線形SVMの近似計算による早期判定を用いたラスタスキャンの高速化

後藤 雄飛† 土屋 成光† 山内 悠嗣† 藤吉 弘亘† 秋田 時彦‡

†中部大学 ‡アイシン精機株式会社

E-mail: yuhi@vision.cs.chubu.ac.jp

Abstract

物体検出は，入力画像を網羅的にラスタスキャンさせて得られる，膨大な数の検出ウィンドウを検出対象か非検出対象に分類する．各検出ウィンドウは，抽出した局所特徴量と統計的学習手法により学習した識別器により出力値を算出し，対象のクラスに分類する．近年では，物体検出に一般的に利用されている線形SVMにより学習した識別器を，対応点追跡の問題において近似計算をして高速な識別を実現する方法がある．そこで，本研究では物体検出に対して線形SVMの近似計算法を導入し，ラスタスキャンの高速化を検討する．さらに，提案手法では近似計算結果に応じて識別器の出力値を早期判定する．これにより，入力された検出ウィンドウに対して，近似計算に必要な基底数を適応的に判断して分類することで線形SVMの高速な識別を可能にする．また，提案手法では線形SVMを近似計算する際に用いる，HOG特徴量をバイナリコード化したB-HOG特徴量を共起表現して物体検出の高精度化を行う．人検出による評価実験より，提案手法はHOG特徴量と線形SVMによる物体検出手法と比較して，約6.1% 高精度な識別に加えて，約17倍高速な識別計算を実現した．

1 はじめに

物体検出技術は，車載カメラによるIntelligent Transport System(ITS:高度道路交通システム)，監視カメラによるセキュリティ，マーケティング等の多くの分野での実用化が期待されている．物体検出は，学習サンプルから抽出した局所特徴量と統計的学習手法を用いて学習した識別器により実現されている．代表的な物体検出として，2005年にDalalらはHOG特徴量と線形SVM[1]による人検出手法[2]を提案した．HOG特徴量と線形SVMの組み合わせは，検出対象の隠れの推定[3][4]やパーツモデル[5]等の物体検出の高精度化に関する研究においても，ベースとなる手法として用いられている．このような物体検出の実用化には識別性能の高精度化だけではなく，検出過程の高速化と省メモリ化が課題である．検出過程では，入力画像を網羅的にラスタスキャンした際に発生する膨大な数の検出ウィンドウを処理する必要があり，高速な物体検出を実現するには各検出ウィンドウの処理である特徴抽出過程と識別過程をそれぞれ高速化する必要がある．特徴抽出過程の高速化としては，HOG特徴量の算出に積分画像[6]やGPU[7]を利用する手法が提案されている．中でも，GPUで実装されたfast HOG[8]は，CPUによる特徴抽出と比較して約95倍高速な処理を実現した．識別過程では，AdaBoost[9]をカスケード型に構成し，非検出対象を早期判定することにより高速化する手法[10]が提案されている．しかし，物体検出に多く利用されている線形SVMの高速化に関する検討は従来行われていない．

そこで，本研究では線形SVMによる識別結果の早期判定を導入して識別器の高速化を検討する．提案手法は，検出ウィンドウから抽出した特徴ベクトルに応じて線形SVMを近似計算し，識別結果を早期判定することで高速なラスタスキャンを実現する．このとき，線形SVMの近似計算を高速に求めるために入力特徴ベクトルはバイナリコードである必要があるため，提案手法ではHOG特徴量をバイナリコード化したB-HOG特徴量[11]を用いる．B-HOG特徴量はHOG特徴量と比較して1/8のメモリ量で特徴ベクトルを表現することができるが，HOG特徴量を2値化した際の量子化誤差の影響により，識別精度が低下するという問題がある．そこで，提案手法ではB-HOG特徴量のバイナリコードにビット演算を用いた共起表現を導入して，セル間の関係性を表現することで計算コストを最小限に抑えながら識別性能の高精度化を行う．このように，本研究では識別器の早期判定によるラスタスキャンとB-HOG特徴量の共起表現により，高精度でかつ高速な物体検出を実現する．

2 ラスタスキャンベースの物体検出

物体検出は，局所特徴量と統計的学習手法を用いて学習した識別器を利用する手法が一般的である．Algorithm 1にラスタスキャンによる物体検出アルゴリズム

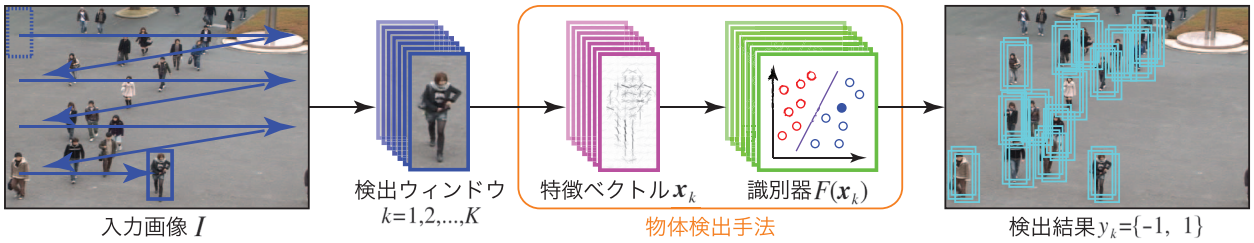

図 1　ラスタスキャンベースの物体検出

Algorithm 1 ラスタスキャンによる物体検出

Require: 入力画像 I

1. 画像 I に対して検出ウィンドウをラスタスキャン

for $k = 1$ to K **do**　//K：検出ウィンドウの総数

2. 検出ウィンドウ $I(k)$ から特徴ベクトル $\boldsymbol{x}_k$ を抽出
3. 識別器 $F(\boldsymbol{x}_k)$ の出力値を算出
4. 閾値 th により対象のラベル y に判定

$$y_k = \begin{cases} 1 & \textbf{if} \quad F(\boldsymbol{x}_k) > th \\ -1 & \textbf{otherwise} \end{cases}$$

end for　//ラスタスキャン終了

return $y_1, y_2, ..., y_K$

を示す．物体検出を実現するには図 1 に示すように，入力画像 I に対して検出ウィンドウを網羅的にラスタスキャンする．そして，ラスタスキャンして得られた全ての検出ウィンドウに対して特徴ベクトルを抽出し，事前に統計的学習手法により構築した識別器を用いて，クラス y_k を検出対象の場合 1，非検出対象の場合-1 として判別する．

