Vol.2010-IS-114 No.4
2010/12/1

ActionScript3.0 を用いたストリーミング動画像処理による顔発見と隠蔽

田 中 亮 太[†1]　小 枝 正 直[†1]

近年，YouTube など Flash を用いた動画共有サイトが多く存在しており，様々な動画がアップロードされている．その中には，許可を得ていない他人の顔などが映った動画もアップロードされており，個人情報の漏洩やプライバシーの侵害などの危険性がある．そこで本研究では，ActionScript3.0 を用いた疑似ストリーミング動画像に対する顔発見と隠蔽処理を行い，その実現可能性を検証した．

Face detection and hiding using ActionScript3.0 for streaming video on the Internet

Ryouta Tanaka[†1] and Masanao Koeda[†1]

Recently, video streaming and video sharing using Flash technology such as YouTube are increasingly prevalent in the Internet. However, most of the streaming video are created without any permission and it is deeply concerned about the intrusion of privacy and the leaking of personal information. Such a background, we are targeting the protection of the face privacy in the streaming videos. In this research, face detection and hiding system for flash by using ActionScript3.0 was developed, and we conducted feasibility experiments.

1. はじめに

近年，YouTube, Google Video, Live Leak, Veoh, ニコニコ動画など多くの動画共有サービスが普及し，多くの人々が利用している．日本での動画共有サービスの認知率は 9 割以上，利用経験率も 7 割以上[1] と高くインターネットユーザの一般的な情報メディアとなっている．動画共有サービスが普及した要因は 3 つ挙げられる．まず 1 つは，高速大容量通信が整備されたことである．日本では 2000 年頃，定額安価で常時接続可能な通信サービスが一般家庭に普及し始めた．２つめは，Adobe Flash Video(FLV,F4V) の登場である．Flash Video は FlashPlayer6 以降を利用してインターネット上で動画配信するために利用されるコンテナ型のファイルフォーマットである．Flash Video が普及するまでの HTML 上における動画表現は Windows Media Video や QuickTime ムービーが主流であったが，ユーザはファイルのコーデックのたびにプラグインやプレイヤのインストールが必要だった．しかし，Flash Video が登場してから web の表現方法が革新される要因になった．３つめは，新しい表現の方法が増えたということである．テキストと静止画ベースだったインターネットのインフラが動画ベースへと変わり，世界中の人と繋がることができるようになった．今までのようにニュースでしか知り得なかった世界の情勢も動画共有サービスによって，リアルな情報が得られるようになった．

しかし，動画共有サービスの普及により自分や他人の顔を出している動画も大量にアップロードされるようになった．顔を出すことによって，その人の人柄や，その瞬間の感情を読み取ることができるという利点もあるが，不特定多数が利用するインターネットの世界において顔を出すということは，その人の情報を世間に露見させる行為であり，その後の生活に不測の事態を招く危険性を含んでいる．また，一度インターネット上にアップロードされた情報が複製・転載・加工され，被写体の望まない形で上映・流布される可能性もある．

そこで，アップロードされた映像に対して顔隠蔽処理が必要であると考えた．過去の研究では，監視カメラ映像に対するプライバシ保護の研究[2)3)] や組織内での公共スペース内の状況映像に対するプライバシ保護に対する研究[4)5)] がある本研究では Adobe Flash Video を採用している動画共有サービスに対して ActionScript3.0 を用いて顔隠蔽処理を行い，その人の情報を世間に露見させることを防ぐという手法を試みる．

2. システム概要

2.1 ActionScript について

本実験で使用する言語について説明を行う．ActionScript とは，ECMA-262 仕様に準拠する Flash Player 用のプログラム言語である．ActionScript のバージョンは現在 1.0，2.0，3.0 がある．また，どのバージョンの ActionScript のプログラムが実行できるかは，実行環境の Flash Player に依存する．Flash Player では下位互換性があり，Flash Player9 以降の

