Image Deformation with Vector-field Interpolation based on MRLS-TPS

HUABING ZHOU1,2, YUYU KUANG1, ZHENHONG YU3, SHIQIANG REN1, YANDUO ZHANG1, TAO LU1, JIAYI MA4

1School of Computer Science and Engineering, Hubei Key Laboratory of Intelligent Robot, Wuhan Institute of Technology, Wuhan, 430205, China.
2Department of Computer and Information Sciences, Temple University, Philadelphia, 19122, USA.
3Guangdong Polytechnic of Science and Technology, Zhuhai, 519090, China
4Electronic Information School, Wuhan University, Wuhan, 430072, China

This work was supported in part by the National Natural Science Foundation of China under Grant 41501505, Grant 61503288, Grant 61502354, Grant 61502354, Grant 61501413, in part by the PhD Start-up Fund of Guangdong Natural Science Foundation under Grant 2016A030310306 and in part by the financial support from China Scholarship Council.

ABSTRACT Image deformation has been successfully applied in many different kinds of fields. However, how to get an approach with high efficiency and perfect visual effect remains a challenge task. In this paper, we present a vector-field interpolation method for non-rigid image deformation, which is based on Moving Regularized Least Squares optimization with Thin-plate Spline (MRLS-TPS). The proposed approach takes user-controlled points as input data, and estimates the spatial transformation for each pixel by the control points. In order to achieve a realistic deformation, we formulate the deformation as a novel closed-form transformation estimation problem by Moving Regularized Least Squares (MRLS). Unlike Moving Least Squares (MLS), we model the mapping function by a non-rigid Thin-plate Spline (TPS) function with a regularization coefficient. Therefore, the deformation can not only satisfy global linear affine transformation, but also adapt to local non-rigid deformation. In terms of the transformation, we derive a closed-form solution and achieve a fast implementation. Furthermore, the approach can show us a wonderful user experience, and give us fast and convenient manipulating. Extensive experiments on 2D images and 3D surfaces demonstrated that the proposed method performs better than other state-of-the-art methods like MLS and the commercial software as Adobe PhotoShop CS 6, especially in the case of flexible object motion.

INDEX TERMS Deformation; vector-field interpolation; MRLS-TPS

I. INTRODUCTION

Image deformation is developed upon image processing and computer graphics, it renders new visual effects by pixel interpolation and geometric transformation [1]. However, it is a big challenge to get the natural and smooth distortion effect in real-time, eliminating the external factors. In order to solve this problem, Wolberg et al. [2] researched a grid-based image warping algorithm. After operators select several feature control points and establish the image feature network model, the rest of the image was automatically adjusted under certain rules and constraints by changing the limited control points on the photo, ensuring only a small area near the point affected by the change position of the control point [3]. However, this method requiring alignment of the grid lines with the control point of the parameterized curve, is really hard for convenient operation, and this limits the breadth and depth of deformation applications in many ways.

Currently some state-of-the-art works have focused on specifying deformations with different handles, taking the forms of points, lines, and even polygon grids such as triangular grid. When the operator alters the position of these handles, corresponding images deform in an intuitive fashion. Song et al. [4] introduced B-spline surface method to construct the mapping relationship in image morphing. Firstly, a set of feature points are calibrated on the starting image, and B-spline surfaces are generated by interpolation. Then the position of the feature points is changed by operation, B-spline surfaces corresponding to the ending image are generated by interpolation. These methods provide a convenient way for researchers to experiment, making the deformation process easier to control and more flexible. And
we mean to use a set of user-controlled handles to generate a dense correspondence. However, complex calculations take a lot of time.

In order to deal with the limitation, some methods investigate the transformation type to get the better desirable results, these algorithms view the deformation as a scattered data interpolation problem. Ruprecht et al. [5] proposed a face morphing algorithm with discrete feature points, which applied distortion in the real life that people can realistically perceive, causing a sensation at the time. Meanwhile, Lee et al. [2] raised a multiple free-form deformation approach, and computed the energy minimization method later, greatly simplifying computational complexity and saving computing runtime. Adams et al. [6] completed the motion of a meshless shape with physical simulation. These deformation approaches above are based on basic functions and weights for scattered data interpolation, avoiding a number of basis functions by changing the resolution of control points adaptively. However, these algorithms ignore considering topological relations of the shape in the process of image deformation, narrowing the scope of deformation application.

