータには探索の値域と実数においては幅が指定される．表 3.4 にこれらの値を示す．

本実験では最適化の評価値計算時間を短縮するために，表 3.5(a)に示す AdaBoost の設定により評価値を計算するための学習を実行する．最適化後に得られた FTN(Feature Transform Network)を用いて最終的な識別器を構築する時には表 3.5(b)の設定を用いる．他にも評価値計算時間を短縮するために FTN により得られた特徴量を全て用いるのではなく $\dim \tilde{\boldsymbol{x}}$ ×次元数率 D_{ratio}(<1)の特徴量をランダムに選択し学習に用いる．

仮説を検証するための予備実験として実際に FTOP を求解し，得られた最良解に対して最終的な識別器を学習し，未知画像に対する識別精度を計測した．実験では表 3.6 に示す最適化パラメータと表 3.7 に示す FTOP の設定を用いて実験した．表 3.7 の $\boldsymbol{Q}$ は T 行，p_1 列が 1 の場合 T から p_1 への接続を許すことを表す．

表 3.1　INRIA Person Dataset のサンプル数.

学習 Positive サンプル数	2416
学習 Negative サンプル数	11504
評価 Positive サンプル数	1126
評価 Negative サンプル数	11560
解像度　横×縦　[pixel]	64x128

表 3.2　ソフトウェアの構成.

ランタイム(コンパイラ)	Visual C++ 2012 64bit
表色系変換，畳み込みフィルタ，Max フィルタ	OpenCV 3.0
AdaBoost	OpenCV 3.0
FTOP 求解	スクラッチ実装

表 3.3　計算機環境.

OS	Windows7 Pro 64bit
CPU	Intel(R) Core(TM) i7-3770K CPU 3.50GHz
メモリ	32GB

表 3.4　処理のパラメータの値域と刻み幅の分割数.

処理の種類	項目		最小	最大	分割数
表色系変換	-				
畳み込み	Gaussian (式 4.32)	Width	3	11	-
		Height	3	11	-
		sigma	0.1	10	100
	DoG (式 4.33)	Width	3	11	-
		Height	3	11	-
		sigma1	0.1	10	100
		T	0.1	0.99	100
	Gabor (式 4.34)	Width	3	11	-
		Height	3	11	-
		sigma	0.1	10	100
		Theta	-π	π	100
		Gamma	0.1	1	100
Max (式 4.40)	Width		3	11	-
	Height		3	11	-

表 3.5　AdaBoost の設定．(a)最適化用，(b)最終的な識別器構築用.

(a)

弱識別器数	100
弱識別器の種類	決定木
決定木の深さ	2
葉の最小サンプル数	10

(b)

弱識別器数	1000
弱識別器の種類	決定木
決定木の深さ	2
葉の最小サンプル数	10

表 3.6　FTOP の求解アルゴリズムの設定.

$\|\boldsymbol{M}_l\|$	1392
$\|\boldsymbol{M}_v\|$	12528
D_{ratio}	0.1
初期解生成回数	10
近傍探索収束回数	100
エッジ削除率	0.2

表 3.7　FTOP の設定.

<table>
<tr><td>処理の種類</td><td colspan="6">T:Top, p_1:Convert Color, p_2:Convolution, p_3:Max, B:Bottom</td></tr>
<tr><td>$cost_j$
$\forall j \in \{1,2,...,J\}$</td><td colspan="6">1</td></tr>
<tr><td>$Cost_{\max}$</td><td colspan="6">25</td></tr>
<tr><td>$Cost_{p1}$</td><td colspan="6">1</td></tr>
<tr><td>$Cost_{p2}$</td><td colspan="6">15</td></tr>
<tr><td>$D_{\max}$ を満たすチャンネル数</td><td colspan="6">10</td></tr>
<tr><td rowspan="6">$\boldsymbol{Q}$</td><td></td><td>T</td><td>p_1</td><td>p_2</td><td>p_3</td><td>B</td></tr>
<tr><td>T</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>p_1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>p_2</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>p_3</td><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>B</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>

実験結果を図 3.8 に示す. 図 3.8 では横軸に最適化の探索における評価値を，縦軸に各最良解を用いて最終的な識別器を構築し未知画像の AUC を計算した結果をそれぞれプロットした．実験時間の観点から得られた評価値が等間隔になるように 5 個の解を選択しプロットした．また，これらの評価値と AUC の相関係数は 0.8973 であった．　図 3.9，図 3.10 に評価値が最も小さい解と最も大きい解により得られた FTN と特徴量の変換結果を可視化した結果をそれぞれ示す. 図 3.9, 図 3.10 の見方を説明する．(a)では処理は左から右へ流れる．四角の中が処理の種類であり，数字はノードを一意に識別する任意の数字である．エッジ上の数字が入力されるチャンネルを表す．処理が持つパラメータ $\boldsymbol{\lambda}$ は表現されておらず，パ

ラメータ Λ に注目した表現となっている．(b)では Bottom にて集められた特徴量のテンソルをダイナミックレンジを 0 から 255 に調整し可視化している．

表 3.8 に処理時間と最終的な識別器を構築した時の特徴量次元数及び機械学習の処理時間を示す．

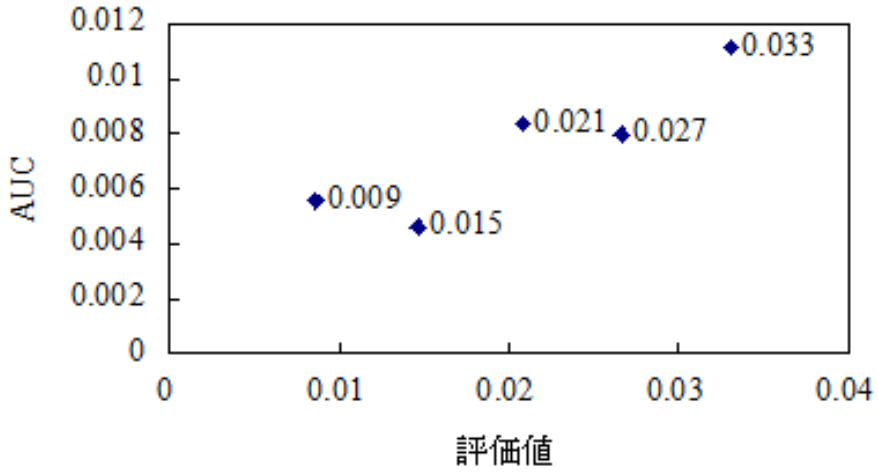
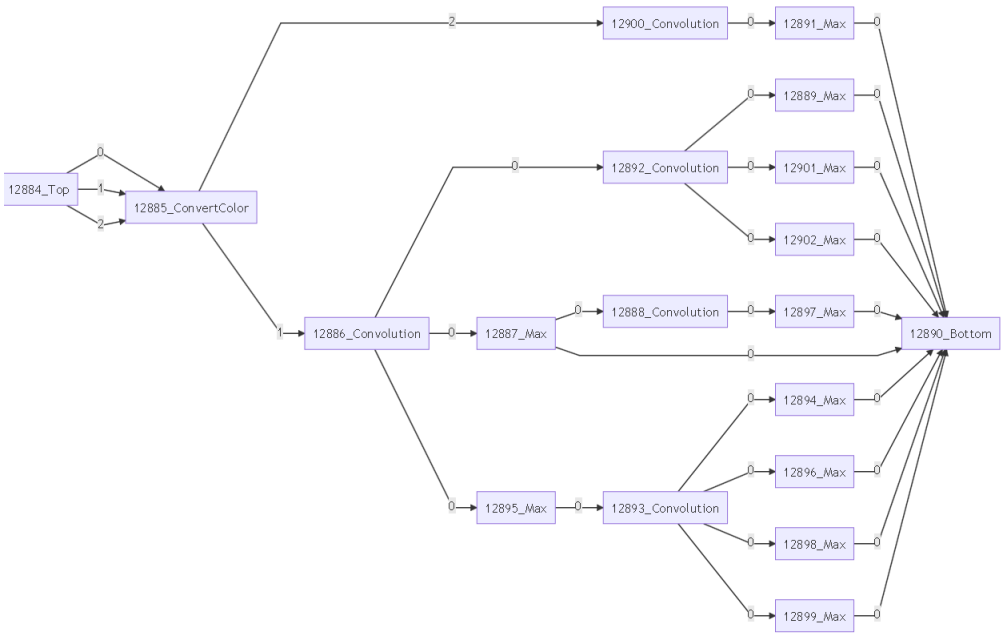


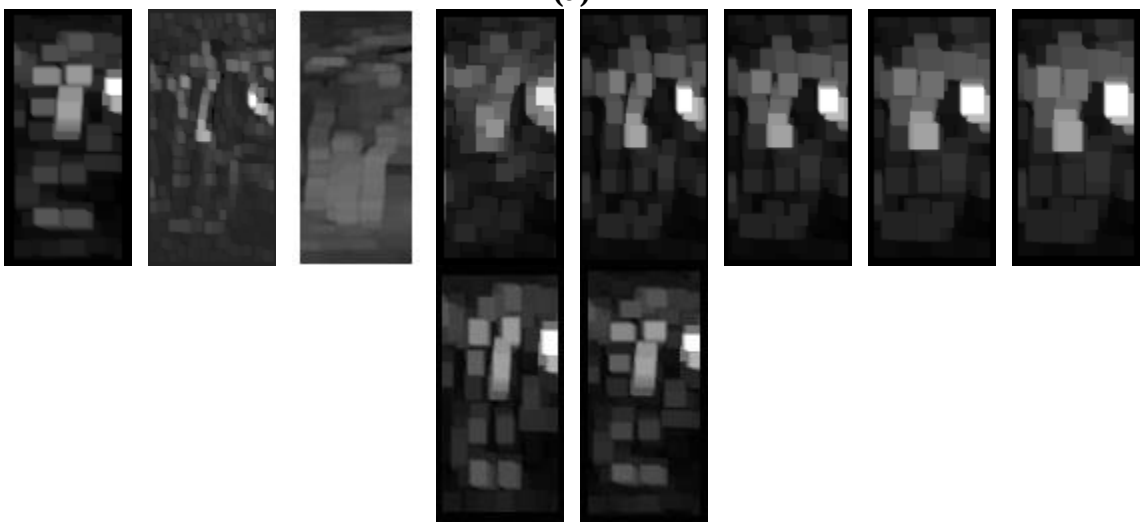
図 3.8　実験から得られた評価値と AUC の関係.

表 3.8　実験結果の処理時間.

Obj.	0.033	0.027	0.021	0.015	0.009
最適化[min]	1620				
機械学習[min]	2519	2530	2843	2527	2184
特徴量次元数	64688	48568	54916	53656	56504
AUC	0.0111	0.0080	0.0084	0.0046	0.0056

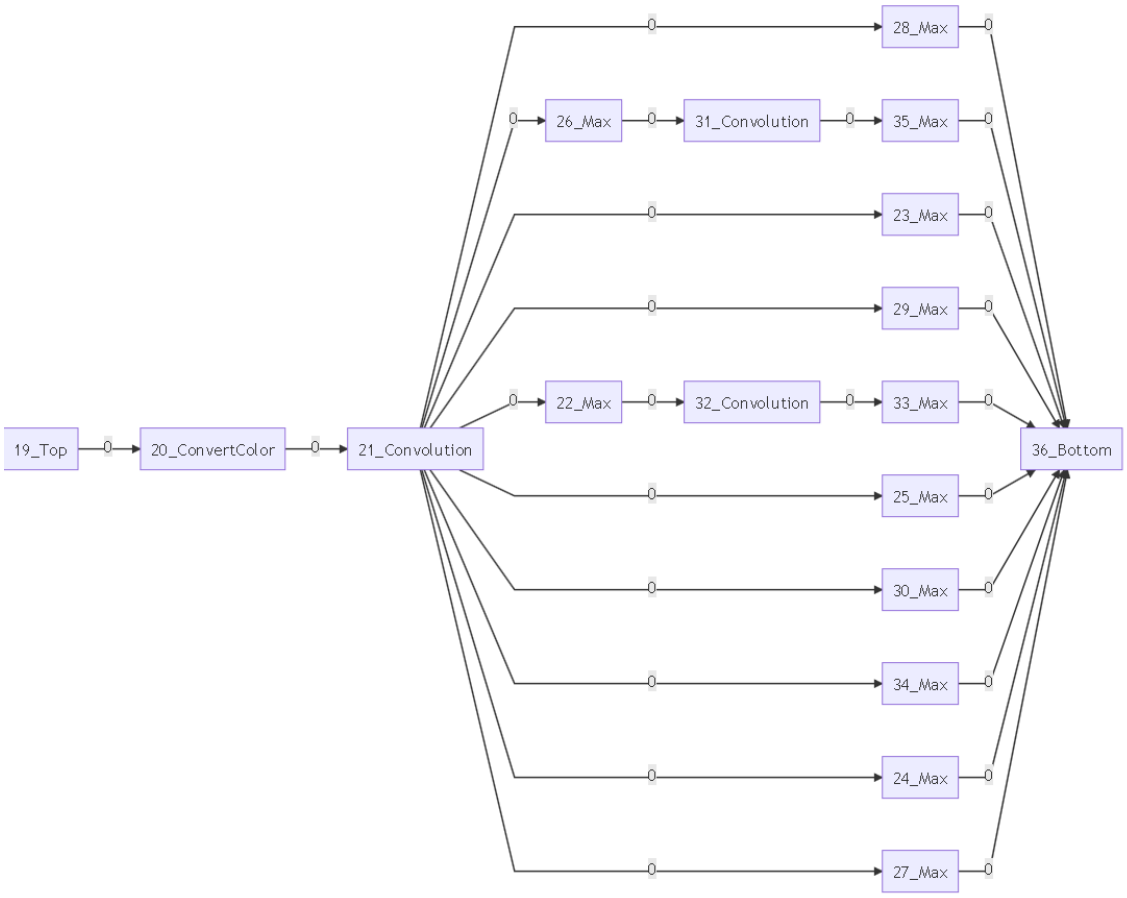


(a)



(b)

図 3.9　最も評価値が小さい最良解．(a)FTN，(b)特徴量の可視化.



(a)

(b)

図 3.10　最も評価値が大きい最良解．(a)FTN，(b)特徴量の可視化.

3.6　仮説検証実験に対する考察

FTOP の式(3.13)は学習には用いない評価用のサンプルに対する識別能力の評価と未知画像に対する汎化性能との間に相関があるという仮説に基づいている．この仮説が正しければ，FTOP を求解することは汎化性能向上に効果があるといえる．本実験における結論は FTOP を求解することが汎化性能向上に効果的であるということである．この実験では汎化性能を未知画像に対する AUC により計測している. 図 3.8 の評価値と AUC の相関係数 0.8973（ギルフォードの基準[108] によれば 0.7 から 0.9 は高い相関）によれば高い相関があると考えられる．

しかしながら図 3.8 を見ると, 必ずしも評価値が小さくなれば AUC が小さくなっているわけではない．これは，評価値計算時に最適化用の学習サンプルと評価サンプルに分ける時にサンプルの偏りが発生し，その偏りによって評価値が変わってしまうことが原因として考えられる．この偏りを減らすためには交差検定により評価値を計算することが挙げられる．ただし，これを実施するとサンプリング回数を増やす分，評価値算出に時間を要すことになり，サンプリング回数と探索時間とのトレードオフとなる．つまり，評価の精度が向上する代わりに，評価値の計算に時間を要し解の評価回数が減ってしまうために，よい解が得られない場合があるということである．評価値の計算には機械学習が実行されるが，最適化全体に対する評価値の計算に要する時間の割合を調査したところ，97.8%であった．従って，交差検定のフォールド数に比例して最適化の処理時間がかかってしまうことになる．

表 3.8 より特徴量の次元数との関係を考察する. 特徴量の次元数と AUC との相関係数は 0.5439（ギルフォードの基準[108] によれば 0.4 から 0.7 は有意に相関）によれば正の相関が出ている．しかし，AdaBoost の多くの特徴量から互いを補間し合う特徴量を選択するという特性から考えれば，特徴量が多い方が汎化性能を向上させる可能性があると考えられる. AUC は小さい方が汎化性能が高いことを表す指標であるため，正の相関はこの特性とは逆を示している．特徴量の次元数が多くても, 互いを補間する特徴量が少なければ(同一の性能を持つ特徴量が多く含まれれば)AdaBoost の特性を効果的に発揮できなくなると考えられる．本実験における次元数と AUC の相関は FTOP の求解の効果としては望ましい．なぜなら，少ない特徴量でも汎化性能がよい結果を得ることができていることを示しているからである．少ない特徴量でも互いを補間する特徴量が多く含まれていれば AdaBoost の特性は汎化性能の向上に効果的である．FTOP の求解は評価値に汎化性能を測る AUC を用いることにより，特徴量の次元数に関わらず，互いを補間する特徴量が多く含まれるように FTN を最適化していると考えられる．

図 3.9(a)，図 3.10(a)より FTN の構造図について考察する．図 3.9(a)は最も良い評価値(評価値が最も小さい)を取得した FTN であり，図 3.10(a)は最も悪い評価値

(評価値が最も大きい)を取得した FTN である．図 3.10(a)では畳み込みの種類別に FTN の経路を見れば，①21，②21→26→31，③21→22→32 の 3 種類であり，最終的な識別器を構築するための特徴量である Bottom における出力テンソルの 10 チャンネル中，8 チャンネルが経路①に対する Max フィルタであることが分かる．これに対し，図 3.9(a)の畳み込みの種類別に FTN の経路を見れば，①12900，②12886→12892, ③12886→12887, ④12886→12887→12888, ⑤12886→12895→12893 の 5 種類ある．さらに，Bottom における出力テンソルの 10 チャンネルには経路⑤に対する Max フィルタが 4 チャンネル含まれ, 他のチャンネルは分散されている. これにより, 歩行者識別における畳み込みフィルタと Max フィルタの関係は, 少ない畳み込みフィルタに対して複数の Max フィルタを設定するよりも, 畳み込みフィルタの種類を多く含むほうが効果的であると考えられる．これは[75]の報告における NN のある層において畳み込みフィルタの種類を増やすこと(Inception 構造)により汎化性能が向上することとも合致した結果であると考えられる. つまり，FTOP の求解は手動での特徴量の設計による改善を自動で得たとも解釈できる.

3.7 まとめ

本章では FTN，FTOP に至る問題提起と，FTOP を求解することが画像識別に効果があるかを確認するための仮説検証実験について述べた．画像識別において特徴量変換は汎化性能に寄与する重要な技術であり，多くの研究者によって提案されている．従来は手動により，近年では CNN を用いて自動的に画像識別における最適な特徴量変換式を得る手法が提案されている．しかし，近年の CNN を用いた手法では微分可能でない式を導入することができず，これにより特徴量変換の探索範囲が狭められ，より最適な特徴量変換式が得られていない可能性を指摘した．これに対し，最適な特徴量計算式を得る問題を，特徴量計算式をグラフ理論のネットワーク構造で表現した FTN を用いた組合せ最適化問題として捉える FTOP とその解法を提案した．提案法の枠組みを用いて，歩行者識別を対象にこの枠組みが効果的かを確認するための仮説検証実験を実施した．仮説検証実験では，最適化時に得られた評価値(AUC)と未知のサンプルに対する評価結果である AUC の相関を計算した．その結果，0.8973 という良好な結果が得られたため，FTOP を求解することは歩行者識別に効果があると結論付けた.

第 4 章 FTOP の歩行者識別への適用

本章では FTOP を歩行者識別に適用する方法を述べる．FTOP を用いるためには，L，$\boldsymbol{Q}$，$\forall j \in \{1,2,...,J\}$に対する $cost_j$，$p \in \boldsymbol{P}$，$Cost_p$，$Cost_{\max}$，$D_{\max}$ を設定する必要がある．$\boldsymbol{Q}$，$\forall j \in \{1,2,...,J\}$に対する $cost_j$，$Cost_p$，$Cost_{\max}$，$D_{\max}$ は実験時に指定する定数である．一方，L, $p \in P$ は計算式を指定する必要がある．本章では，L, $p \in P$ の指定について述べる．L について計算するには機械学習と，機械学習により得られた識別器の出力から計算される識別精度を測る指標を指定する．これは，4.1 にて述べる．$p \in P$ については特徴量変換式の一部となる計算式を指定し，計算式が持つパラメータを定義する．これは，4.2 にて述べる．

4.1 FTOP の評価関数と機械学習

L は識別器 *Classifier* の出力 $\boldsymbol{z}^{(m_v)}$，$\forall m_v \in \boldsymbol{M}_v$ を用いて計算される．従って，*Classifier* には機械学習を指定する必要がある．2 クラス分類に対応した機械学習として本論では AdaBoost[100]を用いる．選定理由は以下の通りである．AdaBoost は[22]，[74]のように画像識別における有効性が報告されている手法である．さらに，AdaBoost は OpenCV[101]，R の ada パッケージ[102]や Python の scikit-learn[103]でもプログラミングの実装が提供されており，提案法を比較対象にする時に再現実験が容易となる．従って，提案法では AdaBoost を最適化及び最終的な識別器を構築するために AdaBoost を採用する．なお，上記のどの機械学習も提案法の FTOP の *Classifier* に利用可能である．4.1.1 に AdaBoost について，4.1.2 に AdaBoost に用いる Decision Tree について述べる．

機械学習が決まれば *Classifier* が決まるため式(3.12)より $\boldsymbol{z}^{(m_v)}$，$\forall m_v \in \boldsymbol{M}_v$ が計算できる．従って，式(3.13)の評価関数から評価値を計算するためには L を指定すればよい．提案法では L に AUC(Area Under the Curve)を用いる．AUC は[74]によって，識別精度の向上が報告されている識別精度を測るための指標である．AUC は 4.1.3 にて説明する．

4.1.1 AdaBoost

AdaBoost[100]は機械学習アルゴリズムの中でも Boosting と呼ばれる手法に分類される．Boosting は複数の識別器の出力をアンサンブルする手法の一つである．この複数の識別器は弱識別器と呼ばれる．複数の弱識別器がアンサンブルされた強識別器に対して，さらに弱識別器を加えることを繰り返すことにより識別精度の向上を図ることが Boosting の特徴である．AdaBoost では強識別器に弱識別器を加える時に，強識別器が誤って識別したサンプルに対して正しく識別する弱識別

弱識別器の学習

Input: サンプル重み分布 $Dist_t$

Output: 信頼度 α_t，弱識別器 $weak_t$

1: サンプル重み分布 $Dist_t$ において，誤り率 ε_t が最小となる弱識別器 $weak_t$ を学習する．学習サンプルに対する誤り率 ε_t を次式で定義する．

$$\varepsilon_t = \sum_{m \in \left\{ m \mid m \in M' \wedge label_m \neq weak_t\left(\tilde{\mathbf{x}}^{(m)}\right) \right\}} Dist_t(m). \tag{4.1}$$

2: 誤り率 ε_t 最小となる識別器 $weak_t$ の誤り率 ε_t から信頼度 α_t を次式を用いて計算する．

$$\alpha_t = \frac{1}{2}\log\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right) \tag{4.2}$$

3: Return α_t，誤り率 ε_t 最小となる識別器 $weak_t$

AdaBoost

Input: 弱識別器数

Output: *strong*

1: サンプル重み $Dist_0(m) \leftarrow 1/|\boldsymbol{M}'|$, $m \in \boldsymbol{M}'$.