人を検出対象としたラスタスキャンベースの物体検出手法には，2005 年に Dalal らが提案した HOG 特徴量と線形 SVM による手法 [2] が多く用いられている．HOG 特徴量を小規模なハードウェアで実装するために，HOG 特徴量を 2 値化した B-HOG 特徴量 [11] が提案されている．本章では，HOG 特徴量，B-HOG 特徴量，線形 SVM[1] により構築した識別器とその問題点ついて述べる．

2.1　HOG 特徴量

Histograms of Oriented Gradients(HOG) 特徴量 [2] は図 2 に示すように，検出ウィンドウからセルと呼ばれる局所領域毎に作成した勾配方向ヒストグラムを特徴量とする．また，複数のセルで構成されるブロック領域毎に特徴量を正規化することで，照明変化や幾何学的変化の影響を受けにくい特徴量となる．

算出手順は，はじめに検出ウィンドウ中の各ピクセルの輝度値 $L(x, y)$ の勾配強度 m と勾配方向 θ を式 (1) ～ 式 (3) より算出する．

$$m(x, y) = \sqrt{L_x(x, y)^2 + L_y(x, y)^2} \tag{1}$$

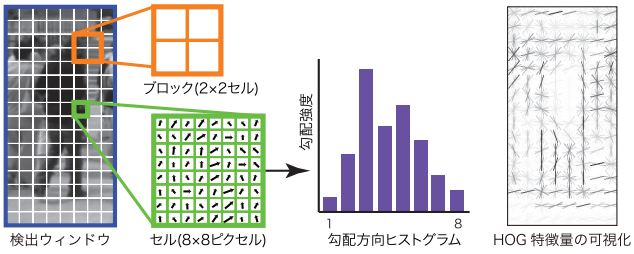

図 2　HOG 特徴量

$$\theta(x, y) = \tan^{-1} \frac{L_y(x, y)}{L_x(x, y)} \tag{2}$$

$$\begin{cases} L_x(x, y) = L(x + 1, y) - L(x - 1, y) \\ L_y(x, y) = L(x, y + 1) - L(x, y - 1) \end{cases} \tag{3}$$

算出した勾配方向 θ は 0° ～ 360° の値で算出されるが，180° より大きくなる方向は 180° 引いて 0° ～ 180° とする．これにより，検出対象と背景領域の輝度の明暗関係に依存しない勾配方向を得ることができる．次に，算出した勾配強度 m と勾配方向 θ を用いて，式 (4) によりセル c ($M \times M$ ピクセル) における勾配方向ヒストグラム $\boldsymbol{V}_c = \{v_c(1), v_c(2), ..., v_c(N)\}$ を作成する．

$$v_c(n) = \sum_{x}^{M} \sum_{y}^{M} m(x, y) \delta[f(\theta(x, y)), n] \tag{4}$$

ここで，$n = 1, 2, ..., N$ はヒストグラムのビンの番号，$f(\theta)$ は勾配方向 θ を N 方向に量子化した値，$\delta[\cdot]$ はクロネッカーのデルタ関数を表しており，二つの要素が等しい場合は 1，それ以外は 0 を出力する関数である．このように，セル c 毎にヒストグラム化することにより，局所領域内の微小な幾何学的変化に対して頑健な特徴量となる．最後に，式 (5) を用いて各セル c で作成した勾配方向ヒストグラム $\boldsymbol{V}_c$ を，複数のセルで構成されるブロック領域 ($R \times R$ セル) ごとに正規化する．

$$v'_c(n) = \frac{v_c(n)}{\sqrt{\sum_{k=1}^{q} v_c(k)^2 + \epsilon}} \qquad (\epsilon = 1) \tag{5}$$

ここで，q はブロック領域内の勾配方向の数 ($R \times R \times N$)，ϵ は分母が 0 になることを防ぐための定数である．ブロッ

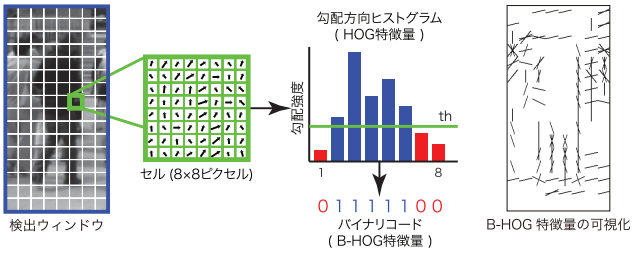

図 3　B-HOG 特徴量

ク領域は，1 セル毎に正規化対象セル領域が重なるように移動しながら正規化する．正規化後の勾配方向ヒストグラム $\boldsymbol{V'}_c$ は，$\boldsymbol{V'}_c = \{v'_c(1), v'_c(2), ..., v'_c(q)\}$ となる．64×128 ピクセルの検出ウィンドウに対して，ブロック領域は $\{(64/M) - (R-1)\} \times \{(128/M) - (R-1)\}$ 個となる．$M = 8$，$R = 2$，$N = 8$ のとき 105 個のブロック領域により正規化処理が行われるため，HOG 特徴量は $2 \times 2 \times 8 \times 105 = 3,360$ 次元の特徴ベクトル $\boldsymbol{x}$ となる．

2.2　B-HOG 特徴量

B-HOG 特徴量 [11] は，図 3 に示すように HOG 特徴量を 2 値化したバイナリコードで物体形状を表現する．HOG 特徴量により作成した勾配方向ヒストグラム $\boldsymbol{V}'_c$ を式 (6) に示すように閾値 th^{bhog} により 2 値化することによりバイナリコード $\boldsymbol{P'}_c = \{p'_c(1), p'_c(2), ..., p'_c(q)\}$ を生成する．

$$p'_c(n) = \begin{cases} 1 & \text{if} \quad v'_c(n) > th^{bhog} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