Schaefer et al. [7] used linear functions such as rigid transformation to deform images based on MLS in 2006. The approach takes controlling points or line segments for deforming original images, performing three kind of deformation respectively: affine transformation, similarity transformation, and rigid transformation. Especially, the application of MLS [8] on rigid transformations makes the deformation ‘as-rigid-as-possible’ [9], greatly improving the current situation of image deformation. However, the morphological topological relations of image morphing are not considered. These approaches mentioned above have some limitations that they take rigid transformation to model the deformation. Since some deformation do not have the property of segmental rigidity, such as the burning candle flame and the changing facial expression. This inspires us to solve this problem with non-rigid model. In 2013, Ma et al. [10] proposed an image warping algorithm based on MRLS using the regularization techniques [11]–[17]. This method adds a regularized term based on the moving least square method, and constrains the best displacement function through this regularity. The deformation function in a reproducing kernel Hilbert space [18], [19], so that it has a high computational efficiency with a simple closed solution. And the deformation results in some ways perform even better than the most advanced algorithms because of the presence of constraints. Inspired by this article, a thin plate spline function [20] is introduced instead of its non-rigid fitting function [21]. Since the spline tool TPS can be decomposed into two parts with a close-form solution: global linear affine transformation and local non-rigid deforming component controlled by coefficients. TPS produces a smooth functional mapping for supervised learning, and it has no free parameters to manual tuning. For the control points in each deformation case, we choose the TPS for parameterization to generate a smooth mapping after a lot of experimental comparisons finally.

Therefore, we formulate the transformation estimation as a vector-field interpolation problem by a moving regularized least squares method with thin-plate spline, and we abbreviate it to MRLS-TPS from now on. MRLS-TPS can deal with both 2D image warping and 3D surface deformation. In this paper, the contribution includes the following three aspects.

- Firstly, we bring the TPS function with regularization technology in the deformation case, which can help us to get more realistic deformation especially for non-rigid morphing.
- Secondly, we propose a unified model to deal with both global image deformation and local image morphing.
- Thirdly, we get a fast implementation, it enables our approach to handle large-scale datasets.

The layout of the rest paper is distributed as follows. Part II describes related works and analyzes their limitations. Part III presents the framework of our approach and details the basic deformation algorithm of MRLS-TPS. Part IV performs the experimental verification of the MRLS-TPS method. Finally, conclusions and some suggestions for the future work are offered in part V.

II. RELATED WORK

Image deformation has a large number of useful applications ranging from virtual reality, movie effects, medical imaging, remote sensing animation production to criminal detection [3], [7], [22]–[27]. According to the form based on the image features, image deformation methods can be divided into three categories: image deformation based on feature grids, image deformation based on feature lines, and image deformation based on feature points.

A. IMAGE DEFORMATION BASED ON FEATURE GRIDS

The technique of image morphing studied firstly and widely used is the grid-based deformation, such as free-form deformation [28] and skeletal-based deformation [29]. Free-form deformation has been widely applied in commercial software. The approach puts embedding deformed objects into a geometric shape such as a quadrilateral, the geometry is distorted by editing the geometric shapes surrounding the object. However, setting up and operating control grids is time-consuming. Skeletal-based deformation generates the skeleton according to the deformation model. Each vertex on the geometric model of the object is parameterized to one skeleton corresponding to the other, then the skeleton manipulate to move, driving vertex movement to achieve deformation. But the distortions can produce unjustified distortions around the price nodes. Nishita et al. [30] described features by using non-uniform controlling grids, generating an image transformation method based on 2D spline grid mapping. The method firstly defines the corresponding grid points between the starting original image and the ending image, and the displacement among the grids is calculated by different solutions. The approach has the advantages of fast deformation speed, partial control function and smooth
morphing effect, but the deformation control is more difficult. Kouzani et al. [31] proposed a triangular mesh deformation algorithm to obtain a more extensive deformation effect. Subsequently, Takashi et al. [32] raised a 3D grid deformation algorithm based on multi-resolution mesh interpolation (MIMesh), which was easy to store data, control morphing and overcome the existing instability problems in the traditional technology. However, there are some shortcomings such as difficulty in control and connection of irregularities. In total, these approaches have complicated feature annotation and complex operations. It motivates researches to deep further studies on image morphing algorithm based on feature lines.