2: For t=0; t<弱識別器数; t++

3: α_t, $weak_t \leftarrow$ 弱識別器の学習($Dist_t$)

4: 次式よりサンプル重み分布 $Dist_t$ を次式により更新する．

$$Dist_{t+1}(m) \leftarrow \frac{Dist_{t+1}(m)}{\sum_{m \in M'} Dist_{t+1}(m)} = \frac{Dist_t(m)\exp\left(-\alpha_t label_m weak_t\left(\tilde{\mathbf{x}}^{(m)}\right)\right)}{\sum_{m \in M'} Dist_t(m)\exp\left(-\alpha_t label_m weak_t\left(\tilde{\mathbf{x}}^{(m)}\right)\right)}, \ \forall m \in M'. \tag{4.3}$$

5: End for

6: 強識別器 $strong(\boldsymbol{x})$ を以下の式により生成する．

$$strong(\tilde{\mathbf{x}}) = \mathrm{sign}\left(\sum_{t=1}^{T} \alpha_t weak_t(\tilde{\mathbf{x}})\right). \tag{4.4}$$

7: Return *strong*

図 4.1　AdaBoost の学習アルゴリズム．

器を選択することにより，識別精度を向上させている．図 4.1 に AdaBoost のアルゴリズムを示す．図 4.1 において FTOP の最適化の時には $\boldsymbol{M}'=\boldsymbol{M}_l$(FTOP の最適化に用いる学習サンプル)であり，最終的な識別器を生成する時には，$\boldsymbol{M}'=\boldsymbol{M}$(FTOP

の最適化に用いる全サンプル)であることに注意する．また，最終的な識別器 *Classifier* は $\boldsymbol{M}'=\boldsymbol{M}$ により学習した図 4.1 の *strong* を用いる．

4.1.2 Decision Tree

Decision Tree は Tree 構造を用いた機械学習である．Decision Tree を用いたクラス分類では分けた後のグループにおいて同一のラベルを持つサンプルが集まるように分けるルールが探索される．ルールは $\tilde{\boldsymbol{x}}$ の d 次元目の要素 $\tilde{\boldsymbol{x}}_d$ と閾値 th_d より以下の式で与えられる．

$$\begin{matrix} \tilde{\boldsymbol{x}}_d \le th_d \text{ then Left} \\ \textit{else Right} \end{matrix} . \tag{4.5}$$

このルールは Tree のノードに指定される．サンプル m の特徴量ベクトル $\tilde{\boldsymbol{x}}^{(m)}$ が Decision Tree に入力されると，ルートからリーフに向かって上記のルールに従い左右どちらかに移動する(図 4.2)．最終的にリーフに含まれるラベル毎のサンプル数から計算される割合によって，ラベルを推定する．ここで，リーフに含まれる *label* のサンプル数 n_{label}，学習に用いた *label* のサンプル数 M_{label}，推定されたラベル *label** とすれば以下の式となる．

$$label^* = \arg\max_{label} \left(\frac{1}{M_{label}} n_{label} \right). \tag{4.6}$$

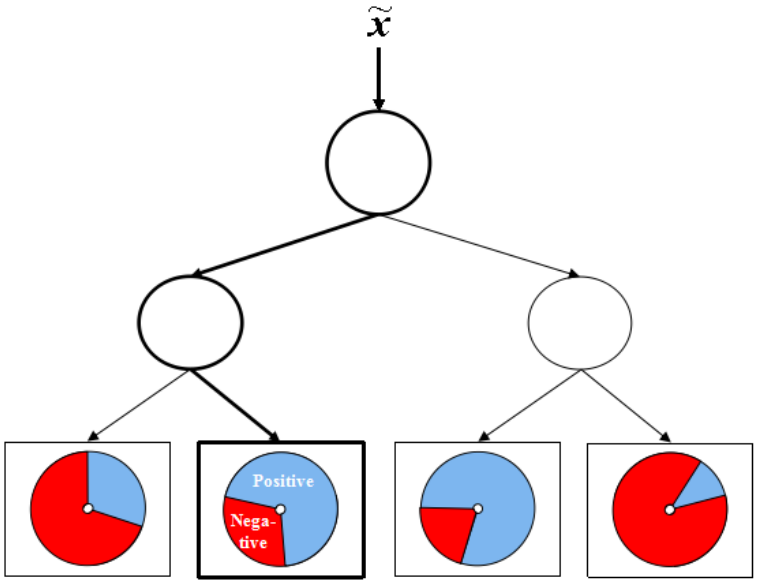


図 4.2　Decision Tree の推定の流れ．

分割ルールの探索

Input: Δ，ノード，サンプル，深さ，葉に含まれるサンプル数の最小数
Output: ノード

1: If 深さ = 最大深さ or サンプルが単独ラベル or
サンプル数≤葉に含まれるサンプル数の最小数
2: 式(4.6)により葉に推定ラベル *label** を設定
3: Return
4: End if
5: For d=0; d<D; d++
6: For $th_d = \min\left(\tilde{\boldsymbol{x}}_d^{(m)}, \forall m \in M'\right)$; $th_d \le \max\left(\tilde{\boldsymbol{x}}_d^{(m)}, \forall m \in M'\right)$; $th_d += \Delta$
7: Gini ← th_d を用いて式(4.5)よりサンプルを 2 分割後の Gini 係数を
式(4.7)より計算
8: If Gini < min
9: $d_{\min}$ ← d
10: $th_{d\min}$ ← th_d
11: End if
12: End for
13: End for
14: $d_{\min}$，$th_{d\min}$ を式(4.5)に設定したものをルールとする
15: ノードにルールを設定し，Left ノードと Right ノードを生成する
16: Left ノードのサンプル，Right ノードのサンプル ←
15:のルールによりサンプルを分割
17: 分割ルールの探索(Δ，Left ノード，Left ノードのサンプル，深さ←深さ+1)
18: 分割ルールの探索(Δ，Right ノード，Right ノードのサンプル，
深さ←深さ+1)
19: Return ノード

Decision Tree

Input: 最大深さ，葉に含まれるサンプル数の最小数， Δ
Output: ルート

1: ルートを生成
2: ルート ← 分割ルールの探索(Δ，ルート，$\boldsymbol{M}'$，0)
3: Return ルート

図 4.3 Decision Tree の学習アルゴリズム.

図 4.3 に Decision Tree の学習アルゴリズムを示す．図 4.2 において，FTOP の最適化では $\boldsymbol{M}'=\boldsymbol{M}_v$，最終的な識別器を構築する時は $\boldsymbol{M}'=\boldsymbol{M}$ を用いる．提案法において Decision Tree は AdaBoost の弱識別器 *weak* に用いられる．

学習においてはノードのルールにより分割されたサンプルの分割度合いを示す指標として Gini 係数[114]が用いられる．分割前のサンプル数を M_{before}，分割後に左に分けられたサンプル数を M_{left}，左に分けられたサンプルのうち Positive ラベルのサンプル数を $M_{\text{left positive}}$，それらが右の場合をそれぞれ，M_{right}，$M_{\text{right positive}}$ とすれば，分割後の Gini 係数は以下の式により計算される．

$$Gini=\frac{M_{\text{left}}}{M_{\text{before}}}\frac{M_{\text{left postivie}}}{M_{\text{left}}}\left(1-\frac{M_{\text{left postivie}}}{M_{\text{left}}}\right)+\frac{M_{\text{right}}}{M_{\text{before}}}\frac{M_{\text{right postivie}}}{M_{\text{right}}}\left(1-\frac{M_{\text{right postivie}}}{M_{\text{right}}}\right). \quad (4.7)$$

Gini 係数は小さいほど分割が良いことを表している．例えば，左に Positive ラベルのみ，右に Negative ラベルのみに分割された場合 Gini 係数は最小値 0 となる．

ここで, AdaBoost の弱識別器に Decision Tree を用いる場合の連携方法について述べる．図 4.1 の弱識別器の学習の 1:において，Decision Tree を学習する．その時，Gini 係数は式(4.1)の *Dist* に従って計算される．*Dist* には強識別器が誤答するほど大きくなる値が設定されているため, この値を用いて Gini 係数を計算すれば, 強識別器が誤答するサンプルに対して正答しやすい Decision Tree が学習される．FTOP の求解においては 4.1.1，4.1.2 に述べた手法により式(3.12)の *Classifier* を学習する．

4.1.3 AUC(Area Under the Curve)

提案法では L に AUC を用いる．AUC は横軸に False Positive Rate，縦軸に Miss Rate を指定した DET(Detection Error Trade-off)曲線により得られる面積である．False Positive Rate は Negative ラベルを Positive ラベルと誤った割合であり，Miss Rate は Positive ラベルを Negative ラベルと誤った割合である．閾値を変えるとそれに伴い False Positive Rate と Miss Rate が変わる. 閾値を大きくするに従い, Miss Rate が大きくなり，False Positive Rate が小さくなる．これは閾値を大きくするほど，未知のサンプルが Negative ラベルであると判定されるためである．この関係を用いて閾値を変化させると図 4.4 に示す曲線が得られる．この曲線が DET である．DET の面積即ち AUC を計算することにより，識別精度を比較できる．False Positive Rate も Miss Rate も誤りを表すため，面積は小さいほど識別精度がよいと考えられる. 単独の閾値に対する誤り率(全サンプルに対して誤答したサンプルの割合)に比べ，複数の閾値に対する識別精度を総合的に判断できる指標が AUC である．

ここで，閾値について説明する．式(3.12)の $\boldsymbol{z}^{(m_v)}$ は AdaBoost を用いる場合はス

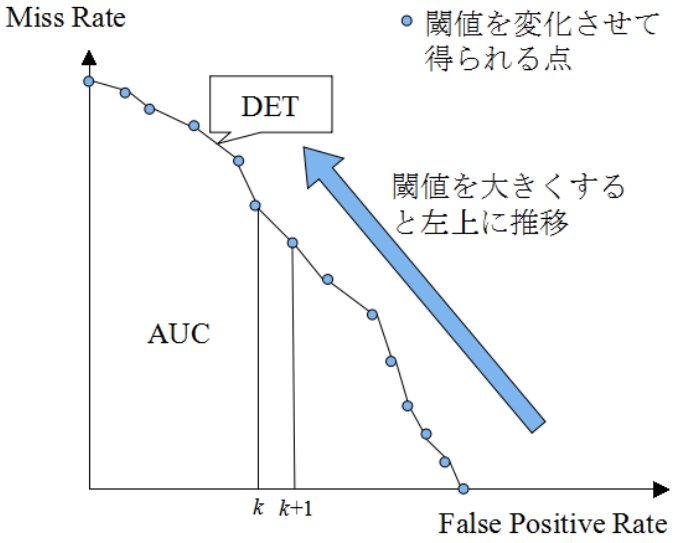


図 4.4　DET と AUC.

カラー値 $z^{(m_v)}$ となることに注意する．推定されるラベル *label**に Positive ラベルもしくは Negative ラベルを以下の式により $z^{(m_v)}$ に対して閾値 *th* を用いて割り当てる．

$$lebel^* = \begin{cases} Positive\ if\ \ z^{(m_v)} \geq th \\ Negative\ else \end{cases}. \tag{4.8}$$

ここで得られる DET は離散的な点により構成される．従って AUC は閾値の変化回数を k={1,2,...,K}，そのときの False Positive Rate を FPR_k，Miss Rate を MR_k とすれば積分の台形近似により以下の式となる．

$$AUC = \sum_{k=1}^{K-1} \frac{\left(MR_{k+1} - MR_k\right)\left(FNR_{k+1} + FNR_k\right)}{2}. \tag{4.9}$$

4.2 ノードに用いる処理の種類

$p \in P$ は歩行者識別において有効と考えられる様々な処理の種類を指定する．以降では，提案法にて採用した処理の種類，その処理の種類が持つ関数のパラメータ $\boldsymbol{\lambda}$ 及び $\boldsymbol{\lambda}$ に対する近傍操作について述べる．本提案における処理の種類は歩行者識別において有効性が報告された処理及びその処理を構成する処理から選定している．複数のチャンネル方向の成分 c に対して，要素毎の四則演算を用いる．以降では，$\boldsymbol{S}_c^j = \underline{\boldsymbol{S}}_{c(3)}^j \in \boldsymbol{R}^{W_S \times H_S}$ は関数 f_j に入力するテンソル $\underline{\boldsymbol{S}}^j$ を第 3 モード，すなわちチャンネル方向に対して第 c 成分にてスライスした行列である．以降，それ

ぞれの処理による画像の変換結果を図 4.5 を入力に例示する．それらは可視化のため画素値のダイナミックレンジを 0 から 255 に変換している．

4.2.1　表色系変換

表色系変換とは画像の RGB から色空間に変換することである．λ^* を一意のパラメータとすれば以下の式により与えられる．

$$\begin{aligned}&\underline{\boldsymbol{Y}}^{j} = f_{\text{convert color}}\left(\underline{\boldsymbol{I}}, \lambda^*\right),\\ &\lambda^* \in \boldsymbol{\lambda}_{\text{convert color}} = \left\{\text{GRAY}, \text{HSV}, \text{L*a*b*}, \text{L*u*v*}, \text{YCrCb}\right\} \subset \boldsymbol{\lambda}.\end{aligned} \tag{4.10}$$

例えば，モノクロに変換するグレイスケール変換は表色系変換の一種である．以下に, 提案法にて採用した表色系変換を示す. 提案法では, グレイスケール変換, HSV 変換，L*a*b*変換，L*u*v*変換，YCrCb 変換を用いる．以降，入力画像 $\underline{\boldsymbol{I}}$ の第 1 チャンネルを Red，第 2 チャンネルを Green，第 3 チャンネルを Blue とする．グレイスケール変換後の $\underline{\boldsymbol{I}}^{\text{GRAY}}$ は以下の式となる．

$$\begin{aligned}&\underline{\boldsymbol{I}}^{\text{GRAY}}{}_{1wh} = f_{\text{convert color}}\left(\underline{\boldsymbol{I}}, \text{GRAY}\right) = \begin{pmatrix}0.299 & 0.587 & 0.114\end{pmatrix}\begin{pmatrix}\underline{\boldsymbol{I}}_{1wh}\\ \underline{\boldsymbol{I}}_{2wh}\\ \underline{\boldsymbol{I}}_{3wh}\end{pmatrix},\\ &\forall w \in \{1,2,...,W\}, h \in \{1,2,...,H\}.\end{aligned} \tag{4.11}$$

グレイスケール変換は輝度の大きさに着目した変換である．

HSV(Hue，Saturation，Value)変換後の $\underline{\boldsymbol{I}}^{\text{HSV}}$ は以下の式となる．

$$\begin{aligned}&MAX\left(\underline{\boldsymbol{I}}\right) = \max\left(\underline{\boldsymbol{I}}_{1wh}, \underline{\boldsymbol{I}}_{2wh}, \underline{\boldsymbol{I}}_{3wh}\right),\ MIN\left(\underline{\boldsymbol{I}}\right) = \min\left(\underline{\boldsymbol{I}}_{1wh}, \underline{\boldsymbol{I}}_{2wh}, \underline{\boldsymbol{I}}_{3wh}\right),\\ &\underline{\boldsymbol{I}}^{\text{HSV}}{}_{\text{Value}wh} = f_{\text{convert color}}\left(\underline{\boldsymbol{I}}, \text{HSV}\right)_{\text{Value}} = MAX\left(\underline{\boldsymbol{I}}\right),\\ &\underline{\boldsymbol{I}}^{\text{HSV}}{}_{\text{Saturation}wh} = f_{\text{convert color}}\left(\underline{\boldsymbol{I}}, \text{HSV}\right)_{\text{Saturation}} = \begin{cases}0 & if\ MAX\left(\underline{\boldsymbol{I}}\right) = MIN\left(\underline{\boldsymbol{I}}\right) = 0\\ \dfrac{MAX\left(\underline{\boldsymbol{I}}\right) - MIN\left(\underline{\boldsymbol{I}}\right)}{MAX\left(\underline{\boldsymbol{I}}\right)} & \end{cases},\\ &\underline{\boldsymbol{I}}^{\text{HSV}}{}_{\text{Hue}wh} = f_{\text{convert color}}\left(\underline{\boldsymbol{I}}, \text{HSV}\right)_{\text{Hue}}\\ &= \begin{cases}0 & if\ MAX\left(\underline{\boldsymbol{I}}\right) = MIN\left(\underline{\boldsymbol{I}}\right) = 0\\ 60 \times \dfrac{\underline{\boldsymbol{I}}_{wh2} - \underline{\boldsymbol{I}}_{wh3}}{MAX\left(\underline{\boldsymbol{I}}\right) - MIN\left(\underline{\boldsymbol{I}}\right)} & if\ \underline{\boldsymbol{I}}_{wh1} = MAX\left(\underline{\boldsymbol{I}}\right)\\ 60 \times \dfrac{\underline{\boldsymbol{I}}_{wh3} - \underline{\boldsymbol{I}}_{wh1}}{MAX\left(\underline{\boldsymbol{I}}\right) - MIN\left(\underline{\boldsymbol{I}}\right)} + 120 & if\ \underline{\boldsymbol{I}}_{wh2} = MAX\left(\underline{\boldsymbol{I}}\right)\\ 60 \times \dfrac{\underline{\boldsymbol{I}}_{wh1} - \underline{\boldsymbol{I}}_{wh2}}{MAX\left(\underline{\boldsymbol{I}}\right) - MIN\left(\underline{\boldsymbol{I}}\right)} + 240 & if\ \underline{\boldsymbol{I}}_{wh3} = MAX\left(\underline{\boldsymbol{I}}\right)\end{cases},\\ &\forall w \in \{1,2,...,W\},\ \forall h \in \{1,2,...,H\}.\end{aligned} \tag{4.12}$$

図 4.5　入力画像.

HSV 変換は色相，彩度，明度に変換する.

L*a*b*変換では XYZ 変換から均等色空間に変換される．XYZ 変換後の $\underline{\boldsymbol{I}}^{\mathrm{XYZ}}$ は以下の通りである.

$$\begin{pmatrix} \underline{\boldsymbol{I}}^{\mathrm{XYZ}}{}_{\mathrm{X}\,wh} \\ \underline{\boldsymbol{I}}^{\mathrm{XYZ}}{}_{\mathrm{Y}\,wh} \\ \underline{\boldsymbol{I}}^{\mathrm{XYZ}}{}_{\mathrm{Z}\,wh} \end{pmatrix} = \begin{pmatrix} 0.412453 & 0.357580 & 0.180423 \\ 0.212671 & 0.715160 & 0.072169 \\ 0.019334 & 0.119193 & 0.950227 \end{pmatrix} \begin{pmatrix} \underline{\boldsymbol{I}}_{1wh} \\ \underline{\boldsymbol{I}}_{2wh} \\ \underline{\boldsymbol{I}}_{3wh} \end{pmatrix}, \quad \forall w \in \{1,2,...,W\}, \forall h \in \{1,2,...,H\}. \tag{4.13}$$

次に，I'を用いて L*a*b*変換した後の $\underline{\boldsymbol{I}}^{\mathrm{L^*a^*b^*}}$ は以下の式となる.

$$\underline{\boldsymbol{I}}^{\mathrm{L^*a^*b^*}}{}_{\mathrm{L}^* wh} = f_{\mathrm{convert\,color}}\left(\underline{\boldsymbol{I}}, \mathrm{L^*a^*b^*}\right)_{\mathrm{L}^*} = \begin{cases} 116\sqrt[3]{\underline{\boldsymbol{I}}^{\mathrm{XYZ}}{}_{\mathrm{Y}\,wh}} - 16 & if\ \underline{\boldsymbol{I}}^{\mathrm{XYZ}}{}_{\mathrm{Y}\,wh} > 0.008856 \\ 903.3\underline{\boldsymbol{I}}^{\mathrm{XYZ}}{}_{\mathrm{Y}\,wh} & else \end{cases},$$

$$\underline{\boldsymbol{I}}^{\mathrm{L^*a^*b^*}}{}_{\mathrm{a}^* wh} = f_{\mathrm{convert\,color}}\left(\underline{\boldsymbol{I}}, \mathrm{L^*a^*b^*}\right)_{\mathrm{a}^*} = 500\left(f_{\mathrm{liner\ or\ nonliner}}\left(\frac{\underline{\boldsymbol{I}}^{\mathrm{XYZ}}{}_{\mathrm{X}\,wh}}{X_n}\right) - f_{\mathrm{liner\ or\ nonliner}}\left(\underline{\boldsymbol{I}}^{\mathrm{XYZ}}{}_{\mathrm{Y}\,wh}\right)\right),$$

$$\underline{\boldsymbol{I}}^{\mathrm{L^*a^*b^*}}{}_{\mathrm{b}^* wh} = f_{\mathrm{convert\,color}}\left(\underline{\boldsymbol{I}}, \mathrm{L^*a^*b^*}\right)_{\mathrm{b}^*} = 200\left(f_{\mathrm{liner\ or\ nonliner}}\left(\underline{\boldsymbol{I}}^{\mathrm{XYZ}}{}_{\mathrm{Y}\,wh}\right) - f_{\mathrm{liner\ or\ nonliner}}\left(\frac{\underline{\boldsymbol{I}}^{\mathrm{XYZ}}{}_{\mathrm{Z}\,wh}}{Z_n}\right)\right),$$

$$f_{\mathrm{liner\ or\ nonliner}}(g) = \begin{cases} \sqrt[3]{g} & if\ g > 0.008856 \\ 7.787g + \dfrac{16}{116} & else \end{cases},$$

$$\forall w \in \{1,2,...,W\},\ \forall h \in \{1,2,...,H\}. \tag{4.14}$$

ここで X_n=0.950456，Z_n=1.088754 である．L*a*b*変換では知覚的に等歩度が大きくなる(人の知覚による色の差が，色の空間内の距離に対応する)ように変換される．X_n，Y_n，Z_n はホワイトポイントの三刺激値である.

