Proposal for

Real Time Face Tracking and Recognition (RTFTR)

`http://rtftr.sourceforge.net`

Submitted to
Open Software Challenge – 2009

Feb. 19, 2009

Submitted by

Abhishek Dutta Anjan Nepal Bibek Shrestha Lakesh Kansakar

{ adutta.np, anjannepal, bibekshrestha, lakesh.kansakar } @ gmail.com

Table of Contents

Introduction

1.1 Purpose

RTFTR will assist security personnels in keeping track of all the people entering and leaving a secured zone. The core functionality of RTFTR will allow real time tracking and recognition of human faces in a video stream. It will have additional features like motion detection, appearance history recording, tracking of specified person. It will also provide a centralized solution for security cameras present at different geographical location by streaming video to central server through internet.

1.2 Scope

RTFTR will aim to assist security personnels and not replace them. The scope of RTFTR, with respect to final submission in OSC-2009, will not be to present a complete software based solution for the task of securing an area.

1.3 Assumptions and Dependencies

Assumptions:

1. The target area, where RTFTR will track people, has sufficient illumination.
2. People in the scene are moving at normal pace and not running. At least one frame of video stream must capture the complete face in frontal profile, or 15 degree left/right profile.
3. For the purpose of recognition, the target person facial data must exist in training set of RTFTR.

Dependencies:

1. RTFTR will achieve real time functionality using High Performance Distributed computing. OpenMPI and OpenMP (libraries that support HPC) will be used to distribute the task of face extraction/recognition among a set of networked computers (nodes).
2. Each node will be a part of LAN and will contain required libraries and rtftr executables.
3. RTFTR will depend on the following libraries: OpenCV, Vigra, Octave C++ library

Business Requirements

1.4 Business/Market Needs

Securing an area by tracking the human entry/exit in that area is the most generic need of most business enterprise thinking about security. Placing security personnels in each such area adds up to the total cost of security. Business enterprises are in great need of a centralized, cost effective and robust mechanism of securing its premises. RTFTR aims to provide such software based solutions to business and government agencies.

1.5 Benefits to Community

RTFTR cannot be used as a complete software based solution for security in this proposed form. However, it will mark a beginning of an open source tool that will help business enterprise (or anyone who requires to secure an area) to implement a low cost solution.

Anyone who wants to track the entry/exit of humans in a secured area (a generic problem in security field) can use RTFTR. RTFTR is an open source project and hence it will encourage community participation for research and development in creating a complete software based solution for securing an area.

1.6 Collaborative Gains

It will provide us an opportunity to study and implement the present state of art in face extraction/recognition technology. There are two novel techniques that are being tested in RTFTR

1. Study the effect of using two or more face extraction/recognition algorithms, in parallel, on overall accuracy of tracking/recognition task
2. High Performance Distributed Computing (using OpenMPI and OpenMP) has promising capabilities that can be used to achieve real time tracking/recognition task.

Performance results obtained after implementing these two techniques will provide new insight into the future prospects of creating a complete software based security solution. This project will also improve our skills of carrying out a research based project. RTFTR will also provide a good foundation for further research into the emerging field of "face analysis".

Detailed Requirements

1.7 Functional Requirements

Name - FR-1 : Grab frames from the source
Rationale - There are various sources of input. So, for selecting a proper image out of the several sources of input, capturing of the frame from input sources (camera, video and image) has to be done.
Requirement - When this function is invoked, the frame from the selected input has to be captured according to the criteria for capturing, eg. frame capture rate.

Name - FR-2 : Extract face from the frame image
Rationale - Before the recognition process, the face has to be extracted before it can be sent for recognition.
Requirement - When this function is invoked, one of the two face extraction (detection) algorithms must be executed and it should return the extracted faces in the given image.

Name - FR-3 : Recognize given face image
Rationale - The extracted face images of the frame are individually recognized from known set of faces.
Requirement - When this function is invoked, one of the two face recognition algorithms must be executed and should return the match of the given faces with the known face database.

