一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
PRMU2015-104(2015-12)

対応点探索のための特徴量表現

安倍　　満†　　長谷川　昂宏††　　藤吉　弘亘††

† 株式会社デンソーアイティーラボラトリ
〒 150–0002 東京都渋谷区渋谷二丁目 15 番 1 号 渋谷クロスタワー 28 階
†† 中部大学　〒 487–8501 愛知県春日井市松本町 1200
E-mail: †manbai@d-itlab.co.jp, ††{tkhr@vision., hf@}cs.chubu.ac.jp

あらまし　対応点探索とは，ある対象を異なる視点から撮影して得られた複数の画像から，物理的に同一の個所（対応点）見つける処理のことを指す．本稿では，過去に提案されてきた手法を，**(1) 実数ベクトルで局所特徴量を表現する方法，(2) 二値特徴量で局所特徴量を表現する方法，(3) 深層学習による方法**，以上３つの分類の観点からまとめる．(1) については，過去に多くの解説論文や発表資料が公開されているため，本稿では特に (2) の二値特徴量による特徴量表現について詳しく解説する．これは，局所特徴量を実数のベクトルで表現せず，代わりに数十〜数百個程度の 0 と 1 の列（すなわち二値のベクトル）で表現するというものである．この方法により，計算速度・メモリ消費量の問題が飛躍的に改善されたことから，コンピュータビジョンの研究者の幅広い関心を集めるようになった．今では，SIFT や SURF に代わる新たな局所特徴量として急速に普及しつつある．これに加え，今後のトレンドとなると思われる (3) 深層学習による手法についても，代表的な手法をいくつか紹介する．また，研究目的に利用可能なソフトウェア，データセット，および実装方法についてのノウハウについても併せて紹介する．
キーワード　二値特徴量，深層学習，対応点探索

Local feature description for keypoint matching

Mitsuru AMBAI†, Takahiro HASEGAWA††, and Hironobu FUJIYOSHI††

† Denso IT Laboratory, Inc.
Shibuya CROSS TOWER 28th Floor 2–15–1 Shibuya Shibuya-ku Tokyo, 150–0002 Japan
†† Chubu University　1200, Matsumoto, Kasugai, Aichi, 487–8501 Japan
E-mail: †manbai@d-itlab.co.jp, ††{tkhr@vision., hf@}cs.chubu.ac.jp

Abstract　A task of finding physically the sample points among multiple images captured from different viewpoints is called as key point matching for which discriminative local feature representation is very important. We categorize proposed methods in the past into three groups: (1) real-valued feature representation, (2) binary feature representation and (3) feature representation by deep learning, respectively. In this paper, we briefly overview the classic real-valued feature representation and especially focus on investigating the recent binary feature representation. In addition, we introduce a new trend that utilizes deep learning for nonlinear feature representation. Open source software, dataset and implementation techniques are also reviewed.
Key words　Binary features, deep learning, keypoint matching

1. はじめに

局所特徴量は，画像間のマッチングを可能とする強力なツールであり，物体認識，三次元復元，画像検索など，様々なアプリケーションの飛躍的な進歩に貢献してきた．局所特徴量は，キーポイント検出と特徴量記述という二段階の処理からなる．キーポイント検出と特徴量記述の両方を，幾何学的変化（回転，スケール，アフィン変形）や照明変化に対して頑健になるように設計する点がポイントであり，Scale-Invariant Feature Transform (SIFT) [20] をはじめとして，数多くの局所特徴量が提案されてきた．本稿では，キーポイント検出については扱わず，特徴量の表現方法について詳しく紹介する．

近年の動向を見渡すと，対応点探索のための特徴量表現方法は，大きく分けて次の３つに分類できるようである．

（１） 実数ベクトルで局所特徴量を表現する方法 (1999〜)
（２） 二値ベクトルで局所特徴量を表現する方法 (2010〜)

（3） 深層学習による方法 (2015〜)

SIFT が提案されてから 10 年ほど経った頃に大きな動きがあり，特徴量表現に「二値特徴量 (注1)」を用いるという考え方が広く浸透した．これは，特徴量の記述に必要なデータサイズをできるだけ小さくすることで，局所特徴量の消費メモリ量を減らし，かつ局所特徴量間のマッチング速度を高速化しようと試みるものである．二値特徴量は処理負荷が軽く，モバイル端末などでもリアルタイム処理が可能であることから，SIFT に代わる手法として定着した．その後，一般物体認識における識別タスクにおいて，深層学習が比類なき成功を収めた影響を受け，対応点探索においても Convolutional Neural Network を用いる方法が提案されるようになった．

「(1) 実数ベクトルで局所特徴量を表現する方法」については，これまでに数多くの解説論文や発表資料が公開されている．そのため，本稿では基本的な考え方と残課題についてのみ解説し，詳細は関連文献に譲る．「(2) 二値ベクトルで局所特徴量を表現する方法」については，2010〜2015 年の動向を体系的にまとめ，代表的な手法について詳しく解説する．「(3) 深層学習による方法」については，本原稿執筆時点で公開されていない論文 (注2) も多く，動向を体系的に論ずるのは難しいが，現時点で公開されている代表的な文献を中心に解説を試みる．

1.1 本論文の構成

2. 節では，実数ベクトルで局所特徴量を表現する方法について記載する．先にも述べた通り，この種の手法に関する数多くの解説論文や発表資料が公開されているため，ここでは基本的な考え方や代表的な手法，これらの課題についてのみ解説し，詳細は関連文献に譲る．

3.〜6. 節では，二値ベクトルで局所特徴量を表現する方法について記載する．後で詳しく述べるが，この種の手法はパッチから二値特徴量を直接生成する方法と，実数の局所特徴量を得てからこれを二値に変換する方法に大別できる．これらをそれぞれ 4. 節，5. 節で解説する．6. 節では，4. 節および 5. 節で解説した手法について，体系的にまとめる．

7. 節では，最新のアプローチである深層学習による方法を紹介し，今後の方向性について考察する．

8. 節では，前節までに紹介した手法を利用するためのソフトウェアライブラリ，実装方法，研究を進めるうえで有用な評価ツールやデータセットについて紹介する．

なお本稿は，2013 年 11 月の CVIM 研究会における発表原稿 [40] を発展させたものである．また執筆にあたり，著者らによる精密工学会の解説記事 [42] を参考としている．まとめるにあたり，数学的な記法の整合性をとるため，原著論文で用いられている記法を部分的に変更した．ただし，あまり変え過ぎると原著論文を参考する際に混乱するため，変更は必要最小限にとどめた．

(注1)：英語では，Binary Features, Binary Descriptors, Binary Codes などの表記がこれに対応する．

(注2)：本原稿執筆時は ICCV2015 開催前であり，論文のタイトルのみが参照できるが，そのリストを見る限り，深層学習を用いた対応点探索に関する論文が幾つか発表されるようである．

2. 実数ベクトルで局所特徴量を表現する方法

まず初めに，局所特徴量による対応点探索の一般的な処理の流れについて，SIFT を例にとって簡単に紹介しておく（図 1）．SIFT では，はじめに Difference of Gaussian(DoG) 画像を作成することでスケールに不変なキーポイントを検出する．次にそれぞれのキーポイントに対してオリエンテーション（回転方向）と輝度変化に不変な 128 次元の局所特徴量を抽出し，これを画像間で比較することで対応点を得る．SIFT の詳細については文献 [41] による解説が詳しい．

SIFT 特徴量は，対応点探索の目的の他に，一般物体認識における特徴量表現手法としても活用され，この分野の発展に多大なる影響を与えた．コンピュータビジョンの研究者から圧倒的な支持を得た SIFT は，数多くの研究者によって改良が試みられている．例えば，特徴量の記述性能の改善を試みた GLOH [21] や PCA-SIFT [18]，特徴抽出の高速化を達成した SURF [7]，特徴量自体に回転不変性を持たせた RIFF [33] などが提案されている．また，パッチを無数にアフィン変換させ，総当たりでマッチングすることでアフィン不変な対応点探索を実施する ASIFT [22] などもよく知られた手法である．2010 年 6 月に刊行されたコンピュータビジョン最先端ガイド第二巻 [41] の第一章では，SIFT [20]，SURF [7]，PCA-SIFT [18]，GLOH [21] 等の代表的な局所特徴量について体系的に解説されているので参考にされたい．

2.1 残された課題

二画像間の対応点探索手法としては，SIFT や SURF は十分に優れたものであった．しかしながら，大規模な画像検索 [23], [25] やリアルタイム処理への応用が検討されるにつれ，処理速度やメモリ消費量に関する課題が指摘されるようになった．

（1） メモリ消費量

例えば SIFT の場合，局所特徴量は 128 次元の実数ベクトルで表現される．これを適切にスカラー量子化し，最も小さなデータ型である unsigned char で格納したとしても，1 つのキーポイントあたり 128byte を消費する．これがキーポイントの数だけ必要であるから，仮に画像から 1,000 点のキーポイントが検出されたとすると，$1,000 \times 128\text{byte} \approx 125\text{Kbyte}$ のメモリを消費することになる．扱う画像枚数が増えれば増えるほど，メモリ消費量の問題は深刻となる．

（2） 距離計算の速度

2 つの特徴量 $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^D$ の距離は，次の (1)(2) 式に示すユークリッド距離 d_{E} やベクトル間角度 d_θ で表せる．

$$d_{\mathrm{E}}(\mathbf{x}_1, \mathbf{x}_2) = \sqrt{(\mathbf{x}_1 - \mathbf{x}_2)^\top (\mathbf{x}_1 - \mathbf{x}_2)} \tag{1}$$

$$d_\theta(\mathbf{x}_1, \mathbf{x}_2) = \arccos\left(\frac{\mathbf{x}_1^\top \mathbf{x}_2}{|\mathbf{x}_1||\mathbf{x}_2|}\right) \tag{2}$$

例えばユークリッド距離の場合，D 回の引き算，D 回の掛け算，$D-1$ 回の足し算，および 1 回の平方根の演算が必要である．SIFT をはじめとする多くの局所特徴量は次元数 D が大きいため，距離計算の負荷が高くなりやすい．

近年では，大規模な画像検索などの応用において数百〜数千万

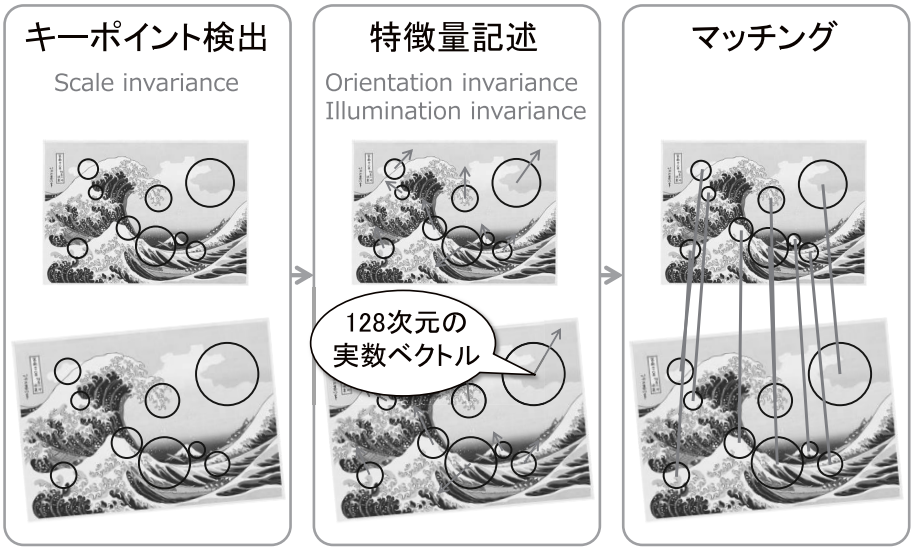


図 1　SIFT による対応点探索

個の局所特徴量をメモリ上に格納して類似度計算を行うことも珍しく無い．そのため，メモリ消費量と距離計算の速度の 2 点における改善が求められるようになった．

3.　二値ベクトルで局所特徴量を表現する手法

これに対し，キーポイントの特徴記述を実数ベクトルで表現せず，二値のベクトルで表現するという手法が提案されている．B 次元の二値特徴量 $\mathbf{y}$ は，

$$\mathbf{y} \in \{-1, +1\}^B \tag{3}$$

もしくは

$$\mathbf{y} \in \{0, +1\}^B \tag{4}$$

として定義される．つまり，$\mathbf{y}$ の各要素は，二通りの整数のうち，いずれか一方の値のみを持つ．(3) 式を用いるか，(4) 式を用いるかは各提案手法によって異なるが，単に数学的な定式化において，都合の良い方が選ばれているだけであり，これらの間に本質的な違いは無い（例えば，後述する Binary Hashing による手法の場合は，(3) 式で定義されることが多い）．ポイントは，(3)(4) 式のいずれの場合においても，$\mathbf{y}$ を B ビットの 0 と 1 の列でメモリ上に格納できるという点である(注3)．そのため，メモリの消費量を大幅に減らせるというメリットがある．

また，もう一つのメリットは，二値特徴量間の距離をハミング距離で測れるという点である．ハミング距離とは，二つの同じ長さのビット列が与えられたとき，対応する位置において一致していないビットの個数に相当する．これは，2 つのビット列の XOR を計算し，1 が立っているビットの数を数えるだけで得られるため，極めて高速に計算可能である．図 2 に例を示しておく．現在，一般に利用されている CPU では，ビットカウント用の専用命令を実装しているものも多く，これを用いることでさらなる高速化も可能である．

(注3)：(3) 式の定義に従う場合でも，-1 を 0 とみなしてメモリ上に配置すればよい．

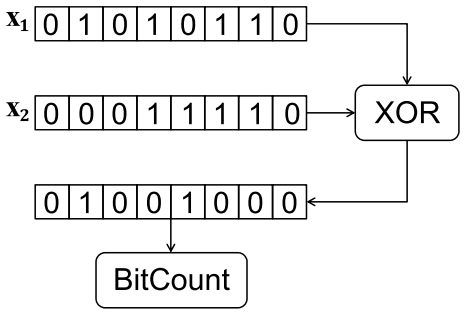


図 2　ハミング距離の計算

3.1　研 究 動 向

特徴量を二値のベクトルで表現できれば，メモリ消費量の削減と距離計算の高速化という二つの利点が得られる．そこで近年では，局所特徴量の記述性能を十分に保ったまま，如何にして二値で記述をするか，という点について議論が深まっている．図 3 は，その研究の動向を示したものである．対応点探索のための二値特徴量を抽出する手法は，次の 2 つに大別できる．

（1）　直接的に二値特徴量を生成する方法

実数で記述された特徴量を介さず，二値の特徴ベクトルを直接求める．

（2）　間接的に二値特徴量を生成する方法

従来の手法で実数の局所特徴量を算出し，これを二値特徴量に変換する．

直接／間接いずれのアプローチでも，初めは機械学習に基づかない単純な方法が検討され，その後に教師なし学習によって記述性能が改善できることが示された．特に近年では，教師あり学習によりさらなる記述能力の向上が見込めることが報告されており，(実験条件にもよるが) 64bits の二値特徴量で SIFT に匹敵する性能が達成された事例も報告されている [34].