L*u*v*変換後の $\underline{\boldsymbol{I}}^{\mathrm{L^*u^*v^*}}$ は以下の式となる.

$$
\begin{aligned}
&\underline{\boldsymbol{I}}^{\mathrm{L^*u^*v^*}}{}_{\mathrm{L^*}wh} = f_{\mathrm{convert\,color}}\left(\underline{\boldsymbol{I}}, \mathrm{L^*u^*v^*}\right)_{\mathrm{L^*}} = \begin{cases} 116\sqrt[3]{\underline{\boldsymbol{I}}^{\mathrm{XYZ}}{}_{\mathrm{Y}\,wh}} & if\ \underline{\boldsymbol{I}}^{\mathrm{XYZ}}{}_{\mathrm{Y}\,wh} > 0.008856 \\ 903.3\underline{\boldsymbol{I}}^{\mathrm{XYZ}}{}_{\mathrm{Y}\,wh} & else \end{cases}, \\
&\underline{\boldsymbol{I}}^{\mathrm{L^*u^*v^*}}{}_{\mathrm{u^*}wh} = f_{\mathrm{convert\,color}}\left(\underline{\boldsymbol{I}}, \mathrm{L^*u^*v^*}\right)_{\mathrm{u^*}} = 13\underline{\boldsymbol{I}}^{\mathrm{L^*u^*v^*}}{}_{\mathrm{L^*}wh}\left(u' - u_n'\right), \\
&\underline{\boldsymbol{I}}^{\mathrm{L^*u^*v^*}}{}_{\mathrm{v^*}wh} = f_{\mathrm{convert\,color}}\left(\underline{\boldsymbol{I}}, \mathrm{L^*u^*v^*}\right)_{\mathrm{v^*}} = 13\underline{\boldsymbol{I}}^{\mathrm{L^*u^*v^*}}{}_{\mathrm{L^*}wh}\left(v' - v_n'\right), \\
&u' = \frac{4\underline{\boldsymbol{I}}^{\mathrm{XYZ}}{}_{\mathrm{X}\,wh}}{\underline{\boldsymbol{I}}^{\mathrm{XYZ}}{}_{\mathrm{X}\,wh} + 15\underline{\boldsymbol{I}}^{\mathrm{XYZ}}{}_{\mathrm{Y}\,wh} + 3\underline{\boldsymbol{I}}^{\mathrm{XYZ}}{}_{\mathrm{Z}\,wh}}, \; v' = \frac{9\underline{\boldsymbol{I}}^{\mathrm{XYZ}}{}_{\mathrm{Y}\,wh}}{\underline{\boldsymbol{I}}^{\mathrm{XYZ}}{}_{\mathrm{X}\,wh} + 15\underline{\boldsymbol{I}}^{\mathrm{XYZ}}{}_{\mathrm{Y}\,wh} + 3\underline{\boldsymbol{I}}^{\mathrm{XYZ}}{}_{\mathrm{Z}\,wh}}, \\
&\forall w \in \{1,2,...,W\},\ \forall h \in \{1,2,...,H\}.
\end{aligned}
\tag{4.15}
$$

ここで，u_n=0.19793943，v_n=0.46831096 である．L*u*v*は L*a*b*と同様に人の知覚による色の差が，色の空間内の距離に対応するように変換される．L*a*b*との違いは知覚的な等歩度を調整された時に，アクアダムの楕円偏差(楕円の中心の色と区別することができない色の範囲)を考慮して知覚的な等歩度を調整したかである．L*a*b*は後者である．従って，知覚的な等歩度は L*a*b*の方が大きい．

YCrCb 変換後の $\underline{\boldsymbol{I}}^{\mathrm{YCrCb}}$ は以下の式となる．

$$
\begin{aligned}
&\underline{\boldsymbol{I}}^{\mathrm{YCrCb}}{}_{\mathrm{Y}\,wh} = f_{\mathrm{convert\,color}}\left(\underline{\boldsymbol{I}}, \mathrm{YCrCb}\right)_{\mathrm{Y}} = \underline{\boldsymbol{I}}^{\mathrm{GRAY}}{}_{1wh}, \\
&\underline{\boldsymbol{I}}^{\mathrm{YCrCb}}{}_{\mathrm{Cr}\,wh} = f_{\mathrm{convert\,color}}\left(\underline{\boldsymbol{I}}, \mathrm{YCrCb}\right)_{\mathrm{Cr}} = 0.713\left(\underline{\boldsymbol{I}}_{1wh} - \underline{\boldsymbol{I}}^{\mathrm{YCrCb}}{}_{\mathrm{Y}\,wh}\right) + 0.5, \\
&\underline{\boldsymbol{I}}^{\mathrm{YCrCb}}{}_{\mathrm{Cb}\,wh} = f_{\mathrm{convert\,color}}\left(\underline{\boldsymbol{I}}, \mathrm{YCrCb}\right)_{\mathrm{Cb}} = 0.564\left(\underline{\boldsymbol{I}}_{3wh} - \underline{\boldsymbol{I}}^{\mathrm{YCrCb}}{}_{\mathrm{Y}\,wh}\right) + 0.5, \\
&\forall w \in \{1,2,...,W\},\ \forall h \in \{1,2,...,H\}.
\end{aligned}
\tag{4.16}
$$

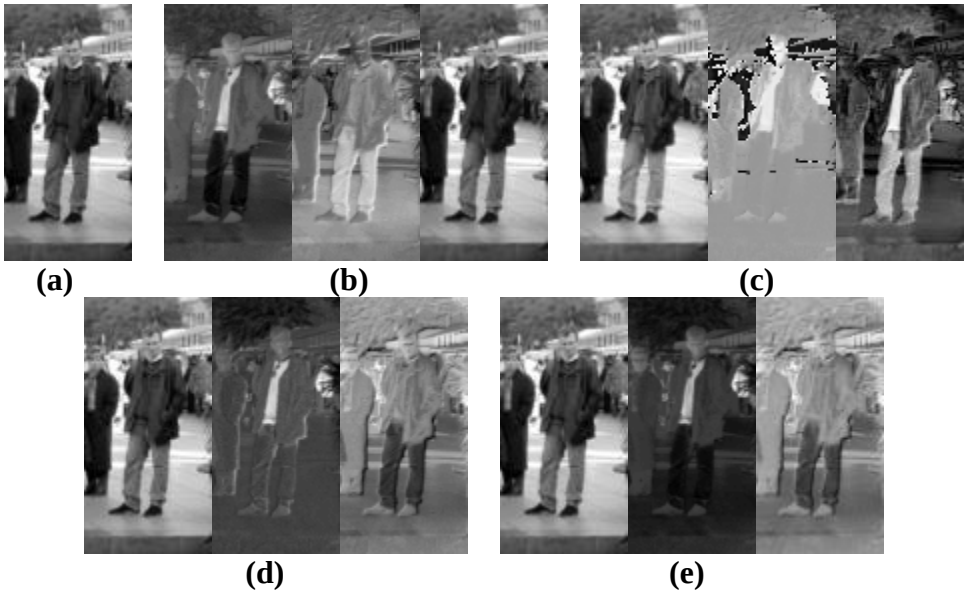


図 4.6　表色系変換. (a)グレイスケール変換；(b)HSV 変換, 左から Hue, Saturation, V；(c)L*a*b*変換, 左から L*, a*, b*；(d)L*u*v*変換, 左から L*, u*, v*；(e)YCrCb 変換，左から Y，Cr，Cb.

YCrCb 変換は L*a*b*変換や L*u*v*変換とは異なり，人が知覚する感度の違いに着目した変換である．人は輝度の変化には敏感であるが，色差(色の変化)には鈍感であるという性質がある．この性質を生かし，輝度には大きい解像度を与え，色差には小さい解像度を与えることにより人の知覚に合ったデータ圧縮が可能となる．そのために，RGB を輝度と色差に分ける必要がある．これが，YCrCb である．なお，YUV も輝度と色差に着目した変換であるが，YCrCb との違いは値域の違いのみ(圧縮コーデックの違いによる)のため，提案法では YCrCb を用いる．

表色系変換を処理の種類とすれば，表色系変換における関数のパラメータ $\boldsymbol{\lambda}$ には上記の変換式のいずれを使用するかが含まれる．従って，$\boldsymbol{\lambda}$ の近傍操作では表色系変換のいずれかの式から別の式へと変更する．例えば，HSV 変換から L*a*b*への変換へ変更することが近傍操作となる．表色系の変換は RGB 空の変換を前提としているため，FTN の始点 TOP からの接続のみ可能とする．

図 4.6 に各表色系変換の結果を示す．

4.2.2 四則演算

四則演算では加算，減算，乗算，除算について説明する．

$$\underline{\boldsymbol{Y}}^j = f_{\text{arithmetic}}\left(\underline{\boldsymbol{S}}^j, \lambda^*\right),\ \lambda^* \in \boldsymbol{\lambda}_{\text{arithmetic}} = \{\text{Plus}, \text{Minus}, \text{Multiply}, \text{Divide}\} \subset \boldsymbol{\lambda}\,. \quad (4.17)$$

加算は以下の式を用いる．減算は-1 倍して加算することと同じであるため式を省略する．加算は後述する Box フィルタの出力を複数計算し，その出力を加算することにより Box フィルタ近似[61]を実現することができる．なお，$C_S \geqq 2$ である．

$$\underline{\boldsymbol{Y}}^j = f_{\text{arithmetic}}\left(\underline{\boldsymbol{S}}^j, \text{Plus}\right) = \sum_{c=1}^{C_S} \underline{\boldsymbol{S}}_c^j. \quad (4.18)$$

乗算は以下の式を用いる． $\underline{\boldsymbol{S}}_1^j, \underline{\boldsymbol{S}}_2^j, ..., \underline{\boldsymbol{S}}_{C_S}^j$ は関数 f_j に入力されるテンソルを構成する行列のある一つ表し，⊗は直積である．乗算を用いれば[31]のような共起を表現できる．

$$\underline{\boldsymbol{Y}}^j = f_{\text{arithmetic}}\left(\underline{\boldsymbol{S}}^j, \text{Multiply}\right) = \underline{\boldsymbol{S}}_1^j \otimes \underline{\boldsymbol{S}}_2^j ... \otimes \underline{\boldsymbol{S}}_{C_S}^j. \quad (4.19)$$

除算は以下の式を用いる．/は要素毎に割り算することを表す．

$$\begin{aligned} &\underline{\boldsymbol{Y}}_{1wh}^j = f_{\text{arithmetic}}\left(\underline{\boldsymbol{S}}^j, \text{Divide}\right) = \underline{\boldsymbol{S}}_{1wh}^j / \underline{\boldsymbol{S}}_{2wh}^j ... / \underline{\boldsymbol{S}}_{C_S wh}^j\ , \\ &\forall w \in \{1,2,...,W_S\},\ \forall h \in \{1,2,...,H_S\}\,. \end{aligned} \quad (4.20)$$

除算を用いれば，例えば，HOG（後述）に用いられる正規化を実現できる．

四則演算を処理の種類とすれば，四則演算における関数のパラメータ $\boldsymbol{\lambda}$ には上

(a)　　(b)　　(c)

図 4.7　R, G 成分における四則演算．(a)R+G，(b)R⊗G，(c)R/G.

(a)　　(b)　　(c)　　(d)

図 4.8　R, G 成分におけるノルム．(a)L_0，(b)L_1，(c)L_2，(d)L_∞ .

記のいずれを使用するかが含まれる．従って，λ の近傍操作ではある四則演算から別の四則演算へと変更する．例えば，加算から乗算に変更することが近傍操作となる．

図 4.7 に入力画像の R と G 成分を対象にした四則演算の結果を示す．ただし，減算は-1 倍しての加算のため省略する．

4.2.3　ノルム

ノルムの計算式を説明する．ノルムを用いれば，例えば，勾配の大きさを求める式を実現できる．ノルムでは L_0，L_1，L_2，L_∞ ノルムを用いる．なお，$C_S \geqq 2$ である．

$$\underline{\boldsymbol{Y}}^j = f_{\mathrm{norm}}\left(\underline{\boldsymbol{S}}^j, \lambda^*\right),\ \lambda^* \in \boldsymbol{\lambda}_{\mathrm{norm}} = \left\{\mathrm{L}_0, \mathrm{L}_1, \mathrm{L}_2, \mathrm{L}_\infty\right\} \subset \boldsymbol{\lambda}\,. \tag{4.21}$$

L_0 ノルムは以下の式を用いる．

$$\underline{\boldsymbol{Y}}^j_{1wh} = f_{\mathrm{norm}}\left(\underline{\boldsymbol{S}}^j, \mathrm{L}_0\right) = \sum_{c=1}^{C_S} 1 \quad if\ \underline{\boldsymbol{S}}^j_{cwh} \neq 0,\ \forall w \in \left\{1,2,...,W_S\right\},\ \forall h \in \left\{1,2,...,H_S\right\}. \tag{4.22}$$

L_1 ノルムは以下の式を用いる．

$$\underline{\boldsymbol{Y}}^j_{1wh} = f_{\mathrm{norm}}\left(\underline{\boldsymbol{S}}^j, \mathrm{L}_1\right) = \sum_{c=1}^{C_S} \left|\underline{\boldsymbol{S}}^j_{cwh}\right|,\ \forall w \in \left\{1,2,...,W_S\right\},\ \forall h \in \left\{1,2,...,H_S\right\}. \tag{4.23}$$

L_2 ノルムは以下の式を用いる．

$$\underline{\boldsymbol{Y}}_{1wh}^{j} = f_{\mathrm{norm}}\left(\underline{\boldsymbol{S}}^{j}, \mathrm{L}_2\right) = \sqrt{\sum_{c=1}^{C_S}\left(\underline{\boldsymbol{S}}_{cwh}^{j}\right)^2},\ \forall w \in \{1,2,...,W_S\},\ \forall h \in \{1,2,...,H_S\}. \tag{4.24}$$

L_∞ ノルムは以下の式を用いる．

$$\begin{aligned}&\underline{\boldsymbol{Y}}_{1wh}^{j} = f_{\mathrm{norm}}\left(\underline{\boldsymbol{S}}^{j}, \mathrm{L}_\infty\right) = \max\left\{\underline{\boldsymbol{S}}_{cwh}^{j},\ \forall c \in \{1,2,...,C_S\}\right\},\\ &\forall w \in \{1,2,...,W_S\},\ \forall h \in \{1,2,...,H_S\}.\end{aligned} \tag{4.25}$$

ノルムを処理の種類とすれば，ノルムにおける関数のパラメータ $\boldsymbol{\lambda}$ には上記のいずれを使用するかが含まれる．従って，$\boldsymbol{\lambda}$ の近傍操作ではあるノルムの式から別のノルムの式へと変更する．例えば，L_0 ノルムから L_1 ノルムに変更することが近傍操作となる．

図 4.8 に入力画像の R と G を対象に各ノルムを用いた計算結果を示す．(a)は黒い画像となっているが，これは L_0 ノルムの結果，全ての画素値が 1 となり，ダイナミックレンジの最小値を 0 として表示しているためである．(b)と(c)は同じに見えるが，これは可視化のためにダイナミックレンジを調整した結果であり，実際の画素値は異なる．また(d)は(b)，(c)より明るい部分があるこれは，L_∞ ノルムにより R と G の大きいほうが画素値として採用されるためである．

4.2.4　方向

方向の計算式を説明する．方向を用いれば，例えば，勾配の方向を求める式を実現できる．方向では arctan と arctan2 を用いる．

$$\underline{\boldsymbol{Y}}^{j} = f_{\mathrm{orientation}}\left(\underline{\boldsymbol{S}}_{1(3)}^{j}, \underline{\boldsymbol{S}}_{2(3)}^{j}, \lambda^*\right),\ \lambda^* \in \boldsymbol{\lambda}_{\mathrm{orientataion}} = \{\arctan, \arctan 2\} \subset \boldsymbol{\lambda}. \tag{4.26}$$

グレイスケール変換した画像の勾配の方向を計算することを考えてみる．arctan では暗い画素から明るい画素への勾配方向と暗い画素から明るい画素への勾配方向は等しい．しかし，arctan2 では，それが異なる．どちらが適しているかは識別対象による．

arctan を用いた方向は以下の式となる．なお，C_S=2 である．

$$\begin{aligned}&\underline{\boldsymbol{Y}}_{1wh}^{j} = f_{\mathrm{orientation}}\left(\underline{\boldsymbol{S}}_{1wh}^{j}, \underline{\boldsymbol{S}}_{2wh}^{j}, \arctan\right) = \arctan\left(\frac{\underline{\boldsymbol{S}}_{1wh}^{j}}{\underline{\boldsymbol{S}}_{2wh}^{j}}\right),\\ &\forall w \in \{1,2,...,W_S\},\ \forall h \in \{1,2,...,H_S\}.\end{aligned} \tag{4.27}$$

arctan は割る数と割られる数それぞれの符号は考慮せず，値域は $(-\pi/2, \pi/2)$ である．

arctan2 を用いた方向は以下の式となる．

$$\underline{\boldsymbol{Y}}_{1wh}^{j} = f_{\text{orientation}}\left(\underline{\boldsymbol{S}}_{c1wh}^{j}, \underline{\boldsymbol{S}}_{c2wh}^{j}, \arctan 2\right) = \arctan 2\left(\underline{\boldsymbol{S}}_{c1wh}^{j}, \underline{\boldsymbol{S}}_{c2wh}^{j}\right),$$

$$\arctan 2(s_1, s_2) = \begin{cases} \arctan(s_1 / s_2) & s_2 > 0 \\ \arctan(s_1 / s_2) + \pi & s_1 \geq 0, s_2 < 0 \\ \arctan(s_1 / s_2) - \pi & s_1 < 0, s_2 < 0 \\ \pi / 2 & s_1 > 0, s_2 = 0 \\ -\pi / 2 & s_1 < 0, s_2 = 0 \\ 0 & s_1 = 0, s_2 = 0 \end{cases}, \tag{4.28}$$

$$s_1, s_2 \in R, \forall w \in \{1,2,...,W_S\}, \forall h \in \{1,2,...,H_S\}.$$

arctan2 は割る数と割られる数それぞれの符号を考慮し，値域は$(-\pi, \pi]$である．

方向を処理の種類とすれば，方向における関数のパラメータλには上記のいずれを使用するかが含まれる．従って，λの近傍操作では arctan もしくは arctan2 から arctan2 もしくは arctan へと変更する．

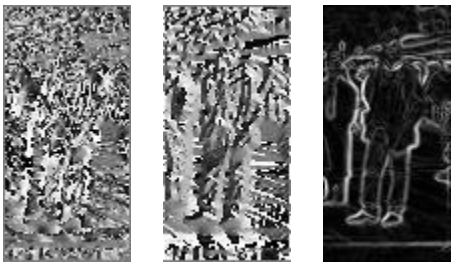
図 4.9 に入力画像の R と G に対する arctan，arctan2 の変換結果を示す．R と G は全て 0 以上となるため, この例では(a)arctan の結果と(b)arctan2 の結果は等しい．図 4.10 に勾配の例を示すが，同図(a)arctan と(b)arctan2 の違いが現れている．

4.2.5 勾配

勾配の計算式を説明する．勾配は HOG[29]や Covariance[44]に用いられている

(a)　(b)

図 4.9　R, G 成分における方向．(a)arctan，(b)arctan2.



(a)　(b)　(c)

図 4.10　R 成分における勾配．(a)勾配の方向 arctan，(b)勾配の方向 arctan2，(c)勾配の大きさ.

処理である．勾配の計算は $f_{\text{norm}}\left(\underline{\boldsymbol{S}}^{j}, \mathrm{L}_2\right)$，$f_{\text{orientation}}\left(\underline{\boldsymbol{S}}^{j}, \boldsymbol{\lambda}_{\text{orientation}}\right)$から計算されるため，勾配関数が持つパラメータ $\boldsymbol{\lambda}_{\text{gradient}}$ は $\boldsymbol{\lambda}_{\text{orientation}}$ と一致する．従って，近傍操作も方向の計算と同様である．なお，C_S=1 である．

$$
\begin{aligned}
&\underline{\boldsymbol{Y}}^{j}_{\text{orientation}} = f_{\text{gradient}}\left(\underline{\boldsymbol{S}}^{j}, \lambda*\right)_{\text{orientation}} = f_{\text{orientation}}\left(\Delta h, \Delta w, \lambda*\right),\\
&\underline{\boldsymbol{Y}}^{j}_{\text{magnitude}} = f_{\text{gradient}}\left(\underline{\boldsymbol{S}}^{j}, \lambda*\right)_{\text{magnitude}} = f_{\text{norm}}\left(\Delta h, \Delta w, \mathrm{L}_2\right),\\
&\Delta w = \underline{\boldsymbol{S}}^{j}_{1w-1h} - \underline{\boldsymbol{S}}^{j}_{1w+1h}, \Delta h = \underline{\boldsymbol{S}}^{j}_{1wh-1} - \underline{\boldsymbol{S}}^{j}_{1wh+1},\\
&\forall w \in \{1,2,\ldots,W_S\}, \forall h \in \{1,2,\ldots,H_S\},\\
&\lambda* \in \boldsymbol{\lambda}_{\text{gradient}} = \boldsymbol{\lambda}_{\text{orientation}}.
\end{aligned}
\tag{4.29}
$$

図 4.10 に入力画像の R 成分に対する勾配の例を示す．同図(a)と(b)に arctan と arctan2 の違いが現れている．arctan の場合，勾配の方向は明暗，暗明どちらも同じになるが，arctan2 の場合は異なる．歩行者を識別したい場合，明暗，暗明を反転させても撮影対象が歩行者であることに変わりはないことを考えると arctan の方が適していることが期待される．

4.2.6 畳み込みフィルタ

畳み込みフィルタの計算式を説明する．畳み込みフィルタは畳み込みの重みの値によって様々なフィルタを実現できる．提案法では以下の畳み込みの重み計算のアルゴリズム $\boldsymbol{\lambda}_{\text{convolution type}}$ を用いる．畳み込みフィルタの場合，関数が持つパラメータは一つではないため，一意のパラメータは集合 $\boldsymbol{\lambda}*$ で表す．

$$
\begin{aligned}
&\underline{\boldsymbol{Y}}^{j} = f_{\text{convolution}}\left(\underline{\boldsymbol{S}}^{j}_{1(3)}, \boldsymbol{\lambda}*\right),\\
&\boldsymbol{\lambda}* = \left\{\lambda_{\text{type}}, \boldsymbol{\lambda}_{\text{param}}\right\},\\
&\lambda_{\text{type}} \in \boldsymbol{\lambda}_{\text{convolution type}} = \{\text{Gaussian, DoG, Gabor, Derivative}\} \subset \boldsymbol{\lambda},\\
&\boldsymbol{\lambda}_{\text{param}} \subset \boldsymbol{\lambda}_{\lambda_{\text{type}}\ \text{param}} \subset \left\{\boldsymbol{\lambda}_{\text{Gaussian param}}, \boldsymbol{\lambda}_{\text{DoG param}}, \boldsymbol{\lambda}_{\text{Gabor param}}, \boldsymbol{\lambda}_{\text{Derivative param}}\right\} \subset \boldsymbol{\lambda},\\
&\boldsymbol{\lambda}_{\text{Gaussian param}} = \{\boldsymbol{Width}, \boldsymbol{Height}, \boldsymbol{Sigma}\},\\
&width \in \boldsymbol{Width} \subset \boldsymbol{Z},\ height \in \boldsymbol{Height} \subset \boldsymbol{Z},\ sigma \in \boldsymbol{Sigma} \subset \boldsymbol{R},\\
&\boldsymbol{\lambda}_{\text{DoG param}} = \{\boldsymbol{Width}, \boldsymbol{Height}, \boldsymbol{Sigma}1, \boldsymbol{Sigma}2\},\\
&width \in \boldsymbol{Width} \subset \boldsymbol{Z}, height \in \boldsymbol{Height} \subset \boldsymbol{Z},\\
&sigma1 \in \boldsymbol{Sigma}1 \subset \boldsymbol{R}, sigma1 \in \boldsymbol{Sigma}1 \subset \boldsymbol{R},\\
&\boldsymbol{\lambda}_{\text{Gabor param}} = \{\boldsymbol{Width}, \boldsymbol{Height}, \boldsymbol{Sigma}, \boldsymbol{Theta}, \boldsymbol{Gamma}\},\\
&width \in \boldsymbol{Width} \subset \boldsymbol{Z}, height \in \boldsymbol{Height} \subset \boldsymbol{Z}, sigma \in \boldsymbol{Sigma} \subset \boldsymbol{R},\\
&theta \in \boldsymbol{Theta} \subset \boldsymbol{R}, gamma \in \boldsymbol{Gamma} \subset \boldsymbol{R},\\
&\boldsymbol{\lambda}_{\text{Derivative param}} = \{\text{D}w, \text{D}h, \text{Sobel\_}w, \text{Sobel\_}h, \text{Scharr\_}w, \text{Scharr\_}h,\\
&\qquad\qquad \text{Prewitt\_}ru, \text{Prewitt\_}rd, \text{Laplacian\_4, Laplacian\_8}\}.
\end{aligned}
\tag{4.30}
$$

Z は実数集合である．ここで，畳み込みフィルタは以下により計算される．なお，C_S=1 である．

$$
\begin{aligned}
&\underline{\boldsymbol{Y}}^{j} = f_{\text{convolution}}\left(\underline{\boldsymbol{S}}^{j}\right) = \sum_{\bar{h}=-height/2}^{height/2} \sum_{\bar{w}=-width/2}^{width/2} \boldsymbol{A}_{\bar{w}\bar{h}}\, \underline{\boldsymbol{S}}^{j}_{1w-\bar{w}\,h-\bar{h}} \ , \\
&\forall w \in \{width/2, width/2+1, ..., W_S - width/2\}, \\
&\forall h \in \{height/2, height/2+1, ..., W_H - height/2\}, \\
&\boldsymbol{A} = f_{\boldsymbol{A}}(\boldsymbol{\lambda}_{\text{convolution}}), \boldsymbol{A} \in \boldsymbol{R}^{width \times height} \ , \\
&size = \{width, height\}.
\end{aligned}
\tag{4.31}
$$

以下に $\boldsymbol{\lambda}_{\text{type}}$ 下毎の $\boldsymbol{A}$ の計算を示す．

$\boldsymbol{\lambda}_{\text{type}}$=Gaussian の場合，$\boldsymbol{A}$ は以下の式により計算される．

$$
\begin{aligned}
&\boldsymbol{A} = f_{\boldsymbol{A}}(\text{Gaussian}, width, height, sigma) \\
&\boldsymbol{A}_{\bar{w}\bar{h}} = \text{Gaussian}(\bar{w}, \bar{h}, sigma), \\
&\forall \bar{w} = \{-width/2, -width/2+1, ..., width/2\}, \\
&\forall \bar{h} = \{-height/2, -height/2+1, ..., height/2\}, \\
&\text{Gaussian}(\bar{w}, \bar{h}, sigma) = \frac{1}{2\pi\, sigma^2} \exp\left(-\frac{\bar{w}^2 + \bar{h}^2}{2sigma^2}\right).
\end{aligned}
\tag{4.32}
$$

