平成19年度 修士論文

CSEEGAIA:

ユーザ提示画像を基にした クリッカブル動画自動生成システム

電気通信大学 大学院情報システム学研究科

情報システム設計学専攻

0650031　三樹 一貴

指導教員　多田 好克　教授
岡本 敏雄　教授
古賀 久志　准教授

提出日　平成20年1月29日

目 次

図 目 次

第 1 章

はじめに

インターネットの発展に伴い、今日では個人での Web ページ、ブログの開設や閲覧、SNS の利用が広く普及している。最近の傾向としては YouTube[1](図 1.1)、Google Video[2](図 1.2) といったインターネット上の動画投稿サービスが積極的に利用されるようになってきている。更に、それら投稿されている動画に対して後付けでコメントを付加する、2005 年度上期の IPA 未踏ソフトウェア創造事業に採択された Synvie[3] に発想を得て開発されたニコニコ動画 [4](図 1.3) といったようなサービスも現れ、話題を呼んでいる。そして今や個人の Web ページやブログ、SNS (Social Networking Service) でも動画の掲載が可能となっている。

このような状況の下、動画視聴者が動画中で興味を持った対象をクリックした際、即座に関連性のある詳細な情報に当たれれば有益である。現在でも、特定箇所をクリックすることにより視聴者をリンク先に導く Web 上の動画もいくつかは存在する。しかし、それらは動画のフレームごとに人手で位置を指定し、リンク先をタグ付けしているのが現状である。一般の人々がそのような事をするには、ある程度の計算機スキルが必要であり、何より繁雑で面倒である。

そこで本研究では、Web 上において、個人の計算機スキルや、計算機に関する専門的な知識に関わらず、クリッカブル動画を自動で生成するシステム、CSEEGAIA (Clickable movie System with Easy and Effective Generator on Automatic Image Analysis) の設計と実装を行う。ここでいうクリッカブル動画とは、動画中の任意の時点、位置にクリック可能なオブジェクト、すなわちクリッカブルオブジェクト

図 1.1: YouTube

を含む動画のことである。

また、視聴者がその動画中に表れたクリッカブルオブジェクトを容易に認識でき、クリックしやすいインタフェースの設計と実装も同時に行う。

本論文では、まず第 2 章で CSEEGAIA の概要を述べる。続く第 3 章で CSEEGAIA の設計、第 4 章でその実装を述べる。第 5 章で CSEEGAIA の評価を行い、第 6 章で関連技術について述べる。そして最後に、第 7 章で本論文をまとめる。

図 1.2: GoogleVideo

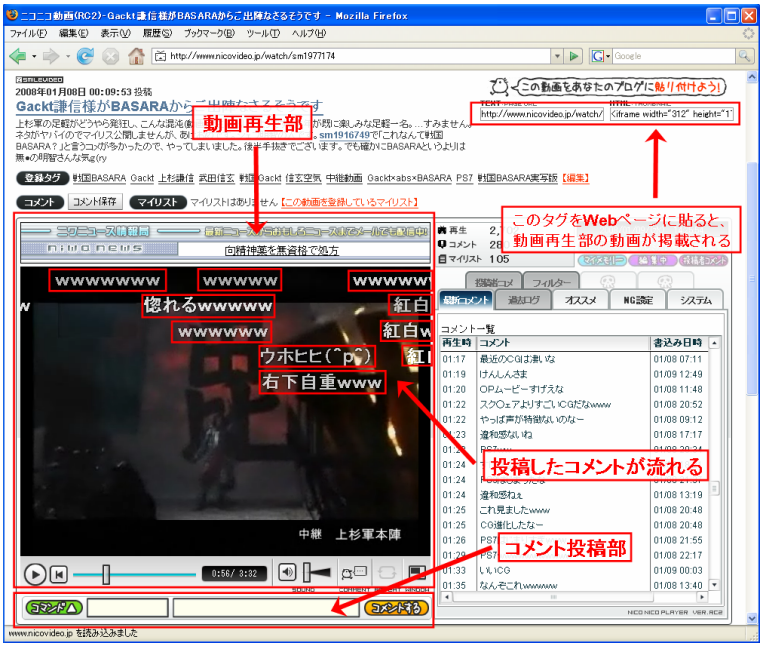

図 1.3: ニコニコ動画

第 2 章

システムの概要

本システム CSEEGAIA は、計算機を持っている人なら、誰もが手軽にクリッカブル動画を作成することを可能にするシステムである。それと同時にクリッカブル動画視聴インタフェースも提供するため、作ったクリッカブル動画を Web ページやブログ、SNS などにすぐに掲載でき、視聴者をより注目させることが可能である。

システムの処理のイメージを図 2.1 に示す。

自身の利用する Web ページやブログ、SNS などにクリッカブル動画視聴インタフェースを掲載したい場合、対象動画 (図 2.1 の movie) とイメージ (図 2.1 の image)、リンク先の URL(図 2.1 の information) を、Web アプリケーションとして実装された本システム (図 2.1 の CSEEGAIA) にブラウザを通して入力する。すると、本システムは動画中に表れるイメージをクリッカブルオブジェクトとしたクリッカブル動画を生成し、処理が終わり次第終了音を鳴らし、ブラウザに処理結果を出力する。ブラウザの出力表示に従って、出力されたタグを自身の Web ページやブログ、SNS などに貼り付ければ、クリッカブル動画視聴インタフェースが掲載される。

該当する Web ページ、ブログ、SNS にアクセスすれば、クリッカブル動画の視聴が可能となる。

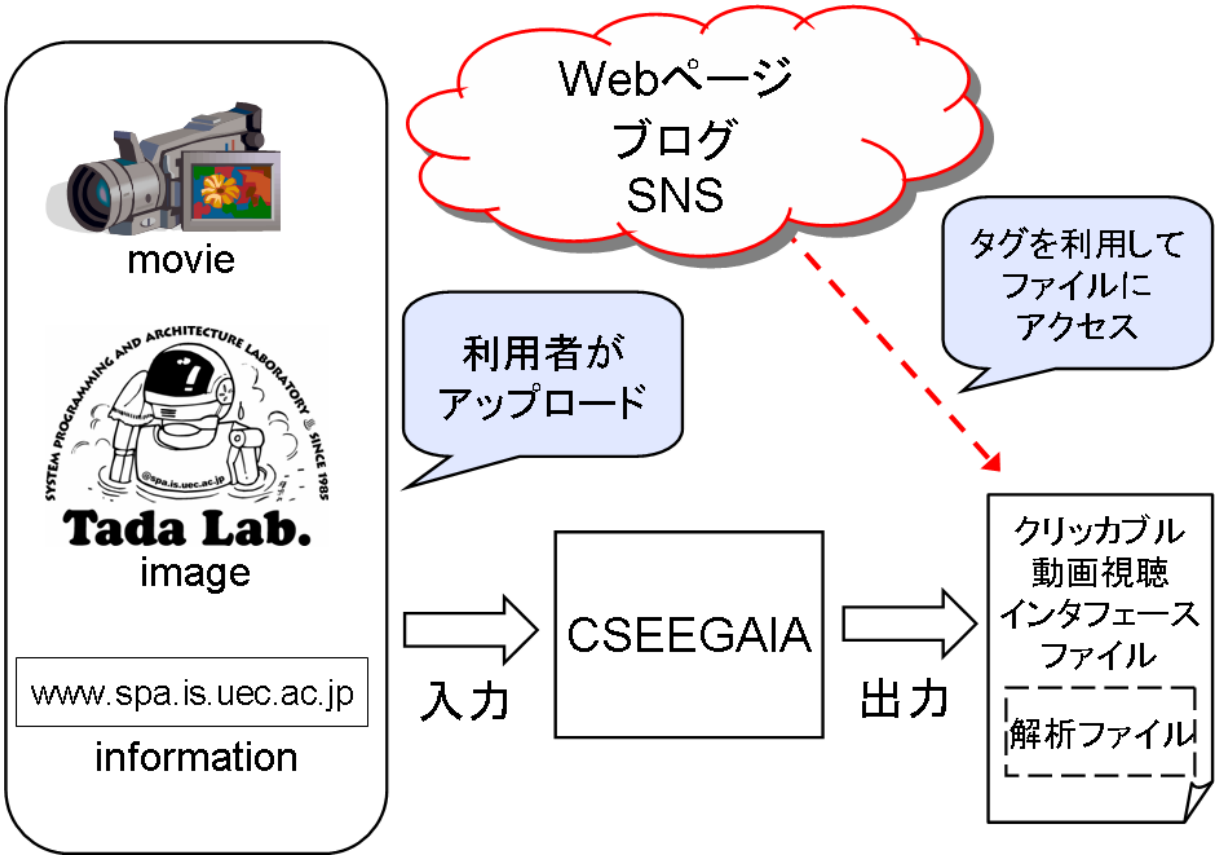

図 2.1: CSEEGAIA の処理イメージ

システムを利用する条件として、システムの利用者サイド、視聴者サイド共にブラウザに flash player plugin がインストールされている必要がある。また、システムの利用者はシステムに対して情報を与え、システムから返される結果を利用すれば良いだけであるので、本論文ではシステムの内部的な処理時間は考慮しない。

ここでいうシステムの利用者とは、自身の Web ページやブログ、SNS に掲載するために、本システムを用いてクリッカブル動画及びその視聴インタフェースを作る人のことを指す。また、視聴者とは、クリッカブル動画視聴インタフェースが掲載された Web ページやブログ、SNS などにアクセスして、実際にクリッカブル動画を視聴する人のことを指す。

なお、本論文ではイメージの例としてロゴや広告看板を使用し、動画中のそれらをクリッカブルオブジェクトとする [16]。また、関連情報として URL を使用する。

2.1 システムの利用方法

実際にどのようにシステムを使うのか、システムの利用者サイド、視聴者サイドに分けて例を挙げる。

2.1.1 利用者サイド

ここでは具体的なシナリオ例を挙げる。

- **シナリオ例**

とある企業に勤める会社員エヌ氏は、旅先で撮影した動画を自分の Web ページに掲載しようと思った。撮影した動画を自宅で見返していると、その動画中に偶然、エヌ氏の勤める企業の看板広告が映り込んでいる場面をいくつか見つけた。そこでエヌ氏は、自分の企業の宣伝も兼ねて、その看板広告をクリック可能にしたクリッカブル動画を掲載しようと思い、ブラウザから CSEEGAIA にアクセスした。すると CSEEGAIA は、動画とイメージ、関連情報の入力をブラウザ上で促してきたので、撮影した旅先の動画と企業の看板広告のイメージ、企業の URL をシステムに入力した。

エヌ氏が Web ページの記事を書きながらしばらく待っていると、システムの内部での処理が終わったのか、ブラウザを通して処理の終了音が鳴った。それと同時にブラウザに出力結果が表示された。ブラウザ上に出力されたコメントとタグに従い、そのタグをエヌ氏の Web ページに貼り付けた。エヌ氏がブラウザで自身の Web ページを確認すると、クリッカブル動画視聴インタフェースが掲載されており、クリッカブル動画が視聴できるようになっていた。

