Relevance feedback using Head Movements

Student Name: Rajat Vikram Singh
Roll Number: 2008044

Indraprastha Institute of Information Technology
New Delhi

Advisors
Dr. Srikanta Bedathur (Primary)
Dr. Mayank Vatsa
Dr. Richa Singh

Submitted in partial fulfillment of the requirements
for the Degree of B.Tech. in Computer Science & Engineering

15-04-2012

Student's Declaration

I hereby declare that the work presented in the report entitled **Relevance Feedback using Head motion** submitted by me for the partial fulfillment of the requirements for the degree of *Bachelor of Technology* in *Computer Science & Engineering* at Indraprastha Institute of Information Technology, Delhi, is an authentic record of my original work carried out under guidance of **Dr. Srikanta Bedathur**. Due acknowledgments have been given in the report to all material used. This work has not been submitted anywhere else for the reward of any other degree.

.... **Place & Date: ....**
Rajat Vikram Singh

Certificate

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

.... **Place & Date: ....**
Dr. Srikanta Bedathur

Abstract

The ultimate aim of any search engine is to provide relevant search results to the user in order to satisfy his information need. Search results can be improved if the user provides a feedback to the system with the documents that he found useful. This is called relevance feedback. In this project, we have used head movements of the user as a form of implicit relevance feedback. The main reason for using head movements is that users don't have to explicitly mark out relevant results, the system uses involuntary gestures of the user, making it less tedious.
The project was divided into two parts - (i) Gesture recognition and (ii) Relevance feedback. Gesture recognition was achieved using optical flow and tracking features. We have used relevance feedback for Automatic Query Reformulation. The system worked well in both recognizing gestures and providing relevant results to the user based on the feedback. The users who used the system found it helpful but not fully unobtrusive. Using this study we can infer that better, improved and more relevant search results can be obtained by using user's gestures.

Keywords: information retrieval, machine learning, image analysis, relevance feedback, algorithms.

Acknowledgments

First and foremost I offer my sincerest gratitude to my supervisor, Dr. Srikanta Bedathur, who has supported me throughout my thesis with his patience and knowledge whilst allowing me the room to work in my own way. I would like to thank him for his encouragement and effort and without him this thesis, too, would not have been completed or written.
I would like to thank my co-supervisors, Dr. Mayank Vatsa and Dr. Richa Singh, for helping me throughout the project work and providing their valuable insights all along.
I would like to thank the participants who participated in the database collection and evaluation of the system, without any incentive. I would also like to thank my friends who encouraged and helped me when I got stuck.

Work Distribution

This thesis has been completed during the course of two semesters. The first semester was focussed on developing an gesture recognition system. The review of related work, algorithm development and results discussed in Chapters 2.1, 4.1.1, 4.1.2, 5.2 and 6.1 was done as part of this project work last semester. Some of the results are also from the first semester. Rest of the chapters dealing with the new approach of gesture recognition and the information retrieval aspects like developing search engine using Lucene and relevance feedback were done during the second semester.

Contents

Chapter 1

Introduction

In present times of interactive and intelligent computing, search engines and information retrieval have served as the backbone of internet revolution. Improvement in technology has led to an increase in online content which ranges from simple text, blog posts to images and videos. There has been an exponential increase in the number of pages providing information about how to change your email password to finer details about how a nuclear reactor works. But, to find something you have to look in the right place. This is where, information retrieval comes into picture. Information retrieval is an important part of modern computer science. As the name suggests information retrieval is the field of computer science which is concerned with searching documents and providing information using documents and its metadata, etc. Search engines are the most visible form of information retrieval systems around us. The goal is to present the most relevant set of pages that fulfils the information need of the user.

Search engines, in their most basic form, use a search query to understand the information need of the user. But with changing times, new ways of determining search results have been developed. Most of the search engines maintain a portfolio of the web pages that the user finds interesting and visits more often. Based on this information, search engines return more personalized results for the user. In this project, we used the movement of the user as an added modality. We used the movement of head as a feedback about the relevance of a search result. Now the search engine can retrieve pages depending on the search query and feedback from the users. For eg: If a person searches for "George Bush", maybe he wants to look at some jokes on George Bush or want to know about the life of George Bush. In this situation the search engine will retrieve both kinds of results. The user can now use his head movement to classify the set of documents that he found relevant. The system will then generate better and more relevant search results using this feedback. Similarly, there can be many more examples where relevance feedback can be helpful to get the most relevant results.

In this project, we focused mainly on obtaining a proof of concept, which would strengthen our belief that user emotions or gestures can be used to retrieve information relevant to users information need. We assume that the user nods "yes" for a relevant result and "no" for a non-relevant result. We developed our system to recognize this head movement as positive or negative feedback to search results. We tried two different approaches to recognize this gesture. We used a machine learning based algorithm which used video frames during the motion to find "features". This method is explained in detail later in the report. We then used an optical flow algorithm to recognize head movements because it not only improved the classification accuracy but also improved the speed. Using this feedback, we use our system to generate better, more relevant results for the user.

The report starts with a small summary of related work done on gesture recognition and use of emotions and gestures in the field of information retrieval and an in-depth discussion of the algorithms used, which is followed by the methodology of the project and the implementation details. The result section follows implementation details. We conclude the report with the conclusions and the future work. We also list the limitations of the project at the end of the report.

Chapter 2

Related Work

As stated earlier, the project is divided into (i) Gesture recognition and (ii) Relevance feedback system. First we start with the work done in gesture recognition and then move on to the usage of emotions and gestures in information retrieval.

2.1 Gesture Recognition

The importance of gesture recognition lies in its ability to help in building efficient human-computer interaction systems. Examples of its use range from sign language recognition through gaming to virtual reality.
Any recognition and classification approach faces these challenges: pre-processing, feature extraction and classification. In case of our work this will be: face detection in a facial image or image sequence, facial data extraction and facial movement classification.
Most previous systems assume that a full frontal face view is present in the image or the image sequence being analyzed, which gives us some knowledge of the location of the face. To give the exact location of the face, Adaboost algorithm is used to exhaustively pass a search sub-window over the image at multiple scales for rapid face detection [1]. Essa et al. [2] extract motion blobs from image sequences. Principal Component Analysis (PCA) is used to detect the presence of a face [3]. These blobs are evaluated using the eigenfaces method to calculate the distance of the observed region from a face space of some sample images [3]. In [4], Connected Components are used to find the position of the pupils. The biggest two connected components in the image are assumed to be the pupils. The position of these components is used to find the pupils in the next images. These are then matched to pre-defined templates of the eye-brow and eyes to detect features. Image subtraction and spatio-temporal filtering is used to find the areas of motion which helps to find the position of the face. HMM is also used to classify a motion when some sequence of events form a part of a motion. One such system was developed in [5], where the authors trained a HMM using the forward-backward procedure of the Baum Welch algorithm.
To extract the positions of prominent facial features, eigen features and PCA are used. The distance of an image from a feature space given a set of sample images via FFT and a local energy computation is calculated. Cohn et al. [6] first manually localize feature points in the first frame of an image sequence and then use hierarchical optical flow to track the motion of small windows surrounding these points across frames. The displacement vectors for each landmark between the initial and the peak Haar-like features have been used to detect face with good accuracy in [7].

For classification, [8] calculates ideal motion energy templates for each expression category and takes the Euclidean norm of the difference between the observed motion energy in a sequence of images and each motion energy template as a similarity metric. Another way which is closer to the natural interface is to use additional hardware. Hand-gesture detection systems use hand-gloves as additional accessory to get data [9]. Infra-red sensor with camera can provide the depth information, which can be used to segment bare hand from live video very accurately and efficiently. Hardware like Microsoft's Kinect and Sony's Move are being increasingly used to track hand as well as body motion. The major disadvantage of this method is its lack of easy accessibility, user needs to have an infra-red sensor equipped camera to use the system.
Optical flow is a technique used to find the direction of movement of an object. The optical flow methods try to calculate the motion between two image frames which are taken at times t and t+Δt.
Optical flow was used by Kenji Mase [10] to find the movement of various facial muscles. And since an expression is formed by moving these facial muscles, he was able to determine the expression. He used the gradient in texture as features for tracking. According to him, facial skin has the texture of a fine-grained organ and this helped in extracting the optical flow [10].
Optical flow was used for determining subtle changes in facial expressions by Kanade et al [11]. Their approach was based on the assumption that the intensity of a pixel would not change in the next frame and the movement would not be big so that the pixels can be found locally. The user marked the first frame of the face for facial features on the eyes, lips and nose etc. Then, optical flow was used to track these features in the next frame. The authors reported good accuracy of 92% which was higher than any of the algorithms available then [11].
Saeed and Dugale [12] used optical flow to track the movement of the head to move a cursor for physically challenged and blind people. They used the Viola-Jones detector to detect faces and then used a corner and edge detector to separate out the features. These features were then tracked in next frames using optical flow. To determine the movement the centroid of the points was taken. The change in position of the centroid was used to determine the movement. This approach forms the basis of our implementation [12].

