平成19年度

筑波大学第三学群情報学類

卒業研究論文

題目
顔向き解析による大画面への注視情報の取得

主専攻　　情報科学主専攻

著者　　南竹　俊介

指導教員　田中　二郎　高橋　伸　三末　和男　志築　文太郎

要　旨

プラズマディスプレイやプロジェクタなどの大画面を公共の場に設置し, それらを用いて情報を提示する機会が増加している. たとえばこれらの大画面から広告を提供する場合, 大画面を見た歩行者の数と注目された時間などが広告提供者にとって有益な情報となる. しかし現在, 屋外広告の効果測定は通行量のみを基準にして判断されていることが多く, 通行者のうち, 実際に何人が大画面に表示された情報を視聴したのかを知ることは容易ではない.

本研究では大画面ディスプレイなど情報提示媒体の前を通る歩行者を USB カメラで撮影しその画像を解析することによって, 歩行者の顔の向きを判別し, 公共大画面などへの注目情報を計測することを可能にするシステムの実装を行った.

目 次

図 目 次

第1章　序論

本章では，本研究の背景について述べ, その後, 今日急速に普及しつつあるユビキタスコンピューティングとデジタルサイネージについて触れた後, 本研究の目的を述べる．そして, 最後に本論文の構成を述べる.

1.1　研究背景

プラズマディスプレイやプロジェクタなどの大画面の低価格化が進み, 一般の家庭やオフィスなどに普及してきた. 特に駅前などの公共の場に設置される大画面の中には壁を覆うほど巨大なものも見られるようになってきた. これらの大画面は, 学校やオフィスなどの場合はプレゼンテーションに, 駅前やショッピングモールなどの場合は広告の提示や掲示板代わりに用いられることが多い（図 1.1）. 公共の場での大画面とのインタラクションを行う研究も盛んに行われており, 今後もこのような大画面が設置され, 利用される機会は増大していくと考えられる.

1.1.1　ユビキタスコンピューティング

ユビキタスとは,「遍在する」という意味をあらわす英単語であり, ユビキタスコンピューティングとは, コンピューティング環境の将来像として 1991 年に MarkWiser によって考案された概念である [1]. ユビキタスコンピューティング環境では, 無数のコンピュータやセンサが利用者からは見えない形で存在し, それぞれが無線 LAN などを介してネットワークで接続されおり, ユーザはそれらから必要に応じて情報を取得することが可能となる. また, これらのコンピューターがどのように動作を行っているか, ユーザから直接見えないという点が大きな特徴となっている.

これらの環境を真に実現するためには,「コンテクスト」を取得することが重要であるとされている. 情報工学におけるコンテクストには様々な意味があるが, ここでいう「コンテクスト」は,「ユーザやユーザをとりまく状況」を意味し, それらを取得するための研究も数多く行われている.

図 1.1: 研究室に設置されている場合の例

コンテクストアウェアネス

コンテクストアウェネスとは, 先述したコンテクストを取得するための技術や, それらに関する概念を意味する. ユーザの位置情報を取得するための手法としては, 様々な手法が研究されており, 代表的なものとして,RFID や測域センサなどを利用した手法や, カメラから取得した画像を解析する手法などが存在する. これらの技術を用いることでユーザは, 従来の手法ではユーザがデータとして入力しなくてはならなかったコンテクストを, コンピュータから自動で取得することが可能になった. これらの情報を利用することで, これまでにない新しいサービスや環境を提供できるのではないかと期待されており, 今後もコンテクストアウェアネスに対する関心は増大していくと考えられる.

1.1.2 デジタルサイネージとその問題点

技術の進歩により, 広告の提供方法もどんどん変化してきた. その最たるものが, 公共の場にモニタやプロジェクタなどを設置し広告を提示する, デジタルサイネージである. デジタルサイネージはデジタル情報で情報を提示するため, 特にそれらがネットワークに繋がっている場合, 掲示板などによる情報発信とは異なり, 一括して最新情報への差し替えを行うことなどが可能となる. デジタルサイネージの利点として設置された場所ごとのターゲット層にあわせて, 情報をピンポイントに提示することができる点がよく挙げられる.

しかし, これが本当に機能しているかは疑問であるといえる. なぜなら, 現状のデジタルサイネージは, 情報の提示方法こそ従来の掲示板や看板などと比較して, 容易かつ多彩に行えるものの, それらを見ているユーザの情報の取得に関しては行われていないことが多く, 行っていた場合でも Daily Effective Circulation(DEC) のように通行量を基準に判断を行っている場合が多いためである [2].

DEC とは, 公共大画面や広告塔などの前を通る歩行者, 自転車, 自動車の通行量を計測し, そのデータを基にそれらを見る可能性がある一日あたり通行量を計測するための指標である.[1] そのため,「実際に」どのくらいの人物がデジタルサイネージを見ているかという情報を取得することは極めて困難である. 図 1.1 は歩行者の数だけでは, 正当な評価を行えないケースの一例である. 手前の歩行者は大画面上の情報を見ているが, 奥の急いでいる歩行者は時計を見ているため, 大画面上の情報を一切視聴していない.

歩行者が広告を見ているか人間が直接計測するサービスも存在するが, 提示する情報が変わるたびに再び測定する必要がある. そのため, 表示を頻繁に更新可能なデジタルサイネージの場合, 人間の手による広告の効果測定は, ほぼ不可能であるといえる.

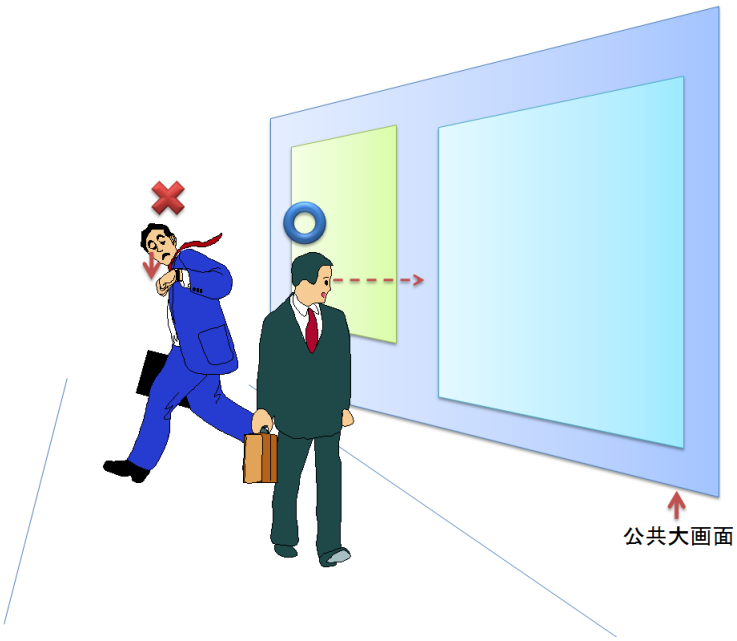

図 1.2: 通行量だけでは不十分な例

[1]屋外広告調査フォーラムより引用 http://www.okugai-forum.jp/DEC/kijyun.pdf

1.1.3 大画面とのインタラクションとコンテクスト

近年, 大画面とのインタラクションを行う研究が盛んに行われているが,PC のモニタのような, 小型の画面とのインタラクションと異なり, 大画面でのインタラクションは複数人で同時に操作を行うことを想定して設計されていることが多い [3]. その際に, 操作を行っている人物の識別などは必須であり, 操作者のコンテクストを取得することができれば, よりスムーズに人と大画面とのインタラクションが行うことができると考えられる.

人のコンテクストの測定には, 距離センサ (図 1.4)[2]や,RFID(図 1.3) [3] を用いて行う手法がある. 塩見ら [4] は科学館への来場者にその人物の情報を登録した RFID を渡し, その人物のコンテクストを展示案内ロボットがうけとることによって, 来場者とロボットの対話的行動を行うことを可能にした. しかし,RFID や距離センサでは公共大画面の前にいる人物のコンテクストを十分に取得できるとは言いがたい. 距離センサは, 画面の前にいる人物の正確な位置を取得することは可能ではあるが, 画面を見ていない人物も検知してしまうので誤作動を起こしてしまう可能性が潜在的に存在する. RFID に関しては, 大画面の前にいる人物が RFID を所持する必要があるという点で現実世界での応用には適さないといえる. また, これらの手法では, 大画面の前に人がいるかどうか判別することはできても, そのユーザが実際に大画面を見ているかどうかまでは判別できない点で問題があるといえる.