クリッカブル動画とその視聴インタフェースを作成するにあたり、エヌ氏がやったことは、CSEEGAIA に単に情報を入力し、返ってきた出力結果からタグを自分の Web ページに貼り付けただけであり、作業負担はほとんど無かった。

2.1.2 視聴者サイド

ここでも具体的なシナリオ例を挙げる。

- **シナリオ例**

単身で上京している大学生の息子を持つエムさんは、息子が大学でいったい何を勉強しているのか、日頃から気にかかっていた。息子に質問してもいつも適当にはぐらかされるだけで、結局聞けず仕舞いの状況が続いていた。

そんなある日、息子から電話がかかってきて、研究室内の様子を撮影したクリッカブル動画を、息子のブログに掲載した旨の報告を受けた。息子がブログを書いていたことすら知らなかったエムさんは、早速教えてもらったアドレスにアクセスした。

すると確かにブログに研究室内の様子が映し出された動画が掲載されていた。息子からの電話でクリッカブル動画と聞いていたので、動画中に映っていた研究室ロゴをクリックしてみると、ディー大学のティー研究室の Web ページに飛んだ。そこにはメンバーとして息子の名前も書かれていた。Web ページでその研究室の研究分野や内容を読むことで、ようやく息子がちゃんと大学に通って真面目に勉強していることが分かり、エムさんは一安心した。息子が何を勉強しているか、一通りの情報が得られたので、エムさんは見ていたクリッカブル動画をまた続きから観直し始めた。

息子の所属する研究室の実際の様子を動画で観ると同時に、その中で気になったものをクリックして情報の詳細にすぐにあたることができ、エムさんは満足であった。

第 3 章

システムの設計

この章では、システムの設計をシステムの利用者サイド、視聴者サイドのそれぞれに分けて述べる。

なお、本システムは、Web ブラウザからインターネットを介して利用可能な Web アプリケーションとして設計されている。

3.1 利用者サイド

この節では、利用者サイドの処理の流れについて述べる。

システムは、図 3.1 のように、段階的に処理を行う。

step1. 対象動画及びイメージ、関連情報の収受

利用者が対象動画とイメージ、関連情報をアップロードする形でシステムに入力を行う。Web アプリケーションとして設計するので、この処理はブラウザを通して行う。

step2. 対象動画のキャプチャ

システムは単位時間ごとに対象動画のキャプチャを行う。全てのフレームをキャプチャせずに単位時間を定めることで、画像解析に要する時間を大幅に短縮する。

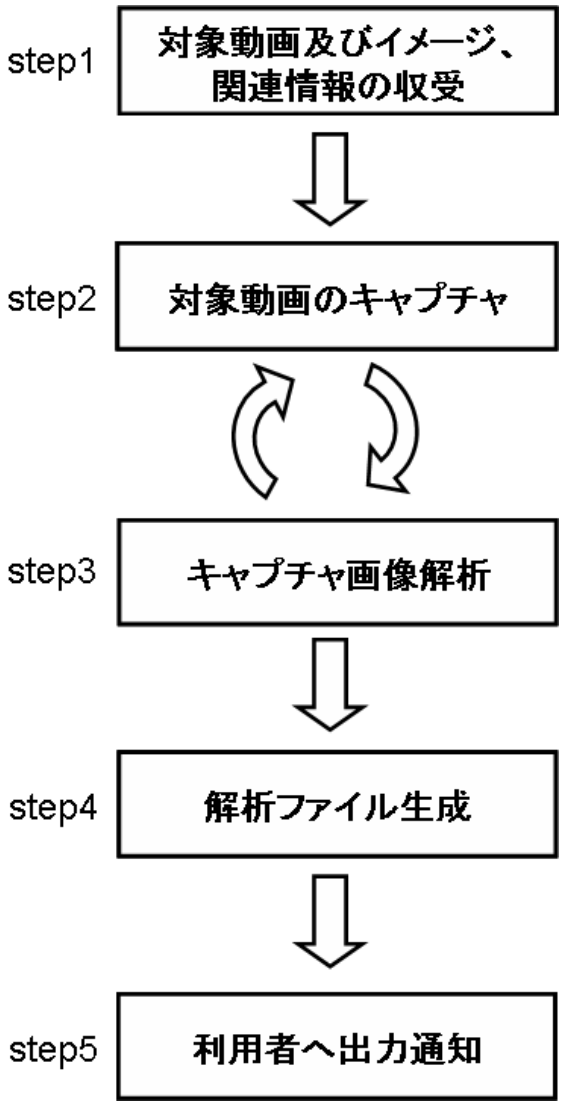

図 3.1: 利用者サイド処理の流れ

step3. キャプチャ画像解析

キャプチャしたフレームにユーザ提示画像が認識されれば、そのフレームの前後のキャプチャ時間間隔を短くして step2 に戻る。こうすることで、単位時間がある程度長い時間であっても、的確にフレーム内の認識対象を追うことができる。

この段階で、ユーザ提示画像とキャプチャ画像の対応付けに用いる特徴量を計算する。ここでは、Lowe によって提案された SIFT (Scale Invariant Feature Transform)

[8, 9, 10] を用いて、それぞれの画像の特徴計算を行うこととする。

SIFT は、回転やスケール変化、輝度変化に不変な特徴領域の設定と、それら領域内の特徴を記述する 2 段階の処理からなる。特徴領域の設定は、スケールスペース内の極値に基づく特徴点を利用して設定する。スケールとは、特徴をより表現できる領域の範囲を定めるパラメータのことである。記述子の計算は、輝度勾配方向ヒストグラムを利用して計算する。特徴量抽出には他にも、エッジ抽出に基づく方法 [13] や領域分割に基づく方法 [14] などが存在するが、その中から SIFT を選択した理由は、第一に、複数の特徴領域を利用することで、認識対象のある程度の視点変化や輝度変化、スケール変化や隠れに対処できるため、第二に、Mikolajczyk らの比較実験 [11] で他の手段の特徴量抽出よりも高い成果を挙げているためである。

step4. 解析ファイル生成

システムは画像解析処理後、解析ファイルを生成する。解析ファイルには、対象動画中にユーザ提示画像が表れた時間、位置、リンク先の URL 情報を含める。

step5. 利用者へ出力通知

全ての処理の終了後、利用者のブラウザに処理結果を通知する。システムが最終的な処理結果を出力するのにいくらか時間を要しても、その間別の作業ができるように、処理の終了は音を鳴らして知らせる。

こうすることで、処理の待ち時間に起因する利用者の心理的なストレスを軽減する。

3.2 視聴者サイド

この節では、視聴者サイドの設計について述べる。

クリッカブル動画視聴インタフェースは図 3.2 の 1 の部分のアイコン表示部、2 の部分の動画表示部、3 の部分のシークバー、4 の部分の動画制御部、5 の部分の音量調節部によって構成する。

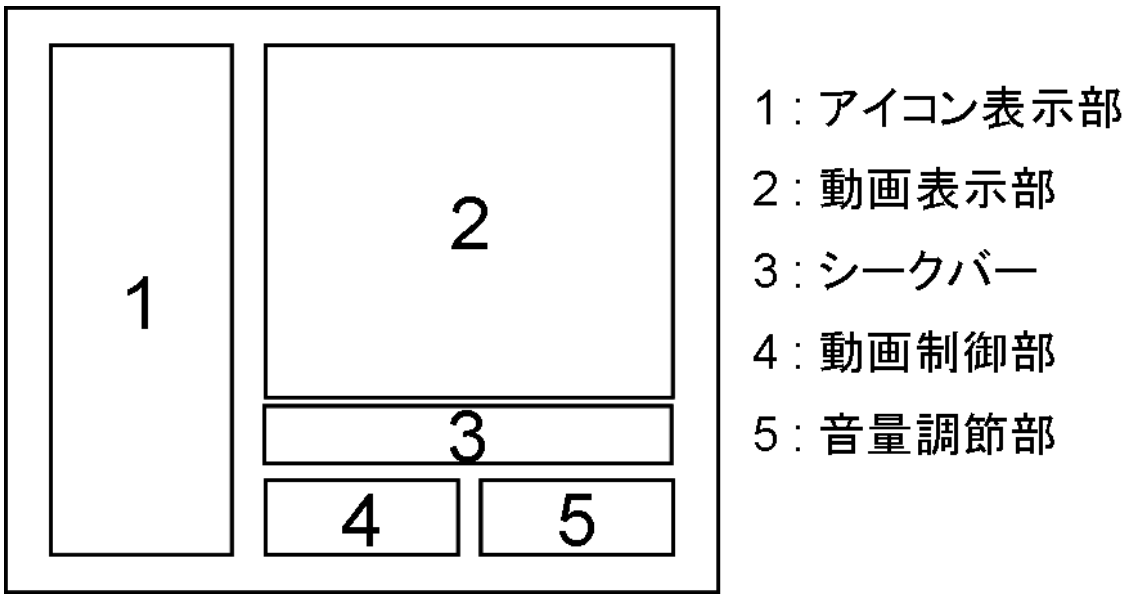

図 3.2: クリッカブル動画視聴インタフェースデザイン

アイコン表示部では、定常状態でデフォルトアイコン (図 4.6) を表示しておく。動画中にユーザ提示画像が認識された時点で、ユーザ提示画像を加工して作ったアイコンを表示し、動画中のどれがクリッカブルオブジェクトなのかということを視聴者に知らせる。ユーザ提示画像を基にしたアイコンはクリック可能であり、クリックされれば視聴者を関連づけられた情報へと導く。更に、そのアイコンを一定時間表示させることで、動画中のクリッカブルオブジェクトが一瞬しか映っておらず、クリックする暇がなかった、あるいはクリックし損ねたというような問題を

解決する。アイコン表示部に表示するアイコンの総数については、動画表示部の大きさとの兼ね合いで、上限は3つまでとする。

動画表示部では、視聴者が動画表示部中のクリッカブルオブジェクト近傍をクリックすれば、関連づけられた情報に導くようにする。

また、視聴者がシークバーを操作することで、動画を任意の時間から再生できるようにする。

動画制御部はplay&pauseボタン、stopボタンの2つで操作するようにし、音量調節部はスライドバーによって実現する。

3.3 フレキシビリティ

利用者がシステムにアップロードする動画ファイル形式及び画像ファイル形式は問わない。

システム利用者がアップロードするイメージの総数については、内部的な制限は設けない。しかし、クリッカブル動画視聴インタフェースで、アイコン表示の最大数を3つとしたことから、システム利用者がアップロードするイメージの総数も最大3つまでとする。