†1 大阪電気通信大学 総合情報学部 メディアコンピュータシステム学科
Osaka Electro-Communication University, Faculty of Information Science and Arts

環境では，ActionScript1.0〜3.0 までの任意のバージョンを動作させることが可能である．ただし，Flash Player 8 以前では ActionScript3.0 のコンテンツを動作させることは不可能である．ActionScript の，プログラムが埋め込まれた Flash Video はプラットホームに依存しない．ActionScript で記述されたプログラムをコンパイルすると，拡張子.swf というファイルが生成される．この swf ファイルは，FlashPlayer の中にある AVM(Actionscript Virtual Machine) 上で動作する．したがって，Windows 用や Mac 用などの AVM が組み込まれた Flash Plyaer を使用することによって，どのプラットホームでも ActionScript を使用した Flash Video が再生することが可能になる．AVM には AVM1 と AVM2 がある．AVM2 は，ActionScript3.0 を処理する専用の仮想マシンで，ActionScript1.0 や 2.0 は動作しない．しかし，FlashPlayer9 以降には AVM1 と AVM2 が併せて搭載されており動作するようになっている．AVM2 になって処理が最適化され，計算速度が向上している．しかし，アニメーションあるいはテキストの描画，サウンドの制御，サーバのネットワークのやり取りなどは，Flash Player 本体で扱うため PC の性能に依存する．

2.2 ストリーミングについて

ストリーミングとはインターネットなどネットワークを通して映像や音声などのマルチメディアデータを視聴する際に，データを受信しながら同時に再生を行う方式である．ストリーミングには 2 種類の配信方式があり，疑似ストリーミング（プログレッシブ・ダウンロード方式）とリアルタイムストリーミングである．

疑似ストリーミングは，web サーバにアップロードされた動画ファイルをローカル記憶領域にダウンロードしながら再生をする方式である．HTTP で転送することが可能であるため，YouTube など多くの動画共有サービスに利用されている．しかし，ローカル記憶領域にデータが残ってしまうことが問題視されている．

リアルタイムストリーミングはダウンロードした動画ファイルを破棄しながら再生する方式であり，リアルタイムストリーミングを行うことができる専用サーバが必要である．リアルタイムストリーミングではローカル記憶領域にデータが残らないため，著作物の配信などによく使われている．また長時間の配信の場合，疑似ストリーミングで配信すると保存される動画データが膨大になるため，ライブイベントの生放送，プレゼンテーション，研修用ビデオなどにも利用されている．