4.　直接的に二値特徴量を生成する方法

2010 年に，機械学習によらない単純な手法として，BRIEF [12]

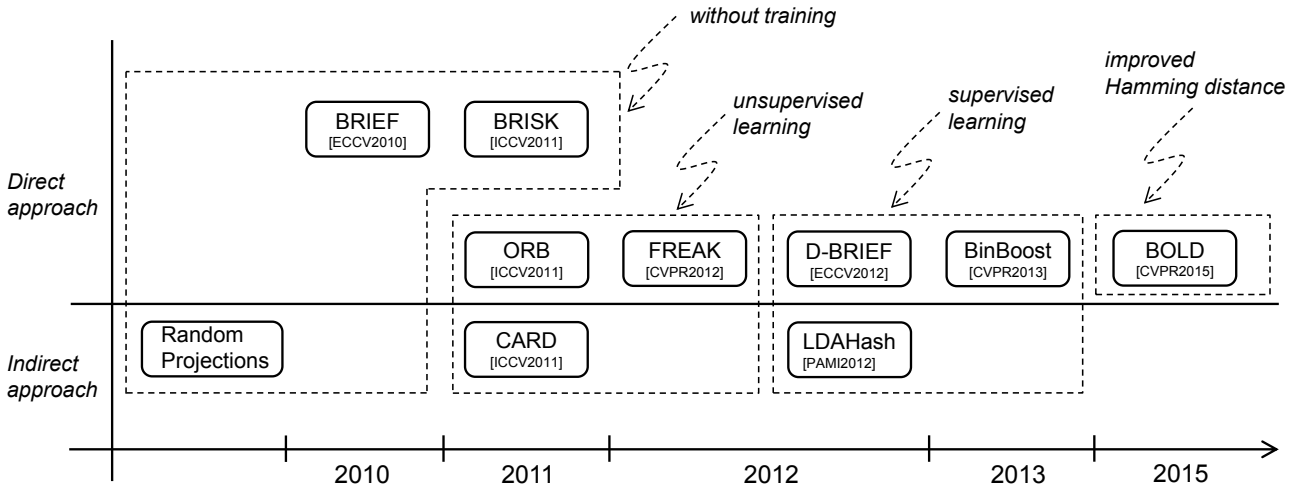


図 3　対応点探索のための二値特徴量の研究動向

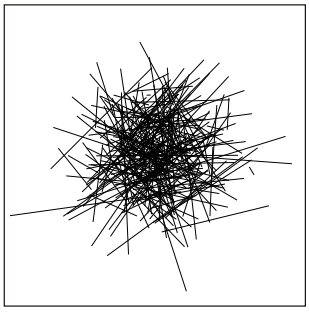


図 4　BRIEF のサンプリングパターン

が提案された．これはピクセル間の輝度差の情報から二値特徴量を生成するというものであり，アルゴリズムが非常にシンプルでありながらもそれなりの性能を発揮したため，注目を集めた．この考え方は後続の BRISK [19] や ORB [27] に受け継がれ，スケール・回転不変性に関する検討や，特徴量記述能力の改善が試みられた．ORB は OpenCV を開発している Willow Garage の研究であり，いち早く OpenCV に実装されたことから，発表直後から広く使われるようになった．ORB や FREAK [3] で教師なし学習の概念が取り入れられ，機械学習により特徴量記述性能が改善できることが示された．その後に提案された D-BRIEF [35] や BinBoost [34] では，教師あり学習に関する検討が行われた．これは，パッチの positive ペアと negative ペアを収集し，positive ペア同士のハミング距離が小さく，negative ペア同士のハミング距離が大きくなるように学習をすることで，より記述能力の高い特徴抽出を狙うというものである．また異なるアプローチとして，選択したペアを実行時に動的に切り替える BOLD [6] という手法も提案されている．本節では，以上に紹介した直接的に二値特徴量を生成する方法を，提案された順に沿って紹介する．

4.1 BRIEF: Binary Robust Independent Elementary Features

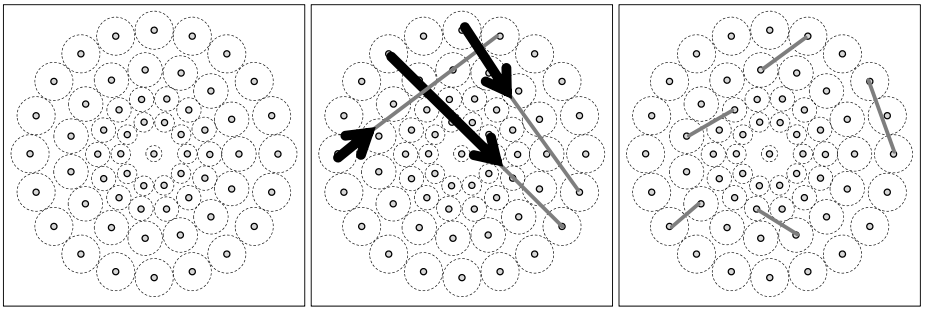
Calonder ら [12] によって提案された BRIEF は，キーポイント周辺のパッチ内においてランダムに選択された 2 点の輝度差の符号から二値特徴量を生成するというシンプルなアイディアに基づいた特徴記述手法である．パッチ内からランダムに選ばれた 2 点を $\mathbf{u}_i, \mathbf{v}_i \in \mathbb{R}^2$ と記述する．これを B 組用意しておく．すなわち $i = 1, \cdots, B$ である．それぞれの点における輝度を $I(\mathbf{u}_i), I(\mathbf{v}_i)$ と記述する．このとき，輝度差 $I(\mathbf{u}_i) - I(\mathbf{v}_i)$ が正であれば 1, 負であれば 0 を割り当てることで，B bits の二値特徴量を生成する．$\mathbf{u}_i, \mathbf{v}_i$ のペアの選び方には様々な方針が考えられるが，著者らは単純にキーポイントの位置を中心としたガウス分布に基づいてランダムに選択するのが良いという実験結果を示している．これを図 4 に示した．なお，ノイズに対する耐性を高めるため，パッチにはあらかじめガウシアンフィルタを適用してスムージングをかけておく．このように，2 点間の輝度差を所望のビット数分計算するだけで済むため，高速に特徴抽出を行える．

ただし，BRIEF のアルゴリズム自体にはスケール・回転不変性が無いため，これらの不変性が必要である場合は，SIFT や SURF によるキーポイント検出と組み合わせるなどといった工夫が必要である．このような不完全さがありながらも，2 点の輝度差で二値特徴量を生成し，ハミング距離でマッチングを行えば対応点探索が可能という実験事実は興味深いものである．この考え方は，後続の BRISK や ORB でも受け継がれている．

4.2 BRISK: Binary Robust Invariant Scalable Keypoints

Leutenegger ら [19] によって提案された BRISK は，離れた 2 点の輝度差に着目するという BRIEF の考え方にスケール不変性と回転不変性を導入したものである．

まずキーポイント検出について概略を述べる．スケール不変性を獲得するために，入力画像を多段階に縮小したピラミッド画像を作成する．それぞれのレイヤに対して FAST [26] を適用し，全ての画素に関して特徴点らしさのレスポンス値を計算しておく．この値がしきい値以上であり，かつ xy 方向およびスケール方向の 3 次元空間において局所最大となる点をキーポイントとして抽出する．キーポイントの座標およびスケールは，レスポンス値に対して 2 次の多項式フィッティングを適用する



(a) サンプリング位置　(b) 長距離ペア $\mathcal{L}$　(c) 短距離ペア $\mathcal{S}$

図 5　BRISK のサンプリングパターン

ことで高精度に求めている．

次に，特徴量記述について述べる．BRISK では，パッチに配置された 4 つの同心円上において，等間隔にサンプリングされた 60 箇所の輝度値を用いる（図 5(a)）．BRIEF はランダムサンプリングに基づいているため，ビット数× 2 回分の画素値のアクセスが必要だが，BRISK では規則的に並んだ 60 点の画素値のみが分かれば良い．各サンプリング点における輝度は，中心からの距離に比例する分散を持つガウスフィルタによりスムージングをかけることでノイズ耐性を獲得している．図 5(a) では，この分散の値を点線の丸で示した．以下，60 個のサンプリング点を $\mathbf{u}_i \in \mathbb{R}^2 \ (i = 1, \cdots, 60)$ と記述する．

BRISK では，サンプリング点間の距離が $\delta_{\min}$ 以上であるペアの集合 $\mathcal{L}$（長距離ペア）を用いてオリエンテーションの推定を行う．また，サンプリング点間の距離が $\delta_{\max}$ 以下であるペアの集合 $\mathcal{S}$（短距離ペア）を用いて特徴量の記述を行う．図 5(b)(c) に $\mathcal{L}, \mathcal{S}$ として選択されたペアの一部を示した．オリエンテーション推定に長距離 $\mathcal{L}$ を用いる理由は，パッチの大局的な勾配方向を捉えることを意図しているからである．一方，特徴量記述に短距離ペア $\mathcal{S}$ を用いるのは，パッチ内の局所的な画像特徴を捉えることを狙っているからである．

パッチのオリエンテーションは，長距離ペア $\mathcal{L}$ を用いて次のように算出される．まず，一組のペア $(\mathbf{u}_i, \mathbf{u}_j) \in \mathcal{L}$ に関して，勾配に相当する量を (5) 式で定義する．

$$\mathbf{g}(\mathbf{u}_i, \mathbf{u}_j) = (\mathbf{u}_j - \mathbf{u}_i) \cdot \frac{I(\mathbf{u}_j, \sigma_j) - I(\mathbf{u}_i, \sigma_i)}{||\mathbf{u}_j - \mathbf{u}_i||^2} \tag{5}$$

$I(\mathbf{u}_i, \sigma_i), I(\mathbf{u}_j, \sigma_j)$ は分散 σ_i, σ_j のガウスフィルタでスムージングされた後の輝度値である．勾配 $\mathbf{g}(\mathbf{u}_i, \mathbf{u}_j)$ はペア $\mathbf{u}_j, \mathbf{u}_i$ を結ぶ直線と同じ方向を向いており，その長さはペア間の輝度差で与えられる．図 5(b) では，これを太い矢印で示した．オリエンテーション α は，集合 $\mathcal{L}$ に所属するペアの平均勾配ベクトル $\mathbf{g}$ の角度として定義される．

$$\mathbf{g} = (g_x, g_y)^\top = \frac{1}{|\mathcal{L}|} \sum_{(\mathbf{u}_i, \mathbf{u}_j) \in \mathcal{L}} \mathbf{g}(\mathbf{u}_i, \mathbf{u}_j) \tag{6}$$

$$\alpha = \mathrm{atan2}(g_y, g_x) \tag{7}$$

推定した α に従って $\mathbf{u}_i, \mathbf{u}_j$ を回転させた座標位置を $\mathbf{u}_i^\alpha, \mathbf{u}_j^\alpha$ とする．これを用いて，(8) 式により二値特徴量の各ビットを算出する．

$$y_{ij} = \begin{cases} 1 & I(\mathbf{u}_j^\alpha, \sigma_j) > I(\mathbf{u}_i^\alpha, \sigma_i) \\ 0 & (\text{otherwise}) \end{cases}, \forall (\mathbf{u}_i, \mathbf{u}_j) \in \mathcal{S} \tag{8}$$

このように，特徴量記述には短距離ペア $\mathcal{S}$ を用いることで局所的な画像特徴を捉えている．$\mathcal{S}$ の要素数が 512 個になるように $\delta_{\max}$ が設定されているため，最終的に出力される二値特徴量は 512 ビットになる．

4.3　ORB: Oriented FAST and Rotated BRIEF

BRISK と同様に，ORB も BRIEF にスケールと回転不変性を導入した手法であり，2 点の輝度差を用いて二値特徴量を生成するという考え方に基づいている．キーポイント検出に FAST を用いるという点は BRISK と同じだが，オリエンテーションの推定方法と，輝度差を求める 2 点のペアの選択方法が異なっている．ここではキーポイント検出の説明は省略し，オリエンテーション推定と二値特徴量の生成方法について解説する．

ORB では，オリエンテーション α を推定するときに勾配を用いない．パッチの輝度値に関する 0,1 次モーメントから重心 (intensity centroid, C) を計算し，キーポイント中心 O と重心 C を結ぶ直線の傾きを α として用いる．キーポイント中心 O を原点とすると，モーメント m_{pq} と重心 C は次のように計算できる．

$$m_{pq} = \sum_{x,y} x^p y^q I(x, y) \tag{9}$$

$$C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right) \tag{10}$$

従って，オリエンテーション α は次式で与えられる．

$$\alpha = \mathrm{atan2}(m_{01}, m_{10}) \tag{11}$$

二値特徴量を求める際には，輝度差を求める 2 点のペアの座標位置を α だけ回転させることで，回転不変性を獲得している．

ORB では，統計的に「良い」性質を持つサンプリングペアを教師なし学習により選択する．さて，一体どのような二値特徴量が「良い」特徴量であろうか？情報量の観点から考えると，各ビット y_i が 0 もしくは 1 になる確率はそれぞれ 50% となることが望ましい．また，異なる二つのビット y_i, y_j は確率的に独立である方が良い．そのような二値特徴量はエントロピーが高く無駄がないといえるからである．これについては，ORB に限らず，様々な文献で同様の指摘がされている [36], [38]．これ

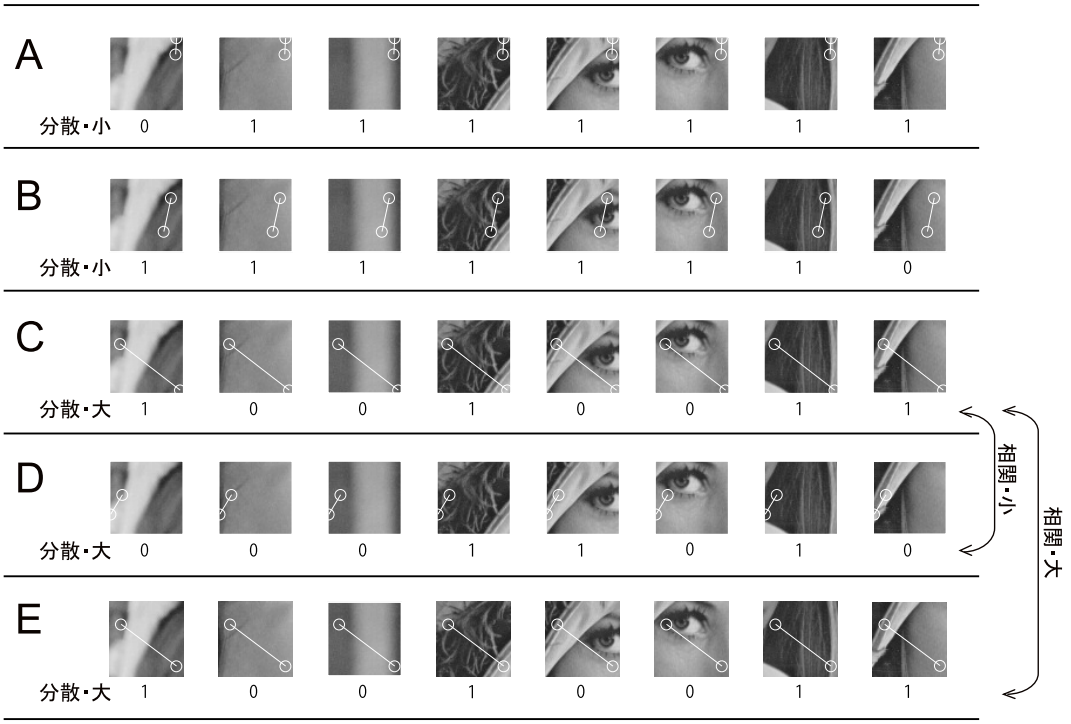


図 6　サンプリングペアの統計的な性質

は，異なる二つのビット y_i, y_j の相関が小さく，y_i の分散が大きい方が良い二値特徴量である，と言い換えても良い．例えば，y_i が 100%の確率で 0 を取ってしまうような極端なケースの場合，y_i の分散はゼロになる．この場合，明らかに y_i は二値特徴量としては不適切である．逆に，0 と 1 が等確率でバランスよく発生するのであれば，y_i は最大の分散を持つ．これは良い二値特徴量であるといってよい．

