Received May 15, 2019, accepted August 17, 2019, date of publication August 30, 2019, date of current version November 25, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2938671

3D Face Reconstruction From Volumes of Videos Using a Mapreduce Framework

WANSHUN GAO[1], XI ZHAO[2,3], (Member, IEEE), ZHIMIN GAO[4], JIANHUA ZOU[1], (Member, IEEE), PENGFEI DOU[4], AND IOANNIS A. KAKADIARIS[4], (Senior Member, IEEE)

[1]School of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an 710049, China
[2]School of Management, Xi'an Jiaotong University, Xi'an 710049, China
[3]Key Lab of the Ministry of Education for Process Control and Efficiency Engineering, Xi'an 710049, China
[4]Department of Computer Science, University of Houston, Houston, TX 77204, USA

Corresponding author: Jianhua Zou (jhzou@sei.xjtu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 91746111 and Grant 71702143, in part by the Ministry of Education and China Mobile Joint Research Fund Program under Grant MCM20160302, in part by the Shaanxi Provincial Development and Reform Commission under Grant SFG2016789, in part by the Xi'an Science and Technology Bureau under Grant 2017111SF/RK005-(7), and in part by the Fundamental Research Funds for the Central Universities.

ABSTRACT As video blogs become favorable to the commonage, egocentric videos generate tremendous big video data, which capture a large number of interpersonal social events. There are significant challenges on retrieving rich social information, such as human identities, emotions and other interaction information from these massive video data. Limited methods have been proposed so far to address the issue of the unlabeled data. In this paper, we present a fully-automatic system retrieving both sparse 3D facial shape and dense 3D face, from which more face-related information can be predicted during social communication. First, we localize facial landmarks from 2D videos and retrieve sparse 3D shape from motion. Second, we apply the retrieved sparse 3D shape as a prior estimation of dense 3D face mesh. To deal with big social videos in a scalable manner, we design the proposed system on a Map/Reduce framework. Tested on FEI and BU-4DFE face datasets, we improve time efficiency by 92% and 73% respectively without accuracy loss.

INDEX TERMS Facial shape retrieval, 3D face reconstruction, cloud computing, map/reduce.

I. INTRODUCTION

People share their videos on the video sharing platforms, such as YouTube or Vimeo, for public access [1]. These platforms are foreseen to explode with egocentric videos, while mobile and wearable devices are available for various scenarios. Egocentric videos are generally recorded by wearable devices (*e.g.*, google glass), and typically contain faces of an individual or a group of people socially interacting with wearable device users. These videos normally contain rich social information, such as the identities of these people, emotions, and social interaction content. Retrieving such semantic information can be further employed in applications such as lifelogging, social science, and behavioral research. Currently, little information has been retrieved from these videos. After a simple data filtering, these videos are stored in the cloud-based platform for further viewing. The potential value of this type of big data has been limited by a lack of proper cloud-based analytic tools to automatically explore the semantic information of facial reactions appearing on others in social interaction.

A critical topic of facial analysis is retrieving 3D face from footage. Social information stored as big video data will confront the problems of network traffic and storage space. As a type of text file, retrieved 3D faces significantly reduces above two problems. Meanwhile, the 3D face model contains well-rounded facial information, which can be utilized to pose estimation, facial expression, face identification, and verification analysis. Furthermore, sparse facial geometry information can be used as a prior estimation in reconstructing dense 3D face mesh, which has been widely analyzed and is claimed to have the advantage of being robust for head pose and illumination conditions.

However, there are several typical challenges when performing video analytics on the cloud platform. First, most cloud front-ends (*e.g.*, Storm, Spark, and VoltDB) are built on Hadoop which is mainly designed for processing text data. Second, it is not straightforward to apply Map/Reduce

The associate editor coordinating the review of this manuscript and approving it for publication was Zhenhua Guo.

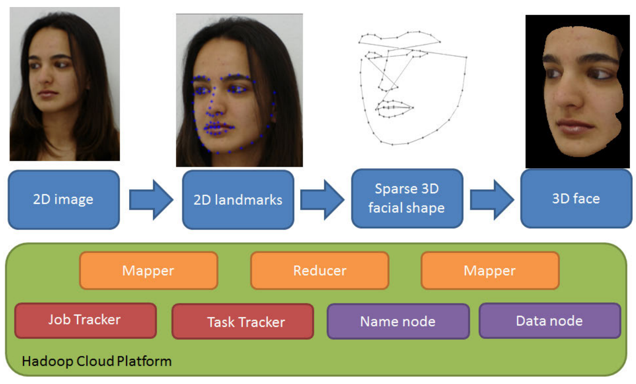


FIGURE 1. The proposed 3D facial shape reconstruction framework. Facial landmarks are first located on 2D facial image sequences frame-wisely. Then sparse 3D facial shapes are retrieved from the batch of images in a sequence. These shapes are then frame-wisely applied as a priori in 3D face reconstruction.

framework for processing data other than text. The basic idea of the Map/Reduce framework is to segment text files into small sections and process them separately. However, it is not always feasible to segment 2D data as images and texture maps into small segments because essential features may be disrupted after the segmentation. Third, it is challenging to mitigate single machine based video analytic tools onto a cloud platform to waive the human intervention on data organization of inter-algorithm and scale out handling large volumes of video data. To the best of our knowledge, we are the first to overcome these challenges.

In this paper, we propose an automatic video analytic system on a Map/Reduce framework which retrieves both sparse 3D facial shape and dense 3D face in a scalable manner, as depicted in Fig. 1. The sparse shape is used as prior knowledge to reconstruct dense 3D face meshes. We first decompose the sparse 3D facial shape retrieval into 2D facial landmark detection for mappers and sparse 3D shape retrieval for reducers. Image sequences of video footage are sent to mappers for landmark detection. A set of facial landmarks that detected on each frame in the footage is computed by the reducers to retrieve sparse 3D facial shapes using a SFM(structure from motion) method. The output of the reducers is a single text file including sparse 3D shapes in frames. Then these 3D shapes are fed into another mapper for reconstructing dense 3D face meshes frame-wisely. By reimplementing the Map/Reduce framework, the proposed system can reconstruct 3D faces from big video data automatically.

The initial design of the Map/Reduce framework is to process segmented text data on a distributed data storage cloud. The proposed system provides a novel method for reconstructing 3D faces from videos on a cloud platform. Different from a typical SFM method or shape-from-shading 3D reconstruction method, it decouples the 3D face reconstruction into three components, namely 2D facial landmark localization, sparse 3D shape retrieval, and dense 3D mesh reconstruction. Moreover, in order to perform video analysis, the proposed pipeline adapts the Hadoop file access which avoids its original 'split' action on an input file.

The contributions of this paper are:

1) we propose an automated video analytics pipeline which reconstructs dense 3D face meshes from 2D videos.
2) we improve the 3D face reconstruction by using sparse 3D shapes as a prior estimation to reconstruct the dense face mesh and register its corresponding facial texture.
3) we improve the Hadoop file I/O, which enables processing video data in a scalable manner.
4) we evaluate the proposed pipeline on two publicly accessible databases and demonstrate its efficiency and scalability on the cloud platform.

The rest of the paper is organized as follows: Section II discusses related works. The proposed approach to reconstruct 3D faces is presented in Section III. The system implementation on the cloud is detailed in Section IV. We present the experimental results in Section V and conclude the paper in Section VI.

II. RELATED WORKS

Many research problems related to human face analysis have been extensively studied (*e.g.,* human detection [2], [3], identity verification and recognition [4] and emotion analysis [5]). Driven by the recent public availability of unconstrained facial data [6], facial analysis has become more robust when applied in natural environments.

Several studies reconstructed a 3D face from 2D images or videos via SFM [7], shape from shading [8] and 3D model fitting [9] based approaches. Dovgard and Basri [10] introduced a SFS(shape from shading)-based method by taking into account not only statistical constraints but also a geometric constraint of facial symmetry. By assuming the orthographic projection and Lambertian reflectance model, brightness constraints that incorporate unknown surface albedos and surface depths could be derived. Smith and Hancock [11] proposed to embed a statistical model of surface normals instead of surface depths into the SFS framework. Their approach recovers a field of surface normals from a single intensity image by exploiting the direct relationship between surface orientations and image intensities. Rara *et al.* [12] proposed another SFS-based approach. By employing spherical harmonics (SH), the proposed method achieves the capability of dealing with arbitrary illumination. Jiang *et al.* [13] have studied the complete 3D face reconstruction by fitting a 3D morphable face model to 2D images. Kemelmacher-Shlizerman and Basri [14] proposed to use the input image as a guide to "mold" a single reference model to recover the corresponding 3D shape of either a different individual or a generic face. Lei *et al.* [15] recover face meshes from a single image using CCA mapping between tensor spaces.