3. 処理の流れ

本研究で行う処理のフローチャートを図 1 に記載する．

3.1 疑似ストリーミング再生

疑似ストリーミング再生から動画像処理を行えるようにするまでの手順について説明する．

(1) ネットワークのコネクションを確立するために NetConnection オブジェクトを生成する．

(2) NetConnection オブジェクトの connect() メソッド[6] を null を引数に実行しコネク

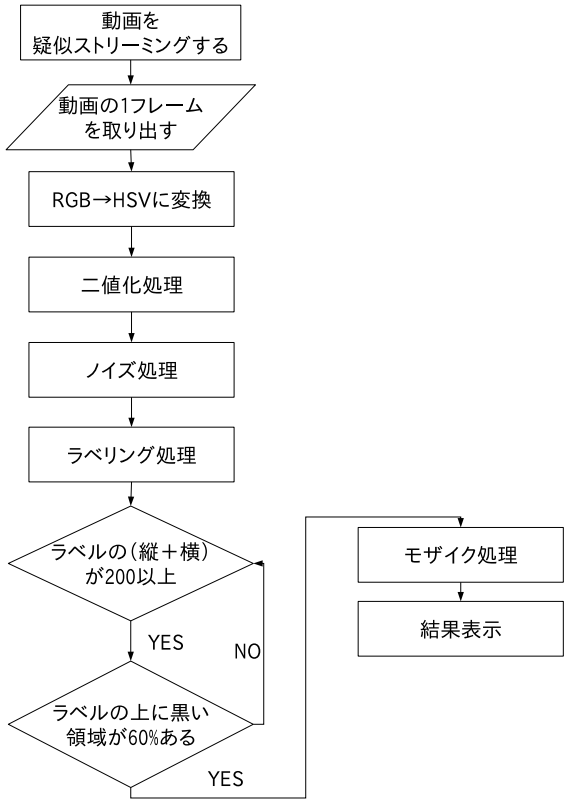


図 1 処理の流れ
Fig. 1 Flowchart

ションを生成する．

(3) ストリーミング接続を行う NetStream クラスのコンストラクタをあらかじめ生成した NetConnection オブジェクトを引数に実行し，NetStream オブジェクトを生成する．

(4) Video コンストラクタをビデオの幅と高さを引数に実行し，Video オブジェクトを生成する．

(5) Video オブジェクトの attachNetStream() メソッド[6] を NetStream オブジェクトを引数に実行する．

(6) 参照しているビデオファイルを引数とし NetStream オブジェクトの Play（）メソッドを実行して再生を開始する．

(7) addChild() メソッド[6] を使用して Video オブジェクトを表示コンテナに追加する．

(8) RGB のピクセル単位で処理を行うために，毎フレームのイベント処理を Video オブジェクトに追加し，生成した BitmapData オブジェクトに draw() する．

(9) draw() された BitmapData オブジェクトに動画像処理を行う．

3.2 RGB → HSV

以下の式 1 を用いて，HSV 変換を行う．

$$H = \begin{cases} 60 * \frac{G-B}{MAX-MIN} + 0 & if \quad MAX = R \\ 60 * \frac{B-R}{MAX-MIN} + 120 & if \quad MAX = G \\ 60 * \frac{R-G}{MAX-MIN} + 240 & if \quad MAX = B \end{cases} \tag{1}$$

さらに

$$H += 360 \quad if \quad H < 0 \tag{2}$$

$$S = \frac{MAX - MIN}{MAX}$$
$$V = MAX$$

3.3 二 値 化

HSV 変換した H に条件式を与え，条件式に当てはまるピクセルを白に，当てはまらなければ黒に塗りつぶす．条件の H 範囲は赤から黄色 (10〜60) までの色を白にしている．

3.4 ノイズ処理

二値化した画像は，まだ細かいノイズが残っているので，この細かいノイズを除去するために膨張・収縮をかける．ノイズ処理には，畳み込みフィルタで全てのピクセルにフィルタをかけ全ピクセルの値を 0〜9 に変える．その後に ActionScript3.0 の threshold メソッド[6] を使い，ピクセルの値が 1 以上なら白で塗る，1 未満なら黒で塗るという膨張と，ピクセルの値が 9 以上なら白，9 未満なら黒で塗る収縮をそれぞれ一回ずつかける．

表 1 ConvolutionFilter で使用する行列
Table 1 matrix used in ConvolutionFilter

1	1	1
1	1	1
1	1	1

畳み込みフィルタとは，入力された動画像 f のすべてのピクセル $(f(i,j))$ に対して，U 行× V 列を持つフィルタ W を適用するというフィルタでこのフィルタを適用すると，全てのピクセル $(f(i,j))$ の色を変化させる．変化させた後のピクセルを $(g(i,j))$ とする．その式を式 3 に示す．

$$g(i,j) = \sum_{x=0}^{U-1} \sum_{y=0}^{V-1} f(i+x-U/255, j+y-V/255) * w(x,y) \tag{3}$$

なお

$w(x,y)$

で使用する行列の値を表 1 に示す．

本研究で行ったノイズ処理は $U = 3, V = 3$ の 3 行 3 列を全てのピクセルに乗算してやり，出できた 9 つの値の和をとり，255 で除算を行って値を 0〜9 の範囲で出力する．

3.5 ラベリング処理

2 値画像の白色ピクセルに対して ActionScript3.0 にある floodfill メソッド[6] を使い，ラベルを設定する．floodfill は (x,y) 座標を始点として所定の色で塗りつぶすメソッドである．

