An optimal transportation-based recognition algorithm for 3D facial expressions

Tiexiang Li*, Pei-Sheng Chuang, and Mei-Heng Yueh

Facial expression recognition (FER) is an active topic that has many applications. The development of effective algorithms for FER has been a competitive research field in the last two decades. In this paper, we propose a fully automatic 3D FER method based on the sparse approximation of 2D feature images. For a prescribed feature defined on the 3D facial surface, we apply a parameterization that not only maps the facial surface onto the unit disk but also locally preserves the feature. To ensure the uniqueness of the solution, some aligning constraints are further taken into account while computing the desired parameterization. The facial surface associated with the feature is then converted into the 2D image of the parameter domain. To recognize the expression of a test facial image, we apply an existing 2D expression recognition model, which is built upon sparse representation. Numerical experiments indicate that the accuracy of the proposed FER algorithm reaches 71.42% on a benchmark facial expression database, which is promising for practical applications.

AMS 2000 subject classifications: Primary 68U05, 52C26; secondary 65D18.

Keywords and phrases: Facial expression recognition, optimal mass transportation, projected gradient descent method.

1. Introduction

Facial expression is one of the most natural ways for human beings to express emotions and opinions. The automatic recognition of facial expressions aims to determine the emotional status of people using facial appearance information in an automated fashion. Automatic facial expression recognition (FER) technology can be applied to a variety of home and commercial applications, such as the analysis of consumer behavior [41], human-computer interaction [38, 22], or monitoring the student’s behavior in learning environments [17].

*Corresponding author.
Extensive FER research has led to an abundance of FER approaches. A comprehensive analysis of the existing FER methods can be found in the survey paper [25]. To better understand the pros and cons of each FER method, they categorize the existing methods according to the data type of the inputs and the scheme of the algorithms. The data types are classified into 2D and 3D formats. The 2D face data are the image of the face captured by the camera. 2D images are currently obtained easily through camera-equipped mobile phones or live streams on the Internet. Due to this fact, 2D FER is dominant in the application field. However, the quality of 2D face images is susceptible to illumination and head poses. On the other hand, the 3D face data are stored as a set of 3D point clouds or a polygonal mesh, which can be acquired through 3D scanners. The 3D face data contain the geometric information of the face, which are immune to illumination and head poses.

The algorithms for FER can be classified into three categories: model-based, deep learning-based and feature-based. The model-based algorithms rely on the generic face model, usually created by the neutral expression, to determine the emotion by measuring its shape deformation relative to the face model. The deep learning-based algorithms employ deep learning techniques to solve the expression recognition problems. The processes of the deep learning-based algorithms, which are analogous to many of the deep learning applications, learn and classify the features using the deep learning techniques. The feature-based algorithms, which is the scheme we follow in the paper, extract the features from the input data and feed them into a chosen classifier. In most of the 3D feature-based FER methods, the features consist of the differential quantities calculated regarding the regions enclosing the salient facial landmarks. For example, [21] and [20] consider the histograms of mesh gradient (meshHoG) and histogram of mesh shape index (meshHoS) at each facial landmark. The meshHoG and meshHoS methods express the surface gradient and curvature of the neighborhood of a facial landmark in the form of histograms and view the histograms as the feature vector of this landmark. The overall feature vector is then obtained by concatenating the feature vectors of each facial landmark. In addition, [49] segments the 3D facial surface based on the facial muscle anatomy model and treats the spatial coordinates, surface normal, and surface curvature as the feature on each muscle region. They create a support vector machine classifier for each muscle region and choose the best linear combination of all the classifiers defined on muscle regions as the overall decision function using the genetic algorithm. Zeng et al. [47] compute a conformal parameterization \( \phi_C : \mathcal{M} \rightarrow \mathbb{D} \) that maps the facial surface \( \mathcal{M} \) onto the unit disk...
\( \mathbb{D} \subset \mathbb{C} \) and use the pull-backed mean curvature \( H \circ \phi^{-1} \) and conformal factor \( \lambda \circ \phi^{-1} \) on the unit disk as the feature images. The feature images are subsequently fed to a sparse representation-based classification model to determine the emotion.

1.1. Contribution

In this paper, we propose an algorithm for the 3D FER problem based on optimal mass transportation mappings to automatically recognize the 6 prototypical emotions of human beings: anger, disgust, fear, happiness, sadness, and surprise as defined by Ekman and Friesen [9] in 1971. The contributions of this paper are threefold.

- We generalize the stretch energy minimization algorithm [46] for the computation of measure-preserving parameterizations of simply connected open surfaces.
- We modify the projected gradient method [44] for the computation of disk-shaped OMT maps of simply connected open surfaces.
- We combine the FER algorithm [47] with the OMT mappings and propose an algorithm for the 3D FER problem with significantly improved accuracy.

1.2. Notations and overview

In this paper, we use the following notations.

- Bold letters, e.g., \( \mathbf{f} \), denote vectors or matrices.
- Capital letters, e.g., \( \mathbf{L} \), denote matrices.
- Typewriter letters, e.g., \( \mathbf{I}, \mathbf{B} \), denote ordered sets of indices.
- \( n_V \) denotes the number of elements of the set \( V \).
- \( f_i \) denotes the \( i \)th entry/row of the vector/matrix \( f \).
- \( f_{i} \) denotes the subvector/submatrix of \( f \) composed of \( f_i \), for \( i \in \mathbf{I} \).
- \( |f| \) denotes the vector with entries being \( \|f_i\| \).
- \( L_{i,j} \) denotes the \( (i,j) \)th entry of the matrix \( L \).
- \( L_{i,j} \) denotes the submatrix of \( L \) composed of \( L_{i,j} \), for \( i \in \mathbf{I} \) and \( j \in \mathbf{J} \).
- \( \mathbb{D} := \{ \mathbf{x} \in \mathbb{R}^2 | \|\mathbf{x}\| < 1 \} \) denotes the unit disk in \( \mathbb{C} \).
- \( [v_0, \ldots, v_m] \) denotes the \( m \)-simplex of the vertices \( v_0, \ldots, v_m \).
- \( |[v_0, \ldots, v_m]| \) denotes the volume of the \( m \)-simplex \( [v_0, \ldots, v_m] \).
- \( \mathbf{0} \) denotes the zero vectors and matrices of appropriate sizes.
The remainder of this paper is organized as follows. The discrete differential geometry, surface parameterization and optimal mass transportation play crucial roles in the proposed FER algorithm. They are introduced in Sections 2, 3 and 4, respectively. The proposed 3D FER algorithm and related numerical experiments are presented in Sections 5 and 6, respectively. Concluding remarks are given in Section 7.

2. Discrete differential geometry

In this section, we introduce the discrete surfaces and mappings, discrete differential operators, and discrete curvatures in Subsections 2.1, 2.2 and 2.3, respectively.

2.1. Discrete surfaces and mappings

In this paper, discrete surfaces refer to triangular meshes composed of flat triangular faces. A triangular mesh \(M\) is the underlying space of a simplicial 2-complex composed of vertices \(\mathcal{V}(M) = \{v_i \,|\, 1 \leq i \leq n_v\} \subset \mathbb{R}^3\), edges \(\mathcal{E}(M) = \{[v_i, v_j] \mid \text{if there is an edge connecting } v_i, v_j\}\) and triangular faces \(\mathcal{F}(M) = \{[v_i, v_j, v_k] \mid \text{if } v_i, v_j, v_k \text{ form a face}\}\). Figure 1 illustrates an example of a triangular mesh.

![Figure 1: Triangular mesh of Nefertiti.](image)

On the other hand, the discrete mapping \(f : M \to \mathbb{R}^d\) refers to a simplicial mapping defined by the values on the vertices. Suppose the value at vertex \(v_i \in \mathcal{V}(M)\) is \(f_i := f(v_i) \in \mathbb{R}^d\) and \(x \in M\) is a point lying in triangle \([v_i, v_j, v_k] \in \mathcal{F}(M)\). The value \(f(x)\) is determined by the barycentric
coordinates as
\[
(1) \quad f(x) = f[v_i, v_j, v_k](x) = \frac{[x, v_j, v_k]}{[v_i, v_j, v_k]} f_i + \frac{[v_i, x, v_k]}{[v_i, v_j, v_k]} f_j + \frac{[v_i, v_j, x]}{[v_i, v_j, v_k]} f_k.
\]

Figure 2 shows the graph of a simplicial function. The formula (1) indicates that the simplicial function \( f \) can be concisely represented by its values at each vertex; namely, \( f \) is represented by the matrix
\[
\begin{pmatrix}
  f_1 \\
  \vdots \\
  f_n
\end{pmatrix} \in \mathbb{R}^{n_V \times d},
\]
where \( n_V \) is the number of vertices. The bold notation \( \mathbf{f} \) is hereafter called the inducing matrix of the map \( f \). In particular, a surface parameterization is a diffeomorphism that maps the surface to a domain of simple shape in \( \mathbb{R}^2 \).

### 2.2. Discrete differential operators

In this section, we introduce the explicit formulas for the gradient, divergence, curl, and Laplacian in Subsections 2.2.1, 2.2.2, and 2.2.3, respectively. The derivations for these formulas can be found in [39] and [11].

#### 2.2.1. Gradient.

Suppose that \( f : \mathcal{M} \rightarrow \mathbb{R} \) is a real-valued simplicial function and \( x \in \tau = [v_i, v_j, v_k] \subset \mathcal{M} \). The gradient of \( f \) at \( x \) is
\[
(2) \quad \nabla f(x) = \frac{1}{2|\tau|} \left[ (\mathbf{N}_\tau \times (v_k - v_j)) f_i + (\mathbf{N}_\tau \times (v_i - v_k)) f_j + (\mathbf{N}_\tau \times (v_j - v_i)) f_k \right],
\]
where \( \mathbf{N}_\tau := (v_j - v_i) \times (v_k - v_i)/\| (v_j - v_i) \times (v_k - v_i) \| \) is the normal vector of \( \tau \). Note that the discrete gradient \( \nabla f \) is a face-based vector field.