According to state-of-the-art evaluation [16], most 3D face mesh recovery approaches have a heavy computational burden, which prevents applying them to video data that have large volume [17]–[19]. Since the optimization tools widely used in these methods are designed for single-threaded computation on local machines, it is also challenging to reinvent these methods into the Map/Reduce framework and deploy them on a cloud platform. In contrast to these methods, our method decomposes the reconstruction process into three steps, frame-wise landmark localization, batch-wise sparse 3D shape reconstruction, and frame-wise 3D face reconstruction. This decomposition makes the method possible to fit into the Map/Reduce framework. Also, the pipeline is flexible to fulfill different requirements. In some scenarios where only the critical information of faces is needed (*i.e.*, the motion of key feature points in 3D space), the pipeline on the cloud can skip the 3D face reconstruction process to save the computational time and resources.

Several video analytic tools have been migrated to the cloud platform (*e.g.*, motion detection [20], video tagging [21], and face recognition [22]). However, there is no cloud-based video analytic system to recover 3D facial information from videos. To our best knowledge, our system is the first cloud-based system to address this challenge.

III. 3D FACIAL SHAPE RECONSTRUCTION METHOD

In this section, we present our approach to retrieve sparse 3D facial shapes and reconstruct 3D faces. To achieve this objective, we decompose our approach to three algorithms, namely facial landmark localization, sparse 3D shape retrieval and 3D face reconstruction.

A. EFFICIENT FACIAL LANDMARK LOCALIZATION

To initialize the facial landmarking algorithm, we detect faces in every frame using a face detector, which searches regions of interest (ROI) covering faces and predicts four vertexes of their bounding boxes. The face detector, which is adapted from [23], uses a multi-scale sliding window to detect potential faces and employs the Histogram of Oriented Gradients (HOG) feature to classify the possibility of potential faces linearly. It performs well on faces with expressions [24].

Then, a cascade of tree-based regressors is applied to localize vital facial points. Let $x_i \in \mathbb{R}^2$ be the x, y-coordinates of the i^{th} facial landmark in an image I. The facial shape is denoted as a vector $\mathbf{S} = (x_1^T, x_2^T, ..., x_p^T,)^T \in \mathbb{R}^{2p}$, where p equals 68 in our tests. The localization method can be initialized using a mean shape centered and scaled according to ROIs. The regressor in each cascade t, denoted as $\mathcal{R}^t$, takes the image I and a shape $\mathbf{S}^{t-1}$ as input. It predicts an updated vector $\Delta\mathbf{S}^t$ that is added to the current shape $\mathbf{S}$:

$$\begin{aligned}\Delta\mathbf{S}^t &= \mathcal{R}^t(I, \mathbf{S}^{t-1})\\ \mathbf{S}^{t+1} &= \mathbf{S}^t + \Delta\mathbf{S}^t.\end{aligned} \tag{1}$$

For each regressor $\mathcal{R}^t$, a constant vector $\Delta\mathbf{S}_l^t$ is fit to each leaf node. The $\mathbf{S}^t$ starts at the root node of tree $\mathcal{R}^t$ and traverses a sequence of internal nodes until it hits one leaf node. These nodes are labeled with functions, and the edges or branches below them labeled by the function outputs. Features and thresholds have been assigned to each interior node during the training process. A feature pool is created hosting a large set of relative point coordinate pairs and thresholds for their intensity difference values. Each regressor is trained using the gradient boosting tree algorithm.

B. SPARSE 3D FACIAL SHAPE RETRIEVAL

Sparse 3D facial shapes are retrieved from an image sequence with shape and motion estimated. However, facial shape and motion are ambiguous while the videos are generally recorded by only one camera. To resolve this ambiguity, we assume that the 3D shape follows a Gaussian Probability Density Function (PDF). Therefore, we only need to extract a set of labeled landmark tracks from an image sequence without additional parameters known in advance.

The 2D facial landmark vector $\mathbf{S_t}$ is orthographically projected from 3D facial point $\mathbf{X_t}$, $\mathbf{S_t} = \mathbf{R_t}(\mathbf{X_t} + \mathbf{D_t}) + N$, where $\mathbf{R_t}$ is the projection matrix, $\mathbf{D_t}$ is the translation vector, N is zero-mean Gaussian noise with variance σ, t is the frame index. The shape is factorized by shape basis vector $\bar{\mathbf{X}}$, $\mathbf{V}$ and weights $\mathbf{Z}$ as $\mathbf{X_t} = \bar{\mathbf{X}} + \mathbf{VZ}$. Shape $\mathbf{X_t}$ is assumed to follow a probability distribution $p(\mathbf{X_t}|\theta)$ with a known parameter θ which follows $\mathcal{N}(\bar{S}; \sigma^2 I)$. The shape and motion are estimated by maximizing

$$p(\mathbf{R}, \mathbf{D}, \theta, \sigma^2|\mathbf{S}) \propto \int \prod_t p(\mathbf{S}_t|\mathbf{X_t}, \mathbf{R_t}, \mathbf{D_t}, \sigma^2)p(\mathbf{X_t}|\mathbf{D})d\mathbf{X} \tag{2}$$

To estimate shape and motion while learning the parameters of the PDF $p(\mathbf{S}|\theta)$, a generalized EM algorithm is adopted. In the E-step, we estimate the distribution over shape weights $\mathbf{Z}$ given the current motion and shape estimates for each frame. Then we estimate shape basis $\bar{\mathbf{X}}$, $\mathbf{V}$, noise variance σ^2, projection parameters $\mathbf{R}$, $\mathbf{T}$ in M-step. The number of EM iterations is normally set to 100 with a balance between convergence and performance.

C. 3D FACE RECONSTRUCTION

After we obtain the sparse 3D facial shapes, we aim to reconstruct corresponding dense 3D facial meshes. Due to the intrinsic correlation between sparse 3D facial shape, which is represented as a collection of facial landmarks, and dense 3D face mesh, we could assume that each sparse 3D facial shape is a sub-sampled version of the dense 3D face mesh. To recover the 3D face model from sparse observation, we adopt a coupled subspace model, which encodes their underlying relationship by forcing them to share the same coefficients in a coupled subspace. Specifically, from a set of N 3D faces $\gamma_{3D}^d = (Y_{3D}^1, \cdots, Y_{3D}^N)$ with their corresponding 3D facial landmarks $\mathbf{X} = (X_{3D}^1, \cdots, X_{3D}^N)$, we build a coupled dictionary model (Eq. 3).

$$\underset{\alpha, \Lambda_{3D}^d, \Lambda_{3D}^s}{\arg\min} \left\| \begin{bmatrix} \beta_0 \gamma_{3D}^d \\ \mathbf{X} \end{bmatrix} - \begin{bmatrix} \beta_0 \Lambda_{3D}^d \\ \Lambda_{3D}^s \end{bmatrix} \alpha \right\|_2^2 \quad s.t.\ \|\alpha\|_1 \leq \beta_1. \tag{3}$$

Eq. 3 can be solved off-line using the K-SVD [25] algorithm, where Λ_{3D}^d is the dictionary of 3D facial

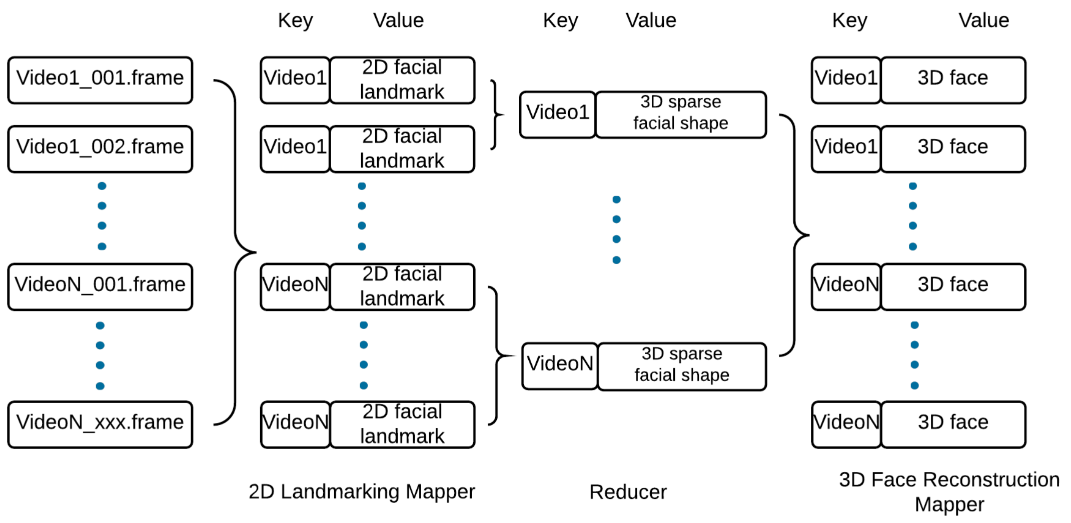


FIGURE 2. The proposed pipeline design using the Map/Reduce framework.

landmarks, Λ_{3D}^{s} is the dictionary of 3D facial shape, and β_1 and β_0 are two parameters that control the sparsity of the coefficient and restrict the dimensionality imbalance between two types of training data. All parameters including dictionary size T are tuned empirically during experiments.