2.2 Emotions/Gestures in Information Retrieval

Information Retrieval is a well established field in computer science research. The prime focus of any information retrieval system is to provide most relevant search results to the user. With more sophistication in information retrieval systems the need of including users emotion to provide better search results arises [13]. The field which is closest to this kind of research in terms of applicability is human-computer interaction (HCI) [13]. We can find applications of HCI all around us which shows that researchers have worked on making applications which would improve usability. Progress has been made in developing affective systems that are capable of recognising and appropriately responding to human emotions, and ultimately making human-computer interaction experiences more effective [13].
Emotions not only regulate our social encounters but also influence our cognition, perception and decision-making through a series of interactions with our intentions and motivations [14]. Work has been done in understanding emotions in ways which are interpretable and useful to computers. There are many theories on emotions and how emotions can be studied in the context of computer-related tasks. A detailed discussion about the classification of emotions and popular ways of measurement has been done in [14].
Till now we have talked broadly about how emotions are measured and how they are used in

studies. We will discuss the researches in which these methods have been used and the results of some such results. Studies have been done which help find the emotion of documents. For example, In [15] Kevin et al. have studied what emotions are triggered in news paper readers on reading the news articles. They used Yahoo Chinas news web-pages as the corpus for the study. The web-pages were divided into 7 categories depending on the input from the user which the user gave after reading the news article. The 7 categories were happy, angry, sad, surprised, heart-warming, awesome, bored and useful. Since useful is not an emotion, the authors found results both including and discarding the useful tag for the web-pages. For classifying the emotion of the web-pages they used a combination of feature extractors. The feature extractors were used in various combination to find the combination which gives the best results. The features were (i) all the Chinese character bi-grams that appear in the articles, (ii) Stanford NLP groups Chinese segmentation tool on the title and content of each article, (iii) the metadata of the articles, which are the news reporter, news category, location of the news event, time (hour of publication) and news agency etc, (iv) emotion categories of words [15]. It was found that the combination of all the four feature classes produced the best results. However, suffering from the obvious problem of specificity to chinese news articles it also has the problem of highly noisy data from the crowd, which is not very reliable [15].
In [16], Lopatovska argues that emotion correlates to information retrieval behaviors of a user. In other words, knowing one of them can help to find the other. In the experiment, she conducted a study in which the participants were given two search scenarios and they were asked to research them using the live Google search engine. They were guided by a questionnaire which presented the information needed, instructions and the text of the problem. She monitored the facial expressions of the searchers (participants) using the eMotion Recognition software, public release 1.65, which gives automated readings of the participants emotions analyzes video stream by constructing a three dimensional wireframe mesh over the recorded face, noting the positions of certain facial features (e.g., eye brow, corner of a mouth and eyes, etc), and feeding the readings into the classifier developed from a subset of the Cohn-Kanade database [16]. The author selected five 3 second intervals before the event and five 3 second intervals after the event to identify dominant emotion(s) within the various time intervals around selected search behaviors. A total of 12 search behaviors like left click, mouse scroll etc. were selected for the study [16]. The author concluded from the results that a search behavior is often accompanied by similar changes in user emotions. Using this result the author showed correlation between the emotions and search behaviors of the users.
In [17], Moshfeghi et al. have used emotions to diversify document rankings during document retrieval. Document diversification is done in information retrieval systems because it improves the effectiveness of the system as diversity avoids redundancy, resolves ambiguity and effectively addresses users information needs [17]. Considering the fact that documents also have a emotional content within, the authors have included it in the diversifying process. They have used Maximal Marginal Relevance to decide the next document to be ranked. The MMR equation takes a parameter (lambda) which controls the impact of emotional similarity on the selection of the next document [17]. Making some changes to the MMR model and including the average of the emotional similarity of all the previous selected gives the average interpolation approach (AVG-INT) [17]. AVG-INT is also used to determine which document will be ranked next. For determining the emotion the authors have used an emotion detection system which works on OCC model which specifies 22 emotional classes and 2 cognitive states for classification. The emotion detection system is referred without any explanation. The emotion detection system is sentence based and gives a emotion vector for all the 24 classes in binary [17]. For determining the emotion the emotion vector of all the sentences is averaged. Cosine similarity or Pearsons correlation is used for finding the similarity between the documents [17]. The authors came to

the conclusion that emotion classes improve retrieval effectiveness and obtained 20% gains for 30% of the queries over language model (LM).
In [18], Arapakis et al. studied the role of emotions in the information seeking process and the potential impact of task difficulty on users emotional behaviour. The users were given 3 tasks which they were asked to fulfill using the search engine provided during the research. The 3 tasks were given in order of increasing difficulty. The users actions were captured using a logging software which captured users clicks and link navigation etc. The expressions of the users were also secretly recorded using a concealed camera. The user was asked to fill questionnaires during various stages of the study to get information about the task, its difficulty and the emotional state of the user [18]. The results show that the user emotions changed with the task difficulty with irritation being the most common emotion in the most difficult task [18]. The changing emotion captured by the camera showed change in users emotion as the task changed. According to the author this emotion change can be used as a feedback mechanism to the information retrieval system.

Relevance feedback is of 3 types: (i) Explicit, (ii) Implicit and (iii) Pseudo. When the user knows that his feedback will improve the search results it is called explicit feedback. [19] In explicit feedback, users mark the relevance of a document either as a binary feedback or a graded value. Implicit feedback is inferred from user behavior, for example, time spent on a particular page, selecting a page to view its contents, or head or hand gestures. Pseudo relevance feedback automates the manual part of relevance feedback. It takes the top few documents returned by the search engine and takes them as relevant, and it uses this assumption to get more results [20,21]. In [19], Carpineto and Romano talk extensively about the kind of work done in relevance feedback using Automatic Query Expansion (AQE). According to the authors, AQE can be used to improve search results based on the feedback by the user about the relevancy of the document. The content of the assessed documents is used to adjust the weights of terms in the original query and/or to add words to the query. Relevance feedback essentially reinforces the system's original decision, by making the expanded query more similar to the retrieved relevant documents, similarly AQE tries to form a better match with the user's underlying intentions [19]. But, the authors also caution that including many terms may, in fact, reduce the precision of the system, ranking relevant results below irrelevant results.

Chapter 3

Algorithms Used

This section talks in detail about the algorithms used in the project.

3.1 Head Movement Detection

3.1.1 Support Vector Machines (SVMs)

In this algorithm, our main goal was to distinguish between two types of motions i.e. horizontal motion and vertical motion of the head. For this task, we used Support Vector Machines(SVMs) because of their wide-scale applicability in image based applications and good performance overall in 2-class classification.
Suppose we have data which can be represented in 2-dimensional plane, we need to find a linear function which will divide this plane such that the two classes are on either sides with the maximum confidence and least variability. Let the two classes be defined as $+1, -1$
Suppose the equation of that line is:

$$q(x) = w^T x + b$$

in the region between the two classes many line segments can be drawn which will divide the two classes, but out of those the plane with the maximum distance from the closest points on wither side will be the most suitable plane. For eg:

Now, given a set of data points:

$$(x_i, y_i), i = 1, 2, \ldots, n, \text{ where}$$
$$\text{For } y_i = +1, w^T x_i + b \geq 1,$$
$$\text{For } y_i = -1, w^T x_i + b \leq -1,$$

The margin width is:

$$m = \tfrac{2}{||w||},$$

that needs to be maximized, or minimize:

$$m' = \tfrac{1}{2}||w^2||,$$

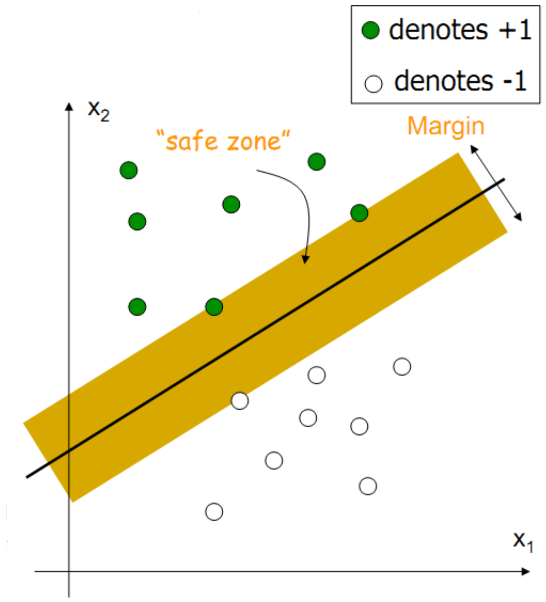


Figure 3.1: The linear function with the maximum length of the "safe" zone. Courtesy: [22]

such that,

$$y_i = +1, w^T x_i + b \geq 1,$$
$$y_i = -1, w^T x_i + b \leq -1,$$

or,

$$y_i(w^T x_i + b) \geq 1.$$

We can write this as a Lagrangian function, where we have to minimize,

$$L_p(w, b, \alpha_i) = \tfrac{1}{2}||w^2|| - \sum_{i=1}^{n} \alpha_i (y_i(w^T x_i + b) - 1) \text{ s.t. } \alpha_i \geq 0.$$

Differentiating w.r.t. w and b and equating to 0, we get

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i,$$
$$\sum_{i=1}^{n} \alpha_i y_i = 0.$$

Substituting the value of w in the equation above, we get a Lagrangian dual problem in which we have to maximize:

$$\sum_{i=1}^{n} \alpha_i - \tfrac{1}{2} \sum i = 1^n \sum j = 1^n \alpha_i \alpha_j y_i y_j x_i^T x_j,$$
$$\text{s.t. } \alpha_i \geq 0 \text{ and } \sum_{i=1}^{n} \alpha_i y_i = 0.$$

To solve this dual problem, we use KKT (Karush - Kuhn - Tucker conditions) to get the necessary condition, which is:

$$\alpha_i(y_i(w^T x_i + b) - 1) = 0,$$

which means that only support vectors have $\alpha_i \neq 0$.
. So, the solution is as follows:

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i = \sum_{i \epsilon SV} \alpha_i y_i x_i.$$

We can get b from:

$$y_i(w^T x_i + b) - 1 = 0,$$

where x_i are the support vectors (SV).
So, this makes the discriminant function (function which is used to distinguish between classes) as:

$$g(x) = w^T \phi(x) + b = \sum_{i \epsilon SV} \alpha_i \phi(x_i)^T \phi(x) + b.$$

$\phi(x_i)^T \phi(x)$ can be replaced by a kernel function. The use of kernel functions help in reducing the number of computations. The kernel function can be set by the user depending on the type of data and the number of classes.

