備考

著者

PAUL VIOLA, MICHAEL J. JONES

掲載

International Journal of Computer Vision, Vol. 57, No. 2, pp. 137–154, 2004.

Abstract

本稿では、高い検出率を実現しつつ、画像を極めて高速に処理することが可能な顔検出フレームワークについて述べる。 本論文では、3つの重要な貢献をしています。 1つ目は、「積分画像」と呼ばれる新しい画像表現を導入したことで、検出器で使用する特徴量を非常に高速に計算できるようにしたことです。 2つ目は、AdaBoost学習アルゴリズム（Freund and Schapire, 1995）を使用して構築されたシンプルで効率的な分類器で、非常に多くの潜在的な特徴のセットから少数の重要な視覚特徴を選択することができます。3つ目の貢献は、分類器を「カスケード」で結合する方法で、画像の背景領域を素早く破棄する一方で、有望な顔のような領域により多くの計算量を費やすことができます。顔検出の領域での実験を紹介する。このシステムは、これまでの最高の顔検出システムに匹敵する性能を示した(Sung and Poggio, 1998; Rowley et al., 1998; Schneiderman and Kanade, 2000; Roth et al., 2000)。従来のデスクトップに実装した場合、顔検出は毎秒15フレームで行われる。

Introduction

この論文では、ロバストで非常に高速な視覚検出のためのフレームワークを構築するために、新しいアルゴリズムと見識を結集した。この目的のために、我々は公表されている最良の結果（Sung and Poggio, 1998; Rowley et al.,1998; Osuna et al.,1997a; Schneiderman and Kanade, 2000; Roth et al., 2000）と同等の検出率と誤検出率を達成する正面顔検出システムを構築した。この顔検出システムは、非常に高速に顔を検出できる点で、これまでのアプローチと最も明確に区別されている。384×288ピクセルの画像上で動作し、顔は、従来の700MHzのIntel Pentium III上で毎秒15フレームで検出される。他の顔検出システムでは、動画像の画像差やカラー画像の画素色などの補助情報を用いて高フレームレートを実現していますが、本システムでは、単一のグレースケール画像に存在する情報のみを用いて、高フレームレートを実現しています。また、これらの代替情報源をシステムに統合することで、より高いフレームレートを達成することができます。

私たちの顔検出フレームワークには、大きく分けて3つの貢献があります。以下では、それぞれの考え方を簡単に紹介し、その後のセクションで詳細に説明します。

本論文の最初の貢献は、非常に高速な特徴評価を可能にする積分画像と呼ばれる新しい画像表現である。Papageorgiou et al. (1998)の研究に刺激されて，我々の検出システムは画像強度を直接扱うものではありません．これらの研究者のように、我々はHaar基底関数を連想させる特徴のセットを使用します（ただし、Haarフィルタよりも複雑な関連フィルタも使用します）。これらの特徴を多くのスケールで高速に計算するために、画像の積分画像表現を導入しました（積分画像は、コンピュータグラフィックス（Crow, 1984）でテクスチャマッピングのために使用される和面積表に非常に似ています）。積分画像は、1ピクセルあたり数回の

操作で画像から計算することができます。一度計算されると、これらのHaarに似た特徴のうちの1つは、任意のスケールや場所で一定時間内に計算することができます。

本論文の2番目の貢献は、AdaBoost(Freund and Schapire, 1995)を用いて、潜在的な特徴の膨大なライブラリから少数の重要な特徴を選択することによって構築された、シンプルで効率的な分類器である。画像のサブウィンドウ内では、Haar特徴の総数は非常に多く、ピクセル数よりもはるかに大きいです。高速な分類を確実に行うためには、学習プロセスは利用可能な特徴の大部分を除外し、少数の重要な特徴に焦点を当てなければなりません。Tieu and Viola (2000)の研究に触発されて、特徴の選択はAdaBoost学習アルゴリズムを用いて、弱い分類器が単一の特徴のみに依存するように制約することで達成されます。その結果、新しい弱い分類器を選択するブースティングプロセスの各段階は、特徴選択プロセスとみなすことができます。AdaBoostは，効果的な学習アルゴリズムと一般化性能に強いバウンズを提供する (Schapire et al., 1998).

この論文の3番目の主要な貢献は、画像の有望な領域に注目することで検出器の速度を飛躍的に向上させるカスケード構造の中で、より複雑な分類器を連続的に組み合わせる方法である。注意集中アプローチの背後にある概念は、画像のどこに顔があるかを迅速に判断できることが多いということである(Tsotsos et al., 1995; Itti et al., 1998; Amit and Geman, 1999; Fleuret and Geman, 2001)。より複雑な処理は、これらの有望な領域のためだけに予約されている。このようなアプローチの重要な尺度は、注意処理の「偽陰性」率である。すべて、あるいはほぼすべての顔インスタンスが注目フィルタによって選択されている場合でなければならない。

1 顔検出の注意演算子を学習し、画像の50%以上をフィルタリングして99%の顔を保存することができます（大規模なデータセットで評価された場合）。このフィルタは非常に効率的で、1つの場所/スケール（約60のマイクロプロセッサ命令）あたり20の単純な操作で評価できます。

最初の分類器によって拒否されなかったサブウィンドウは、それぞれが最後の分類器よりも少し複雑な分類器のシーケンスによって処理されます。いずれかの分類器がそのサブウィンドウを拒否した場合，それ以上の処理は行われません．カスケード検出プロセスの構造は，基本的には退化決定木の構造であり，Fleuret and Geman (2001) や Amit and Geman (1999) の研究に関連しています.