図 1.3: RFID

図 1.4: 距離センサ

[2]http://www.hokuyo-aut.co.jp/
[3]http://www.columnnetwork.org/blog/tag/workshop/

1.2 研究目的

ユビキタス時代の公共大画面を快適に利用するためには, 大画面の前にいる人物のコンテクストを取得することが必要不可欠であると考えた. しかし, 現在設置されている大画面の多くは, ただ一方的に情報を提示しているだけのものが多く, 能動的に大画面の前にいる人物のコンテクストを取得しようというものは少ない. また, 前節で述べたとおり距離情報や位置情報のみでは公共大画面を対象とする場合, 不十分であるといえる. ユビキタス時代の新しい大画面環境の実現のために取得するコンテクストとして, 大画面の前にいる人物の画面への注視情報を取得することを考えた. また, 本研究では公共の場に設置された大画面での利用を前提とするため, 歩行者に特殊な装置をつけさせることなく大画面への注視情報を取得することができるようシステムの設計を行った. 本研究では USB カメラを用いて大画面の前を通る歩行者を撮影することにより, 歩行者の顔の向きを推定し, 大画面への注目情報を大画面設計者に伝える視線推定システムを実装し, それを用いて屋外広告を評価し, その評価に基づき自動で広告を変更するアプリケーション,SinageGazer を実際に製作する.

1.3 本論文の構成

本稿では, まず 2 章で関連研究について述べ,3 章で視線情報を取得する大画面の提案,4 章で設計,5 章でその実装, 第 6 章では顔角度推測の精度実験,7 章では大画面への注視情報を利用した広告評価アプリケーション SignageGazer の実装について述べ,8 章で考察と課題について述べ, 結論で論を締める.

第2章　関連研究

この章では本研究と関連の深い研究について述べる. 本研究と関連が深い研究として, 特殊なハードウェアを用いた視線測定に関する研究, 大画面への視線情報を測定する研究の 2 分野にわけてそれぞれを説明する.

2.1　特殊なハードウェアを用いた視線測定の研究

ユーザの視線を検出する方法の一つとして, アイカメラを装着し直接眼球を撮影して, 視線を測定する研究がある. 中道ら [5] はアイカメラを装着することによって, web 閲覧時の視線の動きを解析し web 閲覧時のユーザビリティに関する問題点を収集した. 神代ら [6] は視線情報を GUI の操作へと応用した. ユーザは, 選択対象であるアイコン付近を見ることによって, マウスカーソルを大まかに移動することができ, その後マウスを動かし微調整を行った後に, アイコンの選択操作を確定することができる.eyebox2[7] は赤外線 LED をカメラに埋め込むことにより, 広告などへの人の視線を検知することができる.

これらの研究のような特殊なハードウェアを用いた視線計測は, 詳細かつ正確な視線情報の取得が期待できるものの, 装置の設置や着用に対するハードルが高いため, 屋外などの広告の評価, または日常でのインタラクションに気軽に用いることは難しい. また, 人の眼球の形状には個人差があるため, 数分間のキャリブレーションが事前に必要である点も問題であるといえる [8]. 本研究において必要なデバイスは, 一般に普及してきた USB カメラと計測用の PC のみであるため, ユーザが視線計測を行う上での敷居は比較的低いものであると考えられる.

2.2　大画面への視線情報を測定する研究

大画面への視線を測定する研究としては EnhancedWall[9] が存在する. EnhancedWall は大画面の前にステレオカメラユニットを設置し, カメラの前にいる人物の顔の向きを, 更新型テンプレートを用いて検出する. これによって取得した顔の向きを利用することによって, ユーザが大画面上のどの情報を見ているかを知ることが可能であるという点で, 本研究とは関連がある. また, 取得した視線情報の利用方法として, 表示する情報を拡大したり, 透化するシステムの提案を行っている. 同時に複数人数の顔の向きを取得できない点, 特徴点抽出のために, ユーザがステレオカメラユニットの前から動くことができない点で, 本研究とは異なると言える.

また, 駒木ら [10] は公共の場に設置された, サービスプラットフォームを見るユーザの顔の動きを基に, ユーザにサービスを提供する FaceConnect フレームワークを実装し, ユーザがモニ

タを見ると音楽が再生される広告アプリケーションの実装を行っている. FaceConnect フレームワークで対象とする大画面は縦置き型モニタであり, インタラクションは「モニタの中」を見続けることによって実現されるが, 本研究は大画面の「大画面のどこを」見ているかという情報を取得するという点でスタンスの違いがある.

第3章　大画面が取得する情報

本章では, 大画面の前にいる人物から取得するコンテクストとして, 大画面への注視状況を取得することを提案する. 取得する情報は, 顔の向きとモニタを見ている人物の位置情報に基づいて推定し, その人物がモニタをみているか, 更にモニタのどこを見ているのかを推定する.

3.1　取得する注視情報

公共大画面の一般的な利用方法は, 大きく二種類に分類することが可能である. いわゆるデジタルサイネージとしての利用, そして, 大画面の前にいる人物とインタラクションを行う目的での利用である. また,UBWALL[11] [1]のようにこれらを組み合わせて利用を行うケースもある. デジタルサイネージとして利用を行う場合, 大画面設置者が取得したい情報は,

- 広告を具体的に何人が見たか
- 広告のどこが見られたか
- 動画の場合, どのシーンが興味を引いたか

であると考えられる[2]. またこれらの大画面をインタラクションを行うために用いる場合は, これに加えて

- 現在誰が, 大画面のどの位置を, どこから見ているか

ということを知ることができれば, インタラクションの設計や実行の際に有益であると考えられる. 以下に, これらの情報の取得するための方法と解決策について述べる.

測定デバイス

ユーザの視線を推定する研究は古くから数多く行われており, それを測定するためのデバイスも多岐にわたる. 人間の眼球を直接測定する手法には, 視線測定装置, いわゆる装着型アイカメラを用いた研究や, 近赤外線カメラを用いた研究が存在する. 本研究では視線情報を取得す

[1]http://www.fujitsu-general.com/jp/products/ubwall/index.html

[2]屋外広告調査フォーラムが, 各企業の広告担当者に行ったアンケート結果によると, 屋外広告を設置する上で最も重視する調査データとして, 広告認知率（注目率）が最も多く挙げられている.http://www.okugai-forum.jp/PDF/chousakekka.pdf

る対象が公共大画面を見ている人物である. そのため, 課題の一つとして, その人物に無線タグや装着型アイカメラ[3]（図 3.1）などの視線測定のための装置をつけさせることなく視線情報を取得する必要がある点が挙げられる. また, 視線情報の取得を行う側も, 出来る限り特別な装置を用いることなく視線測定を行うことが可能になるのが理想であると考えた. 本研究では一般に普及してきた USB カメラ[4]（図 3.2）を用いて歩行者を撮影, その画像を解析するのでこの条件を満たしているといえる.

図 3.1: 装着型アイカメラ

図 3.2: USB カメラ

眼球検出による視線推定の問題点

人の正確な注視情報を取得しようとする場合, 眼球の情報を取得することが重要になってくる.

カメラ画像の解析によって白目部分と黒目部分を抽出し, これらの領域の比率から視線方向を求める研究や, 赤外線カメラを用いて, 角膜表面における反射像（プルキニエ像）の位置を求めることにより視線方向を求める研究もなされており [8][12], 正確な視線情報を取得しようとする場合, 有効な手段であるといえる. しかし, 公共の場に設置されたカメラから, 歩行者の眼球の情報を取得することは極めて困難である. 眼球の検出, 特に白目の部分の検出のためには, 顔画像を大きく撮影する必要がある. なぜなら, カメラから撮影された顔画像が小さくなるほど, 白目部分の検出が困難になり, 眼球の白目の部分と黒目の部分の境界がうまく撮影されなくなるためである. そのため, カメラから眼球を検出しようとする場合, ユーザがカメラから一定距離以上離れることができなくなってしまう点が問題である.

[3]株式会社クレアクト　http://www.creact.co.jp/jpn/topnews.html
[4]株式会社 Logicool　http://www.logicool.co.jp/

これは PC の操作インタフェースなどへの利用など, ユーザがカメラからあまり離れることがない場合であれば有効であるが, 公共大画面の前にいる人物の視線を検出するには大きな制約となってしまう. また, 顔の撮影領域を大きくしなければならないので, 複数名からの情報の取得も困難になる.

公共大画面をみる人物の視線

小画面での場合とは異なり, 人は大画面上の情報を見わたすとき, 目を動かすだけで情報を見るのではなく, 顔全体を大きく動かすことによって情報の閲覧を行うことが多くなる [13]. そのため公共大画面を見ているユーザの注視点を考える場合, 歩行者の顔の向いている方向から視線を推定できるのではないかと考えた. 眼球の動きを解析するわけではないので, 正確に 1 度や 0.1 度単位での正確な注視情報を取得することはできないが, 大画面に表示されている情報のうちどこを見ているのか推測する上でそこまで正確な情報は必要とはされない. なぜなら小画面に表示される情報とは異なり, 大画面上に表示される情報は必然的に大きく表示されることが多いためである. また, 歩行者は小画面とは異なり, 大画面からある程度距離を取らないと情報を閲覧できないため,10 度 20 度単位での顔の向きが取得できれば注視情報の取得は十分可能である考えられる.

顔の向いている方向を元に注視点を取得するには

- 大画面を見ている人物の顔の位置
- 大画面を見ている人物の顔の角度

を取得する必要があり, さらに複数名からも, これらの情報を取得可能なよう設計を行う必要がある.

第4章　視線情報取得システムの設計

この章では, 視線情報取得システムの設計方針について述べる.