3.6 顔 抽 出

ラベリング処理でラベル番号がついたラベルに 2 つの条件を与え，その条件を 2 つとも満たしたラベルを顔とみなしている．その条件について 1 つめは，ラベルの大きさである．ラベルの高さ，幅の和を求め，その和が 200 以上なら顔であるとみなしている．2 つ目は，ラベル上の黒い領域の有無である．映っている人の顔の髪の色が黒であるという前提で，ラベルの開始位置の $(x+20, y-10)$ の領域のピクセルを見て，黒いピクセルが 60%以上有るなら顔とみなしている．顔と判別したラベルの (x,y) 座標，高さ，幅の値をモザイク処理に渡す．

表 2 使用した動画の情報
Table 2 Video information

動画サイズ	320 × 240[pixel]
動画の長さ	49[sec]
動画の fps	30[fps]

表 3 使用したカメラ，PC のスペック
Table 3 Camera, PC's spec

OS	Microsoft Windows XP Professional SP3
CPU	Intel Core i7 860 @ 2.80[GHz]
メモリ	3[GB]
Flash Player	Var.10,0,42,34
カメラ	Logicool 2-MP Portable Webcam C905m
swf のサイズ	138.7 [Kbyte]

3.7 モザイク処理

顔抽出後に渡されたラベルの (x, y) 座標，幅，高さの値で x と y が 10 ずつ増えるループ文を作り，その時の (x, y) 座標のピクセルの色を (x, y) を始点として 10*10 の範囲で塗りつぶす．それを渡されたラベルの幅，高さの範囲内で繰り返す．

4. 実　　験

今回の研究では，まず動画処理を行う swf ファイルが埋め込まれた HTML を作成する．その HTML を開き，ブラウザ上にローカル内で実験動画を参照し，疑似ストリーミングを行う．その疑似ストリーミング動画に対して，顔発見，隠蔽処理を行いながら，表示させる，という実験を行う．使用する動画は，誰も映ってない状態から，人が 1 人，2 人，3 人と映り込む動画を使用した．使用した動画の情報を表 2 に示す．また，使用したカメラ，PC のスペックを表 3 に示す．

4.1 実 験 方 法

評価方法として，以下の 2 つを挙げる．

- 処理精度
- フレーム数

隠蔽処理を行う際に，隠蔽処理に失敗していた場合に，個人情報が漏洩してしまう．また，顔で無い箇所に隠蔽処理が行われている場合，その動画の情報量が減ってしまうため，処理の精度を調べる必要がある．さらに，本隠蔽処理により計算機への負荷が高まった場合フレームレートが低下してしまい，動画を見ているユーザはストレスを感じるので，フレーム数を調べる必要がある．

4.2 実 験 結 果

本手法を評価するために，自分で撮影をした動画に対して処理を行った．その実験の処理の様子を図 2，図 3 に示す．これらの処理結果の様子から動画に対して顔発見と隠蔽処理が行われ，個人の特定を避け，プライバシー保護が実現できていることが分かる．

また，顔発見を行う前のラベル数と経過時間 [sec] を図 4 に，顔発見を行ったあとの経過時間 [sec]，ラベル数を図 5 に，処理を行っている時の経過時間とそのときのフレームレートの変化をグラフを図 6 に示す．これらの結果から，顔発見処理を行っていないラベル数と顔発見処理を行ったラベル数を比べた時，発見処理を行ったラベル数は映像中に映っている人数の数になっていることが分かる．そして，顔発見処理を行っていないラベル数が多い程，フレームレートの低下が見られる．

今回の実験で以下の問題が確認された．

- 顔発見について，ラベルの高さ，幅の和が 200 以上とラベルの上に黒いピクセルが 60%以上の 2 つの条件を満たすラベルを顔と判別しているのでまだ不正確なところがある
- 処理速度が遅いため，音声が途切れてしまっていた．
- 顔を出してもいい芸能関係などの人も処理対象になってしまうこと．