本稿ではアップロードするイメージの例としてロゴや広告看板を挙げるが、一面に渡る海や空、雲などといった特徴の少ない風景以外の物体なら何でも良い。このような抽象的な風景を除外する理由は、特徴量があまりに少ないため、利用者サイドの画像解析のステップで、ユーザ提示画像とキャプチャ画像の対応関係を上手く認識することが難しいためである。

第 4 章

システムの実装

4.1 実装概要

本章では、システムの実装について、設計に沿い、システムの利用者サイドと視聴者サイドに分けて述べる。

実装環境は Linux(Fedora Core 5)、サーバサイドで利用する Web サーバは Apache2.2.2 である。

利用者サイドの実装は、画像解析処理で C 言語を用い、それ以外の処理で PHP5.1.6 を用いた。プログラム行数は C 言語で書いた部分が 7000 行弱、PHP で書いた部分が 1000 行弱である。

視聴者サイドの実装は、クリッカブル動画視聴インタフェースで MXML (Macromedia Flex Markup Language) [17] を用い、インタフェースのイベント処理で ActionScript3.0[18] を用いて実装を行った。視聴者サイドの実装にかかったプログラム行数は合わせて 1000 行弱である。

また、動作確認のため使用したブラウザは、Firefox と Internet Explorer である。

4.2 利用者サイド

この節では、利用者サイドの実装の詳細について述べる。

4.2.1 [step1] 対象動画及びイメージ、関連情報の収受

システム利用者は図 4.1 のように、Web ブラウザを通して本システムに対象動画、イメージ、関連情報としてのリンク先 URL をアップロードする。

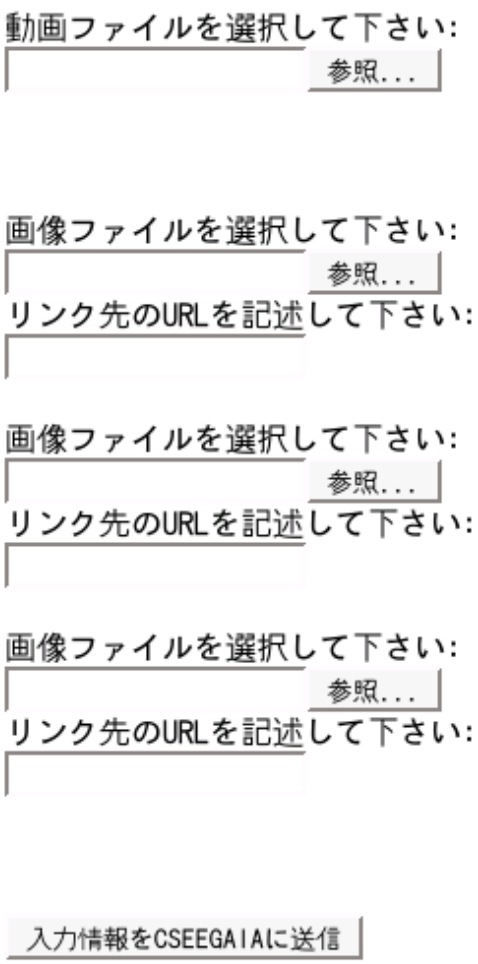

図 4.1: CSEEGAIA への情報アップロード画面

システムは、利用者が情報をアップロードした日時を基にして、CSEEGAIA のルートディレクトリ以下に新たにディレクトリを作成し、以後の処理を同ディレクトリ内で進める。

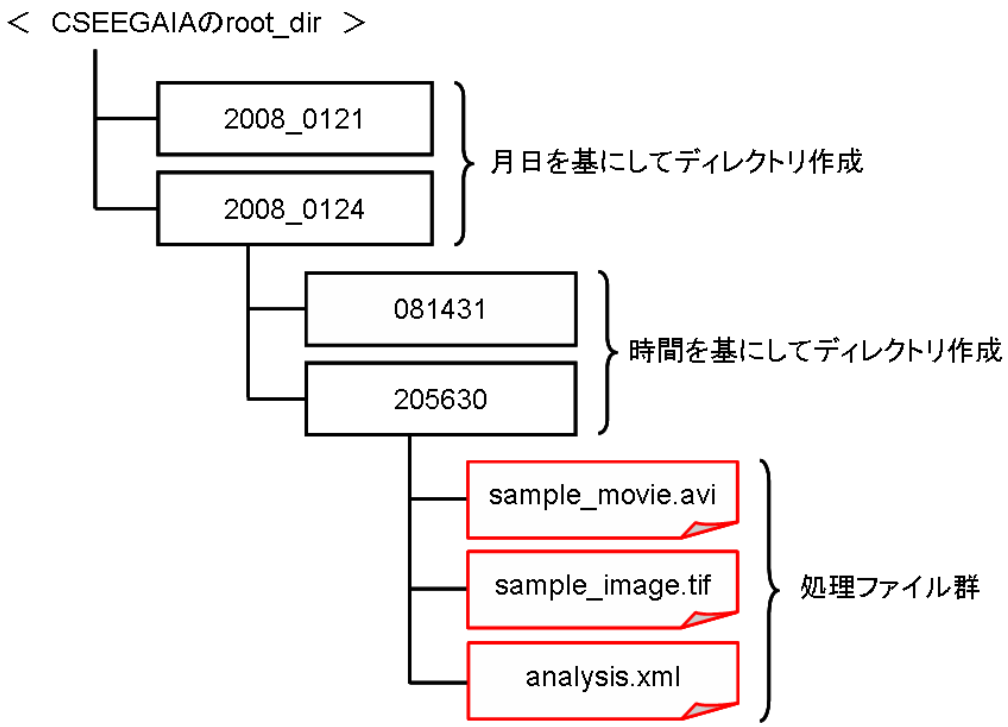

図 4.2: 作成されたディレクトリ及びファイルの例

図 4.2 で示してあるディレクトリ中のファイルは、sample_movie.avi がシステム利用者のアップロードした動画、sample_image.tif がアップロードしたイメージであり、analysis.xml が step4 の画像解析終了時にできる解析ファイルである。

4.2.2 [step2] 対象動画のキャプチャ

まず、利用者がアップロードした動画から単位時間ごとにキャプチャを行う。step3 でキャプチャ画像内にユーザ提示画像を認識したら、その前後の時間刻み幅を半分ずつ細かくしていき、認識対象を更に追う。こうすることで、動画中の認識対象の移動をより正確に追尾できる。一般の人々が撮影する動画に激しいカメラワークが伴うとは考えにくいため、終了条件を時間間隔が 0.2 秒より小さくなった場合と定める。

また、利用者からアップロードされた動画が、クリッカブル動画視聴インタフェースの画面で視聴できるように、終了条件を満たした段階で、動画ファイルの形式を flv 形式のファイルへと変換する。flv 形式のファイルに変換することで、swf (Shockwave Flash) 形式で出力されるクリッカブル動画視聴インタフェースファイル内に動画を埋め込めるようになる。

step2 と step3 に関して、具体的な例を挙げる。例えば 20 秒の動画に対して単位時間を 2 秒とした場合、0 秒から 20 秒まで 2 秒間隔であらかじめキャプチャを行う。ここで例えば 6 秒でキャプチャした画像にユーザ提示画像が認識されたとする。その場合、単位時間 2 秒の半分の時間である 1 秒間隔で 6 秒の前後のフレーム、すなわち 5 秒と 7 秒のフレームをキャプチャする。次に、5 秒と 7 秒のキャプチャ画像に対して画像解析を行う。5 秒のキャプチャ画像でユーザ提示画像は見つからず、7 秒のキャプチャ画像で見つかった場合、今度は 1 秒の半分の時間である 0.5 秒で 7 秒の前後のフレーム、すなわち 6.5 秒と 7.5 秒のフレームを切り出して画像解析を行う。このように、キャプチャ画像内にユーザ提示画像が認識された場合は、時間間隔が 0.2 秒より小さくなるまで step2 と step3 の処理を繰り返す。

4.2.3 [step3] キャプチャ画像解析

ユーザ提示画像とキャプチャ画像の特徴量を比較することにより対応点を得る。図 4.3 に画像解析の流れを示す。

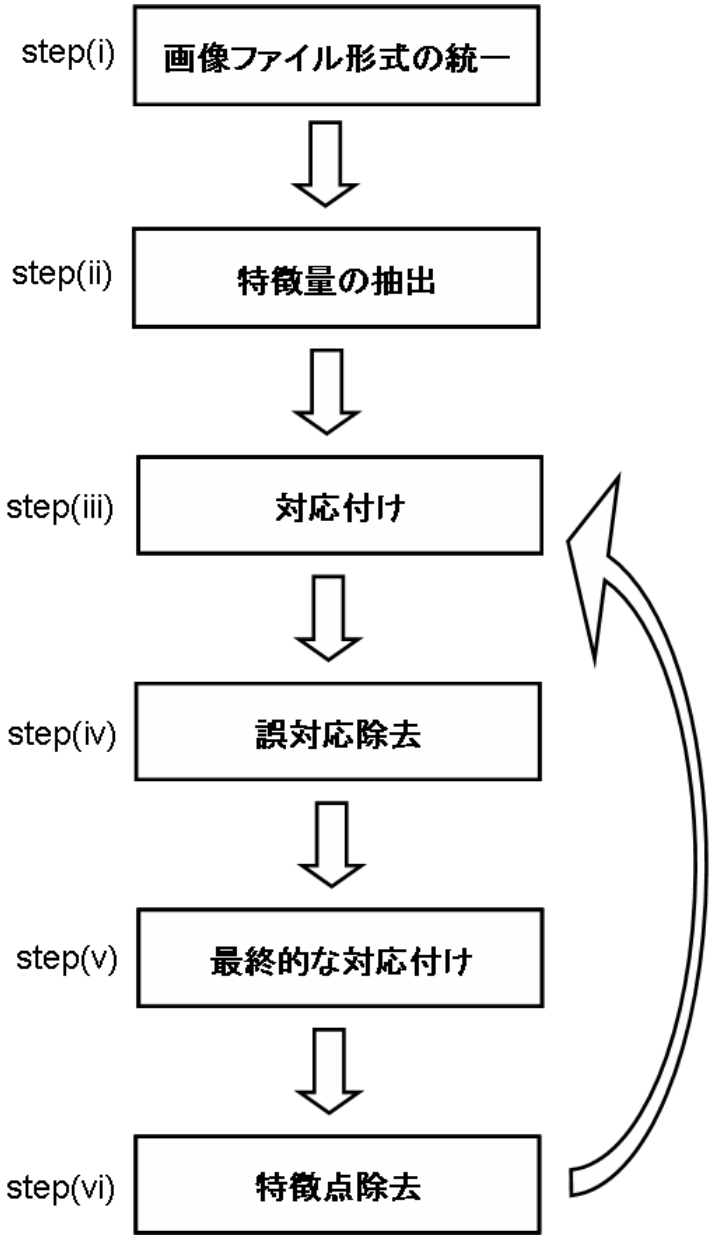

図 4.3: 画像解析部の処理の流れ

step(i). 画像ファイル形式の統一

この段階で、ユーザ提示画像とキャプチャ画像のファイル形式を統一し、特徴量の比較を容易にする。画質の劣化がない方が正確な対応関係が得られるため、圧縮することで画質が劣化する jpg などには変換せず、tif に統一して変換する。

step(ii). 特徴量の抽出

ユーザ提示画像、キャプチャ画像それぞれの特徴領域設定とその領域内の特徴を表す記述子を計算し、特徴量を抽出する。

特徴領域設定に用いる特徴点抽出には、DoG (Difference of Gaussian) によるスケールスペースの極値を利用する [8, 9, 10]。DoG はスケール σ の異なる平滑化画像間の差分である。スケール σ のガウス関数

$$G(u, v, \sigma) = \frac{1}{2\pi\sigma^2} exp\left(\frac{-(u^2 + v^2)}{2\sigma^2}\right)$$