4.1　大画面を見ている人物の顔の位置

カメラから撮影された画像を基に, 顔の位置, 角度を推定するには, 撮影画像中から顔を発見し認識を行う必要がある. 顔を認識するには様々な方法が存在しており, 特徴点を利用した方法や, パターンマッチングを利用した方法など様々な手法が存在している. 特徴点を利用した顔認識の場合, 肌色領域中の目や鼻, 口のような特定の部位の特徴量を取得し, それらの情報を対応付けることによって顔の認識を行う. また, 肌色領域が一定サイズを越えたら顔と認識する方法も存在するが, 顔と判断するための基準が肌色領域の大きさのみなので, 正確性という点では不十分であるといえる. 本研究では, 顔の位置の認識には顔のデータベース情報を基にしたパターンマッチングを用いて認識を行う. パターンマッチングによる方法の利点は画像全体に対し処理を行うため, 複数人の顔を抽出できる可能性がある.

4.2　顔のトラッキングと ID の設定

カメラ画像への顔のパターンマッチングから分かるのはそこに顔があるという情報のみであり, 前のフレームに写っていた人物と同一人物であるかということは分からない. そこで, 大画面中の情報を何人が見たのか, また, 今何人がみているのかという情報を知るには, 認識した顔のトラッキングを行う必要が生じてくる. また, カメラ画像から顔認識を行っている際, 照明光の変化や顔の移動などによって, 顔のトラッキングが一瞬だけ外れた後に再びトラッキングを開始することがある. この場合, トラッキングが再開されたときに, その人物がトラッキングが外れる前と同一人物であると指定する復帰処理が必要となる. 顔のトラッキングを行うには, 顔認識を行った顔画像から特徴量を抽出し, 次に取得した画像とのパターンマッチングを行い, 同一人物か認識を行う方法などが考えられるが, 計算量の増大によるシステムへの負荷が大きくなる点が問題であるといえる.

今回は, 顔認識が一定時間成功した顔に対し ID を設定し,ID が設定された顔のトラッキングが外れた場合, 削除候補 ID として顔リストへと登録を行う. 削除候補 ID は一定フレーム数以内に最後にトラッキングが外れた座標から, 一定範囲内の座標でトラッキングが再開されない場合, その人物はもうその場にいないものとして削除され, それまでの情報が登録される. 一定時間かつ一定範囲内でトラッキングが再開された場合, その人物は先ほどまでの人物と同一人

物とみなされ, その ID は削除候補リストから顔リストへと復帰し, 継続して情報を取得し続ける. 顔リスト登録の際のアルゴリズムを以下に示す.

if 顔認識成功 **then**
　顔リスト探索開始
　if 顔リストに登録されている ID の顔と認識を行った顔の距離が閾値以下 **then**
　　前のフレームと同じ画像と判断し, 情報を更新する
　end if
　削除候補リストの探索開始
else if 削除候補顔リストに登録されている **then**
　顔リストに復帰
else
　顔リストに新しい ID とともに登録
end if

4.3　大画面を見ている人物の顔の角度

人間の鼻は一般に, 顔の中央, 凸部の位置に存在している. 本研究ではその特徴に着目し, おおよその鼻の位置と, 肌色領域の重心の情報を基に大画面の前にいる人物の顔の向きを取得し, その人物が画面を注視している場所を判断する. 鼻を中心に, 顔を覆う程度の大きさの矩形領域に撮影画像を切り抜いた場合, カメラから見て正面を向いている顔画像の肌色領域の重心は, おおよそ矩形の中央に位置する. しかし, 顔の向きが上下左右いずれかの方向に傾いた場合, 顔の中心からみて肌色領域の重心は, 顔を傾けた方向とは逆の方向に移動する. これは, 顔を傾けると矩形領域内部に背景画像が多く含まれるようになるため起こる.

また, このときの X 軸方向の重心の位置の特徴として, 重心から縦に線を引いたとき, その直線はほぼ体の軸と一致するが挙げられる. これは, 人が横を向いたとき, あごの下に出来る空間と新しくみえてくる髪の毛の領域がほぼ同一面積のため, 顔の向きが変わっても大きく中心からは移動しないためこのようなことが起こる. これを利用して, 顔の中心位置と重心とのずれを計算することによって顔の向きを測定することが可能である. 顔の向きを変更したときの肌色領域の重心の移動を図 4.1 に示す. 図では矩形内部の二本の直線の交点が肌色領域の重心の位置を表している.

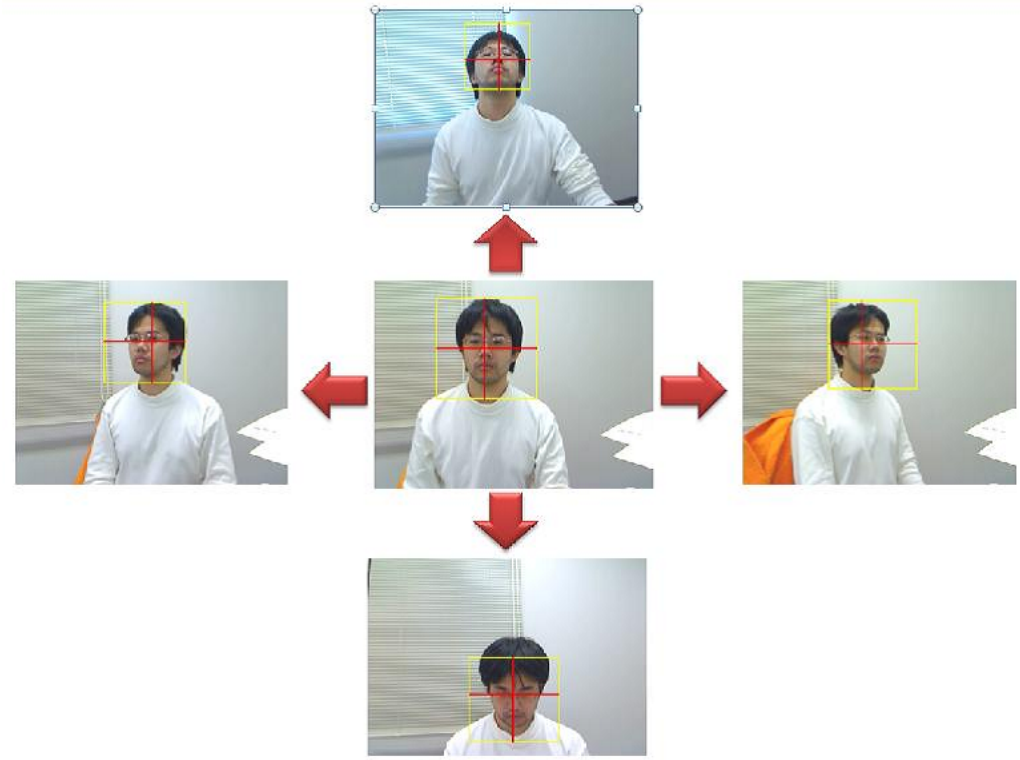

図 4.1: 顔の向きの変動による重心の移動

4.4 注視座標の推定

顔の角度の推定が終了後, 注視点の推定を行う. 注視点推定の際の座標系を図 4.2 に示す. 注視点推定の際の座標系は, カメラ座標を原点とし, 人物座標を $fi(X_i,Y_i,Z_i)$, 注視を行っていると推測される地点の座標を $Wi(X_i',Y_i',Z_i')$ とおく (カメラを大画面上部に設置した場合,Z_i は 0 とする). カメラと顔の中心の距離を R, カメラの光軸を基準にしたカメラと顔の位置の角度を θ_α とし, カメラが観測した顔の角度 θ_β をとすると, 注視点 W_i は

$$W_i = R\cos\theta_\alpha(\tan(\theta_\beta - \theta_\alpha))$$

より算出することができる. 注視座標推定の際の座標系を図 4.2 に示す.