$\boldsymbol{\lambda}_{\text{type}}$=DoG(Difference of Gaussian)の場合，$\boldsymbol{A}$ は以下の式により計算される．

$$
\begin{aligned}
&\boldsymbol{A} = f_{\boldsymbol{A}}(\text{DoG}, width, height, sigma1, sigma2), \\
&\boldsymbol{A}_{\bar{w}\bar{h}} = \text{Gaussian}(\bar{w}, \bar{h}, sigma1) - \text{Gaussian}(\bar{w}, \bar{h}, sigma2), \\
&sigma2 = \tau\, sigma1, \\
&\forall \bar{w} = \{-width/2, -width/2+1, ..., width/2\}, \\
&\forall \bar{h} = \{-height/2, -height/2+1, ..., height/2\}.
\end{aligned}
\tag{4.33}
$$

$\boldsymbol{\lambda}_{\text{type}}$=Gabor の場合，$\boldsymbol{A}$ は以下の式により計算される．

$$
\begin{aligned}
&f_{\boldsymbol{A}}(\text{Gabor}, width, height, sigma, theta, gamma, lambda) = \boldsymbol{A}, \\
&\boldsymbol{A}_{\bar{w}\bar{h}} = \text{Gabor}(\bar{w}, \bar{h}, sigma, theta, gamma, lambda), \\
&\text{Gabor}(\bar{w}, \bar{h}, sigma, theta, gamma, lambda) = \exp\left(\frac{\bar{w}' + gamma^2 \bar{h}'^2}{2sigma^2}\right) \cos\left(\frac{2\pi \bar{w}'}{lambda} + \psi\right), \\
&\bar{w}' = w\cos(theta) + y\sin(theta), \\
&\bar{h}' = -h\sin(theta) + y\cos(theta), \\
&\forall \bar{w} = \{-width/2, -width/2+1, ..., width/2\}, \\
&\forall \bar{h} = \{-height/2, -height/2+1, ..., height/2\}.
\end{aligned}
\tag{4.34}
$$

ここで$\psi = 2/\pi$とする．ψは位相ずれであるが画像中フィルタをずらしながら適用するため，提案法では位相ずれは固定する．

λ_{type}=Derivative の場合，$\boldsymbol{A}$ は以下の式により計算される．$\boldsymbol{\lambda}_{\text{Derivative type}}$のそれぞれについて説明する．D*w*，D*h* はそれぞれ横方向，縦方向の 1 次微分フィルタである. 勾配を計算する時に用いられたり, Texton[9]の一部として用いられたりする．Sobel_*w*，Sobel_*h* は横方向，縦方向の Sobel フィルタである．このフィルタは 1 次微分フィルタに対して, 計算に含める範囲が D*w*, D*h* よりも広い. これにより, 線として特徴を捉えることができる．同様の考え方において，Sobel よりもさらに強く線として特徴を捉えるフィルタが Scharr フィルタ(Scharr_*w*，Scharr_*h*)である．また，斜め方向も特徴として捉えるために Prewitt_*ru*，Prewitt_*rd* を用いている．ここで *ru* は right up，*lu* は left up であり，本節において斜めの向きを定義した記号である．Laplacian_4，Laplacian_8 は 2 次微分である．変化量の変化量を捉えるため，エッジ部分に反応する傾向にある．

$$\boldsymbol{A} = f_{\boldsymbol{A}}(\text{Derivative}, \text{D}w) = \begin{pmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}, \boldsymbol{A} = f_{\boldsymbol{A}}(\text{Derivative}, \text{D}h) = \begin{pmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix},$$

$$\boldsymbol{A} = f_{\boldsymbol{A}}(\text{Derivative}, \text{Sobel\_}w) = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}, \boldsymbol{A} = f_{\boldsymbol{A}}(\text{Derivative}, \text{Sobel\_}h) = \begin{pmatrix} -1 & -2 & 1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix},$$

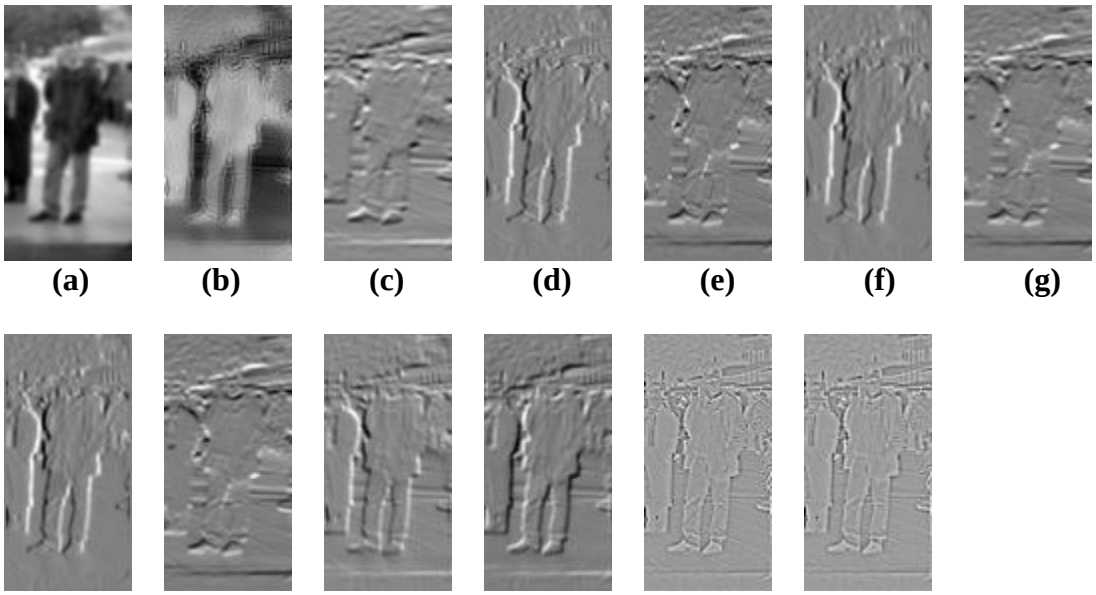
$$\boldsymbol{A} = f_{\boldsymbol{A}}(\text{Derivative}, \text{Scharr\_}w) = \begin{pmatrix} -3 & 0 & 3 \\ -10 & 0 & 10 \\ -3 & 0 & 3 \end{pmatrix}, \boldsymbol{A} = f_{\boldsymbol{A}}(\text{Derivative}, \text{Scharr\_}h) = \begin{pmatrix} -3 & -10 & -3 \\ 0 & 0 & 0 \\ 3 & 10 & 3 \end{pmatrix},$$

$$\boldsymbol{A} = f_{\boldsymbol{A}}(\text{Derivative}, \text{Prewwit\_}ru) = \begin{pmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{pmatrix}, \boldsymbol{A} = f_{\boldsymbol{A}}(\text{Derivative}, \text{Prewwit\_}rd) = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -1 \end{pmatrix},$$

$$\boldsymbol{A} = f_{\boldsymbol{A}}(\text{Derivative}, \text{Laplacian\_4}) = \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}, \boldsymbol{A} = f_{\boldsymbol{A}}(\text{Derivative}, \text{Laplacian\_8}) = \begin{pmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{pmatrix}. \tag{4.35}$$

畳み込みフィルタを処理の種類とすれば，畳み込みフィルタにおける関数のパラメータ$\boldsymbol{\lambda}$には上記から使用するアルゴリズム及びそのアルゴリズムのパラメータが含まれる．従って，$\boldsymbol{\lambda}$の近傍操作では，例えば，Gaussian から Gabor への変更，Gaussian(*sigma*)から Gaussian(*sigma'*)，　ただし *sigma*≠*sigma'*へ変更する．

図 4.11 に各畳み込みフィルタの例を示す.



(a) (b) (c) (d) (e) (f) (g)

(h) (i) (j) (k) (l) (m)

図 4.11　R 成分における畳み込みフィルタ．(a)Gaussian，(b)DoG，(c)Gabor，(d)*w* 方向の 1 次微分，(e)*h* 方向の 1 次微分，(f)*w* 方向ソーベル，(g)*h* 方向のソーベル，(h)*w* 方向の Scharr，(i)*h* 方向の Scharr，(j)右上斜めの Prewwit,(k)左上斜めの Prewwit，(l)4 近傍の Laplacian，(m)8 近傍の Laplacian.

4.2.7　Box フィルタ

Box フィルタの計算式を説明する．Box フィルタは矩形領域内の値を全て足し合わせる．すなわち，Box フィルタは畳み込みの重み $\boldsymbol{A}$ の全要素が 1 の畳み込みフィルタと同じである．Box のパラメータは矩形領域のサイズ *width*, *height* である．

$$\begin{aligned} &\underline{\boldsymbol{Y}}^{j} = f_{\mathrm{Box}}\left(\underline{\boldsymbol{S}}^{j}_{1(3)}, \boldsymbol{\lambda}^{*}\right), \\ &\boldsymbol{\lambda}^{*} = \{width, height\} \subset \boldsymbol{\lambda}_{\mathrm{Box}} = \{\boldsymbol{Width}, \boldsymbol{Height}\} \subset \boldsymbol{\lambda}, \\ &width \in \boldsymbol{Width} \subset \boldsymbol{Z}, height \in \boldsymbol{Height} \subset \boldsymbol{Z}. \end{aligned} \tag{4.36}$$

Box フィルタを処理の種類とすれば，Box フィルタにおける関数のパラメータ $\boldsymbol{\lambda}$ には矩形サイズ *width*, *height* が含まれる．従って，$\boldsymbol{\lambda}$ の近傍操作では，例えば，3x3 の矩形サイズを 5x9 に変更する．

図 4.12 に入力画像の R 成分に対する Box フィルタの結果を示す．
なお，Box フィルタは Integral Image[85]を用いることにより高速に計算可能である．入力 $\underline{\boldsymbol{S}}^{j}$ のチャンネル c にてスライスした行列 $\boldsymbol{S} = \underline{\boldsymbol{S}}^{j}_{c(3)}, \boldsymbol{S} \in \boldsymbol{R}^{W_S \times H_S}$ とすれば Integral Image $\boldsymbol{ii} \in \boldsymbol{R}^{W_S \times H_S}$ は以下の式で定義される．

$$\boldsymbol{ii}_{wh} = \sum_{\bar{w} \le w, \bar{h} \le h} \boldsymbol{S}_{\bar{w}\bar{h}}, \ \forall w \in \{1,2,...,W_S\}, \ \forall h \in \{1,2,...,H_S\}. \tag{4.37}$$

図 4.12　R 成分における Box フィルタ.

すなわち $\boldsymbol{ii}_{wh}$ は, 画像左上原点から(w, h)までの画素値の総和である. Integral Image は以下の式により逐次的に算出される.

$$\begin{aligned} &SUM_{wh} = SUM_{wh-1} + \boldsymbol{S}_{wh} \ , \\ &\boldsymbol{ii}_{wh} = \boldsymbol{ii}_{w-1\,h-1} + SUM_{wh} \ , \\ &\forall w \in \{1,2,...,W_S\}, \forall h \in \{1,2,...,H_S\} . \end{aligned} \tag{4.38}$$

ここで SUM_{wh} は同一行の要素の総和である．なお $\boldsymbol{S}_{w\ -1}$=0，$\boldsymbol{ii}_{-1\ y}$=0 とする．この Integral Image を用いれば，ある矩形領域内の画素値の総和を高速に算出できる．**図 4.13** に示す矩形領域 D の画素値の総和 $Area_{\mathrm{D}}$ は以下の式となる.

$$Area_{\mathrm{D}} = \boldsymbol{ii}_{w1h1} + \boldsymbol{ii}_{w4h4} - \boldsymbol{ii}_{w2h2} - \boldsymbol{ii}_{w3h3} \ . \tag{4.39}$$

これにより，矩形の大きさによらず 4 点の加減算で算出できる.

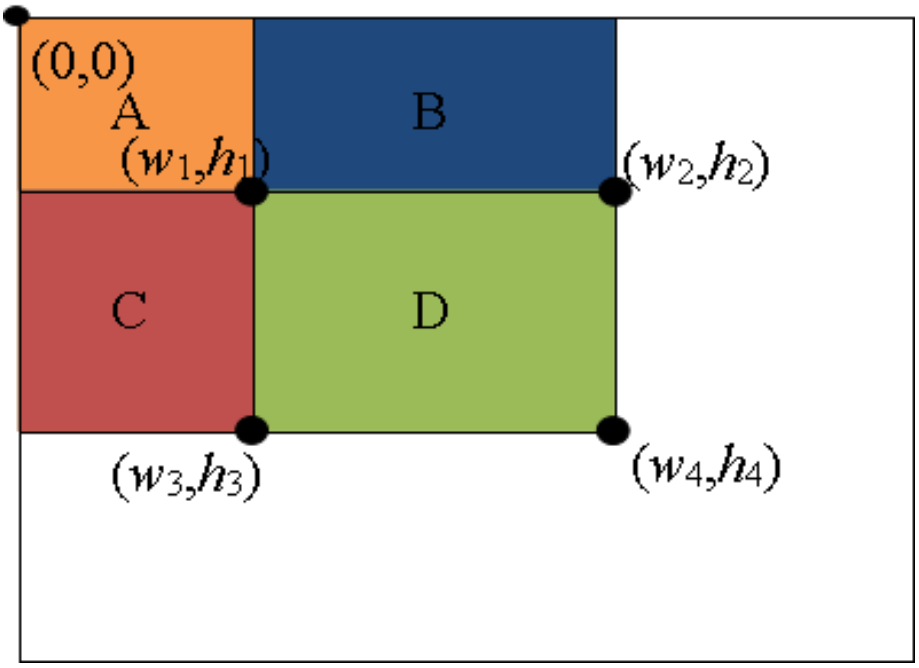


図 4.13　Integral Image.

図 4.14　R 成分における Max フィルタ.

4.2.8　Max フィルタ

Max フィルタの計算式を述べる．Max フィルタは CNN[10]や sp-Cov[74]に用いられている．矩形領域の要素値の最大値を返すフィルタである．なお，C_S=1 である．

$$\begin{aligned}
&\underline{\boldsymbol{Y}}^j = f_{\mathrm{Max}}\left(\underline{\boldsymbol{S}}^j, \boldsymbol{\lambda}^*\right), \\
&\boldsymbol{\lambda}^* = \{width, height\} \subset \boldsymbol{\lambda}_{\mathrm{Max}} = \{\boldsymbol{Width}, \boldsymbol{Height}\} \subset \boldsymbol{\lambda}, \\
&width \in \boldsymbol{Width} \subset \boldsymbol{Z}, height \in \boldsymbol{Height} \subset \boldsymbol{Z}.
\end{aligned} \tag{4.40}$$

$$\begin{aligned}
&\underline{\boldsymbol{Y}}^j = f_{\mathrm{Max}}\left(\underline{\boldsymbol{S}}^j, \lambda\right) = \max\left\{\underline{\boldsymbol{S}}_{1w-\overline{w}h-\overline{h}}, \forall \overline{w}, \overline{h}\right\}, \\
&\overline{w} \in \{-width/2, -width/2+1, ..., width/2\} \\
&\overline{h} \in \{-height/2, -height/2+1, ..., height/2\}
\end{aligned} \tag{4.41}$$

Max フィルタを処理の種類とすれば，Max フィルタにおける関数のパラメータ $\boldsymbol{\lambda}$ には矩形サイズ *width*, *height* が含まれる．従って，$\boldsymbol{\lambda}$ の近傍操作では，例えば，3x3 の矩形サイズを 5x9 に変更する．

図 4.14 に入力画像の R 成分に対する Max フィルタの結果を示す.

4.2.9　HOG(Histograms Oriented Gradients)

提案法に用いる HOG は Dalal らの HOG を拡張している．拡張法は２章に述べた通りである．ヒストグラムの分割数，セルのサイズ，正規化するブロックサイズを変数として，HOG を関数の種類とした場合に $\boldsymbol{\lambda}$ に含める．これにより，HOG のパラメータが最適化される．

$$\begin{aligned}
&\underline{\boldsymbol{Y}}^j = f_{\mathrm{HOG}}\left(\underline{\boldsymbol{S}}^j, \boldsymbol{\lambda}^*\right), \\
&\boldsymbol{\lambda}^* = \{bin, cellsize, blocksize\} \subset \boldsymbol{\lambda}_{\mathrm{HOG}} = \{\boldsymbol{Bin}, \boldsymbol{Cellsize}, \boldsymbol{Blocksize}\} \subset \boldsymbol{\lambda}, \\
&bin \in \boldsymbol{Bin} \subset \boldsymbol{Z}, cellsize \in \boldsymbol{Cellsize} \subset \boldsymbol{Z}, blocksize \in \boldsymbol{Blocksize} \subset \boldsymbol{Z}.
\end{aligned} \tag{4.42}$$

なお，C_S=1 である．ここで，*bin* はヒストグラムの分割数，*cellsize* はセルに含める矩形領域の要素数，*blocksize* は正規化のサイズである．HOG を関数の種類とす

れば，近傍操作は *bin*，*cellsize*，*blocksize* を変更することにあたる．

4.2.10　ULBP(Uniformed Local Binary Patterns)

ULBP[104]の計算について説明する．まず，LBP について述べ，次に ULBP について述べる．LBP[14]は注目画素とその周辺画素の画素値の大小関係からパターンを定義する特徴量である．LBP の考え方を図 4.15 に示す．LBP では注目画素と周辺画素の大小関係から得られる 0，1 の羅列を 10 進数に変換した値をパターンとして用いる．全画素に対してパターンを割り当てた後，矩形領域内のパターンの出現頻度を算出し，特徴量として学習に用いる．また，出現頻度を算出するかわりに，注目画素と周辺画素の差分の大きさを足す手法[14]もある

LBP がとる周辺画素は識別対象に応じて適したサイズが存在することが想定される．しかし，LBP の問題点は周辺画素が多くなるにつれパターンの総数が指数関数的に増加することである．これに対し，ULBP[104]では図 4.9 のように，0，1 の反転回数を規定し，その規定に則った場合のみパターンとして用いる．これにより，周辺画素が多くなってもパターンが指数爆発しない．ULBP[104]では反転回数を 2 回まで(Uniform 2)としている．Uniform 2 とそうではない例を図 4.16

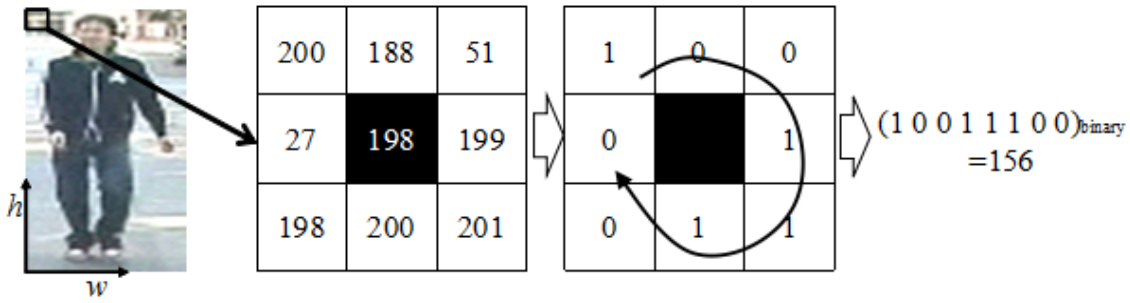


図 4.15　LBP の概念図.

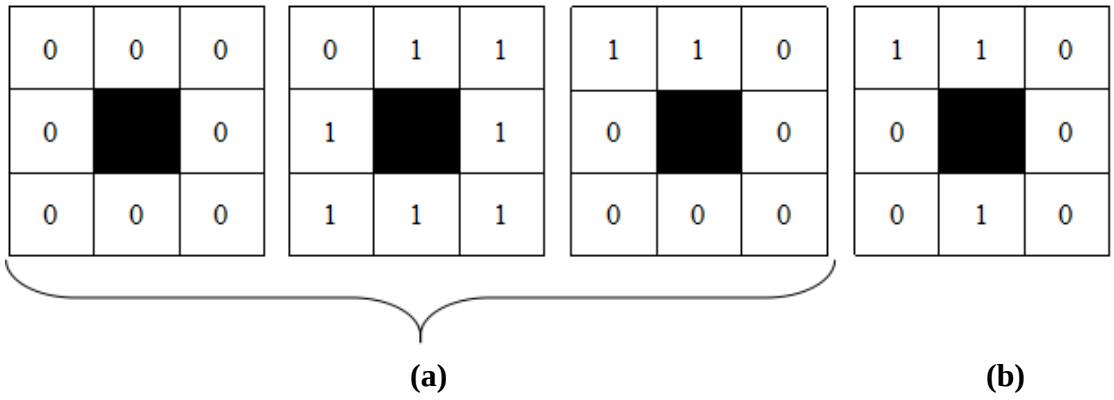


図 4.16　ULBP(Uniform 2)．(a)Uniform 2 の例，(b)Uniform 2 ではない例.

に示す. 提案法では最適な周辺画素のサイズを取得する目的から ULBP を用いる. なお，*Cs*=1 である．***Neighbor*** を周辺画素 **neighbor** の種類集合とすれば，ULBP は以下の式により表される.

$$\begin{aligned}
&\underline{\boldsymbol{Y}}^{j} = f_{\mathrm{ULBP}}\left(\underline{\boldsymbol{S}}^{j}, \boldsymbol{\lambda}^{*}\right), \\
&\boldsymbol{\lambda}^{*} = \left\{\boldsymbol{neighbor}, type, width, height\right\} \subset \boldsymbol{\lambda}_{\mathrm{Box}} = \left\{\boldsymbol{Neighor}, \boldsymbol{Type}, \boldsymbol{Width}, \boldsymbol{Height}\right\} \subset \boldsymbol{\lambda}, \\
&neighor \subset \boldsymbol{Neighor}, type \in \left\{\mathrm{one}, \mathrm{diff}\right\}, \\
&width \in \boldsymbol{Width} \subset \boldsymbol{Z}, height \in \boldsymbol{Height} \subset \boldsymbol{Z}\ .
\end{aligned} \tag{4.43}$$

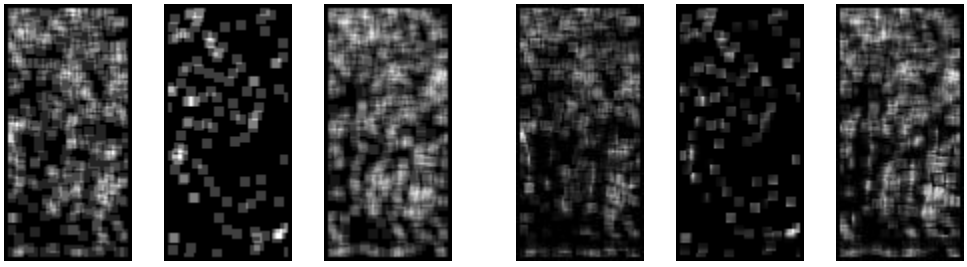
ULBP の計算式は以下の通りである．まず，*type*=one の場合は以下となる.

$$\begin{aligned}
&pattern_{wh} = f_{\mathrm{decimalize}}\left(f_{\mathrm{uniform2}}\left(f_{\mathrm{binary}}\left(\underline{\boldsymbol{S}}^{j}_{1wh}, \underline{\boldsymbol{S}}^{j}_{1\bar{w}\bar{h}}\right), \forall \bar{w}, \bar{h} \in \boldsymbol{neighbor}\right)\right), \\
&f_{\mathrm{binary}}\left(s1, s2\right) = \begin{cases} 1 & if\ s1 > s2 \\ 0 & else \end{cases}, \\
&\underline{\boldsymbol{\Gamma}}_{pattern_{wh}\, wh} = 1, \underline{\boldsymbol{\Gamma}} \in \left\{0, 1\right\}^{N_{pattern} \times W_S \times H_S}, \\
&\underline{\boldsymbol{Y}}^{j} = \left(f_{\mathrm{box}}\left(\underline{\boldsymbol{\Gamma}}_{1(3)}, \boldsymbol{\lambda}^{*}\right), f_{\mathrm{box}}\left(\underline{\boldsymbol{\Gamma}}_{2(3)}, \boldsymbol{\lambda}^{*}\right), ..., f_{\mathrm{box}}\left(\underline{\boldsymbol{\Gamma}}_{N_{pattern}(3)}, \boldsymbol{\lambda}^{*}\right)\right).
\end{aligned} \tag{4.44}$$

ここで，f_{uniform2} は反転回数が 2 回以下の 0, 1 の羅列のみを許す関数である．$f_{\mathrm{decimalize}}$ は 2 進数から 10 進数へ変換する関数である． $N_{pattern}$ はパターンの総数である. *type*=diff の場合は $\underline{\boldsymbol{\Gamma}}_{pattern_{wh}\, wh}$ の値を以下の式により与える.

$$\underline{\boldsymbol{\Gamma}}_{pattern_{wh}\, wh} = \sum_{\forall \bar{w}, \bar{h} \in Neighbor} \left| \underline{\boldsymbol{S}}^{j}_{1wh} - \underline{\boldsymbol{S}}^{j}_{1\bar{w}\bar{h}} \right|, \underline{\boldsymbol{\Gamma}} \in \boldsymbol{R}^{N_{pattern} \times W_S \times H_S}\ . \tag{4.45}$$

ULBP を関数の種類とすれば，ULBP における関数のパラメータ $\boldsymbol{\lambda}$ には周辺領域 ***neighbor***，頻度の計算方式 *type*，頻度算出時のサイズ *width*，*height* が含まれる. 従って， $\boldsymbol{\lambda}$ の近傍操作ではこれらの値を変更する.