B. IMAGE DEFORMATION BASED ON FEATURE LINES

Neely and Beier [33] proposed image deformation algorithm based on the feature line segments firstly in 1992, and applied it to face morphing. The method allowing the operator to control the deformation object intuitively, is convenient to achieve expected visual effect by specifying interactive feature line pairs of the image. But the calculation turns complicated with the number of control points increased, leading to rather cumbersome process and achieving undesirable results in most cases. Bao and Xu [34] presented a viewing angle synthesis technique by using 2D discrete wavelet to generate viewpoint gradients. They did not know camera calibration and image depth information actually at that time. Firstly, the basic matrices of the image viewpoint pairs are estimated, and then parallel viewpoint conversion and image pre-transform are settled. The image is deformed to the stratification using wavelet transformation. The corresponding wavelet coefficients in the two images generate a multi-resolution middle viewpoint by linear interpolation, and finally the image is processed to generate the correct viewpoint image. Surazhsky and Gotsman [35] made use of the convex boundary features uniformed in homogeneous planar triangles to achieve the deformation. This approach uses the theory of triangular isomorphism, keeping the local characteristics stable, making the result of gradient deformation natural. Lee et al. [36] raised a piecewise linear approximation of the optimization method, speeding up the image deformation algorithm based on feature lines, and getting a more smooth and natural morphing image. In such algorithms, the position of each pixel is constrained by the distance invariance among the feature points and line segments before and after deformation. Though these approaches are more expressive than grid-based morphing algorithms, the setting lines are often influenced by the complexity of the scene, leading to distortion and pixel misalignment in local deformation particulars. This inspires the emergence of point-based image morphing approaches.

C. IMAGE DEFORMATION BASED ON FEATURE POINTS

All feature descriptions can be completely unified as a point collection for lines and curves, which are all made up of points. The deformation function of point morphing algorithm is completely based on the interpolation of discrete feature points. Seitz et al. [37] raised a view morphing technology, which can guarantee the authenticity of the generated image. General view morphing needs to know the projection matrix of the image, but it is very difficult to get the projection matrix directly from the image with the current technology. Though the image projection matrix is unknown in advance, researchers can interpolate two images from camera body motion to obtain the middle viewpoint images. And they do not need to know the 3D information, avoiding the 3D reconstruction of the image. Thus this method has a great application value. Lee et al. [38] extends the traditional deformation method based on two images and proposes an algorithm based on multiple image morphing. This is an effective image feature region selection and synthesis technology, accomplishing seamless blending or non-uniform blending of features across multiple image control selection areas. Igarash et al. [39] triangulated the input image, and solved a system of linear equations with unknown numbers equal to the sum of triangle vertices, minimizing image distortion. Li raised an image deformation approach based on asymmetric radial basis functions [40], and solved the problem of image boundary distortion caused by the global nature of symmetric radial basis functions. Su and Liu [41] proposed the feature mapping algorithm based on kohonen self-organizing to achieve 2D image deformation. It determined the neighborhood location function by using a simplified image model, and enriched the natural effect of image deformation with only a small number of control points. Ma [10] generalized the formation of MLS to the non-rigid case, and can produce more realistic deformation results than rigid model-based methods in handing the motion of coherent objects. These algorithms mentioned above are the most widely used morphing approaches, taking the shape outline information of the image into account, parting off the transform region effectively, reaching realistic deformation. However, most of them without fast implementation, this limits the effective application of image deformation. To address this problem, our algorithm based on point deformation manage to make image morphing realizing in real-time.

III. METHOD

In this article, the deformation is built on the user controlled point sets. Set \( \{x_i\}_{i=1}^n \) be a set of control points, let \( \{y_i\}_{i=1}^n \) be the corresponding deformed positions of the control points, \( x_i \) and \( y_i \) are both \( D \) dimensional column vectors, typically \( D = 2 \), \( n \) is the point numbers in the two sets. The deformation can be viewed as a function \( f \) mapping the points in the input image to those in the deformed image, and we formulate the function estimation as a vector-field interpolation satisfying the following three properties: (i) Interpolation: the points \( \{x_i\}_{i=1}^n \) should map directly to \( \{y_i\}_{i=1}^n \) under deformation; (ii) Smoothness: the function \( f \) should produce smooth deformations; (iii) Identity: \( f \) should be the identity function if the deformed handles \( \{y_i\}_{i=1}^n \) are the same as \( \{x_i\}_{i=1}^n \) (\( \forall i, p_i = q_i \Rightarrow f(x) = x \) with \( x \) being...
an arbitrary point in the image). For any point \( p_i \) in the image, \( q_i \) is the corresponding deformed position of \( p_i \). So \( p_i \) is be defined as original points, \( q_i \) is the deformed points of \( p_i \). We should distinguish \( x_i \) from \( p_i \), \( x_i \) is the control points, which gets the target point position by the handle, \( p_i \) is the ordinary pixel points to obtain the target point position by interpolation. These properties are very similar to those used in scattered data interpolation. Next, we put up a non-rigid deformation function \( f \) having these three properties with a closed-form solution. It should be noted that the proposed approach is not influenced by the dimension of our input data.