3.1.2 Optical Flow

According to Horn and Schunck [23] who did the earliest works on optical flow in images, optical flow is the distribution of apparent velocities of movement of brightness patterns in an image. Optical flow can arise from the relative motion of objects and the viewer. In simpler words, optical flow is estimating the movement of a body by looking at the brightness patterns (pixel values of few interesting points).
The logic behind the algorithm is: Given a pixel in the first image, look for a nearby pixel in the second image with the same brightness. The assumptions which it takes into account are: (i) brightness is consistent in the frames and (ii) small motion between consecutive frames. These assumptions are the reason that it suffers from some problems, which are: (i) large displacements and (ii) changing illumination in frames.
For a 2D dimensional case a pixel at location (x, y, t) with intensity $I(x, y, t)$ will have moved by $\Delta x, \Delta y, \Delta t$, and between the two image frames, and the following image constraint equation can be given as:

$$I(x, y, t) = I(x + \Delta x, y + \Delta y, t + \Delta t).$$

Assuming the movement to be small, the image constraint at $I(x, y, t)$ with Taylor series can be expanded to get:

$$I(x+\Delta x, y+\Delta y, t+\Delta t) = I(x,y,t) + \frac{\delta I}{\delta x}\Delta x + \frac{\delta I}{\delta y}\Delta y + \frac{\delta I}{\delta t}\Delta t + H.O.T. \text{ (Higher Order terms)}$$

From these equations it follows that:

$$\frac{\delta I}{\delta x}\Delta x + \frac{\delta I}{\delta y}\Delta y + \frac{\delta I}{\delta t}\Delta t = 0$$

or

$$\frac{\delta I}{\delta x}\frac{\Delta x}{\Delta t} + \frac{\delta I}{\delta y}\frac{\Delta y}{\Delta t} + \frac{\delta I}{\delta t}\frac{\Delta t}{\Delta t} = 0$$

which gives:

$$\frac{\delta I}{\delta x}V_x + \frac{\delta I}{\delta y}V_y + \frac{\delta I}{\delta t} = 0$$

Thus:

$$I_xV_x + I_yV_y = -I_t$$

or

$$\Delta I^T.\vec{v} = -I_t$$

This is an equation in two unknowns and cannot be solved as such. To find the optical flow another set of equations is needed, by some additional constraint. All optical flow methods introduce additional conditions for estimating the actual flow. Some of the famous optical flow methods are Lucas-Kanade method [24] and Horn-Schunck method [23]. The difference between these two methods lies in the different assumptions of the smoothness in the motion field. We assume that the velocity is locally constant, and neighbouring points belong to the same patch that have similar motion because under well-lit conditions, the movement of all parts of the face would be similar. Because of this assumption we decide to use the Lucas-Kanade method.

Lucas-Kanade Method

The Lucas-Kanade method [24] assumes that the displacement of the image contents between two nearby instants (frames) is small and approximately constant within a neighborhood of the point p under consideration. Thus the optical flow equation can be assumed to hold for all pixels within a window centred at p. Namely, the local image flow (velocity) vector (V_x, V_y) must satisfy:

$$\begin{gathered}I_x(q_1)V_x + I_y(q_1)V_y = -I_t(q_1)\\ I_x(q_2)V_x + I_y(q_2)V_y = -I_t(q_2)\\ \vdots\\ I_x(q_n)V_x + I_y(q_n)V_y = -I_t(q_n)\end{gathered}$$

where $q_1, q_2, \ldots, q_n$ are the pixels inside the window, and $I_x(q_i), I_y(q_i)$ and $I_t(q_i)$ are the partial derivatives of the image respect to position x, y and time t, evaluated at the point and at the current time.
These equations can be written in matrix form $Av = b$, where

$$A = \begin{bmatrix} I_x(q_1) & I_y(q_1)\\ I_x(q_2) & I_y(q_2)\\ \vdots & \vdots\\ I_x(q_n) & I_y(q_n)\end{bmatrix}$$

$$v = \begin{bmatrix} V_x\\ V_y\end{bmatrix}$$

and

$$b = \begin{bmatrix} -I_t(q_1)\\ -I_t(q_2)\\ \vdots\\ -I_t(q_n)\end{bmatrix}$$

Multiplying both sides by A^T,

$$\begin{gathered}A^TAv = A^tb\\ \text{or}\\ v = (A^TA)^{-1}A^Tb\end{gathered}$$

As we know A and b we can compute velocity of movement v.

3.2 Information Retrieval

3.2.1 Relevance Feedback

Information retrieval part of the project uses the relevance feedback to take the feedback of the user about the importance of each document. The Rocchio algorithm is the classic algorithm for implementing relevance feedback. It models a way of incorporating relevance feedback information into the vector space model. We want to find a query vector, denoted as q_{opt}, that maximizes similarity with relevant documents while minimizing similarity with non-relevant documents. If C_r is the set of relevant documents and C_{nr} is the set of non-relevant documents, then we wish to find :

$$\vec{q}_{opt} = \arg\max_{\vec{q}}([sim(\vec{q}, C_r) - sim(\vec{q}, C_{nr})]),$$

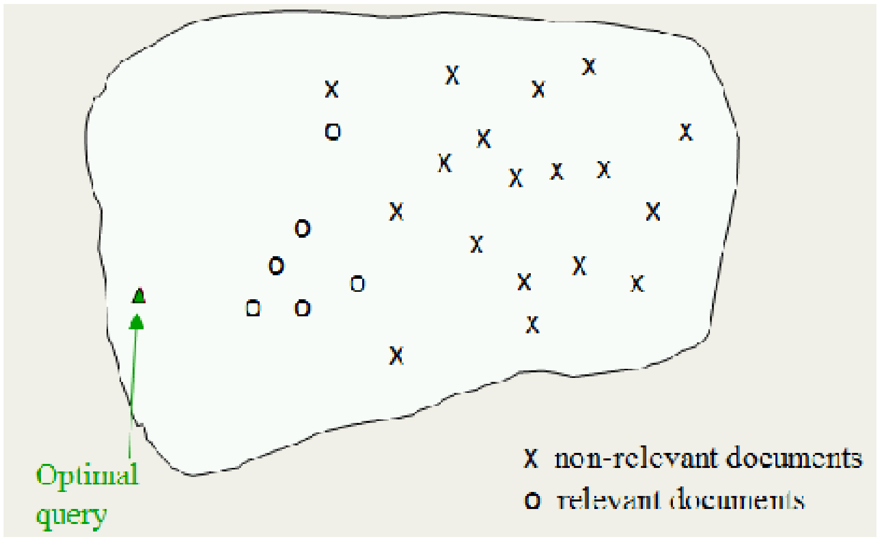


Figure 3.2: Finding an optimal query to separate relevant and non-relevant documents [16]

Under cosine similarity, the optimal query vector q_{opt} for separating the relevant and non-relevant documents is:

$$\vec{q}_{opt} = \frac{1}{|C_r|} \sum_{\vec{d}_j \epsilon C_r} \vec{d}_j - \frac{1}{|C_{nr}|} \sum_{\vec{d}_j \epsilon C_{nr}} \vec{d}_j$$

The equation above shows that the optimal query is the vector difference between the centroids of the relevant and non-relevant documents. Our approach is finding the "pseudo"-optimal query from the set of relevant documents given by the user.

$$\vec{q}_{opt} = \frac{1}{|C_r|} \sum_{\vec{d}_j \epsilon C_r} \vec{d}_j$$

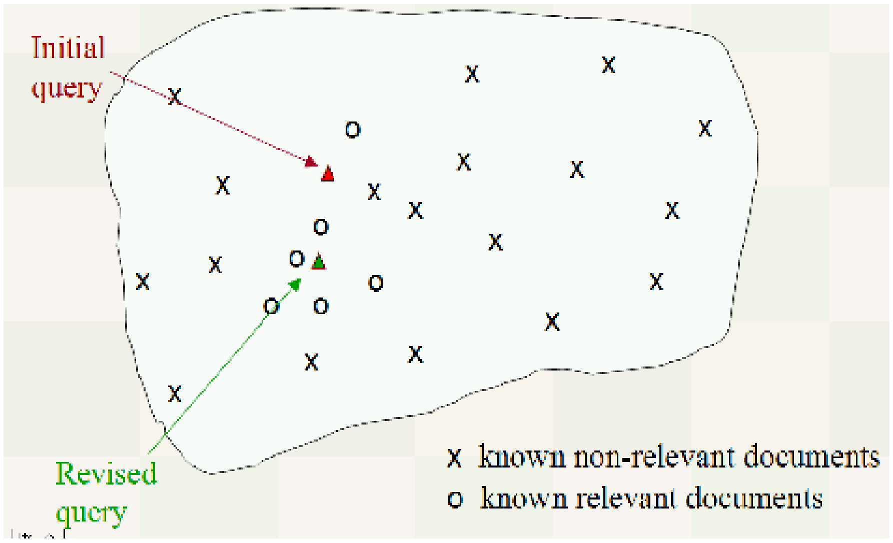


Figure 3.3: Finding an optimal query from the set of relevant documents [16].

Once we get this query we find results which match this optimal query.

3.2.2 Automatic Query Reformulation

It is derived from Automatic Query Expansion (AQE) where the query is expanded to include more terms to make the query more descriptive of the information need of the user. It is further derived from query expansion. The central question in this form of query expansion is how to generate alternative or expanded queries for the user. The most common form of query expansion is global analysis, using some form of thesaurus. For each term in a query, the query can be automatically expanded with synonyms and related words of from the thesaurus. In automatic query expansion, user's original query is increased by adding words that defines the information need in a better way. We have used a similar concept called automatic query reformulation where the query is reformulated from the original query to alternative queries the user may have intended. We use the relevant documents to get these terms and improve the search query.

Chapter 4

Methodology

The implementation was divided into two parts. One was the detection of the head movement and the other was the information retrieval system. On one hand we had to detect the motion and on the other hand we had to make sure that the results from the information retrieval system were relevant.

4.1 Gesture Recognition

The head movement detection part was further divided into two parts. One was the collection of the database and then developing the system for recognizing the facial movements.