5. お わ り に

本研究では，ActionScript3.0 を用いたストリーミング動画像処理による顔発見と隠蔽を行う方法を試みた．実験結果から，顔発見と隠蔽処理を適応し，3 人の顔に隠蔽処理を行うことが可能であることを示した．

今後の課題として以下のことが挙げられる．

- ローカル記憶領域内に処理されてない動画データが残ってしまう
- Flash Video 以外のストリーミング動画に対する隠蔽処理
- 電話番号，住所，ナンバープレートなどの文字の隠蔽処理

参 考 文 献

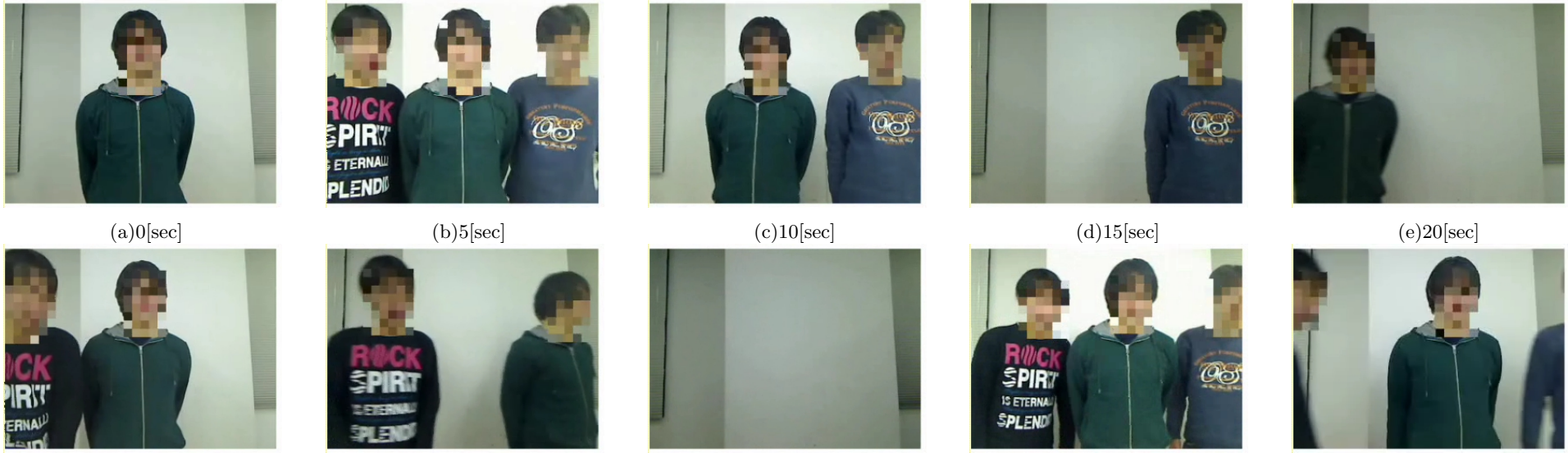
1) 「情報メディア」に関する調査 (第 5 回)-Yahoo!リサーチのヤフー・バリュー・インサイト http://www.yahoo-vi.co.jp/research/100112.html
2) 難場　仁，小谷　周平，北浦嘉浩，他：顔領域に基づく JPEG2000 の ROI 機能を利用

(a)0[sec] (b)5[sec] (c)10[sec] (d)15[sec] (e)20[sec]

(f)25[sec] (g)30[sec] (h)35[sec] (i)40[sec] (j)45[sec]