### A. PROBLEM FORMULATION

1) **MRLS**

A common mathematical optimization technique is Least Square, which can simulate data simply to minimize the square error of the data at known points. The minimized error is actually the square of the distance from each target point calculated by vector interpolation to each true target point. It is often used for curve fitting and can also be used for image warping. The mathematical expression of Least Square is that:

\[
\min_f \sum_{i=1}^{n} \| f(x_i) - y_i \|^2. \tag{1} \]

In practical deformation, the weight of each point is often different, some points may be more important, adding the weight is equivalent to controlling the effect of each point on the whole image, this is called “mobility”. That means a different transformation function can be obtained for each point in the image, for the position of each point relative to the control point is different, and its corresponding weight will be different. Moving Least Squares (MLS) is based on the Least Square for each point by adding the weight. The mathematical formulation is based on MLS as an interpolation function. The selection of weight function directly influences the convergence, stability and accuracy of the deformation results. And the construction and properties of the spline weight function are described in detail in the following subsection. For any point \( p \) in the image, MLS solves for a rigid-body transformation \( f_p(x) \) that minimizes a weighted least squares error functional:

\[
\min_{f_p} \sum_{i=1}^{n} w_i(p) \| f_p(x_i) - y_i \|^2. \tag{2} \]

The smoothness of its interpolation is determined by basis functions and weight functions (mainly controlled by the weight function). \( w_i(p) \) is defined as a non-negative weight function:

\[
w_i(p) = \| p - x_i \|^{-2\alpha}, \tag{3} \]

where \( \alpha \) controls the weight of each control point, while \( \| \cdot \| \) is the Euclidean distance metric. Each control point has an impact negatively related to their distance on each pixel. The farther the control point, the smaller the effect on each image pixel. And checkerboard distance or city distance can also be used as appropriate, as long as it meeting the definition of distance.

Equation (3) shows that there is a relationship among the weight function \( w_i(p) \), pixels \( p \), and control points \( x_i \). That is to say, it has a different weight function for changing deformation points, corresponding to different deformation functions. The value of pixel weight approaches infinity when \( p \) approaches the control point: \( \{x_i\}_{i=1}^{n} = \{y_i\}_{i=1}^{n} \), satisfying the identity. Each control point has an impact negatively related to their distance on each pixel. The farther the control point, the smaller the effect on each image pixel. To make the function optimal, it is necessary to minimize the error of (2). We obtain global deformation function \( f \) from a set of local functions defined as \( f(p) = f_p(p) \), it is continuously differentiable. Here we can conclude that this transformation function \( f \) is convergent everywhere except at the control point \( x_i \).

For traditional MLS methods, the transformation function is usually defined as a parameter mode, which named as rigid transformation. However, for coherent motions such as swinging flame and the shape of flowing water, the object shape deforms under certain nonrigid models, so that the warping principle of as-rigid-as-possible might not work well. Therefore, we introduce a regularization term and generalize this formulation to these non-rigid cases. Given any arbitrary point \( p \) in an image, we solve for the optimal transformation function \( f \) that minimizes a weighted regularized least squares error functional

\[
\min_{f_p} \sum_{i=1}^{n} w_i(p) \| f_p(x_i) - y_i \|^2 + \lambda \phi(f_p). \tag{4} \]

We can see that the solution form is similar to MLS except for the regularization term \( \phi(f_p) \), we use \( \lambda \) to control the balance of two aspects: the error effect on the image can be controlled, and the optimal transformation function can also be constrained. Such that the regular item offers our non-rigid deformation method a better transformation than rigid deformation of MLS. Since the error totally depends on the weight and the distance of the image points to the control points. The rigid transformation is hard to control without the regular item. With constraint of optimal transformation function, we take deformation function for minimizing the error of the weighted regularized least squares. While the constrained deformation function depends on the control points with a regularization term to smooth the deformation, which is different from the previous one.