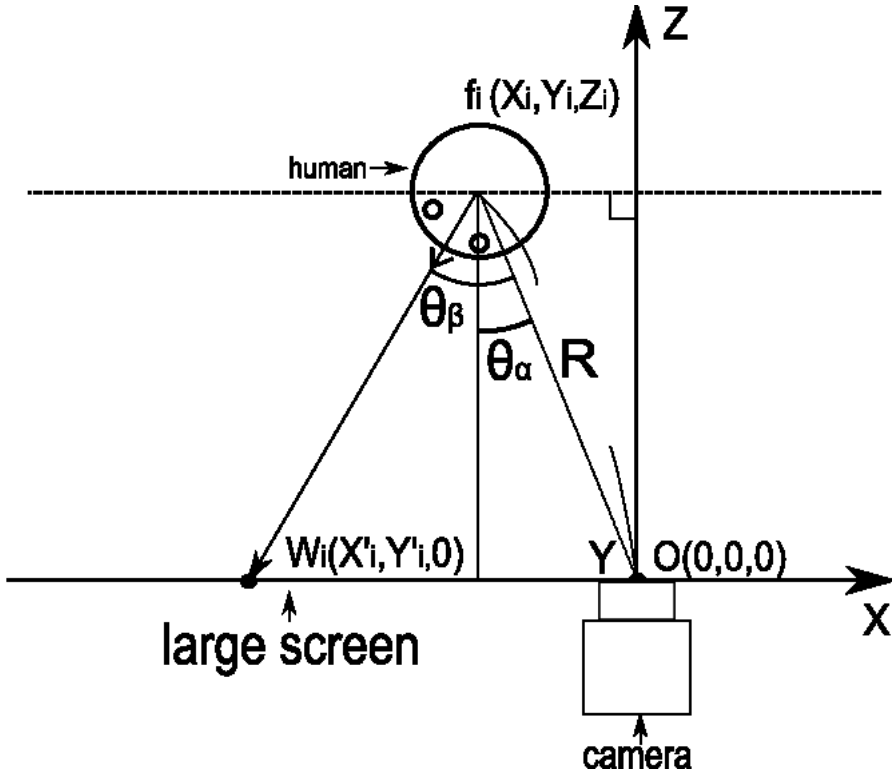

図 4.2: 注視点推定の座標系
i : 顔の ID
f : 顔の座標
W : 注視点座標
θ_α : かメラの光軸を基準とした顔の角度
θ_β : カメラから見た顔の角度
R : カメラからの顔の距離

現在の実装では, 距離Rは顔の大きさ（切り抜いた画像の大きさ）を基準にして判断をしている. そのため, 人の顔の大きさ次第で顔と画面との距離を誤ってしまう可能性がある. 正確な距離情報の取得のためには, 距離センサを用いる方法や, ステレオカメラによる両眼視差を利用した三角測量 [14] を行うことによって取得することが可能である. 具体的に大画面のどこを見ているか推定するには, 歩行者からみて大画面左下を基準にカメラの設置座標を指定し, 先ほど算出した値を引くことによって求めることができる.

第5章　視線情報取得システムの実装

前節で述べたとおり, 人間の鼻は一般に, 顔の中央, 凸部の位置に存在している. 本研究ではその特徴に着目し, おおよその鼻の位置と, 肌色領域の重心の情報を基に大画面の前にいる人物の顔の向きを取得し, その人物が画面を注視している場所を判断するシステムの実装を行った.

5.1　主要クラスの説明

- Capture クラス　USB カメラから画像を取得するクラス. 背景差分用の背景のセットも行う
- FaceDetect クラス　顔認識を行うクラスで, 顔の位置座標と大きさの指定を行う.
- ImageProccess クラス　カメラから取得した画像への処理を統括するクラス.
- FaceList クラス　認識した顔のリストアップ,ID の設定, 管理を行うクラス.
- WatchingArea クラス　FaceDetect クラス,ImageProcess クラスから取得した情報を統合し, 注視情報を推定するクラス.

5.2　顔位置認識

本プロトタイプでは顔の位置認識にコンピュータビジョン向けライブラリである OpenCV を用いている. OpenCV には多数の顔画像と非顔画像を基に作成された顔認識データベースとして,haarcascade_ frontalface_ alt2.xml が存在しており本研究ではそれを用いて顔の位置の特定を行う [15].

具体的な処理の流れは, まず Capture クラスからキャプチャしてきた画像を ImageProcess クラスに受け渡しグレースケールへと変換する. その後, その画像を縮小し, ヒストグラムの平滑化を行った後に FaceDetect クラス中の cvHaarDetectObjects 関数に画像を渡し検出を開始する.openCV プログラミングブック [16] によると,cvHaarDetectObjects における物体認識は, 最初に数百の正例と負例によって学習を行う必要があるこの場合, 正例とは，同一のサイズにスケーリングされた特定のオブジェクト, つまり顔を含むサンプルであり, 負例とは，正例と同一サイズの任意の画像を意味する. 学習後, 分類器は学習に用いられた画像と同じサイズの領域に対して適用される．その領域にオブジェクト顔が写っていると思われる場合は，分類器は ”1” を出力し，それ以外では，”0” を出力する. また,cvHaarDetectObjects による顔認識は複

数名の顔画像を認識することが可能である.cvHaarDetectObjects による顔認識は, 顔領域を指定する際に, おおよそ正面からみた顔の中心（鼻頭付近）を顔領域の候補として示す. これは顔の角度が変化したときも同様であり, 本プロトタイプではこの位置と肌色領域のずれを基に顔の角度を推定する.

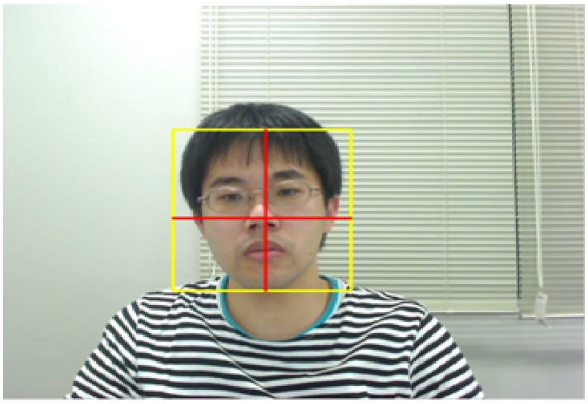

図 5.1: cvHaarDetectObjects による顔認識の実行

肌色領域の検出

顔認識の終了後,FaceDetect クラスから送信された顔の位置座標を基に,ImageProcess クラスは肌色領域の抽出を開始する. 肌色抽出を行う場合, 一般に指定された色空間の成分ごとに閾値を設定し抽出を行う閾値法が用いられることが多い. OpenCV では, カメラから画像を取得するとき,RGB 表色系で画像が取得される.

RGB 表色系とは,BMP 画像やモニタなどで標準的に用いられている色空間であり, RGB は, 赤（Red）緑（Green）青（Blue）の頭文字である. 一般に加法混色を表現するために用いられる. RGB 表色系を用いた肌色認識の問題点として, 照明環境の影響を強く受けやすいため, 肌色抽出のためのパラメータの設定が難しいため別の表色系に変換を行ってから閾値を設定することが効果的である. これは公共の場から肌色を抽出する場合特に注意を払う必要がある.

本システムでは RGB に代わる表色系として YUV 表色系を用いた.YUV 表色系のそれぞれの成分は Y が輝度 (luminance),U が青色成分の,V が赤色成分の色差 (chrominance) を表している. YUV 表色系の場合輝度が分離されるため,RGB 表色系と比較して, 肌色抽出の際, 閾値の設定が容易である点が特徴となる. 以下に RGB 表色系から YUV 表色系への変換式を示す.

$$Y = 0.256R + 0.504G + 0.098B$$

$$U = -0.148R - 0.291G + 0.439B$$

$$V = 0.439R - 0.368G - 0.071B$$

表色系それぞれの成分に対して閾値を以下の通りに設定し, それらを満たす画素を肌色画素として, 特に U の値に注目して二値化画像へと変換を行う. なお, それぞれの RGB から YUV への変換, 肌色抽出のための閾値の設定に関しては [17] を参考に実装を行った.

$$48 < Y < 224$$

$$-34 < U < -3$$

$$3 < V < 127$$

5.3 ノイズ処理

カメラから撮影した画像に対しそのまま肌色抽出を行うと, 背景画像中の肌色らしい色の画素などがノイズとして画像中に残ってしまう. 本システムでは, 顔の向きを肌色領域で顔の向きを判断するので, この画像から極力ノイズを排除する必要がある. そのため, 肌色抽出を行う前に顔画像に対しノイズ処理として背景差分と肌色領域によるラベリング処理を施している. 図 5.2 に前処理の流れを示す. システム起動時に Capture クラスのコンストラクタは背景画像

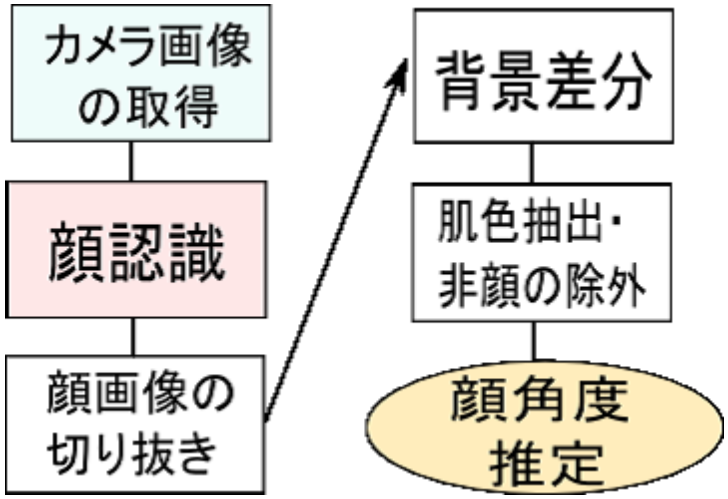

図 5.2: 顔向き取得のための前処理の流れ

として 10 フレーム分の画像を取得し, 画像の取得後, それらの画像の輝度の振幅などを計算し, それをもとに顔画像との比較を行い背景要素の除去を行う. また, 背景画像は背景差分を行った画像中の肌色領域が一定値以下である場合, 随時更新されていく. 処理を行う前の画像を図 5.3 に, 肌色抽出後背景差分を行った画像を図 5.4 に示す. 背景差分が終了次第,ImageProcess クラスはその画像の探索を行い, 肌色画素が 20 個以上隣接している場合, それを肌色領域としてラベリングを行う. ラベリングを行った画像中の最大領域が顔の肌色領域であると推測されるため, その領域を顔領域と認識し顔の角度の推定に用いる. 最大領域を抽出した画像を図 5.5 に示す.