と入力画像 $I(u, v)$ を畳み込んだ平滑化画像を

$$L(u, v, \sigma) = G(u, v, \sigma) * I(u, v)$$

とすると、DoG は

$$D_i(u, v, \sigma) = L(u, v, \sigma_{i+1}) - L(u, v, \sigma_i)$$

で表される。この注目画素近傍の DoG の極値を求めることで、特徴点の位置と、それが発見されたスケールを得る。そして、その特徴点のスケールに比例した局所領域を設定する。

記述子については、まず、見つかった特徴点の近傍領域での輝度勾配方向ヒストグラムを求める。各画素の勾配 $m(u, v)$ とその勾配方向 $\theta(u, v)$ は以下の式で求められる。

$$\begin{aligned} m(u, v) &= \sqrt{f_u(u, v)^2 + f_v(u, v)^2} \\ \theta(u, v) &= tan^{-1}\frac{f_v(u, v)}{f_u(u, v)} \end{aligned}$$

ここで、

$$\begin{aligned} f_u(u, v) &= L(u + 1, v) - L(u - 1, v) \\ f_v(u, v) &= L(u, v + 1) - L(u, v - 1) \end{aligned}$$

である。これらの式により計算したヒストグラムのピークが最も高い方向に、DoGで求めた特徴点を中心として特徴領域を回転させ、新たな座標系を作る。回転させた特徴領域を、この座標系で更にブロックに分割し、それぞれのブロックに対して再度、勾配方向ヒストグラムを作成する。そしてそのヒストグラムの各方向を特徴ベクトルとし、それを正規化したものを記述子とする。

step(iii). 対応付け

キャプチャ画像とユーザ提示画像の特徴量を比較して対応付けを行う。step(ii) において、特徴量をベクトルとして表現したため、距離が定義できる。そこで、それぞれの特徴間のユークリッド距離を算出し、それが最小となる点を対応点とする。

また、この段階で、対応付けに使う特徴点の数の上限を定めて処理時間の負荷を軽減した。

step(iv). 誤対応除去

step(iii) の結果に対して、ロバスト推定 [12, 15] を用いることで、外れ値を取り除き、誤対応の除去を行う。

step(v). 最終的な対応付け

step(iv) で誤対応を取り除いた結果に対して、検索領域を再度設定し、最終的な対応付けを行う。

step(vi). 特徴点除去

step(v) でキャプチャ画像内にユーザ提示画像が見つかった場合、キャプチャ画像内のその領域の特徴点を取り除き、再び step(iii) に戻り、キャプチャ画像内に認識対象が無くなるまでこれを繰り返す。この処理を加えることにより、キャプチャ画像内にユーザ提示画像が複数存在したとしても、正しい対応関係を得ることに成功した。

4.2.4 [step4] 解析ファイル生成

システムはstep3で行ったキャプチャ画像解析に基づき、動画中のいつ、どの範囲にユーザ提示画像が出現したのかを記述したXML (Exteneible Markup Language)形式の解析ファイルを生成する。生成するファイル形式にXMLを選択した理由は、まず第一に、視覚的にデバッグしやすいため、第二に、木構造であるため、クリッカブル動画視聴インタフェースの実装で用いるAction Scriptのイベントクエリに対して、ファイル内の各々のタグへのアクセスが容易になるためである。

実際にキャプチャ画像に出現するユーザ提示画像の倍率を、step3のキャプチャ画像解析処理で得られた対応点の組み合わせのうち2組の対応点を用いて計算し、動画表示部内のクリッカブルオブジェクト矩形近似座標を算出した。

また、解析ファイルには、利用者がアップロードした動画の名前やイメージの名前、それに関連づけられたリンク先URLの情報も併せて記述する。以下に解析ファイルのフォーマットを例示する。

解析ファイルフォーマット

```
<info>
 <movie> //動画に関する情報
  <path>動画までのパスを含むファイル名</path>
  <duration>動画の長さ</duration>
 </movie>
 <icons> //アイコンに関する情報
  <IconTotalNumber>ユーザ提示画像の総数</IconTotalNumber>
  <icon>
   <name>ユーザ提示画像の名前</name>
   <image>ユーザ提示画像までのパスを含むファイル名</image>
   <url>利用者が入力したリンク先の URL</url>
   <appearance>
    <time>
     <second>時間</second>
```

```
      <coordinate> //クリッカブルオブジェクト領域
       <left_upper_x>左上の x 座標</left_upper_x>
       <left_upper_y>左上の y 座標</left_upper_y>
       <right_lower_x>右下の x 座標</right_lower_x>
       <right_lower_y>右下の y 座標</right_lower_y>
      </coordinate>
     </time>
    </appearance>
   </icon>
  </icons>
</info>
```

<icons> タグで囲まれた部分の <IconTotalNumber> タグに示されるユーザ提示画像の総数に応じて <icon> タグが、それ以下の木構造も含めて増加する。もし、キャプチャした時間の画像にユーザ提示画像が見付からなかったら <coordinate> タグの中身は空とする。

また、<coordinate> タグ内の <left_upper_x> タグ、<left_upper_y> タグ、<right_lower_x> タグ、<right_lower_y> タグには、認識対象が見付かった場合に、それを矩形近似した時の座標値が入る。この <coordinate> タグにより、動画中に認識されたユーザ提示画像の近傍が対応づけられる。

実際の解析ファイルの例を以下に挙げる。例でのアイコン数は 2 つで、単位時間は 2 秒である。また、解析ファイル内に適宜コメントを記述してある。

実際の解析ファイル

```
<info>
 <movie> //動画に関する情報
  <path>http://www.spa.is.uec.ac.jp/~kazmit/P1050194.flv</path>
  <duration>18.7</duration> //動画の総時間
 </movie>
 <icons>
```

```
<IconTotalNumber>2</IconTotalNumber> //ユーザ提示画像の総数

<icon>      //1 個目のアイコン情報
 <name>tllogo2</name>  //1 個目のアイコンの名前
 <image>http://www.spa.is.uec.ac.jp/~kazmit/tllogo2.gif</image>
 <url>http://www.spa.is.uec.ac.jp/</url> //リンク先の URL 情報
 <appearance>
  <time>
   <second>0</second> //0 秒では tllogo2 は見つからなかった
   <coordinate>       //よって<coordinate>タグは空
   </coordinate>
  </time>
  <time>
   <second>2</second> //2 秒でも tllogo2 は見つからなかった
   <coordinate>       //よって<coordinate>タグも空
   </coordinate>
  </time>
  <time>
   <second>4</second> //4 秒で tllogo2 が見つかった
   <coordinate>       //4 秒のフレームにおけるアイコンの領域
    <left_upper_x>109.0460165</left_upper_x>
    <left_upper_y>134.180283</left_upper_y>
    <right_lower_x>119.0460165</right_lower_x>
    <right_lower_y>144.180283</right_lower_y>
   </coordinate>
  </time>
        //4 秒で tllogo2 が見つかったので、
  <time>  単位時間を半分の 1 秒にして、1 秒前の 3 秒のフレームを調べる
   <second>3</second> //3 秒でも tllogo2 が見つかった
   <coordinate>       //3 秒のフレームにおけるアイコンの領域
    <left_upper_x>9.9576113701901</left_upper_x>
    <left_upper_y>134.4217601937</left_upper_y>
    <right_lower_x>105.6075637564</right_lower_x>
    <right_lower_y>210.82139754849</right_lower_y>
   </coordinate>
  </time>
        //3 秒でも tllogo2 が見つかったので、
```

```
          単位時間を半分の 0.5 秒にして、
  <time>  0.5 秒前の 2.5 秒のフレームを調べる
   <second>2.5</second> //2.5 秒でも tllogo2 が見つかった
   <coordinate>          //2.5 秒のフレームにおけるアイコンの領域
    <left_upper_x>-24.148698382635</left_upper_x>
    <left_upper_y>143.72276827509</left_upper_y>
    <right_lower_x>74.899493488846</right_lower_x>
    <right_lower_y>222.75687141976</right_lower_y>
   </coordinate>
  </time>
        //2.5 秒でも tllogo2 が見つかったので、
          単位時間を半分の 0.25 秒にして、
  <time>  0.25 秒前の 2.25 秒のフレームを調べる
   <second>2.25</second> //2.25 秒でも tllogo2 が見つかった
   <coordinate>          //2.25 秒のフレームにおけるアイコンの領域
    <left_upper_x>-43.258415594841</left_upper_x>
    <left_upper_y>143.17053076115</left_upper_y>
    <right_lower_x>62.117743154921</right_lower_x>
    <right_lower_y>227.11035415673</right_lower_y>
   </coordinate>
  </time>
  <time>
                        ・
                        ・(中略)
                        ・
  <time>
   <second>18</second> //18 秒では tllogo2 は見つからなかった
   <coordinate>        //よって<coordinate>タグは空
   </coordinate>
  </time>
 </appearance>
</icon>      //1 個目のアイコン情報終わり

<icon>      //2 個目のアイコン情報
 <name>uec</name>  //2 個目のアイコンの名前
 <image>http://www.spa.is.uec.ac.jp/~kazmit/uec.gif</image>
 <url>http://www.uec.ac.jp/</url> //リンク先の URL 情報
```

```
<appearance>
 <time>
  <second>0</second> //0秒ではuecは見つからなかった
  <coordinate>       //よって<coordinate>タグは空
  </coordinate>
 </time>
 <time>
  <second>2</second> //2秒でuecが見つかった
  <coordinate>       //2秒のフレームにおけるアイコンの領域
   <left_upper_x>35.770174666008</left_upper_x>
   <left_upper_y>205.52448546705</left_upper_y>
   <right_lower_x>204.0964317555</right_lower_x>
   <right_lower_y>338.26601376444</right_lower_y>
  </coordinate>
 </time>
       //2秒でuecが見つかったので、
 <time>  単位時間を半分の1秒にして、1秒前の1秒のフレームを調べる
  <second>1</second> //1秒ではuecは見つからなかった
  <coordinate>       //よって<coordinate>タグは空
  </coordinate>
 </time>
       //2秒でuecが見つかったので、
 <time>  単位時間を半分の1秒にして、1秒後の3秒のフレームを調べる
  <second>3</second> //3秒でもuecが見つかった
  <coordinate>       //3秒のフレームにおけるアイコンの領域
   <left_upper_x>108.38316931551</left_upper_x>
   <left_upper_y>188.06884479892</left_upper_y>
   <right_lower_x>267.17494672453</right_lower_x>
   <right_lower_y>313.41882207366</right_lower_y>
  </coordinate>
 </time>
                    ・
                    ・(中略)
                    ・
 <time>
  <second>18</second> //18秒でuecが見つかった
  <coordinate>        //18秒のフレームにおけるアイコンの領域
```

```
      <left_upper_x>154.07992895234</left_upper_x>
      <left_upper_y>172.53899772783</left_upper_y>
      <right_lower_x>331.61419729923</right_lower_x>
      <right_lower_y>312.41898426385</right_lower_y>
     </coordinate>
    </time>
   </appearance>
  </icon>      //2 個目のアイコン情報終わり

 </icons>
</info>
```