For each video frame, we first recover the coefficient α^* by solving Eq. 4:

$$\arg\min_{\alpha^*} \left\|\mathbf{X}_t - \Lambda_{3D}^{s}\alpha^*\right\|_2^2 + \beta_2 \left\|\alpha^*\right\|_1 + \beta_3 \left\|\alpha^*\right\|_2 . \quad (4)$$

We force α^* to have similar sparsity as in Eq. 3 by automatically tuning β_2 and adopt the LASSO algorithm [26] to obtain the solution. We also apply L_2 regularization on the solution α^*. When we get the coefficient α^*, the corresponding dense 3D facial mesh $\mathbf{Y}_{3D}^{R}$ is reconstructed via $\mathbf{Y}_{3D}^{R} = \frac{\Lambda_{3D}^{d}\alpha^*}{\beta_0}$.

IV. SYSTEM IMPLEMENTATION USING THE MAP/REDUCE FRAMEWORK

A. SYSTEM OVERVIEW

The Map/Reduce framework manages and processes big data on multi-nodes in parallel. It is an implementation of cloud computing, inheriting its two critical concepts, integration and distribution. To initialize a Map/Reduce job, the job tracker slices the source data into small batches and assigns them to mappers. Mappers compute batches in parallel, and yield $\langle key, value\rangle$ pairs as a result. Then, the job tracker convenes and conveys entire values tagged the same key to a reducer, which enumerates all intermediate values from mappers and outputs the ultimate result. In our case, we employ two types of mappers and one type of reducers. Landmarking mappers output $\langle key, value\rangle$ pairs. In each pair, the "key" is a unique number for identifying image sequences, and the "value" is a pack of facial landmarks extracted from a frame. The reducers retrieve sparse 3D shape motion from landmarking packs with the same key. With the corresponding sparse 3D shape, reconstruction mappers reconstruct a dense 3D facial mesh from every frame. We depict the whole procedure in Fig. 2.

Our system comprises of three executable binaries. One is a C++ executable binary which extracts landmarks to CSV files. Others are Matlab programs compiled by Matlab Compiler Runtime (MCR), which output sparse 3D facial shapes in CSV files, or 3D face meshes in wrl files. During the Map/Reduce process, we apply these executable binaries to two types of mappers and one type of reducers respectively. Typically, Hadoop supports mappers or reducers running executable binaries but requires to implement the Map/Reduce API. However, our pre-compiled programs only accept string data types as the initial parameters. As a compromise, we utilize Hadoop Java API which invokes the pre-compiled programs by system calls. The details are explained as follows.

B. 2D LANDMARKING MAPPER IMPLEMENTATION

In Hadoop, each mapper copes with an independent task. It output an intermediate record as an input of the following reducer. The types of input records of mappers and intermediate records may be very different. Therefore, to define a "Map" class, we need to declare both types of input and intermediate records. In this paper, each landmarking mapper invokes a core C++ program and preserves the result of facial landmarks. The input parameters of the C++ program are paths of files in three types, which are feature file, facial image, and CSV file. The system produces the feature file with identical and unchangeable local location and replicates it to different nodes before running the program. Besides, the system shares the same folder for input and output files. Consequently, the landmarking mapper feeds one path to the core C++ program. Meanwhile, the system would not split a file into pieces, which is different for Hadoop when shedding

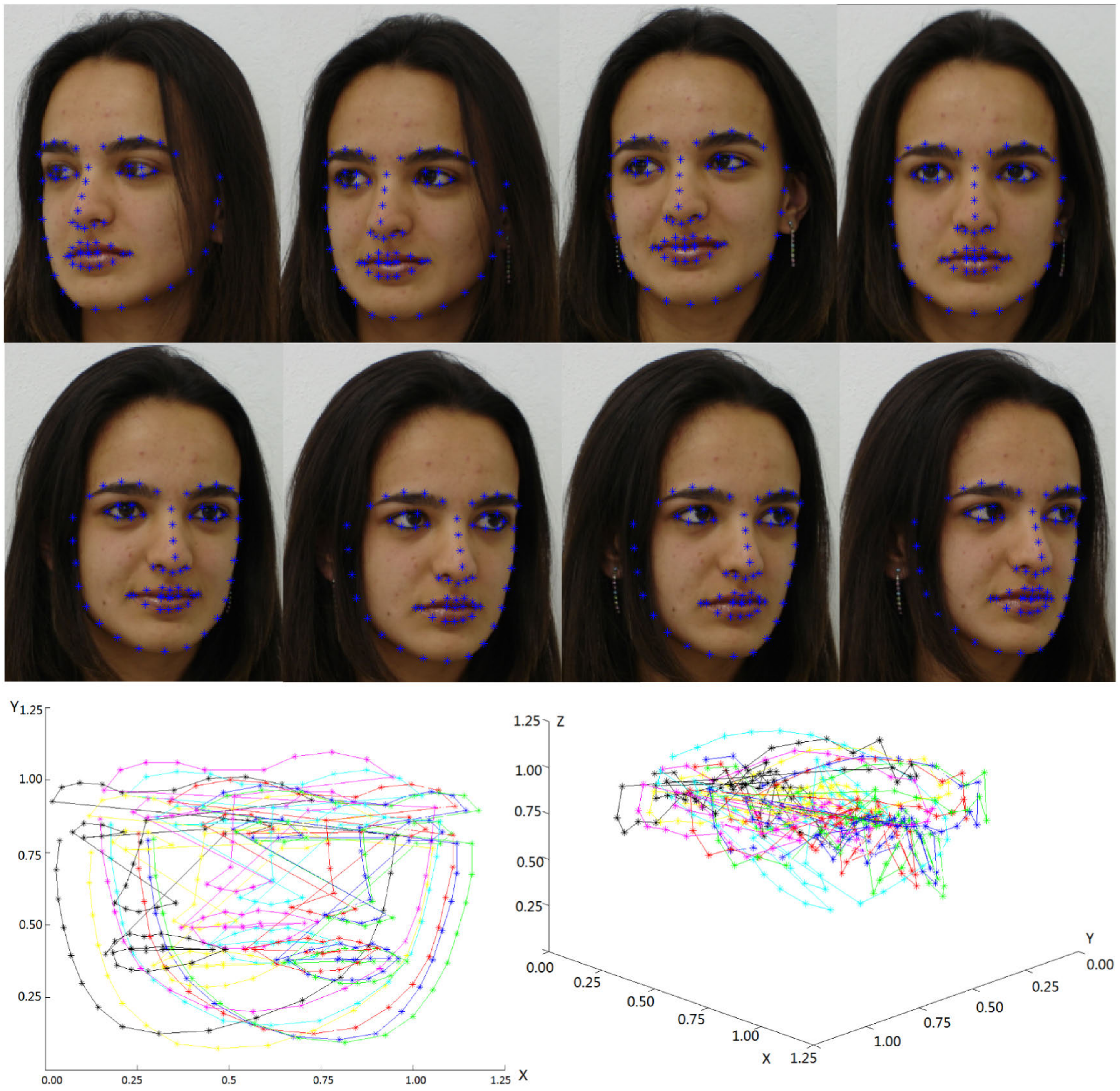


FIGURE 3. Results of sparse 3D facial shape retrieval on a female subject. Plots in the two rows depict the facial landmark localization result on image sequences conducted in the map phase. These frames contain faces from the same video with pose variations.

an input file to mappers. In consequence, we implement two InputFormat and RecordReader classes in Java.