図 2 元画像
Fig. 2 Original image

したプライバシー保護の実現，電子情報通信学会技術研究報告， Vol.108, No.86, pp. 27—32 (2008.6) 1235–1244 (1990).
3) Mitsuji MUNEYASU, Shuhei ODANI, Yoshihiro KITAURA and Hitoshi NAMBA: An Implementation of Privacy Protection for a Surveillance Camera Using ROI Coding of JPEG2000 with Face Detection, *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, Vol.E99, No.11, pp.2858–2861 (2009).
4) タンスリヤボン　スリヨン，千葉　正広，花木　真一: 状況映像における顔認識を用いた人物隠蔽，映像情報メディア学会技術報告，Vol.24, No.49, pp.1–6 (2000.9).
5) タンスリヤボン　スリヨン，千葉　正広，花木　真一: 状況映像における顔認識を用いた選択的人物隠蔽，情報処理学会論文誌，Vol.56, No.12, pp.1980–1988 (2002).
6) Adobe Flex 3.2 リファレンスガイドhttp://livedocs.adobe.com/flex3_jp/langref/index.html.
7) 西田義人，田中成典 ，馬石尚登 ，坂田能一 ，木本直樹:プライバシー保護を考慮した映像マスキングに関する研究， 全国大会講演論文集， Vol.71, No.2, pp.449–450 (2009.3).



(a)0[sec] (b)5[sec] (c)10[sec] (d)15[sec] (e)20[sec]

(f)25[sec] (g)30[sec] (h)35[sec] (i)40[sec] (j)45[sec]

図 3 処理結果
Fig. 3 Processing Results

Vol.2010-IS-114 No.4
2010/12/1

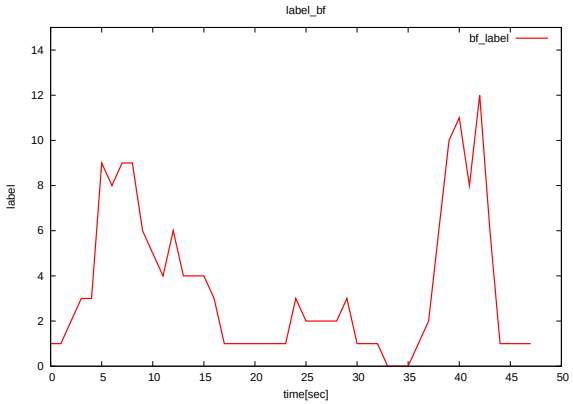


図 4 実験:顔発見処理前のラベル数
Fig. 4 Experiment:Number of labels before Face detection processing

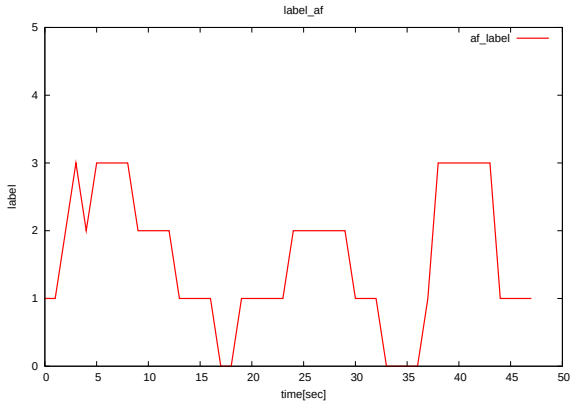


図 5 実験:顔発見処理後のラベル数
Fig. 5 Experiment:Number of labels before Face detection processing

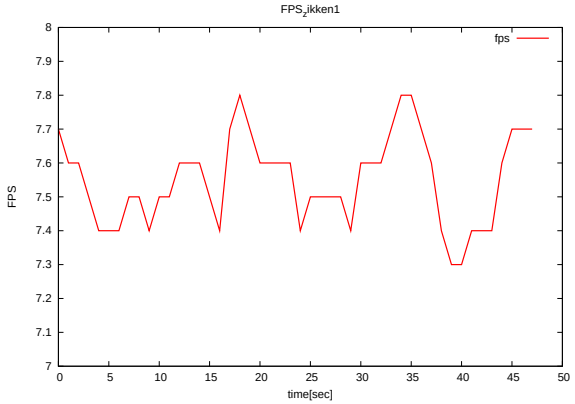


図 6 実験:経過時間,fps
Fig. 6 Experiment:Number of labels before processing