The function \( f_p(\cdot) \) as a vector-valued function is built on the set \( \{x_i\}_{i=1}^{n} \), and it has \( y_i = f_p(x_i) \) for any correspondence \( (x_i, y_i) \). When point \( p_i \) approaches control point \( x_i \), \( w(p) \) approaches infinity, then for any point \( p \) the optimal transformation function \( f \) exists that \( y_i = f_p(x_i) \). We lie it in a specific functional space \( \mathcal{H} \), namely a Reproducing Kernel Hilbert Space (RKHS). When point \( p_i \) approaches control point \( x_i \), \( w(p) \) approaches infinity, then the optimal transformation function \( f \) for point \( p \) exists that \( y_i = f_p(x_i) \). When point \( p \) approaches control point \( x_i \), \( w(p) \) approaches infinity, then the optimal transformation function \( f \) for point
p exists that \( y_i = f_p(x_i) \). The functional \( \phi \) has the form
\[
\phi(f_p) = \| f_p \|_{\mathcal{H}}^2,
\]
where \( \| \cdot \|_{\mathcal{H}} \) denotes the norm of \( \mathcal{H} \). We will discuss the detailed forms of \( f_p \) and \( \| \cdot \|_{\mathcal{H}} \) later.

2) TPS
In most cases, the function \( f \) is an affine transformation, you can use a linear transformation matrix \( M \) and a translation transformation term \( T \). Finally the \( f \) in the two-dimensional image deformation can be written as follows:
\[
f_p(p) = pM + T,
\]

To produce a smooth mapping of fitting control point transformation, here we take the TPS function for constructing the model. TPS is a general spline tool. In the background of supervised learning, it can produce smooth mapping function, and it has free parameter and closed solution. Meanwhile, TPS model could be decomposed into a global affine transformation and a local bending function, which are controlled by affine matrix \( A \) and bending function \( g_p(p) \) respectively:
\[
f(p) = AP + g_p(p),
\]
and it has free parameter and closed solution. Meanwhile, TPS model could be decomposed into a global affine transformation and a local bending function, which are controlled by affine matrix \( A \) and bending function \( g_p(p) \) respectively:
\[
f_p(p) = pM + T,
\]

To produce a smooth mapping of fitting control point transformation, here we take the TPS function for constructing the model. TPS is a general spline tool. In the background of supervised learning, it can produce smooth mapping function, and it has free parameter and closed solution. Meanwhile, TPS model could be decomposed into a global affine transformation and a local bending function, which are controlled by affine matrix \( A \) and bending function \( g_p(p) \) respectively:
\[
f(p) = AP + g_p(p),
\]

where \( A \) defined as the global affine transformation. \( A \) is a \((D + 1) \times (D + 1)\) matrix, and \( \tilde{p} \) is a \((D + 1) \times 1\) matrix. \( \tilde{p} \) is homogeneous coordinates of all points, thus we set \( \tilde{p} = (pT, 1)^T \).

Radial basis function is defined as TPS kernel. And here we define \( K \) as a TPS kernel matrix.
\[
\varphi(r) = r^2 \log r.
\]

It can be noted that the smooth regularization term could be written as
\[
\| f_p \|_{\mathcal{H}}^2 = \sum_{i=1}^{N} \sum_{j=1}^{N} (K(p_i,p_j) \tilde{r}_{ij} \tilde{r}_{ij}),
\]

where \( C = (\tilde{r}_{11}, \ldots, \tilde{r}_{NN})^T \) is a bending coefficient matrix, its size is \( N \times (D + 1) \). \( K \in \mathbb{R}^{N \times N} \) is an entry of the kernel matrix, and here we set \( K_{ij} = K(p_i,p_j) \).

The habitude of TPS depends on the smooth transformation function. Meanwhile, \( w_i(p) \) approaches infinity when the control point \( p \) approaches pixel \( x_i \). Then for any control point \( p \) in the optimal transformation function, there is \( f(x_i) = y_i \), so \( f \) has interpolation property. Moreover, for all control points, \( \forall i, x_i = y_i \Rightarrow A = I \), and \( g_p(x) = 0 \). So we call that \( f \) is the identity transformation.

The evaluation point \( p \) influences the weights \( w_i \), and we take the TPS kernel with regularization technique to impose smoothness. Therefore, we name our approach as a moving regularized least squares optimization method based on the thin-plate spline minimization.