#### 2.2.2. Divergence and curl.

Suppose that \( u : \mathcal{M} \rightarrow \mathbb{R}^3 \) is a piecewise constant vector field with \( u_\tau \) being the vector assigned to the triangular face \( \tau \in F(M) \). The discrete divergence and curl with respect to \( u \) are defined as
\[
(3) \quad \text{Div}_\mathcal{M} u(v_i) = -\frac{1}{2} \sum_{\tau \in N^*_\mathcal{M}(v_i)} \langle u_\tau, \mathbf{N}_\tau \times (v_k - v_j) \rangle
\]
and
\[
\text{Curl}_\mathcal{M} u(v_i) = \frac{1}{2} \sum_{\tau \in N^*_\mathcal{M}(v_i)} \langle u_\tau, (v_k - v_j) \rangle,
\]
respectively, where \( N(F_i) = \{ \tau \in F(M) \mid \tau = [v_i, v_j, v_k] \} \) is the neighboring faces of \( v_i \).

These definitions reflect that divergence and curl measure the flux and circulation of a tangent vector field around small loops, respectively. See Figure 3 for an illustration.

\[ L_M f(v_i) = \sum_{j \in N_i(v_i)} \left[ \frac{1}{2} \sum_{[v_i, v_j, v_k] \in F(M)} \cot \alpha_{ij}^k \right] (f_i - f_j), \]

where \( \alpha_{ij}^k \) is the angle opposite to edge \([v_i, v_j]\) in triangle \([v_i, v_j, v_k]\), as shown in Figure 4.

2.2.3. The Laplace–Beltrami operator. Suppose that \( f : M \to \mathbb{R} \) is a real-valued simplicial function. The discrete Laplace–Beltrami operator defined by the composition of discrete negative divergence (3) and discrete gradient (2) is

\[ \mathcal{L}_M := -\text{Div}_M \circ \nabla. \]

The negative sign is meant to make the discrete Laplace operator become positive semidefinite. The value \( \mathcal{L}_M f \) can be explicitly formulated as

Figure 3: (a) Discrete divergence; (b) discrete curl.
Figure 4: Angles opposite to the edge \([v_i, v_j]\).

Note that \(\mathcal{L}_f\) is a simplicial function with the inducing matrix being

\[
\begin{pmatrix}
\mathcal{L}_f(v_1), & \cdots, & \mathcal{L}_f(v_n)
\end{pmatrix}^T = Lf,
\]

where \(L\) is the Laplacian matrix

\[
L_{ij} = \begin{cases}
-w_{ij} & \text{if } [v_i, v_j] \in \mathcal{E}(\mathcal{M}), \\
\sum_{k \neq i} w_{ik} & \text{if } i = j, \\
0 & \text{otherwise},
\end{cases}
\]

with

\[
w_{ij} = \frac{1}{2} \sum_{[v_i, v_j, v_k] \in \mathcal{F}(\mathcal{M})} \cot \alpha_{ij}^k.
\]

2.3. Discrete curvatures

In this section, we introduce the formulas for the normal vector, mean curvature, and Gaussian curvature at vertex \(v \in \mathcal{V}(\mathcal{M})\). The formula for the normal vector can be found in the course note \([5]\). The derivations of the discrete mean curvature and the Gaussian curvature can be found in paper \([24]\).

2.3.1. Normal vector. The volume-gradient-based normal vector at a vertex \(v\) is defined as

\[
\mathbf{N}_v = \frac{\left( \sum_{\tau \in \mathcal{F}(v_i)} |\tau| \cdot \mathbf{N}_\tau \right)}{\left\| \sum_{\tau \in \mathcal{F}(v_i)} |\tau| \cdot \mathbf{N}_\tau \right\|_2},
\]

i.e., the area-weighted average of normal vectors of its neighboring faces. It is the direction along which the volume enclosed by the surface increases the fastest.
2.3.2. Gaussian curvature. The derivation of discrete Gaussian curvature is inspired by the Gauss–Bonnet theorem:

\[ \iint_{\mathcal{U}} K \, dA = 2\pi - \sum_{j} \epsilon_j, \]

where \( \epsilon_j \) are the exterior angles of the boundary \( \partial \mathcal{U}. \) To obtain the integral around a vertex \( v \in \mathcal{E}(\mathcal{M}) \), we assign a local neighborhood to the vertex \( v \), which is referred to as the surface patch for \( v \). It is defined as follows.

Within each neighboring face \( \tau \in N_{\mathcal{F}}(v) \) of \( v \), pick a point \( u_{\tau} \) that lies in the interior of \( \tau \). The associated surface patch for the vertex \( v \), denoted by \( \mathcal{U}_v \), is the region surrounded by the polygon passing through \( \{u_{\tau}\}_{\tau \in N_{\mathcal{F}}(v)} \) and the midpoints of the edges emanating from the vertex \( v \). An illustration of the surface patch is shown in Figure 5. The surface patch \( \mathcal{U}_v \) is enclosed within the dashed line. The red dots denote the points within the interior of faces, while the blue dots are the midpoints of the edges emanating from the vertex \( v \).

![Figure 5: Surface patch for the vertex \( v \).](image)

In particular, the surface patch \( \mathcal{U}_v \) is called the barycentric cell, denoted as \( \mathcal{U}_{\text{barycenter}} \), if the points \( u_{\tau} \) are the barycenters of the neighboring triangles. Similarly, the surface patch is called the Voronoi cell, denoted as \( \mathcal{U}_{\text{Voronoi}} \), if the points \( u_{\tau} \) are the circumcenters of the neighboring triangles.

The discrete Gaussian curvature at vertex \( v \) is defined by taking the spatial average of the integral of the Gaussian curvature on the surface patch \( \mathcal{U}_v \) as

\[ K_v = \frac{1}{|\mathcal{U}_v|} \iint_{\mathcal{U}_v} K \, dA = \frac{1}{|\mathcal{U}_v|} \left( 2\pi - \sum_{j} \epsilon_j \right). \]
Calculating the external angles $\epsilon_j$ requires additional computations on the boundary $\partial U_v$. However, the definition of the surface patch leads to the equation
\[
\sum_j \epsilon_j = \sum_{\tau \in N^F_1(v)} \theta_\tau,
\]
where $\theta_\tau$ is the angle of triangle $\tau \in N^F_1(v)$ at vertex $v$, as shown in Figure 5. The proof of this proposition in the case of Voronoi cells can be found in [24]. Consequently, the discrete Gaussian curvature can be formulated as
\[
K_v = \frac{1}{|U_v|} \left( 2\pi - \sum_{\tau \in N^F_1(v)} \theta_\tau \right).
\]

2.3.3. Mean curvature. For a parameterized smooth closed surface $M_0$, the mean curvature is related to the normal vector, and their relationship is given by
\[
\Delta M_0 x = 2HN,
\]
where $\Delta M_0 := -\nabla \cdot \nabla$ is the Laplace–Beltrami operator defined on $M_0$ and $x : M_0 \to \mathbb{R}^3$ is the coordinate function. The vector $H := HN$ is called the mean curvature normal. The discrete mean curvature normal at vertex $v$ is defined by the spatial average of the integral of $H$ on the surface patch $U_v$ as
\[
H_v = \frac{1}{|U_v|} \iint_{U_v} H \ dA = \frac{1}{2|U_v|} \iint_{U_v} \Delta M x \ dA.
\]
Suppose that $v = v_i$. Appendix A in [24] proves that
\[
\iint_{U_v} \Delta M x \ dA = L_M x(v_i) = [LV]_i,
\]
where $L_M$ is the discrete Laplace operator (4), $L$ is the Laplacian matrix defined in (6) and (7) in Subsection 2.2.3, $V = (v_1, \ldots, v_n)^\top$ is the matrix containing all vertices, and $[LV]_i$ is the i-th row of $LV$. As a result, the discrete mean curvature vector $H_{v_i}$ can be formulated as
\[
H_{v_i} = \frac{1}{2|U_{v_i}|} [LV]_i.
\]
Finally, the mean curvature can be computed by
\[
H_v = -\langle H_v, N_v \rangle.
\]
The negative sign is derived from these two observations

1. $H_v$ points towards the direction in which the surface area increases the fastest [6].

2. $H_v < 0$ if $\mathcal{M}$ is convex at $v$, and $H_v > 0$ if $\mathcal{M}$ is concave at $v$.

3. Surface parameterizations

In this section, we introduce the computation of conformal and measure-preserving parameterizations of simply connected open surfaces in Subsections 3.1 and 3.2, respectively.

3.1. Conformal parameterizations

A conformal map between two parameterized Riemannian surfaces $\mathcal{M}_1$ and $\mathcal{M}_2$ is a diffeomorphism that satisfies $\lambda^2 I_1 = f^* I_2$, where $I_1$ and $I_2$ are the first fundamental forms of $\mathcal{M}_1$ and $\mathcal{M}_2$, respectively, $\lambda$ is the conformal factor for the map $f$, and $f^* I_2$ is the pullback metric of $I_2$ by $f$ [36]. The existence of conformal parameterization $f : \mathcal{M} \rightarrow \mathbb{D}$ is guaranteed by the uniformization theorem [12].

Conformal energy minimization (CEM) [45, 18] is an efficient approach for the computation of disk-shaped conformal parameterizations of simply connected open surfaces. The conformal energy of a smooth map $f : \mathcal{M} \rightarrow \mathbb{D}$ is defined by

\[
E_C(f) = E_D(f) - A(f),
\]

where $A(f)$ is the area of the image $f(\mathcal{M})$, and $E_D(f)$ is the Dirichlet energy of the function $f$

\[
E_D(f) = \frac{1}{2} \int_{\mathcal{M}} \|\nabla f\|^2 \, dvol_{\mathcal{M}}.
\]

The connection between conformal energy (8) and conformal map was established by Hutchinson in 1991 [13], which states that a function $f$ is conformal if and only if it is a minimizer of the conformal energy (8). The assumption that the image $f(\mathcal{M})$ is the unit disk $\mathbb{D}$ implies that the second term of conformal energy (8) is constant, i.e., $A(f) = \pi$. As a result, minimizing (8) is equivalent to minimizing the Dirichlet energy (9).
Under the assumptions that the function \( f \) is piecewise linear and the surface \( M \) is a triangular mesh, Pinkall and Polthier [29] in 1993 derived the discrete formula for the Dirichlet energy

\[
E_D(f) = \frac{1}{2} \text{trace} \left( f^\top L f \right),
\]

where \( L \) is the Laplacian matrix defined in (6) and (7).