The RecordReader of Hadoop is a function to generate $\langle key, value\rangle$ pairs for mappers and created by the job tracker. To implement this function, the input format of the job tracker has to be pre-defined. To prevent the job tracker from splitting a whole file into pieces, two classes "ImageLocationInputFormat" and " ImageLocationRecordReader" are defined. The class "ImageLocationInputFormat" extends the Hadoop class "InputFormat", and returns false directly by overriding the function isSplitable(). Furthermore, the class "ImageLocationRecordReader" modifies the type of $\langle key, value\rangle$ pairs for the mapper. So the system would ignore keys, which are the indexes of split pieces by default. We declare the type of $\langle key, value\rangle$ pairs as $\langle NullWritable, Text\rangle$, which represent a skipped key and the location of the input image respectively. We also implement the function nextKeyValue() to grab the new $\langle key, value\rangle$ pair, which is transmitted to an allocated mapper by a job tracker. The following codes indicate the modification of this function.

```
@Override
public boolean nextKeyValue()
  throws IOException, InterruptedException {
    if(value==null) {
      Path path = fileSplit.getPath();
      String pathstr = path.toString();
      value = new Text(pathstr);
      return true;
    }
  return false;
}
```

The system distinguishes image sequences with unique tags, and names input files for mappers and reducers in "tag-index" format. When a landmarking mapper is allocated by the job tracker, it fetches the location of an input image and invokes the landmarking program with three parameters. The output CSV file shares the same name with the input image except for the extension. After all, the output ⟨*key*, *value*⟩ pair to the reducer is ⟨*tag*, *csv_location*⟩ and we define the "Map" class as follows:

```
public static class Map extends
    Mapper<NullWritable, Text, Text, Text> {
     @Override
     protected void map(NullWritable key,
         Text value, Context context)
     throws IOException,
         InterruptedException {
       //Initialization
       //Call C++ 'Landmark Localization'
           program
       //Collect output
       context.write(new Text(tag), new
           Text(csv_location));
     }
}
```

C. REDUCER IMPLEMENTATION

A reducer is to merge and process the intermediate values of pieces from mappers to a new file. Sorted intermediate values are transmitted by the HTTP protocol. When a reducer is allocated, it calls the reduce() function to enumerate and process all intermediate values with the same key.

To retrieve sparse 3D facial shapes from an image sequence, the reducer invokes a Matlab program, which can only read a file from the local file system. However, the values (csv_locations) are HDFS URLs and stored on the cloud. Hence, the reducer enumerates all values and downloads CSV files from HDFS to the local file system, before launching the Matlab program. These downloaded files are cataloged in temporary folders by tag. The reducer transmits three parameters to the Matlab program, which are the paths of Matlab runtime library, temporary folder, and output folder respectively. The Matlab program loads all CSV files in the temporary folder and saves the result of 3D face reconstruction to the target folder. The output of the reducer is also a ⟨*tag*, *csv_location*⟩ pair. The csv_location indicates the path of retrieved facial shape motion in CSV files.

The Reduce class is defined as:

```
public static class Reduce extends
    Reducer<Text, Text, Text, Text> {
     @Override
     protected void reduce(Text key,
         Iterable<Text> values, Context
         context)
     throws IOException,
         InterruptedException {
       for (Text text: values) {
         //Download files from HDFS
       }
       //Call Matlab 'Sparse SFM' program
           and process the temporary folder.
       //Output result.
     }
}
```

D. 3D FACE RECONSTRUCTION MAPPER IMPLEMENTATION

We reuse the customized class "ImageLocationInputFormat" and "ImageLocationRecordReader" to retrieve the exact location of the input data for the mapper. The input data are the output of the reducer in the last step, which is in the format of ⟨*video_tag*, *csv_location*⟩ according to Fig. 2. Thus, we need to edit the nextKeyValue() function again and set the key to video tag.

```
key = new Text(video_tag);
value = new Text(pathstr);
```

Because the external program in this round is also a Matlab program, before executing it, we need to set up the right environment by passing the location of MCR library as an argument to the program.

```
public static class Map extends
    Mapper<Text, Text, Text, Text> {
     @Override
     protected void map(Text key, Text
         value, Context context)
     throws IOException,
         InterruptedException {
       //Set environment for MatLab program.
       String[] cmd = {program_location,
           matlab_lib_location,
                      program_arguments...,
                      csv_location,
                          video_tag,
                          output_location};
       //Call Matlab '3D Face
           Reconstruction' program
       //Collect output
       context.write(new Text(tag), new
           Text(wrl_location));
     }
}
```

Each mapper will generate a WRL file, representing a reconstructed 3D face mesh. All files are uploaded and preserved in HDFS. We reserve the "Reduce" phase in this Map/Reduce job for further application such as face recognition. Intermediate information from the mapper including time consumption will be saved in HDFS permanently.

V. EXPERIMENTAL RESULTS

A. EXPERIMENTAL SETUP

We test the proposed system on a private cloud with multiple Virtual Machine (VM) instances running Ubuntu 14.04 OS, physically hosted by four Xeon E5603 servers. The virtual machine hypervisor allocates one virtual core, 1 GB memory and 100 GB disk space for each instance. The maximum number of VM instances was 29. Besides, the 3D face reconstruction program runs on an MCR version of R2014b.

To evaluate the performance of the proposed system, we have adopted two publicly accessible face databases, the FEI face database [27], and the BU-4DFE database [28]. The FEI face database contains a set of face images taken between June 2005 and March 2006. There is a total of 2,800 images from 200 individuals, 14 images for each. All color images are acquired against a homogenous white background in an upright frontal position with profile rotation. The original size of each image is 640x480 pixels. We use eight images from each subject captured as a sequence when the subjects turn their heads from left to right.

BU4D-FE contains 606 dynamic 3D sequences from 101 subjects. Each subject performs six prototypical expressions (*i.e.,* anger, disgust, fear, happiness, sadness, and surprise). Each sequence contains expressions performed gradually from neutral appearance, low intensity, high intensity, and back to low intensity and neutral. The frame number of each sequence varies from 68 to 110, as pairs of colorful images and their co-registered 3D face meshes from the high-resolution 3D scanner.

B. RESULTS ON SPARSE 3D FACIAL SHAPE RETRIEVAL

Fig. 3 depicts the results of sparse 3D facial shape retrieval. The facial landmark algorithm was revoked by mappers for each frame and located 68 facial landmarks on the major facial components, including the eyebrows, the eyes, the nose, the mouth, and the cheek. It can be observed that the facial landmarks have been located accurately across different poses. After the landmarks have been located, the results are collected by the reducer and used to reconstruct a sequence of 3D facial shapes via the SFM algorithm. The reconstructed shape coordinates were projected into a normalized 3D space. We have compared the outputs from these algorithms obtained from both a local machine and the cloud platform, and the results were identical. This verification excluded unexpected computation errors introduced by different computing platform in our tests.

Fig. 4 depicts the completion profiles of mappers and reducers throughout sparse 3D shape retrieval in four configurations. We have set the number of mappers to 1, 5, 10, and 15, while the Hadoop task manager automatically sets the number of reducers. In general, we observed that the completion rate of reducers boosted after mappers finished their jobs. It is because reducers allocated more computing cores which were released from mappers. As the number of pairs increased, the time intervals were decreased between the completion of mappers and reducers. It is because more reducers had shared the tasks and improved time efficiency.

Fig. 5 depicts the comparison of completion time where the number of mappers increased from one to fifteen.