B. A CLOSED-FORM SOLUTION
After substituting (7) and (8) into (2), and selecting regularization parameter \( \lambda \), we can obtain the non-linear mapping \( f \) by minimizing the TPS energy function:
\[
E(A,C) = \| W^{1/2}(\tilde{Y} - XA^T - KC) \|^2 + \lambda tr(C^{T}KC),
\]

where \( \tilde{Y} \) and \( X \) are caculated as
\[
\tilde{Y} = (y_1, \ldots, y_N)^T,
\]

\[
\tilde{X} = (x_1, \ldots, x_N)^T,
\]

and the weight matrix \( W \) is a diagonal matrix with the \( i \)-th entry \( w_i(p) \) determined by (3). TPS kernel conveys the internal structural relationship in control points. We can obtain non-rigid deformation by combining with bending coefficient \( C \). The transformation function can be presented by matrix \( A \) and \( C \) here. Here we set
\[
\tilde{Y} = W^{1/2}Y,
\]

\[
\tilde{X} = W^{1/2}X.
\]

Because image points employ homogeneous coordinates, while the so-called bending energy is the standard TPS regularization term, which is the second smoothing term. In the process of image deformation, TPS has a special physical meaning. Meanwhile, it is independent of the linear section of the map \( f_p \).

In order to compute the TPS parameters \( A \) and \( C \), we have to use QR decomposition on matrix \( X \):
\[
\tilde{X} = [Q_1|Q_2] \begin{bmatrix} R \end{bmatrix}^T,
\]

where \( Q_1 \) and \( Q_2 \) are set as orthogonal matrix of \( N \times (D + 1) \) and \( N \times (N - D - 1) \), \( R \) is defined as an upper triangular matrix with the scale of \((D + 1) \times (D + 1)\). After substituting QR decomposition into (13), we can get:
\[
E(A, C) = \| Q_2^T(\tilde{Y} - W^{1/2}KQ_2\tilde{C}) \|^2 + \| Q_2^T\tilde{Y} - RA^T + Q_2^T W^{1/2}KQ_2\tilde{C} \|^2 + \lambda tr(\tilde{C}^{T}KQ_2^T\tilde{C}),
\]

where \( C \) means \( X^T = 0 \), we can divide the mapping into two parts: affine mapping and non-affine mapping. While \( C \) is a matrix with the scale of \((N - D - 1) \times (D + 1)\). By minimizing \( C \) and \( A \) in the energy equation (19), we can always obtain:
\[
C = Q_2\tilde{C} = Q_2(S^T S + \lambda T + \varepsilon I)^{-1} S^T Y, \quad (20)
\]

\[
A = (\tilde{Y} - W^{1/2}KC)^T Q_1 R^{-T}, \quad (21)
\]
where \( S = Q_2^T W^{1/2} K Q_2, \) \( T = Q_2^T K Q_2, \) with \( \varepsilon I \) controlling the stability of the numerical value.

We get a closed-form solution of \( f \) by substituting (20) and (21) into (7) and (8):

\[
f(p) = A\bar{p} + (K_p C)^T
\]  

(22)

with \( K(p, p_i) \) denoted as the \( i \)th element, \( K_p \) is defined as a \( 1 \times N \) row vector.

Each point could be transformed by solution (4) to get a deformation image. To demonstrate our MRLS-TPS clearly, here we summarize our approach in detail in Algorithm 1.

#### Algorithm 1: The Proposed Algorithm

**Input:** Image, parameters \( \alpha \) and \( \lambda \)

**Output:** Deformed image

1. Choose a set of control correspondences \( \{x_i, y_i\}_{i=1}^{n} \);
2. Approximate the image with a grid;
3. repeat
   4. Choose a vertex \( p \) on the grid;
   5. Compute the weight \( W \) by (3);
   6. Compute the vector \( K_p \) by (10);
   7. Compute the affine matrix \( A \) by (21);
   8. Compute the bending coefficient \( C \) by (20);
   9. Compute \( f \) at vertex \( p \) by using (22);
4. until all the vertices are computed;
11. The deformed image is generated by a bilinear interpolation of \( f(p) \).

### C. FAST IMPLEMENTATION

Using linear system (20) to solve the transformation \( f_p \) is the most complex step in our algorithm, which costs \( O(M^3) \) time complexity. Meanwhile, it may bring a serious problem to large values of \( M \). It is better to choose a suboptimal but faster method even when it is implementable. In the following paragraph, we will present a fast implementation, which is based on the similar kind of way as the subset of regressors approach.