Name - FR-4 : Display result
Rationale - The display of the output on the current frame is done.
Requirement - When this function is invoked, individually detected faces are enclosed on a rectangular box and recognized faces are labeled with the individual's name from the known face database.

Name - FR-5 : Select face extraction module
Rationale - This helps the user to select between the two of the face extraction algorithms or both for detection.
Requirement - The user can use the extraction module of his choice. The choice can be presented such that one or all of the face extraction modules can be selected at once. The software should work according to the selected modules.

Name - FR-6 : Select face recognition module
Rationale - This helps the user to select between the two of the face recognition algorithms or both for recognition.
Requirement - The user can use the recognition module of his choice. The choice can be presented such that one or all of the face extraction modules can be selected at once. The software should work according to the selected modules.

Name - FR-7 : Select specific individual tracking option
Rationale - This should provide the option to track one of the specified persons who is known in the face database.
Requirement - The specified user can be tracked in all the sequences of the video. The person's id should be provided for this function.

1.8 Nonfunctional Requirements

1.8.1 Usability

The UI should be intuitive and should require minimum effort and training to perform the task of tracking entry/exit of humans in a secured area.

1.8.2 Reliability

The face extraction/recognition task should have acceptable level (~80% success rate) of reliability.

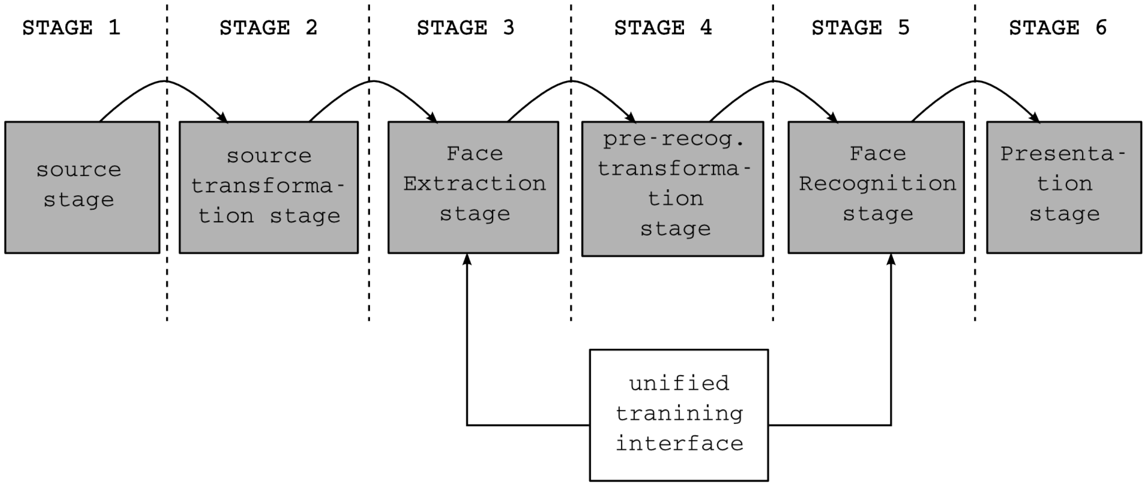
1.8.3 Performance

Must be able to perform face extraction/recognition tasks in real time (30 fps)

Project Plan

1.9 Project Execution Plan/Approach

The design of “Real Time Face Tracking and Recognition[RTFTR]” system is based on modular architecture. This allows flexibility to design the processing pathway for visual data according to our requirements. This architecture is suitable for testing the performance of different algorithms when they work in unison.



Now we give a brief description of each module in the block diagram:

Source Stage : This stage refers to the source of visual data for the system. The possible components of this stage are CCD camera, MPG video file, sequence of images, etc. Most face recognition systems are tightly coupled with the source,like a camera, which limits the capability of the software. This module will allow the rest of the system to fetch visual data from a variety of sources.

Source Transformation Stage : Face extraction algorithms usually process raw visual data. However, most of them have good performance when supplied with visual data that has been preprocessed or modified in some desirable way. This stage handles the task of transforming the visual data as required by the next stage algorithm. Identity Source transformation, which makes this stage transparent between STAGE 1 and STAGE 3, will be applied to those algorithms that do not require any kind of source transformation.