4.2.5 [step5] 利用者へ出力通知

一連の処理が終わると、システムはブラウザを通して終了音を鳴らし、利用者にブラウザに結果が表示されたことを知らせる。これは、HTML の EMBED タグを利用し、デフォルト終了音をプログラム内に埋め込むことで実現している。投稿者のブラウザのアップロード画面は、図 4.4 のように結果の出力画面に遷移する。出力画面には解析ファイルを含んだタグと、それを Web ページやブログ、SNS に貼り付ける必要があるという旨の、簡単なメッセージが表示される。

図 4.4: ブラウザに表示されたタグ

実際に表示されたタグの内容例を以下に示す。

実際のタグの例

```
<embed src="http://www.spa.is.uec.ac.jp
 /~kazmit/2008_01221/224809/CSEEGAIA.swf"
 width=438 height=328 quality="high"
 pluginspage="http://www.macromedia.com/go/getflashplayer"
 type="application/x-shockwave-flash"
 FlashVars="FILE=http://www.spa.is.uec.ac.jp
            /~kazmit/2008_01221/224809/analysis.xml">
</embed>
```

上の例に表示されている項目 src の URL が、視聴者インタフェースファイルまでのパスであり、項目 FlashVars に記述されている URL が、step4 で生成された解析ファイルまでのパスである。

図 4.5: ブログへの掲載例

ブラウザに表示されたタグを Web ページやブログ、SNS に貼り付けると図 4.5 のようにクリッカブル動画視聴インタフェースが掲載される。図 4.5 のクリッカブル動画作成段階で与えたイメージは、アイコン表示部の最上段に表示されている電気通信大学のロゴと、アイコン表示部の真ん中に表示されている電気通信大学情報システム学研究科多田研究室のロゴである。

4.3 視聴者サイド

視聴者サイドの実装の詳細について述べる。ここでは、構成要素である動画表示部、アイコン表示部、動画制御部、シークバー、音量調節部に分けて述べる。実際に完成したクリッカブル動画視聴インタフェースは図 4.6、4.7 のようになる。実装するプログラムは swf 形式のファイルで出力する。swf ファイル形式で出力することにより、ブラウザに flash player さえプラグインされていれば、どのような計算機環境でも視聴が可能になる。

図 4.6: デフォルトのクリッカブル動画視聴インタフェース

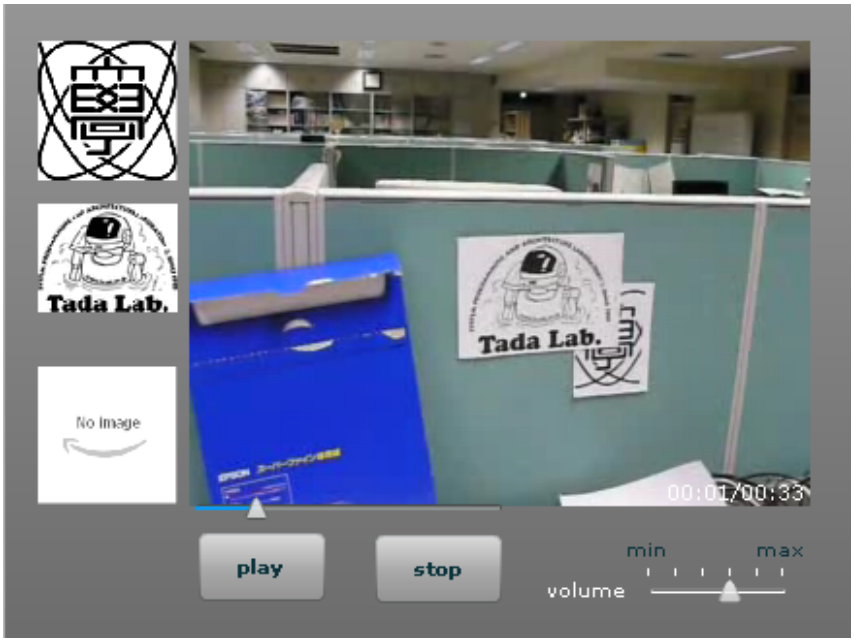

図 4.7: クリッカブル動画再生中の視聴インタフェース

4.3.1 動画表示部

動画表示部では、動画がロードされている時に限って、動画表示部上のマウスカーソルの移動イベントを常に捉える。カーソルの移動が起こる度に、イベントの起こった時間とスクリーン座標 (x 座標、y 座標) をキーとして解析ファイル内の最も近い時間 (<second> タグ) に問い合わせる。もし、解析ファイルに記述されてある該当時間内の <coordinate> タグに、クエリとして送られてきた x 座標、y 座標が共に含まれれば、マウスカーソルを矢印のアローカーソルから手の形をしたハンドカーソルへと変化させる。視聴者からは、クリック可能領域に入ったら、カーソルがハンドカーソルに変わったように見え、その部分がクリック可能領域であることが分かる。

クリック可能領域はクリッカブルオブジェクトの真上だけでなく、その周りを動画表示部の大きさに応じて矩形に近似し、「あそび」部分を付け加えることで、動

画表示部中のクリッカブルオブジェクトの移動が速くても視聴者が対象を容易にクリックできるようにした。

図 4.8 の赤丸で示したマウスカーソルは、クリック可能領域外であるためアローカーソルとなっている。図 4.9 の赤丸で示したマウスカーソルは、クリック可能領域内であるためハンドカーソルへと変化している。

視聴者からのクリックイベントを捕まえたら、マウスカーソルの移動イベントのクエリと同様に、時間と x 座標、y 座標をキーとして解析ファイル内の最も近い時間 (<second> タグ) に問い合わせる。その時間内の <coordinate> タグに x 座標、y 座標が共に含まれればリンク先の URL(<url> タグの内容) を返し、視聴者をリンク先に導く。

