Parallel Architecture & Programming Models for Face Recognition

Submitted by –
Sagar Kukreja
Computer Engineering Department
Rochester Institute of Technology
Agenda

• Introduction to face recognition
• Feature Extraction
• Principle Component Analysis
• Distributed Parallel Programming Model for face recognition using MPI
• Improving recognition on huge databases
• MMX based Recognition Algorithm
• Distributed Parallel System Architecture for face recognition
• Performance
• Results
Introduction

• **A face recognition system** is a computer application capable of identifying or verifying a person from a digital image or a video frame from a video source. One of the ways to do this is by comparing selected facial features from the image and a face database.
Face Detection

• The face recognition applications are widely used in different fields like security and computer vision. The recognition process should be done in real time to take fast decisions.

• Basic idea: slide a window across image and evaluate a face model at every location
Related Work in Feature Extraction

• Appearance based methods in which features are mainly from pixel intensity values
  • Linear Analysis Methods like Principle Component Analysis, Independent Component Analysis.
  • Non-linear Analysis Methods

• Geometrical Feature Matching - The geometrical feature approach construct feature vectors based on the geometrical representation of the face. The feature vector contains the face outline, eyes, mouth and nose positions.
Principle Component Analysis

• PCA is considered as a feature extraction technique and is widely used in facial recognition applications by projecting images in new face space.

• PCA can reduce the dimensionality of the image.

• However, PCA consumes a lot of processing time due to its high intensive computation nature.
PCA Recognition System

Acquisition → Preprocessing → Normalized face image → Feature extraction → Create Eigen Face Space

Training Step

Project into Eigen Face Space → Matching process

Recognition Step
Problem With PCA

• The main problem of PCA is consuming huge amount of time because of its computationally intensive nature.
• A major step of the algorithm is calculating the covariance matrix of the dataset to be able to get the eigenvectors and eigenvalues.
• This step intensively increases the execution time of the algorithm when the training data-base contains a lot of pictures.
Various approaches to solve the problem

• One approach is calculating the eigenvectors and eigenvalues without calculating the covariance matrix. Methods like the expectation maximization algorithm (EM) to reduce the determinant matrix manipulation.

• Another approach is use of distributed computing to improve recognition on huge databases, specifically, TH-FACE. The proposed algorithm uses special distributed parallel architecture with mmx technology to speed-up the matching process in PCA algorithm.
Various approaches to solve the problem

- Compute PCA in one pass on a large data set based on summarization matrices. Furthermore, that algorithm is applied on database management systems (DBMS). They use parallel data set summarization via user-defined aggregations and solve Singular Value Decomposition (SVD) by using Math Kernel Library (MKL) parallel variant of the Linear Algebra PACKage (LAPACK) library.

- In another approach, a distributed parallel system consisting of one host, four slaves and some clients is used. The Parallel Virtual Machine (PVM) is established by the host. In addition, the communication is done over TCP socket and the whole system is communicated and linked over 100M network switch, and achieves an acceleration ratio of 4.133 if the whole system works together.
Steps to compute PCA

• For the purpose of computing the mean, convert the image to be a column in a 2D matrix called A. This matrix represents all images.

$$X = \begin{bmatrix} \text{img}_1 & \text{img}_2 & \ldots & \text{img}_n \end{bmatrix}_{(M \times N \times P)}.$$

• Calculating the average of all pixels in all images (each pixel with its corresponding pixels) and subtracting the average from X to remove any lighting. consequently, all pixels are returned back to the origin of the system.

$$\text{avg} = \frac{1}{P} \sum_{i=1}^{P} \text{img}_i$$

$$A = X - \text{avg}$$
Steps to compute PCA

• Get the eigenvalues and eigenvectors by calculating the covariance matrix. The number of eigenvalues and eigenvector is equal to P-1.

\[
C = AA^T \\
L = A^TA \\
[V,D] = \text{eig}(L)
\]

Where:
- \(V\) is the matrix of eigenvectors.
- \(D\) is the diagonal matrix for eigenvalues.

• Sort the eigenvectors based on their eigenvalues. The longest vectors which have the largest eigenvalues can split the points in the proper way.

• Project the training images to this new space by applying dot product between \(A\) and \(V\) matrices,

• Store the resulting face space.
Steps to Compute PCA (In recognition Phase)

• Convert image into a \((MN)1\) vector.
• Subtract the mean calculated in the training phase from this image vector.
• Project the testing image in the face space.
• Calculate the Euclidean distance between the projected test image and all training images. Images having the distance less than or equal to a predetermined threshold is matched to, the testing image belong to that image.
Proposed Approach

• We exploit distributed parallel programming models to improve the execution of PCA for face recognition. This enables the distribution of either data or tasks over a network of connected computers.