To illustrate the functioning of this stage, let us consider a hypothetical scenario in which the visual source, a CCD camera, is operating at 30 fps. The algorithm in STAGE 3 is not capable of processing this huge amount of data and hence only desires to receive the frame numbers 1,5,10,15,20,25,30. So a module in Source Transformation Stage will filter out the frames that are not required and only pass the desired frames to the next stage.

Face Extraction Stage : This stage implements the face extraction algorithms chosen for this project. It will extract human face area from a cluttered background and pass the details of this extracted region to next stage. Two algorithms will be implemented for this stage

- Neural Network Based face Detection[1]
- AdaBoost Algorithm and Cascaded Detector[2]

PRE-RECOG Transformation Stage : This stage is similar to Source Transformation Stage except that it performs transformation as required by the face recognition algorithms in STAGE 5.

To illustrate the functioning of this stage, let us consider a hypothetical scenario in which Eigenfaces[6] based recognition scheme is implemented in STAGE 5. Eigenfaces scheme cannot recognize faces having size not same as the training sample image size ie: Eigenfaces algorithm is not scale invariant. Hence, the face extracted by STAGE 3 algorithm must be scaled to match the size of training samples used to train the Eigenfaces algorithm. Hence a PRE-RECOG Transformation module, that performs scaling, can be used in this stage to increase the robustness of algorithm implemented in STAGE 5.

Face Recognition Stage : This stage implements the face recognition algorithms chosen for this project. It will identify the faces extracted by STAGE 3 by performing a match against know face database. Two algorithms will be implemented for this stage

- Subspace Linear Discriminant Analysis (LDA)[3]
- Face Recognition using Gabor Wavelet Transform[4]

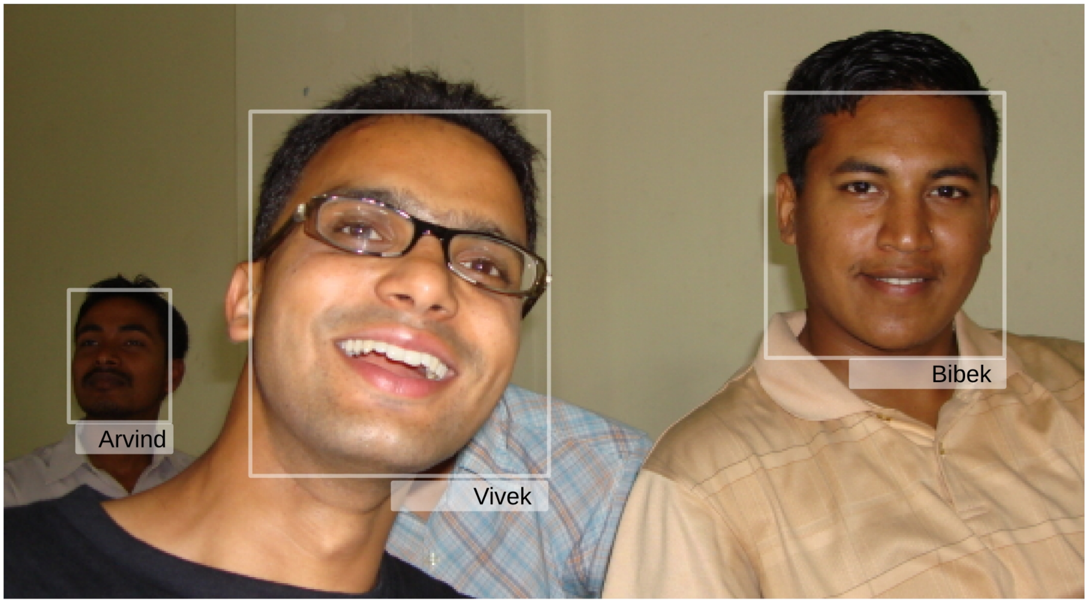
Presentation Stage : The result of face tracking/recognition will be made available by this stage. The output format can vary according to the requirement of user. The most general form of STAGE 6 output is depicted in

[1] “Neural Network-Based Face Detection”,Henry A. Rowley, Shumeet Baluja, and Takeo Kanade, IEEE Transactions on Pattern Analysis and Machine Intelligence, volume 20, number 1, pages 23-38, January 1998. http://vasc.ri.cmu.edu/NNFaceDetector/pami98.ps.gz

[2] “Rapid Object Detection using a Boosted Cascade of Simple Features”,P. Viola and M. Jones, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, December 9-14, 2001.