The CEM algorithm proposed by Yueh et al. [45] can be used to efficiently compute a conformal parameterization \( f : M \rightarrow \mathbb{D} \) by minimizing the Dirichlet energy (9). It is straightforward from calculus that the derivative of the discrete Dirichlet energy \( E_D(f) \) vanishes at all the minimizers of \( E_D(f) \), i.e., \( \nabla E_D(f) = L f = 0 \), if \( f \) is a conformal map. Suppose that \( I \) and \( B \) are the ordered index sets of the interior and boundary vertices, respectively. Under a certain reordering of vertices, equation \( L f = 0 \) can be reformulated as

\[
\begin{align*}
L_{II} f_I &= -L_{IB} f_B, \\
L_{BB} f_B &= -L_{BI} f_I.
\end{align*}
\]

The CEM algorithm updates the interior and boundary part of \( f \), i.e., \( f_I \) and \( f_B \), alternatively. The computation procedure of the CEM in detail is summarized in Algorithm 1.

**Algorithm 1 Conformal Energy Minimization**

**Input:** A simply-connected open surface \( M \)

**Output:** The inducing matrix \( f \) for a conformal parametrization \( f : M \rightarrow \mathbb{D} \)

1: procedure CEM(\( M \))
2: \( I \leftarrow \{ \text{indices of interior vertices} \} \) and \( B \leftarrow \{ \text{indices of boundary vertices} \} \)
3: Let \( f \) be the disk harmonic map given in [45].
4: \( f_B \leftarrow \text{modifyBoundary}(f_B) \)
5: \( f_I \leftarrow -L_{II}^{-1} L_{IB} f_B \)
6: while not convergent do
7: \( f_B \leftarrow -L_{BB}^{-1} L_{BI} f_I \) \( \triangleright \) Update the boundary part
8: \( f_B \leftarrow \text{modifyBoundary}(f_B) \)
9: \( f_I \leftarrow -L_{II}^{-1} L_{IB} f_B \) \( \triangleright \) Update the interior part
10: end while
11: return \( f \)
12: end procedure
13: procedure modifyBoundary(\( f_B \))
14: \( f_B \leftarrow (I_n - \frac{1}{n_B} 1_n 1_n^\top)f_B \) \( \triangleright \) Centralize the mass center to origin
15: \( f_B \leftarrow \text{diag}(|f_B|)^{-1} f_B \) \( \triangleright \) Normalize the length to 1
16: return \( f_B \)
17: end procedure
Remark 3.1. For point clouds as input, algorithms such as [23] can be applied to compute the required conformal parameterization.

3.2. Measure-preserving parameterization

Suppose $\mathcal{M}_1$ and $\mathcal{M}_2$ are 2D Riemannian surfaces equipped with measures $\mu$ and $\nu$, respectively. A measurable map $f : (\mathcal{M}_1, \mu) \to (\mathcal{M}_2, \nu)$ is said to be measure-preserving if $f_*\mu = \nu$, where $f_*\mu$ is the pushforward of measure $\mu$ through $f$ defined as

$$f_*\mu(B) = \mu(f^{-1}(B)), \forall B \subset \mathcal{M}_2 \text{ is measurable}.$$

In particular, for a triangular mesh $\mathcal{M}$ and a bijective, simplicial function $f$, the condition (11) for measure preservation is

$$\nu(f(\tau)) = \mu(\tau), \forall \tau \in \mathcal{F}(\mathcal{M}).$$

When the measures $\mu$ and $\nu$ are chosen to be the area measure, denoted as $\lambda(A) := |A|$, any measure-preserving map $f : (\mathcal{M}_1, \lambda) \to (\mathcal{M}_2, \lambda)$ is equiareal. This can be observed from the condition (11)

$$\lambda(B) = f_*\lambda(B) = \lambda(f^{-1}(B)) \iff |B| = |f^{-1}(B)|,$$

for all measurable subset $B \subset \mathcal{M}_2$.

Unlike the uniformization theorem that guarantees the existence of conformal parameterization $f : \mathcal{M} \to \mathbb{D}$, there is no theorem stating that measure-preserving parameterization $f : (\mathcal{M}, \mu) \to (\mathbb{D}, \nu)$ does exist for arbitrary measures $\mu$ and $\nu$. Nevertheless, the measure-weighted stretch energy minimization (MSEM) algorithm described in the following section manages to approximate a measure-preserving parameterization $f : (\mathcal{M}, \mu) \to (\mathbb{D}, |\cdot|)$ for any given positive measure $\mu$ by minimizing a modified stretch energy functional.

3.2.1. Measure-weighted stretch energy minimization (MSEM) algorithm. We now generalize the stretch energy minimization (SEM) procedure of [46] and propose the MSEM procedure for the computation of measure-preserving parameterization $f : (\mathcal{M}, \mu) \to (\mathbb{D}, |\cdot|)$ from $(\mathcal{M}, \mu)$ onto a unit disk equipped with an area measure.
Let $\mu : \mathcal{F}(\mathcal{M}) \rightarrow \mathbb{R}^+$ be an area measure for $\mathcal{M}$. Similar to the stretch energy in [46], we define the measure-weighted stretch energy as

\begin{equation}
E_S(f) = \frac{1}{2} \text{trace} \left( f^\top L(f) f \right),
\end{equation}

where $L(f)$ is the Laplacian matrix

\begin{equation}
[L(f)]_{ij} = \begin{cases} 
-w_{ij}(f) & \text{if } [v_i, v_j] \in \mathcal{E}(\mathcal{M}), \\
\sum_{k \neq i} w_{ik}(f) & \text{if } i = j, \\
0 & \text{otherwise}
\end{cases}
\end{equation}

with

\begin{equation}
w_{ij}(f) = \frac{1}{2} \sum\limits_{[v_i, v_j, v_k] \in \mathcal{F}(\mathcal{M})} \frac{\cot \alpha_{ij}^k(f)}{\sigma_{f^{-1}}([v_i, v_j, v_k])},
\end{equation}

The argument $\alpha_{ij}^k(f)$ is the angle opposite to the edge $[f_i, f_j]$ in the triangle $[f_i, f_j, f_k]$, as illustrated in Figure 6, and

\begin{equation}
\sigma_{f^{-1}}([v_i, v_j, v_k]) := \frac{\mu([v_i, v_j, v_k])}{\| [f_i, f_j, f_k] \|}
\end{equation}

is called the stretch factor.

![Figure 6: Angles opposite to the edge $[f_i, f_j]$.](image)

The following proposition gives an equivalent expression for the weight function $w_{ij}(f)$ in (13).

**Proposition 3.1.** The weight function $w_{ij}(f)$ in (13) can be written as

\begin{equation}
w_{ij}(f) = \frac{1}{2} \sum\limits_{[v_i, v_j, v_k] \in \mathcal{F}(\mathcal{M})} \frac{\| f_i - f_k \|}{\mu([v_i, v_j, v_k])} \cot \alpha_{ij}^k(f) \left( \frac{f_j - f_k}{\| f_j - f_k \|} \right) \left( \frac{f_i - f_k}{\| f_i - f_k \|} \right).
\end{equation}
Proof. A direct calculation yields that
\[ w_{ij}(f) = \frac{1}{2} \sum_{[v_i, v_j, v_k] \in F(M)} \cot \alpha_{ij}^k(f) \sigma_{f^{-1}}([v_i, v_j, v_k]) \]
\[ = \frac{1}{2} \sum_{[v_i, v_j, v_k] \in F(M)} \sin \alpha_{ij}^k(f) \sigma_{f^{-1}}([v_i, v_j, v_k]) \]
\[ = \frac{1}{2} \sum_{[v_i, v_j, v_k] \in F(M)} \frac{\|f_i - f_k\| \|f_j - f_k\| \cos \alpha_{ij}^k(f)}{\|f_i - f_k\| \|f_j - f_k\| \sin \alpha_{ij}^k(f) \sigma_{f^{-1}}([v_i, v_j, v_k])} \]
\[ = \frac{1}{2} \sum_{[v_i, v_j, v_k] \in F(M)} \frac{(f_i - f_k)^\top (f_j - f_k)}{2 \mu([v_i, v_j, v_k])}. \] (By formula (14))

The iterative MSEM procedure is identical to that of the SEM algorithm [46] except for the new definitions of the stretch factor \( \sigma_{f^{-1}} \).

The MSEM algorithm is summarized in Algorithm 2.

**Algorithm 2 Measure-Weighted Stretch Energy Minimization**

**Input:** A simply-connected open surface \( M \), a face-based measure \( \mu \) for \( M \) such that \( \mu |_{F(M)} > 0 \)

**Output:** The inducing matrix \( f \) for a measure-preserving parametrization \( f : (M, \mu) \to (D, | \cdot |) \)

1: \hspace{1cm} procedure MSEM(\( M, \mu \))
2: \hspace{1cm} I \leftarrow \{ \text{indices of interior vertices} \} \quad \text{and} \quad B \leftarrow \{ \text{indices of boundary vertices} \}
3: \hspace{1cm} Let \( f \) be the disk harmonic map given in [46].
4: \hspace{1cm} L \leftarrow L(f)
5: \hspace{1cm} \text{while not convergent do}
6: \hspace{2cm} f_B \leftarrow -L_B^{-1}L_I f_I \hspace{1.5cm} \triangleright \text{Update the boundary part}
7: \hspace{2cm} f_B \leftarrow \text{MODIFYBOUNDARY}(f_B)
8: \hspace{2cm} f_I \leftarrow -L_I^{-1}L_B f_B \hspace{1.5cm} \triangleright \text{Update the interior part}
9: \hspace{2cm} L \leftarrow L(f)
10: \hspace{1cm} \text{end while}
11: \hspace{1cm} \text{return } f
12: \hspace{1cm} \text{end procedure}

13: \hspace{1cm} procedure MODIFYBOUNDARY(\( f_B \))
14: \hspace{2cm} f_B \leftarrow (I_n - \frac{1}{n}1_n 1_n^\top) f_B \hspace{1.5cm} \triangleright \text{Centralize the mass center to origin}
15: \hspace{2cm} f_B \leftarrow \text{diag}(\|f_B\|^{-1}) f_B \hspace{1.5cm} \triangleright \text{Normalize the length to 1}
16: \hspace{1cm} \text{return } f_B
17: \hspace{1cm} \text{end procedure}
4. Optimal mass transportation

The optimal mass transportation (OMT) problem was initially raised by Monge in 1781: How can a pile of dirt be moved from one place to another while minimizing the total effort? More explicitly, Monge’s OMT problem can be formulated as follows:

**OMT Problem.** Let \((X, \mu), (Y, \nu)\) be two measure spaces equipped with probability measures \(\mu\) and \(\nu\), which have the same total mass \(\int_X d\mu(x) = \int_Y d\nu(y)\). Let

\[ MP = \{ f : (X, \mu) \rightarrow (Y, \nu) \mid \nu = f \# \mu \} \]

denote the space of measure-preserving maps, and let \(c : X \times Y \rightarrow \mathbb{R}^+\) be the transportation cost. Find an optimal transport map \(f^* \in MP\) that minimizes the overall transportation cost, as illustrated in Figure 7. Namely,

\[
(15) \quad f^* = \arg \min_{f \in MP} \int_X c(x, f(x)) \, d\mu.
\]

Figure 7: A transport map.