4.1.1 Data Collection

A database in which 41 participants participated was created for use in the project. As part of the database collection the participants were asked to move their head in horizontal direction and then in the vertical direction. A video of 50 frames was captured for each movement (horizontal and vertical). The participant had the freedom of moving his/her head any number of times. The number of movements for one participant ranges from 1 to 3. The videos in the database were taken in a number of different places with different light and different distance using a Logitech C910 USB webcam capable of 1080p HD recording. The database was created by capturing images at 25 fps.

4.1.2 Machine Learning based Algorithm

As part of recognizing head movements, we started working on a machine learning based approach to recognize head movements.

Pre-Processing: Pre-processing includes doing some operations on the data so that it can be used for the purpose of feature extraction. In this project, pre-processing would refer to capturing frames of the movement of the person and then sequentially detecting and storing the faces from the frames. The movement was captured with the frames of the movement saved in

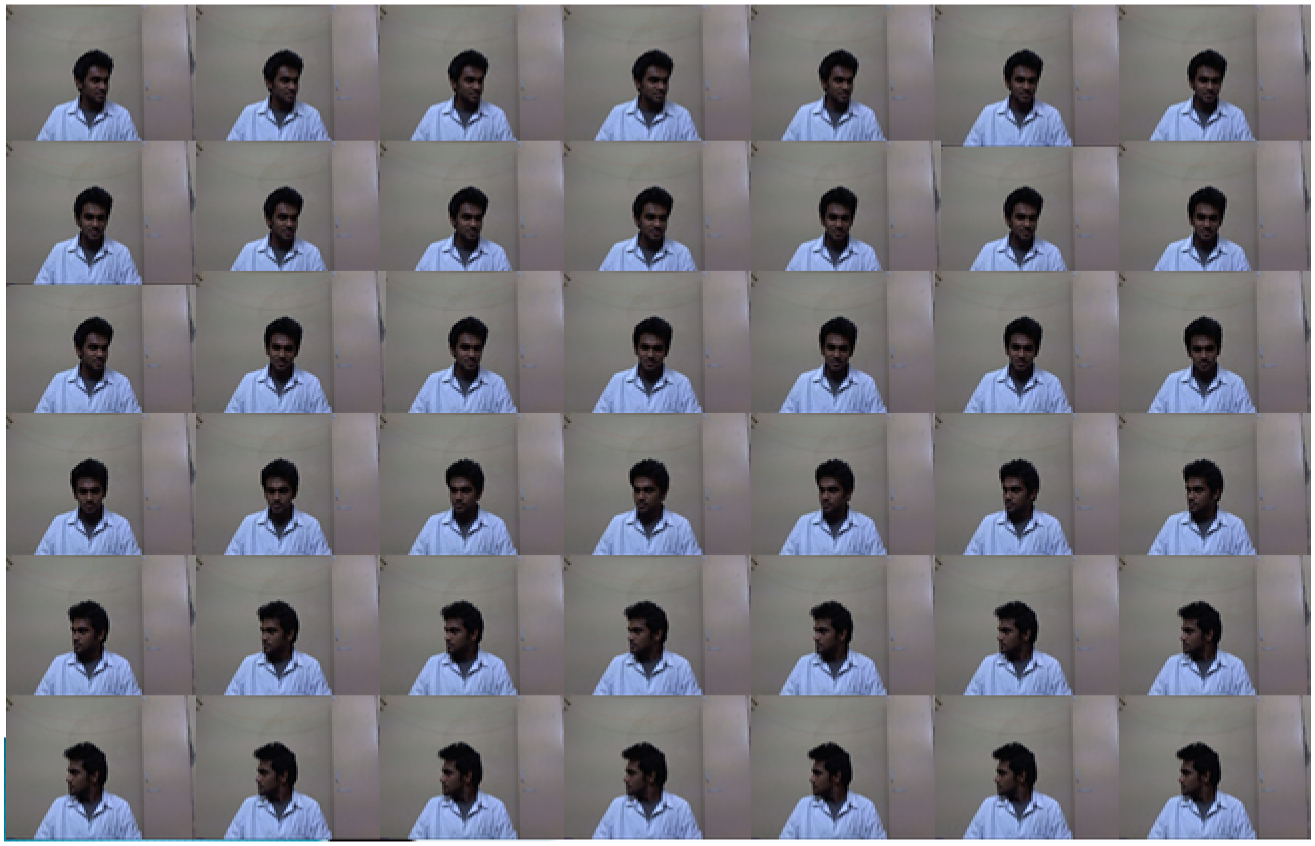


Figure 4.1: Example of data collection. This is the data for horizontal movement.

a storage device. The faces were first detected using the Viola-Jones face detector [1]. These faces were then saved in a separate location. Rest of the algorithm works on these faces and the rest of the frames are not used.

Feature Extraction: For the purpose of recognizing gestures, local binary pattern (LBP) [25] was chosen as the feature extractor. Local binary pattern is a texture based feature extractor which gives the information about the texture of the frame. Plain texture information in itself was not enough to work as features for detection because it did not indicate change or sequence of events. So we used the chi-squared distance criterion to find the distance between consecutive frames of the movement to form a feature vector, where each distance acts as a feature. The LBP from each detected face was calculated, then we find the chi-squared distance between the LBP's of (i) consecutive frames, (ii) first and the middle frame, (iii) middle and the last frame, (iv) first and the last frame. This set of distance was constituted in to a vector and was chosen as the feature vector for classification.

Classification: We defined the problem to be a two-class problem, the two classes being horizontal movement and vertical movement. We used a two class SVM classifier. Given the wide range of applications using SVM in image based recognition, SVM was chosen for the project. The features which we extracted in the step above were used for training the SVM. The training samples for a particular movement along with the proper label were used. The specifics of the kernel of the SVM were adjusted and the results were calculated using different permutations and combinations. The results are discussed in the results section.

Training: The features that we get from the participants are combined to form a feature vector. All the feature vectors are combined and along with the labels are given to the SVM. The parameters of the SVM are adjusted to optimize the classification accuracy.

Validation: Like the training set, features from the testing set are also given to the classifier along with the correct labels. The classifier then returns the labels which it calculates from the SVM and then compares with the correct labels and then calculates the accuracy of the system.

4.1.3 Optical Flow based Algorithm

Although we got decent accuracies with the machine learning algorithm, the time for feature extraction and classification made it very slow. This made the whole process of taking feedback from the user very tedious, which was not what we wanted feature for our information retrieval system. Besides this, since we were focussing on head movement recognition only for user feedback, it made sense to look for simpler algorithms which produced good results and took less time. Keeping this in mind, we started developing an deterministic algorithm using optical flow techniques.

The algorithm is described in points below.

Algorithm:

i) The faces are detected using the Viola-Jones face detector. This step is similar to the pre-processing step described above.

ii) From the face images of the subject, we find interest points using the Shi-Tomasi corner detection algorithm which are also known as "good features to track".

iii) Reduce the number of feature points polygon using convex hull. Find the centroid of this polygon. The centroid of the first frame will serve as the reference centroid for others.

iv) Give this set of points to the optical flow algorithm. The optical flow algorithm will find the set of these points in the next frame.

v) Find the centroid of the points in the next frame. Compare this to the reference centroid.

vi) Repeat this for all frames.

vii) Depending on the relative position of the centroids determine if it is a horizontal or a vertical movement.

4.2 Relevance Feedback

In order to test if input from the gesture recognition system can be used to provide relevance feedback to a information retrieval system, we had to built a IR system of our own. This part of the research was divided into two parts: (i) Development of a IR system (search engine) and (ii) Developing ways to improve search results using the feedback from the user.

4.2.1 Information Retrieval System

The information retrieval system was built using the standard Lucene library available in Java. The corpus which was used in the project was a freely available database [26] which was created from a snapshot of the english Wikipedia, downloaded from static.wikipedia.org in early September 2008 and contains approximately 121,790 documents. All pages containing the words Talk, Category, Portal, Template, User, Image, or Wikipedia in the URL were removed from the snapshot, as well as all redirect pages. This snapshot was then sampled uniformly to create the collection. This collection is a 5% sample from everything [26].

Indexing and Searching: All the documents of the corpus were indexed using the standard analyzer and an index was created. This index-building exercise was done only once and was used for the entire project.
When a search query is fired, this index is searched using the same analyzers and the results are returned.

4.2.2 Search Results Improvement System

Out of the results returned by the application, the user can select a single (or multiple) search results and then provide input with the movement of his/her head. The vertical movement of the head which coincides with the general movement of the head when a person says “yes” is taken as a positive feedback and vice-versa. This feedback is given to the system which then uses these results to find out similar results for each of the positive result.
For measuring the performance of the system, the system was tested with a set of test queries and a user was asked to mark the number of relevant results that the system retrieved. The test queries were ambiguous in nature with atleast two different possible meanings. The user can choose whichever meaning he wanted the search engine to return the results for. Depending on his choice, he would then choose the relevant documents and give that as feedback using the gesture recognition system. Then, the system will use the algorithms described below to generate new results, and display in front of the user. The user will then again mark the set of relevant documents. The precision of the system will be evaluated using this feedback.

Using MoreLikeThis:
In order to give more relevant results to the user, we used the MoreLikeThis tool provided by Lucene [27]. MoreLikeThis takes a document as input and returns documents which are similar to this, hence the name MoreLikeThis.
This approach gives good results with relevancy of the results improving considerably but it suffers from a few problems. They are: (i) MoreLikeThis takes just one document as input. So, we have to repeat this operation for all the documents separately, giving each document as input and storing the results. We then take the intersection of the relevant documents to get our answer, (ii) The second problem was that MoreLikeThis forms a new query depending on the set of documents and it doesn’t use the original query for the formulation of the new query. So although this approach improved results, we didn’t use it.