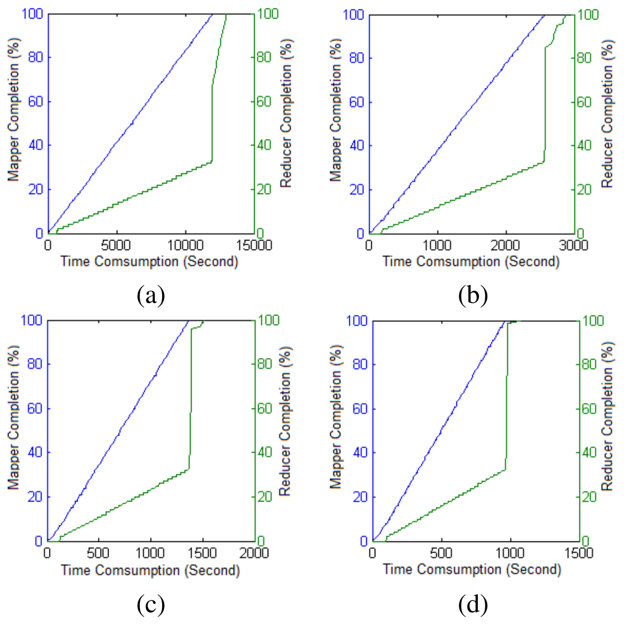


FIGURE 4. Completion profiles for sparse 3D facial shape retrieval with different numbers of mapper and reducer pairs in the configuration. The completion profile using one mapper (a), five mappers (b), ten mappers (c), and 15 mappers (d).

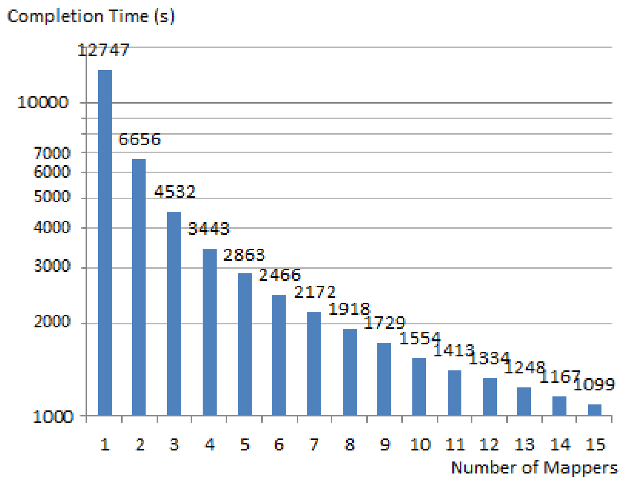


FIGURE 5. Completion profiles under different number of mappers. The completion time decreased from 12747*s* to 1099*s*, as the number increased from one to fifteen.

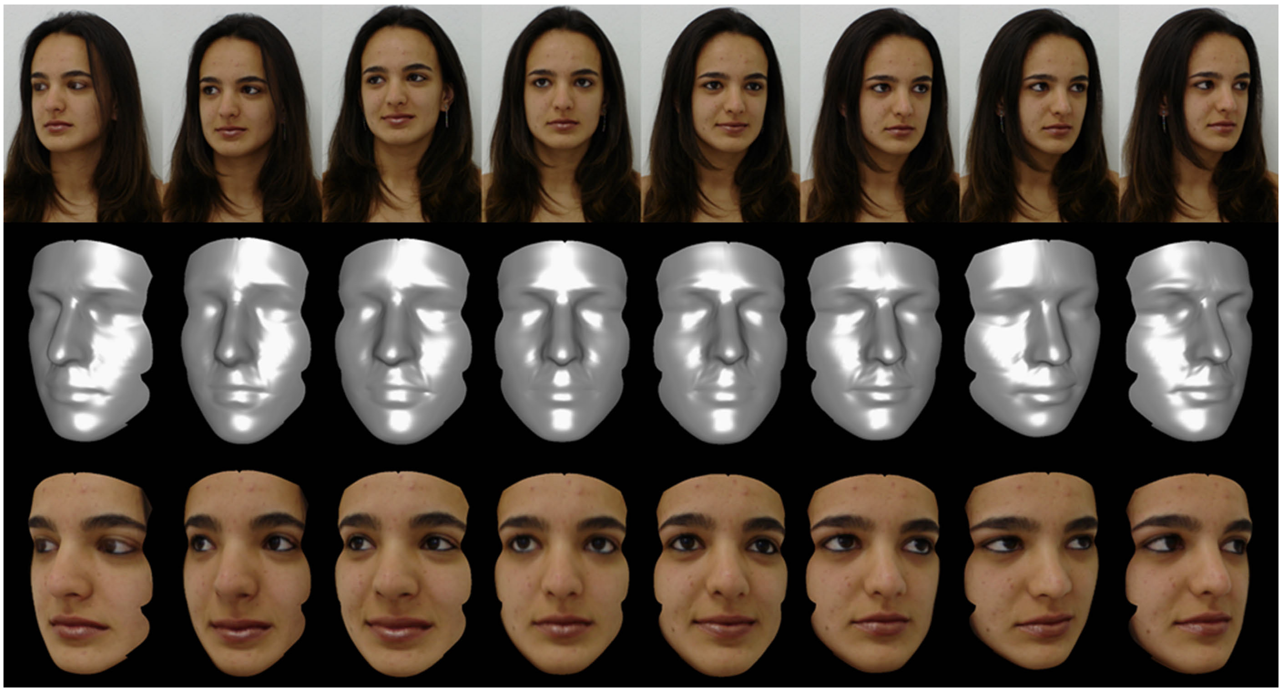


FIGURE 6. Results on 3D face reconstruction from original facial images and sparse 3D facial shapes. The top row depicts the original faces in a sequence. The middle row depicts their corresponding reconstructed face meshes. The bottom row depicts their co-registered facial texture.

When one mapper is in Hadoop, the completion time was more than 100,000 seconds. When 15 mappers are in Hadoop, the completion time was 1099 seconds. The completion time is reduced by 92%, as the number of mappers increased.

C. RESULTS ON 3D FACE RECONSTRUCTION

Fig. 6 depicts the reconstructed 3D faces from the original 2D facial images as well as their co-registered facial textures. Each image has been retrieved as a sparse 3D shape which was used as a prior for dense 3D shape reconstruction. 3D face reconstruction mappers have been assigned with a facial image and its sparse shape and outputted the 3D facial mesh as well as its registered facial texture.

Fig. 7 compares the 3D faces reconstructed directly from the single image based method by [29] (column b) and the proposed pipeline (column c). It can be observed that the artifacts in the nasal region have been reduced significantly. The reason behind is monocular facial shape reconstruction like [29] has an inherent deficit of accuracy on estimating depth, especially when faces are frontal. However, in the proposed system, sparse 3D facial shapes were retrieved from a sequence of images with different head poses. The SFM-based computation in Sec.III.B was constrained by a statistical shape model learned from its specific image sequences. With this sparse shape as a prior estimation, 3D face reconstruction in our pipeline is more photo-realistic and accurate in depth estimation.

BU-4DFE is widely used on emotion recognition [30] and 3D face alignment [31]. In the test on BU-4DFE, we first used 2,340 reconstructed 3D faces from 36 sequences of facial

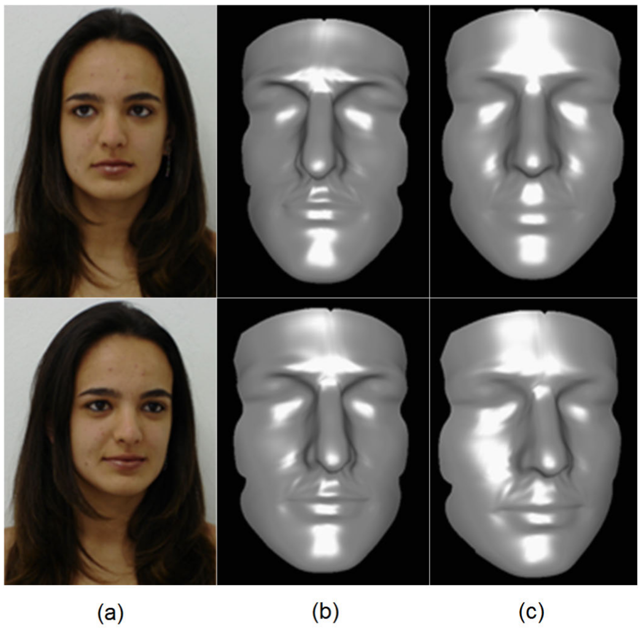


FIGURE 7. Comparison on 3D face reconstruction. Column (a) depicts the original images; column (b) depicts the 3D face reconstructed from single images by [29]; column (c) depicts the 3D face reconstructed from our pipeline.

expressions (six from each expression, randomly selected). The reconstructed faces were compared to the original 3D face meshes. The root-mean-square error was computed for each 3D face. Fig. 8 depicts the statistics of 3D face reconstruction errors w.r.t. each facial expression. The error

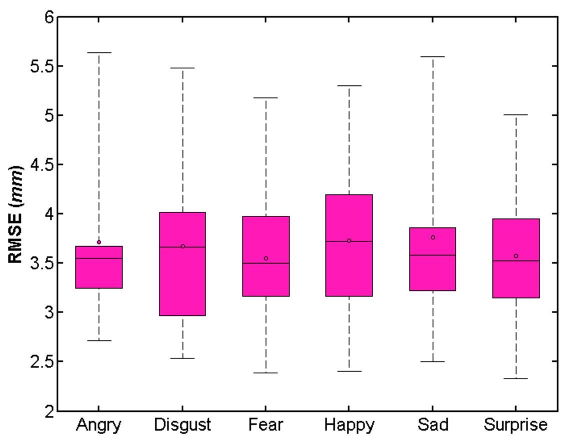


FIGURE 8. Boxplot of reconstruction error on BU-4DFE dataset. The Y axis is the root-mean-square error (RMSE) in mm, and the X axis is the facial expression categories. The bottom and top of the box are the first and third quartiles, the band inside the box is the median, and the circle is the mean.