閾値 th^{bhog} は学習サンプルから得られる HOG 特徴量の各次元毎の平均値を用いる．B-HOG 特徴量は $M = 8$，$R = 2$，$N = 8$ のとき 3,360 ビットのバイナリコードとなり，HOG 特徴量と比べて特徴ベクトルを格納するためのメモリ量を 1/8 に削減することができる．しかし，B-HOG 特徴量は HOG 特徴量と比較して，2 値化の際の量子化誤差の影響により物体検出の識別精度が低下するという問題がある．

2.3　線形 SVM による識別器

Support Vector Machine(SVM)[1] は多くの画像認識に用いられる統計的学習手法であり，物体検出では線形 SVM が多く用いられている．線形 SVM は，式 (7) に示す識別関数により識別器 $F(\boldsymbol{x})$ の出力値を算出する．

$$F(\boldsymbol{x}) = \sum_{j=1}^{J} y_j \alpha_j < \boldsymbol{x}_j^{sv}, \boldsymbol{x} > \tag{7}$$

ここで，J は学習により求めたサポートベクタの数，α はラベル y のサポートベクタに対する重み，$< \boldsymbol{x}^{sv}, \boldsymbol{x} >$ はサポートベクタの特徴ベクトル $\boldsymbol{x}^{sv}$ と入力特徴ベクトル $\boldsymbol{x}$ の内積を表している．ここで，線形 SVM の識別関数 $F(\boldsymbol{x})$ は，入力特徴ベクトル $\boldsymbol{x}$ (D 次元) に対する重みとして，$\boldsymbol{w} = \sum_{j=1}^{J} y_j \alpha_j \boldsymbol{x}_j^{sv}$ を予め計算しておくことで，式 (8) に示すように実数ベクトル間の内積となる．

$$F(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x} \tag{8}$$

$$= \sum_{i=1}^{D} w_i x_i \tag{9}$$

物体検出を行うためには，ラスタスキャンにより得られた検出ウィンドウから抽出した特徴ベクトル $\boldsymbol{x}_k$ に対して，識別器 $F(\boldsymbol{x}_k)$ の出力値を求め，対象のクラス y_k に判定する．

2.4　問題点

VGA サイズ (640×480 ピクセル) の入力画像 I に対して横幅 l_w，縦幅 l_h の検出ウィンドウをスケール l_s，ずらし幅 l_m にてラスタスキャンしたとき，発生する検出ウィンドウの数 K は次式により求めることができる．

$$K = \frac{(640 - l_w \times l_s - 1) \times (480 - l_h \times l_s - 1)}{l_m^2} \tag{10}$$

例えば，検出ウィンドウのサイズ $(l_w, l_h) = (64, 128)$ をマルチスケール $l_s = 1.0, 2.0$ でずらし幅 $l_m = 4$ としてラスタスキャンした場合，検出ウィンドウは約 2 万個となる．これら全ての検出ウィンドウから特徴抽出を行い，識別器 $F(\boldsymbol{x}_k)$ の出力値を求める必要があり，物体検出をリアルタイムに実現するためには，特徴抽出過程と識別過程をそれぞれ高速化する必要がある．特徴抽出過程の高速化としては，GPU を用いた HOG 特徴量の高速化 [8] が提案されている．一方，線形 SVM により構築した識別器の高速化に関する検討は，従来行われていない．ラスタスキャンベースの物体検出では，膨大な数の検出ウィンドウを処理する際に線形 SVM の実数ベクトル間の内積にかかる計算コストが高いという問題がある．そのため，識別過程である線形 SVM の高速化が課題となっている．

3　提案手法

本研究では，ラスタスキャンの高速化として線形 SVM に近似計算による早期判定を導入する．また，線形 SVM の近似計算に基づく高速化に必要となる入力特徴ベクトルのバイナリ化に伴う性能低下を解決するために，HOG 特徴量をバイナリコード化した B-HOG 特徴量の共起性を表現することで，物体検出の高精度化を行う．以下に提案手法によるラスタスキャンの高速化とバイナリコードにおける共起表現について述べる．

3.1 線形 SVM の近似計算

線形 SVM により学習した識別器 $F(\boldsymbol{x})$ の計算は，式 (8) に示すように特徴ベクトル $\boldsymbol{x}$ と重み $\boldsymbol{w}$ の内積であり，膨大な数の検出ウィンドウを処理すると時間を要するという問題がある．そこで，本研究では Hare らによって提案された近似計算法 [12] を利用して，実数ベクトル間の内積をバイナリコード間の内積に置き換えることで，線形 SVM の高速な識別演算を実現する．

線形 SVM の近似計算法では，重みベクトル $\boldsymbol{w}$ を実数ベクトル $\boldsymbol{\beta}$ と基底バイナリコード $\boldsymbol{b} \in \{-1,1\}^D$ に分解する．Algorithm 2 に重みベクトル $\boldsymbol{w}$ の分解アルゴリズムを示す．