以上の議論は，具体例を挙げて考えるとより直感的で分かりやすい．図 6 は，8 つのパッチに対して五種類のペア（A〜E）から二値特徴量を生成した例である．AB では，どのパッチについてもほとんど同じ値が出力されているため，このようなペアは特徴記述能力に乏しく適切ではない．一方，CDE は 0 と 1 の値がほぼ等確率で出力されており，優秀なペアだといえる．さて，ここで C をベストなペアだとして採用したとしよう．このとき，次なる選択肢として優秀なペアは DE のどちらかである．単純に分散値の大小で考えれば，D よりも E の方が，分散が大きく適切であるかのように思われるが，これは正しくない．なぜなら，E は既に採用した C と特徴量の出力値が似ており，これを選択しても情報量は増えないからである．この場合は，C の出力とは相関が小さい D を選ぶ方が良い．このように，エントロピーが高くなるようにペアを選択することで，限られたビット数を最大限に活かせる無駄のない二値特徴量を生成できる．

実際の処理では，あらかじめ大量のパッチを集めておき，パッチ内で有り得る 205,590 種類のペアを全て候補として列挙し，ビットの分散が大きく，かつペア同士の相関が低くなるようなペアを Greedy アルゴリズムで選択している．以下にアルゴリズムの概要を示す．

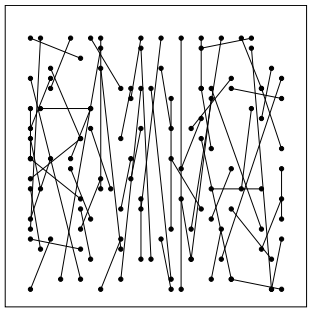


図 7　ORB のサンプリングパターン

（1） 全てのペアの候補について，ビットの分散が大きいもの順にソート．

（2） 最大の分散を持つペアを採用．

（3） 次に大きな分散を持つペアに着目し，このペアが採用済みのどのペアとも相関が低いのであれば採用．相関が高ければ棄却．

（4） 3 番目のステップを繰り返し，256 組のペアが見つかった時点で終了．

最終的に得られる特徴量は 256 ビットである．図 7 に ORB によって選択された上位 64 個のペアを示した．選択されたペアには縦方向の偏りが生じていることが分かる．これは，オリエンテーションの補正が入ることでパッチ内の輝度の分布に偏りが生じるからである．

4.4　FREAK: Fast Retina Keypoint

Alahi ら [3] によって提案された FREAK もまた，ORB や BRISK と同様に輝度差の符号によって二値特徴量を生成する手法であり，生物の網膜の構造を模している点が特徴的である．

FREAK では，眼球内において，網膜の中心に近づくほど神経

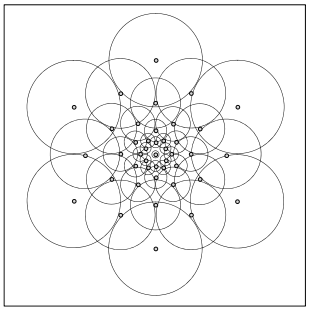


図 8　FREAK のサンプリングパターン

節細胞の密度が高くなるという生物学的事実を参考とし，キーポイント中心に近づくほど密度が高くなるようなサンプリングパターンを用いることを提案している．図 8 は FREAK で用いられているサンプリングのパターンである．それぞれのサンプリング点を中心とする円は，平滑化のためのガウスフィルタの範囲を表す．この円の大きさは外側に向かうほど指数関数的に大きくなるように設計されている．外側のサンプリング点における平滑化の範囲は，互いに重なり合うほど大きくなるから，サンプリング点における値は隣り合う点と独立にはならない．しかしながら，このような冗長性は生物の視覚システムの中にも存在しており，これが識別性能の向上に寄与すると Alahi らは主張している．

FREAK で採用されているサンプリング点の数は ORB や BRISK と比べても少なく，たった 43 箇所である．しかしながら，これらの全ての組み合わせを用いて二値特徴量を抽出すると，そのビット長は数千ビットにも及ぶため，ここでもやはりペアの選択が必要である．そこで，ORB と同じ Greedy アルゴリズムを適用することで，二値特徴量の情報量が大きくなるような 512 種類のペアを選択している．

ここで Alahi らは，選ばれた 512 種類のペアを観察すると，そこには生物の視覚システムと似た挙動が現れていると主張している．生物の視覚システムは，周辺視野で大まかに対象を把握したのちに，中心視野で対象物をより細かく観察するように機能する．一方で，FREAK において用いられている 512 種類のペアを，Greedy アルゴリズムで選ばれた順に追っていくと，初めは外側のサンプリングパターンが選択されていることが多く，後半になると中心に近いサンプリングパターンが選択されていることが多いことがわかる．すなわち，生物の視覚システムと同様に coarse-to-fine な特徴選択が行われていることがわかる．

この点に着目し，FREAK では 512 ビットの二値特徴量の距離計算を行う際に，128 ビットごとに 4 分割し，4 段のカスケード構造を取ることで，マッチング処理の高速化を図っている．すなわち，各段では 128 ビット分のみの距離計算を行い，距離の値が所定のしきい値を超える場合は計算を打ち切ることで，マッチング処理を高速化している．実際には，初段で 90%以上の候補が棄却されるため，実質 128 ビット分の距離計算のみでマッチングが行える．

なお，FREAK におけるキーポイント位置検出，スケールおよびオリエンテーション推定は BRISK とほぼ全く同じである．

4.5　D-BRIEF: Discriminative BRIEF

BRIEF, BRISK, ORB, FREAK では，事前に定めたサンプリング点の周辺に平滑化などのフィルタを適用した後に，2 つのサンプリング点間の輝度差の符号を用いて二値特徴量を定義した．キーポイント周辺のパッチを $\mathbf{x}$ とベクトルで表記 (注4) すれば，フィルタの演算と輝度差の演算は，$\mathbf{x}$ に関して線形であるから，結局のところ i 番目のビット y_i は (12) 式で表せる (注5)．

$$y_i = \mathrm{sgn}(\mathbf{w}_i^\top \mathbf{x} + \tau_i) \tag{12}$$

例えば，フィルタ演算を用いず，単純にサンプリング点間の輝度差の符号で二値特徴量を生成するのであれば，$\tau_i = 0$ とし，$\mathbf{w}_i$ のサンプリング点のペアに対応する位置にそれぞれ+1 と-1，それ以外に 0 を代入すればよい．サンプリング点周辺について平滑化フィルタをかける場合であっても，$\mathbf{w}_i$ に適切な重みを設定してやればよい．これを可視化したものを図 9 に示す．結局のところ，BRIEF, BRISK, ORB, FREAK の違いは，$\mathbf{w}_i$ の選び方の違いに他ならないといえる．Trzcinski ら [35] によって提案された D-BRIEF は，教師あり学習を適用することで，より記述能力の高い二値特徴量を生成できる $\mathbf{w}_i$ と τ_i を求めようとするものである．図 10 は，教師あり学習で得られた $\mathbf{w}_i$ を可視化したものである．教示データとしては，物理的に同一の個所を捉えており，かつスケールとオリエンテーションについて正規化済みのパッチのペアを用いる．以下，これを positive ペアと呼ぶ．また，全く異なる対象を捉えているパッチのペアを negative ペアと呼ぶ．図 11 にそれぞれの例を示した．positive ペアと negative ペアを用いて $\mathbf{w}_i$ と τ_i を学習することで，識別的な二値特徴抽出を行うことができる．実際には $\mathbf{w}_i$ はパッチのピクセル数分の次元を持つため，$\mathbf{w}_i$ と $\mathbf{x}$ の内積を直接計算すると大きな計算コストがかかる．そこで D-BRIEF ではさらなる工夫として，高速に演算可能な線形フィルタの候補を多数用意しておき，$\mathbf{w}_i$ をこれらのフィルタの中から選ばれた少数のフィルタの線形和で表現することで，$\mathbf{w}_i^\top \mathbf{x}$ の計算を高速化している．

$$\mathbf{w}_i \approx \mathbf{D}\mathbf{s}_i \tag{13}$$

$\mathbf{D}$ はフィルタの辞書であり，$\mathbf{s}_i$ は係数である．$\mathbf{s}_i$ がスパースであれば，$\mathbf{w}_i$ は少ない数のフィルタで再構成されることになる．フィルタの候補としては，矩形やガウスフィルタを用いる．例えば矩形のフィルタであれば，積分画像を用いて高速に演算できる．図 12 に，矩形フィルタで近似した例を示す．

D-BRIEF における学習処理は，(14) 式のコスト関数を最小化することで達成できる．

$$\min_{(\mathbf{S}_i,\tau_i)} \sum_{i\in 1,\cdots,B} \sum_{(\mathbf{x},\mathbf{x}')\in\mathcal{N}} \mathrm{sgn}((\mathbf{D}\mathbf{s}_i)^\top \mathbf{x} + \tau_i)\mathrm{sgn}((\mathbf{D}\mathbf{s}_i)^\top \mathbf{x}' + \tau_i)$$

(注4)：例えば，$N \times N$ ピクセルのサイズのパッチから二値特徴量を抽出することを考えるのであれば，$\mathbf{x}$ はパッチ内の輝度を並べた N^2 次元のベクトルで表せる．

(注5)：これは，5. 節で後述する Binary Hashing と本質的には同じ考え方である．

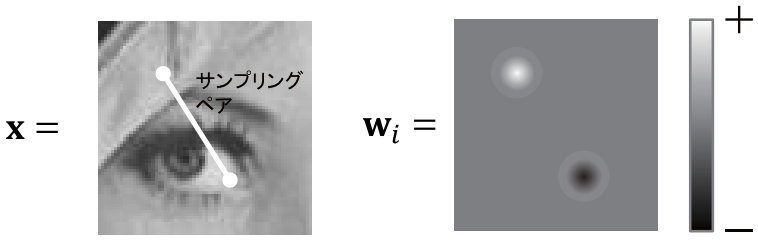


図 9　BRIEF, BRISK, ORB, FREAK における重みベクトル $\mathbf{w}_i$

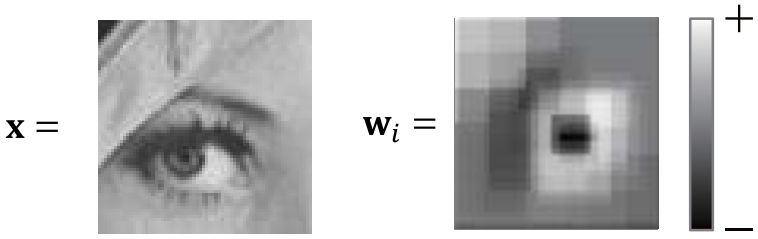


図 10　D-BRIEF における重みベクトル $\mathbf{w}_i$(右側の重みを可視化した濃淡画像は，原著論文 [35] の図 1 から引用した．)

$$
\begin{aligned}
&- \sum_{(\mathbf{x},\mathbf{x}')\in\mathcal{P}} \mathrm{sgn}((\mathbf{D}\mathbf{s}_i)^\top\mathbf{x}+\tau_i)\mathrm{sgn}((\mathbf{D}\mathbf{s}_i)^\top\mathbf{x}'+\tau_i) \\
&+ \lambda|\mathbf{s}_i|_1 \\
&\quad \text{subject to } (\mathbf{D}\mathbf{s}_i)^\top(\mathbf{D}\mathbf{s}_j)=\delta_{ij} \qquad (14)
\end{aligned}
$$

ここで，$(\mathbf{x},\mathbf{x}')\in\mathcal{P}$ は positive ペア，$(\mathbf{x},\mathbf{x}')\in\mathcal{N}$ は negative ペアである．λ は $\mathbf{s}_i$ をどれだけ疎にするかを決める係数である．λ が大きいほど $\mathbf{s}_i$ が疎になり，$\mathbf{w}_i$ はより少ない数のフィルタで再構成されることになる．また，δ_{ij} は $i=j$ ときに 1，それ以外は 0 を取る関数である．制約条件 $(\mathbf{D}\mathbf{s}_i)^\top(\mathbf{D}\mathbf{s}_j)=\delta_{ij}$ は，異なるビット同士を独立にして二値特徴量が持つ情報量を高めることを狙っており，これは ORB の学習における考え方に類似している．

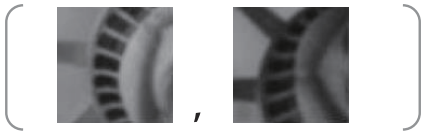
(14) 式は原著論文に記載の表記そのものであるが，やや複雑で理解しにくいと思われるため，少し整理しておく．$\mathbf{x},\mathbf{x}'$ を二値特徴量に変換したものをそれぞれ $\mathbf{y},\mathbf{y}'$ と表記すると，(14) 式は以下のように簡単に整理できる．

$$
\min_{(\mathbf{s}_i,\tau_i)} \sum_{(\mathbf{x},\mathbf{x}')\in\mathcal{N}} \mathbf{y}^\top\mathbf{y}' - \sum_{(\mathbf{x},\mathbf{x}')\in\mathcal{P}} \mathbf{y}^\top\mathbf{y}' + \lambda|\mathbf{s}_i|_1 \qquad (15)
$$

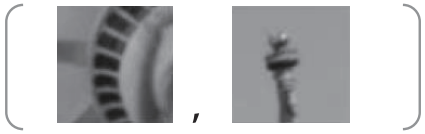
つまり，$\mathcal{N}$ に所属するペアに関しては，二値特徴量同士の内積を小さくし，$\mathcal{P}$ に所属するペアに関しては，二値特徴量同士の内積を大きくするように $\mathbf{s}_i,\tau_i$ を学習していることが分かる．D-BRIEF においては，$\mathbf{y},\mathbf{y}'\in\{-1,+1\}^B$ と定義されているから、二値特徴量のハミング距離 $d_H(\mathbf{y},\mathbf{y}')$ と内積 $\mathbf{y}^\top\mathbf{y}'$ の間には (16) 式の関係がある．

$$
\mathbf{y}^\top\mathbf{y}' = B - 2d_H(\mathbf{y},\mathbf{y}') \qquad (16)
$$

つまり，ふたつの二値特徴量 $\mathbf{y},\mathbf{y}'$ の内積を大きくするという作用は，$\mathbf{y}$ と $\mathbf{y}'$ のハミング距離を小さくするという作用に相当する．したがって，(14) 式を最小化することで，$\mathcal{P}$ に所属するペア間のハミング距離が小さくなり，$\mathcal{N}$ に所属するペア間のハミング距離が大きくなるように $\mathbf{w}_i,\tau_i$ が最適化されることが分かる．



(a) positive ペア



(b) negative ペア

図 11　教示データ

以上，D-BRIEF における問題は (14) 式の最小化問題に帰着できることを示したが，実際には (14) 式を $\mathbf{s}_i,\tau_i$ に関して直接解くことは困難であるため，最適化のための工夫が必要である．そこで，sgn 関数と L1 正則化の項を取り除き，$\mathbf{s}_i$ ではなく，$\mathbf{w}_i$ について解くことを考えると，(17) 式のように簡略化できる．

$$
\{\mathbf{w}_i^0\} = \arg\min_{\{\mathbf{w}_i\}} \sum_i \frac{\sum_{(\mathbf{x},\mathbf{x}')\in\mathcal{P}}(\mathbf{w}_i^\top(\mathbf{x}-\mathbf{x}'))^2}{\sum_{(\mathbf{x},\mathbf{x}')\in\mathcal{N}}(\mathbf{w}_i^\top(\mathbf{x}-\mathbf{x}'))^2} \qquad (17)
$$

$\mathbf{w}_i$ に直交性を仮定すると，これは Linear Discriminant Embedding (LDE) [11] の問題設定そのものであり，固有値計算で解を得ることができる．sgn 関数を取り除くことでコスト関数から τ_i が消えてしまっているが，LDE によって求めた $\mathbf{w}_i$ を固定し，τ_i に関する一次元の探索問題として (14) 式を解きなおすことで τ_i の解を得ることができる (注6)．

$\mathbf{w}_i^0$ を少数のフィルタの線形和で近似するためには，$\mathbf{w}_i^0$ と $\mathbf{D}\mathbf{s}_i$ の誤差が小さく，かつ $\mathbf{s}_i$ ができるだけ疎になるような解を