図 5.3: 顔画像

図 5.4: 背景差分を行い肌色抽出を行った画像

図 5.5: 最大領域を抽出した画像

5.4 非顔画像の除外

cvHaarDetectObjects による顔認識の問題点として, 撮影画像中の照明の変化などが原因で, 肌色領域がほとんど含まれない領域でも, 近隣の領域との比較を行った結果, 特徴が顔に類似した場合, 顔と誤認識してしまうケースがあることが挙げられる. これは,cvHaarDetectObjects による顔認識を行うとき, グレースケール画像中から顔を探索する必要があるため発生する問題点である. 図 5.6 の場合, 一見顔とはほど遠い画像であるが顔と誤認識を行ってしまっている. この誤認識を抑制するために,ImageProcess クラスでは, 顔として認識された領域内部の肌色領域が一定面積以下である場合, 顔以外のものを顔として認識していると判断し, 顔候補リストに追加しないよう FaceList クラスへと通知を行う.

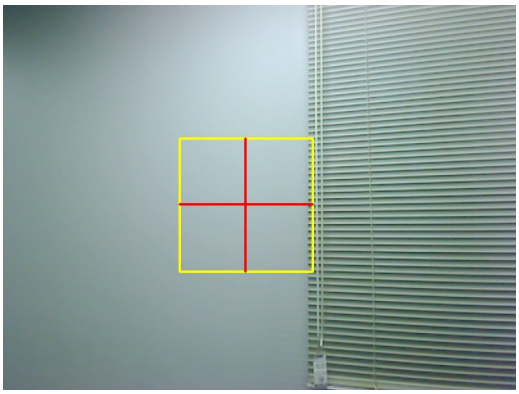

図 5.6: 壁が顔と誤認識されてしまった場合

5.5　顔向きの推定

前節の処理が終了後, ラベリング画像から矩形内部の肌色領域の重心を算出する．重心の算出が終了次第, 重心の座標を WatchingArea クラスへと渡し, 顔角度の推定を開始する．まず,FaceDected クラスより渡された顔の中心部分の座標と重心部分の座標の差を求める. X 軸方向の重心の位置は, おおよそ体の軸と一致するため, 顔を球体であると考えた場合, 矩形の一片の長さを W, 顔の半径の大きさを H, 矩形の左端から見た重心の X 座標を G とすると, 顔の角度 θ は,$\theta = \arcsin((W/2 - G)/H)$ から算出できる. この数式を元にシステムは顔の向きの推定を行う.

5.5.1　首領域による重心のずれ

矩形内部の Y 軸方向の肌色領域の重心を計算する場合, 矩形画像を鼻を中心とした正方形で切り抜いているため, 首も顔領域として計算されてしまう. このままでは, 矩形内部の肌色領域の,Y 軸方向の重心位置の計算の際に, ユーザが正面を向いていても, 重心が中心より下のほうに傾いてしまう. そのため,Y 軸方向の重心位置の計算の際には, 切り取った矩形内に, 首の領域がおおよそ 20 から 30% 程度含まれていると仮定して, 補正をかけて計算を行い, 正面を向いている時に顔の中心に重心がくるよう調整を行っている. また, 予備実験の結果上下の場合, 重心の移動量が左右と比較して小さいことが分かったため, 本システムでは上下に関しては上の方を向いているか, 正面を向いているか, 下の方を向いているかということのみを通知する.

5.6　結果の出力

これまでの処理で算出した顔の角度や ID, 視線情報, 人物の位置などは全て FaceList クラスの Face 構造体に登録される. ユーザはこの構造体にアクセスすることによって, これらの情報を取得することが可能になる. 算出した顔の角度, 処理にかかった時間などは CSV 形式で出力される.

第6章　実験

実装を行った顔向き推定システムの精度に関して実験を行った. カメラの前にいる被験者に異なる角度に設置されたオブジェクトを注視してもらい, 実際の注視点とシステムの推定する注視点とのずれを計測した.

6.1　被験者

被験者は情報科学コンピュータサイエンスに携わる 22 歳の男性 3 名である。

6.2　実験内容

被験者にはカメラから 2m 離れた位置に立ってもらい, 被験者からみて 10 度おきに配置されたオブジェクトを移動せずにそれぞれ 20 秒間ずつ注視してもらった. カメラは被験者の目の高さとほぼ同じ位置に配置した. 実験の手順は

1. 被験者は正面を向いた状態から測定を開始する.
2. 20 秒間正面にあるオブジェクトを注視したら, データの区切りを作るため, 一度被験者に自分の足元を見てもらう.(顔の向きを 90 度真下に向ける)
3. 被験者に足もとに配置されている角度計を参考に今度は被験者から見て 10 度左に配置されているオブジェクトの注視を開始する.
4. 顔の角度がカメラからみて 60 度になるまでこれを繰り返す.
5. 左が終了したら今度は右向きで同様のことを行う (正面は計測済みなので右向き 10 度から測定を行う).
6. 一度休憩を挟んだあと, これを 2 セット繰り返す

という手順で行った. また,20 秒の間に顔認識率が 50% を切った場合, 検出失敗とする. また、今回の実験はストップウォッチで 20 秒を計測した。

以下に実験結果を示す.

被験者の注視を行う地点が 60 度を越えると途端に顔認識に失敗してしまうことが表から読み取れる（図 6.1）. 被験者 2 は縁の太い眼鏡をかけていたため, 顔の角度が 50 度を超えたときにフレームで目が隠れてしまい認識に失敗してしまった.

	被験者1−1	被験者1−2	被験者2−1	被験者2−2	被験者3−1	被験者3−2	平均
右60度	計測失敗	計測失敗	計測失敗	計測失敗	計測失敗	計測失敗	計測失敗
右50度	−37.6818	−43.7955	計測失敗	計測失敗	−48.3529	−50.2	−45.0076
右40度	−28.8889	−34.4894	−47.9167	−46.4	−45.1333	−42.3333	−40.8603
右30度	−21.0889	−25.2553	−34.1957	−32.1489	−37.9762	−36.7955	−31.2434
右20度	−12.881	−16.3261	−25.6	−24.1667	−17.8723	−28.0682	−20.819
右10度	−5.73469	−14.0638	−14.1455	−11.6809	−13.1837	−15.7037	−12.4187
0度	3.617021	−5.46154	−2.88636	−4.92308	−3.91228	3.689655	−1.6461
左10度	16.5	14.44444	16.15686	11.13333	8.265306	−2.34694	10.69217
左20度	24.36957	21.80435	23.55556	17.28261	18.10909	18	20.52019
左30度	27.45946	27.97872	28.59574	25.15556	22.76316	24.7963	26.12482
左40度	計測失敗	計測失敗	29.92105	26.40625	31.5	35.81395	30.91031
左50度	43.90698	計測失敗	計測失敗	計測失敗	41.67568	49.13514	44.90593
左60度	計測失敗	計測失敗	計測失敗	計測失敗	計測失敗	計測失敗	計測失敗

図 6.1: 被験者, 試行ごとの平均取得角度

エラーの数

	被験者1−1	被験者1−2	被験者2−1	被験者2−2	被験者3−1	被験者3−2	合計	エラー率平均(誤差+−10度以上)
右60度	計測失敗	計測失敗	計測失敗	計測失敗	計測失敗	計測失敗	計測失敗	
右50度	13/44	8/43	計測失敗	計測失敗	0/34	0/45	21/167	0.125748503
右40度	18/45	7/47	13/36	4/33	14/45	8/43	64/249	0.258064516
右30度	6 / 45	0/47	4/46	3/46	24/42	14/44	51/270	0.189591078
右20度	0/42	0/46	3/44	2/42	0/42	10/45	15/260	0.071428571
右10度	0/49	1/47	0/55	0/47	0/49	0/54	1/301	0.003984064
0度	0/47	0/52	0/44	0/52	0/57	0/58	0/310	0
左10度	2/44	1/45	0/51	0/45	0/47	28/49	31/283	0.108391608
左20度	2/46	0/46	0/45	0/46	0/55	0/48	2/286	0.003496503
左30度	1/37	0/47	0/47	0/45	2/37	3/44	6/267	0.02247191
左40度	計測失敗	計測失敗	28/38	24/32	13/46	0/43	65/159	0.386904762
左50度	1/43	計測失敗	計測失敗	計測失敗	8/37	1/37	10/155	0.006451613
左60度	計測失敗	計測失敗	計測失敗	計測失敗	計測失敗	計測失敗	計測失敗	

図 6.2: 被験者, 試行ごとの平均エラー率

推定される顔の角度について, 全体の平均をみると角度がシステムが要求する程度に取れているように見える。しかし, 被験者個々の平均を比較して見てみると, 推定される角度に最大で 20 度近いばらつきが見られた. これは被験者によって, オブジェクトを注視する際の顔の角度が異なる点と顔の大きさや形によって推定する顔の角度が異なる点が原因の一つであると考えられる。また, 同じ被験者の試行の平均値にも誤差が生じていることが確認された. 誤差はおおよそ 10 度以内に収まっているものの中には 20 度近い差がでているものもあり, 顔の推定された顔の角度と注視点のずれは時に大きくなることが観測された.