Algorithm 2 重みベクトル $\boldsymbol{w}$ の分解

Require: 重みベクトル $\boldsymbol{w}$，基底数 N_b

1. 初期化：線形 SVM の重みベクトル $\boldsymbol{w}$ を実数ベクトル $\boldsymbol{r}$ に格納
$$\boldsymbol{r} = \boldsymbol{w}$$
2. $\boldsymbol{w}$ を実数ベクトル $\boldsymbol{\beta}$ と基底バイナリコード $\boldsymbol{b}$ に分解

for $i = 1$ to N_b **do** //N_b：基底数

2.1 実数ベクトル $\boldsymbol{r}$ を {-1,1} に 2 値化して基底バイナリコード $\boldsymbol{b}_i$ を算出
$$\boldsymbol{b}_i = \mathrm{sign}(\boldsymbol{r})$$
2.2 実数ベクトル $\boldsymbol{r}$ と基底バイナリコード $\boldsymbol{b}_i$ の内積
$$\beta_i = \frac{< \boldsymbol{r}, \boldsymbol{b}_i >}{||\boldsymbol{b}_i||^2}$$
2.3 近似による残差を実数ベクトル $\boldsymbol{r}$ に代入
$$\boldsymbol{r} \leftarrow \boldsymbol{r} - \beta_i \boldsymbol{b}_i$$

end for

return $\{\beta_i\}_{i=1}^{N_b}$，$\{\boldsymbol{b}_i\}_{i=1}^{N_b}$

線形 SVM による識別器 $F(\boldsymbol{x})$ は，重みベクトル $\boldsymbol{w}$ を分解して求めた実数ベクトル $\boldsymbol{\beta}$ と基底バイナリコード $\boldsymbol{b}$ を用いることで，$F(\boldsymbol{x}) \approx f(\boldsymbol{x}) = \sum_{i=1}^{N_b} \beta_i \boldsymbol{b}_i^{\mathrm{T}} \boldsymbol{x}$ と近似計算することができる．ここで，基底バイナリコード $\boldsymbol{b}$ を $\boldsymbol{b}^+ \in \{0,1\}^D$ と $\bar{\boldsymbol{b}}^+$ に分解 $(\boldsymbol{b} = \boldsymbol{b}^+ - \bar{\boldsymbol{b}}^+)$ することで，式 (11) に示すように線形 SVM の近似計算を内積 $< \boldsymbol{b}_i^+, \boldsymbol{x} >$ とノルム $|\boldsymbol{x}|$ により計算することができる．

$$\begin{aligned} f(\boldsymbol{x}) &= \sum_{i=1}^{N_b} \beta_i \boldsymbol{b}_i^{\mathrm{T}} \boldsymbol{x} \\ &= \sum_{i=1}^{N_b} \beta_i (< \boldsymbol{b}_i^+, \boldsymbol{x} > - < \bar{\boldsymbol{b}}_i^+, \boldsymbol{x} >) \\ &= \sum_{i=1}^{N_b} \beta_i (2 < \boldsymbol{b}_i^+, \boldsymbol{x} > - |\boldsymbol{x}|) \end{aligned} \tag{11}$$

ここで，入力特徴ベクトル $\boldsymbol{x}$ が B-HOG 特徴量のようなバイナリコード $\boldsymbol{x} \in \{0,1\}^D$ である場合，内積 $< \boldsymbol{b}_i^+, \boldsymbol{x} >$ とノルム $|\boldsymbol{x}|$ は，それぞれ論理積とビットカウントからなるビット演算で計算することが可能である．このような，バイナリコード間の論理積は実数間の積に比べて高速に演算することができる．また，ビットカウントは SSE4.2 から CPU に直接実装されている POPCNT 関数を用いることで高速な演算が可能である．

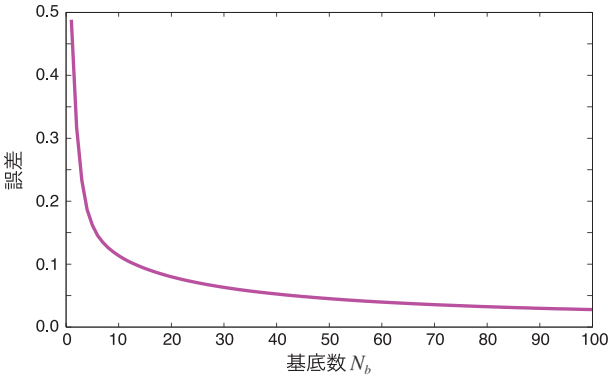

図 4　基底数 N_b による $\boldsymbol{w}$ の近似精度

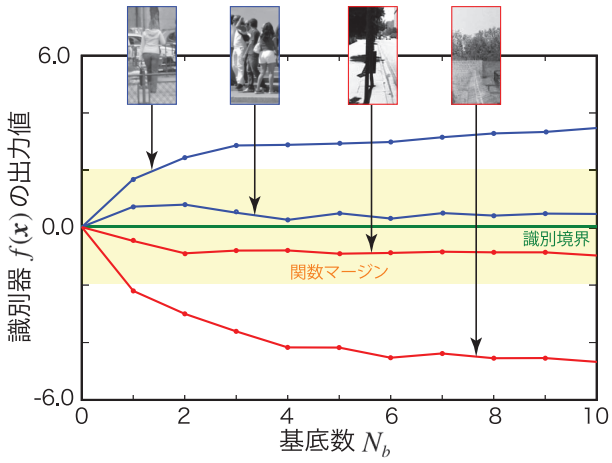

図 5　線形 SVM の近似計算結果

3.2 基底数 N_b と近似計算結果

我々は，線形 SVM の近似計算過程における基底数 N_b と近似計算結果に着目する．図 4 は，重みベクトル $\boldsymbol{w}$ と実数ベクトル $\boldsymbol{\beta}$ と及び基底バイナリコード $\boldsymbol{b}$ とのユークリッド距離により，近似計算の基底数 N_b と誤差の関係をグラフで表したものである．重みベクトル $\boldsymbol{w}$ の近似精度は，基底数 N_b が増加するにつれて高くなっていることがわかる．線形 SVM の識別精度を維持するためには，基底数 N_b を多く用いて識別器を近似計算する必要がある．その反面，高速なラスタスキャンを行うためには基底数 N_b をできるだけ削減して，識別器を近似計算することが求められる．

図 5 は，データセットのサンプルを例に基底数 N_b による線形 SVM の近似計算結果をグラフ化したものである．図 5 より，線形 SVM の近似計算では基底数 N_b が少ない段階では，おおまかに識別器の出力値を算出している．基底数 N_b が多くなるにつれて，線形 SVM