(注6)：話が前後するが，このしきい値 τ_i を一次元の探索問題として解くという解き方は，後述する LDAHash [32] で行われているやり方と全く同じである．

求めればよい．すなわち，求めたそれぞれの $\mathbf{w}_i$ について，次のコスト関数を最小化することで，$\mathbf{s}_i$ を得る．

$$\{\mathbf{s}_i^0\} = \arg\min_{\mathbf{s}_i} ||\mathbf{w}_i^0 - \mathbf{D}\mathbf{s}_i||_2^2 + \lambda|\mathbf{s}_i|_1 \quad (18)$$

これは凸関数であるから，近接勾配法などで解くことができる．

4.6 BinBoost

Trzcinski ら [34] によって提案された BinBoost もまた教師あり学習に基づく手法であり，K 個の弱識別器の線形和の符号で 1 ビット分の二値特徴量を生成する．

$$y_i(\mathbf{x}) = \mathrm{sgn}(\mathbf{w}_i^\top \mathbf{h}_i(\mathbf{x})) \quad (19)$$

$\mathbf{x}$ はパッチの輝度，$y_i \in \{-1, +1\}$ は i 番目のビット，$\mathbf{h}_i(\mathbf{x}) = (h_{i,1}(\mathbf{x}), h_{i,2}(\mathbf{x}), \cdots, h_{i,K}(\mathbf{x}))^\top \in \mathbb{R}^K$ は K 個の弱識別器の出力，$\mathbf{w}_i \in \mathbb{R}^K$ はそれらに対する重みである．$h_{i,j}(\mathbf{x})$ はパッチの輝度 $\mathbf{x}$ に関して非線形の関数でもよく，任意の特徴抽出手法を適用できる．この点が D-BRIEF と異なる部分であり，弱識別器に適切な特徴抽出器を用いることで，高い記述能力を持った二値特徴量が生成できる．

論文中では，パッチ内の矩形領域内における輝度勾配の方向を弱識別器として提案している．弱識別器 $h_{i,j}(\mathbf{x})$ は矩形の領域 R, 勾配のオリエンテーション e，しきい値 T の 3 つのパラメータを持ち，領域 R 内の正規化勾配方向ヒストグラムに関して，オリエンテーション e に該当する値がしきい値 T を超える場合は -1，そうでない場合は $+1$ を出力する関数である．図 13 に弱識別器の例を示す．これらの弱識別器を K 個集めることで，はじめて $\mathbf{h}_i(\mathbf{x})$ が得られる．すなわち，ビットごとに K 個の弱識別器が使用されるため，B ビットの二値特徴量を得たいのであれば，合計で $B \times K$ 個の弱識別器が必要になる (図 14).

BinBoost では，AdaBoost で採用されている考え方に近い定式化が行われている．まず，N 個のラベル付き訓練サンプルを $\{(\mathbf{x}_n, \mathbf{x}'_n, l_n)\}_{n=1}^N$ で定義する．$\mathbf{x}_n$ と $\mathbf{x}'_n$ が positive ペアである場合は $l_n = +1$, negative ペアである場合は $l_n = -1$ とする．このとき，(20) 式のコスト関数を $\mathbf{w}_i$ と $\mathbf{h}_i$ に関して最小化することで学習を行う．

$$\mathcal{L} = \min_{\{\mathbf{w}_i, \mathbf{h}_i\}_{i=1}^B} \sum_{n=1}^N \exp(-\gamma l_n \sum_{i=1}^B c_i(\mathbf{x}_n, \mathbf{x}'_n; \mathbf{w}_i, \mathbf{h}_i)) \quad (20)$$

γ は学習のパラメータであり，関数 c_i は次のように定義されている．

$$\begin{aligned} c_i(\mathbf{x}_n, \mathbf{x}'_n; \mathbf{w}_i, \mathbf{h}_i) &= y_i(\mathbf{x}) y_i(\mathbf{x}') \quad (21) \\ &= \mathrm{sgn}(\mathbf{w}_i^\top \mathbf{h}_i(\mathbf{x})) \mathrm{sgn}(\mathbf{w}_i^\top \mathbf{h}_i(\mathbf{x}')) \end{aligned}$$

このコスト関数の最小化は，$l_n = +1$ のペアに関してはハミング距離を小さくし，$l_n = -1$ のペアに関してはハミング距離を大きくするように作用する．これは (16) 式の関係が成立することから明らかである．

AdaBoost における考え方に倣い，$i = 1$ から順番に弱識別器 $\mathbf{h}_i$ と重み $\mathbf{w}_i$ を最適化することを考える．これは，(20) 式の代わりに (22) 式のコスト関数を最大化することで達成される．

$$\max_{\mathbf{w}_i, \mathbf{h}_i} \sum_{n=1}^N l_n W_i(n) c_i(\mathbf{x}_n, \mathbf{x}'_n; \mathbf{w}_i, \mathbf{h}_i) \quad (22)$$

$W_i(n)$ は訓練サンプルごとに定義される重み係数であり，次のように定義される．

$$W_i(n) = \exp(-\gamma l_n \sum_{i'=1}^{i-1} c_{i'}(\mathbf{x}_n, \mathbf{x}'_n; \mathbf{w}_i, \mathbf{h}_i)) \quad (23)$$

すなわち，$1, \cdots, i-1$ 番目までのビットで正しく識別できていない訓練サンプルは重みが大きくなり，既に識別できている訓練サンプルは重みが小さくなる．これは AdaBoost の考え方と非常に似ている．

関数 c_i の中にある sgn 関数は微分できないため，依然として (22) 式は解くことが難しい．そこで sgn 関数を取り除き，近似したコスト関数を最大化する．これを (24) 式に示す．

$$\max_{\mathbf{w}_i, \mathbf{h}_i} \mathbf{w}_i^\top (\sum_{n=1}^N l_n W_i(n) \mathbf{h}_i(\mathbf{x}) \mathbf{h}_i(\mathbf{x}')^\top) \mathbf{w}_i \quad (24)$$

まず，$\mathbf{h}_i$ に関してこの問題を解くことを考える．弱識別器の候補は無数に存在して良いが，それらの中から大きな重み $W_i(n)$ が与えられている訓練サンプルを適切に識別できる弱識別器の集合を求めなければならない．これは特徴選択の問題であり，BinBoost では, [30] で提案されている特徴選択手法を適用してビット毎に K 個の弱識別器を選択している．

次に，$\mathbf{w}_i$ に関して (24) 式を解くことを考える．$\mathbf{h}_i$ が決まれば，(24) 式は次のように簡単に整理できる．

$$\max_{\mathbf{w}_i} \mathbf{w}_i^\top \mathbf{M} \mathbf{w}_i \quad (25)$$

なお，

$$\mathbf{M} = \sum_{n=1}^N l_n W_i(n) \mathbf{h}_i(\mathbf{x}) \mathbf{h}_i(\mathbf{x}')^\top \quad (26)$$

と置いた．(19) 式からも明らかである通り，$\mathbf{w}_i$ はスケール倍の不定性があるため $||\mathbf{w}_i||_2 = 1$ としても一般性は失われない．従ってこれを仮定すると，(25) 式の最大化問題は $\mathbf{M}$ の最大固有値に対応する固有ベクトルを求めることで解くことができる．

以上の手続きにより，i 番目のビットを生成するための K 個の弱識別器 $\mathbf{h}_i(\mathbf{x})$，重み $\mathbf{w}_i$ を得ることができた．これを所望のビット数 B が得られるまで繰り返すことで二値特徴量を得る．論文中では，64 ビットの二値特徴量で SIFT の性能を超えたとする報告が記載されている．

4.7 BOLD: Binary Online Learned Descriptor

Balntas らが提案する BOLD [6] は，BRIEF，BRISK，ORB，FREAK 等と同様にキーポイント周辺のパッチ内のサンプリングペアの輝度差により二値特徴量を生成する手法である．従来の直接的に二値特徴量を生成する方法 [12], [19], [27], [3] は，学習アルゴリズムによりオフラインでサンプリングパターンを最適化していた．しかし，事前に決定したサンプリングパターンは固定されているため，入力されるパッチ画像との相性が悪いと大きな性能低下を招くことがある．BOLD では二値特徴量の

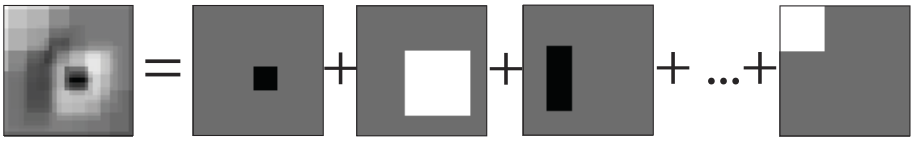
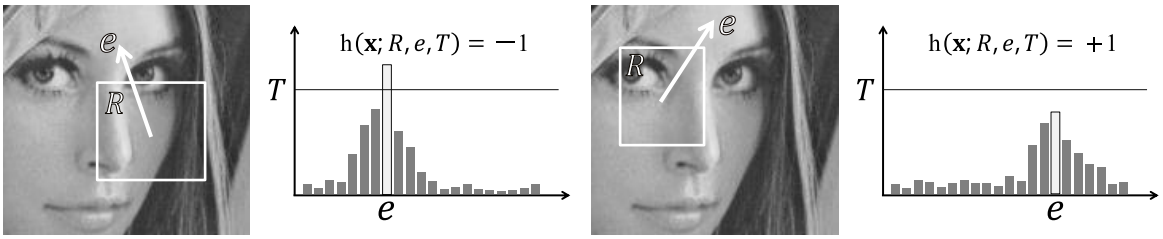


図 12　$\mathbf{w}_i$ の近似による高速化（原著論文 [35] の図 1 から引用）



(a) 弱識別器の出力が負　　(b) 弱識別器の出力が正

図 13　弱識別器の例

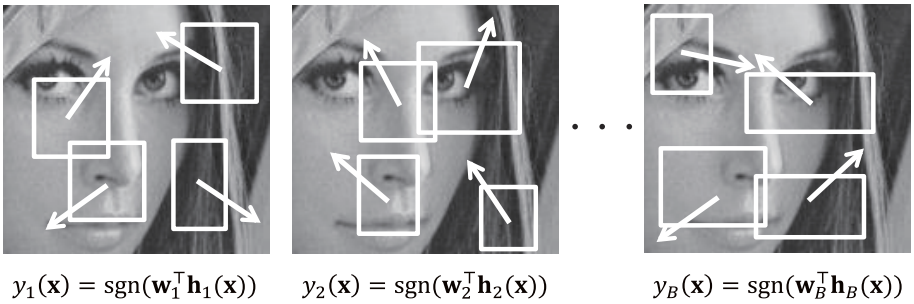


図 14　BinBoost における二値特徴量の生成．1 ビットあたり K 個の弱識別器が用いられるため，B ビットの二値特徴量を生成するためには合計 $B \times K$ 個の弱識別器が選択される．

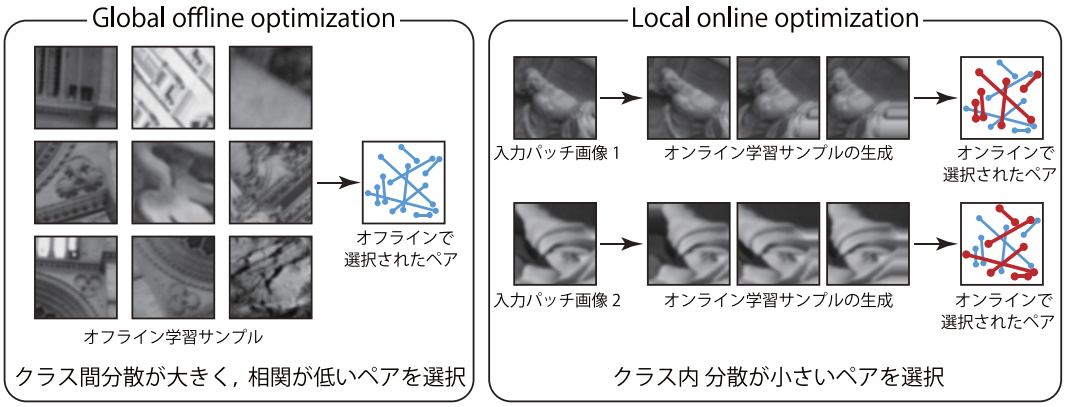


図 15　BOLD によるサンプリング点の選択方法

サンプリングパターンを入力パッチ画像に応じてオンラインで選択することで，この問題を解決している．

BOLD は図 15 に示すように“グローバルオフライン最適化”と“ローカルオンライン最適化”の 2 段階のアルゴリズムを用いてサンプリングペアを選択する．グローバルオフライン最適化では大量の学習用パッチ画像から最適なサンプリングペアを選択する．サンプリングペアの選択方法は ORB と同様に各学習パッチ画像のバイナリコードのビット間の分散が大きく，かつペア同士の相関が低くなるペアを Greedy アルゴリズムで選択する．ORB では 256 組の最適なペアを選択するが，BOLD では 512 組のペアをグローバルオフライン最適化で選択する．さて，ここで問題となるのは，パッチの歪み方によっては著しくマッチング性能が低下し得るという点である．そこで，グローバルオフライン最適化で選択した 512 ビットのサンプリングペアから入力画像に最適なサンプリングペアを選択するローカルオンライン最適化を行う．まず，変形や歪みに不変な特徴量を生成するために，入力パッチ画像にアフィン変換を施して追加パッチ画像を生成する (図 15)．次に，グローバルオフライン最適化で選択した 512 ビットのペアを用いて追加パッチ画像間のクラス内分散を計算する．そして，クラス内分散が 0 となるペア (すなわちアフィン変換の影響を受けにくいペア) をオンラインで選択し，特徴量間のハミング距離を計算する．

これは次のように定式化できる．オンラインで選択したペア (クラス内分散=0) を 1，選択されなかったペア (クラス内分散 $\neq$0) を 0 としたバイナリマスクを $\boldsymbol{\beta}$ と記述する．このとき，パッチ p, q の二値特徴量 $\mathbf{y}_p, \mathbf{y}_q$ のハミング距離 $d_H(\mathbf{y}_p, \mathbf{y}_q)$ はバイナリマスク $\boldsymbol{\beta}_p, \boldsymbol{\beta}_q$ を用いて (27) 式で計算できる．

$$d_H(\mathbf{y}_p, \mathbf{y}_q) = \frac{1}{G_p}\boldsymbol{\beta}_p \wedge \mathbf{y}_p \oplus \mathbf{y}_q + \frac{1}{G_q}\boldsymbol{\beta}_q \wedge \mathbf{y}_p \oplus \mathbf{y}_q \quad (27)$$

ここで，G はバイナリマスク $\boldsymbol{\beta}$ の 1 が立っているビットの数である．著者らはローカルオンライン最適化でのパッチ画像のアフィン変換の数は 2 回程度で最も良い性能が得られることを実験的に確認している．このようにして BOLD はアフィン変換にロバストなペアをオンラインで選択し，選択されたペアのみについてハミング距離を計算することで，性能の高い二値特徴量を表現している．

5. 間接的に二値特徴量を生成する方法

実数ベクトルを二値ベクトルに変換する手法は Binary Hashing と呼ばれ，コンピュータビジョンの分野だけでなく，機械学習や web，大規模検索などといった他の分野でも活発に研究されている．その中でも最も基本的なアプローチは，線形写像と符号関数による手法である．これを (28) 式に示す．

$$\mathbf{y} = \mathrm{sgn}(\mathbf{P}\mathbf{x} + \mathbf{t}) \quad (28)$$

$\mathbf{x}$ は D 次元の実数ベクトル，$\mathbf{y}$ は B 次元の二値ベクトル，$\mathbf{t}$ はしきい値，$\mathbf{P}$ は B 行 D 列の変換行列である．$\mathbf{P}$ および $\mathbf{t}$ をどのように設計するのかがポイントであり，これまでに様々な研究が行われている．