(a) **(b)**

図 4.17　ULBP. (a)*type*=one の例, 左から反転回数 0, 反転回数 2 の一つ, Uniform2 ではない，(b)*type*=diff，左から反転回数 0，反転回数 2 の一つ，Uniform2 ではない.

図 4.17 に入力画像の R 成分に対する ULBP の結果を示す．(a)，(b)の中央の画像は反転回数が 2 の場合の一つである．例えば，周辺画素を 9 とれば反転回数が 2 となるパターンは 36 通りある．そのうちの 1 パターンを示している．

4.2.11 Covariance

Covariance の計算について説明する．まず，Covariance[44]について述べ，次に提案法に用いている Covariance について述べる．Covariance は共分散を用いる．共分散を計算するためには 2 つの行列とパッチサイズを指定する．図 4.18 に共分散の計算の概要を示す．共分散の計算は以下の式である．なお，$Cs \geqq 2$ である．

$$
\begin{aligned}
&\underline{\boldsymbol{Y}}^j = \left(f_{\mathrm{cov}}\left(\underline{\boldsymbol{S}}^j_{1wh}, \underline{\boldsymbol{S}}^j_{2wh}\right),\ f_{\mathrm{cov}}\left(\underline{\boldsymbol{S}}^j_{1wh}, \underline{\boldsymbol{S}}^j_{3wh}\right), ...,\ f_{\mathrm{cov}}\left(\underline{\boldsymbol{S}}^j_{c1wh}, \underline{\boldsymbol{S}}^j_{c2wh}\right), ...,\ f_{\mathrm{cov}}\left(\underline{\boldsymbol{S}}^j_{C_S wh}, \underline{\boldsymbol{S}}^j_{C_S-1wh}\right)\right),\\
&f_{\mathrm{cov}}\left(\underline{\boldsymbol{S}}^j_{1wh}, \underline{\boldsymbol{S}}^j_{2wh}\right) = \sum_{\bar{w}\in Width, \bar{h}\in Height} \left(\underline{\boldsymbol{S}}^j_{1w-\bar{w}h-\bar{h}} - E_1\right)\left(\underline{\boldsymbol{S}}^j_{2w-\bar{w}h-\bar{h}} - E_2\right),\\
&E_1 = \frac{\sum_{\bar{w}\in Width, \bar{h}\in Height}\left(\underline{\boldsymbol{S}}^j_{1w-\bar{w}h-\bar{h}}\right)}{|\boldsymbol{Width}\|\boldsymbol{Height}|}, E_2 = \frac{\sum_{\bar{w}\in Width, \bar{h}\in Height}\left(\underline{\boldsymbol{S}}^j_{2w-\bar{w}h-\bar{h}}\right)}{|\boldsymbol{Width}\|\boldsymbol{Height}|},\\
&\boldsymbol{Width} = \{-width/2, -width/2+1, ..., width/2\},\\
&\boldsymbol{Height} = \{-height/2, -height/2+1, ..., height/2\},\\
&\forall w \in \{width/2, width/2+1, ..., W_S - width/2\},\\
&\forall h \in \{height/2, height/2+1, ..., W_H - height/2\}.
\end{aligned}
\tag{4.46}
$$

共分散の値は，$\underline{\boldsymbol{S}}^j_1$ のパッチの平均より値が大きく，同様に $\underline{\boldsymbol{S}}^j_2$ のパッチの平均より値が大きいほど大きな値をとる．$\underline{\boldsymbol{S}}^j_1$ のパッチの平均より値が大きく，$\underline{\boldsymbol{S}}^j_2$ のパッチの平均より値が小さいほど小さな値をとる．従って，値が大きいほど相関が強く，小さいほど相関が弱いことがわかる．識別問題においては，特徴量として共分散を用いることにより，あるパッチにおけるある行列とある行列の相関を特徴として捉えることが可能となる．例えば，入力した画像を対象に共分散を計算することも可能である．この場合，RGB から R と G を選択し，共分散を計算する．2 つの組から 1 つの共分散を格納した行列が計算できる．RGB の場合 ${}_3C_2=3$ つの行列が計算され，それを順番に第 3 モード方向に並べたテンソルが出力される．sp-Cov[74]では横位置 w を格納した行列，縦位置 h を格納した行列， 横方向の 1 次微分 $f_{\mathrm{convolution}}$(Derivative, Dw)， 縦方向の 1 次微分 $f_{\mathrm{convolution}}$(Derivative, Dh)，勾配 f_{gradient}(arctan), 横方向の 2 次微分 $f_{\mathrm{convolution}}$($f_{\mathrm{convolution}}$(Derivative, Dw), Derivative, Dw)，縦方向の 2 次微分 $f_{\mathrm{convolution}}$($f_{\mathrm{convolution}}$(Derivative，Dh)，Derivative，Dh)を入力に共分散を計算している．

Covariance[44]ではパッチ毎の分散の大きさに対して正規化されておらず，相関を特徴として捉えるには不十分であると考えられる．そこで，sp-Cov[74]では，

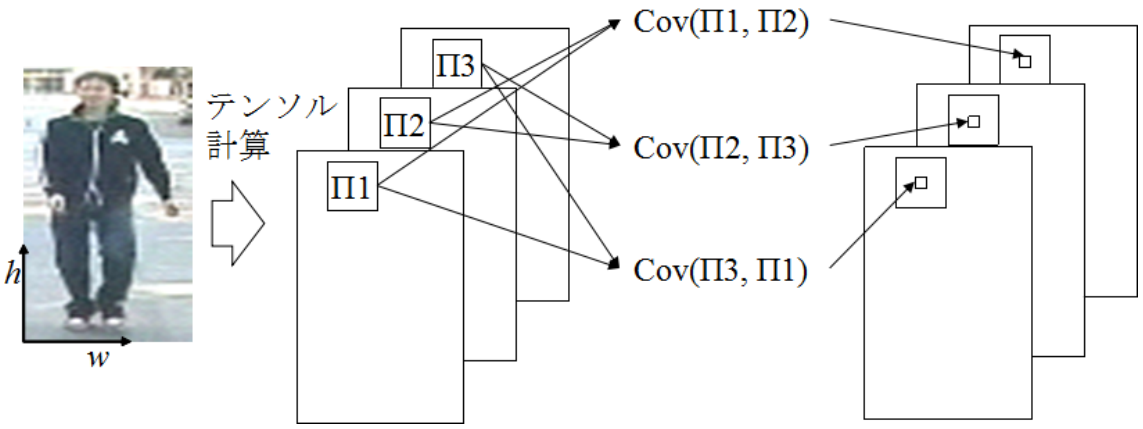


図 4.18　Covariance 特徴量.

共分散ではなく相関係数を用いることにより正規化する．そのために，式(4.46)の f_{cov} を以下の式に置き換える．

$$f_{\text{correlation coefficient}}\left(\underline{\boldsymbol{S}}^{j}_{1wh},\underline{\boldsymbol{S}}^{j}_{2wh}\right)=\frac{\sum_{\bar{w}\in Width,\bar{h}\in Height}\left(\underline{\boldsymbol{S}}^{j}_{1w-\bar{w}h-\bar{h}}-E_1\right)\left(\underline{\boldsymbol{S}}^{j}_{2w-\bar{w}h-\bar{h}}-E_2\right)}{\sqrt{\sum_{\bar{w}\in Width,\bar{h}\in Height}\left(\underline{\boldsymbol{S}}^{j}_{1w-\bar{w}h-\bar{h}}-E_1\right)^2}\sqrt{\sum_{\bar{w}\in Width,\bar{h}\in Height}\left(\underline{\boldsymbol{S}}^{j}_{2w-\bar{w}h-\bar{h}}-E_2\right)^2}}. \tag{4.47}$$

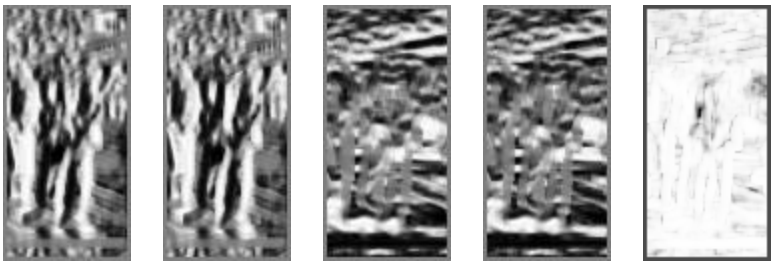
また，位置に対する頑健性を得るために，CNN[10]にて用いられる Max プーリング, すなわち Max フィルタ $f_{\max}$ を用いる. sp-Cov[74]の sp は spatial の意味であり，Max プーリングによる位置頑健性を付与していることを表している．sp-Cov[74]では[44]に加えて，$f_{\text{orientation}}$(arctan2)も用いている．なお，提案法では Covariance を計算するための 2 つの行列は $\boldsymbol{\Lambda}$ の最適化により自動的に選ばれる．従って，従来法に用いられている sp-Cov の入力に限られない．また，Max フィルタすべきか否かも $\boldsymbol{\Lambda}$ の最適化により自動的に選ばれる．

提案法における Covariance の式は以下の通りである．

$$\begin{aligned}&\underline{\boldsymbol{Y}}^{j}=f_{\text{covariance}}\left(\underline{\boldsymbol{S}}^{j},\boldsymbol{\lambda}^{*}\right),\\&\boldsymbol{\lambda}^{*}=\{width,height\}\subset\boldsymbol{\lambda}_{\text{covariance}}=\{\boldsymbol{Width},\boldsymbol{Height}\}\subset\boldsymbol{\lambda},\\&width\in\boldsymbol{Width}\subset\boldsymbol{Z},height\in\boldsymbol{Height}\subset\boldsymbol{Z}\ .\end{aligned}\tag{4.48}$$

なお，$f_{\text{covariance}}$ は式(4.46)の f_{cov} を式(4.47)に置き換えたものである．

Covariance を関数の種類とすれば，Covariance における関数のパラメータ $\boldsymbol{\lambda}$ には Covariance 計算時のパッチサイズ *width*，*height* が含まれる．従って，$\boldsymbol{\lambda}$ の近傍操作ではこれらの値を変更する．図 4.19 に入力画像の R 成分と G 成分を用いた Covariance の結果を示す．Covariance では画像中の位置との相関も対象となる．



(a)　(b)　(c)　(d)　(e)

図 4.19　R, G 成分における Covariance．(a)R と *h*，(b)G と *h*，(c)R と *w*，(d)G と *w*，(e)R と G.

4.3 従来法との比較実験

ベンチマークデータセットを用いて従来法と比較実験を行った．研究者が手動により設計した特徴量の変換式を用いる手法と CNN により畳み込み積分の重みが最適化される手法と提案法を比較し，提案法の優位性を確認した．実験に用いたデータセット，計算機環境は 3.5 節と同様である．

本実験の実験に用いたソフトウェアの実装について述べる．本実験に用いたライブラリを表 4.1 に示す．本実験は 3.5 節と同様に，実験の再現性を高めるため OSS のみを用いて開発した．

本実験では従来の特徴量と比較する．特徴量は研究者が計算式をパラメータも含めて定義する場合と，変換式のパラメータを自動調整する手法がある．前者との比較のために sp-Cov と ULBP を用いる手法[74]を, 後者との比較のために CNN に Inception 構造を用いた google の手法[75]と比較する．前者は Caltech Pedestrian Detection Benchmark[105]の INRIA Person Dataset の中で特徴量の計算式を研究者が定義する手法としては最もよい結果を得ている．後者は INRIA Person Dataset には用いられていないが，ILSVRC2014 にて最も良い結果を得た手法である．現在 INRIA Person Dataset において CNN を用いた手法が提案されているが，[74]よりも良い結果を与えている CNN を用いた手法[77]，[80]は画像識別のための特徴量を算出する以外にローカライズの機能も備えており，単純比較できないため比較対象から外している．sp-Cov と ULBP[104]を用いた手法では表 4.1 に示すリファレンス実装を実験に用いた．表 4.2 に sp-Cov と ULBP のパラメータを示す．これらのパラメータは[74]に沿っている．

図 4.20 に示す Inception 構造を用いた CNN[75]には NN のソフトウェアである caffe[106]を用い, ネットワークは web で提供されている GoogLeNet[107]を用いた．ただし，このネットワーク設定は本実験と画像の解像度が異なる．そのため元の設定では出力層へ入力するニューロンが 0 になるため，元の 38 層から 15 層に変更している．違いは図 4.20 に示す Inception 構造の数である．元は Inception 構造が 9 回連なる(図 4.21)が，実験では 2 回(図 4.22)となっている．学習の繰り返

表 4.1　ソフトウェアの構成.

ランタイム(コンパイラ)	Visual C++ 2012 64bit
ULBP，sp-Cov，HOG	リファレンス実装
上記以外の画像処理	OpenCV 3.0
AdaBoost	OpenCV 3.0
FTOP 求解	スクラッチ実装

表 4.2　sp-Cov と ULBP のパラメータ.

sp-Cov	
相関の計算対象	横位置，縦位置，横方向の 1 次微分，縦方向の 1 次微分，横方向の 2 次微分，縦方向の 2 次微分, 輝度勾配の大きさ, 輝度勾配の方向(arctan), 輝度勾配の方向(arctan2)
パッチサイズ[pixel]	8x8，16x16，32x32
Max プーリングサイズ[pixel]	4x4
ULBP	
周辺画素[pixel]	3x3
ヒストグラムの矩形領域サイズ[pixel]	4x4
Max プーリングサイズ[pixel]	8x8

し回数は 100 エポックとした．ここで，Inception 構造について述べる．Inception 構造は異なるサイズの畳み込みフィルタをネットワークの幅方向に用意し，それらの結果を Max プーリングすることを繰り返す構造を指す. この構造自体を複数回重ねることにより識別率を向上させている.

従来法との比較では，特徴量の計算式による違いに注目するため，提案法と同様に最終的な識別器を学習する時には AdaBoost を用いた．CNN の場合は，特徴量に CNN の出力層一つ手前の隠れ層を用いた．なお，CNN の学習には caffe を用いているが，機械学習には OpenCV に caffe のネットワーク構造をインポートしてから OpenCV の AdaBoost を用いた.

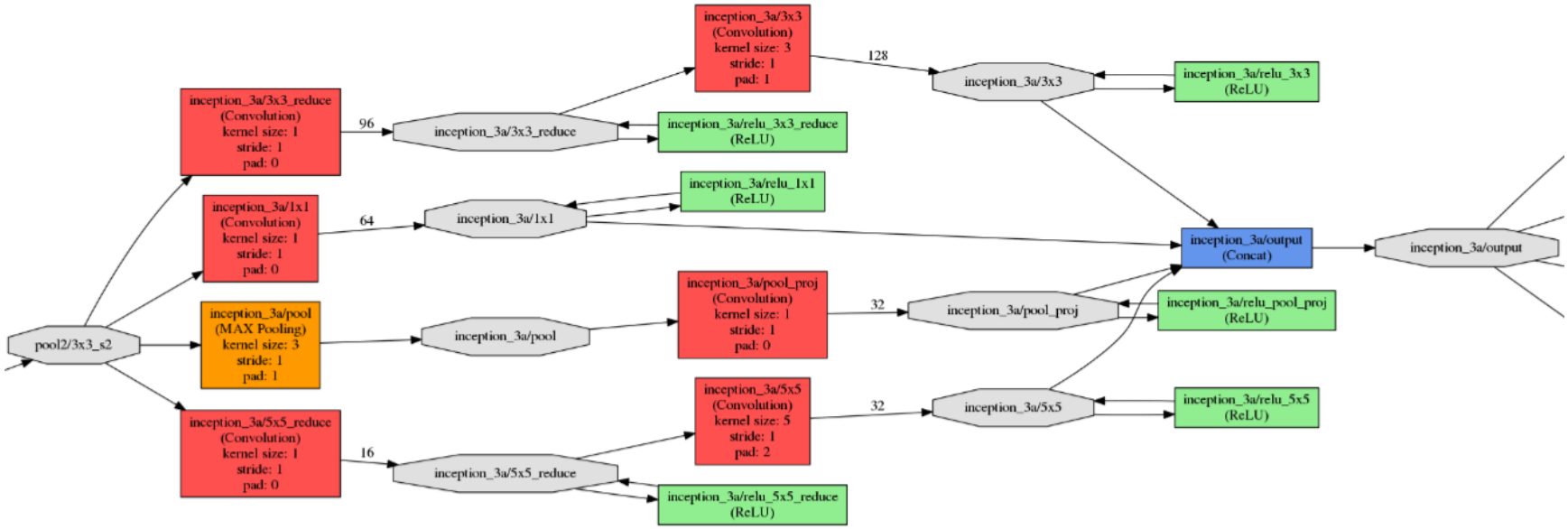


図 4.20　Inception 構造.

図 4.21　[107]の Inception 構造.

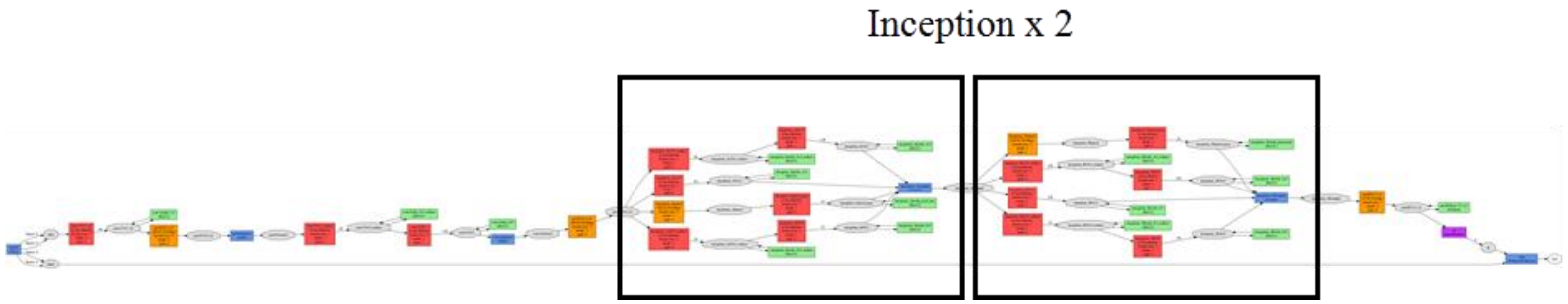


図 4.22　実験に用いた Inception 構造.

表 4.3　処理のパラメータの値域と刻み幅の分割数.

処理の種類	項目		最小	最大	分割数
表色系変換	-				
四則演算	入力チャンネル数		2	5	-
ノルム	入力チャンネル数		1	2	-
方向	-				
勾配	-				
畳み込み	Gaussian (式 4.32)	Width	3	11	-
		Height	3	11	-
		Sigma	0.1	10	100
	DoG (式 4.33)	Width	3	11	-
		Height	3	11	-
		sigma1	0.1	10	100
		T	0.1	0.99	100
	Gabor (式 4.34)	Width	3	11	-
		Height	3	11	-
		Sigma	0.1	10	100
		Theta	-π	π	100
		gamma	0.1	1	100
Box (式 4.36)	Width		3	11	-
	Height		3	11	-
Max (式 4.40)	Width		3	11	-
	Height		3	11	-
HOG (式 4.42)	Bin		3	9	-
	CellSize Width		3	10	-
	CellSize Height		3	10	-
	BlockSize Width		3	15	-
	BlockSize Height		3	15	-
ULBP (式 4.43)	Neighbor		3x3	3x3	-
	Width		3	5	-
	Height		3	5	-
Covariance (式 4.46)	入力チャンネル数		3	9	-
	Width		9	31	-
	Height		9	31	-

これらの従来法と提案法を比較する．表 4.3 に本実験に用いる処理の種類毎のパラメータの最大，最小，刻み幅の分割数を示す．3.5　仮説検証実験では表色系変換，畳み込みフィルタ，Max フィルタのみ用いたが，本実験では 4.2　ノードに用いる処理の種類にて述べた処理を全て用いる．提案法のパラメータ(FTOP*)を表 4.4～4.5 に示す．AdaBoost は特徴量次元が多いほど汎化性能が向上する可能性がある．この点において提案法が有利とならないように，従来法よりも少ない次元数となるような FTOP を設定した．ここで，表 4.3～4.5 に示したパラメータにて実験を行おうとすると，最適化における 1 回の評価値計算のために 2 時間以上要することが判明した．これでは，FTOP の求解が進まない．そこで表 4.6 に示すように評価値計算に用いるサンプル数を減らし，図 4.23 に示すように小さい FTN から大きな FTN を構築する手法を用いることにより FTOP を求解した．図 4.23 では処理が左から右へ流れる．まず，表 4.5 の子問題として異なる最適化問題を 5 個(FTOP1～FTOP5)設定した(表 4.8). これらを表 4.7 の求解アルゴリズムのパラメータを用いて多点局所探索により求解した．この求解により得られた FTN1～FTN5 を並列に接続し FTN を生成した．この FTN を初期解として FTOP*を多点局所探索により求解し，FTN を得た．これにより，表 4.5 の最適化問題(FTOP*)を満たしつつ，効率よく求解できると考えられる．

FTOP1～FTOP5 では従来法を参考に FTOP のパラメータを設定することにより効率的に求解されることが期待される. このように提案法では FTN に組み込む処理に制限が無いため，過去の研究成果を組み込むことができる．元の問題(表 4.5)の $Cost_{\max}$ や $D_{\max}$ を満たすチャンネル数を満たすためには, それぞれ 1/5 にした値を FTOP1～FTOP5 に設定すればよい. 表 4.8(a)に示すように FTOP1 の問題設定は CNN[10]を参考にしている．同表(b)と(c)における FTOP2，FTOP3 の問題設定は sp-Cov と ULBP[74]を参考にしている．同表(d)における FTOP4 の問題設定は HOG[29]を参考にしている. 同表(e)における FTOP5 はこれらの特徴量の計算式の要素として用いられている計算式を問題設定に組み込んでいる．

FTOP*では FTOP1～FTOP5 にて分割及び制限していた処理の種類の接続制限すなわち $\boldsymbol{Q}$ を表色系の変換を除いて接続可能にする．表 4.6 の$|\boldsymbol{M}_l|$(最適化に用いる学習用サンプル数)と$|\boldsymbol{M}_v|$(最適化に用いる評価用サンプル数)が表 4.4 の半分になっているが，最適化における評価値の計算にかかる時間を減らすためである．FTOP1～FTOP5 に比べ FTOP*では生成される FTN から計算される特徴量の次元数が大きくなる．特徴量の次元数が大きくなることにより評価値の計算にかかる時間が大きくなることを防ぐためである．

これらの設定により実験した従来法との比較結果を表 4.9 及び DET(Detection Error Trade-off)として図 4.24 に示す．

表 4.4　FTOP*の求解アルゴリズムの設定.

$\|\boldsymbol{M}_l\|$	1392
$\|\boldsymbol{M}_v\|$	12528
D_{ratio}	0.1
初期解生成回数	10
近傍探索収束回数	100
エッジ削除率	0.2

表 4.5　FTOP*.