Using Automatic Query Expansion (AQE):
We needed an alternate approach to improve the original query, so we used automated query reformulation to change the old query depending on the set of relevant documents provided by the user [19]. We decided to use the tf*idf scoring criterion to retrieve important words from the relevant documents. The algorithm is explained in detail below :

i) We find all the terms from the set of relevant documents. We use it to find the term frequency ($tf_{(t,d)}$) of a terms t in a document d. We find the term frequencies of the terms in the relevant documents.

ii) We then find the inverse document frequency (IDF) for these terms in all of the collection. We use IDF because it returns higher value for rare terms. The terms which occur in fewer documents (rare) are better indicators of topic, so this method returns larger values for rare terms, and smaller values for common terms.

iii) Multiply the term frequency and the inverse document frequency to find a score of all the relevant terms in a document.

iv) Repeat it for all the relevant documents. Sort them according to their score. Store the top 25 terms from the sorted list.

v) Use it to reformulate the old query i.e. merge all the queries to form a new query. Give the original query a higher boost value to give it more weight.

vi) Use this reformulated query to search the index and return search results.

The search results are expected to be more relevant because it uses the representative terms from the relevant documents marked by the user. The search results from this new query are displayed to the user to take final feedback regarding the relevancy of the results. No other iteration is done after this step, this feedback is just to check the performance of the system.
It may be argued that the frequencies for whole set of the documents can be used for finding matching documents with the relevant set of documents. But since we are using the same measure for all the documents it would not matter [19].

Chapter 5

Implementation

In this chapter, the implementation details of the project will be discussed in detail, with some code snippets to make the functioning clear.

5.1 Software framework

Basic system structure:

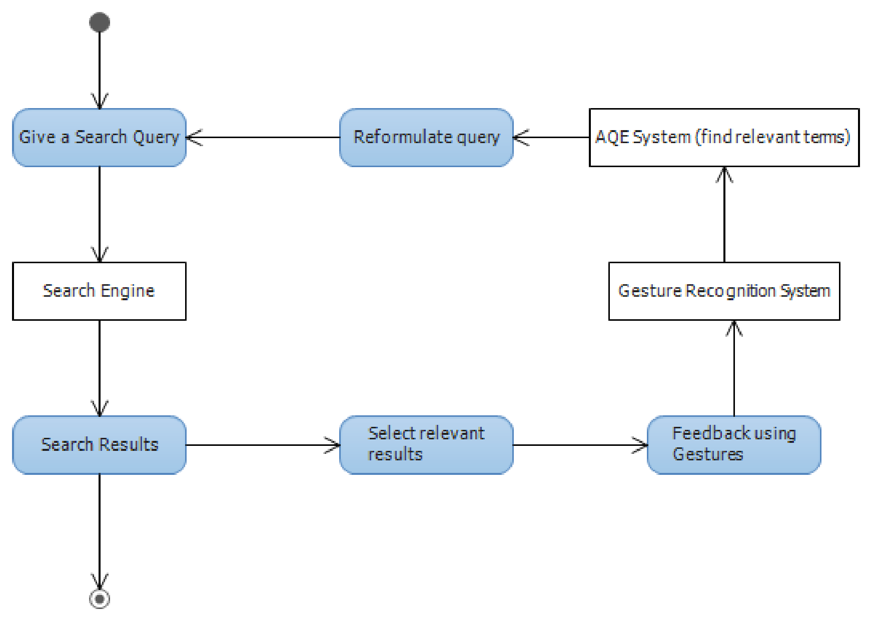


Figure 5.1: System Structure.

Majority of the project, including both the head movement recognition and the search engine development, was done in C#. The OpenCV port for C#, EmguCV [28], was used to handle images and to implement the gesture recognition system. Lucene .NET [29] which is the C# port of popular IR library Lucene, was used to code the search engine. The data-collection module and the UI were also formed using standard C# libraries.
The earlier version of the project which included the implementation of SVM, was developed in

MATLAB. It was not part of the final project.

5.2 Software Details

5.2.1 Relevance Feedback and Search Engine

Search engine:
Search engine was developed using Lucene .NET, which is a port of Lucene for .NET. 121,790 documents of the dataset were indexed.
The searching and indexing were part of separate classes called SimpleFileSearcher and SimpleFileIndexer. All the indexing and searching methods were part of these classes respectively. Indexing and searching were done using the StandardAnalyzer. It should be noted that the analyzer with the same configuration must be used to read the index. Otherwise it may return no results, and was a problem faced during the project. StandardAnalyzer does both lower-case and stop-word filtering, and in addition tries to do some basic clean-up of words, for example taking out apostrophes and removing periods from acronyms (for eg. "T.L.A." becomes "TLA"). A custom set of stop-words comprising of HTML tags was given to the analyzer as the corpus consisted of HTML pages and they were full of HTML tags, which interfered with the searching [27].
Indexing:

```
SimpleFileIndexer indexer = new SimpleFileIndexer();
int numIndex = indexer.index(indexDir, dataDir, suffix);
```

Figure 5.2: Code for indexing documents.

For indexing, the code takes the path of the dataDir (directory which contains the database), the indexDir (directory where index is stored) and the suffix of the files which we want to index. The program then searches recursively in the dataDir for files matching the given file prefix and creates the index in the indexDir. Indexing is needed one time before we start searching.

```
SimpleSearcher searcher = new SimpleSearcher();
searcher.searchIndex(indexDir, query, hits);
```

Figure 5.3: Code for searching documents.

Similarly, during searching the code takes the path of the indexDir. It also takes the search query and the number of hits to return. a value of 100 is used for hits.
In the implementation of searchIndex the query is parsed by QueryParser to correct the format of the query. This query is then given to an instance of the SearchIndex class of Lucene. SearchIndex returns a set of documents for the given query in TopDocs. This is how the search results are generated the first time.

These search results are displayed to the user using a DataGridView control of .NET 4.0. The multi-select option of the DataGridView was enabled to allow users to select multiple results at

once. A web browser was also embedded in the system to allow the user to check the web page in the system itself, and so that he can make better decision about the relevance of a document. Double-clinking on a row of the DataGridView would load that document in the browser. The user then clicks a button to start the gesture recognition. The implementation of the gesture recognition system is explained later in the section.
After the gesture is recognized, the system takes the set of relevant documents as marked by the user and then, uses it to find better and relevant results.

```
Query moreResultsQuery = mlt.Like(hits[num].doc);
topDocs = searcher.Search(moreResultsQuery, maxHits);
```

Figure 5.4: Code for using MoreLikeThis to get similar results in topDocs.

The first approach uses MoreLikeThis class of Lucene. It takes one document as input and generates a query by looking at the interesting terms of the document, which we use to search the index again and generate results. We repeat it for all the relevant documents and then take the intersection of the documents. This set contains the documents which were found to be similar to all the relevant documents, so they were expected to be relevant to the user as well. But, the fact that the original query was not used for the computation of the new query made it a little different from relevance feedback with the results being overly focussed on the relevant documents.

```
TermFreqVector tfq = reader.GetTermFreqVector(num[i], "contents");
String[] Terms = tfq.GetTerms();
int[] Freqs = tfq.GetTermFrequencies();
```

Figure 5.5: Code for getting the term frequencies.

```
int df = reader.DocFreq(new Term("contents", Terms[j]));
double idf = caclIDF(df, reader.NumDocs());
int tf = Freqs[j];
double score = tf * idf;
tfs.Add(new TermFreq(Terms[j], score));
```

Figure 5.6: Code for finding idf and using it to calculate the tf*idf score.

The second approach finds the relevant terms from the set of relevant documents. We used the score of tf * idf to look for interesting terms in a document. We find tf and idf using Lucene. Using the scoring, the top 25 terms were selected and merged to form a new query. The old query was also used to form the new query and was given a larger weight by giving it a greater boost factor. Greater boost factor ensures that search engine finds more similar documents to that query. This new query is used to search the index. The program returns the documents which are shown to the user on the DataGridView.

5.2.2 Gesture Recognition System

In this project, we used EmguCV and MATLAB to work with images. For data collection we created a program in C# which captured an image from the capture device, which was a webcam in our case.

```
capture = new Capture();
timer = new System.Timers.Timer();
timer.Interval = 0.1 * 1000;
timer.Enabled = true;
timer.Start();
timer.Elapsed += new ElapsedEventHandler(timer_Elapsed);
```

Figure 5.7: Code for setting up the webcam and fps.

We used a timer to raise a *timer_elapsed* event after 400 ms which allowed us to capture images at a frame rate of 25 fps. We captured the two movements of the user and stored the frames associated with them in separate folders. A caseId, associated with each participant, was stored in a separate file called cases.txt to automate the process.
In the first approach, we used SVM based classifier described earlier to recognize gestures. We used the cases.txt file to find faces from all the images of a particular person.

```
HaarCascade Face = new HaarCascade("haarcascade_frontalface_alt2.xml");
var faces = GrayImage.DetectHaarCascade(Face, 1.4, 4,
HAAR_DETECTION_TYPE.DO_CANNY_PRUNING, new Size(Image.Width / 8, Image.Height / 8))[0];
```

Figure 5.8: Code for finding faces from the frames.

The faces were identified using the HaarCascade method provided by EmguCV. It is based on the Viola Jones method of face recognition. It is already trained and the trained values are provided through an XML file.

```
img1 = imread(list1(j).name);
img2 = imread(list1(j+1).name);
lbp1 = lbp(img1,1,8,mapping,'h');
lbp2 = lbp(img2,1,8,mapping,'h');
feat1(i,k) = pdist2(lbp1, lbp2, 'chisq');
```

Figure 5.9: MATLAB Code for reading two images, finding their LBP and then finding the chi-squared distance.

Feature extraction was done on MATLAB, where the chi-squared distance between the LBPs of (i) consecutive frames, (ii) first and last frame, (iii) first and middle frame and (iv) middle and last frame were used to form a feature vector. The code above is for finding the features between consecutive frames. Code for other features is similar.