It was not until 1979 that the existence of a solution to the OMT problem with respect to \(l_1\) cost was first shown by Sudakov [33]. The existence for the case of \(l_2\) cost was proved by Brenier [3] later in 1991. Five years later, Gangbo and McCann [10] proved the existence for any strictly convex cost.

The concept of the OMT can be applied to various problems that involve comparison, evolution, or generation of probability distributions. Specifically, problems such as the generative adversarial network [1], transfer learning in machine learning [40], 3D data interpolation [32], nonrigid point cloud registration [19], and image texture blending [30] can be addressed from the perspective of optimal transportation. Please see [16] and [28] for a comprehensive introduction to the OMT problem and its applications.
4.1. Discrete OMT

Now, we consider the scenario in which $X$ is a triangular mesh $M$, $Y$ is the unit disk $D \subset \mathbb{R}^3$, $\nu$ is the area measure $|\cdot|$, and the cost $c(x, y) = \|x - y\|_2^2$. We define the local measure at the vertex $v \in \mathcal{V}(M)$ as

$$m_\mu(v) = \frac{1}{3} \sum_{\tau \in N^F_1(v)} \mu(\tau).$$

The overall transportation cost defined in (15) then becomes

$$C(f) := \int_M \|x - f(x)\|_2^2 \, d\mu = \sum_{i=1}^{n_V} \|v_i - f(v_i)\|_2^2 m_\mu(v_i),$$

where $f$ is the inducing matrix for the piecewise linear map $f : M \to D$. Let

$$\mathbb{F}_\mu = \left\{ f \in \mathbb{R}^{n_V \times 3} \mid f \text{ is the inducing matrix for a measure-preserving map } f : (M, \mu) \to (D, |\cdot|) \right\}$$

denote the space of all inducing matrices for the measure-preserving map $f : (M, \mu) \to (D, |\cdot|)$. Since the image $f(M)$ lies on the $xy$ plane, the third column of $f \in \mathbb{F}_\mu$ is zero.

**Discrete OMT Problem.** The discrete OMT problem with respect to the $l_2$ cost is to find $f^* \in \mathbb{F}_\mu$ that minimizes the total transportation cost $C(f)$:

$$f^* = \arg \min_{f \in \mathbb{F}_\mu} C(f) = \arg \min_{f \in \mathbb{F}_\mu} \sum_{i=1}^{n_V} \|v_i - f(v_i)\|_2^2 m_\mu(v_i).$$

As stated in [44], the discrete OMT problem (18) has a minimizer, but it may not be unique.

4.2. Projected gradient descent method for discrete OMT with backtracking line search

Similar to the OMT algorithm in [44], we find a minimal solution $f^* \in \mathbb{F}_\mu$ to the discrete OMT problem (18) by the projected gradient descent method. Compared to the gradient descent method, the projected gradient descent method first updates the solution along the negative gradient via

$$f^{t+0.5} = f^t - \eta^t \nabla C(f^t)$$
and then projects the intermediate result $f^{t+0.5}$ back to the measure-preserving space $\mathbb{F}_\mu$ as

$$(20) \quad f^{t+1} = P_{\mathbb{F}_\mu}(f^{t+0.5}),$$

where $P_{\mathbb{F}_\mu} : \mathbb{R}^{n_V \times 3} \to \mathbb{F}_\mu$ is a projection map, which will be explained later. The processes of the projected gradient descent are illustrated in Figure 8.

![Figure 8: Processes of the projected gradient descent.](image)

In practice, we let $V = (v_1, \ldots, v_{n_V})^\top$ and $M_\mu = \text{diag}(m_\mu(v_1), \ldots, m_\mu(v_{n_V}))$. The transportation cost $C(f)$ in (17) can be reformulated as

$$C(f) = \sum_{i=1}^{n_V} \left\| v_i - f_i \right\|_2^2 m_\mu(v_i)$$

$$= \sum_{i=1}^{n_V} m_\mu(v_i) (v_i - f_i)^\top (v_i - f_i)$$

$$= \text{trace} \left[ M_\mu (f - V)(f - V)^\top \right].$$

Then, the gradient in (19) can be formulated as $\nabla C(f) = 2M_\mu (f - V)$.

Suppose that $f : M \to \mathbb{R}^3$ maps $M$ to another triangular mesh $f(M)$. The projection map $P_{\mathbb{F}_\mu} : \mathbb{R}^{n_V \times 3} \to \mathbb{F}_\mu$ in (20) consists of two steps:

1. Execute the MSEM Algorithm 2 with input mesh $f(M)$ and prescribed measure $\mu$. Save the output as $\hat{f}$, the inducing matrix for $\hat{f}$.
2. Rotate the point set $\{\hat{f}_1, \ldots, \hat{f}_{n_V}\}$ with respect to the $z$-axis to make it aligned with the point set $\{v_1, \ldots, v_{n_V}\}$ in the least square sense as

$$R^*(\hat{f}) = \arg \min_{R \in \text{SO}(2)} \sum_{i=1}^{n_V} \left\| \begin{pmatrix} R & 0 \\ 0 & 1 \end{pmatrix} \hat{f}_i - v_i \right\|_2^2,$$

which can be solved by using the Kabsch algorithm [15].
Recall that the inducing matrix for $\hat{f}$ is written as $\hat{f} = (\hat{f}_1, \ldots, \hat{f}_n)$, where $\hat{f}_i := \hat{f}(v_i)$. Consequently, the projection map $P_{\mathbb{F}_\mu} : \mathbb{R}^{n_V \times 3} \rightarrow \mathbb{F}_\mu$ is defined by

$$P_{\mathbb{F}_\mu}(f) := \hat{f} \left( \begin{array}{cc} R^*(\hat{f}) & 0 \\ 0 & 1 \end{array} \right)^\top,$$

as illustrated in Figure 9.

![Figure 9: Projection onto the measure-preserving space.](image)

On the other hand, the learning rate $\eta_t$ in (19) at the $t$-th iteration is determined by using the backtracking line search method, which is summarized in Algorithm 3.

The Frobenius norm $\| \cdot \|_F$ in line 5 of Algorithm 3 treats the object $G_{\eta^t}(f^t)$ as a matrix. However, the object $G_{\eta^t}(f^t)$ is intrinsically the inducing matrix for a function, indicating that we should measure it with a norm on function space. In practice, we replace the Frobenius norm $\| \cdot \|_F$ with $L_2(\mathcal{M}, \mu)$ norm

$$\|f\|_{L_2(\mathcal{M}, \mu)}^2 = \int_{\mathcal{M}} \|f(x)\|_2^2 \, d\mu = \sum_{i=1}^{n_V} \|f(v_i)\|_2^2 m_\mu(v_i).$$
Algorithm 3 Backtracking Line Search for Projected Gradient Descent

**Input:** Current solution $f^t$, initial guess of learning rate $\eta_0^t$, step scaling factor $\beta \in (0, 1)$

**Output:** Learning rate $\eta^t$

1: procedure BACKTRACKINGLINESearch($f^t, \eta_0^t, \beta$)
2: $f^{t+1} := P_{\mu}(f^t - \eta^t \nabla C(f^t))$.
3: $G_{\eta}(f^t) := \frac{\eta^t - \eta^{t+1}}{\eta^t}$.
4: $\eta^t \leftarrow \eta_0^t$
5: while $C(f^{t+1}) > C(f^t) - \eta^t \| G_{\eta}(f^t) \|^2_F$ do
6: $\eta^t \leftarrow \beta \cdot \eta^t$
7: end while
8: return $\eta^t$
9: end procedure

Algorithm 4 Disk-shaped OMT algorithm

**Input:** A simply-connected open surface $M$, a face-based measure $\mu$ for $M$, step scaling factor $\beta \in (0, 1)$

**Output:** Optimal transport map $f$

1: procedure DISKOMT($M, \mu, \beta$)
2: $f \leftarrow P_{\mu}(\text{id}_M)$, where $\text{id}_M : M \rightarrow \mathbb{R}^3$ is the identity map. \(\triangleright \) Formula (21)
3: $\eta_0 \leftarrow (2\text{mean}_{v \in V(M)} m_\mu(v))^{-1}$
4: while not convergent do
5: $\eta \leftarrow $ BACKTRACKINGLINESearch($f, \eta_0, \beta$) \(\triangleright \) Algorithm 3
6: $f \leftarrow P_{\mu}(f - \eta \nabla C(f))$ \(\triangleright \) Formula (21)
7: end while
8: return $f$
9: end procedure

The initial guess $\eta_0^t$ is chosen as $\eta_0^t = (2\text{mean}_{v \in V(M)} m_\mu(v))^{-1}$. The full analysis of the backtracking line search method for the projected gradient method can be found in [2, Chap. 10].

The algorithm for the projected gradient descent method with backtracking line search is summarized in Algorithm 4.

5. OMT-based 3D facial expression recognition method

In this section, we propose the OMT-based 3D FER (OMT-FER) algorithm. The core idea is that we regard each facial surface $M$ as a raw feature space and the function $Q : V(M) \rightarrow \mathbb{R}^+$ as a raw feature defined on $M$. To better
compare different facial surfaces equipped with features, we transform the raw feature space $\mathcal{M}$ into a canonical unit disk $\mathbb{D}$. While doing so, the transformation is required to preserve the feature measure $\mu_Q$. This requirement ensures that the feature values $Q$ at each region remain unchanged under this transformation.