完全な顔検出カスケードには38個の分類器があり、合計で80,000回以上の操作が行われています。それにもかかわらず、このカスケード構造は平均検出時間を非常に速くしています。507人の顔と7500万個のサブウィンドウを含む難しいデータセットでは、サブウィンドウあたり平均270個のマイクロプロセッサ命令を使って顔を検出しています。これと比較すると、Rowleyら(1998)が構築した検出システムの実装と比較して、このシステムは約15倍の速さである。

非常に高速な顔検出器は、幅広い実用的なアプリケーションを持つことになります。これには、ユーザーインターフェース、画像データベース、電話会議などが含まれます。この高速化により、従来は不可能であったシステム上でのリアルタイム顔検出アプリケーションが可能になります。高速フレームレートが不要なアプリケーションでは、当社のシステムは大幅な後処理と解析を可能にします。さらに、私たちのシステムは、ハンドヘルドや組み込みプロセッサを含む、広範囲の小型低消費電力デバイスに実装することができます。私たちの研究室では、この顔検出器を、浮動小数点ハードウェアを持たない低消費電力200mipsのStrong Armプロセッサに実装し、毎秒2フレームでの検出を実現しました。

2. Features

我々の顔検出手順では、単純な特徴量の値に基づいて画像を分類しています。特徴量を利用する理由は、2つある.
1つ目は、特徴量が有限の学習データでは学習が困難なアドホックな領域知識を符号化することができるから

です。
2つ目は，特徴ベースのシステムは、ピクセルベースのシステムよりも高速に動作します。

使用する単純な特徴量は、Papageorgiouら(1998)が使用しているHaar基底関数を彷彿とさせるものである。具体的には，3種類の特徴量を用いる．2つの矩形特徴量の値は、2つの矩形領域内の画素の和の差である。2つの領域は同じ大きさと形をしており、水平または垂直に隣接している（図1参照）。3つの矩形特徴量は、2つの外側の矩形内の和を、中央の矩形内の和から差し引いて計算します。最後に、4角形特徴量は、長方形の対角線上のペア間の差を計算します。

検出器の基本分解能が24×24であることを考えると、矩形特徴量の網羅的なセットは16万個と非常に大きくなります。Haar基底とは異なり、矩形特徴量のセットは完全ではないことに注意してください。

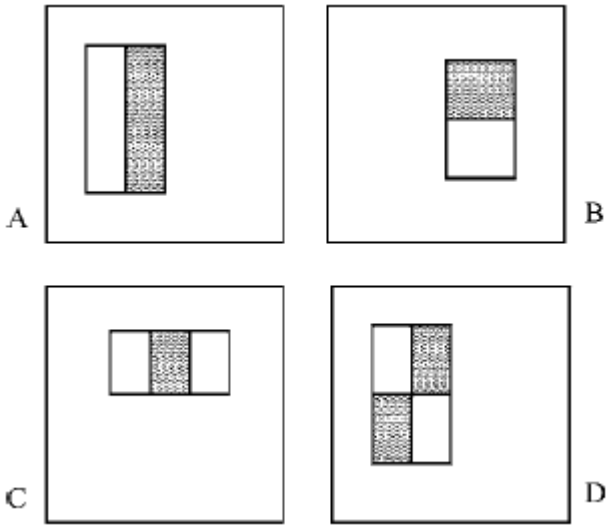

図1: 囲まれた検出ウィンドウに対して示された矩形の特徴の例。白い長方形の中にある画素の合計が灰色の長方形の中にある画素の合計から減算されます。つの長方形の特徴を（A）および（B）に示す。図(C)は3つの矩形特徴量を示し、(D)は4つの矩形特徴量を示す。

2.1. Integral Image

矩形特徴量は，積分画像と呼ばれる画像の中間表現を用いることで，非常に高速に計算することができます．x, y の位置における積分画像は，x, y の上と左のピクセルの合計を含みます．

ii(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y')

ここで、$ii(x, y)$ は積分画像、$i(x, y)$ は原画像である（図 2 参照）。以下の対の再帰を用いて

s(x, y) = s(x, y-1)+i(x, y)

ii(x, y) = ii(x-1, y) + s(x, y)

(ここで$s(x, y)$は累積行和，$s(x, -1) = 0$，$ii(-1, y) = 0$) であり，積分画像は元の画像に対して 1 回のパスで計算することができます．

積分画像を用いて、任意の矩形和を4つの配列参照で計算することができます（図3を参照）。2つの長方形の和の差は明らかに8つの参照で計算できます。上で定義された2つの長方形の特徴は、隣接する長方形のこれらの和は，6つの配列参照で計算され，3角形の特徴量の場合は8つ，4角形の特徴量の場合は9つの配列参照で計算されます.