This feature vector is given with proper labels (horizontal or vertical) to train an SVM. SVM training and classification was also done in MATLAB. The part of code was exported to a .Net

```
SVM_train = svmtrain(trainingset,traininglabel, 'quadprog_opts', options, 'method',
'QP', 'Kernel_Function', 'polynomial', 'polyorder', 5);
svmpredict(test_label, test_data, SVM_train);
```

Figure 5.10: MATLAB Code for training the SVM with the desired options and then using the trained SVM to predict the class of the test data.

dll using the deploytool tool of MATLAB. Although there was a version mismatch between the dll and the present version of dll the system could call the SVM for classification results. The system worked reasonably well with a classification accuracy of 64% but calling separate libraries and extracting features before classifying made the system very slow. So, we moved on to the next approach.

In the next approach we implemented [12], using purely C# and EmguCV methods. The first step of this algorithm is same as the last algorithm's first step. We find faces from the frames using Viola-Jones detector.

```
PointF[][] ActualFeature = faceGrayImage.GoodFeaturesToTrack(400, 0.5d, 5d, 5);
```

Figure 5.11: Code for finding the interest points using the Shi-Tomasi corner detection algorithm.

After finding the faces we find the features which are tracked by optical flow algorithm. We use Shi-Tomasi corner detector algorithm to find these good features to track. We use the EmguCV method to get these features.

```
hull = PointCollection.ConvexHull(ActualFeature[0], storage,
Emgu.CV.CvEnum.ORIENTATION.CV_CLOCKWISE).ToArray();
```

Figure 5.12: Code for finding the convex hull.

After we get the features to track, we further reduce them by taking the outermost points on the face. This is known as finding the convex hull of the face. The convex hull method was used to find the reduced set of points.

```
referenceCentroid = FindCentroid(hull);
```

Figure 5.13: Code for finding the centroid of the interest points.

We find the centroid of the polygon formed by these points in a method FindCentroid where the algorithm for finding the centroid was coded.

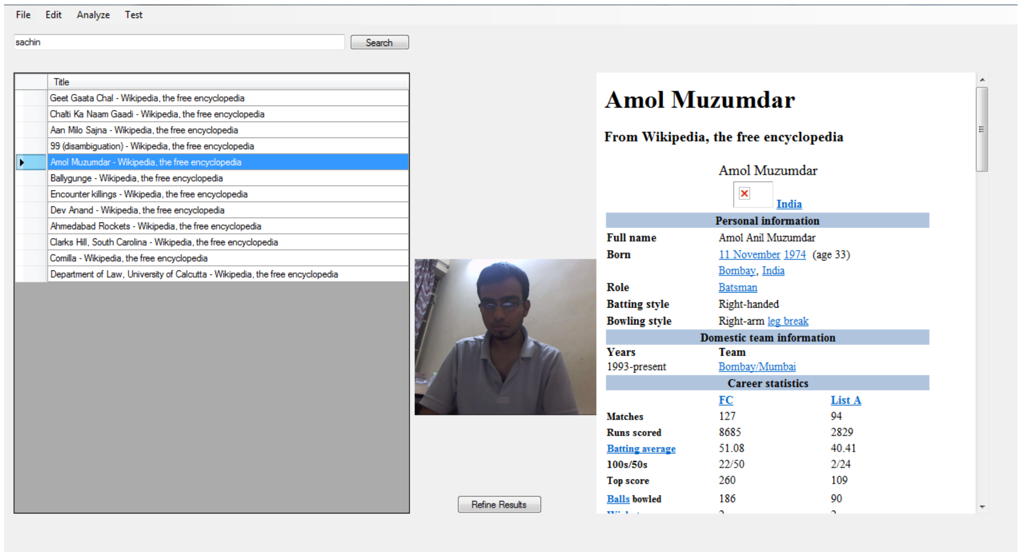
After the first frame, calculates optical flow for this sparse feature set using iterative Lucas-Kanade method in pyramids. The EmguCV implementation of iterative Lucas-Kanade method in pyramids needs the prevFrame, nextFrame, prevFeatures etc. and returns the position of the features in the new frame. We then compute the centroid of these features and compare it with the reference frame to find the direction of the movement.

```
OpticalFlow.PyrLK(grayFrame, nextGrayFrame, ActualFeature[0], new System.Drawing.Size(10, 10),
3, new MCvTermCriteria(20, 0.03d), out NextFeature, out Status, out TrackError);
nextHull = PointCollection.ConvexHull(ActualFeature[0], storage,
Emgu.CV.CvEnum.ORIENTATION.CV_CLOCKWISE).ToArray();
nextCentroid = FindCentroid(nextHull);
```

Figure 5.14: Code for using optical flow to track features.

5.3 User Interface and Screen-shots

The UI consists of a DataGridView - for showing the search results; A web browser - to facilitate users to look at the actual web-page and make better decisions about the relevant documents and an ImageBox to show the images from the webcam. The image in the ImageBox was changed so quickly that it looked like a video feed from the webcam. Following are a few screenshots to illustrate the working of the system.



sachin

	Title
	Geet Gaata Chal - Wikipedia, the free encyclopedia
	Chalti Ka Naam Gaadi - Wikipedia, the free encyclopedia
	Aan Milo Sajna - Wikipedia, the free encyclopedia
	99 (disambiguation) - Wikipedia, the free encyclopedia
▶	Amol Muzumdar - Wikipedia, the free encyclopedia
	Ballygunge - Wikipedia, the free encyclopedia
	Encounter killings - Wikipedia, the free encyclopedia
	Dev Anand - Wikipedia, the free encyclopedia
	Ahmedabad Rockets - Wikipedia, the free encyclopedia
	Clarks Hill, South Carolina - Wikipedia, the free encyclopedia
	Comilla - Wikipedia, the free encyclopedia
	Department of Law, University of Calcutta - Wikipedia, the free encyclopedia

Amol Muzumdar

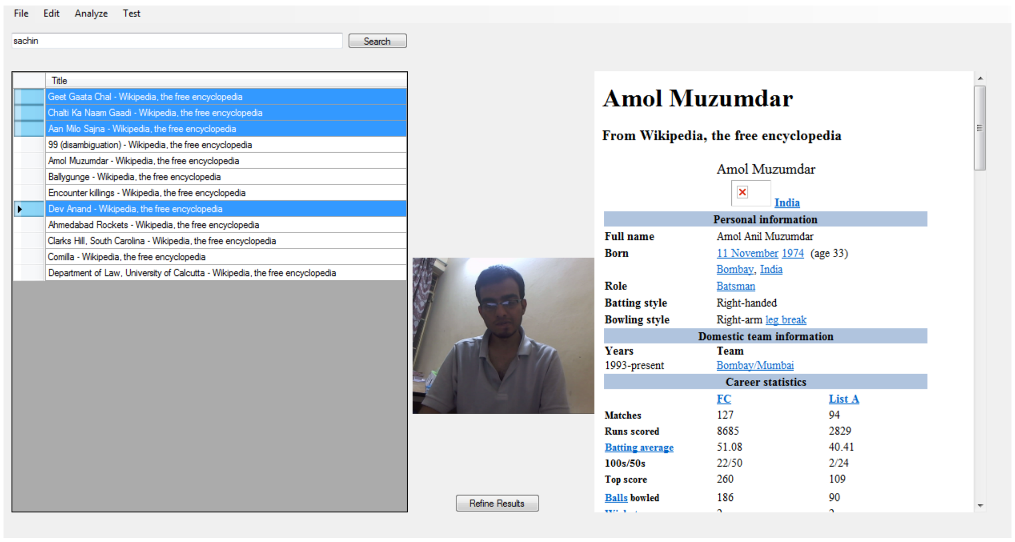
From Wikipedia, the free encyclopedia

Amol Muzumdar

India

Personal information		
Full name	Amol Anil Muzumdar	
Born	11 November 1974 (age 33) Bombay, India	
Role	Batsman	
Batting style	Right-handed	
Bowling style	Right-arm leg break	
Domestic team information		
Years	**Team**	
1993-present	Bombay/Mumbai	
Career statistics		
	FC	**List A**
Matches	127	94
Runs scored	8685	2829
Batting average	51.08	40.41
100s/50s	22/50	2/24
Top score	260	109
Balls bowled	186	90

Figure 5.15: The pane on the left is the DataGridView, web browser is on the right pane and the video from the webcam is between these two panes. In this figure, search results are shown in the DataGridView.



	Title
	Geet Gaata Chal - Wikipedia, the free encyclopedia
	Chalti Ka Naam Gaadi - Wikipedia, the free encyclopedia
	Aan Milo Sajna - Wikipedia, the free encyclopedia
	99 (disambiguation) - Wikipedia, the free encyclopedia
	Amol Muzumdar - Wikipedia, the free encyclopedia
	Ballygunge - Wikipedia, the free encyclopedia
	Encounter killings - Wikipedia, the free encyclopedia
▶	Dev Anand - Wikipedia, the free encyclopedia
	Ahmedabad Rockets - Wikipedia, the free encyclopedia
	Clarks Hill, South Carolina - Wikipedia, the free encyclopedia
	Comilla - Wikipedia, the free encyclopedia
	Department of Law, University of Calcutta - Wikipedia, the free encyclopedia