As illustrated in Figure 10, the proposed algorithm is composed of two parts, namely, feature extraction and classification.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig.png}
\caption{Flowchart of the OMT-FER method.}
\end{figure}

5.1. Conformal-based 3D facial expression recognition

In this subsection, we briefly review the conformal-based 3D FER algorithm (Con-FER) [47] since it is closely related to the proposed OMT-FER algorithm.

First, a disk-shaped conformal map $\psi$ is computed by the discrete surface Ricci flow algorithm [14, 48]. Then, the map is normalized by the Möbius transformation as

$$\phi_C(v) = \psi_C(v) - \psi_C(v_{\text{nose}}) \cdot e^{i\theta},$$

where $\theta = -\text{Arg}\left(\psi_C(v_{R\text{l}eye}) - \psi_C(v_{L\text{l}eye})\right)$, as illustrated in Figure 11.

Once we have the normalized conformal map $\phi_C : \mathcal{M} \to \mathbb{D}$, the feature image for a positive raw feature map $Q : \mathcal{V}(\mathcal{M}) \to \mathbb{R}^+$ is given by $Q \circ \phi_C^{-1} : \mathbb{D} \to \mathbb{R}^+$.

Then, the sparse representation-based classifier [42] is adopted to classify the facial expressions, which will be introduced in Subsection 5.3.
5.2. OMT-based expression feature extraction

Feature extraction involves data normalization, feature creation, and feature transformation. The raw 3D surface is initially processed under some assumptions. Subsequently, a user-specified raw feature $Q$ is created and finally, both the surface and the prescribed raw feature are transformed into the 2D feature image through the normalized measure-preserving map, as illustrated in Figure 12.
The classification step relies on the sparse representation-based classification algorithm [42], which classifies a test feature image \( y \) as the \( i^\ast \)-th expression class if \( y \) is closest to its projection onto the \( i^\ast \)-th expression subspace.

### 5.2.1. Surface normalization.

First, the facial surface \( M \) is required to follow the assumptions stated below.

1. The total surface area equals \( \pi \).
2. The nose tip is located at the origin.
3. The facial surface points toward the positive \( z \)-axis.
4. The \( y \)-axis of the camera coordinate coincides with the \( y \)-axis of the world coordinate.

In Figure 13, assumption 3 means that

\[
\text{(23)} \quad \text{eye}_x = a t_x, \quad \text{eye}_y = a t_y, \quad \text{eye}_z > a t_z, 
\]

while the assumption 4 implies that

\[
\text{(24)} \quad \text{up}_x = 0, \quad \text{up}_y = 0, \quad \text{up}_z > 0. 
\]

The constraints (23) and (24) are naturally satisfied as long as the 3D camera is properly configured.

A nose tip is expected to be the peak of the face, leaving it a convex point on the surface. The HK segmentation [35] suggests that the nose tip falls in the region such that \( K > 0 \) and \( H < 0 \), where \( K \) and \( H \) are the Gaussian curvature and mean curvature, respectively. Assumption 3 indicates that the nose tip is the point having the highest \( z \)-value in that region. Similar works that make use of HK segmentation to detect facial landmarks can be found in [34] and [4].
5.2.2. Feature creation. Suppose that \( Q : \mathcal{V}(\mathcal{M}) \to \mathbb{R}^+ \) denotes the positive raw feature map defined on \( \mathcal{V}(\mathcal{M}) \). The feature value of the triangular face \( \tau = [v_i, v_j, v_k] \in \mathcal{F}(\mathcal{M}) \) can be regarded as an area measure \( \mu_Q : \mathcal{F}(\mathcal{M}) \to \mathbb{R}^+ \) defined by

\[
(25) \quad \mu_Q(\tau) = \frac{1}{3} (Q(v_i) + Q(v_j) + Q(v_k)).
\]

The area measure \( \mu_Q \) is referred to as the feature measure of the feature map \( Q \).

5.2.3. OMT-based feature transformation. For convenience, hereafter we denote \( \mu \) as the feature measure \( \mu_Q \) defined in (25).

Due to the non-uniqueness of the measure-preserving map \( f : (\mathcal{M}, \mu) \to (\mathbb{D}, |\cdot|) \) (see Figure 14 for an example), some selected facial landmarks should be located at certain positions to normalize the measure-preserving map.

![Figure 14](image)

Figure 14: An example of the nonuniqueness of a measure-preserving map. The measure \( \mu \) here denotes the area measure.

To this end, we consider the aligning optimization problem whose solution aligns the selected facial landmarks with their images.

\[
(26) \quad f^* = \arg \min_{f \in F_*} \sum_{i=1}^{n_v} \|v_i - f(v_i)\|_2^2 m_\mu(v_i) + \sum_{j \in S} \alpha_j \|v_j - f(v_j)\|_2^2,
\]

where \( m_\mu \) is the local measure (16), \( S \) is the index set for the selected facial landmarks and \( \{\alpha_j\}_{j \in S} \) are the associated penalty parameters. The solution
f * reduces the distances between the selected landmarks and their images while minimizing the transportation cost (17).

The objective function of the aligning optimization problem (26) can be reduced to

\[
\sum_{i=1}^{n_V} \| v_i - f(v_i) \|_2^2 m^S_\mu(v_i),
\]

where

\[
m^S_\mu(v_i) = \begin{cases} 
  m_\mu(v_i) + \alpha_i & \text{if } i \in S, \\
  m_\mu(v_i) & \text{if } i \notin S.
\end{cases}
\]

Therefore, the objective function (27) can be interpreted as the transportation cost with respect to the enhanced local measure \( m^S_\mu \). This makes the aligning optimization problem (26) a special case of the discrete OMT problem and indicates that the projected gradient descent method (Algorithm 4) stated in Section 4.2 is capable of solving the aligning optimization problem (26).

In numerical experiments, the drastic changes in the value of the enhanced local measure \( m^S_\mu \) incur overlapping triangles near the selected landmarks. Hence, we smooth the enhanced local measure \( m^S_\mu \) and consider the relaxed optimization problem:

\[
f^* = \arg \min_{f \in F_\mu} \sum_{i=1}^{n_V} \| v_i - f(v_i) \|_2^2 \tilde{m}^S_\mu(v_i),
\]

where

\[
\tilde{m}^S_\mu(v_i) = m_\mu(v_i) + \sum_{j \in S} \alpha_j \exp \left[ -\frac{d_g(v_i, v_j)^2}{2\epsilon_j^2} \right]
\]

is the smooth enhanced local measure. The metric \( d_g(v_i, v_j) \) evaluates the geodesic distance between \( v_i \) and \( v_j \) while \( \{\epsilon_j\}_{j \in S} \) are the standard deviations. The solution to (28) is the desired normalized measure-preserving map.

**Remark 5.1.** Let \( \phi_C : \mathcal{M} \to \mathbb{D} \subset \mathbb{C} \) denote the normalized conformal map as in (22). To reduce the computational time cost, we replace the geodesic distance \( d_g \) with the metric induced by the normalized conformal map \( \phi_C \) as

\[
d_C(v_i, v_j) := \| \phi_C(v_i) - \phi_C(v_j) \|_2.
\]
This metric computes the distance on the conformal parameter space $\phi_C(\mathcal{M})$.

The connection between the relaxed optimization problem (28) and the optimization problem defined in (26) is explicitly stated in the following proposition.

**Proposition 5.1.** The objective function of problem (28) converges to that of problem (26), i.e.,

$$\lim_{\epsilon_j \to 0} \sum_{j \in S} \norm{v_i - f(v_i)}^2 m^S_{\mu_j}(v_i) = \sum_{i=1}^n \norm{v_i - f(v_i)}^2 m^S_{\mu}(v_i).$$

**Proof.** A direct computation yields that

$$\lim_{\epsilon_j \to 0} \sum_{j \in S} \alpha_j \lim_{\epsilon_j \to 0} \left\{ \exp \left( -\frac{d_g(v_i, v_j)^2}{2\epsilon_j^2} \right) \right\} = m_{\mu}(v_i) + \sum_{j \in S} \alpha_j \delta(v_i, v_j) = m^S_{\mu}(v_i),$$

where $\delta(v_i, v_j)$ is the Kronecker delta function

$$\delta(v_i, v_j) = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{if } i \neq j. \end{cases}$$

Once the normalized measure-preserving map $f : (\mathcal{M}, \mu) \to (\mathcal{D}, |\cdot|)$ is obtained, the feature image is obtained from the function $f_{\#} \mu : \mathcal{F}(f(\mathcal{M})) \to \mathbb{R}^+$. In particular, if the range of $\mu$ is rescaled to be $[0, 255]$, the intensity on the face $f(\tau) \in \mathcal{F}(f(\mathcal{M}))$ is $\mu(\tau)$. On the feature image, the brighter a region is, the larger the measure of the region is. The resulting feature image is denoted as $[f(\mathcal{M}), f_{\#} \mu]$.

Figure 15 shows the feature images for the mean curvature (row 2) and the Gaussian curvature (row 3) of surfaces.

**Remark 5.2.** The feature image encodes the pushforward measure $f_{\#} \mu := \mu \circ f^{-1}$.

In this part, the feature $Q$ related to the measure $\mu := \mu_Q$ must be a scalar-valued map. If it is a vector-valued map $\vec{Q} = (Q_1, \ldots, Q_q)$, the feature image for $\vec{Q}$ is attained by concatenating the feature images associated with the features $\{Q_i\}_{i=1}^q$. In other words, if $f^{(i)} : (\mathcal{M}, \mu_{Q_i}) \to (\mathcal{D}, |\cdot|)$
Figure 15: Feature images for 3D facial surface “F0001” in the BU-3DFE database [43]. 1st row: 3D surfaces in camera coordinates; 2nd row: feature images for the mean curvature; 3rd row: feature images for the Gaussian curvature.

are the normalized measure-preserving maps, we concatenate the 2D images \{([f^{(i)}(M), f^{(i)}_\mu \mu_Q], q_i = 1 \} to obtain the feature image for \( \tilde{Q} \). Figure 16 illustrates the feature image for vector-valued features representing the mean curvature and the Gaussian curvature.

Figure 16: The feature images of “F0001” in the BU-3DFE database [43] for the mean curvature and the Gaussian curvature.