との誤差を埋めるように識別器の出力値が変化している．また，図5の2つのサンプルは基底数が少ない段階で，線形SVMの関数マージンから離れた出力値となっている．精度の良い物体検出を行うためには，この関数マージン内で2つのクラス y を最も多く分るような識別器の閾値 th を設定する．そのため，線形SVMの近似計算時に，少ない基底数 N_b で関数マージンから離れた出力値を持つサンプルは，基底数 N_b を増加しても閾値 th による識別結果が変わることはないといえる．

3.3 近似計算結果の早期判定によるラスタスキャンの高速化

基底数 N_b による識別精度と速度はトレードオフの関係にあり，膨大な検出ウィンドウを高速に処理するためには基底数 N_b をできるだけ削減したい．そこで，提案手法では関数マージンを利用して近似計算過程における計算結果から早期判定を導入することで，識別器の性能を維持したままラスタスキャンの高速化を行う．Algorithm 3 に近似計算結果の早期判定によるラスタスキャンの高速化アルゴリズムを示す．

Algorithm 3 ラスタスキャンの高速化

Require: 入力画像 I

1. 画像 I に対して検出ウィンドウをラスタスキャン

for $k = 1$ to K **do** //K：検出ウィンドウの総数

2. 検出ウィンドウ $I(k)$ からバイナリコードである特徴ベクトル $\boldsymbol{x}_k$ を抽出
3. 初期化 $f(\boldsymbol{x}_k) \leftarrow 0$
4. 近似計算による識別器の出力値を算出

for $i = 1$ to N_b **do** //N_b：最大基底数

4.1 線形SVMの近似計算：$\beta_i(2 < \boldsymbol{b}_i^+, \boldsymbol{x}_k > -|\boldsymbol{x}_k|)$

$f(\boldsymbol{x}_k)$ += $\beta_i(2\,\mathrm{POPCNT}(\boldsymbol{b}_i^+ \,\&\, \boldsymbol{x}_k) - \mathrm{POPCNT}(\boldsymbol{x}_k))$

4.2 計算結果を判定

if $f(\boldsymbol{x}_k) > th^{pos}$ **or** $f(\boldsymbol{x}_k) < th^{neg}$ **then**

break //近似計算の打ち切り

end if

end for //N_b まで近似計算

5. 閾値 th により対象のラベル y に判定

$$y_k = \begin{cases} 1 & \textbf{if} \quad f(\boldsymbol{x}_k) > th \\ -1 & \textbf{otherwise} \end{cases}$$

end for //ラスタスキャン終了

return $y_1, y_2, ..., y_K$

ここで，th^{pos} 及び th^{neg} はそれぞれ検出対象クラス $y = 1$ と非検出対象クラス $y = -1$ のサポートベクタ (識別境界 ±1.0) を示しており，$th^{pos} < f(\boldsymbol{x}) < th^{neg}$ は関数マージン内の出力値であることを表す．図6に示すように，正と負の方向に大きな値を出力した場合に線形SVMの近似計算を打ち切り，識別結果を早期判定することが可能となる．関数マージン内の識別が困難な入力サンプルは，多くの基底数 N_b を用いて線形SVM

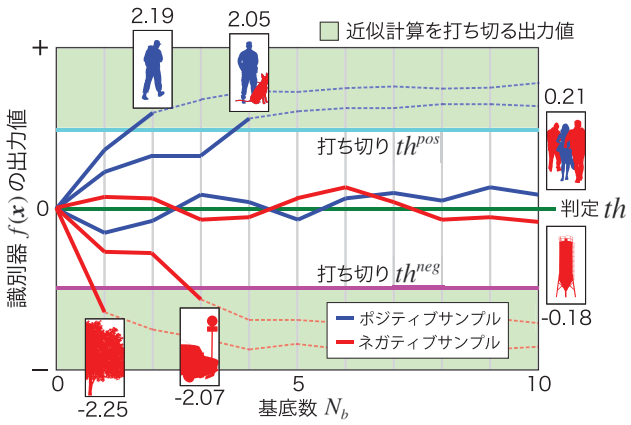

図6　線形SVMの近似計算の打ち切り

との誤差の少ない近似計算を行い判定をする．これにより，線形SVMの識別精度を維持しつつ，検出ウィンドウによっては早期判定することで高速なラスタスキャンが可能となる．

3.4 バイナリコードにおける共起表現

本研究では，3.1節で述べたようにビット演算を利用して近似計算を高速化するために，HOG特徴量をバイナリコード化したB-HOG特徴量を利用する．B-HOG特徴量はHOG特徴量と比較して，2値化の際の量子化誤差の影響により物体検出の識別精度が低下する問題がある．そこで，提案手法ではB-HOG特徴量間のビット演算により共起性を表現することで，計算コストをそれほど増加することなくバイナリコードの特徴ベクトルによる識別精度の高精度化を行う．

提案手法によるバイナリコードの共起表現方法は，ブロックに属する2つのセルのB-HOG特徴量 $\boldsymbol{P}_{c_1}$ と $\boldsymbol{P}_{c_2}$ から式(12)～式(14)に示すビット演算により勾配方向ヒストグラムにおけるバイナリコード間の関係性を表現する．ここで，論理演算子には論理積(AND)，論理和(OR)，排他的論理和(XOR)のいずれかを用いる．

$$\boldsymbol{P}^{and}_{c_1,c_2} = \boldsymbol{P}_{c_1} \,\&\, \boldsymbol{P}_{c_2} \tag{12}$$

$$\boldsymbol{P}^{or}_{c_1,c_2} = \boldsymbol{P}_{c_1} \mid \boldsymbol{P}_{c_2} \tag{13}$$

$$\boldsymbol{P}^{xor}_{c_1,c_2} = \boldsymbol{P}_{c_1} \hat{} \, \boldsymbol{P}_{c_2} \tag{14}$$