また、視聴者がリンク先にジャンプした場合、再生中の動画は一時停止状態になる。

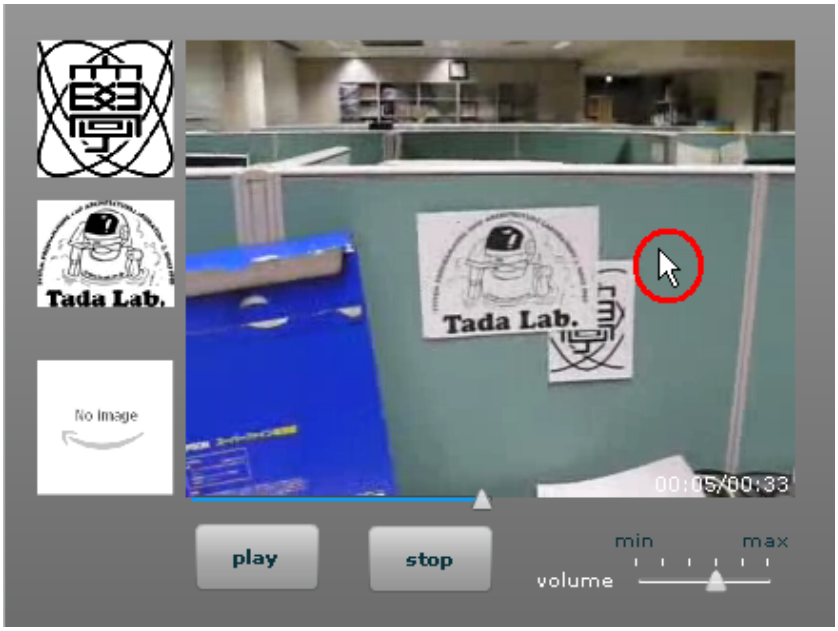

図 4.8: マウスカーソルがクリック可能領域外にある場合

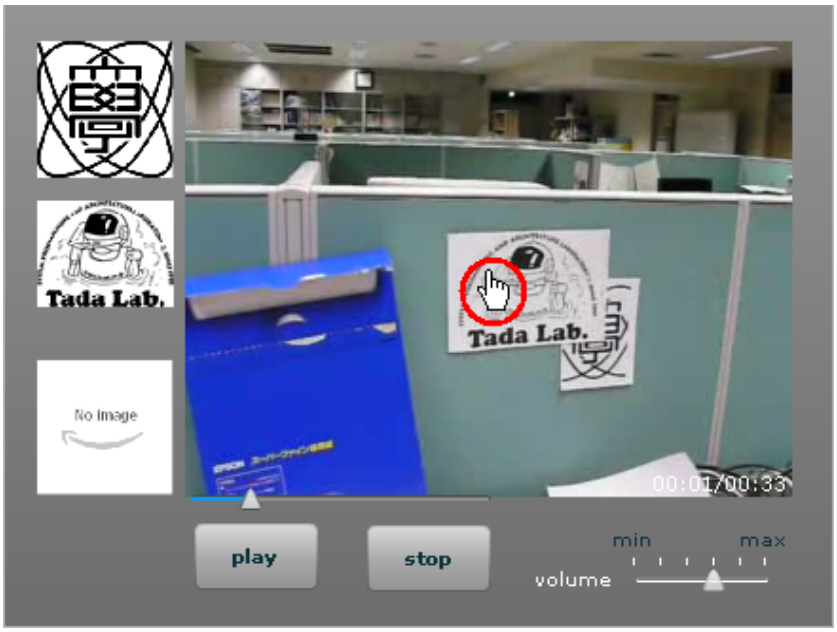

図 4.9: マウスカーソルがクリック可能領域内にある場合

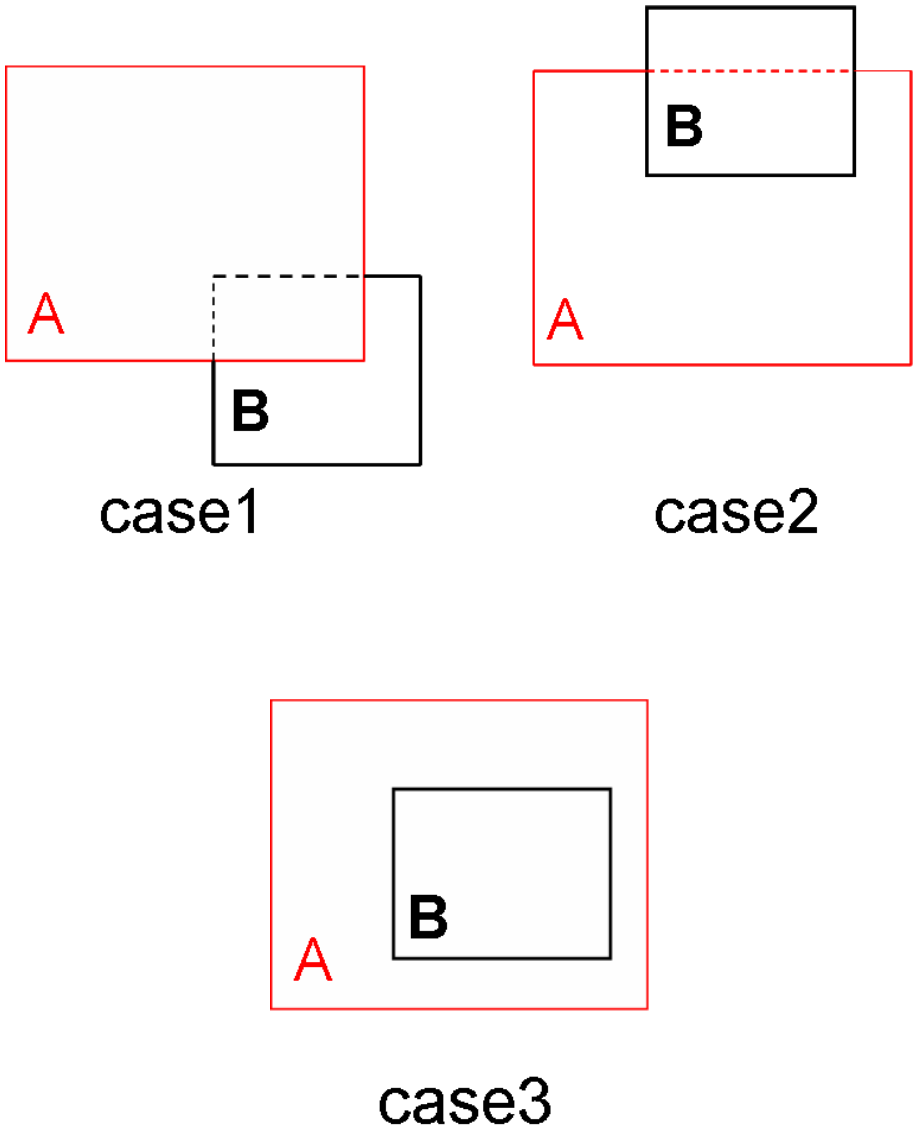

図 4.10: クリッカブルオブジェクトが重なっている場合

もし、クリッカブルオブジェクトが一部重なっている箇所をクリックした場合には、面積の小さい方を優先させる。解析ファイル中で、クリッカブル領域を矩形近似しているため、クリックされた時点で、重なっているそれぞれのオブジェクト面積の比較が容易に可能となる。

例えば、図 4.10 のような場合を考える。case1 のように A が前面、B が後面にあ

る場合、A と B が重なった部分をクリックした視聴者は、A よりも面積の小さい B に関連づけられた URL に導かれる。case2 のように A が後面、B が前面にある場合、A と B が重なった部分をクリックした視聴者は、case1 と同じく B に関連づけられた URL に導かれる。

このように、面積が小さい方の優先順位を高くすることにより、case3 のようにクリッカブルオブジェクト A の中に、それより小さいクリッカブルオブジェクト B が全て含まれている場合でも、視聴者が重なった部分をクリックしたら、B に関連付けされた URL に上手く導くことができる。

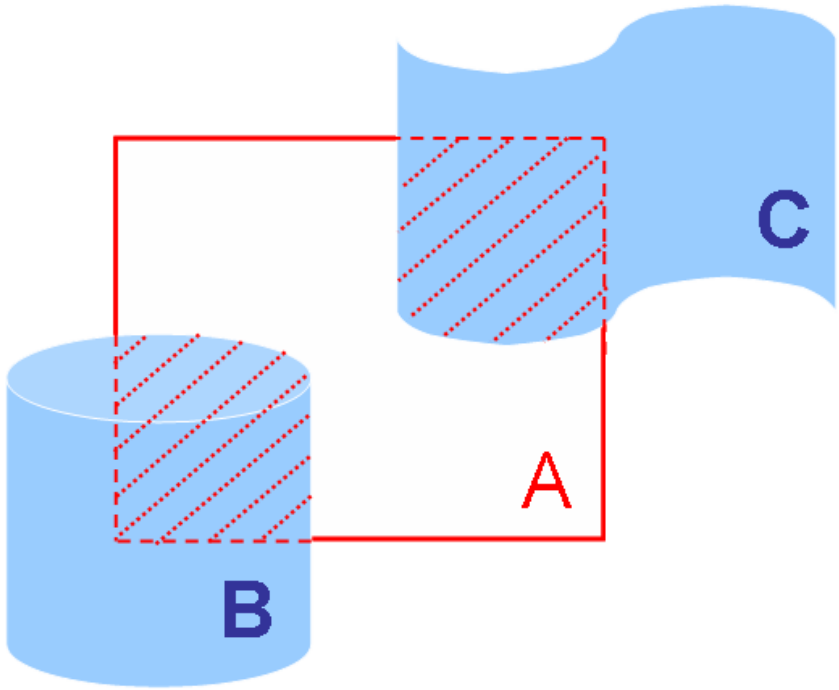

図 4.11: クリッカブルオブジェクトの一部が隠れている場合の図

クリッカブルオブジェクトが他のクリッカブルでないオブジェクトによって部分的に隠れている場合は、隠れている部分もクリック可能領域に含める。

例えば図 4.11 においてクリッカブルオブジェクト A は、A よりも前面にある物体 B と物体 C によって部分的に隠されている。その場合、視聴者のマウスカーソ

ルが斜線部に入った時に、アローカーソルからハンドカーソルに変え、隠れていてもクリック可能にする。実際のクリッカブル動画視聴インタフェース画面上の例を図 4.12 に示す。

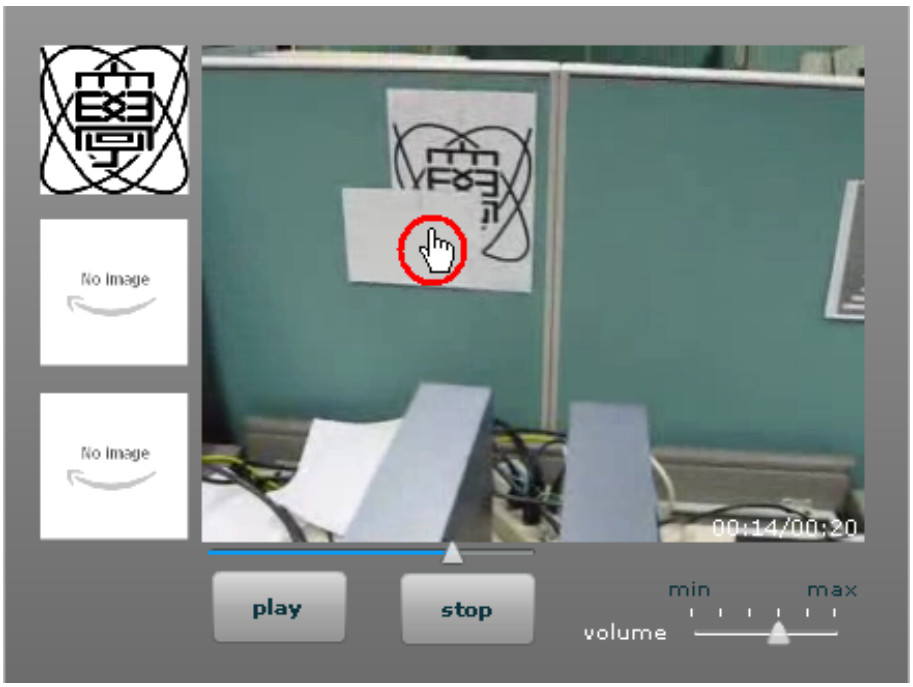

図 4.12: クリッカブルオブジェクトの一部が隠れている場合の画面

経過時間については、図 4.12 の動画表示部右下のように、経過時間と動画の総時間を秒単位で表示 (経過時間 / 総時間) することで視聴者に知らせる。動画の再生時間と、画面に表示する経過時間の対応を取るため、秒単位で動画にクエリを送り、クエリを送った時点での動画の再生時間を取得し経過時間を表示する。

4.3.2 アイコン表示部

アイコン表示部では、動画が再生されてない時はデフォルトアイコン (図 4.6) を表示する。

動画表示部にクリッカブルオブジェクトが表れた時点で、デフォルトアイコンから、ユーザ提示画像を加工して作ったアイコンに変化させる。そのアイコンは動画表示部中のクリッカブルオブジェクトと同様にクリック可能であり、クリックされた場合は、リンク先へと視聴者を導く。ユーザ提示画像を基にしたアイコンは、プログラムのパラメータであらかじめ設定された一定時間表示された後、フェードアウトし、デフォルトアイコンに戻る。これは、各々のアイコンにタイマを持たせることで実現した。

デフォルトアイコンにマウスカーソルが乗った場合は変化しないが、ユーザ提示画像を基にしたアイコンにマウスカーソルが乗った場合は、動画表示部と同様にハンドカーソルに変化させ、クリック可能であることを視聴者に知らせる。

図 4.13 の赤丸で示したマウスカーソルはデフォルトアイコン上にあり、アローカーソルのまま変化しないためクリックできない。図 4.14 の赤丸で示したマウスカーソルはユーザ提示画像を基にしたアイコン上にあるため、アローカーソルからハンドカーソルに変化し、クリック可能であることを示している。