注視点を推定する際+-10 度まで誤差を許容しそれ以上をエラーとした場合, 推定した注視点と実際の注視点とでの全体のエラー率は, 図 6.2 のようになった。おおよそ 30 度程度までは正確に注視情報を取得できているが, それを越えるとエラー率が上昇した。被験者 2 は, 左 40 度の角度の時測定時間の半分以上異なる値が出続けた. また, 被験者 2 は左 40 度を向いている時はエラーが多発したが, 右 40 度を向いているときはそこまで多くのエラーは発生していない. これは被験者の顔の左右のバランスや光のあたる角度が影響したと考えられる. 他の被験者に関してはそこまで大きなエラーは発生しなかった.

いずれにしても, 現時点では被験者の母数が 3 のため, まだ統計情報としては十分とは言い難い. また, 被験者間による精度のばらつきや環境の変動による推定角度のばらつきも観測されている. そのため, 本システムでは, 算出した顔の角度を正確な顔の角度としてではなく,20 度ずつの比較的大きな顔の向きの変化を検出するためのパラメタとして扱っている. 今後はより多くの被験者に対し実験を行い, 正確な情報を取得できているのかの検証を行う予定である.

第7章　広告評価,変更アプリケーション SignageGazerの実装

この章では先述したシステムから取得した大画面の前にいる人物の位置情報, 視線情報を利用した, 公共大画面上に表示される広告を評価するアプリケーション,SignageGazer とその動作について説明を行う. SinageGazer は歩行者のデジタルサイネージへの注目状況を取得することができ, その注目状況に応じて表示する広告画像, または広告動画の変更が可能である. また, 本アプリケーションの開発環境, ならびに実行環境は以下の通りである.

- CPU：Pentium(R)4:3.00GHz, Memory:1GB
- OS：Microsoft Windows XP Professional Version 2002 Service Pack 2
- 開発環境： Microsoft VisualC++ 2005
- 撮影デバイス：Logicool q-cam pro9000

7.1　初期設定

SignageGazer を利用する前にユーザは以下の情報を事前に登録しておく必要がある.

- 大画面の大きさ
- 大画面左下から見たカメラの位置座標
- 設置する USB カメラからキャプチャ可能な画像のサイズ
- カメラから取得した画像を保存するか？

設定終了後, ユーザは大画面に表示を行い, 測定を行いたい広告画像, または広告動画の登録を行う. ユーザは利用したい画像や, 動画ファイルのパスを事前にテキストファイルで保存しておく必要がある. また, 表示したい動画を複数登録しておくことにより, プレイリストを作成することが可能である.

7.2　プログラムの動作

プログラム起動後, 図 7.1 が立ち上がる. ユーザは表示されたダイアログ中の, ファイル参照ボタンをクリックする, または設定ファイルのパスをテキストボックスに入力することのよって, 前節で述べたプレイリストを選択することができる. これらの登録が完了した後, 開始ボタンをクリックすると視線の測定が開始され, 同時に指定された動画, もしくは静止画像が画面に表示される（図 7.2）. 動画を再生する場合は, 動画再生用のスレッドが生成され, 動画の再生が視線推定とは独立して実行され, 停止ボタンをクリックすると動作が停止する. （静止画の場合は特にスレッドの生成は行われない）動画の再生クラスに関しては DirectShow を用いて実装をおこなった. 表示可能な静止画は Jpeg と Bmp, 動画の形式は Mpeg 形式と Avi 形式である.

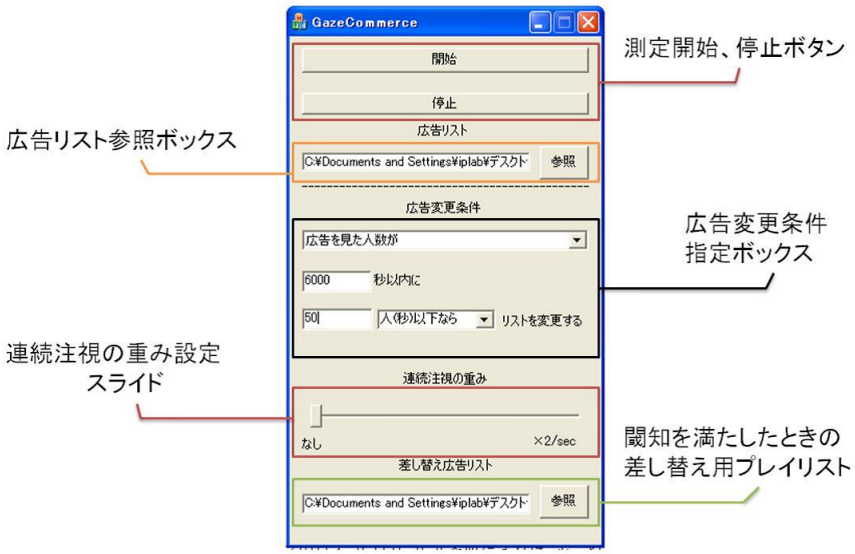

図 7.1: SinageGazer 起動画面

7.3　観測領域の設定

画面のサイズが巨大化すればするほど, 画面を一度に捉えることは難しくなる. そのため, 大画面の前にいる人物が大画面上のどこを見ているか, という情報を得ることが重要になってくる. ユーザは提示する情報の中で, 注目されているか知りたい領域を, 矩形領域で複数指定する

図 7.2: 広告の表示と測定の開始

ことが可能である. ただし, このとき指定する領域同士で重複することがないよう宣言する必要がある. 動画を評価する場合, 場面の変遷とともに観測領域を動的に変更したいという欲求が生じることが考えられる. その場合は, 観測領域を取得したい場面の時間とともに指定するとこによって, 場面ごとに観測領域を変更することが可能である. ただし, 現状の実装では, 動画のタイムラインや場面を解析して領域の変更を行っているわけではない. そのため, 注目情報を取得したい動画の場面を指定する際にという問題点がある. この問題点の改善のために, 動画の解析するなどしてシステムとの正確な同期がとれ, より簡便に領域の設定ができるよう, 今後改良を行っていく予定である.

7.4 取得可能な情報

本アプリケーションを用いることでユーザは提供した広告に関する様々なデータを取得することが可能になる. 現時点での実装でユーザが取得可能な情報は

- 画面を見た人物の数
- 画面が見られた総時間
- ID の設定された人物が注視した画面の位置
- ID の設定された人物が画面を見た時間
- ID の設定された人物の画面前での移動履歴

である. 動画の場合は, それぞれの情報を再生された時間ごとに取得することが可能である. これらの情報はは csv 形式で出力される.ID の設定された人物の顔画像を取得することも可能ではあるが, デフォルトではこの機能はオフとなっている. この機能をオンにした場合, 画面を見た人物の顔が jpeg 形式で保存され, ユーザはそれを閲覧することができる. この機能を用いた場合, ユーザは具体的に誰が大画面を見たかが分かるため, 研究室やオフィス内部など, 顔画像を撮影されてもあまり問題の発生しない場所で用いる場合などには有効であるといえる. 顔画像保存機能をオフにした場合, 画像解析に用いた顔画像は破棄されるため, 駅前やショッピングモールなどプライバシーの面で問題が発生しそうな場所で運用する場合はこの機能を使わないことが望ましい.

7.5 広告の差し替え

広告提供者は表示した広告があまり効果を上げていないと思われる場合, またはその広告の効果が高いためもっとその広告を全面に押し出したいと考えたとき, 別のバージョンの広告を事前に準備しておくことによって, それを自動的に変更することが可能である.

ユーザは, 画面を見た人物の数, 画面が見られた総時間, に対して閾値を設定することができる. 閾値の設定された広告が閾値を満たさなかった場合, 広告差し替え通知部は, その広告はあまり効果をあげていないと判断し, 別の広告に差し替えるよう, 広告表示部に通知を行う. 設定を行うには SinageGazer の起動画面中の広告条件設定ボックスの設定を変更すればよい. 認知率の低い広告を自動的に差し替えるときのイメージ図を図 7.3 と図 7.4 に示す. 大画面内部の広告をを囲っている矩形が観測領域であり, そこへの視線がない右上の広告が認知率の低い広告であるとして差し替えられている.