セルの組み合わせにより，$R = 2$ のときには6パターンの共起バイナリコード $\boldsymbol{P}^{operator} = \{\boldsymbol{P}_{c_1,c_2}, \boldsymbol{P}_{c_1,c_3}, \boldsymbol{P}_{c_1,c_4}, \boldsymbol{P}_{c_2,c_3}, \boldsymbol{P}_{c_2,c_4}, \boldsymbol{P}_{c_3,c_4}\}$ が生成される．このとき，論理演算子がANDの場合，2つのセルに共通して勾配が存在するときを“1”のビットで表現する．ORの場合，2つのセルでどちらかに勾配があるときを“1”のビットで表現し，2つのセルに共通して勾配が存在しないときを“0”のビットで表現する．XORの場合，2つのセルに共通しない勾配を“1”のビットで表現し，2つのセルに共通して勾配が存在する及び存在しないと

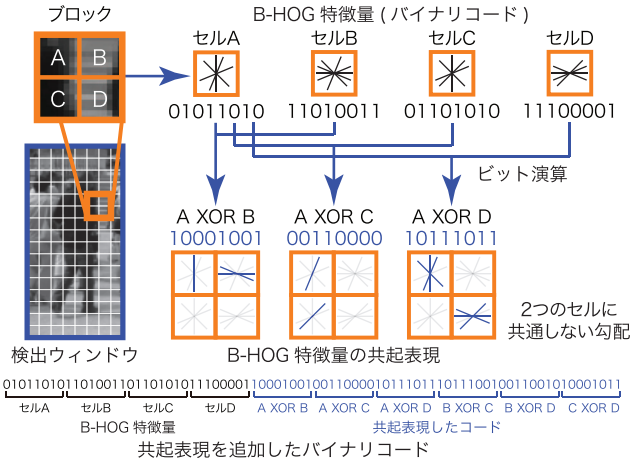

図 7　B-HOG 特徴量の共起表現

きを“0”のビットで表現する．このように，ビット演算を利用して共起表現することでセル間の勾配の関係性を捉えることが可能となり，バイナリコードによる物体検出の高精度化が期待できる．バイナリコード間の共起には，ビット演算を用いるため低コストで共起表現をすることができる．

提案手法は図 7 に示すように，ブロック領域内のセルを組み合わせて共起表現したバイナリコードを B-HOG 特徴量のバイナリコードに追加する．共起表現を追加した特徴量は，$M = 8$，$R = 2$，$N = 8$ のとき 3,360 ビットの B-HOG 特徴量に 5,040 ビットのバイナリコードを追加するため，8,400 ビットのバイナリコードとなる．提案手法では，この B-HOG 特徴量を共起表現した 8,400 ビットのバイナリーコードを入力特徴ベクトル $\boldsymbol{x}$ として用いる．

4　評価実験

提案手法の有効性を評価するために評価実験により識別精度及び識別過程の処理時間の比較を行う．

4.1　データセット

評価実験には，人検出のベンチマークとして広く利用されている INRIA Person Dataset[2] を使用する．学習サンプルには，ポジティブサンプルを 2,416 枚，ネガティブサンプルを 13,161 枚を使用する．識別精度を評価するためのテストサンプルには，ポジティブサンプルを 1,126 枚，ネガティブサンプルは背景画像 453 枚から網羅的にラスタスキャンさせて切り出した画像 1,306,029 枚を使用する．学習サンプル，テストサンプルの画像サイズはそれぞれ 64×128 ピクセルの大きさに正規化して使用する．また，識別器の処理時間を評価するための入力画像には，検出対象と非検出対象を含む INRIA Person Dataset の評価用画像を 433 枚使用する．

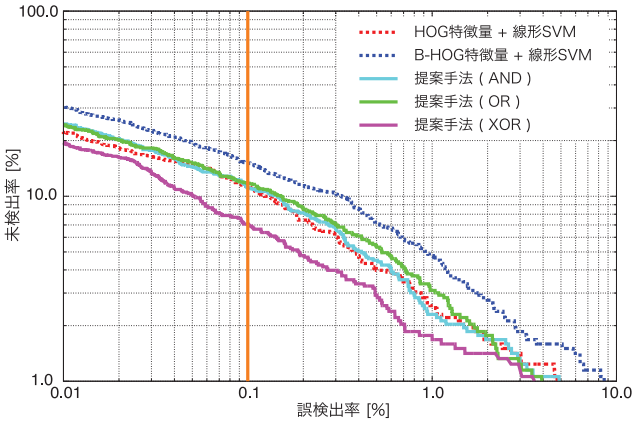

図 8　DET カーブ

4.2　実験概要

提案手法の有効性を評価ために，以下の特徴量と識別器の組み合わせについて識別精度を比較する．

- HOG 特徴量+線形 SVM
- B-HOG 特徴量+線形 SVM
- 提案手法 (“論理演算子”)
  B-HOG 特徴量の共起表現と線形 SVM の近似計算による早期判定

提案手法は，B-HOG 特徴量を共起表現するための論理演算子に，AND，OR，XOR のそれぞれを評価して比較する．評価には Detection Error Trade-off(DET) カーブを用いる．DET カーブは横軸に誤検出率，縦軸に未検出率を表しており，原点に近いほど高精度であることを表す．処理時間の比較では，OS:Windows Server 2008 Enterprise x64，CPU:Intel Xeon CPU X7542 @ 2.67GHz，RAM:256GB の PC を用いる．全ての実験において線形 SVM は SVM Light[13] を用いて学習を行う．各特徴量のパラメータは $M = 8$，$R = 2$，$N = 8$ とする．また，評価実験では予備実験の結果より，線形 SVM の近似計算における基底数 $N_b = 16$ とする．

4.3　識別精度の評価

実験結果の DET カーブを図 8 に示す．DET カーブより，線形 SVM を用いた B-HOG 特徴量は HOG 特徴量と比較して識別精度が低下している．一方，提案手法 (AND)，提案手法 (OR) は B-HOG 特徴量と同様のバイナリコードの特徴量であるが，HOG と同等の識別率を得た．さらに，提案手法 (XOR) は誤検出率 0.1% において HOG 特徴量+線形 SVM と比較して約 6.1% 向上させることができた．