処理の種類	T:Top，p_1:Convert Color，p_2:Box，p_3:Max，p_4:Arithmetic，p_6:Convolution，p_7:Norm，p_8:Orientation，p_9:Gradient，p_{10}:HOG，p_{11}:ULBP，p_{12}:Covariance，B:Bottom														
$cost_j$ $\forall j \in \{1,2,...,J\}$	1														
$Cost_{\max}$	125														
$Cost_{p1}$	1														
$Cost_{p12}$	1														
$D_{\max}$ を満たすチャンネル数	50														
$\boldsymbol{Q}$		T	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}	p_{11}	p_{12}	B
	T	0	1	0	0	0	0	0	0	0	0	0	0	0	0
	p_1	0	0	1	1	1	1	1	1	1	1	1	1	1	1
	p_2	0	0	1	1	1	1	1	1	1	1	1	1	1	1
	p_3	0	0	1	1	1	1	1	1	1	1	1	1	1	1
	p_4	0	0	1	1	1	1	1	1	1	1	1	1	1	1
	p_5	0	0	1	1	1	1	1	1	1	1	1	1	1	1
	p_6	0	0	1	1	1	1	1	1	1	1	1	1	1	1
	p_7	0	0	1	1	1	1	1	1	1	1	1	1	1	1
	p_8	0	0	1	1	1	1	1	1	1	1	1	1	1	1
	p_9	0	0	1	1	1	1	1	1	1	1	1	1	1	1
	p_{10}	0	0	1	1	1	1	1	1	1	1	1	1	1	1
	p_{11}	0	0	1	1	1	1	1	1	1	1	1	1	1	1
	p_{12}	0	0	1	1	1	1	1	1	1	1	1	1	1	1
	B	0	0	0	0	0	0	0	0	0	0	0	0	0	0

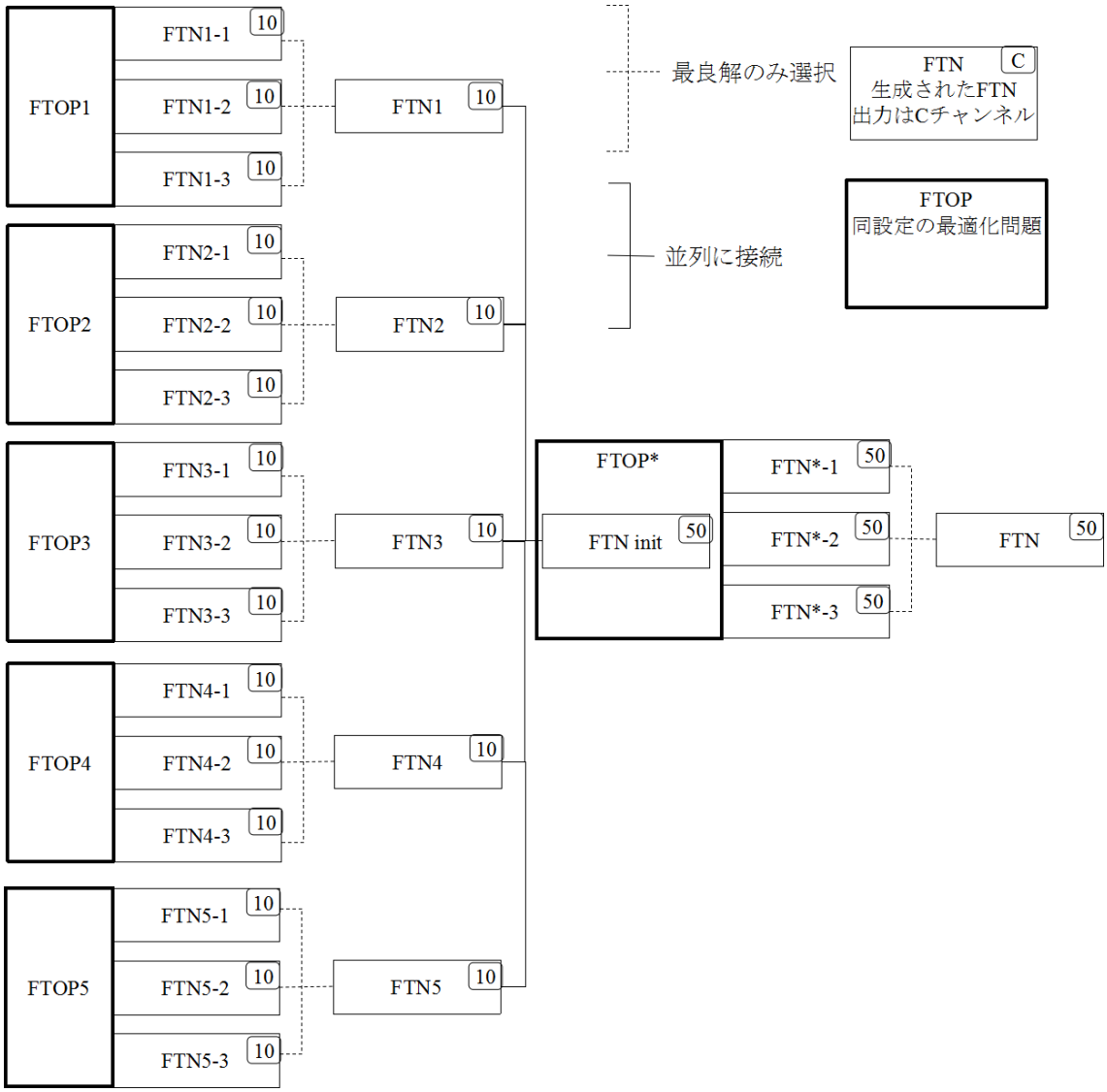


図 4.23　提案法における最適化問題の分割.

表 4.6　FTOP*の求解アルゴリズムの設定(変更後).

$\|\boldsymbol{M}_l\|$	646
$\|\boldsymbol{M}_v\|$	6264
D_{ratio}	0.1
初期解生成回数	-
近傍探索収束回数	100
エッジ削除率	0.3

表 4.7　FTOP1～FTOP5 の求解アルゴリズムの設定.

$\|\boldsymbol{M}_l\|$	1392
$\|\boldsymbol{M}_v\|$	12528
D_{ratio}	0.1
初期解生成回数	10
近傍探索収束回数	100
エッジ削除率	0.2

表 4.8　FTOP*の分割設定．(a)FTOP1，(b)FTOP2，(c)FTOP3，(d)FTOP4，(e)FTOP5．（続く）

(a)

処理の種類	T:Top, p_1:Convert Color, p_2:Convolution, p_3:Max, B:Bottom					
$cost_j$ $\forall j \in \{1,2,...,J\}$	1					
$Cost_{\max}$	25					
$Cost_{p1}$	1					
$Cost_{p2}$	15					
$D_{\max}$ を満たすチャンネル数	10					
$\boldsymbol{Q}$		T	p_1	p_2	p_3	B
	T	0	1	0	0	0
	p_1	0	0	1	0	0
	p_2	0	0	1	1	0
	p_3	0	0	1	0	1
	B	0	0	0	0	0

(b)

処理の種類	T:Top，p_1:Convert Color，p_2:Convolution，p_3:Covariance，p_4:Norm，p_5:Orientation，p_6:Max，B:Bottom
$cost_j$ $\forall j \in \{1,2,...,J\}$	1
$Cost_{\max}$	25
$Cost_{p1}$	1
$Cost_{p3}$	1
$D_{\max}$ を満たすチャンネル数	10

表 4.8　FTOP*の分割設定. (a)FTOP1, (b)FTOP2, (c)FTOP3, (d)FTOP4, (e)FTOP5.
（続く）

Q		T	p_1	p_2	p_3	p_4	p_5	p_6	B
	T	0	0	0	0	0	0	0	0
	p_1	1	0	0	0	0	0	0	0
	p_2	0	1	0	0	0	0	0	0
	p_3	0	0	1	0	1	1	0	0
	p_4	0	0	1	0	0	0	0	0
	p_5	0	0	1	0	0	0	0	0
	p_6	0	0	0	1	0	0	0	0
	B	0	0	0	0	0	0	1	0

(c)

処理の種類	T:Top， p_1:Convert Color， p_2:Convolution， p_3:ULBP， p_4:Max， B:Bottom
$cost_j$ $\forall j \in \{1,2,...,J\}$	1
$Cost_{max}$	25
$Cost_{p1}$	1
$Cost_{p2}$	4
$Cost_{p3}$	1
D_{max} を満たすチャンネル数	10

Q		T	p_1	p_2	p_3	p_4	B
	T	0	0	0	0	0	0
	p_1	1	0	0	0	0	0
	p_2	0	1	1	0	0	0
	p_3	0	1	1	0	0	0
	p_4	0	0	0	0	1	0
	B	0	0	0	1	0	0

(d)

処理の種類	T:Top, p1:Convert Color, p2:HOG, B:Bottom
$cost$ $\forall j \in \{1,2,...,J\}$	1
$Cost_{max}$	25

表 4.8　FTOP*の分割設定. (a)FTOP1, (b)FTOP2, (c)FTOP3, (d)FTOP4, (e)FTOP5.

$Cost_{p1}$	1				
$Cost_{p2}$	3				
D_{max} を満たすチャンネル数	10				
Q		T	p_1	p_2	B
	T	0	1	0	0
	p_1	0	0	1	0
	p_2	0	0	0	1
	B	0	0	0	0

(e)

処理の種類	T:Top, p_1:Convert Color, p_2:Box, p_3:Max, p_4:Arithmetic, p_6:Convolution, p_7:Norm, p_8:Orientation, p_9:Gradient, B:Bottom										
$cost_j$ $\forall j \in \{1,2,...,J\}$	1										
$Cost_{max}$	25										
$Cost_{p1}$	1										
D_{max} を満たすチャンネル数	10										
Q		T	p_1	p_2	p_3	p_4	p_6	p_7	p_8	p_9	B
	T	0	0	0	0	0	0	0	0	0	0
	p_1	1	1	0	1	1	0	1	0	0	0
	p_2	0	1	0	0	0	0	1	0	1	0
	p_3	0	1	0	1	0	1	0	0	0	0
	p_4	0	1	1	0	0	0	1	0	0	0
	p_6	0	0	0	0	0	0	1	0	1	0
	p_7	0	1	0	0	1	1	1	0	0	0
	p_8	0	0	0	0	0	0	0	1	0	0
	p_9	0	1	1	0	0	1	0	0	0	0
	B	0	0	0	1	0	0	0	0	0	0

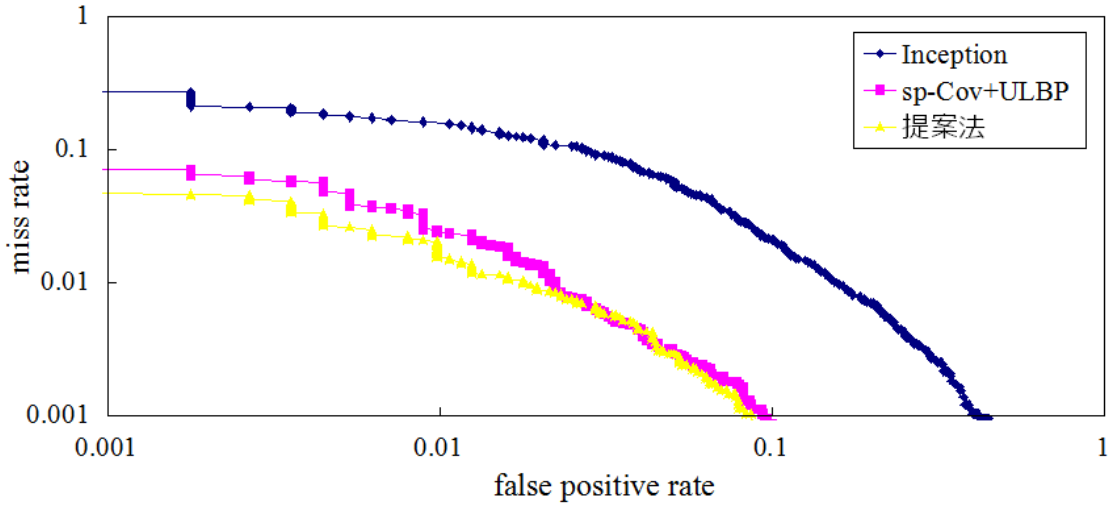


図 4.24　実験結果の DET.

表 4.9　実験結果.

	従来法		提案法
	Inception	sp-Cov+ULBP	
次元数	42368	66262	16009
FNR[%]	9.1474	3.3748	2.931
FPR[%]	2.3616	0.5277	0.424
AUC	0.009437	0.0009959	0.0007076
最適化[min]	4899	-	39164
学習(特徴量算出)[min]	19	267	58
学習(機械学習)[min]	1164	2323	623
識別時間[sec]	0.0804	1.154	0.222

4.4　従来法との比較実験に対する考察

従来法との比較実験に対して考察する．図 4.24 の DET は原点に DET 曲線が近いほど識別性能が高いことを表している．また，表 4.9 の AUC は小さいほど識別性能が高い．従って，図 4.24 と表 4.9 の AUC は提案法が従来法よりも本実験における識別性能が高いことを表している．また，FNR，FPR いずれも提案法が小さい結果が得られており，提案法が従来法に比べ有効であると考えられる．

表 4.9 より次元数について考察する．本実験のように AdaBoost を用いる場合，3.6 節における議論に従って特徴量の候補が多いが, 互いを補間する特徴量が少ない場合，学習時間はかかるが汎化性能が向上しないと考えられる．この観点で次

元数を見ると提案法は従来法に比べて最も小さいにも関わらず AUC が最も小さくなっている．すなわち，従来法よりも少ない特徴量だが，互いを補うような識別に有効な特徴量が算出されていると考えられる．ただし，提案法においては子問題に分割した時に従来の研究に基いた問題設定を与えているため，完全自動によって，上記のような特徴量を獲得したとは言い切れない．子問題の設定はよい初期解を早く得るための工夫であるが，子問題の設定においても表 4.4 の FTOP* の $Cost_{\max}$ のみを小さくした問題設定にすることにより，よりよい解が探索できた可能性はある．

表 4.9 より最適化に要した処理時間について考察する．表 4.9 の最適化時間は提案法が最も大きい．これは，求解時の評価値を算出する時に機械学習を実行しているためである．提案法において高速に評価値を計算することは多くの解を探索しよい解を得るために必要である．提案法では最適化における機械学習に用いる特徴量の次元を一定の割合でランダムに選択すること，学習サンプルを減らすこと，AdaBoost の弱識別器数を減らすことにより評価値の計算時間を削減している．これらは，評価値と未知画像に対する汎化性能とのトレードオフとなると考えられる．特徴量の次元を減らせば機械学習の時間が減り，評価値計算に要する時間が減る．3.6 節において議論したように最適化の 97.8%が評価値の計算に要していることを考えれば，多くの解を探索する効果が得られる．一方で，特徴量の次元をランダムに選択することにより減らす手法は，選ばれなかった特徴量に汎化性能を向上させる特徴量が含まれていた場合に，評価値と汎化性能の相関関係に対する精度を減らしてしまう恐れがある．これは学習サンプルを減らす場合，AdaBoost の弱識別器の数を減らす場合にも同様である．このように，評価値の計算時間を減らす工夫は，評価値と汎化性能との相関関係の精度とのトレードオフになる．理想的にはこの精度は保ったまま高速に最適化を実行したい．つまり，同等の評価値の計算を高速化する工夫が必要である．これにはソフトウェア，ハードウェア双方からのアプローチが考えられる．

まず，ソフトウェアからのアプローチではアルゴリズムの観点とソフトウェア工学の観点がある．アルゴリズムの観点では，評価値を足切りすることが考えられる．評価値と汎化性能の相関の精度が低い評価値計算の設定(上記における，特徴量の次元の選択数を減らす，学習サンプルの選択数を減らす，AdaBoost の弱識別器数を減らす)において評価値を計算し，基準を満たす場合にのみ，評価値と汎化性能の相関の精度が高い評価値計算の設定(上記における，特徴量の次元の選択数を増やす，学習サンプルの選択数を増やす，AdaBoost の弱識別器数を増やす)において評価値を計算することにより足きりする．ソフトウェア工学の観点では各 FTOP の求解を複数プロセスにより並列に処理させることが考えられる．FTN1～FTN5 は全て独立に計算可能である．さらに FTN1～FTN5 は異なる 3 つの乱数

シードを用いて多点局所探索を実行している．つまり，これも並列処理可能である．従って，提案法における FTOP1〜FTOP5 の求解時間は理論上 1/15 に圧縮可能である．さらに，もし FTN init に結合する FTN を増やしたい場合にもプロセスを割り当てることにより並列処理可能である．従って，計算機資源が許す限りスケールアウトによって子問題の求解時間は子問題数に対して定数倍に抑えられる．さらに，最適化における評価値計算の特徴量計算も同様に並列処理可能である．他にも, プロセスを割り当てることによる並列化は機械学習にも適用可能である．ただし, 提案法で用いた AdaBoost は弱識別器の構築がシーケンシャルになるため, プロセス割り当てによる並列処理には向かない．これは Boost による手法が当てはまる問題である．Random Forest のように Bagging の手法であれば，弱識別器の構築が独立に計算可能であるため, プロセス割り当てによる並列化に向いている．この点においては, プロセスによる並列化が可能なメニーコア CPU を搭載している計算機，もしくはマルチノード環境であれば Random Forest のような Bagging の手法を選択することにより評価値計算を高速に実行できる．

次に，ハードウェアからのアプローチである．最適化における評価値の処理時間は特徴量の算出(FTN による算出)と機械学習に分けられる．ここで，FTN の計算は全てテンソル計算である．従って，テンソルの要素毎に独立して並列に処理できる．これを実現するために GPGPU(General Purpose for Graphic Processing Unit) を用いることができる．また，一部の処理の種類(例えば Gabor フィルタ)においては FPGA(Field-Programmable Gate Array)による実装が提案されている[109].

表 4.9 より学習時間について述べる．本実験では最終的な識別器を構築するための AdaBoost は従来法と提案法とで全て同じ設定としている．従って, 学習の処理時間は特徴量の算出と次元数に依存する．学習時間は提案法が最も小さい．これは提案法の次元数が最も小さいことによる．FTOP の最適化によって少ない特徴量だが汎化性能がよい特徴量の計算式を導出できた結果であると考えられる．表 4.9 の特徴量算出の処理時間をみると提案法が 2 番目に小さいが，特徴量の算出よりも機械学習に要する時間のほうが大きいため，学習の処理時間としては提案法が最も小さくなっている．

表 4.9 より識別時間について述べる．識別時間は提案法が 2 番目に小さい．識別を高速化するには表 4.9 の最適化の処理時間にて述べた改善のほかに，FTOP の設定を変更することが考えられる．特徴量の算出時間はコスト *cost* に処理時間を設定することにより，特徴量計算時間を制限することができる．これは提案法のアプリケーションにあわせて設定が可能である．

4.5 まとめ

本章では FTOP を歩行者識別に適用する具体的な手法を詳述した．FTOP を識別問題に適用するためには，評価関数と機械学習の選定，制約に関わる FTOP のパラメータの指定，処理の種類の指定，最適化アルゴリズムのパラメータの指定が必要となる．FTOP を歩行者識別に適用する方法として，本章では評価関数に AUC，評価値を計算するための機械学習に AdaBoost，処理の種類に表色系変換，四則演算，ノルム，方向，勾配，畳み込みフィルタ，Box フィルタ，Max フィルタ，HOG，ULBP，Covariance を用いた．FTOP のパラメータは従来法よりも特徴量次元が小さくなるように設定した．また，FTOP の評価値計算に時間がかかるため，最適化問題を分割した．さらに，評価値の計算を短縮するために，最適化アルゴリズムのパラメータのうち評価値計算に用いるサンプル数を減らした．これを提案法として，従来法である sp-Cov と ULBP を用いる手法[74]と google の Inception 構造を用いた手法[75]と比較実験を行った．比較実験の結果，提案法が最も小さい AUC を得ることができた．最適化時間の短縮のためには，探索中の評価値計算に足きりを導入することや，スケールアウトにより並列に処理することが考えられる. また, 最適化及び識別時間を短縮するためには, GPGPU や FPGA を用いて特徴量計算を高速化することが考えられる.

第 5 章 結論

本研究では画像識別において特徴量の計算式を手動で設計する手法から自動で設計する手法まで取り組んだ．特徴量を手動で設計する手法では HOG の拡張法により歩行者識別精度が向上することを示した．また，この知見を含めて，特徴量の変換式を最適化する手法を提案した. 提案法では Neural Network とは異なり，特徴量の変換式に用いられる関数に微分可能であるという制約はない．特徴量の変換式を最適化するためのデータ構造である FTN，FTN を用いた最適化問題 FTOP 及びその求解アルゴリズムを提案した．提案法は式(3.13)により計算される評価値が汎化性能と相関があるという仮説に基づいており，実験的にこの仮説を検証した．その結果，評価値と汎化性能の指標である AUC との間には相関係数 0.8973 が得られた．これにより，FTOP を求解すれば汎化性能が向上するという結論を得た．また，CNN の一種である Inception 及び sp-Cov と ULBP と提案法との比較実験により提案法の優位性が確認された.

特徴量の変換式の最適化における課題は最適化時の評価値の計算速度と FTOP の設定である．まず，最適化時の評価値の計算速度の課題を述べる.実験では最適化に 39164 分を要しており，これは約 27 日に相当する．評価値の計算が最適化における処理時間の 97%以上を占めているため，評価値の計算時間を短くすることが多くの解を探索し汎化性能を向上させることにつながる．評価値の計算を高速化するためには評価値の計算に足きりを採用することや，機械学習に Bagging の手法を採用し並列に処理することが考えられる．また，特徴量の算出に GPGPU や FPGA を用いることが検討される. 次に, FTOP の設定における課題を述べる．実験では求解の効率化のために処理の種類の接続を制限した子問題に分割した．しかし，この接続を制限せず時間をかければ，多くの解が探索の候補となり，よりよい解を得られる可能性がある．上記の最適化における処理時間の高速化が実現できれば，この制限が外せると考えられる．また，FTN に含める処理の種類の選択基準の整理が必要である．提案法では従来法を参考に処理の種類を決定したが，新しい特徴量の計算式が提案される度に増やしていく，特徴量の計算式の要素を整理しその要素のみを含める，というように処理の種類に何を選択するかにより汎化性能は異なると考えられる.

本研究にて提案した FTOP は歩行者識別のみならず，クラス分類問題全てに適用可能な手法である．従って，FTOP の適用先を増やし，その効果を検証することが今後の展望である.

参考文献

[1] 青木正喜，安田升：“車載赤外線カメラを用いた歩行者検出”，情報処理学会論文誌，Vol. 47，pp.1-9，2006.

[2] 尾形一気，大矢晃久：“巡回警備ロボットのための赤外線カメラを用いた人間検出(ロボットビジョン)”，ロボティクス・メカトロニクス講演会講演概要集 2011，pp.2A1-L02(1) - 2A1-L02(4)，2011.

[3] 横矢直和，山澤一誠，竹村治雄：“全方位ビデオカメラを用いた視覚情報メディア”，情報処理学会論文誌コンピュータビジョンとイメージメディア（CVIM） 42(SIG13(CVIM3))，pp.59-70，2001.

[4] 中島一斗，河村晃宏，倉爪亮：“マルチモーダル全方位画像の共起性と循環性を考慮した畳み込み特徴学習による一般屋外環境識別”，日本ロボット学会学術講演会，p.RSJ2017AC1H2-01，2017.

[5] 首藤幸司，西尾信彦：“センサフュージョンを利用した個人行動の未来予測機構”, 情報処理学会研究報告ユビキタスコンピューティングシステム（UBI）2006(116(2006-UBI-012))，pp.79-84，2006.

[6] 米谷和記，三好力：“センサフュージョンを利用した個人行動の未来予測機構”，情報処理学会全国大会講演論文集 2013(1)，pp.461-462，2013.

[7] 警察庁交通局：“平成 21 年中の交通死亡事故特徴及び道路法違反取締り状況について”，pp.1-48，2010.

[8] K. Fukushima: “Neocognitron: a self organizing NN model for a mechanism of pattern recognition unaffected by shift in position”, *Biological Cybernetics* 36, pp.193-202, 1980.