Amol Muzumdar

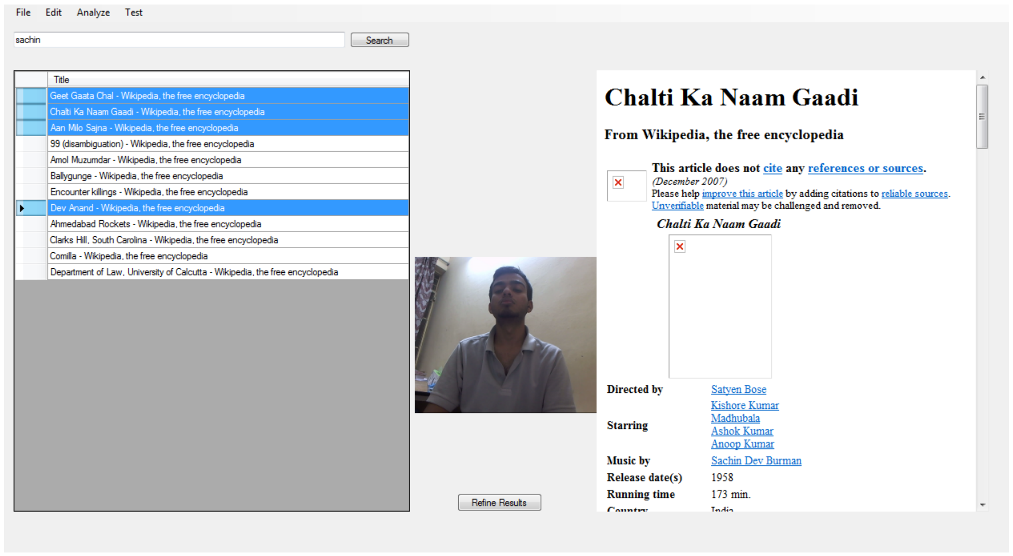
From Wikipedia, the free encyclopedia

Amol Muzumdar

India

Personal information		
Full name	Amol Anil Muzumdar	
Born	11 November 1974 (age 33) Bombay, India	
Role	Batsman	
Batting style	Right-handed	
Bowling style	Right-arm leg break	
Domestic team information		
Years	**Team**	
1993-present	Bombay/Mumbai	
Career statistics		
	FC	**List A**
Matches	127	94
Runs scored	8685	2829
Batting average	51.08	40.41
100s/50s	22/50	2/24
Top score	260	109
Balls bowled	186	90

Figure 5.16: The user selects a set of relevant documents.



	Title
	Geet Gaata Chal - Wikipedia, the free encyclopedia
	Chalti Ka Naam Gaadi - Wikipedia, the free encyclopedia
	Aan Milo Sajna - Wikipedia, the free encyclopedia
	99 (disambiguation) - Wikipedia, the free encyclopedia
	Amol Muzumdar - Wikipedia, the free encyclopedia
	Ballygunge - Wikipedia, the free encyclopedia
	Encounter killings - Wikipedia, the free encyclopedia
▶	Dev Anand - Wikipedia, the free encyclopedia
	Ahmedabad Rockets - Wikipedia, the free encyclopedia
	Clarks Hill, South Carolina - Wikipedia, the free encyclopedia
	Comilla - Wikipedia, the free encyclopedia
	Department of Law, University of Calcutta - Wikipedia, the free encyclopedia

Chalti Ka Naam Gaadi

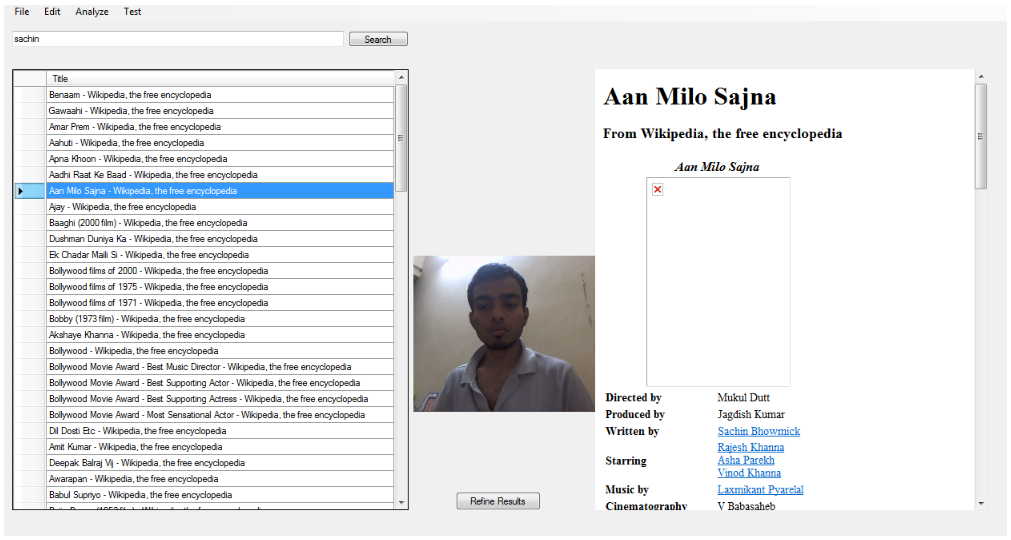
From Wikipedia, the free encyclopedia

This article does not cite any references or sources.
(December 2007)
Please help improve this article by adding citations to reliable sources. Unverifiable material may be challenged and removed.

Chalti Ka Naam Gaadi

Directed by	Satyen Bose
Starring	Kishore Kumar, Madhubala, Ashok Kumar, Anoop Kumar
Music by	Sachin Dev Burman
Release date(s)	1958
Running time	173 min.

Figure 5.17: The user providing feedback by moving head. In this screenshot, the user is giving positive feedback.



Title
Benaam - Wikipedia, the free encyclopedia
Gawaahi - Wikipedia, the free encyclopedia
Amar Prem - Wikipedia, the free encyclopedia
Aahuti - Wikipedia, the free encyclopedia
Apna Khoon - Wikipedia, the free encyclopedia
Aadhi Raat Ke Baad - Wikipedia, the free encyclopedia
Aan Milo Sajna - Wikipedia, the free encyclopedia
Ajay - Wikipedia, the free encyclopedia
Baaghi (2000 film) - Wikipedia, the free encyclopedia
Dushman Duniya Ka - Wikipedia, the free encyclopedia
Ek Chadar Maili Si - Wikipedia, the free encyclopedia
Bollywood films of 2000 - Wikipedia, the free encyclopedia
Bollywood films of 1975 - Wikipedia, the free encyclopedia
Bollywood films of 1971 - Wikipedia, the free encyclopedia
Bobby (1973 film) - Wikipedia, the free encyclopedia
Akshaye Khanna - Wikipedia, the free encyclopedia
Bollywood - Wikipedia, the free encyclopedia
Bollywood Movie Award - Best Music Director - Wikipedia, the free encyclopedia
Bollywood Movie Award - Best Supporting Actor - Wikipedia, the free encyclopedia
Bollywood Movie Award - Best Supporting Actress - Wikipedia, the free encyclopedia
Bollywood Movie Award - Most Sensational Actor - Wikipedia, the free encyclopedia
Dil Dosti Etc - Wikipedia, the free encyclopedia
Amit Kumar - Wikipedia, the free encyclopedia
Deepak Balraj Vij - Wikipedia, the free encyclopedia
Awarapan - Wikipedia, the free encyclopedia
Babul Supriyo - Wikipedia, the free encyclopedia

Aan Milo Sajna

From Wikipedia, the free encyclopedia

Aan Milo Sajna

Directed by	Mukul Dutt
Produced by	Jagdish Kumar
Written by	Sachin Bhowmick
Starring	Rajesh Khanna Asha Parekh Vinod Khanna
Music by	Laxmikant Pyarelal
Cinematography	V Babasaheb

Figure 5.18: Set of relevant results returned by the system.

Chapter 6

Results

As is with most of the project, this section is divided into two parts. (i) Accuracy of Gesture recognition, (ii) Precision of the retrieved documents.

6.1 Accuracy of Gesture Recognition

Accuracy of the Machine Learning based algorithm The algorithm used for detection of facial movements gave an average accuracy of about 64% on 5-fold cross-validation. A k-fold cross-validation is done by dividing the dataset into k-sets of equal size. Out of k-sets, (k-1) sets are used for training the classifier and the remaining set is used for testing the accuracy of the classifier trained by the training samples. A SVM classifier has some parameters which can be used to customize the classifier to get the best results. The parameters of the SVM were also varied to extract the best out of the classifiers. The number of iterations to solve the optimization problem of SVM were varied from 100 to 1500, with an increment of 100 iterations each time. 1000 iterations were chosen because further increasing the iterations was not improving the results. The kernel functions of the SM were also changed. A polynomial kernel function of order 3 gave the best accuracy on repeated testing. A detailed account of the different kernel functions used and the accuracies is given in the table 6.1.

No.	Kernel function	Max. Accuracy (with same testing set)
1.	linear	64.2857
2.	quadratic	42.8571
3.	(polynomial , Order - 3)	71.4286
4.	(polynomial , Order - 4)	64.2857
5.	rbf, Sigma - 0.1)	42.8571
6.	(rbf, Sigma - 0.2)	57.1429
7.	(rbf, Sigma - 0.5)	57.1429
8.	(rbf, Sigma - 1)	50

Table 6.1: Accuracies with change in kernel functions.

SVM uses quadratic optimization for finding the separating hyper planes (decision boundary) irrespective of the kernel functions. Also note that the radial basis function used in the kernel is the Gaussian Radial Basis Function kernel with adjustable sigma values. Only one of the kernel function can be used at a time and with a polynomial kernel of order 3 it gives decent accuracy of 71% and 64% on cross-validating, so we use this kernel.

Accuracy of the Optical Flow based algorithm. The optical flow algorithm was tested on the database collected. The classification accuracy on both the horizontal and vertical movements for 37 subjects came out to be 83%, which is much better than the machine learning based algorithm, which managed to give average of 64% of accuracy. Since it is a deterministic algorithm, there was no need to do a k-fold cross-validation. The above results were obtained on this same dataset.
Apart from the accuracy, this method was also very fast, and gave quick results, which was a desirable trait for our information retrieval system.