XOR による B-HOG 特徴量の共起表現の有効性を確認する．図 9 に B-HOG 特徴量で未検出のテストサンプルに対して，共起表現により検出可能となった全て

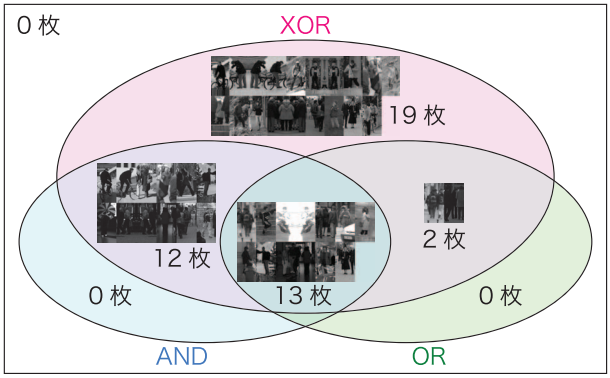

図 9　検出可能になったポジティブサンプル

表 1　各識別器の検出率 [%] と処理時間 [ms]

識別器	検出率 [%]	処理時間 [ms]
線形 SVM	94.16	0.034
近似計算法 (N_b=16)	94.08	0.013
近似計算法 (N_b=2)	91.07	0.002
提案手法	94.08	0.002

のポジティブサンプルを示す．XOR で共起表現した場合，AND と OR で検出可能となった 27 枚と，AND と OR では検出することが不可能な 19 枚のサンプルを検出できたことがわかる．これは，XOR により生成したバイナリコードにおいて，“0”のビットで AND と OR と同じ効果を持つ特徴を表現し，“1”のビットで物体検出に効果的な特徴を表現をすることができたためと考えられる．

4.4　処理速度の評価

表 1 に，XOR を用いて共起表現した特徴ベクトルを用いたときの検出率と 1 つの検出ウィンドウの識別に要する処理時間を示す．近似計算法は基底数 $N_b = 16$ のとき，線形 SVM と同等の識別精度である．処理時間を比較すると，近似計算法は線形 SVM よりも約 3 倍高速な演算が可能である．提案手法は早期判定を行うため近似計算法よりもさらに速く，線形 SVM と 比較して約 17 倍高速な識別器の演算を実現した．このとき，提案手法は平均でポジティブサンプルは 7.78 個，ネガティブサンプルは 1.46 個の基底数による近似計算結果から早期判定している．基底数 $N_b = 2$ のときの近似計算法は，提案手法と同じ処理時間となるが検出率は低下する．VGA サイズ (640×480 ピクセル) の入力画像に対して検出ウィンドウのサイズ $(l_w, l_h) = (64, 128)$ をマルチスケール $l_s = 1.0, 2.0$ でずらし幅 $l_m = 4$ としてラスタスキャンした場合，全ての検出ウィンドウの識別処理に要する時間は，線形 SVM では 674.2[ms](1.48[fps]) なのに対して，提案手法は 39.66[ms](25.21[fps]) と高速にラスタスキャンをすることが可能となる．

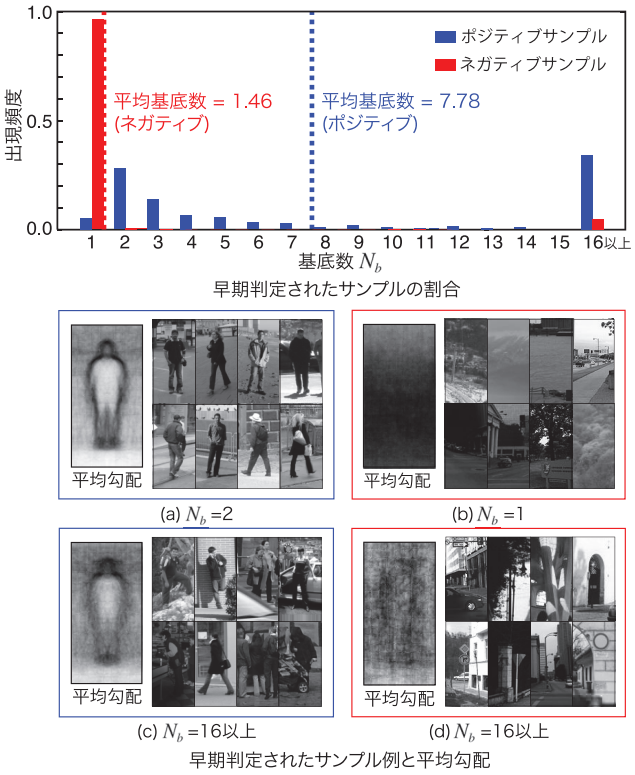

図 10　識別結果が早期判定されたサンプル

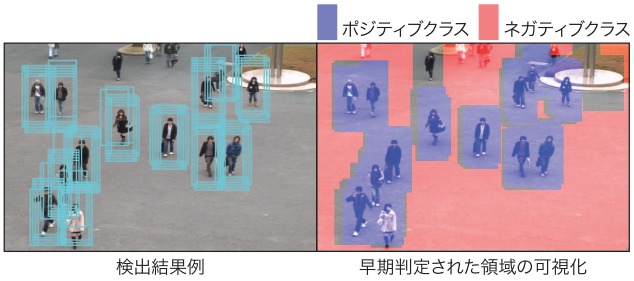

図 11　検出結果例と早期判定された領域の可視化

図 10 に提案手法により識別結果が早期判定されたサンプルの割合とそのサンプル例を示す．基底数が少ない段階では，早期判定されたサンプルの平均勾配画像 (a)，(b) から確認できるように，人と背景に分離しやすいサンプルであることがわかる．一方，困難なサンプル (c)，(d) は最大基底数まで ($N_b = 16$) 継続して近似計算が行われる．また，ネガティブサンプルの 9 割以上は基底数が 1 個で早期判定することで，高速なラスタスキャンが可能となる．一般的な物体検出シーン (入力画像) では，多くの領域が背景 (非検出対象) であるため，図 11 に示すように提案手法は多くの検出ウィンドウにて少ない基底数 N_b で線形 SVM を近似計算していることがわかる．以上より，提案手法は検出ウィンドウに応じて適応的に識別結果を早期判定することで，線形 SVM の性能を維持しつつ高速なラスタスキャンを実現した．