[3] “Subspace linear discriminant analysis for face recognition”, Zhao, W., Chellappa, R., and Phillips, P. J. 1999 Tech. rep. CAR-TR-914, Center for Automation Research, University of Maryland, College Park, MD.

[4] “Face Recognition using Gabor Wavelet Transform”, BURCU KEPENEKCI, SEPTEMBER 2001,THE MIDDLE EAST TECHNICAL UNIVERSITY



Unified Training Interface : The algorithms implemented in STAGE 3 and STAGE 5 require some form of training before they can become capable of performing the face tracking/recognition task. The training method of these algorithms vary considerable. But most of these algorithms take images containing faces as training data. Hence the Unified Training Interface, not the part of any stage, handles the task of training the algorithms with appropriate data.

1.10 Software Development Methodology

RTFTR is a research oriented project and hence standard software development methodologies cannot be directly applied for its design and development. However, we will be using "Agile development methodology" for those portions whose design and development are quite straight forward and does not involve uncertainties usually associated with research.
Group ware tools will be extensively used for collaboration of the project design and development. Subversion(svn) will be used for collaboration on code development, trac will be used for for project management(we will employ milestone based development strategy) and WIKI will be used for collaboration on project documents.

C++ (gcc compiler) will be the main programming language for development of RTFTR. Several libraries OpenCV, Vigra, Octave C++ libraries, etc

dOxygen will be used for comprehensive code documentation. Final project report will be the project document and explain other aspects of RTFTR.

GNU C++ coding standards will be followed by all the project members. Kdevelop (IDE) will be used throught the code development phase.

1.11 Project Scheduling

March 20, 2009 - Completion of Open MPI based integration of two Face Extraction modules (LDA, Kepenekci) and two Face Recognition modules (RowleyNN, Adaboost)

April 20, 2009 - Complete creation of intuitive GUI for visualization of Face tracking/recognition processes

May 01, 2009 - Complete addition of features for appearance history, motion tracking, specified face tracking

May 02 - 15, 2009 - Perform Field testing and obtain statistical data related to performance of RTFTR
May 15 - 22, 2009 - Completion Optimization, Code cleanup
May 23 - 30, 2009 - Project documentation, report preparation

1.12 Milestones

Milestone M1: OpenMPI based integration of RTFTR modules
Milestone M2: GUI for RTFTR
Milestone M3: Addition of features like (motion tracking, appearance history, specified face tracking)
Milestone M4: Optimization to achieve mealtime operation (30fps processing) of RTFTR

1.13 Team Structure

Team Member	*Responsibilities*
Abhishek Dutta	Team Coordinator, Kepenekci(FE) module developer
Anjan Nepal	Adaboost(FR) module developer, Final Field Testing in charge
Bibek Shrestha	RowleyNN(FR) module developer, Project Maintainer & Release Manager
Lakesh Kansakar	LDA(FE) module developer, Documentation Manager

1.14 Unit Testing Plan

CPP Unit tests will be developed to test the proper functioning of each Face extraction/recognition modules. CPP Unit tests cannot fulfill the needs of testing in parallel computing environment. Hence specialized testing procedure for OpenMPI based integrated code "rtftr" will be developed.

1.15 Build Environment

Automake based build system using GNU gcc compiler in Unix platform will be the build environment for rtftr.

1.16 Quality Plan

Software quality will be assessed during field testing session of rtftr.

1.17 Software Release/Packaging Plan

Software will be packaged as standard GNU style project with *"./configure && make && make install"* style build and install facility. Packages for other standard platforms will also be created.