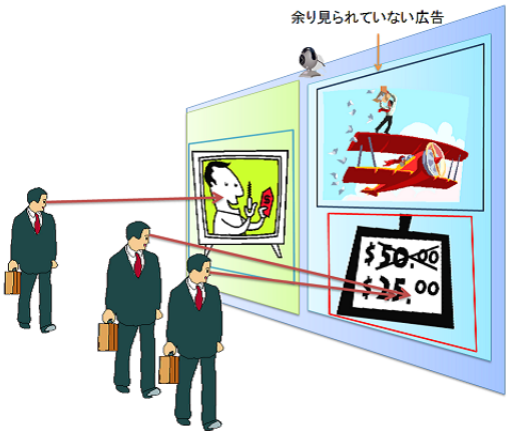

図 7.3: 認知率の低い広告の発見

図 7.4: 広告の差し替え

広告変更の通知を行うまでの時間は秒数で指定を行う. 静止画の場合, 広告変更通知が届い

た時点で即座に画像の変更が行われるが, 動画の場合は急に変更するのではなく, 再生が終了次第差し替えを行う. 動画広告を差し替えるときの処理の流れを図 7.5 に示す.

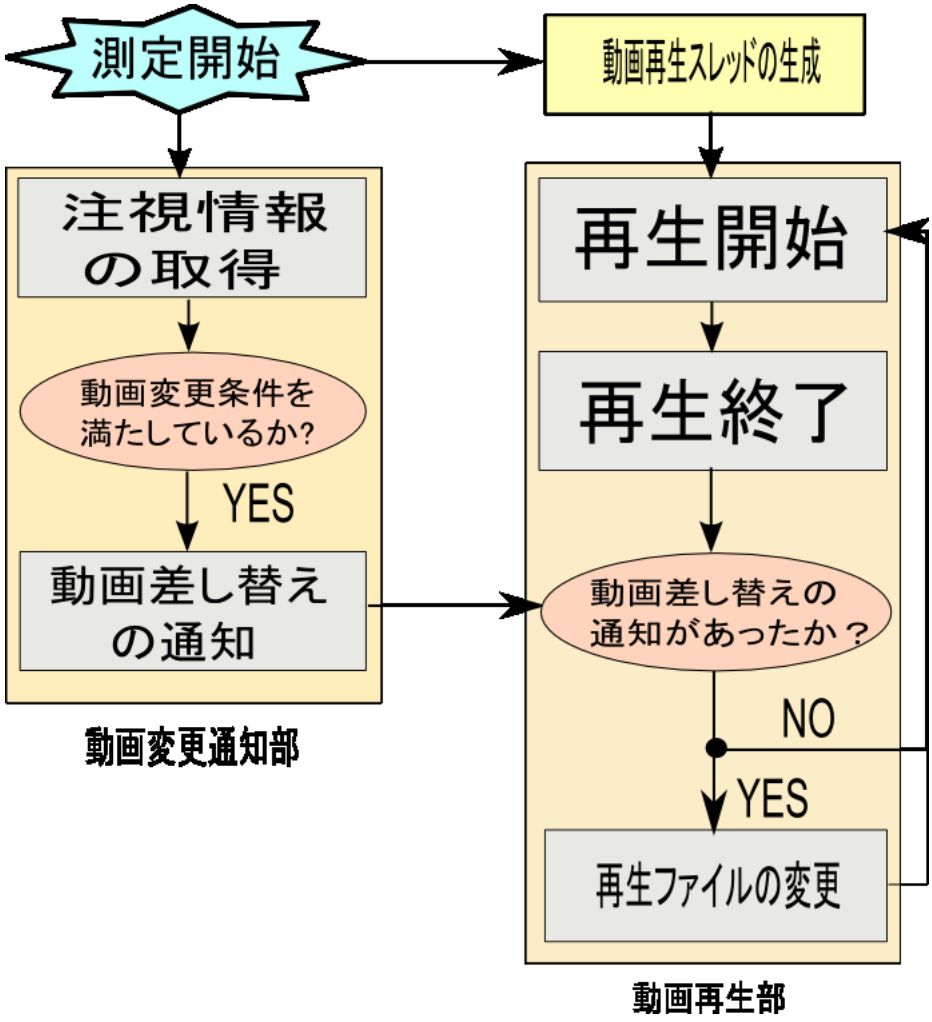

図 7.5: 動画広告を差し替えるときの処理の流れ

7.6　連続的な注視への重みの設定

人が広告を見るとき, その情報がその人物にとって興味のない情報の場合, その広告を長時間見ることはない. しかし, 興味のある情報の場合, 立ち止まるなどして, 長時間見続けることがある. この場合, 前者と後者では注視情報の持つ意味合いが変わってくる. 例えば,5 人が 2 秒ずつ見た広告と,1 人が 10 秒見た広告では見られた合計時間は等しくとも人をひきつける効果に関しては後者の方が高いと考えられる.

ユーザは設定ダイアログ上の連続注視への重みスライダー（図 7.1）を操作することにより, 連続して見られた広告への注視情報について重みをつけることができる. 重みが大きければ大きいほど, ユーザがその広告への連続注視を重視していることになる. 重みをつけられた注視情報は, 広告変更のための閾値の計算に用いられる. 現在の仕様では連続注視が一秒連続するごとに設定された重みが今までみた秒数に積算されていていく. 設定可能な値の範囲は× 1/sec から× 2/sec までの間である.

7.7 アプリケーションの利用シーン

提案するアプリケーションの利用シーンを以下に挙げる.

とある鉄道会社に勤務する A 氏は, 自社の駅前に設置されている公共大画面への広告提供者を探していた. 広告を出資してくれると思われる企業を適当にリストアップし, 必死に営業を行うが思うような成果があがらない. 何故先方から良い返事をもらえないのか, よくよく分析を行ってみたところ, 先方が広告を提供するべきか否か判断するための材料をこちらから余り提供できていない点に問題があるのではないかという結論に至った. 実際,A 氏は交渉のとき「具体的にどの程度の数の人が広告を見ているのか？」「どのようなタイプの広告のうけがよいのか？」と聞かれると具体的なデータを提示することができず, そのまま交渉が終わってしまうことが多くあった.

駅への一日の来場者数は分かっていたため, それをデータとして提供したが, 具体的に大画面を見た人の数やその好みは分からない. そのため広告を提供してもらうための根拠としては説得力が薄く困っていた.

そこで A 氏は本アプリケーションを用い, 大画面の前を通るユーザがどの広告を, どの程度みているかの測定を行うことにした. 取得したデータを解析したところ, 化粧品に関する広告の視聴数が全広告中最も高いことが分かったので,A 氏は営業を行う企業のリストに化粧品会社を追加した. さらに,A 氏は交渉の際に正確なデータを相手方に提供できるようになったため, 以前より容易に交渉を進めることが可能となった.

7.8　今後の構想

今後の拡張の予定として, 各地で測定を行っている SignageGazer の情報を広告提供者のサーバに送信するなどして, 全国各地に設置された, デジタルサイネージへの注視情報を一括して管理できるよう拡張を行う予定である. また, 視線情報に対する重みを閲覧された時間のみではなく, 地域ごとに設定できるようにするなど, より効果的な広告評価が可能になると考えられる. この機能の実現によって, よりピンポイントな広告の提供が可能になると考えられる. また, 現時点の実装では, ユーザは広告変更のための閾値を低めに設定することにより, ユーザが画面を見ると同時に表示を変更することが可能である. 現時点では静止画などの広告の表示のみが可能であるが, 今後はプログラムの呼び出しを行うなどして, 手軽にインタラクションを行うことができるよう拡張を行う予定である.