表 2　各特徴量のメモリ [Byte] と検出率 [%]，処理時間 [ms]

特徴量	次元数	メモリ [Byte]	検出率 [%]	処理時間 [ms]
HOG 特徴量	3360	3360	88.69	1.554
B-HOG 特徴量	3360	420	84.89	1.561
提案手法 (XOR)	8400	1050	94.08	1.576
Random Projection	3360	420	86.57	50.989
	8400	1050	87.63	128.059
	23520	2940	88.69	360.219

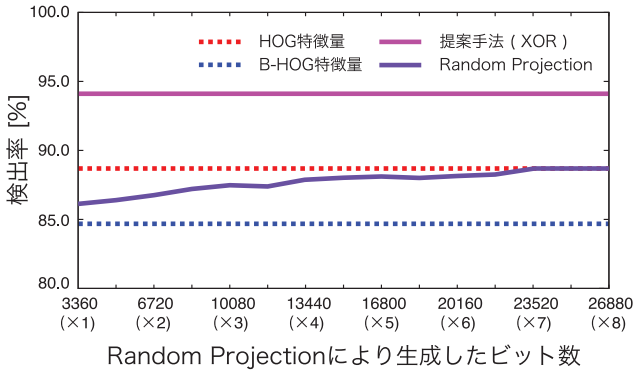

図 12　Random Projection の識別精度

4.5　特徴量のメモリサイズと処理時間の評価

提案手法のバイナリコードによる特徴表現に必要なメモリ量を調査する．ここでは，特徴ベクトルの距離関係を保持したままバイナリコード化することのできる Random Projection[14] を用いて，提案手法の共起表現した特徴量のメモリサイズを評価する．図 12 に，Random Projection により HOG 特徴量を任意のビット数のバイナリコードで空間写像した際の識別精度を示す．Random Projection では，実数ベクトルの HOG 特徴量と同等の識別精度を再現するには 23,520 ビット必要である．一方，表 2 に示すように，提案手法 (XOR) は検出率を約 6.1% 向上させつつそのビット数は 8,400 ビットであり，効率の良いバイナリ表現ができているといえる．さらに，提案手法 (XOR) の共起抽出に要する時間は 1.576[ms]，HOG 特徴量は 1.554[ms] とわずかな時間が増えるだけで特徴抽出を行うことができた．

5　おわりに

本稿では，物体検出における識別過程の高速化について取り組み，本研究の貢献は以下の 2 点である．1. 線形 SVM に早期判定を導入することによるラスタスキャンの高速化．2. B-HOG 特徴量を共起表現したバイナリコードの特徴量による物体検出の高精度化．1. では，線形 SVM を近似計算し，検出ウィンドウに応じて適応的に識別結果を早期判定することで，識別過程において識別精度を維持しつつ約 17 倍高速なラスタスキャンを実現した．2. では，B-HOG 特徴量のバイナリコードに対してビット演算を用いた共起表現により高速かつ効率の良いバイナリコード化を行い，メモリサイズを約 1/3 にしつつ識別精度の高精度化を実現した．今後は，GPU を用いた fast HOG[8] を提案手法に導入して特徴抽出過程を高速化することで，物体検出手法全体において高速なラスタスキャンを実現する予定である．

参考文献

[1] C.Cortes and V.Vladimir, “Support-Vector Networks”, Machine Learning, vol.20, no.3, pp.273–297, 1995.

[2] N. Dalal, and B. Triggs, “Histograms of Oriented Gradients for Human Detection”, Computer Vision and Pattern Recognition, vol.1, pp.886–893, 2005.

[3] X.Wang, T. X. Han and S. yan, “An HOG-LBP human detector with partial occlusion handling”, International Conference on Computer Vision, pp.32–39, 2009.

[4] C.Wojek, S. Walk, S. Roth and B. Schiele, “Monocular 3D scene understanding with explicit occlusion reasoning”, Computer Vision and Pattern Recognition, pp.1993–2000, 2011.

[5] P.F.Felzenszwalb, R.B. Girshick, D.McAllester, and D.Ramanan, “Object Detection with Discriminatively Trained Part Based Models”, Pattern Analysis and Machine Intelligence, vol.32, no.9, pp.1627–1654, 2010.

[6] Q.Zhu, S.Avidan, M.Yeh and K.Cheng, “Fast Human Detection Using a Cascade of Histograms of Oriented Gradients”, Computer Vision and Pattern Recognition, pp.1491–1498, 2006.

[7] V.Prisacariu and I.Reid, “Fast Human Detection with Cascaded Ensembles on the GPU”, Intelligent Vehicles Symposium, pp.325–332, 2010.

[8] V.Prisacariu and I.Reid, “fastHOG - a real-time GPU implementation of HOG”, Departmentof Engineering Science, no.2310/9, 2009.

[9] Y. Freund, and R. E. Schapire, “Experiments with a new boosting algorithm”, International Conference on Machine Learning, pp.148–156, 1996.

[10] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features”, Computer Vision and Pattern Recognition, vol.1, pp.511–518, 2001.

[11] 松島千佳, 山内悠嗣, 山下隆義, 藤吉弘亘, “物体検出のための Relational HOG 特徴量とワイルドカードを用いたバイナリーのマスキング”, 電子情報通信学会論文誌 D, vol.J94-D, no.8, pp.1172–1182, 2011.

[12] S. Hare, A. Saffari and P. H. S. Torr, “Efficient online structured output learning for keypoint-based object tracking”, Computer Vision and Pattern Recognition, pp.1894–1901, 2012.

[13] T. Joachims, SVM light, http://svmlight.joachims.org

[14] M.X.Goemans, “Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming”, Journal of the ACM, vol.42, pp.1115–1145, 1995.