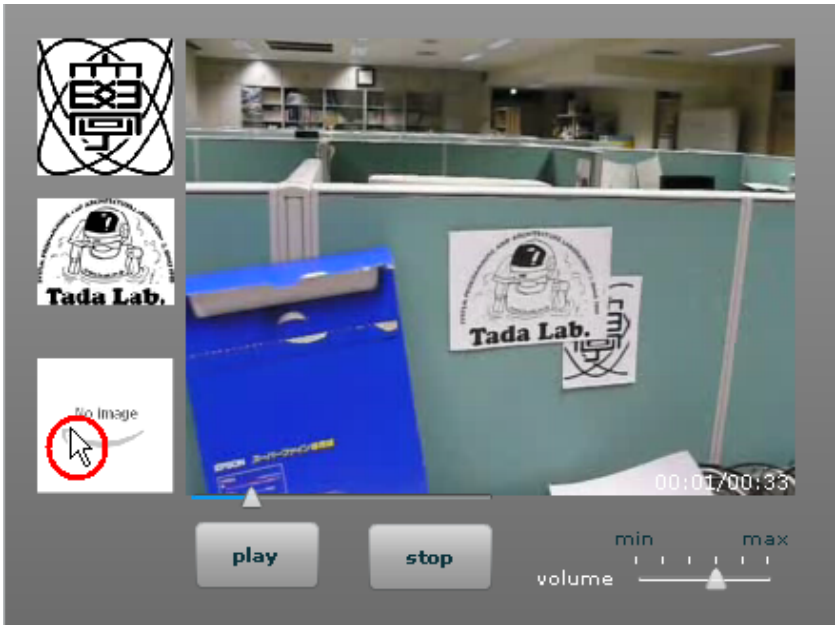

図 4.13: マウスカーソルがデフォルトアイコン上にある場合

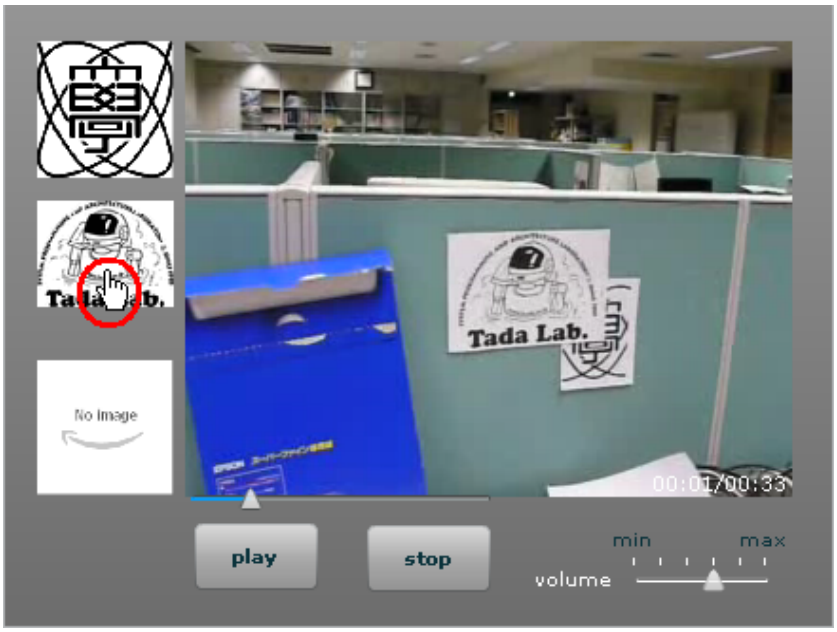

図 4.14: マウスカーソルがユーザ提示画像を基にしたアイコン上にある場合

pause ボタンが押されたり、視聴者がリンク先の Web ページに飛んだ場合は、ユーザ提示画像を基にしたアイコンを表示している部分のアイコンタイマも一時停止状態となる。一時停止状態から play ボタンが押された時は、あらかじめ設定された時間から一時停止状態となるまでに表示されていた時間を引いた、残りの時間だけ表示される。

もし、ユーザ提示画像を基にしたアイコンが表示されている状態で、新たに画面中に同じ認識対象が確認された場合は、その時点から更にそのアイコンの表示時間を一定時間伸ばす。

いつ、どのタイミングでユーザ提示画像を基にしたアイコンを表示するのか知るために、定期的に解析ファイルに対してクエリを送って問い合わせる。この問い合わせは、4.3.1 の経過時間表示のための問い合わせと同じタイミングで行い、最終的な問い合わせ総回数を削減している。クエリが送出された時間と、解析ファイル内のそれぞれの <icon> の最も近い時間が参照される。その <coordinate> が空でない場合に、対応する <image> がアイコン表示部のデフォルトアイコンに変わって表示される。アイコンは最大で縦に並べて 3 つ表示でき、ユーザ提示画像を基にしたアイコンの表示位置は、上方にあるデフォルトアイコンの位置から埋めていく。

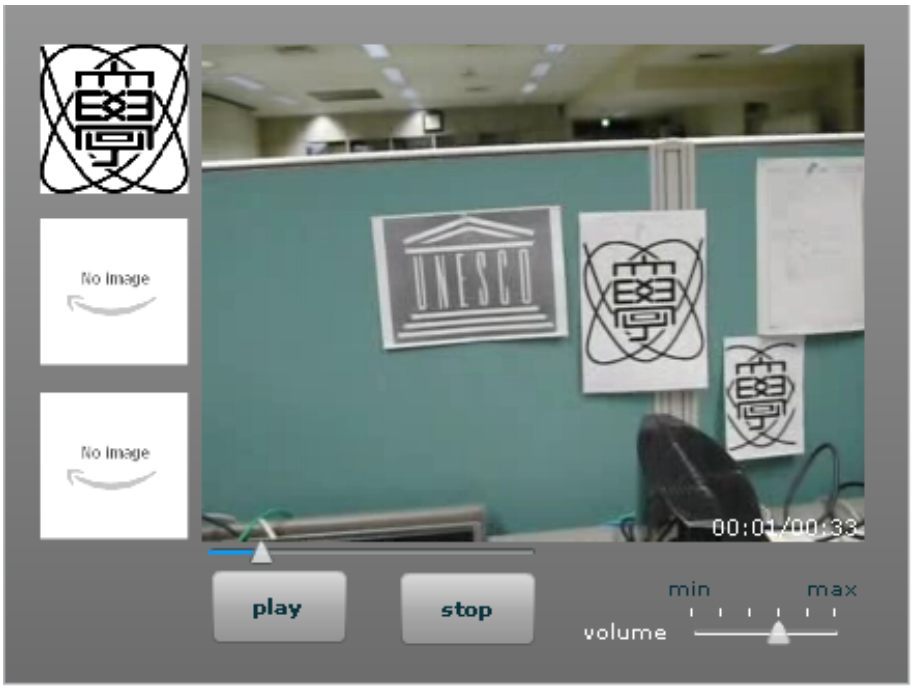

図 4.15: 同一イメージが画面中に複数存在する場合

図 4.15 のように、画面に同じ認識対象が複数映っている場合には、そのイメージに基づいたアイコンは 1 つしか表示しない。これは、アイコン表示部に表示されているアイコン名と、解析ファイル内のアイコン名を比較することで実現している。この例では、アイコン表示部の最上段に表示されている電気通信大学のロゴイメージのみをシステムに与えている。動画表示部では電気通信大学のロゴが 2 つ表示されているが、アイコン表示部では電気通信大学のロゴアイコンは 1 つの表示となる。ちなみに、動画表示部上の 2 つのロゴは共にクリッカブルで、どちらをクリックしても同じ URL に移動する。

4.3.3 動画制御部

動画が静止している時は、ボタンにそれぞれ play、stop が表示される。再生中は play ボタンが pause ボタンに変わり、pause と stop の 2 つのボタンが表示され

る。これは、ボタンが押された時点で再生、あるいは一時停止命令を動画に伝えると同時に、ボタンのラベルを変化させることで実現している。

pause が押された場合は動画は一時停止し、ボタンは play ボタンと stop ボタンになる。一時停止状態であっても、もしアイコン表示部にアイコンが表示されていて、動画表示部にクリッカブルオブジェクトが存在した場合は、どちらもクリック可能である。視聴者がクリッカブルオブジェクト近傍、あるいはアイコンをクリックしてリンク先にジャンプした場合は、自動的に一時停止状態にし、ボタンも play と stop に変わる。

stop が押された場合は、動画とシークバーは開始状態に戻し、アイコン表示部のアイコンはデフォルトアイコンに戻す。

4.3.4 シークバー

動画が停止中、一時停止中、再生中に関わらずシークバーを動かせるようにする。シークバーを動かしている間、ボタンの表示は pause と stop にする。また、シークバーを動かしている間は、その時点の時間が分かり易いように、シークバー上にポップアップで知らせる。

視聴者がマウスでシークバーを動かして見たい場面を決め、シークバーからマウスが離れたその瞬間を一時停止状態にし、視聴者が play ボタンを押すと同時に再生を再開する。もし、シークバーを動かす直前にアイコン表示部にユーザ提示画像を加工したアイコンが表示されていた場合は、シークバーが動かされると同時にデフォルトアイコンに戻る。

4.3.5 音量調節部

音量調節部はスライドバーで表示し、0〜10 までの範囲の音量操作とした。スライドバーを動かして音量調節している間は、シークバーの実装と同じく、その時点での音量をスライドバー上にポップアップで示す。デフォルトは 6.0 である。

第 5 章

評価

実装したシステムを日常的に計算機を使っている、年齢、性別、職業の異なる16名のユーザ (10代男性: 1名、20代男性: 4名、20代女性: 4名、30代男性: 3名、30代女性: 2名、40代男性: 1名、40代女性: 1名) に実際に使用してもらい、アンケート評価を行った。

被験者への質問項目は質問1〜4までの4つで、それぞれの質問項目に対して5段階 (1: いいえ、 2: どちらかといえば「いいえ」、 3: どちらでもない、 4: どちらかといえば「はい」、 5: はい) で評価してもらった。そしてそれらの項目に対して適宜コメントを記入してもらった。

アンケート実施者の負担を省くため、幾つか適当なサンプル動画とサンプル画像を事前に用意し、その中から選択して使ってもらう形でシステムを利用してもらった。

システムの処理に用いる計算機は、CPUが3.40GHzのPentium4、メモリが4G積んでいるマシンである。

サンプル動画は全て640x480のVGA、10fpsで撮影し、撮影時間は30秒前後である。それぞれの動画で複数のサンプル画像が数回、フレームインとフレームアウトを繰り返す場面が含まれる。また、システムが内部的にキャプチャする単位時間はデフォルトで2秒としている。

サンプル画像を1つアップロードし、それがサンプル動画に含まれなかった場合は約2分の処理時間がかかる。サンプル画像を1つアップロードし、それがサン

プル動画に含まれており、そのサンプル画像のフレームインが1回、且つ同キャプチャ画面にサンプル画像が複数存在しない場合は、約3〜4分の処理時間がかかる。このような場合は、利用者のアップロードしたサンプル画像の個数にある程度比例して処理時間が伸びる。また、サンプル画像のフレームインの回数、キャプチャ画像内に映っているサンプル画像の個数に応じて処理時間が伸びる。

ここで、サンプル動画以外の一般の動画に対して言及すると、撮影した動画の背景が複雑になればなるほど、処理時間が増大する。これは、背景が複雑になる分だけ特徴量抽出に時間がかかるためである。