本節では，まずは最も単純な方法である Random projections [5], [13] を紹介する．次に，対応点探索のための局所特徴量を二値に変換することを特に意図して設計された二つの手法 (CARD [4], LDAHash [32]) を紹介する．CARD は教師なし学習，LDAHash は教師あり学習に基づく方法である．

5.1 Random Projections

Random projections [5], [13] は最も簡単な Binary Hashing 手法であり，$\mathbf{P}$ の各要素を正規分布に従う乱数で生成する．これは二値ベクトルに変換するやり方としては一見乱暴に思えるが，実は必ずしもそうではない．なぜなら，$B \to \infty$ の極限において，元々の実数の特徴空間におけるベクトル間角度と，二値特徴量のハミング距離とが，定数倍の関係を持つからである．

この理由は，実数ベクトルが 2 次元 ($D = 2$) のケースで考えると直感的に理解しやすい．図 16 は，2 次元の空間において，原点を中心とした半径 1 の円上に二つの実数ベクトル $\mathbf{x}_1, \mathbf{x}_2$ が存在している様子を示したものである．今，これらのベクトルの間の角度を θ とおく．

(28) 式は線形の変換を行った後にその符号を得るというものである．$\mathbf{t} = 0$ であれば，これは $\mathbf{P}$ の各行が原点を通る直線を定義しており，これが 2 次元の空間をふたつの領域に分割していると解釈できる．二つに分かたれた空間のうち，実数特徴量がどちらに属しているかによって，アサインされるビットが異なるわけである．

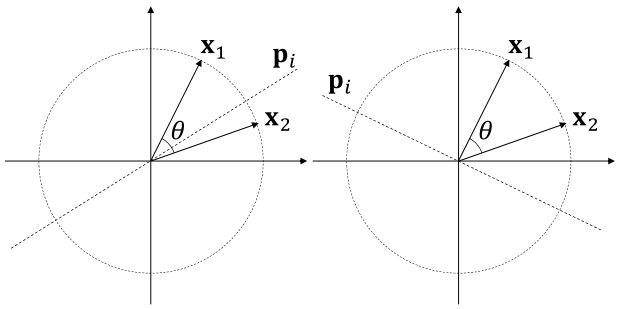


図 16 Random projections におけるベクトル間角度とハミング距離の関係

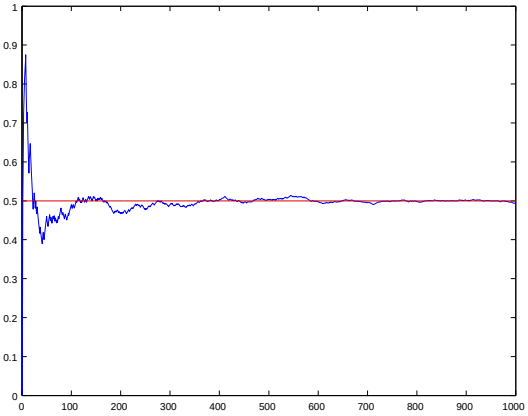


図 17 Random Projections による近似の収束の様子

Random projections では $\mathbf{P}$ を乱数で生成するため，2 次元の空間をふたつに分割する直線もまた，ランダムに決定されることになる．θ が大きければ $\mathbf{x}_1$ と $\mathbf{x}_2$ に異なるビットがアサインされる確率が高くなり，θ が小さければ $\mathbf{x}_1$ と $\mathbf{x}_2$ に異なるビットがアサインされる確率が低くなる．明らかに，この直線が $\mathbf{x}_1$ と $\mathbf{x}_2$ の間に入る確率は，θ に比例している．すなわち，(29) 式が成立する．

$$\Pr[\mathrm{sgn}(\mathbf{p}_i^\top \mathbf{x}_1) \neq \mathrm{sgn}(\mathbf{p}_i^\top \mathbf{x}_2)] = \frac{\theta}{\pi} \quad (29)$$

(29) 式の左辺は，ある直線 $\mathbf{p}_i$ で決定されるビットが異なる確率であり，右辺は $\mathbf{x}_1$ および $\mathbf{x}_2$ の間の角度を正規化した値である．左辺は確率であるが，$B \to \infty$ の極限においてこれはハミング距離をビット長 B で正規化した値と一致する．したがって，Random projections が実数の空間における距離関係を保存することがわかる．

図 17 は，二つのベクトル $\mathbf{x}_1 = (1, 0)^\top, \mathbf{x}_2 = (0, 1)^\top$ のベクトル間角度 θ を π で正規化した値 θ/π を random projections で近似した例である．x 軸はビット数，y 軸はハミング距離をビット長で正規化した値であり，この事例における真値は $\theta/\pi = 0.5$ である．始めは近似精度が悪いが徐々に改善してゆき，500 ビット程度でほぼ収束していることがわかる．

実際に SIFT や SURF をこの方法で二値特徴量にするためには，平均を原点に移動 (mean centering) した後に Random projections を計算すればよい．もちろん，Random projections で十分な性能を出すためには，比較的長いビット長が必要であるが，

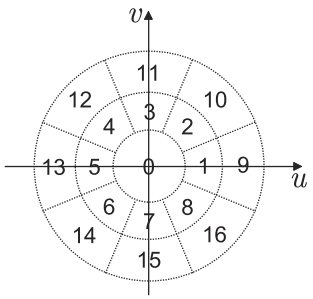


図 18　CARD におけるパッチの分割

それでも十分実用に足る場合も多い．Random Projections は学習処理も不要であるため，手軽に試せるのは大きな利点である．

5.2　CARD: Compact And Real-time Descriptors

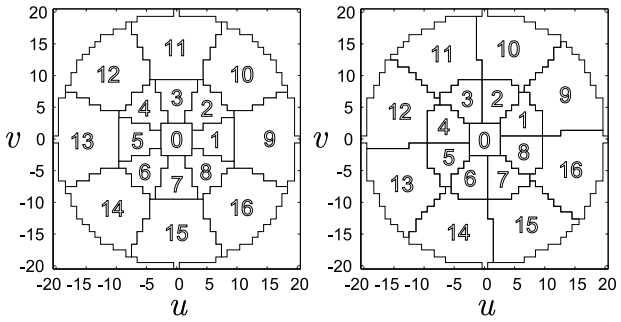
Ambai [4] らによって提案された CARD は，SIFT とほぼ同等の定義に基づく勾配ヒストグラム特徴量を抽出し，これを二値特徴量に変換するという 2 段階の手順を取っている．ポイントは，(1) ルックアップテーブルを用いることで，勾配ヒストグラム特徴量の演算を大幅に高速化したこと，および (2)Binary Hashing の変換行列 $\mathbf{P}$ を疎行列にすることで，実数ベクトルから二値ベクトルへの変換を高速化したことにある．CARD のキーポイント検出は，先に紹介した BRISK，ORB とほぼ同じ考え方（ピラミッド画像に対するコーナー検出）に基づいているためここでは説明を省略する．以下，勾配ヒストグラム特徴量の高速抽出方法と，疎行列による Binary Hashing の部分について解説する．

5.2.1　ルックアップテーブルを用いた勾配ヒストグラム特徴量の高速抽出

SIFT と同様に，CARD ではパッチのオリエンテーションを補正した後に，キーポイント周辺領域を K 個のセルに分割し，それぞれのセルから L 次元の勾配ヒストグラム特徴量を抽出するという考え方に基づいている．CARD では特徴量の記述能力をより高めるために，4×4 のグリッド分割ではなく，放射状に分割している（図 18）．これは文献 [21] で提案されている GLOH 特徴量の考え方に倣ったものである．$K = 17, L = 8$ であるから，特徴量は 136 次元である．

まず，SIFT と同じ手続きに基づいてパッチのオリエンテーション α を算出する．ただし，SIFT では勾配ヒストグラムに 2 次の多項式フィッティングを適用することでオリエンテーションを連続値として求めるが，CARD ではこれを行わない．すなわち，$\alpha = 0, 1, \cdots, M-1$ という量子化された値を取るものとする．オリエンテーションが離散値であることを許すことで，テーブル化のテクニックが適用できるようになる．

CARD では，オリエンテーションに従ってパッチの画素配列を回転させるのではなく，パッチの分割パターンの方を回転させることを考える．α は M 通りの離散的な値しか取りえないため，M 通りの回転させた分割パターンを予め用意しておけば良い．図 19 に，$M = 40$ のとき $\alpha = 0, 3$ にそれぞれ対応する分割パターンを示した．分割パターンを用いることで，パッチ周辺の画素 $(x, y)^\top$ がどのセルに属するかを高速に求められる．



(a) $\alpha = 0$　　(b) $\alpha = 3$

図 19　パッチの分割パターンのテーブル

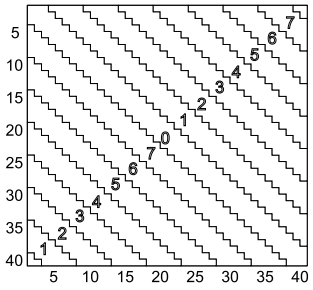


図 20　量子化レベルの変換テーブル

次に各セルから L 次元の勾配ヒストグラムを求めることを考える．ここでのポイントは，オリエンテーションを求めるときに，M 段階に量子化された勾配方向 $\theta(x, y)$ が既に計算済みであるという点である．オリエンテーションを求めるときは比較的細かい量子化間隔を用いるため，$M \gg L$ が成立する．従って，(わずかな誤差を伴うが) M から L への量子化レベルの変換が可能であるから，$\theta(x, y)$ を再利用できる．そこで，量子化レベル数の変換と，オリエンテーションの補正を (30) 式で同時に行う．

$$l = Q_L(Q_M^{-1}(\theta(x, y) - \alpha)) \tag{30}$$

l は小領域内の勾配ヒストグラムの投票位置を示すインデックスであり，$l = 0, \cdots, L-1$ という値を取る．なお，$Q_N(\cdot)$ はラジアンで表現されている角度を 0 から $N-1$ に量子化する関数であり，$Q_N^{-1}(\cdot)$ はその逆関数である．$\theta(x, y) - \alpha$ がオリエンテーションの補正に相当し，$Q_L(Q_M^{-1}(\cdot))$ が量子化レベルの変換に相当する．(30) 式は複雑に見えるが，$\theta(x, y)$ と α がそれぞれ M 通りの値しか取りえないことに着目すると，図 20 に示すように，$M \times M$ のテーブルで表現できることが分かる．従って，図 19，20 に示した 2 種類のテーブルを用いることで，高速に勾配ヒストグラム特徴量を算出できるようになる．

5.2.2　二値特徴量への変換

(31) 式に示す線形の Binary Hashing により実数ベクトルを二値ベクトルに変換する．

$$\mathbf{y} = \mathrm{sgn}(\mathbf{P}\mathbf{x}) \tag{31}$$

ここでは，実数特徴量 $\mathbf{x}$ の平均が原点に位置するようにあらかじめ平行移動されているものとし，しきい値 $\mathbf{t}$ は無視する．

Binary Hashing の課題は，変換速度が遅いという点である．変換行列 $\mathbf{P}$ は B 行 D 列の密な行列であるから，(31) 式の計

算には $B \times D$ 回の掛け算と $B \times (D-1)$ 回の足し算が必要である．

そこで，次の 2 つの条件に基づいて $\mathbf{P}$ を最適化により求めている．

（1） 二値特徴量に変換前の距離 (ベクトル間角度) と変換後の距離（ハミング距離）がなるべく一致するように $\mathbf{P}$ を最適化する．

（2） $\mathbf{P}$ は S 個の非ゼロ要素から成る疎行列であり，各要素が $-1, 0, 1$ のうちいずれかの値のみを取るという制約のもとで $\mathbf{W}$ を最適化する．

条件 (1) により，生成された二値特徴量は元々の特徴量の記述能力を維持できるようになる．また，条件 (2) により，(31) 式による二値特徴量への変換は S 回の足し算のみで行えるようになる．

この最適化のコスト関数は次のように定義される．

$$C(\mathbf{P}) = \sum_{(\mathbf{x},\mathbf{x}')\in\Omega} \left(\frac{d_\theta(\mathbf{x},\mathbf{x}')}{\pi} - \frac{d_h(\mathbf{y},\mathbf{y}')}{B} \right)^2 \qquad (32)$$

Ω は学習サンプルのペア $(\mathbf{x},\mathbf{x}')$ の集合であり，これを二値に変換したものが $(\mathbf{y},\mathbf{y}')$ である．また，$\mathbf{P}$ の i 行 j 列目の要素を p_{ij} とおくと，制約条件は次のように表せる．

$$p_{ij} \in \{-1, 0, +1\}, \qquad \sum_{ij} |p_{ij}| = S \qquad (33)$$

(33) 式の制約のもとで (32) 式を最小化することで変換行列 $\mathbf{P}$ を得る．これは離散最適化問題となるため，原著論文では greedy アルゴリズムを用いて最小化している．

直感的には $\mathbf{P}$ が疎になればなるほど性能が劣化すると考えられるが，興味深いことに $\mathbf{P}$ の要素が 90% 程度ゼロであっても，密な行列の場合と性能がほとんど変わらないことが実験により確かめられている．これは高速化にとって非常に都合が良い性質である．

なお，ビット長 B は目的に応じて任意に決めることができるが，著者らは精度と速度のバランスの観点から 128 ビット程度を推奨している．

5.3 LDAHash

Strecha ら [32] によって提案された LDAHash は，教師あり学習に基づいた手法である．物体のある一点を，異なる 2 つの視点から撮影して得られた局所特徴量のペアの集合を $(\mathbf{x},\mathbf{x}') \in \mathcal{P}$ と定義する．また，物体の異なる 2 点から得られた局所特徴量のペアの集合を $(\mathbf{x},\mathbf{x}') \in \mathcal{N}$ と定義する．このとき，$\mathcal{P}$ に属するペア間のハミング距離を小さく，かつ $\mathcal{N}$ に属するペア間のハミング距離を大きくするような変換行列 $\mathbf{P}$ およびしきい値 $\mathbf{t}$ を選択することを考えることで，より記述能力の高い二値特徴量を得ようというのが LDAHash における考え方である．これは，(34) 式のコスト関数を最小化することで達成できる．

$$L = \alpha E\{d_H(\mathbf{y},\mathbf{y}')|\mathcal{P}\} - E\{d_H(\mathbf{y},\mathbf{y}')|\mathcal{N}\} \qquad (34)$$

d_H は $\mathbf{y}$ と $\mathbf{y}'$ のハミング距離を求める関数である．$E\{\cdot|\mathcal{P}\}$, $E\{\cdot|\mathcal{N}\}$ は $\mathcal{P}$,$\mathcal{N}$ に関する条件付き期待値であり，$\mathcal{P}$,$\mathcal{N}$ それぞれの集合に対する平均値と考えればよい．

ここで，(34) 式の最小化問題の性質について整理する．(34) 式は，次に示す (35)(36) 式の最小化と等価である．

$$L = E\{\mathbf{y}^\top\mathbf{y}'|\mathcal{N}\} - \alpha E\{\mathbf{y}^\top\mathbf{y}'|\mathcal{P}\} \qquad (35)$$

$$L = \alpha E\{||\mathbf{y}-\mathbf{y}'||^2|\mathcal{P}\} - E\{||\mathbf{y}-\mathbf{y}'||^2|\mathcal{N}\} \qquad (36)$$

$\mathbf{y} \in \{-1,+1\}^B$ であるから，(34) 式と (35) 式が等価であることは，ハミング距離 d_H と内積 $\mathbf{y}^\top\mathbf{y}'$ の間に (16) 式の関係が成立することから明らかである．また，(34) 式と (36) 式の関係は，次のように導出できる．まず，ユークリッド距離の二乗の式を次のように展開する．

$$||\mathbf{y}-\mathbf{y}'||^2 = ||\mathbf{y}||^2 - 2\mathbf{y}^\top\mathbf{y}' + ||\mathbf{y}'||^2 \qquad (37)$$

