B-002

入退室管理システムにおける人物特定精度向上に向けた顔検出の高速化
Acceleration of facial detection for improvement of person identification accuracy in entering and exiting management system

木綱 啓人† 佐藤 裕幸†
Hiroto Kizuna Hiroyuki Sato

1. はじめに

現代社会ではあらゆる場所に監視カメラが設置されているが，人手による常時監視には膨大な人件費が必要であり現実的でなく，犯罪発生後の犯行証明が主な用途となっている．映像を自動監視する製品では，撮影した映像から動体検出等による不審者を管理者へ通知する機能などが存在する．しかし物体検出は計算量が高く，映像の解像度を下げるか，別途映像監視用サーバにより解析することが考えられるが，これには広帯域な通信回線が必要であり運用コストが高く，耐故障性の観点からも好ましくない．

学校の出席確認や勤怠時間の管理などでは，顔認識技術を用い，入退室に関わる動作なしにカメラの前を通過するだけで在席時間を収集するシステム事例ある[1]．入退室時間は一般に約 1 秒程度と短く，入退室者の撮り逃しを防ぐ点，より高品質な顔画像を用い認証精度を向上する点において，高解像度な画像から高速に顔検出する必要がある．しかし，顔位置を検出するために画像をラスタスキャンする必要があり，顔検出は非常に計算量が高い．

近年プロセッサ性能は，搭載する演算器の数を増やすことで性能を向上させている．特に GPU (Graphics Processing Units) は，単純な計算ができるコアを数千単位で搭載することで CPU と比べ 10 倍以上の理論性能を有する．GPGPU では，この高性能な GPU を数値計算で活用し，これにより，様々な科学技術計算の高速化がなされている．また近年，GPGPU が利用できる小型で軽量，省電力な SoC ボードである NVIDIA Jetson TX2 (以下 TX2) も登場している．

本研究では，組込み機器内で連結成分抽出手法を用いた高速な物体検出アルゴリズムと，GPGPU による並列化実装を行った．TX2 の GPU 上での提案手法の処理時間は，約 14ms であり，従来手法の CPU 実行時と比べ約 60 倍高速化した．これにより，撮影画像をサーバへ転送することなく，組込み機器内で検出警告でき，運用コストを削減しつつも，ネットワークインフラへの依存度を抑えることで耐故障性を高める．

2. 物体検出手法と計算量

物体検出技術とは入力画像中から検出対象物体の座標と領域を推定する手法である．この技術は，物体認識の前処理として用いられることが多く，分類器へ入力する画像を認識したい物体領域に限定することで認識精度を向上させる事ができる．更に認識対象を明確にすることができるため，分類結果の分析が容易になる利点もある．

一般的で具体的な物体検出手順は，(a) 入力画像の前処理，(b) 特徴量抽出，(c) 一致度を用いた全領域探索，(d) 物体領域推定の 4 つである．顔検出精度に大きく影響するのは (b) であり，これには，色特徴，色差によるエッジ特徴などの低次情報を様々な統計手法や機械学習モデルにより，より頑健な特徴抽出を行う．(a)には画像撮影時のノイズ除去やカメラレンズ特有の歪みの補正処理などが挙げられる．(c) は一般に (b) で定義した特徴量を用い，顔らしさや特定の物体らしさをモデリングする．最後の (d) は，画像全体に対して検出対象物体との一致度を計算する必要があり，計算量が高く，膨大な時間がかかる．

†岩手県立大学大学院 Iwate prefectural Univ.

2.1 一般的な顔検出技術の予備評価

ここでは，高解像度な画像から顔検出を高速化する必要性を示す．予備評価では，著者らが過去に実装した顔検出技術[1]における画像解像度と顔検出処理時間の変化と画像解像度と顔検出精度の関係の 2 種類を評価した．

初めに入力画像サイズと顔検出時間の関係を表 1 に示す．顔検出機能の実装には，画像処理ライブラリである Dlib を用い HOG 特徴量抽出及び線形 SVM により物体を検出する．検証環境には前述の TX2 と安価な SoC ボードとして広く利用されている RaspberryPi3 を用いた．FHD 画像を入力にした TX2 上での顔検出時間は約 1 秒であり，リアルタイム処理にはほど遠いことがわかった．

次に，検出可能な顔画像の解像度について検証した．その結果を表 2 に示す．検証データセットには，必ず 1 人以上の顔が写り込んだ 812 枚の画像を用いた．この画像を顔画像が任意の解像度になるようにリサイズし，正規化した．顔画像解像度の低下に連れ，顔検出精度も低下し，顔画像サイズの一辺が 64pix を下回ると顔検出精度が著しく低下した．このことから，遠くに写る顔画像を検出する点，顔認識においてシワ等の特徴抽出できる高精細画像を撮影する点に於いて，高解像度な入力画像から高速に顔検出を実現することが重要である．

表 1 画像解像度と顔検出時間 (ms) の変化

画像サイズ(pix)	Jetson TX2	RaspberryPi3
360x180	25.3	969.3
640x360	103.2	4,267.2
1,280x720	385.4	17,555.9
1,920x1,080	834.4	40,818.9

表 2 画像解像度と顔検出精度の関係

顔画像サイズ (pix)	顔検出精度(%)
256x256	91.0
128x128	89.3
64x64	67.6
32x32	0.0

3. 提案手法

高速な物体検出に用いる連結成分抽出手法とその GPGPU 向けラベル更新処理の並列アルゴリズムと高速な顔検出を実現するために画像処理手法を提案する．

3.1 並列ラベル書き換えアルゴリズム

連結成分抽出とは，入力 2 値画像の画素値が連続している画素群を連結成分として抽出し，ID を付加する処理である．この処理は昔から多くの研究がなされており，大きくラスタ型と伝播型連結成分抽出が存在する．本稿では GPGPU 等の高い並列性を活かせる伝播型[2]を用いる．この伝播型連結成分抽出手法は，各画素に固有の配列インデックスをラベル ID として格納し，近傍探索により最小のラベル ID を成分内で伝搬させる．最終ラベル画像は図 1(a)のように成分で最小の配列インデックスがその成分のラベル ID になる．しかし，このままでは面積算出時に不都合があるので連続ラベル ID に書き換える．我々は，画素単位の並列性を持つ GPGPU 向け連続ラベル ID への書き換え処理を考案した．その手法を示す．