[9] B. Julesz: “Textons, the elements of texture perception, and their interactions”, *Nature* 290(5802), pp.91-97, 1981.

[10] Y. Le Cun, O. Matan, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, L. D. Jacket, H. S. Baird.: “Handwritten zip code recognition with multilayer networks”, *Proceeding of the International Conference on Pattern Recognition*, pp.35-40, 1981.

[11] V. Govindaraju, S. N. Srihari, D. B. Sher: “A computational model for face location”, *Proceedings of IEEE International Conference on Computer Vision*, pp.718‑721, 1990.

[12] K.-K. Sung, T. Poggio: “Example-Based learning for view-based human face detection”, *Jounal of the IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.20, pp.39-51, 1998.

[13] J. Yang, A. Waibel: “A real-time face tracker”, *Proceedings of Workshop on*

Applicationsof Computer Vision, pp.142-147, 1996.

[14] T. Ojala, M. Pietikainen, D. Harwood: “A comparative study of texture measures with classification based on featured distributions”, *Journal of the Pattern Recognition*, Vol. 29, No. 1, pp. 51‑59, 1996.

[15] E. Osuna, R. Freund, F. Girosi: “Training support vector machines: an application to face detection”, *Proceedings of IEEE International Conference on Computer Vision*, pp. 130‑136, 1997.

[16] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner: “Gradient-based learning applied to document recognition”, *Proceedings of the IEEE* 86(11), pp.2278‑2324, 1998.

[17] Y. Wang, D. Shen, E. K. Teoh: “Lane detection using catmull-rom spline”, *Proceedings of the 1998 IEEE International Conference on Intelligent Vehicles*, Vol. 1, pp.51-57, 1998.

[18] A. Mohan: “Object detection in images by components”, *Proceedings of CBCL Paper Massachusetts Institute of Technology*, pp.1-22, 1999.

[19] C. Papageorgiou, T. Poggio: “A Trainable System for Object Detection”, *International Journal of Computer Vision*, Vol. 38, No. 1, pp. 15‑33, 2000.

[20] 平山高嗣，岩井儀雄，谷内田正彦：”顔認識のためのスケール変化に応じた特徴量抽出”，情報処理学会研究報告コンピュータビジョンとイメージメディア(CVIM)”，2001-CVIM-128，pp.33-40，2001.

[21] T. Leung and J. Malik: “Representing and recognizing the visual appearance of materials using three-dimensional textons”, *International Journal of Computer Vision* 43, pp.29–44, 2001.

[22] P. Viola, M. Jones: “Robust real-time object detection”, *Proceedings of the 2nd International Workshop on Statistical and Computational Theories of Vision-Modeling, Learning, Computing, and Sampling*, pp.1-25, 2001.

[23] R. Lienhart, J. Maydt: “An extended set of haar-like features for rapid object detection”, *Proceedings of International Conference on Image Processing*, Vol. 1, pp. 900‑903, 2002.

[24] P. Viola, M. Jones, D. Snow: “Detecting pedestrians using patterns of motion and appearance”, *Proceedings of IEEE International Conference on Computer Vision*, pp.734‑741, 2003.

[25] D. G. Lowe: “Distinctive image features from 12 scale-invariant keypoints”, *Journal of Computer Vision*, 60, 2, pp.91-110, 2004.

[26] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, C. Bray: “Visual categorization with bags of keypoints”, *Proceedings of European Conference on Computer Vision*, pp.1-2, 2004.

[27] K. Levi, Y. Weiss: “Learning object detection from a small number of examples: the importance of good features”, *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Vol. 2, pp. 53‐60, 2004.

[28] L. Fei-Fei, P. Perona: “A bayesian hierarchical model for learning natural scene categories”, *Proceedings of Computer Vision and Pattern Recognition*, pp.524-531, 2005.

[29] N. Dalal, B. Triggs: “Histograms of oriented gradients for human detection”, *Proceedings* of *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.886-893, 2005.

[30] F. Porikli: “Integral histogram: a fast way to extract histograms in cartesian spaces”, *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 1, pp. 829‐836, 2005.

[31] 三田雄志, 金子敏充, 堀修: “顔検出に適した Joint Haar-like 特徴の提案”, 画像の認識・理解シンポジウム(MIRU), pp. 104‐111, 2005.

[32] N. Dalal, B. Triggs, C. Schmid: “Human detection using oriented histograms of flow and appearance”, *Proceedings of European Conference on Computer Vision*, Vol. 2, pp.428‐441, 2006.

[33] B. Wu, R. Nevatia: “Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors”, *Proceedings of IEEE International Conference on Computer Vision*, pp. 90‐97, 2005.

[34] 高木雅成, 藤吉弘亘 : “SIFT 特徴量を用いた交通道路標識認識”, 第 13 回画像センシングシンポジウム, p.LD2-06, 2007.

[35] C. Hou, H. Z. Ai, S. H. Lao: “Multiview Pedestrian Detection Based on Vector Boosting”,*Proceedings of Asian Conference on Computer Vision*, pp. 210‐219, 2007.

[36] A. Bosch, A. Zisserman, X. Munoz: “Representing shape with a spatial pyramid kernel”, *International Conference on Image and Video Retrieval*, pp.401-408, 2007.

[37] Y. T. Chen, C. S. Chen: “A cascade of feed-forward classifiers for fast pedestrian detection”, *Proceedings of Asian Conference on Computer Vision*, pp. 905‐914, 2007.

[38] J. Yao, J. M. Odobez: “Multi-Layer background subtraction based on color and texture”, *Proceedings of Computer Vision and Pattern Recognition*, pp.1-8, 2007.

[39] E. Shechtman, M. Irani: “Space-Time behavior-based correlation-or-how to tell if two underlying motion fields are similar without computing them?”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 29, No. 11, pp. 2045‐56, 2007.

[40] Z. Lin, L. Davis, D. Doermann: “Hierarchical part-template matching for human detection and segmentation”, *Proceedings of IEEE International Conference on Computer Vision*, p.1-8, 2007.

[41] O. Tuzel, F. M. Porikli, P. Meer: “Human Detection via Classification on Riemannian Manifolds”, *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp.1-8, 2007.

[42] 大西克則，滝口哲也，有木康雄：“HOG 特徴に基づく単眼画像からの人体 3 次元姿勢推定”，画像の認識・理解シンポジウム(MIRU)，pp.960-965，2008.

[43] A. Bosch, A. Zisserman, X. Munoz,: “Representing shape with a spatial pyramid kernel”, *Proceedings of the 6th ACM international conference on Image and video retrieval*, pp.401-408, 2007.

[44] J. Yao, J. Odobez: “Fast human detection from videos using covariance features”, *Proceedings of Visual Surveillance Workshop(in conjunction with ECCV*2008*)*, pp.1-10, 2008.

[45] F. Ren, J. Huang, R. Jiang, R. Klette: “Lane detection on the iPhone”, Proceedings of First International Conference ArtsIT, pp.198-205, 2009.

[46] 伊原有仁:“異なる部分空間における PCA-SIFT を用いた交通道路標識認識”，第 14 回画像センシングシンポジウム SSII08，p.1-8, 2008.

[47] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, D. Ramanan: “Object detection with discriminatively trained part based models”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 32, No. 9, pp. 1627‑1645, 2009.

[48] X. Wang, T. X. Han, S. Yan: “An HOG-LBP human detector with partial occlusion handling”, *Proceedings of IEEE International Conference on Computer Vision*, pp.808-820, 2009.

[49] T. Watanabe, S. Ito, K. Yokoi: “Co-occurrence histograms of oriented gradients for pedestrian detection”, *Proceedings of Pacific-Rim Symposium on Image and Video Technology*, pp. 37‑47, 2009.

[50] J. Gall, V. Lempitsky: “Class-Specific Hough Forests for Object Detection”, *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp.1022-1029, 2009.

[51] 三井相和，山内悠嗣，藤吉弘亘: “Joint 特徴量を用いた 2 段階 Boosting による物体検出”，電子情報通信学会論文誌，Vol. J92-D, No. 9, pp. 1591‑1601, 2009.

[52] P. Ott, M. Everingham: “Implicit color segmentation features for pedestrian and object detection”, *Proceedings of IEEE International Conference on Computer Vision*, pp.723-730, 2009.

[53] P. Doll ´ ar, Z. Tu, P. Perona, S. Belongie: “Integral channel features”, *Proceedings of British Machine Vision Conference*, pp.91.1-91.11, 2009.

[54] X. Xia, W. Yang, H. Li, S. Zhang: “Part-Based object detection using cascades of boosted classifiers”, *Proceedings of Asian Conference on Computer Vision*, pp.556-565, 2009.

[55] 山田寛，松尾直志，島田伸敬，白井良明：“手話認識のための見えの学習による手領域検出と形状識別”，画像の認識・理解シンポジウム(MIRU)，pp.635-642，2009.

[56] 東久保政勝，荻内康雄，小野佑樹，栗田多喜夫，西田健次：“SVM による検出とペア特徴追跡の組み合わせによる車両・二輪車計測”，電気学会電子・情報・システム部門大会講演論文集，TC14-2，pp.510-513，2010.

[57] C. Kanan, G. Cottrell: “Robust classification of objects, faces, and flowers using natural image statistics”, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.2472-2479, 2010.

[58] S. Walk, N. Majer, K. Schindler, B. Schiele: “New features and insights for pedestrian detection”, *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1030‑1037, 2010.

[59] 池村翔，藤吉弘亘：“距離情報に基づく局所特徴量によるリアルタイム人検出”，電子情報通信学会論文誌，Vol. 93-D，No. 3，pp. 355‑364，2010.

[60] M. Enzweiler, A. Eigenstetter, B. Schiele, D. Gavrila: “Multi-Cue pedestrian classification with partial occlusion handling”, *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 990‑997, 2010.

[61] 三井相和，藤吉弘亘，秋田時彦：“近似 Texton 特徴量を用いた高速な人検出法”，電子情報通信学会論文誌. D，情報・システム J94-D(7)，pp.1135-1144 ，2011.

[62] 堀貴博，滝口哲也，有木康雄：“グラフ構造表現による一般物体認識”，電子情報通信学会技術研究報告，pp.19-24 ， 2011.

[63] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, A. Blake: “Real-time human pose recognition in parts from single depth images”, *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp.1297-1304, 2011.

[64] L. Xia, C. Chen, J. Aggarwal: “Human detection using depth information by kinect”, *Proceedings of International Workshop on Human Activity Understanding rom 3D Data(in conjunction with CVPR)*, pp. 15‑22, 2011.

[65] R. Nosaka, Y. Ohkawa, K. Fukui: “Feature extraction based on co-occurrence of

adjacent local binary patterns", *Proceedings of Pacific-Rim Symposium on Image and Video Technology*, pp.82-91, 2011.

[66] 波部斉, ロベルトチポラ : "Joint Hough Forests: 局所パッチ間の共起関係を考慮した投票ベースの物体検出", 画像の認識・理解シンポジウム MIRU2011, pp.381-386, 2011.

[67] A. Krizhevsky, I. Sutskever, G. E. Hinton: "ImageNet classification with deep convolutional NNs", *Proceedings of Neural Information Processing Systems(NIPS)*, pp.1106‐1114, 2012.

[68] 山内悠嗣, 金出武雄, 山下隆義, 藤吉弘亘 : "量子化残差に基づく特徴量の遷移尤度モデルを導入した識別器の提案", 画像の認識・理解シンポジウム (MIRU2011)論文集, pp.373-380, 2011.

[69] T. R. Almaev, M. F. Valstar: "Local Gabor binary patterns from three orthogonal planes for automatic facial expression recognition", *Proceedings of Humaine Association Conference on Affective Computing and Intelligent Interaction*, pp. 356‐361, 2013.

[70] P. F. Alcantarilla, J. Nuevo, A. Bartoli: "Fast explicit diffusion for accelerated features in nonlinear scale spaces", *In British Machine Vision Conference*, pp.1-11, 2013.

[71] Joseph J. Lim, C. Lawrence Zitnick, Piotr Dollar: "Sketch Tokens: A learned mid-level representation for contour and object detection", *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp.*3158-3165, 2013.

[72] K. Simonyan, A. Zisserman: "Very deep convolutional networks for large-scale image recognition", *International Conference on Learning Representations*, arXiv:1409.1556, pp.1-14, 2014.

[73] R. Girshick, J. Donahue, T. Darrell, J. Malik: "Rich feature hierarchies for accurate object detection and semantic segmentation", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.580-587, 2014.

[74] S. Paisitkriangkra, C. She, A. Van Den Hengel: "Strengthening the effectiveness of pedestrian detection", *in proceeding of European Conference on Computer Vision*, part IV, pp.546-561, 2014.

[75] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich: "Going deeper with convolutions", *Proceedings of IEEE CVPR*, arXiv:1409.4842, pp.1-9, 2015.

[76] J. Redmon, S. Divvala, R. Girshick, A. Farhadi: "You only look once: unified, real-time object detection", *Proceedings of Neural Information Processing Systems*, arXiv:1506.02640, pp.1-10, 2015.

[77] X. Du, M. El-Khamy, J. Lee, L. S. Davis: “Fused DNN: A deep NN fusion approach to fast and robust pedestrian detection”, *IEEE Winter Conference on Applications of Computer Vision*, arXiv:1610.03466, p.1-11, 2016.

[78] K. He, X. Zhang, S. Ren, J. Sun: “Deep residual learning for image recognition”, *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, arXiv:1512.03385, p.1-12, 2016.

[79] B. Zoph, Q. V. Le: “Neural architecture search with reinforcement learning”, *International Conference on Learning Representations*, arXiv:1611.01578, pp.1-16, 2016.

[80] S. Wang, J. Cheng, H. Liu, M. Tang, “PCN: part and context information for pedestrian detection with CNNs”, *British Machine Vision Conference*, arXiv:1804.04483, pp.1-13, 2017

[81] J. Redmon, A. Farhadi: “YOLOv3: An incremental improvement”, *University of Washington Technical Report*, arXiv:1804.02767, pp.1-6, 2018.

[82] Z. Zhong, J. Yan, W. Wu, J. Shao, C. L. Liu, “Practical block-wise NN architecture generation”, *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, arXiv:1708.05552, p.1-11, 2018.

[83] S.G. Mallat, “A theory for multiresolution signal decomposition: the wavelet representation”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp.674-693, 1989.

[84] J. P. Jones , L. A. Palmer, “An evaluation of the two-dimensional Gabor filter model of simple receptive fields in cat striate cortex”, *Journal of Neurophysiology*, pp.1233-1258, 1987.

[85] P. Viola, M. Jones: “Robust real-time object detection”, *Proceedings of the 2nd International Workshop on Statistical and Computational Theories of Vision-Modeling, Learning, Computing, and Sampling*, pp.1-25, 2001.

[86] MNIST Database, http://yann.lecun.com/exdb/mnist/, 最終アクセス日 2018/05/25.

[87] LSVRC, http://www.image-net.org/challenges/LSVRC/, 最終アクセス日 2018/05/25.

[88] ImageNet, http://www.image-net.org/, 最終アクセス日 2018/05/25.

[89] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, A. W. M. Smeulders, “Selective search for object recognition”, *International Journal of Computer Vision*, Vol.104, pp.154-171, 2013.

[90] INRIA Person Dataset, http://pascal.inrialpes.fr/data/human/, 最終アクセス日 2018/05/25.

[91] S. Hochreiter, J. Schmidhuber: “Long short-term memory”, *Neural Computation*,

pp.1735‐1780, 1997.

[92] J. J. Hopfield, “NNs and physical systems with emergent collective computational abilities”, *Proceedings of the National Academy of Sciences of the Unitd States of America*, pp.2554-2558, 1982.

[93] C. J. C. H. Watkins, “Learning from delayed rewards”, *Ph.D Thesis in King's College*, pp.1-241, 1989.

[94] J. Su, D. V. Vargas, K. Sakurai: “One pixel attack for fooling deep NNs”, *Kyushu University Technical Report*, arXiv:1710.08864, p.1-11, 2017.

[95] A. Athalye, Logan Engstrom, A. Ilyas, K. Kwok: “Synthsings Robust Adversarial Examples”, *MIT Technical Report*, arXiv:1707.07397, pp.1-18, 2017.

[96] ONNX, https://onnx.ai/, 最終アクセス日 2018/05/25.

[97] NNEF, https://www.khronos.org/nnef, 最終アクセス日 2018/05/25.

[98] P. Sermanet, K. Kavukcuoglu, S. Chintala, Y. Lecun: “Pedestrian detection with unsupervised multi-stage feature learning”, *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp.3626-3633, 2013.

[99] 宮本裕一郎：“混載輸送ネットワーク設計”, 第 4 回 KSMAP, pp.1, 2002.

[100] Y. Freund, R. E. Schapire: “A decision theoretic generalization of on-line learning and an application to boosting”, *Journal of Computer and System Sciences*, Vol. 55, pp.119-139, 1997.

[101] OpenCV, https://opencv.org/, 最終アクセス日 2018/05/25.

[102] R ada パッケージ, https://cran.r-project.org/web/packages/ada/index.html

[103] Python scikit-learn, http://scikit-learn.org, 最終アクセス日 2018/05/25.

[104] T. Ojala, M. Pietikainen, T. Maenpaa: “Multiresolution gray-scale and rotation invariant texture classification with local binary patterns”. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24, No. 7, pp. 971‐987, 2002.

[105] Caltech Pedestrian Dataset Benchmark, http://www.vision.caltech.edu/Image_Datasets/CaltechPedestrians/, 最終アクセス日 2018/05/25.

[106] caffe, http://caffe.berkeleyvision.org/, 最終アクセス日 2018/05/25.

[107] GoogLeNet, http://dl.caffe.berkeleyvision.org/bvlc_googlenet.caffemodel, 最終アクセス日 2018/05/25.

[108] J. D. Guilford, B. Frutcher: “Fundamental statistics in psychology and education”, *New York: McGraw Hill Book Company*, pp.1-576, 1981.

[109] 森江隆，岩田穆：“脳機能に学ぶ画像認識集積システム”，電子情報通信学会技術研究報告，CAS，回路とシステム 102(162)，pp.67-78，2002.

[110] Robert E. Schapire, Yoram Singer: “Improved boosting algorithms using confidence-rated predictions”, *Journal on Machine Learning*, Vol. 37, pp.297-336, 1999.

[111] Y. Nakashima, J. K. Tan, S. Ishikawa, T. Morie: “On detecting a human and its body direction from a video”, *Journal of Artificial Life and Robotics*, Vol. 15, pp.455-458, 2010.

[112] 中島佑樹，タンジュークイ，石川聖二，森江隆：“HOG 特徴量と人マスクを用いた人物および身体方向の検出”，画像電子学会誌，Vol.39，No.6，pp.1104-1111，2010.

[113] Y. Nakashima, J. K. Tan, S. Ishikawa, T. Morie: “Detecting a human body direction using Multiple-HOG”, *Proceedings of the First International Symposium on Future Active Safety Technology toward Zero-traffic-accident*, TS3-8-2-5, pp.1-6, 2011.

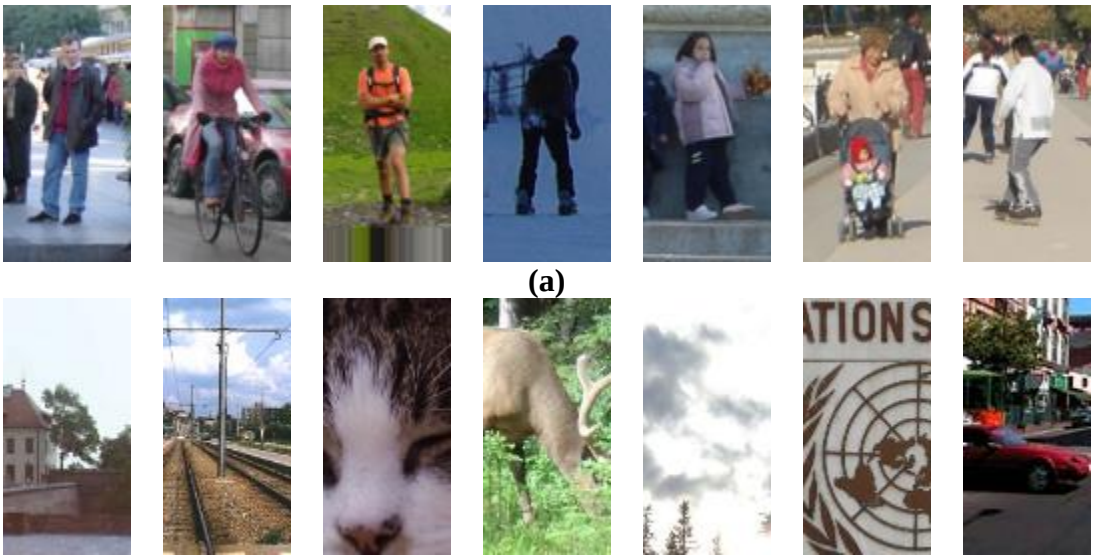
[114] C. Gini: “Sulla misura della concentrazione e della variabilita dei caratteri”, *Atti del R. Istituto Veneto di S. L. A.*, p.73, 1913.

謝辞

本研究を進めるにあたり，素晴らしい研究機会を与えてくださり，様々な相談や論文の添削に応じていただきましたタンジュークイ准教授，本論文審査において多くのご助言をいただきました本学大学院工学研究院の大屋勝敬教授，金亨燮教授，生命体工学研究科の森江隆教授に厚くお礼申し上げます．社会人ということもあり研究室を訪れることが少なかったにも関わらず温かく迎えてくださりました研究室メンバーに感謝の意を表します．また，丹博士には卒業後も相談に乗っていただき感謝しております．皆様のおかげで悔いのない充実した研究生活を送ることができました．最後に，社会人をしながら大学生活を送ることを許してくださった妻の祐香と娘の望に感謝いたします.本当にありがとうございました.