質問1

システムを利用する立場から、アップロードした動画に対して自動で生成されたクリッカブル動画を最後まで通して見て、アップロードしたイメージはクリッカブル動画視聴インタフェース画面でうまく認識されていたか

この質問では、自動で生成されたクリッカブル動画が、手動でフレームごとに選択範囲を指定しタグづけをする動画と比べて謙遜ないレベルの認識結果で再生されているか否かを問うた。 **平均評価 ： 4.2**

この質問に対しては、おおむね満足した結果を得られた。コメントとして、

- 勝手に自動でクリックできる範囲を作ってくれて便利

という意見がある一方で、

- 実際に動画に対して手動でタグ付けしたことがないから、どの程度まで正確に判断できているか分からない

という意見も見られた。

質問 2

システムを利用する立場から、クリッカブル動画の生成は投稿者の計算機の専門的な知識は関係なかったと思うか

この質問では、クリッカブル動画の作成過程で動画のフレーム編集や、指定時間、指定位置へのタグづけといった、計算機に関わるある程度の専門知識が必要か否かを問うた。 **平均評価 ： 4.7**

充分満足した結果を得られた。この質問については、大多数の被験者が最高評価をつけた。被験者からのコメントとして、

- 一般的なユーザでも操作自体は問題ないと思う

というような意見が多数であり、この結果から、CSEEGAIA を利用すれば、専門的な計算機スキルや知識に関わらず、クリッカブル動画を生成できるという目標が達成できたことが確認された。

質問 3

視聴者の立場から、アイコンの表示やクリック可能領域でのカーソルの変化など、本システムのインタフェースのクリック可能領域の判別は分かりやすかったか

この質問では、クリッカブル動画を視聴する時に、いつ、どの範囲がクリック可能なのかを分かり易く視聴者に示せているか否かを問うた。 **平均評価 ： 4.2**

おおむね満足した結果が得られた。この質問については、数名の被験者から同じようなコメントが寄せられた。それは動画表示部で、

- マウスカーソルをハンドカーソルに変えるだけでなく、もっと分かり易いように、マウスカーソルがクリック可能領域に入った場合、動画画面上でそ

の領域の外周を目立つ枠で表示してはどうか

というものであり、ハンドカーソルに変化させるだけで充分満足という結果にはならなかった。

また、動画表示部で対象イメージが認識された場合、そのアイコンをアイコン表示部に一定時間表示し、それもクリックできるというのは好評であった。同時に、視聴者はアイコン表示部に表示したアイコンよりも、動画画面中のクリッカブルオブジェクトの方をよりクリックしたがるという傾向も見てとれた。

質問 4

自動クリッカブル動画生成システムに興味をもったか

この質問では、設計、実装したシステムが実際に世の中に出回った時に、どれくらいの需要があるのかという指標を知るために問うた。 **平均評価 ： 4.0**

おおむね満足した結果が得られた。男性からのコメントには、

- どうやってこのシステムを作ったのか知りたい

という意見が多かった。また、

- ビジネスチャンスがありそう
- 完成度が高い

という意見もあった。

女性からのコメントでは、

- クリッカブル動画を自分で作るのではなく、それが掲載されたブログを視聴するだけなら使いたいと思う

というものが大半であった。これらの結果から、本システムが世の中に出回っても十分に需要があると思われる。

以上のアンケート評価より、CSEEGAIA は充分実用に耐えうるシステムであることが裏付けられた。

第 6 章

関連技術

BML (Broadcast Markup Language) [5] は ARIB (社団法人電波産業会) によって策定されているデータ放送用記述言語で、図 6.1 のようなデジタル放送のコンテンツ製作で用いられる。この技術を用いれば、画面上で視聴者をコンテンツ製作者の意図したリンク先へ導いたりすることが可能である。しかし、あくまで技術仕様であり、一般の人々がコンテンツを製作するとは考えにくい。また、フルセグデータ放送が視聴可能な計算機は今現在、広く普及しているとはいえない。

図 6.1: デジタル放送の例 (nhk/digital より引用)

記憶媒体を用いる技術の例として、株式会社バンダイの特許 [6] に「表示出力装置、表示出力方法およびコンピュータプログラム」がある。これは、DVD などの記憶媒体に記憶されている動画データを再生した際に表示される商品オブジェクトについて、その実物を販売するサイトへネットワークを介して接続する技術を提供するものである。しかしながら、これも記憶媒体の動画データの商品に対してあらかじめ人手でタグづけし、データベースサーバに格納しておく必要がある。更にいえば、この技術は映画配給会社などのスポンサーの商品購入を促すために考えられたものであり、一般の人々が自由にリンク先を選んで記憶媒体にタグづけするといったことはできない。

図 6.2: like.com の類似商品画像検索画面

画像検索の技術に関して、Like.com[7] が開発した visual search engine がある。この技術は、これまで多く用いられてきた、画像に付されたメタデータを頼る手法ではなく、visual signature と呼ばれる特徴指数をイメージごとに算出し、比較することで似かよった画像を検索するものである。現在 Like.com は、アクセサリや時計、靴や服、それらに似ている商品の検索サービスを提供している (図 6.2)。利用者が形、カラー、パターン (柄やレイアウト) のパラメータを操作すれば、似かよった商品が提示される。検索対象になるこれら商品は、あらかじめ登録されてある商品に対してのみであり、今のところ、利用者自身で撮影した欲しい物の写真をアップロードして、それに似た商品を探すなどといったことには対応していない。

第 7 章

おわりに

本研究では、Web 上において、個人の計算機スキルや計算機に関する専門的な知識に関わらず、クリッカブル動画を自動で生成するシステム、CSEEGAIA (Clickable movie System with Easy and Effective Generator on Automatic Image Analysis) と、その視聴インタフェースの設計と実装を行った。

本システムの実装は Web アプリケーションとして行ったため、Web が使える環境であれば、ブラウザを通してどこからでもアクセスが可能である。利用者の手間をほとんどかけることなく、自動で動画中のイメージをクリッカブルオブジェクトとし、クリック先を利用者が指定した URL とするクリッカブル動画を生成するシステムを実現した。実際に、本システム利用者は、動画とイメージ、リンク先 URL のわずか 3 つの情報を用意するだけでよい。

また、システムの一連の処理が終わった時点で終了音を鳴らしてブラウザへの出力表示を利用者に知らせる仕様にしたことで、いつシステムの処理が終わるのか、常に利用者が気にかけている必要が無くなり、利用者がシステムの処理の終了を待っている時間を有効に使えるようにした。

更に、利用者にファイルを直接返すのではなく、タグを用いてシステムに置かれた処理ファイルへアクセスを許可することで汎用性を高めた。こうすることで、利用者からのファイルのアップロードを許していないサイトの Web ページやブログ、SNS であっても、タグを貼り付けるだけでクリッカブル動画視聴インタフェースの掲載を可能にした。

クリッカブル動画視聴インタフェースでは、アイコン表示部を設けて視聴者にクリッカブルオブジェクトを知らせたり、マウスカーソルがクリッカブルオブジェクト近傍に入った場合にアローカーソルをハンドカーソルへ変化するようにした。こうすることで、動画再生中にどの範囲がクリック可能領域なのかということを、容易に視聴者に知らせることが可能となった。更に、クリッカブルオブジェクトが画面に出現した時点で表示される、ユーザ提示画像を基にしたアイコンを一定時間表示させ続けることで、視聴者がクリックし損ねるといった状況を回避することに成功した。

最後に、システムを年齢、性別、職業の異なる様々な人達に実際に使用してもらい、アンケートを行った。その結果、システム CSEEGAIA が充分実用に耐え得るものであり、有用性があることを確認した。

今後の課題として、アンケートで指摘されたように、画面表示部でクリッカブルオブジェクト近傍に目立つ色で外枠を示してクリック可能領域をもっと視覚的に知らせるなど、利用者の提案を積極的に採り入れた、インタフェースの改善が挙げられる。

また、本システムを利用すれば、どのような動画からでもクリッカブル動画を作成できるが、利用者が CSEEGAIA にアップロードした元動画はシステム内部に保存される。そのため、例えば利用者が CSEEGAIA を用いて、Web 上で公開されている動画をクリッカブル動画にした場合、その元動画の著作権に絡む問題を解決する必要があると考える。

謝辞

本研究を遂行するにあたって、指導教員の多田好克教授と佐藤喬助教には日頃から熱心なご指導、そしてご鞭撻を賜わり、貴重なご助言を頂きました。ここに厚く御礼申し上げます。

また、本研究が行なえたことは、研究方針や研究内容について議論をし、共に研究生活をおくってきた多田研、小宮研、そして村山研の学生諸氏のおかげでもあります。これらの皆さんに深く感謝の意を表します。

参考文献

[1] YouTube,
http://www.youtube.com/

[2] Google Video,
http://video.google.com/

[3] Project Synvie,
http://synvie.net/

[4] ニコニコ動画,
http://www.nicovideo.jp/

[5] BML,
http://www.arib.or.jp/tyosakenkyu/kikaku_hoso/hoso_std-b024.html

[6] 株式会社バンダイ. 表示出力装置, 表示出力方法およびコンピュータプログラム. 特開 2006-209658. 2006-08-10.

[7] like visual search,
http://www.like.com/

[8] D. Lowe:“Distinctive Image Features from Scale-Invariant Keypoints,” Int.Journal of Computer Vision, Vol.60, No.2, pp.91–110,2004.

[9] D. Lowe:“Object recognition from local scale-invariant features,” Proc. Int. Conr. Comp. Vis., pp1150-1157, 1999.

[10] M. Brown and D. Lowe:“Recognising panoramas,” Proc. Int. Conf. Comp. Vis., Vol2, pp1218-1225, 2003

[11] K. Mikolajczyk and C. Schmid:“A performance evaluation of locao descriptors,” Proc. Int. Conf. Comp. Vis. and Patt. Recog., pp.384-393, 2003.

[12] R. Hartley and A. Zisserman:“Multiple view geometry in computer vision,” 2nd edition, Cambride University Press, 2003.

[13] K. Mikolajczyk, A. Zisserman and C. Schmid:“Shape recognition with edge-based features,” Proc. British Machine Vis. Conf., pp.384-393, 2003

[14] J. Sivic and A. Zisserman:“Video Google: a text retrieval approarch to object matching in videos,” Proc. Int. Conf. Comp. Vis., Vol.2, pp.1470-1477, 2003.

[15] D. A. Frsyth and jean Ponce 著, 大北 剛 訳:“コンピュータビジョン”, 共立出版株式会社, 2007.

[16] N. Ichimura:“Recognizing Multiple Billboard Advertisements in Videos,” 2006 IEEE Pacific-Rim Symposium on Image and Video Technology (PSIVT06), pp.463–473, 2006.

[17] MXML,
http://www.adobe.com/devnet/flex/articles/paradigm.html

[18] ActionScript 3.0,
http://livedocs.adobe.com/labs/as3preview/docs/