5.3. Sparse representation-based classification (SRC)

Let \( Q : \mathcal{M} \to \mathbb{R}^q \) be a feature map. Suppose that \( \mathcal{T}RI \subset \mathbb{R}^{w \times q} \) is a training set of feature images obtained in the previous section and \( y \in \mathbb{R}^{w \times q} \) is a
test feature image. The workflow of the SRC model is shown in Figure 17. Parts (a), (b), and (c) marked in Figure 17 can be found in Subsections 5.3.1, 5.3.2, and 5.3.3, respectively.

5.3.1. Construction of feature image-based expression dictionary. We compute the feature images with size \((qw) \times w\) from the training dataset and reshape each of them to an \(m\)-dimensional vector \((m = qw^2)\). The integer \(q\) indicates that the feature map takes the value in \(\mathbb{R}^q\); the feature map is scalar-valued when \(q = 1\).
Let \( n_i \) be the number of training samples of the \( i \)th expression class and \( \nu_{ij} \in \mathbb{R}^m \) be the feature image of the \( j \)th training sample within the \( i \)th expression class. The dictionary associated with the \( i \)th expression class is

\[
D_i = (\nu_{i1}, \cdots, \nu_{in_i}) \in \mathbb{R}^{m \times n_i},
\]

and the expression dictionary for all the \( k \) expression classes is

\[
D = (D_1, \cdots, D_k) \in \mathbb{R}^{m \times n},
\]

where \( n = \sum_{i=1}^{k} n_i \) is the total number of training samples, as illustrated in Figure 18.

Figure 18: The construction of the feature image-based expression dictionary. Here, we consider the mean curvature as the feature.

5.3.2. Sparse approximation of feature image. Given a test feature image \( y \in \mathbb{R}^m \), we linearly approximate \( y \) by \( n \) training samples as

\[
y = D\hat{x} + r,
\]

where \( \hat{x} \in \mathbb{R}^n \) is a sparse coefficient vector, called the sparse representation of the sample \( y \) in terms of the expression dictionary \( D \). This is illustrated in Figure 19.
The sparse representation $\hat{x}$ is solved through $l_0$-norm minimization

$$\hat{x} = \text{arg min}_{x \in \mathbb{R}^n} \|x\|_0 \text{ subject to } \|y - Dx\|_2 \leq \epsilon,$$

where $\|x\|_0 := |\text{supp}(x)|$. In the theory of sparse representation and compressed sensing [8], the solution to the $l_0$-norm minimization (33) is equal to that of the $l_1$-norm minimization $\hat{x} = \text{arg min}_{x \in \mathbb{R}^n} \|x\|_1 \text{ subject to } \|y - Dx\|_2 \leq \epsilon$.

In practice, we consider the $l_0$-norm minimization (33) and adopt the orthogonal matching pursuit (OMP) method [26] to solve the problem (33).

### 5.3.3. Expression classification by class-dependent residuals.

Suppose that $y \in \mathbb{R}^m$ is a test feature image and $\hat{x} \in \mathbb{R}^n$ is the sparse representation of $y$ in terms of expression dictionary $D$ defined in (31) and (32).

For each expression class $i$, the characteristic function $\delta_i : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is defined on the space of coefficient vectors. The value $\delta_i(x)$ returns the coefficients associated only with class $i$ and sets the others to zero. Specifically,

$$\delta_i \left( \begin{array}{c} x_{11} \cdots x_{1n} \mid \cdots \cdots \cdots \mid x_{k1} \cdots x_{kn} \end{array} \right) = \left( \begin{array}{c|c|c} 0 \cdots 0 \mid x_{i1} \cdots x_{in} \mid 0 \cdots 0 \end{array} \right).$$

The test sample $y$ is assigned to the class that minimizes the class-dependent error between $y$ and its projection onto the corresponding ex-
Algorithm 5 Orthogonal Matching Pursuit

**Input:** Test image $y \in \mathbb{R}^m$, dictionary $D = (x_1, \ldots, x_n) \in \mathbb{R}^{m \times n}$, level of sparsity $T$

**Output:** Sparse representation $\hat{x} \in \mathbb{R}^n$ subject to $\|\hat{x}\| \leq T$

1: **procedure** OMP($y$, $D$, $T$)
2: Initialization:
   - (support of $x$) $\Omega \leftarrow \emptyset$
   - (linear approximation of $y$) $\hat{y} \leftarrow 0$
   - (residual) $r \leftarrow y - \hat{y}$
3: for $t \leftarrow 1, \ldots, T$ do
4:     Find the atom which is as orthogonal to the subspace span$\{x_j \mid j \in \Omega\}$ as possible.
        $i_t = \arg \max_{i \in \Omega} \langle r, x_i \rangle$
5:     Update the support
        $\Omega \leftarrow \Omega \cup \{i_t\}$
6:     Compute the approximation of $y$ with the atoms $\{x_j \mid j \in \Omega\}$ using orthogonal projection.
        $\begin{cases} 
        \hat{x} = \arg \min_x \|y - Dx\|_2^2 \quad \text{subject to} \quad \text{supp}(x) = \Omega, \\
        \hat{y} = Dx.
        \end{cases}$
7:     Compute the residual
        $r \leftarrow y - \hat{y}$
8: end for
9: **return** $\hat{x}$
10: **end procedure**

Remark 5.3. As mentioned in [37], the performance of the sparse representation-based classification model is susceptible to the misalignment of the images. Thanks to the normalized mesh satisfying the assumptions in Sub-
Figure 20: The classification of a test feature image $y$ by projecting it onto the expression subspaces, where the feature is selected to be the mean curvature.

Section 5.2.1 and the aligning optimization (26) or (28), the resulting feature images are all properly aligned.

The procedure of the SRC is summarized in Algorithm 6.

**Algorithm 6 SRC**

**Input:** Test feature image $y \in \mathbb{R}^m$, training set of feature image $\mathcal{TRI} = \{ \nu_{ij} \in \mathbb{R}^m | 1 \leq j \leq n_i, 1 \leq i \leq k \}$, level of sparsity $T$

**Output:** Predicted class $i^* \in \{1, \ldots, k\}$

1: procedure SRC($y$, $\mathcal{TRI}$, $T$)
2: Construct expression dictionary from the training dataset $\mathcal{TRI}$.
3: Solve the sparse representation $\hat{x} \leftarrow \text{OMP}(y, D, T)$. ▷ Algorithm 5
4: Classify $y$ to the class which minimizes the class-dependent errors.
5: $i_* = \arg \min_{i=1, \ldots, k} \| y - D\delta_i(\hat{x}) \|_2$
6: return $i_*$
7: end procedure
Remark 5.4. The main difference between the proposed OMT-FER and the Con-FER [47] is that the Con-FER encodes a prescribed feature map on the surface to the feature image using the normalized conformal map. In contrast, the OMT-FER encodes the feature using the normalized measure-preserving map.

6. Numerical experiments

In this section, we evaluate the performance of the OMT-FER model and compare it with that of the Con-FER. For this reason, all the configurations except for the hyperparameters of relaxed aligning optimization follow the settings in [47].

6.1. 3D face database and preprocess

We test the OMT-FER method on the BU-3DFE database [43], a benchmark database for the 3D facial expression recognition problem. This database is, for instance, used in [21, 20, 49, 47].

6.1.1. Face database. The BU-3DFE database includes 2500 different facial surfaces of 100 subjects. Each of the subjects contains one neutral face and six basic emotions (anger, disgust, fear, happiness, sadness, and surprise) at 4 levels of intensity. In addition, each facial surface is provided with 83 facial landmarks. The 83 facial landmarks are shown in Figure 21, and the relations between facial features and the 83 landmarks are described in Table 2. An overview of the BU-3DFE database is listed in Table 1.

<table>
<thead>
<tr>
<th>Field</th>
<th>Information</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data type</strong></td>
<td>3D textured mesh + 2D texture image</td>
</tr>
<tr>
<td><strong>Subjects</strong></td>
<td>100 persons</td>
</tr>
<tr>
<td></td>
<td>(56 females, 44 males)</td>
</tr>
<tr>
<td><strong>Faces</strong></td>
<td>2500</td>
</tr>
<tr>
<td><strong>Expressions</strong></td>
<td>6 basic emotions × 4 intensity levels + neutral</td>
</tr>
<tr>
<td></td>
<td>(25 different faces per subject)</td>
</tr>
<tr>
<td><strong>Number of Landmarks</strong></td>
<td>83</td>
</tr>
</tbody>
</table>
Figure 21: The 83 facial landmarks provided by BU-3DFE database.

Table 2: Correspondence between the facial features and the 83 facial landmarks

<table>
<thead>
<tr>
<th>Facial Feature</th>
<th>Indices of Facial Landmarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left Eye</td>
<td>1 ··· 8</td>
</tr>
<tr>
<td>Right Eye</td>
<td>9 ··· 16</td>
</tr>
<tr>
<td>Left Eyebrow</td>
<td>17 ··· 26</td>
</tr>
<tr>
<td>Right Eyebrow</td>
<td>27 ··· 36</td>
</tr>
<tr>
<td>Nose Contour</td>
<td>37 ··· 48</td>
</tr>
<tr>
<td>Outer Mouth</td>
<td>49 ··· 60</td>
</tr>
<tr>
<td>Inner Mouth</td>
<td>61 ··· 68</td>
</tr>
<tr>
<td>Face Contour</td>
<td>69 ··· 83</td>
</tr>
</tbody>
</table>

6.1.2. Preprocessing. All the facial surfaces are stored as triangular meshes. They undergo the following preprocessing steps to become surfaces with disk topology.

1. Conversion into triangular mesh.
2. Removal of small connected components.
3. Holes filling.
4. Mesh smoothing [6].
5. Uniform remeshing [46].

The boundary trimming step first probes a simple closed piecewise geodesic curve that passes through the landmarks of the face contour and encloses all the facial features, i.e., landmarks 1 to 68 described in Table 2.
Then, the exterior patches of the simple closed curve are removed. The fast marching algorithm [31] is employed to find the geodesic curves, and the associated MATLAB toolbox can be found in [27].

6.2. Model configuration

6.2.1. Feature creation. Since the facial surface contains nothing more than the geometric information, the possible choices for the feature maps on the surface are related to the geometric quantities. It is well known that a smooth surface is locally the graph of a differential function $\phi : U \subset \mathbb{R}^2 \to \mathbb{R}$ [7, Proposition 2-2.3 on p. 63]. The Taylor theorem states that

$$
\phi(x) = \phi(x_0) + \nabla \phi(x_0)^\top (x - x_0) + \frac{1}{2} (x - x_0)^\top \text{Hess}_{\phi}(x_c)(x - x_0),
$$

implying that the surface is in part characterized by the surface normal $\nabla \phi$ and the surface curvature $\text{Hess}_{\phi}$.