• The Message Passing Interface (MPI) enables us to run different MPI tasks concurrently on different machines.

• In addition, MPI handles the communication by sending message between nodes. MPICH2 implementation is a high-performance and portable implementation of the MPI standard.
Proposed Approach

• We use MPICH2 with one master and four slaves nodes to implement a distributed database environment to handle two scenarios.
  • Having one test image and a very large database, searching for a matching face could take too long if done on a single processing node.
  • When the training database is somehow fixed, or rarely updated and the training steps are done once or infrequently

• Every processing node has a complete copy of the database. This way, each node performs the training phase once and stores the results in its local memory. When the master receives a stream of testing images, they are distributed to the slaves.
Experimental Results

• Two proposed approaches are evaluated by applying them to different database sizes and measuring the speed-up.
• Main metric for evaluation is the execution time
• The proposed architectures on a cluster hosted on the Faculty of Computer and Information Sciences, Ain Shams University, Egypt.
  • Cluster had 2 blade chassis; each having six blade server.
  • Each blade server has two Quad-Core Intel Xeon CPU E5520 @ 2.27GHz with 24GB RAM.
  • All twelve servers are connected together on an infinite band network with VMware ESXi installed directly on each server.
  • Vsphere client is used to post jobs on these servers.
  • Five servers are used in the presented experiments and the Facial Recognition Technology (FERET) database is used in training and testing
Experimental Results

• Distribute Training Database and Duplicate Test Image
  • The experiments are carried out using one master node and four slaves with five different database sizes. In the training phase, the master divides the training database equally over the slaves and itself.
  • The maximum speed-up scored is 25X when the training database is distributed on five servers, achieving superlinear speed-up.
  • In the recognition phase, the master node sends copies of the testing image to the four slaves. Each slave tries to recognize the testing image against its local training set and sends the result back to the master.
  • After distributing the databases on different numbers of servers, the recognition time decrease and the speed-up increases linearly to 5X when using five servers.
Experimental Results

• Sequence diagram to recognize 1 test image vs 1000 images in training DB
Experimental Results

• Duplicate Training Database and Distributed test Image.
  • The training database is fixed and the training phase is done once at the master node.
  • In the recognition phase, a number of test images are captured from video camera attached to the master node which distributes them to the slaves in a manner similar to the one used in the previous.
  • In case of 500 test images when searching in training database have 500 images, the sequentially required 757.172 seconds and recognition time dropped to 151.313 seconds on five machines.
Experimental Results

• Sequence diagram to recognize stream of testing images in training DB
Conclusion

• Distributing the training set and duplicating the test image is most likely the best solution when the stored training images are updated regularly and there is only input test at a time.

• Having a large number of test faces or processing a video stream for recognition, is best handled by centralizing the training set and distributing the test images.

• The proposed systems improve execution time up to 25X in training and 5X in recognition phase, reaching super-linear and linear speed-up.
Alternative Approach

• One approach suggest a use of distributed computing to improve recognition on huge databases, specifically, TH-FACE.

• Their proposed algorithm uses special distributed parallel architecture with mmx technology to speed-up the matching process in PCA algorithm.

• They use multimodal face recognition method (MMP-PCA).

• A multimodal part face recognition method based on principal component analysis (MMP-PCA) is adopted to perform the recognition task, and the MMX technology is introduced to accelerate the matching procedure.
MMX Technology

• As an extension to the basic Intel Architecture (IA), Intel’s MMX technology is designed to improve the performance of multimedia and communication algorithms.

• The technology includes new instructions and data types, which achieve new levels of performance for these algorithms on host processors. The usage of MMX technology speeds up the matching by 14 times.
System Framework
Client Structure
Distributed parallel system architecture

• Parallel Virtual Machine (PVM)
  • The PC cluster used in this face recognition system forms a parallel virtual machine, which consists of a single host and five slaves. The PVM transparently handles all message routing, data conversion, task scheduling across the network.
  • The PVM environment is set up by the host, and then a PVM table is created and maintained by host. The PVM table is a linked table. The host node holds the face feature database and document information database, while the slave nodes hold their IP address and the name of sub-database [2]. The linked table structure permits to add slaves infinitely and also delete a slave very conveniently.
Distributed parallel system architecture

• Communication Process
Distributed parallel system architecture

• Distributed database
  • The PVM-based distributed system has to transfer messages between each node of the system. To decrease the spending of communication, the main face database (MFDB) is split into five sub-databases (SFDB). So only the face feature data to be queried should be transferred between the host and slaves, also between the clients and the host.