The formula (22) relates to matrix operations of each point. In solution (20), the computation of matrix \( C \) is the most time-consuming matrix operation, because it involves a matrix inversion of \( N \times N \). So that the time complexity of our method is \( O(nM^2) \), with \( n \) defined as the amount of mesh vertices contained in the computation. This greatly reduces the amount of calculation and the time complexity, realizing fast implementation. In terms of image deformation, the amount of control points is generally very few, usually not more than 10, so we can quickly get matrix inversion. Meanwhile, \( X \) is fixed for QR decomposition of point sets generally. So that a number of steps could be completed in advance in solution (20). Therefore, matrix \( A \) and \( C \) can be rewritten as

\[
f(p) = (Y^T - T^T HW^{1/2}) B p + Y^T M.
\]  

(23)

And here we set \( E \) as

\[
E = Q_2 (S^T S + \lambda T + \varepsilon I)^{-1} T - Q_2^T.
\]  

(24)

Combining formula (20) with solution (21), we get \( B = Q_1 R^{-T}, H = E K, \) and \( M = E (K_p)^T \). As mentioned above, \( B, H, \) and \( M \) can be computed in advance, greatly improving the computational efficiency of our approach. Here is only a simple calculation of multiplication and addition left in the process of solving this model. Therefore, this algorithm has better computational efficiency in practical applications.

### D. IMPLEMENTATION DETAILS

**Parameter Setting:** \( \alpha \) and \( \lambda \). There are mainly two parameters setting in our algorithm: \( \alpha \) controls the weight of all control points, \( \lambda \) controls the transformational complexity. After simple calculations, we get the range of value \( \alpha \in [0.3, 1.5] \) and \( \lambda \in [100, 1000] \). Since we know the approximate range of \( \alpha \) and \( \lambda \), we divide the value range of \( \alpha \) and \( \lambda \) into five parts to get the specific value of \( \alpha \) and \( \lambda \). Therefore, we let \( \alpha \) and \( \lambda \) with five values to obtain the deformation results by our method MRLS-TPS respectively. We need to extract feature points possessing some sort of ‘intrinsic geometry’ before operator starts the deformation. Usually feature points are obtained from the shape or the surface of an object. Therefore, we take some algorithms of feature extraction and contour extraction for the feature point sets. While the performance depends on a coordinate system where points are presented. Data normalization is suitable to control for this situation. More specifically, the linear re-scaling of the correspondences is performed to make both control points and deformed control points have zero mean and unit variance. Therefore, we use manual extraction and automatic extraction to extract feature points. For irregular images, we use handles to control feature points. Meanwhile, we use a cascade regression tree based face alignment algorithm which has been implemented in dlib library to extract 68 accurate face landmark points.

### E. OVERVIEW OF THE APPROACH

Based on MLS, this paper proposed a method named MRLS-TPS to deal with non-rigid deformation problems in our daily life. The regularized parameter control the balance of two aspects: the error effect on the image can be controlled, and the optimal transformation function can also be constrained. And the TPS has a close-form solution decomposed into two parts: global linear affine transformation and local non-rigid deformation component controlled by coefficients. It can generate a smooth functional mapping for supervised learning, and it has no free parameters to manual tuning especially. We apply the regularized parameter and thin-plate spline into our non-rigid deformation model, generating our approach MRLS-TPS.

Figure 1 with 7 photos shows an overview of this deformation method. We choose this typical non-rigid deformation to show our framework of MRLS-TPS, which make eyes of the women rise naturally without influences on other parts of
FIGURE 1: Overview of the approach. The marked green and blue points in the original images are initial control points, and those in the deformed images are deformed control points. The marked red crosses in the top right corner image are the deformed feature points.

the face in the picture. In general, MRLS-TPS realizes the deformation following five steps:

- Firstly, inputting original images. The format of reading images supports .jpg,.png and so on. The picture can be in the form of statues, sketches, animation, oil paintings and other 2d photos or 3d models.
- Secondly, choosing control feature points. The feature points in these pictures can be freely selected by the handle control, and the number of feature points extraction is not limited.
- Thirdly, selecting target feature points. The selection of the target control points has a direct impact on the deformation result.
- Then, transformation. In other words, operators move the control feature points to the target pixels appropriately by using MRLS-TPS algorithm automatically.
- Finally, outputting deformed images. We provide the pixels of experimental images no less than $500 \times 500$, ensuring wrapping effects and deformation quality.