In our approach, we adopt the Gaussian and the mean curvatures as the intensity of the feature images. Although the RGB scales of the texture images can also be used as the feature, however, the texture images are usually affected by the difference in skin color, makeup, environmental lighting, etc., and cause inaccuracy. The Gaussian and the mean curvatures only depend on the geometry of the surface so that they have no such drawbacks.

Let $Q_H$ and $Q_K$ be the positive feature maps associated with the mean curvature $H$ and the Gaussian curvature $K$, respectively, defined as

\begin{align}
Q_H &= \arctan(Z_H) + \pi/2 \quad \text{and} \\
Q_K &= \arctan(Z_K) + \pi/2,
\end{align}

where

$$
Z_H := \frac{H - \text{mean}(H)}{\text{std}(H)}, \quad Z_K := \frac{K - \text{mean}(K)}{\text{std}(K)},
$$

are the normalized values of $H$ and $K$, respectively. The arctangent in (34) and (35) is a sigmoid function that takes values in a bounded domain. Such a function is capable of suppressing outliers whose values are extremely large/small (see Figure 22). The last terms of (34) and (35) are meant to make the feature maps strictly positive.
6.2.2. Hyperparameters setting. The hyperparameters for the OMT-FER model are made up of (1) hyperparameters for the relaxed aligning optimization problem (28); (2) the size of the feature image; and (3) the level of sparsity $T$ in the OMP algorithm.

6.2.2.1. Relaxed aligning optimization. Recall that the relaxed aligning optimization (28) is

$$f^* = \arg \min_{f \in F_{\mu}} \sum_{i=1}^{n_v} \| v_i - f(v_i) \|_2^2 \tilde{m}_\mu^S(v_i),$$

where $\mu$ is the feature measure defined in (25), $\tilde{m}_\mu^S$ is the smoothed enhanced local measure

$$\tilde{m}_\mu^S(v_i) = m_\mu(v_i) + \sum_{j \in S} \alpha_j \exp \left[ - \frac{d_C(v_i, v_j)^2}{2\epsilon_j^2} \right],$$

and $d_C$ is defined in (30). The hyperparameters are $S$ (selected landmarks), $\{\alpha_j\}_{j \in S}$ (amplitude) and $\{\epsilon_j\}_{j \in S}$ (standard deviation).

The choices of these hyperparameters are illustrated in Table 3, where

1. $v_{l(i)}$ denotes the vertex for the $i$-th facial landmarks provided by the BU-3DFE database. ($1 \leq i \leq 83$).
2. $v_{\text{nose}}$ denotes the nose tip. (See Section 5.2.1).
3. $u := \phi_C(v)$, $\phi_C$ is the normalized conformal map.
Table 3: Hyperparameters for relaxed aligning optimization

<table>
<thead>
<tr>
<th>Name of Selected Landmark</th>
<th>Derivation</th>
<th>( \tau_j )</th>
<th>( \epsilon_j )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nose tip</td>
<td>( v_{\text{tip}} )</td>
<td>mean ( d_C(v_{\text{tip}}, v_{\text{landmark}}) )</td>
<td>10</td>
</tr>
<tr>
<td>Left eye inner corner</td>
<td>( v_{(9)} )</td>
<td>min ( {d_C(v_{(9)}, v_{(1)}), d_C(v_{(9)}, v_{(11)})} )</td>
<td>8</td>
</tr>
<tr>
<td>Right eye inner corner</td>
<td>( v_{(20)} )</td>
<td>min ( {d_C(v_{(20)}, v_{(1)}), d_C(v_{(20)}, v_{(11)})} )</td>
<td>8</td>
</tr>
<tr>
<td>Left eye center</td>
<td>( v_{(17)} )</td>
<td>max ( {d_C(v_{(17)}, v_{\text{landmark}}) : i \in {1, 2, 3, 4, 5, 6, 7, 8}} )</td>
<td>6</td>
</tr>
<tr>
<td>Right eye center</td>
<td>( v_{(20)} )</td>
<td>max ( {d_C(v_{(20)}, v_{\text{landmark}}) : i \in {10, 11, 12, 13, 14, 15, 16}} )</td>
<td>6</td>
</tr>
<tr>
<td>Left eye outer corner</td>
<td>( v_{(15)} )</td>
<td>min ( {d_C(v_{(15)}, v_{(1)}), d_C(v_{(15)}, v_{(11)})} )</td>
<td>4</td>
</tr>
<tr>
<td>Right eye outer corner</td>
<td>( v_{(33)} )</td>
<td>min ( {d_C(v_{(33)}, v_{(13)}), d_C(v_{(33)}, v_{(11)})} )</td>
<td>4</td>
</tr>
<tr>
<td>Nose root</td>
<td>( v_{\text{root}} ) = ( v_{(39)} )</td>
<td>mean ( {d_C(v_{(39)}, v_{(1)}), d_C(v_{(39)}, v_{(11)})} )</td>
<td>4</td>
</tr>
<tr>
<td>Left mouth outer corner</td>
<td>( v_{(55)} )</td>
<td>min ( {d_C(v_{(55)}, v_{(11)}), d_C(v_{(55)}, v_{(11)})} )</td>
<td>4</td>
</tr>
<tr>
<td>Right mouth outer corner</td>
<td>( v_{(55)} )</td>
<td>min ( {d_C(v_{(55)}, v_{(11)}), d_C(v_{(55)}, v_{(11)})} )</td>
<td>4</td>
</tr>
<tr>
<td>Lower middle mouth corner</td>
<td>( v_{(58)} )</td>
<td>( d_C(v_{(58)}, v_{(11)}) )</td>
<td>4</td>
</tr>
<tr>
<td>Left eyebrow inner corners</td>
<td>( v_{(17)} )</td>
<td>( d_C(v_{(17)}, v_{(17)}) )</td>
<td>4</td>
</tr>
<tr>
<td>Right eyebrow inner corner</td>
<td>( v_{(27)} )</td>
<td>( d_C(v_{(27)}, v_{(27)}) )</td>
<td>4</td>
</tr>
<tr>
<td>Glabella (between eyebrows)</td>
<td>( v_{\text{glabella}} ) = ( v_{(46)} ) (mean ( {v_{(46)}, v_{(46)}, \ldots} )</td>
<td>( d_C(v_{(46)}, v_{\text{glabella}}), i = 11, 20, 27, 36 )</td>
<td>4</td>
</tr>
</tbody>
</table>

Figure 23 marks the selected landmarks \( \{v_j\}_{j \in S} \) as the green dots and the default 83 facial landmarks as the red dots. In Figure 23b, we further plot these landmarks on the conformal parameter space \( \phi_C(M) \) and draw a blue circle of radius \( \epsilon_j \) around the selected landmark \( v_j \) for each \( j \in S \).

![Original mesh M](Image) ![Conformal image \( \phi_C(M) \)](Image)

Figure 23: Selected landmarks on original mesh \( M \) and conformal image \( \phi_C(M) \). The green dots are the selected landmarks, and the red dots are the 83 facial landmarks provided by the BU-3DFE database. Each circle surrounding a selected landmark (green dot) \( v_j \) is of radius \( \epsilon_j \).

### 6.2.2.2. Size of the feature image.

Similar to the strategy in [47], we initially save the feature image with size 256 \( \times \) 256 and crop 20 pixels at each side (216 \( \times \) 216). Finally, we resize the trimmed image to 32 \( \times \) 32. Please see Figure 24.
6.2.2.3. OMP method. While solving the sparse representation, we employ the OMP method and set the level of sparsity $T$ to 30 as in [47].

6.3. Experiment configuration

Similar to the experimental setting of [47], the experiment is executed for 100 rounds. At the beginning of each round, we randomly select 60 subjects and retain the samples of the six basic emotions having the highest two intensity levels, i.e., $60 \times 6 \times 2 = 720$ faces.

To attain the recognition rate for each round, we conduct subject-independent 10-fold cross-validation on the selected 720 faces. The 720 samples of the 60 subjects are divided into 10 partitions, and each partition includes 72 samples derived from 6 subjects (10%). During each of the 10 iterations, one of the 10 partitions serves as the test data set, while the remaining 9 partitions are the training data set. The resulting average recognition rate is estimated by averaging the recognition rates for the results of 100 rounds of 10-fold cross-validations.

6.4. Numerical results

Figures 25 and 26 display a portion of the feature images for two of the prescribed features $Q_H$ and $Q_K$, respectively.

In process visualization, we consider the feature $Q_H$ and obtain a test feature image $y$ with sad expression from a random training-testing split. Then, we visualize how the decision is made within the classification step.

Figure 27(a)-(c) show the original test feature image $y$, the sparse coefficients $\hat{x} \in \mathbb{R}^{648}$ in terms of the dictionary $D \in \mathbb{R}^{1024 \times 648}$ generated by the 648 training feature images, and the linear approximation of $y$ (i.e., $D\hat{x}$).
Figure 25: Feature images for $Q_H$. Each expression contains 2 intensity levels.

Figure 26: Feature images for $Q_K$. Each expression contains 2 intensity levels.

Figure 27: Sparse coefficients and the linear approximation of test feature image $y$.

Figure 28 plots the coefficients associated with each of the expression subspaces (i.e., $\delta_i(\hat{x})$) and the projection of $y$ onto each of the expression subspaces (i.e., $D\delta_i(\hat{x})$).
Finally, the bar plot in Figure 29 reveals the class-dependent errors (i.e., $\|y - D\delta(\hat{x})\|_2$), among which the class-dependent error with respect to sadness is the minimal one.

We test the OMT-FER model with three different features: $Q_K$, $Q_H$, and $\vec{Q} := (Q_H, Q_K)$. The performance associated with these three features is expressed in the form of confusion matrices. Please see Table 5, 4, and 6.