インテグラルイメージのもう一つの動機は、Simardらの「ボックスレット」の研究から来ています。

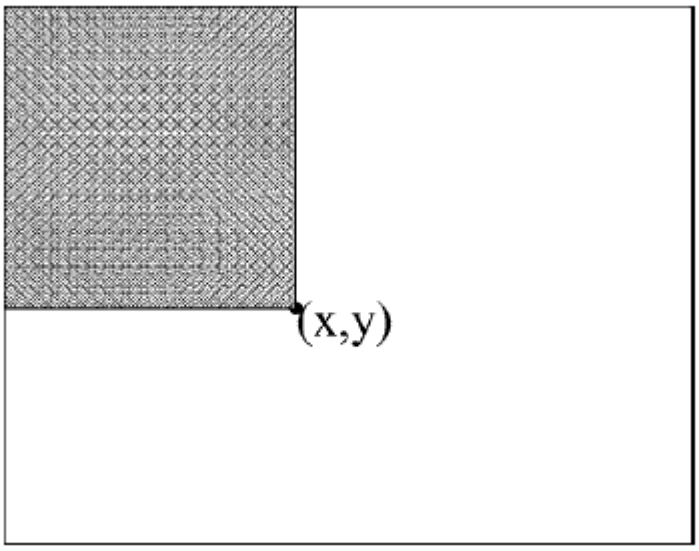

図2: 点(x, y)における積分画像の値は、その上と左にあるすべての画素の総和である。

積分画像のもう一つの動機は、Simardら(1999)の「ボックスレット」の研究から来ている。著者らは、線形演算（例：$f - g$）の場合、その逆数が結果に適用されていれば、どんな反転可能な線形演算でもfまたはgに適用できることを指摘しています。例えば畳み込みの場合，微分演算子を画像とカーネルの両方に適用すると，結果は二重積分されなければなりません.

f \ast g = \int \int \left(f' \ast g' \right)

著者らは、f と g の導関数が疎な場合（または疎な導関数にすることができる場合）には、畳み込みが大幅に高速化されることを示している。同様の洞察は、逆行列の線形演算は、その逆行列が g に適用される場合、f に適用できるということである。

(f") \ast \left(\int \int g \right) = f \ast g

このフレームワークで見ると、矩形の和の計算は、i が画像で、r がボックスカーの画像（関心のある矩形内で値1、外側で値0）である点積、$i \cdot r$ として表現できます。この操作は次のように書き換えることができます.

i \cdot r = \left(\int \int i \right) \cdot r"

積分画像は、実際には画像の二重積分です（最初は行に沿って、次に列に沿って）。矩形の2番目の微分（最初に行に沿って、次に列に沿って）は、矩形の角に4つのデルタ関数をもたらします。2番目のドット積の評価は，4つの配列アクセスで達成されます.

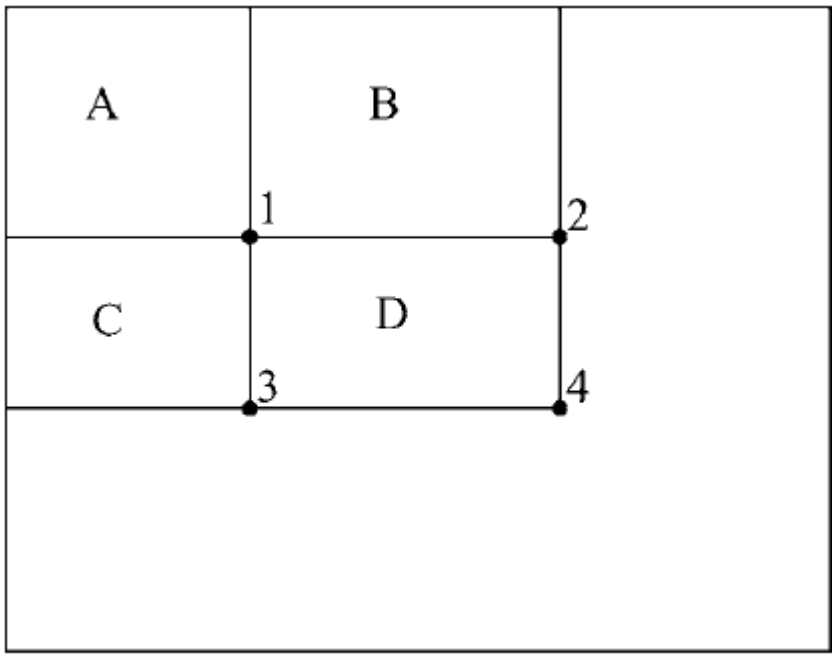

図3: 矩形 D 内のピクセルの合計は、4つの配列参照を用いて計算することができます。位置1の積分画像の値は，矩形 A のピクセルの和です．位置2の値は $A + B$，位置3の値は $A + C$，位置4の値は $A + B + C + D$ です．