付録

各実験に用いた INRIA Person Dataset の一部を図 A.1 に示す．INRIA Person Dataset では図 A.1(a)のように老若男女，スキー，自転車に乗っている，といった様々な歩行者の画像が集録されている．また(b)のように Negative サンプルには街中，自然，動物，建物，車といった様々な歩行者ではない画像が集録されている．

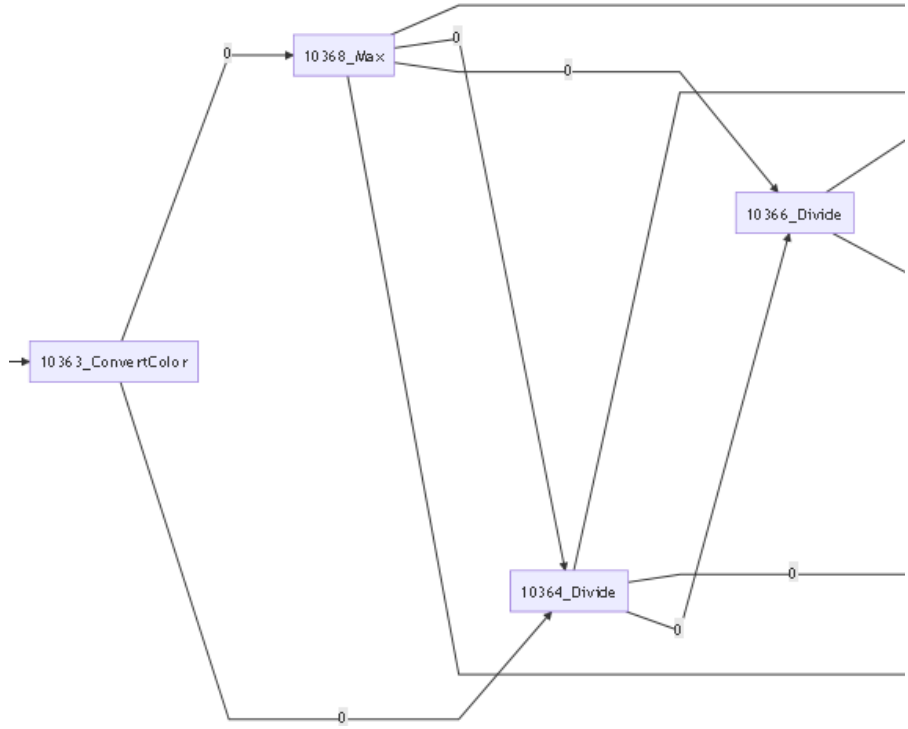


(a)

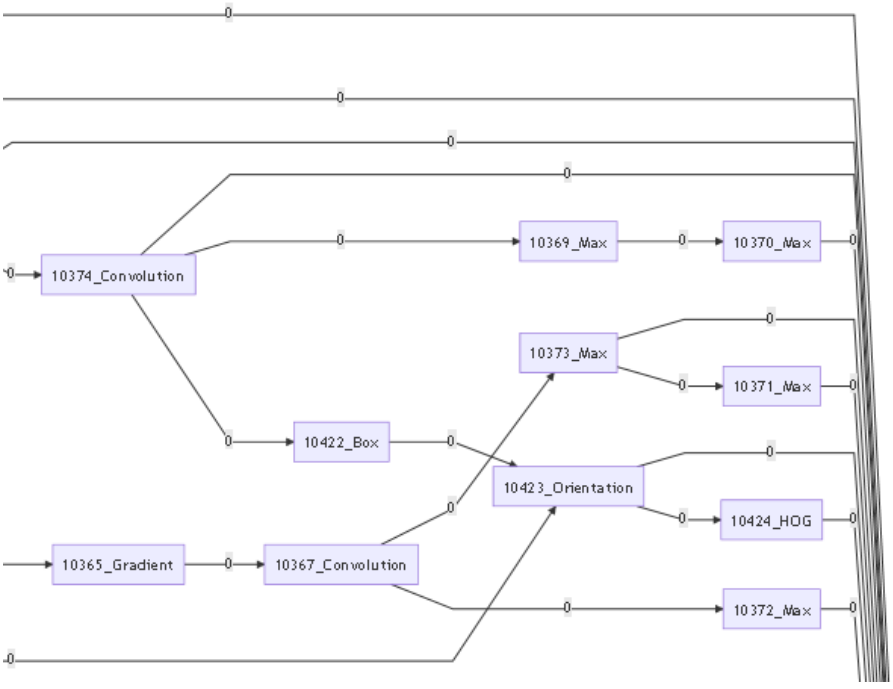
(b)

図 A.1　INRIA Person Dataset の一部．(a)歩行者(Positive サンプル)，(b)歩行者ではない画像(Negative サンプル).

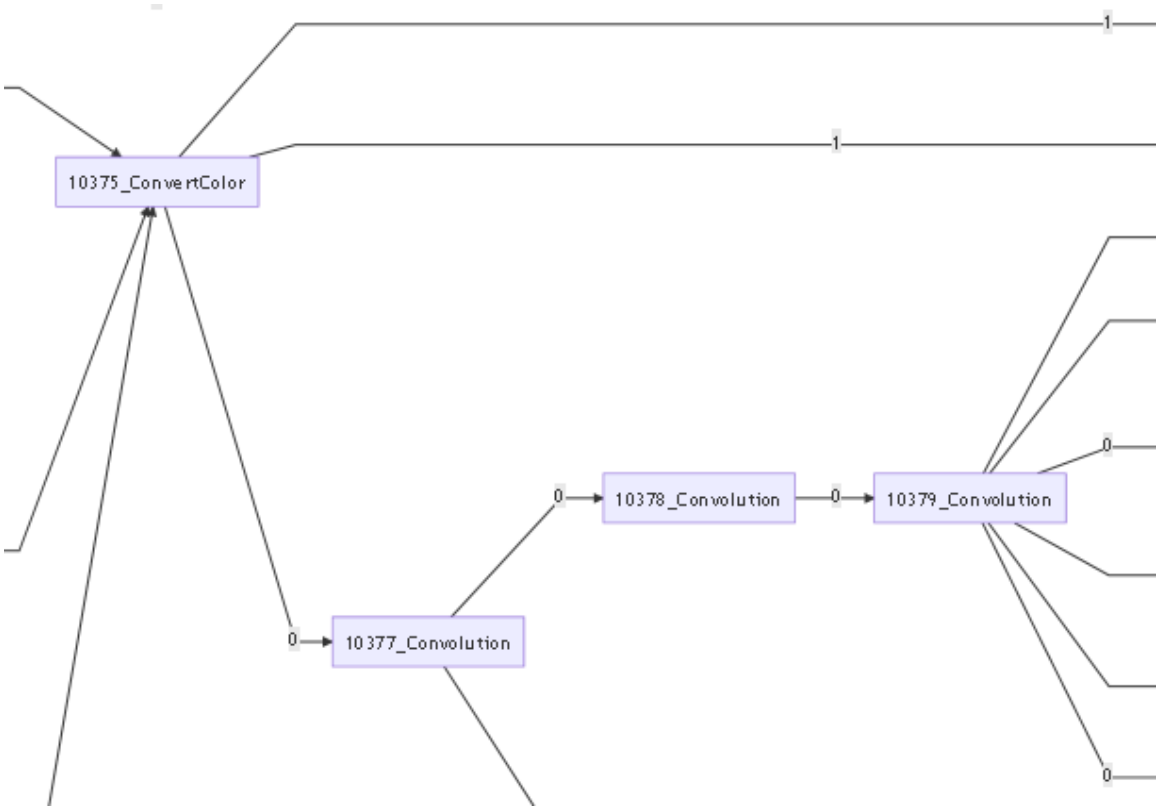
実験にて得られた FTN を図 A.2 に, 図 A.2 よりえられる特徴量を図 A.3 に示す. 図 A.2 は各部を拡大した図を添付している.



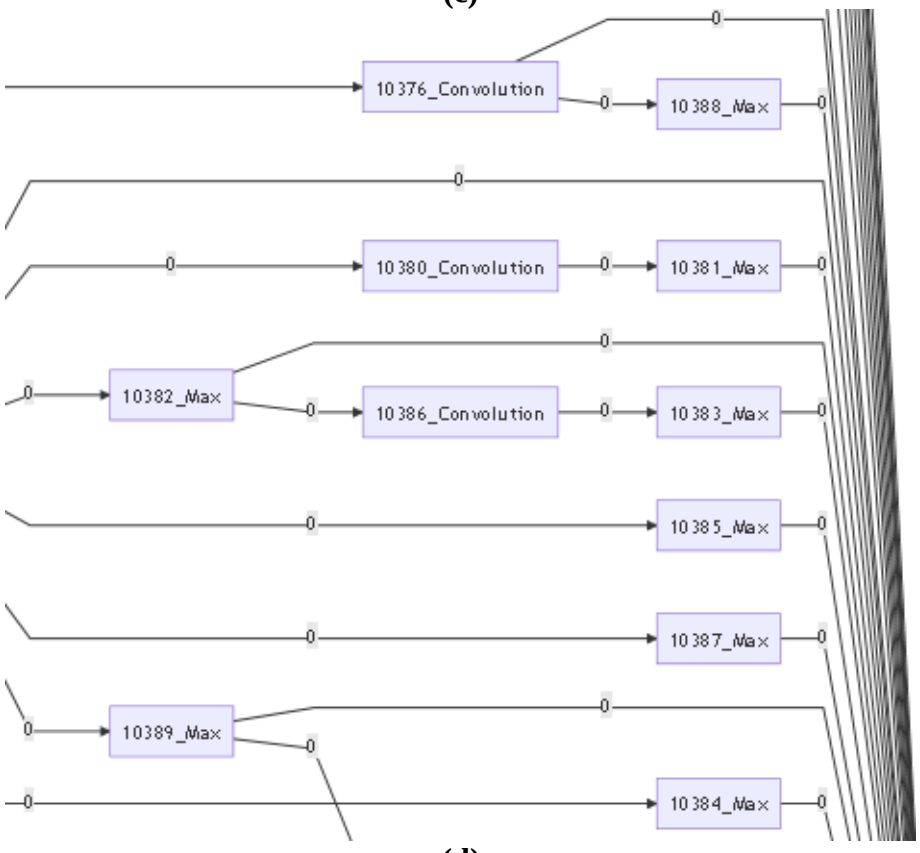
(a)



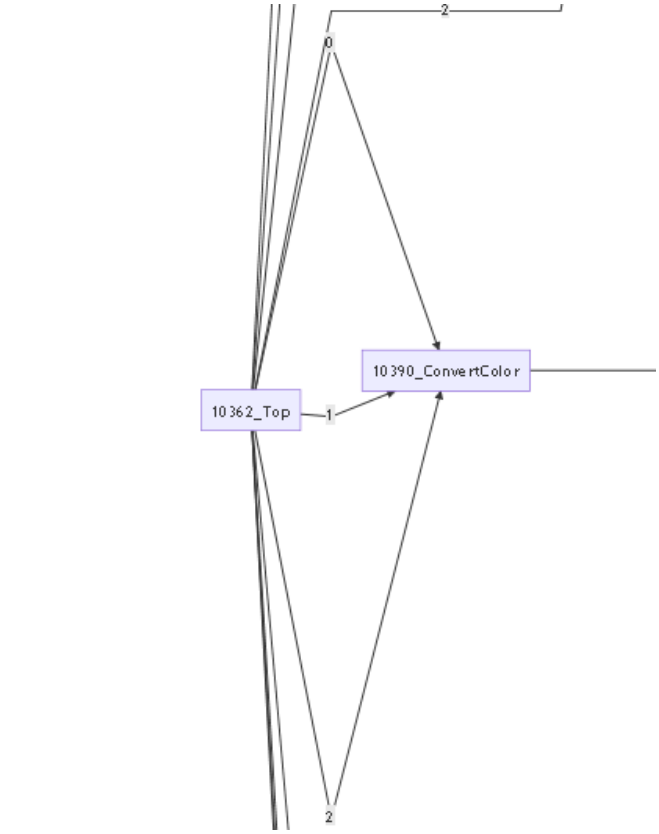
(b)


10375_ConvertColor
10377_Convolution
10378_Convolution
10379_Convolution

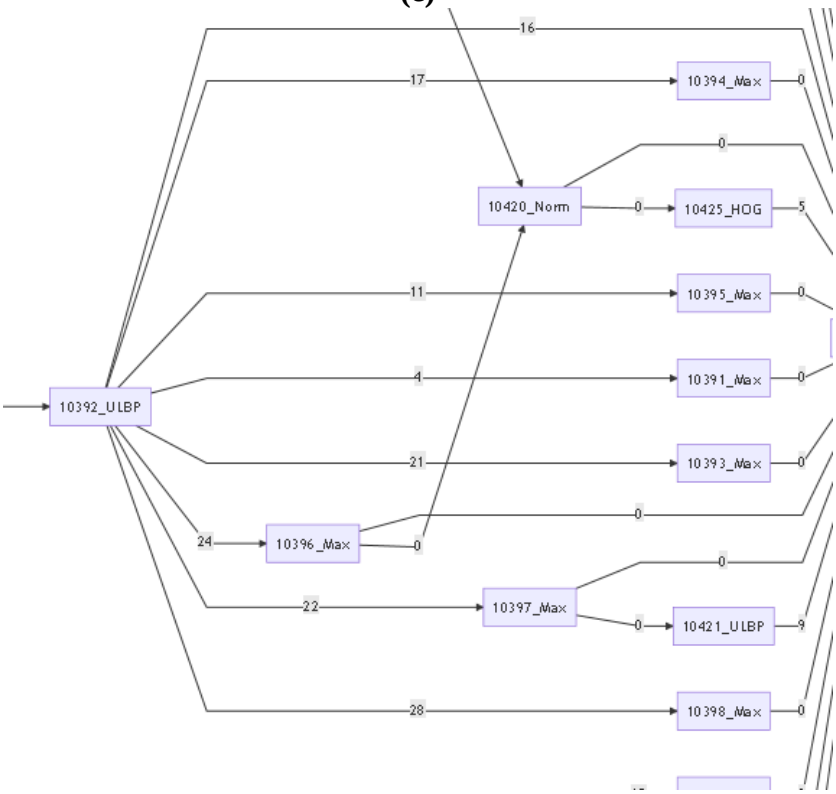
(c)


10376_Convolution
10388_Max
10380_Convolution
10381_Max
10382_Max
10386_Convolution
10383_Max
10385_Max
10387_Max
10389_Max
10384_Max

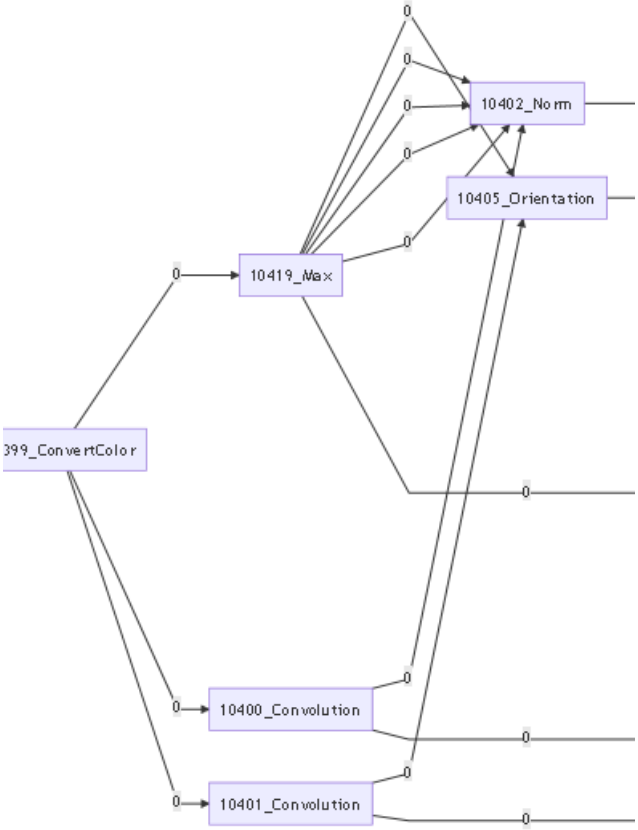
(d)


2
0
10390_ConvertColor
10362_Top
1
2

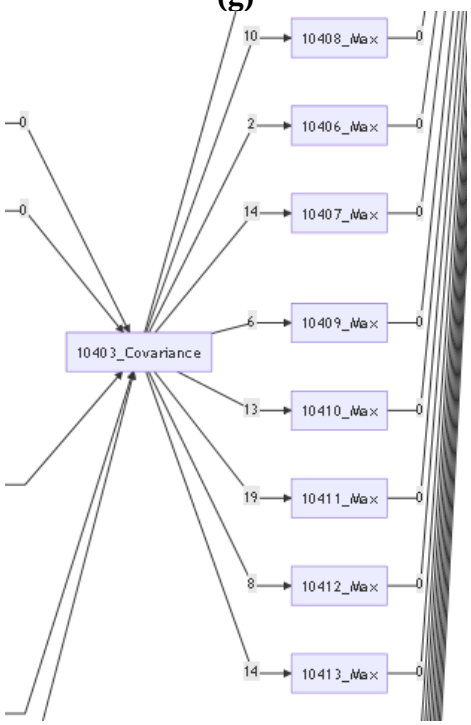
(e)


16
17
10394_Max
0
0
10420_Norm
0
10425_HOG
5
11
10395_Max
0
4
10391_Max
0
10392_ULBP
21
10393_Max
0
0
24
10396_Max
0
0
22
10397_Max
0
10421_ULBP
9
28
10398_Max
0

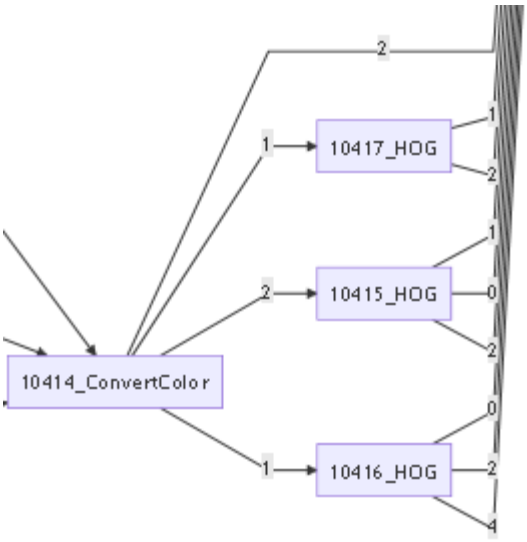
(f)


10402_Norm
10405_Orientation
10419_Max
399_ConvertColor
10400_Convolution
10401_Convolution

(g)


10408_Max
10406_Max
10407_Max
10409_Max
10403_Covariance
10410_Max
10411_Max
10412_Max
10413_Max

(h)



(i)

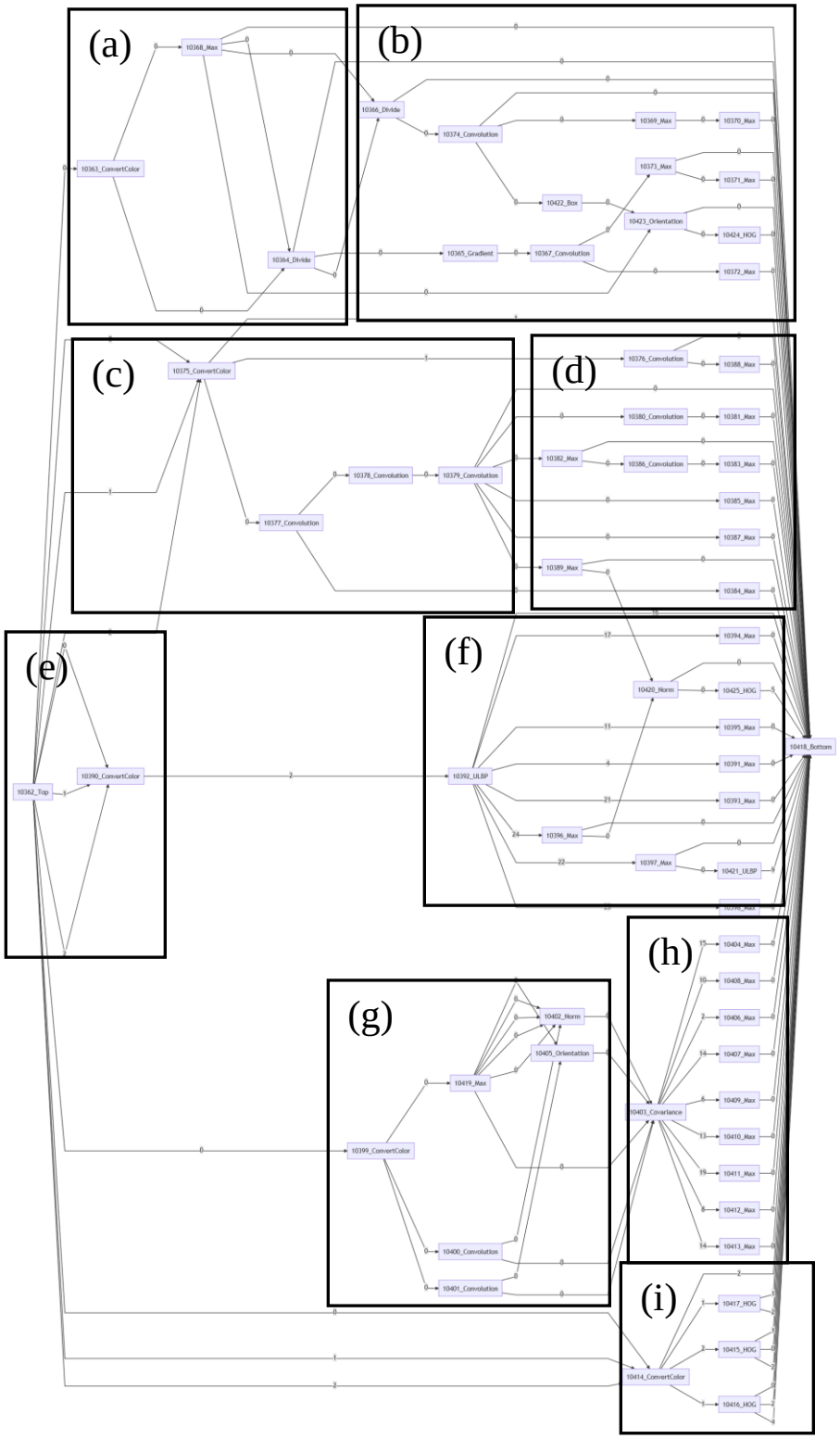


図 A.2　実験にて得られた FTN.

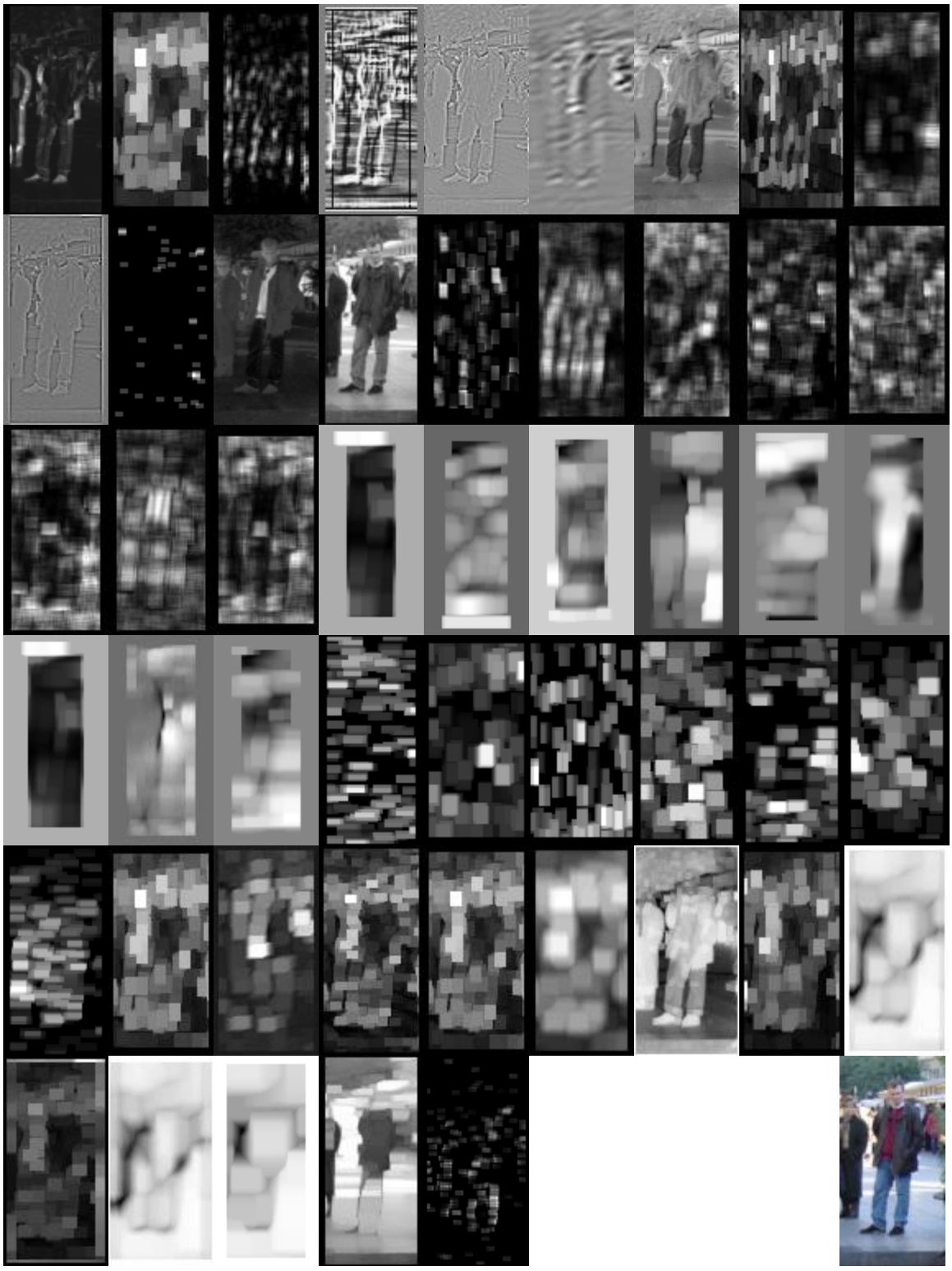


図 A.3　実験にて得られた FTN の出力.