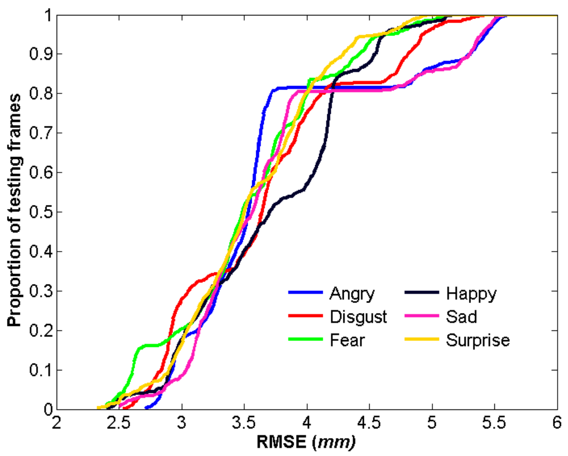


FIGURE 10. Cumulative distribution of reconstruction error on BU-4DFE dataset. The Y axis is the proportion of testing frames, and the X axis is the root-mean-square error (RMSE) in mm.

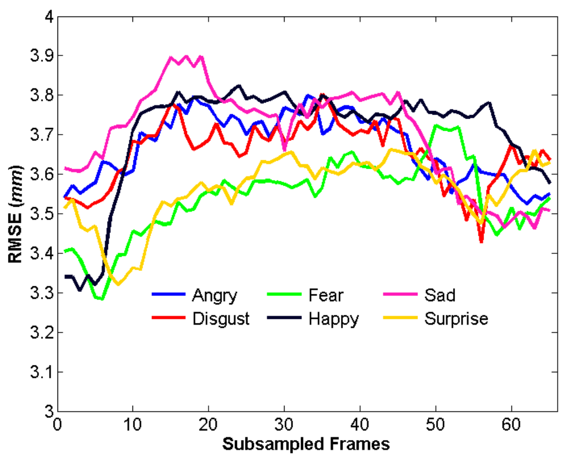


FIGURE 9. Reconstruction errors over the dynamics of facial expressions. The Y axis is the root-mean-square error (RMSE) in mm, and the X axis is the frame number of the facial expression sequence.

means for different expressions ranged from 3.61*mm* to 3.83*mm*. Even with exaggerated expressions (*i.e.,* surprise and happy) where large deformation occurs on faces, no significant variance has been observed on the reconstruction error mean. It demonstrated the robustness of our 3D reconstruction method to the facial expression variation.

Fig. 9 depicts reconstruction errors over the dynamics of facial expressions. The average RMSE for each expression has been plotted as a line, which demonstrates the reconstruction error through the evolution of facial expressions. All the expression sequences evolved from neutral, onset, peak and offset phases. It can be observed that in general errors were relatively small in the neutral phase, and then increased during the onset phase and decreased during the offset phase. The maximum error was obtained frequently in the peak phase. For some expressions (*e.g.,* fear and surprise), the errors during the onset phase are less than the errors during the offset phase, while for some other expressions (*e.g.,* sad and disgust) higher errors are obtained during the onset phase. It is probably because the reconstruction accuracy is influenced by the difference in triggered facial action units [32] in expressions.

Fig. 10 depicts the cumulative distribution of the reconstruction error over 2340 reconstructed 3D faces. It can be observed that most of the reconstruction errors are distributed around 3.5mm across the six universal expressions. Almost all faces have been reconstructed with RMSE error less than 5.5mm. The results demonstrated the accuracy of the proposed 3D face reconstruction pipeline.

We then used the whole BU-4DFE dataset for cloud computing evaluation, which contains 606 image sequences. About 7.3G data has been digested, and about 106G data has been generated in each test. Fig. 11 depicts the completion profiles of mappers and reducers throughout 3D face reconstruction in four configurations. We have set the number of mappers to 1, 10, 15, 20, and 29, while the number of reducers is automatically set by the Hadoop task manager. It is observed that, after reducers finish sparse 3D face shape retrieval, the nodes have been assigned again as mappers to continue 3D face reconstruction tasks.

Fig. 12 depicts the completion time comparison where the number of mappers increased from one to 29 in 3D face reconstruction. When we set one mapper in Hadoop, the completion time was around 504,720 *s*. When we assigned 29 mappers, the completion time was around 140,090 *s*. As the number of mappers increased, the completion time reduced by 72.3%. It can be observed that the completion time achieved by 25 nodes is quite close to the completion time achieved by 29 nodes. The reason is probably that the computation time for file copying has a significant portion in the overall computation, and this time cannot be reduced by increasing the number of nodes.

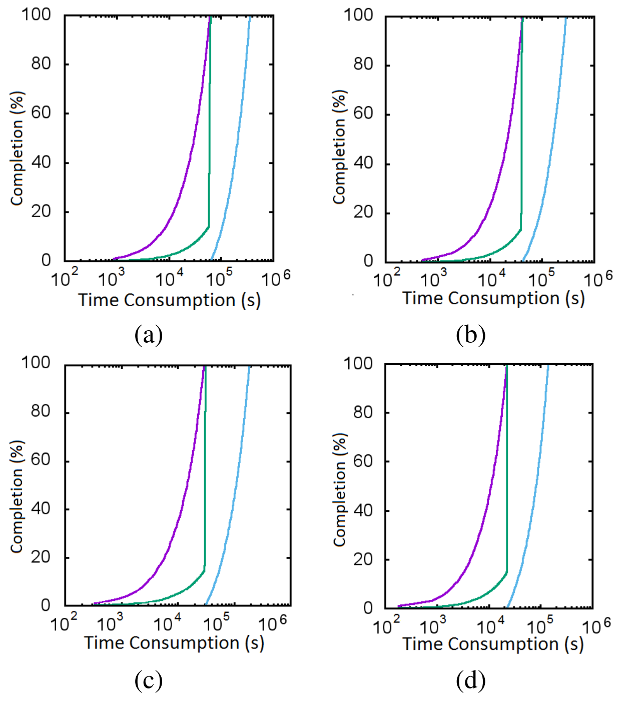


FIGURE 11. Completion profiles for 3D face reconstruction with different numbers of mappers in Hadoop configuration. The purple, green and blue lines represent 2D facial landmarking mapper, reducer and 3D facial reconstruction mapper respectively. (a) the completion profile using ten mappers, (b) the completion profile using 15 mappers, (c) the profile using 25 mappers, and (d) the profile using 29 mappers.

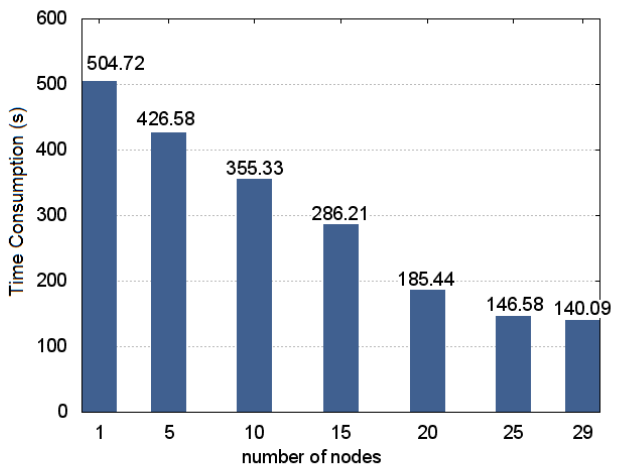


FIGURE 12. Results on completion times under different numbers of mappers. As the number increased from one to 29, the completion time gradually decreased from 504,720 sec to 140,090 sec.