6.2 Precision of the retrieved documents

Both the approaches i.e. MoreLikeThis based approach and the IDF based approach work well given the set of test search queries used for performance review. The performance of a search engine is judged by calculating the recall and precision. Precision is the fraction of retrieved instances that are relevant, while recall is the fraction of relevant instances that are retrieved. We will use only precision as the evaluation criteria in our project, because relevant documents would vary from person to person and it would need manual annotation over whole of the dataset by each person. Also, we can improve recall by retrieving all the documents, so we use precision to measure the performance. Following is a general formula for precision and recall.

$$precision = \frac{|relevant\ documents \cap retrieved\ documents|}{|retrieved\ documents|}$$

and

$$recall = \frac{|relevant\ documents \cap retrieved\ documents|}{|relevant\ documents|}$$

The set of test queries which were given to the users were selected in a way that they have a little ambiguity in their usage or meaning. The user was free to choose any of the meaning which he found suitable. Care was taken that the search queries are not from a specific field so as to remove any bias of user's background. The set of search queries given to the user were as follows:

i) Apple - used for a fruit and a company

ii) Sachin - used for a famous cricketer and a general name

iii) Bug - used for a insect and a software "bug"

iv) Virus - used for a micro-organism and a computer malware

v) World cup - used for any sports world cup

vi) Party - used for a political party or a fun party

vii) Rock - used for a music genre as well as for big stones

viii) Polish - used for shoe shining cream and for relating to the country Poland

ix) Lead - used for a metal and as a verb

x) Gandhi - used for Mahatma Gandhi and members of the Nehru family

The user marked the relevant results based on his perception of the search query. The user was then asked to provide feedback to the system using the gesture. The system then expands the query and returns the new search results. The user then marked the documents that he found relevant to his search. After this exercise the user was asked to fill a short survey asking him about his experience and the performance of the system. 10 users with computer science background from an age-group of 18-22 years agreed to evaluate the system. They evaluated the system in a well-lit room with no specific instructions about the posture given.
The list of questions asked after the evaluation were as follows:

i) Were you satisfied with the system?

ii) Did the search results improve after the feedback?

iii) Do you think that the system is obtrusive?

iv) Would you like to use such a system on a daily basis?

Both the approaches showed considerable increase in precision. 7 users were satisfied with the working of the system. 8 out of the 10 users were satisfied with the improvement of the search results. 5 of the 10 users found the system useful. But 6 out of the 10 people found the system obtrusive and didn't want to use the system on a regular basis.

For each of the test term, there was a variety of precision values which came up. In some of the cases the first approach performed better than the IDF based approach and vice-versa. The average improvement in precision by the MoreLikeThis approach was 42%, and the IDF based approach brought up an improvement of 39% in the precision. This shows that both the approaches give good results.

Chapter 7

Conclusions, Future Work and Limitations

7.1 Conclusions

The project developed a gesture based relevance feedback system. The Haar-like features gave good results for detecting faces from raw video frames. In the first approach, SVMs were found to be easy to train and the classification was not time consuming. It worked well for our approach and gave an accuracy of 64%. Optical Flow algorithm was found to be extremely quick to determine the head movement which is helpful in making the system real-time.
By this part of the project we have been able to detect head movement, and use it to improve search results depending on the prior information about the relevant documents. This serves as a proof of concept and confirmation of our belief that such technology could exist and it would be successful. We can improve the accuracies that we are getting and move on to find better ways of recognizing the movements and use it to build an information retrieval system which would give better and more relevant results to the user. This system has lot of possible extensions and applications.

7.2 Future Work

Our current approach works on the movements of the head. We plan to improve our system to get better accuracies for the current system by using other approaches then our current approach. The project can be extended to include facial emotions to the system. Using a 3-dimensional camera to form a 3-D model of the person, the emotions of the person can be captured more effectively. There are many aspects of the human emotions that are not studied because of the limitations of traditional computers but with the help motion capture devices like Microsoft Kinect we can use the depth information along with the 2-D information we have to study the emotions in more detail. A number of researches have been done using Microsoft Kinect for gesture based studies. It would help us also in forming a better feature vector and hence help in the classification and recognition. Improving our algorithm iteratively gives more robustness and speed to the system.

7.3 Limitations

Our system is hindered by some limitations. The system works for a two-class problem while the human emotion can be categorized in 6 different emotions. We planned to include multiple class recognition to include emotion detection in the system. In the first approach, the features used for the classification depends on the chi-squared distance between frames. The number of frames is not constant and varies for each participant since the speed of movement also varies from participant to participant. As the number of features should be same for all the data points, the number of features has to be limited to the smallest feature vector from the dataset. We need to make the feature vector consistent and more representative of the data that we have, so that we make better classifications.
Good lighting conditions are required. Also the system tend to give wrong result if the user is sitting far from the camera, as the difference from he reference point is not much to make a difference or to make wrong decisions.

Bibliography

[1] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR ’01), 2001, vol. 1, p. 511-518.

[2] I. Essa and A. Pentland, “Coding, analysis, interpretation and recognition of facial expressions”, IEEE Transactions on Pattern Analysis and Machine Intelligence, July 1997, vol. 19, no. 7, p. 757-763.

[3] M. Turk and A. Pentland, “Eigenfaces for recognition”, Journal of Cognitive Neuroscience, 1991, vol. 3, no. 1, p. 71-86.

[4] A. Kapoor and R.W. Picard, “Real-Time, Fully Automatic Upper Facial Feature Tracking”, In Proceedings of 5th International Conference on Automatic Face and Gesture Recognition, May 2002, p. 10-15.

[5] A. Kapoor and R.W. Picard, “A real-time head nod and shake detector”, In Proceedings of the 2001 workshop on Perceptive user interfaces (PUI ’01), 2001, p. 1-5.

[6] J.F. Cohn, A.J. Zlochower, J.J. Lien, and T. Kanade, “Feature-Point Tracking by Optical Flow Discriminates Subtle Differences in Facial Expression”, In Proceedings of the 3rd. International Conference on Face & Gesture Recognition (FG ’98), IEEE Computer Society, 1998, p. 396-401.

[7] S. Mitra and T. Acharya. “Gesture recognition: A survey.” IEEE Transactions on Systems, Man, and Cybernetics; Part C: Applications and Reviews, vol. 37, no. 3, May 2007, p. 311-324.

[8] P. Michel and R. El Kaliouby, “Real time facial expression recognition in video using support vector machines”, In Proceedings of the 5th international conference on Multimodal interfaces (ICMI ’03), 2003, p. 258-264.

[9] Shuai Jing, Yi Li, Guang-ming Lu, Jian-xun Luo, Wei-dong Chen and Xiao-xiang Zheng, “SOM-based Hand Gesture Recognition for Virtual Interactions”, International Symposium on VR Innovation, March 2011, p. 317-322.

[10] K. Mase, "An Application of Optical Flow - Extraction of Facial Expression", MVA'SO IAPR Workshop on Machine Vision Applications, November 1990.

[11] J.F. Cohn, A.J. Zlochower, J.J. Lien, and T. Kanade, "Feature-Point Tracking by Optical Flow Discriminates Subtle Differences in Facial Expression", In Proceedings of the 3rd. International Conference on Face & Gesture Recognition (FG '98), IEEE Computer Society, 1998, p. 396-401.

[12] U. Saeed and J. Dugelay, "Facial Video based Response Registration System", 16th European Signal Processing Conference (EUSIPCO 2008), Lausanne, Switzerland, August 2008.

[13] N.J. Belkin, "Some(what) Grand Challenges for Information Retrieval", SIGIR Forum, vol. 42, no. 1, June 2008, p. 47-54.

[14] I. Lopatovska and I. Arapakis, "Theories, methods and current research on emotions in library and information science, information retrieval and human-computer interaction", Information Processing and Management, vol. 47, no. 4, July 2011, p. 575-592.

[15] Kevin Hsin-Yih Lin, Changhua Yang and Hsin-Hsi Chen, "What Emotions do News Articles Trigger in Their Readers?", In Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '07), p. 733-734.

[16] I. Lopatovska, "Emotional correlates of information retrieval behaviors", In Proceedings of the SSCI 2011 WACI - 2011 Workshop on Affective Computational Intelligence, April 2011

[17] Y. Moshfeghi, G. Zuccon and J.M. Jose, "Using Emotion to Diversify Document Rankings", In Proceedings of the Third international conference on Advances in information retrieval theory (ICTIR'11), 2011, p. 337-341

[18] I. Arapakis, J. M. Jose and P. D. Gray, "Affective feedback: an investigation into the role of emotions in the information seeking process", In Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '08), 2008, p. 395-402

[19] C. Carpineto and G. Romano, F.U. Bordoni, "A Survey of Automatic Query Expansion in Information Retrieval", ACM Comput. Surv., vol. 44, no. 1, January 2012, p. 1-50.

[20] D. Kelly and N.J. Belkin, "Reading time, scrolling and interaction: exploring implicit sources of user preferences for relevance feedback", In Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '01), 2001, p. 408-409.

[21] Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze, "Introduction to Information Retrieval", Cambridge University Press, 2008.

[22] P. N. Belhumeur, Lecture Slides, Retrieved: 10th May, 2012: "http://www1.cs.columbia.edu/ belhumeur/courses/biometrics/2010/svm.ppt".

[23] B.K.P. Horn and B.G. Schunck, "Determining Optical Flow", Technical Report, Massachusetts Institute of Technology, Cambridge, MA, USA, 1980.

[24] B.D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision", In Proceedings of the 7th international joint conference on Artificial intelligence - Volume 2 (IJCAI'81), vol. 2, 1981, p. 674-679,

[25] T. Ojala, M. Pietikäinen and D. Harwood, "A comparative study of texture measures with classification based on featured distributions", Pattern Recognition, vol. 29, no. 1, January 1996, p. 51-59.

[26] Database for IR system, download site, Retrieved: 15th April, 2012: "http://www.search-engines-book.com/collections/".

[27] Lucene Documentation, Retrieved: 15th April, 2012 : "http://lucene.apache.org/core/old_versioned_docs/versions/3_5_0/".

[28] EmguCV; "www.emgu.com".

[29] Lucene .NET; "http://incubator.apache.org/lucene.net/".