From Tables 5 and 4, we can see that the mean curvature contains more expressional information than that of the Gaussian curvature. Nevertheless, Tables 4 and 6 reveal that the Gaussian curvature provides complementary expressional information that the mean curvature does not have.
Table 4: Confusion matrix for OMT-FER with feature $Q_H$

<table>
<thead>
<tr>
<th>%</th>
<th>AN</th>
<th>DI</th>
<th>FE</th>
<th>HA</th>
<th>SA</th>
<th>SU</th>
</tr>
</thead>
<tbody>
<tr>
<td>AN</td>
<td>57.32</td>
<td>12.16</td>
<td>6.99</td>
<td>2.15</td>
<td>20.00</td>
<td>1.38</td>
</tr>
<tr>
<td>DI</td>
<td>11.12</td>
<td>60.27</td>
<td>7.08</td>
<td>7.44</td>
<td>6.34</td>
<td>7.76</td>
</tr>
<tr>
<td>FE</td>
<td>8.91</td>
<td>9.55</td>
<td>45.72</td>
<td>19.18</td>
<td>10.12</td>
<td>6.53</td>
</tr>
<tr>
<td>HA</td>
<td>1.57</td>
<td>5.82</td>
<td>13.93</td>
<td>75.63</td>
<td>1.30</td>
<td>1.76</td>
</tr>
<tr>
<td>SA</td>
<td>17.35</td>
<td>4.77</td>
<td>9.68</td>
<td>3.32</td>
<td>61.46</td>
<td>3.43</td>
</tr>
<tr>
<td>SU</td>
<td>1.24</td>
<td>3.76</td>
<td>4.03</td>
<td>1.90</td>
<td>1.49</td>
<td>87.58</td>
</tr>
<tr>
<td>Avg</td>
<td>64.66</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5: Confusion matrix for OMT-FER with feature $Q_K$

<table>
<thead>
<tr>
<th>%</th>
<th>AN</th>
<th>DI</th>
<th>FE</th>
<th>HA</th>
<th>SA</th>
<th>SU</th>
</tr>
</thead>
<tbody>
<tr>
<td>AN</td>
<td>46.07</td>
<td>13.15</td>
<td>7.93</td>
<td>3.26</td>
<td>26.99</td>
<td>2.60</td>
</tr>
<tr>
<td>DI</td>
<td>11.54</td>
<td>52.87</td>
<td>7.92</td>
<td>5.76</td>
<td>8.22</td>
<td>13.69</td>
</tr>
<tr>
<td>FE</td>
<td>10.49</td>
<td>9.89</td>
<td>34.95</td>
<td>20.27</td>
<td>13.51</td>
<td>10.89</td>
</tr>
<tr>
<td>HA</td>
<td>2.42</td>
<td>6.79</td>
<td>13.00</td>
<td>72.37</td>
<td>1.94</td>
<td>3.48</td>
</tr>
<tr>
<td>SA</td>
<td>24.56</td>
<td>7.21</td>
<td>10.40</td>
<td>2.47</td>
<td>49.95</td>
<td>5.42</td>
</tr>
<tr>
<td>SU</td>
<td>1.41</td>
<td>6.00</td>
<td>5.08</td>
<td>2.01</td>
<td>2.39</td>
<td>83.12</td>
</tr>
<tr>
<td>Avg</td>
<td>56.55</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 6: Confusion matrix for OMT-FER with feature $\bar{Q} = (Q_H, Q_K)$

<table>
<thead>
<tr>
<th>%</th>
<th>AN</th>
<th>DI</th>
<th>FE</th>
<th>HA</th>
<th>SA</th>
<th>SU</th>
</tr>
</thead>
<tbody>
<tr>
<td>AN</td>
<td>59.73</td>
<td>11.14</td>
<td>5.53</td>
<td>1.89</td>
<td>20.52</td>
<td>1.19</td>
</tr>
<tr>
<td>DI</td>
<td>10.57</td>
<td>64.13</td>
<td>6.38</td>
<td>5.31</td>
<td>5.35</td>
<td>8.25</td>
</tr>
<tr>
<td>FE</td>
<td>7.80</td>
<td>8.48</td>
<td>46.79</td>
<td>19.06</td>
<td>10.55</td>
<td>7.32</td>
</tr>
<tr>
<td>HA</td>
<td>1.17</td>
<td>4.28</td>
<td>10.97</td>
<td>81.16</td>
<td>0.88</td>
<td>1.54</td>
</tr>
<tr>
<td>SA</td>
<td>17.60</td>
<td>3.41</td>
<td>8.51</td>
<td>2.72</td>
<td>65.25</td>
<td>2.52</td>
</tr>
<tr>
<td>SU</td>
<td>0.53</td>
<td>2.58</td>
<td>3.57</td>
<td>1.87</td>
<td>0.95</td>
<td>90.51</td>
</tr>
<tr>
<td>Avg</td>
<td>67.93</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Finally, these three tables together show that when the feature is related to surface curvature, the OMT-FER model accurately discriminates happiness and surprise from others but frequently confuses fear with happiness and sadness.

6.4.1. Models comparison and extension. The experiment in [47] chooses the mean curvature as the feature for the Con-FER model. Accordingly, we consider the feature $Q_H$ in the following experiments.
The confusion matrices for the OMT-FER and Con-FER models with feature $Q_H$ are listed in Tables 4 and Table 7, respectively. We observe that the accuracy of the OMT-FER algorithm is slightly less than that of the Con-FER algorithm, with average recognition rates of 64.66% and 67.38%, respectively. To improve the recognition rates, we use both feature images of the Con-FER and OMT-FER and propose a mixture FER (Mix-FER) algorithm. The Mix-FER algorithm concatenates the feature images generated by Con-FER and OMT-FER algorithms in the feature extraction step and adopts the sparse representation-based classifier as in the OMT-FER model. The flowchart of the Mix-FER algorithm is depicted in Figure 30.

![Figure 30: Workflow of the Mix-FER algorithm.](image)

The confusion matrix for the Mix-FER with feature $Q_H$ is shown in Table 8. From Tables 4, 7, and 8, we can see that the Mix-FER model clearly outperforms the Con-FER and OMT-FER models. The average recognition rates of these three models are displayed in Table 9. The outstanding performance of the Mix-FER also reveals that the Con-FER and OMT-FER models are mutually complementary when the feature of the surface is fixed.
Table 8: Confusion matrix for the Mix-FER model with feature $Q_H$

<table>
<thead>
<tr>
<th></th>
<th>AN</th>
<th>DI</th>
<th>FE</th>
<th>HA</th>
<th>SA</th>
<th>SU</th>
</tr>
</thead>
<tbody>
<tr>
<td>AN</td>
<td><strong>61.51</strong></td>
<td>10.31</td>
<td>7.01</td>
<td>1.51</td>
<td>19.28</td>
<td>0.38</td>
</tr>
<tr>
<td>DI</td>
<td>8.78</td>
<td><strong>70.75</strong></td>
<td>7.57</td>
<td>5.30</td>
<td>3.36</td>
<td>4.24</td>
</tr>
<tr>
<td>FE</td>
<td>7.64</td>
<td>9.18</td>
<td><strong>52.17</strong></td>
<td>16.51</td>
<td>8.05</td>
<td>6.46</td>
</tr>
<tr>
<td>HA</td>
<td>0.64</td>
<td>3.11</td>
<td>9.88</td>
<td><strong>83.96</strong></td>
<td>1.05</td>
<td>1.37</td>
</tr>
<tr>
<td>SA</td>
<td>17.36</td>
<td>3.02</td>
<td>7.91</td>
<td>2.13</td>
<td><strong>67.31</strong></td>
<td>2.27</td>
</tr>
<tr>
<td>SU</td>
<td>0.36</td>
<td>1.65</td>
<td>3.43</td>
<td>1.34</td>
<td>0.37</td>
<td><strong>92.85</strong></td>
</tr>
<tr>
<td>Avg.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td><strong>71.42</strong></td>
</tr>
</tbody>
</table>

Table 9: Average recognition rates for the Con-FER, OMT-FER, and Mix-FER

<table>
<thead>
<tr>
<th>FER model</th>
<th>Con-FER</th>
<th>OMT-FER</th>
<th>Mix-FER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recognition rate (%)</td>
<td>67.38</td>
<td>64.66</td>
<td><strong>71.42</strong></td>
</tr>
</tbody>
</table>

7. Concluding remarks

In this paper, we propose a fully automatic 3D facial expression recognition model that is based on the feature image derived from an OMT map.

First, we specify the feature map with some prior knowledge and normalize the size and position of the 3D facial surface. Then, we transform the 3D facial surface into the unit disk through the OMT map that is solved using the projected gradient descent method. Finally, the feature image is created by setting the grayscale intensity of a triangular face on the disk as the measure on it.

In the classification step, we represent the test feature image as a sparse linear combination of training feature images and compute the projection of the test input onto each subspace of expressions by restricting the previously obtained sparse coefficients. The test feature image is assigned to the expression class with the closest subspace of expressions.

Numerical experiments indicate that the proposed OMT-FER with the curvature-based feature has fairly high accuracy and is competitive with the existing Con-FER algorithm. In addition, the Mix-FER combining the OMT-FER and the Con-FER has significantly higher accuracy than both algorithms.

One possible way to further improve the accuracy of FER is to replace the sparse representation-based classifier with the state-of-the-art classification model, e.g., the convolution neural network. This will be pursued in our future work.
Acknowledgements

This work was partially supported by the National Centre of Theoretical Sciences (NCTS), ST Yau Center in Taiwan, the Shing-Tung Yau Center and Big Data Computing Center in Southeast University. T. Li was supported in part by the National Natural Science Foundation of China (NSFC) 11971105. M.-H. Yueh was partially supported by the Ministry of Science and Technology (MoST) 109-2115-M-003-010-MY2 and 110-2115-M-003-014-, and the Ministry of Education (MoE).

References


**Tiexiang Li**  
**School of Mathematics, Southeast University**  
**Nanjing 211189**  
**Nanjing Center for Applied Mathematics**  
**Nanjing 211135**  
**People’s Republic of China**  
**E-mail address: txli@seu.edu.cn**

**Pei-Sheng Chuang**  
**Department of Applied Mathematics**  
**National Yang Ming Chiao Tung University**  
**Hsinchu 300**  
**Taiwan**  
**E-mail address: pschuang96@gmail.com**
Mei-Heng Yueh  
Department of Mathematics  
National Taiwan Normal University  
Taipei 116  
Taiwan  
E-mail address: yue@ntnu.edu.tw  

Received January 17, 2022