Table 1 depicts the mean, min, and max time for each task in a job. The tasks in our system include copying files from local to HDFS, copying files from HDFS to local, mapper 1 task, reducer task, and mapper 2 task. Copying files from the local to HDFS file system consumes a significant amount of time (17.041s on average). It is because HDFS needs to back up several copies of the image data to ensure data redundancy. However, it is quite efficient to copy files from the HDFS to local (0.398s on average), where the system does not need to copy several duplications in the local file system. Among mapper and reducer tasks, the mapper 2 task consumed the longest time, 10.901s on average. It is because of the complexity of the 3D face reconstruction algorithm. The time variations on mapper and reducer tasks were mainly caused by system computation overhead.

TABLE 1. Job time calculation.

(sec)	L2H Copy	H2L Copy	L. Mapper	Reducer	R. Mapper
Mean	17.041	0.398	1.743	0.389	10.901
Min	15.273	0.319	1.142	0.061	7.714
Max	22.747	0.571	2.309	2.312	11.867

Each job has mainly five tasks, including copying files from local to HDFS, copying files from HDFS to local, landmarking mapper task, reducer task, and reconstruction mapper task. Note that it has limited affection for the consumed time of an individual task when changing the number of mappers.

TABLE 2. Time consumption per frame and RMSE.

	Time	RMSE
SiftFlow [33]	15	4.89
MFSF [34]	20	6.33
EPICFlow [35]	50	4.02
Ours 1 Node	12.24	3.72
Ours 29 Nodes	3.40	3.72

TABLE 3. Time consumption - a comparison with Open MPI (Time is in 10^3 seconds).

	10 Nodes	15 Nodes	20 Nodes	29 Nodes
Map/Reduce	63.86	42.24	30.88	22.41
Open MPI	64.84	36.80	24.19	21.80

D. COMPARED TO STATE-OF-THE-ART

To evaluate the performance of the proposed method, we use the whole BU4DFE dataset for computing the time consumption and RMSE per frame (time is in seconds). Compared to other novel methods, Table 2 indicates that the proposed method performs the best performance on both of time consumption and RMSE when we set one mapper in Hadoop. Further, we significantly reduce the time consumption per frame from 12.24s to 3.40s by setting 29 mappers in Hadoop.

E. DISCUSSION

In order to compare cloud-based computation with cluster-based computation, we conducted the proposed sparse 3D facial shape retrieval algorithms using both Map/Reduce and Open MPI. Tested with the whole BU-4DFE dataset, Table 3 indicates that Open MPI spends less time than Map/Reduce does. However, referring to big data processing, time consumption is not the only criteria for evaluation. We opt to use the Map/Reduce framework due to the following reasons:

1) We can implement a similar Map step followed by a Reduce step in Open MPI. But it has a strict

synchronization rule [36] where mappers and reducers can only be called exclusively.

2) Map/Reduce had a better fault tolerance [37], [38]. The task will be terminated if a process fails in Open MPI, while failure can easily be recovered in Map/Reduce.
3) Without HDFS, we need to distribute and label the input data for Open MPI manually.
4) Map/Reduce program is scalable and is reusable.

VI. CONCLUSION AND FUTURE WORK

We have proposed sparse 3D face shape retrieval and 3D face reconstruction system accelerated by the Map/Reduce framework on Hadoop. The system reconstructs 3D facial shape from 2D videos using a three-step scheme, where the first two steps are a facial landmarking algorithm and a sparse 3D shape reconstruction algorithm. Then sparse 3D facial shapes have been applied as a prior estimation for 3D face reconstruction. To process big video data in a scalable manner, we have incorporated a Map/Reduce framework and implemented the algorithms as mapper and reducer appropriately. To overcome the limitation of revoking binary executables without Map/Reduce API supporting, we have reimplemented an appropriate input format and a record reader class in Java. Tested on the cloud platform consisting of 29 VMs, the system has retrieved 3D facial shapes which were consistent with the results obtained on a single machine. The scalability of the system has reduced the computation time by 92% and 73% respectively. The experimental results have also demonstrated the accuracy and scalability of the system. In future work, we will add more jobs following the current one to develop scalable facial expression recognition or face recognition using the reconstructed 3D face on big video data.

REFERENCES

[1] K. Smith. (Jul. 2019). *52 Fascinating and Incredible Youtube Statistics*. [Online]. Available: https://www.brandwatch.com/blog/youtube-stats

[2] C. Gu, L. Mou, Y. Tian, and T. Huang, "Mplboost-based mixture model for effective human detection with deformable part model," in *Proc. ICME*, Jul. 2013, pp. 1–6.

[3] C. Li, M. Liang, K. Xiao, S. Fong, Q. Wang, and W. Song, "Human body and face detection based anti-shoulder attack system on ATM," in *Proc. BDIOT*, Dec. 2017, pp. 145–148.

[4] N. Crosswhite, J. Byrne, C. Stauffer, O. Parkhi, Q. Cao, and A. Zisserman, "Template adaptation for face verification and identification," in *Proc. 12th IEEE Int. Conf. Autom. Face Gesture Recognit.*, May 2017, pp. 1–8.

[5] S. Poria, I. Chaturvedi, E. Cambria, and A. Hussain, "Convolutional MKL based multimodal emotion recognition and sentiment analysis," in *Proc. ICDM*, Dec. 2016, pp. 439–448.

[6] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," Univ. Massachusetts, Amherst, MA, USA, Tech. Rep. 07-49, Oct. 2007.

[7] J. L. Schonberger and J.-M. Frahm, "Structure-from-motion revisited," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 4104–4113.

[8] M. W. Tao, P. P. Srinivasan, S. Hadap, S. Rusinkiewicz, J. Malik, and R. Ramamoorthi, "Shape estimation from shading, defocus, and correspondence using light-field angular coherence," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 3, pp. 546–560, Mar. 2017.

[9] P. Huber, G. Hu, R. Tena, P. Mortazavian, P. Koppen, W. J. Christmas, M. Ratsch, and J. Kittler, "A multiresolution 3D morphable face model 695 and fitting framework," in *Proc. VISIGRAPP*, Feb. 2016, pp. 79–86.

[10] R. Dovgard and R. Basri, "Statistical symmetric shape from shading for 3D 697 structure recovery of faces," in *Proc. ECCV*, May 2004, pp. 99–113.

[11] W. A. P. Smith and E. R. Hancock, "Recovering facial shape using a statistical model of surface normal direction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 12, pp. 1914–1930, Dec. 2006.

[12] H. Rara, S. Elhabian, T. Starr, and A. Farag, "Model-based shape recovery from single images of general and unknown lighting," in *Proc. ICIP*, Nov. 2009, pp. 517–520.

[13] D. Jiang, Y. Hu, S. Yan, L. Zhang, H. Zhang, and W. Gao, "Efficient 3D reconstruction for face recognition," *Pattern Recognit.*, vol. 38, no. 6, pp. 787–798, 2005.

[14] I. Kemelmacher-Shlizerman and R. Basri, "3D face reconstruction from a single image using a single reference face shape," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 2, pp. 394–405, Feb. 2011.

[15] Z. Lei, Q. Bai, R. He, and S. Li, "Face shape recovery from a single image using CCA mapping between tensor spaces," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2008, pp. 1–7.

[16] M. D. Levine and Y. C. Yu, "State-of-the-art of 3D facial reconstruction methods for face recognition based on a single 2D training image per person," *Pattern Recogn. Lett.*, vol. 30, no. 10, pp. 908–913, Jul. 2009.

[17] V. Blanz, A. Mehl, T. Vetter, and H.-P. Seidel, "A statistical method for robust 3D surface reconstruction from sparse data," in *Proc. 3DPVT*, Thessaloniki, Greece, Sep. 2004, pp. 293–300.

[18] L. Zhang and D. Samaras, "Face recognition from a single training image under arbitrary unknown lighting using spherical harmonics," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 3, pp. 351–363, Mar. 2006.

[19] M. Castelan, W. A. P. Smith, and E. R. Hancock, "A coupled statistical model for face shape recovery from brightness images," *IEEE Trans. Image Process.*, vol. 16, no. 4, pp. 1139–1151, Apr. 2007.