(16) 式を (37) 式に代入して整理すると，(38) 式を得る．

$$||\mathbf{y}-\mathbf{y}'||^2 = 4d_H(\mathbf{y},\mathbf{y}') - 2B + ||\mathbf{y}||^2 + ||\mathbf{y}'||^2 \qquad (38)$$

$\mathbf{y} \in \{-1,+1\}^B$ と定義されていることから，特徴量によらず $||\mathbf{y}||^2 = ||\mathbf{y}'||^2 = B$ であることは明らかである．したがって，最適化に影響を与えるのは右辺の第一項目のみであり，やはり (34) 式と (36) 式の最小化問題は等価であることがわかる．

$\mathbf{P}$, $\mathbf{t}$ に関して (36) 式を直接最小化できれば良いが，実際にはこれは困難である．なぜなら $\mathbf{y},\mathbf{y}'$ の計算には微分不可能な sgn 関数が含まれるからである．そこで，(36) 式から sgn 関数を取り除き，緩和したコスト関数を最小化することを考える．

$$\tilde{L} = \alpha E\{||\mathbf{P}\mathbf{x}-\mathbf{P}\mathbf{x}'||^2|\mathcal{P}\} - E\{||\mathbf{P}\mathbf{x}-\mathbf{P}\mathbf{x}'||^2|\mathcal{N}\} \qquad (39)$$

(39) 式の最小化は，しきい値 $\mathbf{t}$ とは独立になる．そこで，まずは (36) 式を緩和した (39) 式を $\mathbf{P}$ について最小化し，次に $\mathbf{P}$ を固定しつつ (35) 式を最小化することで $\mathbf{t}$ を得るという二段階のステップで最適化を行う．以下，それぞれの手順について述べる．

5.3.1 変換行列 P の選択

Strecha らは，(39) 式の最小化方法として，Linear Discriminant Analysis (LDA) による方法と Difference of Covariances (DIF) による方法の二種類を提案しており，いずれも固有値問題に帰着することで解を得ている．

学習サンプル集合 $\mathcal{P},\mathcal{N}$ それぞれについて，ペアの差分ベクトルの共分散行列 $\mathbf{\Sigma}_\mathcal{P}, \mathbf{\Sigma}_\mathcal{N}$ を次のように定義する．

$$\mathbf{\Sigma}_\mathcal{P} = E\{(\mathbf{x}-\mathbf{x}')(\mathbf{x}-\mathbf{x}')^\top|\mathcal{P}\} \qquad (40)$$

$$\mathbf{\Sigma}_\mathcal{N} = E\{(\mathbf{x}-\mathbf{x}')(\mathbf{x}-\mathbf{x}')^\top|\mathcal{N}\} \qquad (41)$$

このとき，LDA では，二つの共分散行列の比 $\mathbf{\Sigma}_\mathcal{R} = \mathbf{\Sigma}_\mathcal{P}\mathbf{\Sigma}_\mathcal{N}^{-1}$ について固有値計算を行い，DIF では，二つの共分散行列の差 $\mathbf{\Sigma}_\mathcal{D} = \alpha\mathbf{\Sigma}_\mathcal{P} - \mathbf{\Sigma}_\mathcal{N}$ について固有値計算を行う．以下，これらの導出過程について，それぞれ説明する．

Linear Discriminant Analysis (LDA)

$\mathbf{\Sigma}_\mathcal{P}, \mathbf{\Sigma}_\mathcal{N}$ を用いると，(39) 式は次のように変形できる．

$$\tilde{L} = \alpha\mathrm{tr}(\mathbf{P}\mathbf{\Sigma}_\mathcal{P}\mathbf{P}^\top) - \mathrm{tr}(\mathbf{P}\mathbf{\Sigma}_\mathcal{N}\mathbf{P}^\top) \qquad (42)$$

表 1　特徴量の比較

分類	特徴量	学習の有無	記述能力	特徴抽出の速度	特徴量のサイズ	スケール・回転への対応
直接	BRIEF	不要	☆	☆☆☆	☆	サンプリングパターンを回転・拡大縮小することで対応可能
	BRISK	不要	☆☆	☆☆☆	☆	
	ORB	教師なし	☆☆	☆☆☆	☆☆	
	FREAK	教師なし	☆☆	☆☆☆	☆	
	D-BRIEF	教師あり	☆☆☆	☆☆	☆☆☆	事前にパッチを正規化するなど，別途対処が必要
	BinBoost	教師あり	☆☆☆☆	☆	☆☆☆	
	BOLD	教師なし	☆☆☆☆	☆☆☆	☆	サンプリングパターンを回転・拡大縮小することで対応可能
間接	Random projections	不要	☆	☆	☆	変換前の特徴量が既にスケール・回転不変性を持っていれば問題ない
	CARD	教師なし	☆☆	☆☆☆	☆☆	
	LDAHash	教師あり	☆☆☆	☆	☆☆☆	

このとき，$\mathbf{x}$ に $\mathbf{\Sigma}_{\mathcal{N}}^{-1/2}$ を掛けて正規化しておくと，(42) 式の第二項目は定数とみなせる．したがって，(42) 式は次のように整理できる．

$$\tilde{L} \propto \mathrm{tr}(\mathbf{P}\mathbf{\Sigma}_{\mathcal{N}}^{-1/2}\mathbf{\Sigma}_{\mathcal{P}}\mathbf{\Sigma}_{\mathcal{N}}^{-\top/2}\mathbf{P}^{\top}) \tag{43}$$

$$= \mathrm{tr}(\mathbf{P}\mathbf{\Sigma}_{\mathcal{P}}\mathbf{\Sigma}_{\mathcal{N}}^{-1}\mathbf{P}^{\top}) = \mathrm{tr}(\mathbf{P}\mathbf{\Sigma}_{\mathcal{R}}\mathbf{P}^{\top}) \tag{44}$$

$\mathbf{\Sigma}_{\mathcal{R}} = \mathbf{\Sigma}_{\mathcal{P}}\mathbf{\Sigma}_{\mathcal{N}}^{-1}$ であり，これは $\mathbf{\Sigma}_{\mathcal{P}}$ と $\mathbf{\Sigma}_{\mathcal{N}}$ の比に相当する．$\mathrm{tr}(\mathbf{P}\mathbf{\Sigma}_{\mathcal{R}}\mathbf{P}^{\top})$ を最小化するためには，$\mathbf{\Sigma}_{\mathcal{R}}$ の小さい方から B 個の固有値に対応する固有ベクトルで張られる部分空間への射影を考えればよい．従って，著者らは

$$\tilde{\mathbf{S}}_{\mathcal{R}}^{-1/2}\tilde{\mathbf{U}}_{\mathcal{R}}^{\top}\mathbf{\Sigma}_{\mathcal{N}}^{-1/2} \tag{45}$$

という変換を推奨している．$\tilde{\mathbf{S}}_{\mathcal{R}}$ は対角要素に小さい方から B 個の固有値を格納した B 行 B 列の行列であり，$\tilde{\mathbf{U}}_{\mathcal{R}}$ はそれに対応する固有ベクトルを格納した D 行 B 列の行列である．

Difference of Covariances (DIF)

(42) 式は，二つの共分散行列の差 $\mathbf{\Sigma}_{\mathcal{D}} = \alpha\mathbf{\Sigma}_{\mathcal{P}} - \mathbf{\Sigma}_{\mathcal{N}}$ を用いて次のようにも表せる．

$$\tilde{L} = \mathrm{tr}(\mathbf{P}\mathbf{\Sigma}_{\mathcal{D}}\mathbf{P}^{\top}) \tag{46}$$

LDA と同様に，$\mathbf{P}$ が正規直交であることを仮定して $\mathbf{\Sigma}_{\mathcal{D}}$ の固有値分解を行えばよい．

対角要素に小さい方から B 個の固有値を格納した B 行 B 列の行列 $\tilde{\mathbf{S}}_{\mathcal{D}}$ と，それに対応する固有ベクトルを格納した D 行 B 列の行列 $\tilde{\mathbf{U}}_{\mathcal{D}}$ を用いると，変換行列 $\mathbf{P}$ は，(47) 式で与えられる．

$$\mathbf{P} = \tilde{\mathbf{S}}_{\mathcal{D}}^{-1/2}\tilde{\mathbf{U}}_{\mathcal{D}}^{\top} \tag{47}$$

LDA とは異なり，DIF は $\mathcal{P}$ と $\mathcal{N}$ の影響のバランスを決定するパラメータ α に選択の余地がある．$\alpha \to \infty$ の極限においては，$\mathbf{\Sigma}_{\mathcal{N}}$ の影響が無くなるため，これは $\mathbf{\Sigma}_{\mathcal{N}} = \mathbf{I}$ を仮定したことと等価になる．

5.3.2　しきい値 $\mathbf{t}$ の選択

次に，$\mathbf{P}$ を固定して (35) 式を $\mathbf{t}$ に関して最小化することを考える．この最適化問題において，$\mathbf{t}$ の各要素は独立している．したがって，$\mathbf{t}$ の i 番目の要素 t_i の解は，次の最適化問題を解くことで得られる．

$$\min_{t_i} E\ \{\mathrm{sgn}((\mathbf{p}_i^{\top}\mathbf{x} + t_i)^{\top}(\mathbf{p}_i^{\top}\mathbf{x}' + t_i))|\mathcal{N}\} - \alpha E\{\mathrm{sgn}((\mathbf{p}_i^{\top}\mathbf{x} + t_i)^{\top}(\mathbf{p}_i^{\top}\mathbf{x}' + t_i))|\mathcal{P}\} \tag{48}$$

$\mathbf{p}_i^{\top}$ は，$\mathbf{P}$ の i 番目の行ベクトルである．これは 1 変数の最適化問題であるから，t_i を変化させて最適値を探索すればよい．

6. 二値ベクトル表現のまとめ

4., 5. 節で紹介した手法の特徴を表 1 にまとめた．この表では，該当項目における星の数が多いほど性能が良いとしている．この表は各論文における評価結果を参考にし，また著者らの主張を尊重した上でまとめたものであるが，この結果は実験条件や実装によっても異なるであろうし，これらの手法を同一の条件で比較した実験例が示されていないことからも，あくまで全体を理解するための一つの目安として参考されたい．

現在までの動向を見る限り，LDAHash, D-BRIEF, BinBoost といった教師あり学習による手法が最も高い記述能力を持つ二値特徴量を生成できるようである．しかしながら，これらの手法にも問題が無いわけではない．LDAHash は Binary Hashing における変換行列が密であるため，特徴抽出の速度が遅く，リアルタイムなアプリケーションには向かないと考えられる．一方，D-BRIEF では二値特徴量を少ない数のフィルタの線形和から生成するという高速化のための配慮がなされているが，スケールと回転不変性を導入したい場合，パッチを回転させた上で所望の形状にリサイズするなどの前処理が必要である．BinBoost は特徴抽出自体が遅く，1 つの二値特徴量を抽出するのに 1 msec ほどかかることが論文中で報告されている．これは局所特徴量抽出にかかる処理時間としてはかなり遅い方である．教師あり学習による記述能力の向上，回転・スケール不変性の導入，特徴量の高速抽出をどのように同時に達成するかが，今後の課題と言えるかも知れない．

4.,5. 節で紹介してきた手法を今一度振り返ってみると，その全ては (49) 式で一般化できる．

$$\mathbf{y} = \mathrm{sgn}(\mathbf{P}f(\mathbf{x}) + \mathbf{t}) \tag{49}$$

$\mathbf{x} \in \mathbb{R}^{N^2}$ は $N \times N$ ピクセルのサイズを持つパッチの輝度を並

表 2　特徴抽出アルゴリズムの一般化

分類	特徴量	変換行列 $\mathbf{P}$	しきい値 $\mathbf{t}$	特徴抽出 $f(\mathbf{x})$
直接	BRIEF	平滑化したサンプリング点の輝度差	$\mathbf{t}=0$	$f(\mathbf{x})=\mathbf{x}$
	BRISK	平滑化したサンプリング点の輝度差	$\mathbf{t}=0$	$f(\mathbf{x})=\mathbf{x}$
	ORB	平滑化したサンプリング点の輝度差	$\mathbf{t}=0$	$f(\mathbf{x})=\mathbf{x}$
	FREAK	平滑化したサンプリング点の輝度差	$\mathbf{t}=0$	$f(\mathbf{x})=\mathbf{x}$
	D-BRIEF	フィルタの線形和	一次元探索	$f(\mathbf{x})=\mathbf{x}$
	BinBoost	弱識別器の重み	$\mathbf{t}=0$	非線形の弱識別器
	BOLD	平滑化したサンプリング点の輝度差	$\mathbf{t}=0$	$f(\mathbf{x})=\mathbf{x}$
間接	Random projections	ランダム	$\mathbf{t}=0$	従来手法 (SIFT, SURF など) ＋ mean centering
	CARD	$\{-1,0,+1\}$ のいずれかの要素のみを取る疎行列	$\mathbf{t}=0$	LUT による勾配ヒストグラム特徴量抽出＋ mean centering
	LDAHash	LDA もしくは DIF による変換行列	一次元探索	従来手法 (SIFT, SURF など)

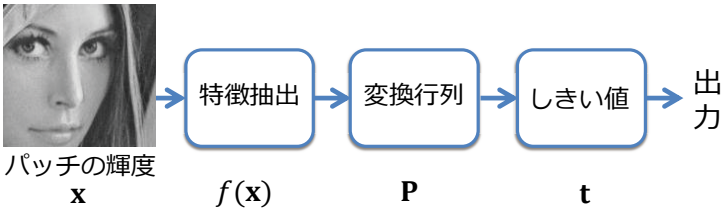


図 21　二値特徴量抽出の流れ図

べたベクトル，$\mathbf{P}\in\mathbb{R}^{B\times D}$ は変換行列，$\mathbf{t}\in\mathbb{R}^{B}$ はしきい値，そして $f(\cdot)$ は $\mathbf{x}$ から何らかの特徴抽出を行う関数であり，

$$f:\mathbb{R}^{N^2}\rightarrow\mathbb{R}^{D} \tag{50}$$

で定義される．どの手法も，特徴抽出，線形変換，しきい値処理の 3 ステップの処理で表現できる (図 21)．各手法の違いは，$\mathbf{P}$, $\mathbf{t}$, $f(\cdot)$ の選び方の違いそのものである．これを表 2 にまとめた．例えば，BRIEF, BRISK, ORB, FREAK, D-BRIEF,BOLD はパッチの輝度に対する線形の演算と符合の判定のみで構成されることから，パッチの輝度 $\mathbf{x}$ に対して直接的に線型の Binary Hashing を適用していると解釈できる．すなわち，$f(\mathbf{x})=\mathbf{x}$ である．一方，BinBoost では，$f(\mathbf{x})$ に D 個の非線形の弱識別器を用い，$\mathbf{P}$ には弱識別器の重みが用いられていると解釈できる．間接的な方法と位置付けた Random projections, CARD, LDAHash では，$f(\mathbf{x})$ は SIFT や SURF のアルゴリズムの演算そのものである．

つまり，ここで紹介したすべての手法は，何らかの特徴変換を施した後に線形の Binary Hashing を適用したものであると，統一的に理解できる．本稿ではこれらの手法を直接的／間接的という二つの観点で分類したが，結局のところ，両者の違いは $f(\mathbf{x})$ にどれだけ複雑な処理を担わせるかの違いであるといえよう．唯一，BinBoost では $f(\mathbf{x})$，$\mathbf{P}$ の両方を Boosting の枠組みで同時に設計する方針をとっており，直接法と間接法の中間に位置しているといえる．