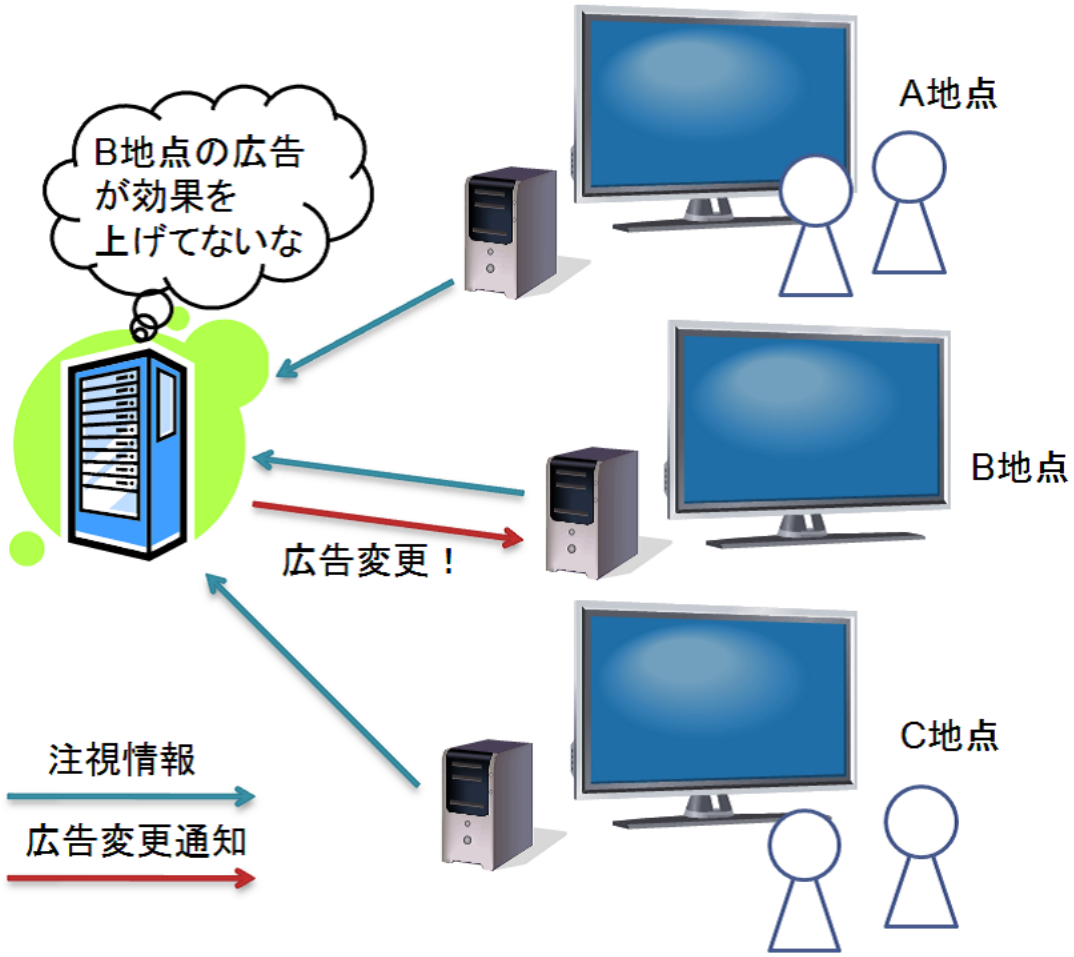

図 7.6: サーバを介した広告の管理イメージ

第8章　考察と課題

今回試作した, 注視情報測定システムと, それを利用したアプリケーション SignageGazer は駅前などの公共の場で運用されることを想定して実装を行った. このシステムによって, 公共大画面を用いて, 広告を提供する場合は広告への注視情報を, しかし, 現状のシステムについて考察を行っていく過程で, いくつか解決しなくてはならない課題があることがわかった.

8.1　顔認識に関する問題点

本システムで利用している OpenCV の顔認識データベースは, 人間を正面から撮影した顔画像を元に作成されているため, 顔の角度がおおよそ 60 度を越えどちらかの眼球が見えなくなると, 顔認識に失敗してしまう. そのため, この角度を超えた人物の顔に関しては, 顔の角度を推定することができない. 特にメガネをかけた人物は眼鏡のフレームで目が隠れてしまうことが多く,50 度を越えた時点で認識が停止することも多かった.

この問題の解決には, 異なる角度から顔の撮影を行うカメラを複数設置することが考えられる. これを行う場合, 複数のカメラからの情報の統合とそれぞれのカメラに映った人物の識別が重要である. 顔の角度ごとにデータベースを新たに作成することも考えられるが, 顔の上下左右の角度ごとにデータベースを用意する必要があり, それら全てに関してパターンマッチングを行う必要があるため, 実装を行う上でなんらかの工夫が必要であると考えられる.

顔向き推定に関しても課題が残る. 現時点での実装では, 顔の角度は顔の中心からの肌色領域の重心のずれから角度を推定しているため, 人の顔の形や髪型次第で推定される角度に変化が生じてしまうことが確認された. 考えられる解決策として, 顔の部品抽出を Active Appearance Model(AAM) を用いて行う方法が考えられる [18]. AAM は, 顔などの画像上で様々な形に変形する物体の形状と, 内部の明度分布を低次元で同時に表現することのできる統計モデルである. AAM 学習モデルから三次元顔画像モデルを作成することが可能であり, これを用いることによって, より正確に顔の向きを判断できることが期待される.

8.2 SinageGazer の課題

また,SignageGazer が測定しているのは, あくまで大画面を見た人物の情報である. そのため, 大画面の前を通った歩行者の人数などは分からないので, 歩行者のうちどの程度の割合の人物が広告を見たのか, という情報などを知ることはできない. これを知るにはレーザースキャナなどを用いて人数を計測する , または, カメラから取得した画像から人物追跡を行う必要があると考えられる. カメラ画像からの人物追跡には, 白井 [19] らによる手法などがすでに存在しており, 今後これらの研究を組み合わせて実装を行っていく方針である.

我々は今後, これらの問題に対応するための具体的な方法を模索し, 実装を行うと同時に, システムに関する客観的な評価を得るための実験を行う予定である.

第9章　結論

本研究では, 大画面付近に設置された USB カメラから, 大画面の前を通る人物の画像を取得し, その人物のおおよその注視点を計測するシステムの実装を行った. このシステムにより, ユーザは手軽に大画面への注視情報を取得することが可能になる. また, 公共大画面の前を通る人物の注視情報を利用した, 屋外広告評価アプリケーションの実装を行った. このアプリケーションによって, より正確な広告の評価を行うことが可能となり, マーケティングや広告デザインの分野などへの幅広い応用が期待できる.　今後の展望としては, より広範囲な顔の角度の取得の実現, より多人数への対応を行っていく. また, 顔向きを利用した, 新しい大画面インタラクションの実現にむけて研究を行っていく予定である.

謝辞

本研究を進めるにあたり, 指導教員である田中二郎先生をはじめ, チームリーダーの高橋伸先生, 三末和男先生, ならびに志築文太郎先生には, 幾度となく丁寧なご指導と適切な助言を頂きました. 心より感謝申し上げます. また, 田中研究室の皆様, 特にユビキタスチームの皆様には, 多くのご意見やご指摘を頂きました. この場を借りて御礼申し上げます. 最後に私が挫けそうなときに私を支えてくれた両親や, すべての友人に感謝を申し上げます. 本当にありがとうございました.

参考文献

[1] Mark Weiser. The computer for the twenty-first century. *In Scientific American*, pp. 94–104, 1991.

[2] 清水公一. アメリカの屋外広告事業と日本の効果測定指標. 国際文化研究所紀要 第 8 号, pp. 53–72, 2002.

[3] S.Takahashi C.Jin and J.Tanaka. Interaction between small size device and large screen in public space, proceedings of the 10th international conference on knowledge-based and intelligent information and engineering systems. *KES2006*, pp. 197–204, 2006.

[4] Masahiro Shiomi, Takayuki Kanda, Hiroshi Ishiguro, and Norihiro Hagita. Interactivehumanoid robots for a science museum. *IEEE Intelligent Systems*, Vol. 22, pp. 25–32, Mar/Apr 2007.

[5] 中道上, 阪井誠, 島和之, 松本健一. 視線情報を用いた web ユーザビリティ評価の実験的検討. 情報処理学会研究報告, ソフトウェア工学, 第 143 巻, pp. 1–8, July 2003.

[6] 神代和範, 大和正武, 門田暁人, 松本健一, 井上克郎. 視線とマウスの併用によるドラッグ＆ドロップ方式の実験的評価. 電子情報通信学会技術研究報告 HIP99-81, 第 99 巻, pp. 37–44, March 2000. 東京.

[7] xuuk. eyebox2. *https://www.xuuk.com/*.

[8] 大野. 視線から何がわかるか-視線測定に基づく高次認知処理の解明. 日本認知科学会『認知科学』』9 巻 4 号, pp. 565–576, 2002.

[9] Hideki Koike Kotaro Kitajima, Yoichi Sato. Enhanceddesk and enhancedwall: Augmented desk and wall interfaces with real-time tracking of user's motion. *Ubicomp2002*, pp. 27–30, 2002.

[10] 駒木亮伯, 岩井将行, 神武直彦, 高汐一紀, 徳田英幸. 顔の移動軌跡に基づくサービス制御機構. 情報処理学会研究報告.UBI no.14, pp. 103–108, 2006.

[11] 富士通ゼネラル. Ubwall, http://www.fujitsu-general.com/jp/products/ubwall/index.html.

[12] Jason S. Babcock and Jeff B. Pelz. Building a lightweight eyetracking headgear. In *ETRA '04: Proceedings of the 2004 symposium on Eye tracking research & applications*, pp. 109–114, New York, NY, USA, 2004. ACM.

[13] 山田光穂, 福田忠彦. 大画面ディスプレイから受ける心理効果の客観的評価に関する基礎検討　-頭と眼の動きの相互関係について-. テレビジョン学会誌, pp. 714–722, 1989.

[14] Jean Ponce David A.Forsyth. *Computer Vision*. Person Education Inc., 2003.

[15] Bernt Schiele. Hannes Kruppa, Modesto Castrillon Santana. Fast and robust face finding via local context. *Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, 2003.

[16] 奈良先端科学技術大学院大学 OpenCV プログラミングブック制作チーム. OpenCV プログラミングブック. 株式会社毎日コミュニケーションズ, 2007.

[17] ”中川弘隆”. ”高次局所自己相関特徴による高速画像認識モジュールの開発と自律走行型ロボットへの応用”. Master's thesis, ”北陸先端科学技術大学院大学”, ”2002”.

[18] Fadi Dornaika and Franck Davoine. Head and facial animation tracking using appearance-adaptive models and particle filters. *CVPR Workshop on Real-Time Vision for Human-Computer Interaction*, 2004.

[19] 白井良明, 三浦純. 複雑背景における人の追跡. 情報処理学会論文誌:コンピュータビジョンとイメージメディア, pp. 33–42, 2002.