• Multithreading in flow control
  • In order to respond to query requests from multi-clients, multithreading technique is needed in flow control. Multithreading means that the concurrent operation of more than one path can be executed within one course. All the threads shared the resource and state of the course.
The MMX-based recognition algorithm

**MMP-PCA face recognition method**

- The first step is to segment the face parts. According to the structure of human faces, we divide a human face into five parts: bare face, eyebrow, eye, nose and mouth.
- Then principal component analysis (PCA) is performed on the facial parts. We calculate the eigenvalues of each facial part, choose $d$ largest eigenvalues (we use $d=60$) and preserve corresponding eigenvectors.
- In this way, the eigenface, eigeneyebrow, eigeneye, eigennose and eigenmouth can be obtained, respectively. The known human faces’ eigenvalues are stored in the database.
The MMX-based recognition algorithm

• **MMP-PCA face recognition method**
  - In the face recognition procedure, we first calculate the projection eigenvalue of the human face, and then calculate its similarity score with the projection eigenvalues stored in the database.
  - After that, we sort the faces in the database according to the similarity scores from large to small.
  - Let $X$ represent the projection eigenvalue of the human face to be recognized, $\bar{X}$ represents the projection eigenvalue of a known human face in database, then the similarity score is computed as follow:

$$s(\bar{X}, Y) = \sqrt{1 - \frac{\|\bar{X} - \bar{Y}\|}{\|\bar{X}\| + \|\bar{Y}\|}}$$
MMX-based matching procedure

• The MMX technology exploits the parallelism inherent in many multimedia and communication algorithms.
• Many of these algorithms exhibit the property of “fixed” computation on a large data set,
  • Small, native data types (for example, 8-bit pixels, 16-bit audio samples)
  • Regular and recurring memory access patterns;
  • Localized, recurring operations performed on the data;
  • Intensive computation.
• The matching procedure is carried out by comparing the projection eigenvalue according to (2). The value is in the range of 0 to 1: the larger the value is, the more similar the two eigenvalues are.
Data Structure of Eigen Value and MMX Implementation

• Each eigenvalue data is presented by 16-bit, and the number of data in eigenface, eigeneyebrow, eigeneye, eigennose and eigenmouth is 100, 60, 80, 50 and 40, respectively. Because the MMX Registers are all 64-bit, each read-in operation can get 64-bit / 16-bit = 4 data.
• In order to achieve the principle of data matching, the length of each eigenvalue should be divided exactly by 4. Only the length of eigennose can not satisfy this requirement, so we add zero in the eigennose, which make the length of eigennose 52.
System Performance

• **MMX acceleration test**
  
  • In this, we use three different size databases. On each database, 10 different face images are recognized, and the matching time is recorded, then we assume the mean value of the 10 recorded times as the matching time.
  
  • As shown in table, with the usage of MMX technology, the matching speed is increased by about 14 times.

<table>
<thead>
<tr>
<th>Database size</th>
<th>without MMX (T1)</th>
<th>with MMX acceleration (T2)</th>
<th>difference in time (T1-T2)</th>
<th>accelerating ratio (T1/T2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>50,000</td>
<td>2.1766 s</td>
<td>0.1526 s</td>
<td>2.0240 s</td>
<td>14.26</td>
</tr>
<tr>
<td>100,000</td>
<td>4.3375 s</td>
<td>0.2992 s</td>
<td>4.0383 s</td>
<td>14.50</td>
</tr>
<tr>
<td>200,000</td>
<td>8.7238 s</td>
<td>0.6026 s</td>
<td>8.1212 s</td>
<td>14.48</td>
</tr>
</tbody>
</table>
System Performance

• Furthermore, as the database size increases, the actual improvement of MMX acceleration is even larger.
System Performance

• **Distributed parallel architecture acceleration test**
  
  • In the test for the distributed parallel architecture, we record the mean value of 10 matching time. One case is searching on one PC and other cases are searching on the distributed parallel architecture with k PCs linked by 1000M network switch.

  • In the first case, the PC in experiment holds a database of 400,000, and in the other cases, each PC holds a database of 400,000/ k. Both cases use the MMP-PCA algorithm without MMX acceleration. As shown, with the adoption of the distributed parallel architecture, the acceleration ratio can approach the number of PCs.
Conclusion

• The system presented above displays excellent performance on large, or even huge databases: querying one face image in 2,560,000 faces costs only 1.094s and the identification rate is above 85% in most cases.

• Such great performance is achieved by the particular architecture and ingenious algorithms of this system.

• Hence, from the two approaches presented here, we can conclude that parallel implementation and parallel architecture are useful in reducing computation time significantly in face recognition.
References


• Parallel Architecture for Face Recognition using MPI, Dalia Shouman Ibrahim, Salma Hamdy ,Computer Science Department, Computer and Information Sciences, Ain shams University, Egypt
Questions ??