6.1　Binary hashing との関連について

実数特徴量を二値特徴量に変換するという研究は，それはそれで Binary Hashing という一つの分野として幅広く研究されている．本節の締めくくりとして，最後にこれらの研究動向についても軽く触れておく．2007 年に SIGIR の Workshop で Hinton らのグループが RBM を用いた Semantic Hashing [28] を発表し，これを受けて Weiss らが 2008 年の NIPS で Spectral Hashing [38] を発表したころから Binary Hashing の研究は存在感を増してきた．コンピュータビジョンの分野でも研究は進み，2011 年の CVPR で発表された Iterative Quantization [15] は理論が明快で実装がしやすく，性能も良かったことから，Binary Hashing の研究におけるベースラインとして頻繁に利用されている．近年では非線形の方法 [17] や，VLAD および Fisher Vector のような高次元の特徴量を二値に変換する手法 [14]，実数特徴量から二値特徴量への変換を高速化した手法 [29] などが検討されている．これらの研究で示されている知見は，対応点探索のための二値特徴量を設計する上でも参考となるところが多い．

7.　深層学習による手法

2015 年以降の動向として，対応点探索においても，深層学習による特徴量表現が検討されつつある．ここでの主役は Convo-

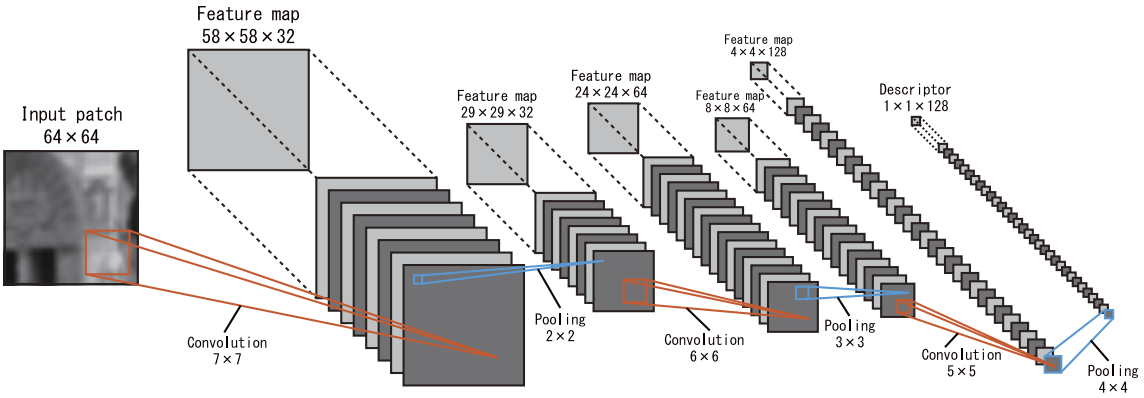


図 22　畳み込みネットワークによる特徴量記述

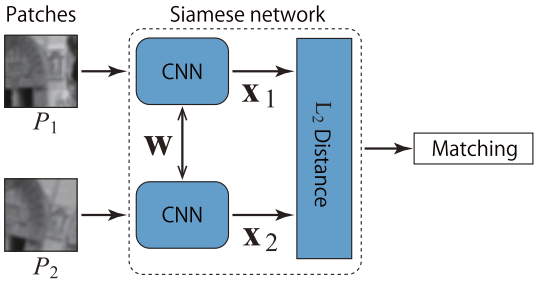


図 23　特徴量記述のための CNN の構成

lutional Neural Network (CNN) である．これは特徴量を抽出する畳み込み層と識別を行う全結合層により構成されるニューラルネットワークであり，今では一般物体認識，文字認識，顔認識，歩行者検出など，多岐にわたるタスクで飛躍的な進歩を遂げている．そこで本節では，特に典型的と思われる二つの研究事例について解説を試みる．

7.1　Discriminative Learning of Deep Convolutional Feature Point Descriptors (ICCV2015)

Simo-Serra らは CNN を用いたパッチ画像の特徴量記述子を提案した [31]．Simo-Serra らによる CNN では入力パッチ画像の特徴量を出力とするため，畳み込み層のみを使用して全結合層は使用しない．畳み込み層は図 22 に示すように 3 層で構成されており，各層で重みフィルタ $\mathbf{w}$ の畳み込み，活性化関数の適用，プーリング，正規化を行う．重みフィルタ $\mathbf{w}$ を畳み込んだ値に活性化関数を適用した結果を特徴マップと呼び，特徴マップに対してプーリングを行うことで特徴量記述に有効な特徴マップを新たに生成する．活性化関数には hyperbolic tangent 関数，プーリングでは L_2 プーリングを用いる．入力パッチ画像が畳み込みネットッワークに入力されると，畳み込みとプーリングを繰り返して最終的に 128 次元の特徴量が得られる．Simo-Serra らによる CNN は Siamese ネットワーク [8] に基づき，図 23 のような構成で特徴量を記述する．図で示す 2 つの CNN は重みフィルタ $\mathbf{w}$ を共有しており，2 つの CNN から出力された特徴量 $\mathbf{x}$ を用いて L_2 ノルムにより距離計算を行う．

学習では，2 つの CNN から出力される特徴量の L_2 ノルムとヒンジ関数を組み合わせたロス関数 $l(\cdot)$ を用いて誤差逆伝播法により畳み込み層のみを学習する．

$$l(P_1, P_2) = \begin{cases} ||\mathbf{x}_1 - \mathbf{x}_2||_2 & \text{positve pair} \\ \max(0, C - ||\mathbf{x}_1 - \mathbf{x}_2||_2) & \text{negative pair} \end{cases} \quad (51)$$

ここで，P は学習パッチ画像，$\mathbf{x}$ は CNN から出力される特徴量，C は任意の定数である．ロス関数 $l(\cdot)$ が最小となるように畳み込み層を学習することで positive ペアの特徴量の L_2 ノルムを小さく，negative ペアの特徴量の L_2 ノルムを大きくするような重みフィルタ $\mathbf{w}$ が得られる．学習画像は Multi-view Stereo Correspondence Dataset (MVS) [2] を使用する．学習された畳み込みネットワークによりパッチ画像の特徴量を記述することができるが，著者らは hard negative mining により，より良いネットワークを学習する方法を提案している．すなわち，ネットワークの順伝播を行った後，ロス関数の出力値が高い negative サンプルである“hard negative”を用いて再度逆伝播を行い，hard negative に適したネットワークに更新する．このとき，ロス関数の出力値が高い positive サンプルである“hard positive”も同様に用いることで，hard positive と hard negative に最適な特徴量を記述する畳み込みネットワークを構築している．

7.2　Learning to Compare Image Patches via Convolutional Neural Networks (CVPR2015)

Zagoruyko らは 2 枚のパッチ画像を CNN に入力して類似度を求める手法を提案した [39]．CNN は，入力された 2 枚のパッチ画像から畳み込み層で特徴量を抽出し，全結合層で 2 枚のパッチ画像間の類似度を出力する．論文では図 24 に示すようにパッチ間の類似度を算出する複数の CNN モデルを提案している．各ネットワークモデルのシアンで示す矩形は畳み込みと活性化関数 (ReLU)，紫で示す矩形はプーリング (max pooling)，黄色で示す矩形は全結合層を表す．全結合層は線形全結合と ReLU の活性化関数からなり，512 個の隠れユニットを持っている．ここでは，5 種類の CNN の構成について述べる．

(a) 2-channel モデル

2-channel モデル (図 24(a)) は，類似度を測る 2 枚のパッチ画像を 2 チャンネルからなる 1 枚の画像としてみなし，これを CNN に入力するモデルである．入力された画像は見かけ上 1 枚の画像として処理されるため，学習の速度が速くなるという利点がある．実行時においても，2 枚のパッチ画像を統合し，2 チャンネルからなる 1 枚の画像を生成してからネットワークに

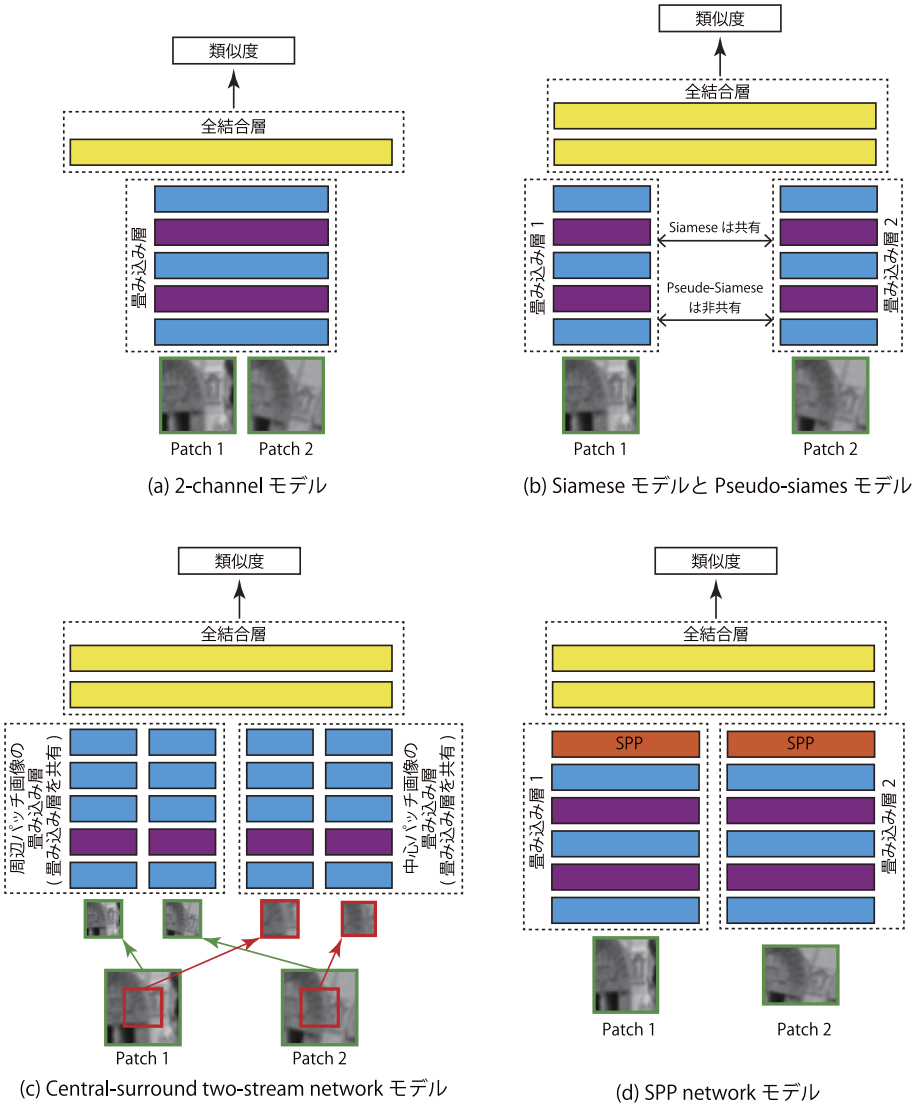


図 24　類似度を算出するための CNN の構成

入力する必要がある．

(b-1) Siamese モデル

Siamese モデル (図 24(b)) は，Siamese ネットワーク [8] に基づき 2 つの畳み込み層を用いたモデルである．このモデルでは，重みフィルタを共有した 2 つの畳み込みネットワークに類似度を求めるパッチ画像を 1 枚ずつ入力する．各畳み込みネットワークの出力は連結して全結合層に与えられ，2 枚のパッチ画像の類似度を算出する．2 枚のパッチ画像は独立で畳み込み層の処理が可能であるため，比較するパッチ画像のペアが変わった場合は片側の畳み込み層の計算だけ行えば良い．

(b-2) Pseudo-siamese モデル

Pseudo-siamese モデル (図 24(b)) は，上記の Siamese モデルと同じ構成であるが，重みフィルタの共有をしないモデルである．そのため学習でのパラメータ数は増加してしまうが，パラメータが制限されている Siamese モデルよりもパッチ画像間の変化に適応したネッワークを構築することができる．

(c) Central-surround two-stream network モデル

Central-surround two-stream network モデル (図 24(c)) では，それぞれのパッチ画像を二つの異なるパッチ画像に分離させて畳み込み層に入力する．すなわち，元となる 64×64 のパッチ画像が与えられたとき，(1) パッチの中心領域を 32×32 画素で切り取った『中心パッチ画像』，(2) パッチ全体を 32×32 画素にダウンサンプリングした『周辺パッチ画像』の二つに分離させる．中心パッチ画像と周辺パッチ画像をそれぞれ畳み込み層に入力することで，パッチの中心部分に特化した特徴とパッチ全体を表現する特徴を両方含んだ特徴量を出力できるため，マッチング精度を向上させることができる．なお，入力される周辺パッチ画像と中心パッチ画像は元の画像サイズの 1/2 になっているため，畳み込み層のプーリングは 1 回のみ行う．図 24(c) では周辺パッチ画像と中心パッチ画像が入力される畳み込み層は Siamese モデルまたは Pseudo-siamese モデルで図示しているが，2-channel モデルを用いることも可能である．

(d) Spatial Pyramid Pooling (SPP) network モデル

上記で述べたネットワークモデルは，サイズ固定のパッチ画像が入力されなければならない．これは，全結合層の入力が所定の次元数でなければならないためである．そのため，キーポイントのスケール領域をリサイズして入力パッチ画像を生成しなければならない．そこで，Spatial Pyramid Pooling (SPP)

network モデル (図 24(d)) では，任意のサイズのパッチ画像を入力できるネットワークを構築している．SPP network モデルは畳み込み層と全結合層の間で Spatial Pyramid Pooling [16] を行うことで，特徴マップの次元を変換させて固定の次元数を持つ特徴量を得ることができる．これにより，入力パッチ画像の解像度を低下させることなく特徴特徴量を抽出することが可能となる．図 24(d) では畳み込み層を Siamese モデルまたは Pseudo-siamese モデルで図示しているが，この手法も 2-channel モデルを用いることが可能である．

以上により，CNN によるパッチ間の類似度を算出するネットワーク構成について記述した．次に CNN の学習方法について記述する．CNN の学習画像は MVS データセット [2] を使用する．このデータセットを用いて誤差逆伝播法により全結合の重みと畳み込み層の重みを学習する．学習には (52) 式に示すのヒンジ関数と L_2 正則化を用いた目的関数を使用する．

$$\min_w \frac{\lambda}{2}||w||_2 + \sum_{i=1}^{N} \max(0, 1 - l_i o_i^{net}) \qquad (52)$$

ここで，w は CNN の重み，o_i^{net} は i 番目の学習画像の CNN の出力，$l_i \in \{-1, +1\}$ は negative ペアと positive ペアのラベルである．λ は正則化の効果を決定する定数であり，論文では $\lambda = 0.0005$ として設定している．ヒンジ関数を用いると，positive ペアでありながら類似度が低いサンプルの出力値を大きくするように重み w を更新する．しかし，ヒンジ関数のみでは重み w が偏ったサンプルに対して過剰に適合してしまうため，正則化項 $\frac{\lambda}{2}||w||_2$ を導入している．(52) 式の目的関数が最小となる w を学習で決定することで，パッチ間の類似度を求めるための最適な CNN を構築することができる．また，学習時は Data Augmentation により学習画像を増加させる．Data Augmentation では各学習パッチ画像に対して，垂直・水平方向の画像反転と 90°，180°，270° の回転処理を施す．Data Augmentation で学習画像を増やすことで画像変換に対してより頑健な CNN を学習することができる．

7.3 深層学習による方法に対する所見

以上，2 つの代表的な手法について紹介した．この種のアプローチは未だ黎明期にあり，動向を体系的に論ずるのは難しいが，特徴抽出と類似度を end-to-end で学習するという考え方は，これから先広く浸透するように思われる．データからよい特徴抽出器が構成できるのであれば，キーポイント検出の時点でスケール推定をするべきなのか，特徴抽出器自体にスケール不変性を持たせるべきなのか，興味深いところでもある．今のところキーポイント検出は独立しているようだが，そもそもここも含めて end-to-end で学習するという考え方が今後出てくるかもしれない．