[20] X. Ma, B. Zhu, T. Zhang, S. Cao, H. Jin, and D. Zou, "Efficient privacy-preserving motion detection for HEVC compressed video in cloud video surveillance," in *Proc. INFOCOM Workshops*, Apr. 2018, pp. 813–818.

[21] B. Wu, E. Zhong, B. Tan, A. Horner, and Q. Yang, "Crowdsourced time-sync video tagging using temporal and personalized topic modeling," in *Proc. KDD*, Aug. 2014, pp. 721–730.

[22] Z. Shaukat, J. Fang, F. Akhtar, M. Azeem, and S. Ali, "Cloud based face recognition for Google glass," in *Proc. ICCAI*, Mar. 2018, pp. 104–111.

[23] V. Kazemi and J. Sullivan, "One millisecond face alignment with an ensemble of regression trees," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Columbus, OH, USA, Jun. 2014, pp. 1867–1874.

[24] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, San Diego, CA, USA, Jun. 2005, pp. 886–893.

[25] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4311–4321, Oct. 2006.

[26] R. Tibshirani, "Regression shrinkage and selection via the LASSO," *J. Roy. Stat. Soc., Ser. B (Methodol.)*, vol. 58, no. 1, pp. 267–288, 1996.

[27] C. E. Thomaz and G. A. Giraldi, "A new ranking method for principal components analysis and its application to face image analysis," *Image Vis. Comput.*, vol. 28, no. 6, pp. 902–913, 2010.

[28] L. Yin, X. Chen, Y. Sun, T. Worm, and M. Reale, "A high-resolution 3D dynamic facial expression database," in *Proc. IEEE Int. Conf. Autom. Face Gesture Recognit.*, Sep. 2008, pp. 1–6.

[29] P. Dou, Y. Wu, S. K. Shah, and I. A. Kakadiaris, "Robust 3D face shape reconstruction from single images via two-fold coupled structure learning," in *Proc. Brit. Mach. Vis. Conf.*, Sep. 2014, pp. 1–7.

[30] D. Fabiano and S. Canavan, "Deformable synthesis model for emotion recognition," in *Proc. FG*, May 2019, pp. 1–5.

[31] L. A. Jeni, J. F. Cohn, and T. Kanade, "Dense 3D face alignment from 2D videos in real-time," in *Proc. FG*, May 2015, pp. 1–8.

[32] Y.-L. Tian, T. Kanade, and J. F. Cohn, "Recognizing action units for facial expression analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 2, pp. 97–115, Feb. 2001.

[33] C. Liu, J. Yuen, and A. Torralba, "SIFT flow: Dense correspondence across scenes and its applications," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 978–994, May 2011.

[34] R. Garg, A. Roussos, and L. Agapito, "A variational approach to video registration with subspace constraints," *Int. J. Comput. Vis.*, vol. 104, no. 3, pp. 286–314, 2013.

[35] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid, "Epicflow: Edge-preserving interpolation of correspondences for optical flow," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1164–1172.
[36] O. Vega-Gisbert, J. E. Roman, and J. M. Squyres, "Design and implementation of Java bindings in Open MPI," *Parallel Comput.*, vol. 59, pp. 1–20, Nov. 2016.
[37] W. Wang, K. Zhu, L. Ying, J. Tan, and L. Zhang, "Maptask scheduling in mapreduce with data locality: Throughput and heavy-traffic optimality," *IEEE/ACM Trans. Netw.*, vol. 24, no. 1, pp. 190–203, Feb. 2016.
[38] R. Sadikin, A. Arisal, R. Omar, and N. H. Mazni, "Processing next generation sequencing data in map-reduce framework using hadoop-BAM in a computer cluster," in *Proc. ICITISEE*, Nov. 2017, pp. 421–425.

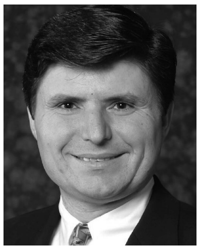
WANSHUN GAO received the B.E. degree in machine design and manufacturing and automation from Kunming University, and the M.E. degree in detection technology and automatic equipment from the Zhengzhou University of Light Industry. He is currently pursuing the Ph.D. degree with the School of Electrical and Information Engineering, Xi'an Jiaotong University, Xi'an, China.

XI ZHAO (S'07–M'11) received the Ph.D. degree (Hons.) in computer science from the Ecole Centrale de Lyon, Lyon, France, in 2010. After graduation, he conducted research in the fields of biometrics, face analysis, and pattern recognition as a Postdoctoral Fellow with the Computational Biomedicine Laboratory. He is currently a Research Assistant Professor with the Department of Computer Science, University of Houston, Houston, TX, USA. His current research interests include face analysis, biometrics, mobile computing, computer vision, and assistive technology. He recently served on the Program Committee of Workshop on 3D Face Biometrics, 2013 and Registration Chair of the Biometrics: Theory, Applications and Systems, 2013. He currently serves as a Reviewer for the IEEE Transactions on Systems, Man, and Cybernetics Part B, the IEEE Transactions on Infomation Forensics & Security, Image Vision Computing etc.

ZHIMIN GAO received the Ph.D. degree in computer science from the University of Houston, Houston, TX, USA, in 2017. After graduation, he joined a blockchain lab in Houston and held two international conferences in supply chain management on the blockchain. He is currently an Assistant Professor with Auburn University at Montgomery. He was a senior Investigator of several Research Projects from the Department of Homeland Security and North Atlantic Treaty Organization in cyber/mobile security and big data analytics. His research interests include blockchain, cloud computing, cybersecurity, the IoT, and 5G.

JIANHUA ZOU (M'10) received the Ph.D. degree from the Institute of Plasma Physics, Chinese Academy of Sciences, Beijing, China, in 1991. He conducted the Postdoctoral Research with the Huazhong University of Science and Technology, Wuhan, China, and Xi'an Jiaotong University, in 1991 to 1993 and 1993 to 1995, respectively. He became a Professor with the System Engineering Institute, Xi'an Jiaotong University. He is currently the Vice Dean with the School of Electronics and Information Engineering, Xi'an Jiaotong University, Xi'an, China. He was the Principle Investigator of two projects funded by the National Science Foundation of China, about ten key projects supported by industry and military. As a Senior Visiting Scholar, he has established partnerships with Eindhoven University, Eindhoven, The Netherlands, and Philips Company, Amsterdam, The Netherlands. He has published more than 60 academic articles. His current research interests include intelligent control, computer vision and pattern recognition, data mining, and knowledge discovery.

PENGFEI DOU received the B.E. degree in automation and the M.E. degree in traffic information engineering and control from Beijing Jiaotong University, Beijing, China. He is currently pursuing the Ph.D. degree with the Department of Computer Science, University of Houston, Houston, TX, USA. He is currently a Research Assistant with the Computational Biomedicine Laboratory. His current research interests include computer vision, machine learning, and biometrics.



IOANNIS A. KAKADIARIS (SM'89) received the B.Sc. degree in physics from the University of Athens, Athens, Greece, the M.Sc. degree in computer science from Northeastern University, Boston, MA, USA, and the Ph.D. degree from the University of Pennsylvania, Philadelphia, PA, USA. In August 1997, he joined UH, after a Postdoctoral Fellowship with the University of Pennsylvania. He is currently the Hugh Roy and Lillie Cranz Cullen University Professor of computer science, electrical and computer engineering, and biomedical engineering with the Department of Computer Science, University of Houston (UH), Houston, TX, USA. He is also the Founder of the Computational Biomedicine Laboratory, UH. His current research interests include multimodal biometrics, face recognition, computer vision, pattern recognition, biomedical image analysis, and predictive analytics. He was a recipient of a number of awards, including the NSF Early Career Development Award, the Schlumberger Technical Foundation Award, the UH Computer Science Research Excellence Award, the UH Enron Teaching Excellence Award, and the James Muller Vulnerable Plaque Young Investigator Prize. His research has been featured on the Discovery Channel, National Public Radio, KPRC NBC News, KTRH ABC News, and KHOU CBS News. His selected professional service leadership positions include: General Co-Chair of the 2013 Biometrics: Theory, Applications and Systems Conference (BTAS 2013), General Co-Chair of the 2013 and 2014 SPIE Biometric and Surveillance Technology for Human and Activity Identification, Program Co-Chair of the 2015 International Conference on Automatic Face and Gesture Recognition Conference, and Vice President for Technical Activities for the IEEE Biometrics Council.