各成分ラベル ID と新たな連続ラベル ID との対応を記録するための LUT (Look Up Table) を用いる．ユニークなラベル ID を得るためにまず，各成分で代表スレッドを選出する．各スレッドが担当画素のラベル ID を使い LUT を参照し，当該ラベル ID が未更新であれば，LUT からアトミック演算により 1 を引く．最初にアトミック演算したスレッドが得た結果が-1 であれば，そのスレッドが代表スレッドとして選出される．代表スレッドはユニークなラベル ID を取得し，LUT へ格納する．最後この LUT を用いてラベル画像を書き換えることで，図 1(b)のように連続ラベル ID へ更新できる．

3.2 ラベリングを用いた入力画像探索領域の削減

具体的な物体領域推定手順は，HSV 色空間から閾値により人肌か否かの 2 値画像を生成する．次に，拡大・縮小処理により細かいノイズを除去し，比較的大きなノイズ領域 (図 2 の 3 の左下参照) を除去するために，前節で説明した連結成分抽出した結果を用いる．成分の画素数が規定値以下であれば，その成分をノイズとして破棄する．最終的に残った連結成分を顔候補領域として抽出する (図 2 の 4 の矩形)．この候補領域に対して高精度な顔検出をすることで全体の処理時間を削減出来る．またこの手法は，色特徴以外にも動体検出など 2 値の特徴量画像が抽出可能であれば様々な用途に応用可能である．

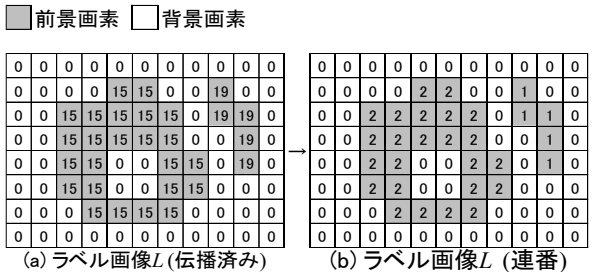

図 1 連結成分抽出と連続ラベル ID への書き換え

図 2 提案する顔検出の処理手順

4. 検証評価

評価では，本提案手法による顔検出速度の高速化率と GPGPU 実装時の高速化率を検証した．GPGPU 実装のために，NVIDIA 社が提供する開発環境 CUDA を用いた．

3 章で述べた，提案手法について CPU 実装での処理時間を表 3 に示す．尚，CPU 上ではラスタ型連結成分抽出の方が高速だったため，そちらを採用した．TX2 の CPU 上で FHD 画像を入力にした提案手法の処理時間は約 60ms であり，従来手法と比べ約 14 倍程度の高速化を実現した．また RaspberryPi3 では，極めて高い高速化率を実現し，安価な SoC ボードでも有用な手法であることがわかった．

次に 3 章で述べた 2 つの提案手法を GPU 向けに実装し，その処理時間と提案手法の CPU 実装からの高速化率を表 4 に示す． FHD 画像での顔領域推定時間は約 14 ms であり，約 71.4 fps を実現した．FHD 画像において提案手法 CPU 実行時より GPU 実行時のほうが約 4 倍高速だった．

表 3 顔領域推定の CPU 処理時間 (ms) と高速化率

画像サイズ (pix)	Jetson TX2		RaspberryPi3	
	処理時間	高速化率	処理時間	高速化率
864x486	11.8	14.4	67.7	118.2
1,280x720	24.8	15.5	151.1	116.2
1,920x1,080	56.4	14.8	333.9	122.2

表 4 提案手法の GPU 処理時間 (ms) と高速化率

画像サイズ (pix)	Jetson TX2	
	処理時間	CPU/GPU 高速化率
864x486	4.9	2.4
1,280x720	7.7	3.2
1,920x1,080	14.0	4.0

5. おわりに

本稿では，比較的単純な画像処理手法を組み合わせることで，一般的な顔検出の欠点である高い計算量を低減した．また並列計算機向けに連番ラベル ID への書き換え方式を考案した． TX2 の CPU 上で顔検出した場合，従来手法と比べ提案手法は約 14 倍高速であった．また，約 17.7fps のフレームレートを実現し，十分なリアルタイム性を実現した．更に TX2 の GPU 上での実行時間は FHD 画像に対し約 14ms であり，約 71.4fps のフレームレートを実現した．従来手法 CPU 実行時と比べ約 60 倍高速であった．安価な SoC ボードである RaspberryPi3 の CPU 実装時でも FHD 画像を約 334ms で処理を完了しており，低速な移動物体の領域推定用途において実用的な性能を確認した．

今後の展望として，顔検出だけではなく動体検出技術と組み合わせることで監視カメラ等への応用を検討する．

参考文献

[1] 木綱 啓人, 比嘉 優樹, 佐藤 裕幸, ”Deep-CNN を用いた人物判定による入退室管理システム,” 電子情報通信処理学会技術報告, Vol.116, No.259, PRMU2016-97, pp.37-42 (2016).
[2] Naoki Shibata, Shinya Yamamoto: GPGPU-Assisted Subpixel Tracking Method for Fiducial Markers, Journal of Information Processing, Vol.22, No.1, pp.19-28 (2014).