応用の観点から論じると，例えば異なるセンサー（例えば可視光カメラと赤外光カメラ）間での対応点探索を，Pseudo-siamese モデルでマッチングする，というような方向性がありうる．SLAM のような対応点探索にリアルタイム性を要求するアプリケーションの場合，処理の高速化も必須の検討項目であろう．

```
#include "intrin.h"
int _mm_popcnt_u32 ( unsigned int a);
int _mm_popcnt_u64 ( unsigned __int64 a);
```

図 25　SSE4.2 のビットカウント

```
inline unsigned int popcnt32(unsigned int x)
{
  x = (x & 0x55555555) + ((x>> 1) & 0x55555555);
  x = (x & 0x33333333) + ((x>> 2) & 0x33333333);
  x = (x & 0x0f0f0f0f) + ((x>> 4) & 0x0f0f0f0f);
  x = (x & 0x00ff00ff) + ((x>> 8) & 0x00ff00ff);
  x = (x & 0x0000ffff) + ((x>>16) & 0x0000ffff);
  return x;
}
```

図 26　ビットを数えるアルゴリズム：分割統治法

8. 実装方法／研究用のリソースについて

本節では，研究を進めるにあたり有用と思われる実装方法，研究用のリソースに関する情報を記載する．

8.1 高速なビットカウント

二値特徴量を比較する際に用いるハミング距離は，XOR とビットカウントの組み合わせで計算できる．ハミング距離計算においてボトルネックとなるのはビットカウント処理の部分であるが，この実装方法には様々なアプローチがある．そこで本節では，ビットカウントの実装方法についてまとめる．

Intel の Core i7 など，SSE4.2 に対応しているプロセッサであれば，専用の演算命令を用いることでビットカウントを極めて高速に処理できる．C/C++言語であれば，intrin.h を include することで 32/64 ビット用のビットカウント命令を使えるようになる (図 25) (注7)．ただし，64 ビット用のビットカウント命令_mm_popcnt_u64() は，64 ビットアプリケーションでのみ利用可能であることに注意する必要がある．

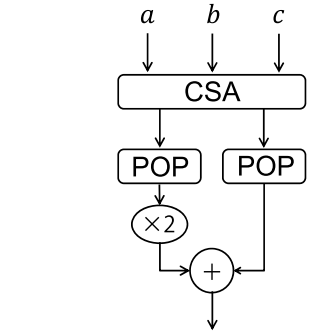
ビットカウントを計算する専用命令が搭載されていない場合でも，基本的な四則演算と論理演算の組み合わせで高速に計算することが可能である．このテクニックについては文献 [24], [37] が詳しい．誰でも簡単に思いつく方法としては，ビット列を 8 ビットずつに分割し，256 通りのビットカウントの結果をルックアップテーブルに保存しておくというアプローチがある．多くの環境では，この実装方法でも十分高速に動作する．また別の方法として，分割統治法によるアルゴリズムが知られている．これを図 26 に示す．興味深いことに，このアルゴリズムではループを必要としない．どちらが速いかは一概には言えず，環境や実装方法によるようである．

ビットカウントをしたいビット列が配列に保存されており，全体のビット数が非常に長い場合については，キャリー保存加算器 (carry-save adder, CSA) を用いると高速化できることがあ

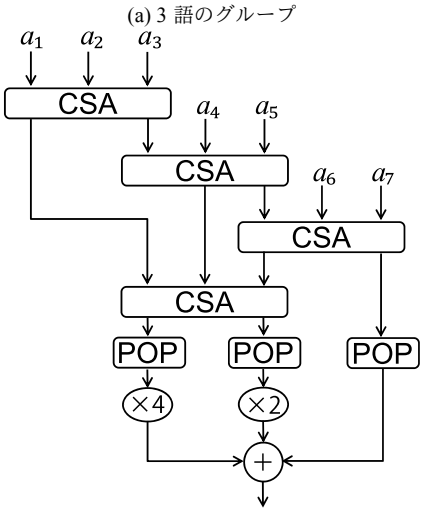
(注7)：GCC ではコンパイル時にオプション (-msse4.2) の指定が必要．

```
#define CSA(h,l,a,b,c) \
  {unsigned u = a ^ b; unsigned v = c; \
    h = (a & b) | (u & v); l = u ^ v;}
```

図 27　キャリー保存加算器 (carry-save adder, CSA)



(a) 3 語のグループ



(b) 7 語のグループ

図 28　キャリー保存加算器によるビットカウント

る [24]．キャリー保存加算器は独立した全加算器が並んだものであり，3 つの入力 a, b, c と 2 つの出力 h, l を持つ．C/C++言語であれば図 27 のようにマクロで定義できる．本来，a, b, c のビットカウントの総和を得るためには，3 回のビットカウントを行わなければならないが，CSA が持つ次の性質を利用すると，1 回の CSA と 2 回のビットカウントに置き換えることができる．

$$\mathrm{pop}(a) + \mathrm{pop}(b) + \mathrm{pop}(c) = 2 \cdot \mathrm{pop}(h) + \mathrm{pop}(\mathrm{l}) \quad (53)$$

$\mathrm{pop}(\cdot)$ はビットを数える関数である．すなわち，a, b, c に関してビットカウントを 3 回行う代わりに，a, b, c に関して CSA を 1 回適用して h, l を求め，l のビットカウント値と h のビットカウント値の 2 倍を足すことで，全く同じ結果が得られる．ビットカウントの演算速度よりも，CSA の演算速度が速い場合は，a, b, c を CSA によって 3 つまとめて処理した方が高速に演算できるわけである．この計算過程は，図 28(a) のように回路図として書きあらわすと分かりやすい．同様の考え方で，CSA の出力を別の CSA に入力し，CSA の回路を多段に構成することで，より多くの数をグルーピングして処理できる．図 28(b) は 7 つをグルーピングした場合の例である．このように，4 回の CSA と 3 回のビットカウントで処理できることがわかる．ビットカウントよりも CSA の方が速い場合は，グルーピングする数が多いほど高速に処理できるようになる．また，多段に組んだ CSA の回路構成において，最終的な出力の一部を最初の入力にフィードバックするというテクニックも高速化に寄与することが知られている．このあたりのテクニックは文献 [24] が詳しい．

このテクニックはビットカウントの専用命令を備えていない CPU で，比較的長いビット長のビットカウントを行いたい場合には利用を検討してもよい．しかしながら，CPU がビットカウントの専用演算命令を備えており，その実行速度が CSA よりもはるかに速い場合はこれらの工夫は無意味であり，素直に専用演算命令のみでビットカウントを行うべきである．

8.2　ソフトウェア／データセット

本稿執筆の時点で公開されているソフトウェアについて，表 3 にまとめておく．画像間の対応点探索性能を評価するためのデータセットとしては，Mikolajczyk ら [21] による Affine Covariant Regions Datasets [1] が最も頻繁に利用されている．これは，ほぼ平面で近似可能な対象を複数の視点から撮影した画像を集めたものである．画像間の Homography 行列が与えられているため，これを用いて真の対応点の位置を求めることができる．Brown ら [9] による Multi-view Stereo Correspondence Dataset [2] では，Difference of Gaussian (DOG) や Harris コーナー検出器によって得たキーポイント周辺のパッチを，スケールとオリエンテーションに関して正規化して得た 64×64 ピクセルの画像データを提供している．パッチ同士がマッチしているか否かのラベル情報も与えられているため，教師あり学習の研究用途に向いている．

9.　おわりに

本稿では，対応点探索のための特徴量表現を **(1) 実数ベクトルで局所特徴量を表現する方法**，**(2) 二値特徴量で局所特徴量を表現する方法**，**(3) 深層学習による方法**，以上 3 つの分類の観点からまとめた．さて，結局のところ，どの手法を用いるのが良いだろうか？筆者らが考えるに，この問いに対する万能の回答は存在しない．その理由は，目的に応じて適切な手法が異なるからである．速度やメモリ消費量が最優先課題であるならば，二値ベクトルによる手法が有望であろう．一方，データが豊富にあり，マッチング時に十分な計算時間を使うことが許されているのであれば，今後は深層学習による方法が有利になるかもしれない．SIFT や SURF はその性能がよく知られており，良い実装も公開されていることから，性能評価のベースラインとして今後も使われ続けるであろう．本稿で示した包括的な視点が，読者の理解の一助となれば幸いである．この分野の今後の発展に期待したい．

文　献

[1] Affine covariant regions datasets. http://www.robots.ox.ac.uk/~vgg/data/data-aff.html.

[2] Multi-view stereo correspondence dataset. http://www.cs.

表 3　利用可能なソフトウェア

特徴量	開発環境	入手先
BRIEF	C++(OpenCV2.0 以上必須)	http://cvlab.epfl.ch/research/detect/brief
BRISK	C++(OpenCV2.2 以上必須), MATLAB	http://www.asl.ethz.ch/people/lestefan/personal/BRISK
	OpenCV	http://opencv.org/
ORB	OpenCV	http://opencv.org/
FREAK	C++(OpenCV 必須)	http://www.ivpe.com/freak.htm
	OpenCV	http://opencv.org/
	MATLAB	MATLAB R2013a Computer Vision System Toolbox 5.2 以降
D-BRIEF	C++(OpenCV2.0 以上必須), MATLAB	http://cvlab.epfl.ch/research/detect/dbrief
BinBoost	C++	http://infoscience.epfl.ch/record/186246?ln=en
Bold	C++(OpenCV 必須)	https://github.com/vbalnt/bold
CARD	MATLAB	https://github.com/DensoITLab/CARD (Denso IT Laboratory, Inc.)
	iOS 用サンプルアプリケーション	アプリ名: CARDesc, AppStore (Apple Inc.)
LDAHash	C++(OpenCV 必須)	http://cvlab.epfl.ch/research/detect/ldahash
CNN Descriptor [Simo-Serra, ICCV15]	不明	https://github.com/etrulls/deepdesc-release (公開予定)
CNN Similarity [Zagoruyko, CVPR15]	Torch, MATLAB	https://github.com/szagoruyko/cvpr15deepcompare

ubc.ca/~mbrown/patchdata/patchdata.html.

[3] Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst. FREAK: Fast retina keypoint. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 510–517, 2012.

[4] Mitsuru Ambai and Yuichi Yoshida. CARD: Compact and real-time descriptors. In *International Conference on Computer Vision (ICCV)*, pp. 97–104, 2011.

[5] Alexandr Andoni and Piotr Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Communications of the ACM*, 2008.

[6] V. Balntas, L. Tang, and K. Mikolajczyk. BOLD - binary online learned descriptor for efficient image matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[7] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, Vol. 110, pp. 346–359, June 2008.

[8] J. Bromley, I. Guyon, Y. Lecun, E. Sackinger, and R. Shah. Signature verification using a “siamese” time delay neural network. In *Neural Information Processing Systems (NIPS)*, 1994.

[9] M. Brown, G. Hua, and S. Winder. Discriminant learning of local image descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 33, pp. 43–57, 2011.

[10] M. Brown, G. Hua, and S. Winder. Discriminative learning of local image descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 33, No. 1, pp. 43–57, 2011.

[11] Matthew Brown, Gang Hua, and Simon Winder. Discriminative learning of local image descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 33, No. 1, pp. 43–57, 2011.

[12] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. BRIEF: Binary robust independent elementary features. In *European Conference on Computer Vision (ECCV)*, pp. 778–792, 2010.

[13] Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, Vol. 42, pp. 1115–1145, November 1995.

[14] Yunchao Gong, Sanjiv Kumar, Henry A. Rowley, and Svetlana Lazebnik. Learning binary codes for high-dimensional data using bilinear projections. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 484–491, 2013.

[15] Yunchao Gong and S. Lazebnik. Iterative quantization: A procrustean approach to learning binary codes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 817–824, 2011.

[16] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 37, No. 9, pp. 1904–1916.

[17] Jae-Pil Heo, Youngwoon Lee, Junfeng He, Shih-Fu Chang, and Sung-Eui Yoon. Spherical hashing. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2957–2964, 2012.

[18] Yan Ke and Rahul Sukthankar. PCA-SIFT: A more distinctive representation for local image descriptors. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 506–513, 2004.

[19] S. Leutenegger, M. Chli, and R.Y. Siegwart. BRISK: Binary robust invariant scalable keypoints. In *International Conference on Computer Vision (ICCV)*, pp. 2548–2555, 2011.

[20] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, Vol. 60, pp. 91–110, November 2004.

[21] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 27, pp. 1615–1630, 2005.

[22] Jean-Michel Morel and Guoshen Yu. ASIFT: A new framework for fully affine invariant image comparison. *SIAM Journal on Imaging Sciences*, Vol. 2, No. 2, pp. 438–469, April 2009.

[23] David Nister and Henrik Stewenius. Scalable recognition with a vocabulary tree. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2161–2168, 2006.

[24] Andy Oram and Greg Wilson. *Beautiful Code: Leading Programmers Explain How They Think*. Oreilly & Associates Inc, 2007.

[25] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[26] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *European Conference on Computer Vision (ECCV)*, pp. 430–443, 2006.

[27] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. ORB: An efficient alternative to SIFT or SURF. In *International Conference on Computer Vision (ICCV)*, pp. 2564–2571, 2011.

[28] R. R. Salakhutdinov and G. E. Hinton. Semantic hashing. In *SIGIR*

workshop on Information Retrieval and applications of Graphical Models, 2007.

[29] Ikuro Sato, Mitsuru Ambai, and Koichiro Suzuki. Sparse isotropic hashing. *IPSJ Transactions on Computer Vision and Applications*, Vol. 5, No. 0, pp. 40–44, 2013.

[30] Gregory Shakhnarovich. Learning task-specific similarity. *Ph.D thesis, Massachusetts Institute of Technology. Dept. of Electrical Engineering and Computer Science*, 2006.

[31] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *International Conference on Computer Vision (ICCV)*, 2015.

[32] C. Strecha, A.M. Bronstein, M.M. Bronstein, and Pee. Fua. LDA-Hash: Improved matching with smaller descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, Vol. 34, No. 1, pp. 66–78, 2012.

[33] Gabriel Takacs, Vijay Chandrasekhar, Sam Tsai, David Chen, Radek Grzeszczuk, and Bernd Girod. Unified real-time tracking and recognition with rotation-invariant fast features. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 934–941, 2010.

[34] T. Trzcinski, M. Christoudias, V. Lepetit, and P. Fua. Boosting binary keypoint descriptors. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2874–2881, 2013.

[35] Tomasz Trzcinski and Vincent Lepetit. Efficient discriminative projections for compact binary descriptors. In *European Conference on Computer Vision (ECCV)*, pp. 228–242, 2012.

[36] Jun Wang, Sanjiv Kumar, and Shih-Fu Chang. Sequential projection learning for hashing with compact codes. In *International Conference on Machine Learning (ICML)*, 2010.

[37] Henry S. Warren. *Hacker's Delight (2nd Edition)*. Addison-Wesley Professional, 2012.

[38] Yair Weiss, Antonio Torralba, and Robert Fergus. Spectral hashing. In *Neural Information Processing Systems (NIPS)*, pp. 1753–1760, 2008.

[39] S. Zagoruyko and N. Komodakis. Learning to compare image patches via convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[40] 安倍満. 対応点探索のための二値特徴量. 情報処理学会研究報告. CVIM, [コンピュータビジョンとイメージメディア], Vol. 2013, No. 19, pp. 1–14, nov 2013.

[41] 藤吉弘亘, 山下隆義, 岡田和典, 前田英作, ノジク・ヴァンソン, 石川尋代, ドゥソルビエ・フランソワ. コンピュータビジョン最先端ガイド 2. アドコム・メディア株式会社, 2010.

[42] 藤吉弘亘, 安倍満. 局所勾配特徴抽出技術. 精密工学会誌, Vol. 77, No. 12, pp. 1109–1116, 2011.