IV. EXPERIMENTAL RESULTS

In order to demonstrate the feasibility of our approach, we test the performance of our proposed method and verify the regularization validity on both 2D images and 3D surfaces. The experimental pictures can be human beings, plants, animals and any other items. In particular, for the face deformation, we can use the dlib library to automatically extract 68 feature points, and then control the deformation of the face features. Here we use the handle to extract feature points uniformly. Therefore, the experiments are completed from the following three aspects: (i) results on parameter setting; (ii) results on 2D images (including local deformation, Rigid deformation and Non-rigid deformation); (iii) results on 3D surface. In parameter setting case, we obtain the best values of parameter $\alpha$ and $\lambda$ through a large number of testing experiments, supporting for 2D morphing and 3D deformation experiments. In 2D experiment, we compare our deformation algorithm with other two state-of-the-art approaches: MLS and commercial software Adobe PhotoShop CS 6, they are both well-established. Besides, compared withMLS and commercial software Adobe PhotoShop CS 6, we present three deformation results and three runtime tables to demonstrate the better performance of our MRLS-TPS on local deformation, rigid deformation and non-rigid deformation respectively. In 3D models, we verify the performance of our algorithm. We should noted that the parameters of the last two approaches are all fixed. All experiments are performed on a laptop with 2.5-GHz Intel Core CPU, 8-GB memory, and MATLAB code. In general, our algorithm can always produce satisfying results.

A. RESULTS ON PARAMETER SETTING

1) step

From Figure 2 to Figure 4, we test the runtime of MLS and MRLS-TPS on 100 pictures with different steps respectively. X label represents step, and here we set step = 1, 2, 4, 5, 6, 8, 10, 12, 14, 16, 18, 20 respectively. Y label stands for the runtime of MLS and MRLS-TPS. The red circles and lines show the mean and standard deviation of runtime on MLS, the green circles and lines demonstrate the mean and
FIGURE 2: The mean and standard deviation of runtime on MLS and MRLS-TPS. The red is defined as the mean and standard deviation of runtime on MLS. And the green is the mean and standard deviation of runtime on MRLS-TPS respectively.

FIGURE 3: The mean and standard deviation of runtime on MLS.

standard deviation of runtime on MRLS-TPS. We uniform these pictures with the size of $550 \times 500$ to keep experiments uniform. In order to get obvious contrast results, we shift the green lines to the right by 1.5 and the red lines to the right by 0.5 from X label.

As can be seen from Figure 3, the runtime of MLS fluctuates greatly when the step value is smaller. While the runtime of our MRLS-TPS is hardly affected by the change value of the step, and the runtime of MRLS-TPS does not exceed 0.105S, which can be concluded from Figure 4. Finally, we focus Figure 3 and Figure 4 on Figure 2 for comparison.

Though the runtime of MLS float a lot at the beginning, the red lines tend to be steady when the step value is greater than 6. Compared with MLS, the runtime of MRLS-TPS does not seem to change, that means our MRLS-TPS is always faster than MLS. And different steps will not have an impact on our experiments. Therefore, we will set the step=10 in the following experiments in order to facilitate the experiment.

2) $\alpha$ and $\lambda$

As can be seen from Figure 5, we set four fixed points of the human eyes as deformation control points, ensuring the same position of all warping pixels in the process of each parameter setting experiment. In order to promise the fairness of the experiment, we only choose a photo here. And we still have experiments on human eyes, which are deformed obviously.

FIGURE 4: The mean and standard deviation of runtime on MRLS-TPS.

FIGURE 5: Parameter setting. From left to right: $\alpha = 0.3, \alpha = 0.5, \alpha = 0.8, \alpha = 1.2$ and $\alpha = 1.5$ respectively. From top to bottom: $\lambda = 100, \lambda = 300, \lambda = 500, \lambda = 700$ and $\lambda = 1000$ respectively.
TABLE 1: Runtime of local deformation.

<table>
<thead>
<tr>
<th>#ctrl pt</th>
<th>Animation</th>
<th>Photo</th>
<th>Sketch</th>
<th>Oil painting</th>
</tr>
</thead>
<tbody>
<tr>
<td>grid (step=10)</td>
<td>4</td>
<td>6</td>
<td>15</td>
<td>6</td>
</tr>
<tr>
<td>MLS(s)</td>
<td>0.5632</td>
<td>0.5824</td>
<td>0.7963</td>
<td>0.5451</td>
</tr>
<tr>
<td>MRLS-TPS(s)</td>
<td>0.3351</td>
<td>0.3458</td>
<td>0.4961</td>
<td>0.3372</td>
</tr>
</tbody>
</table>

It is convenient for us to select the best parameter values efficiently. Here we plan to deform the lines around eyes to make the western woman look more serious than before. The marked green and blue points in the original images are initial control points, and those in the deformed images are deformed control points. The marked red circles in the top right corner are the deformed feature points. A great number of experiments and statistics show that:

- From the first row to the fifth row, the smaller the value of parameter $\alpha$, the softer the deformation lines on the woman’s eyes in the images.
- From the first column to the fifth column, the greater the value of parameter $\lambda$, the smoother the deformation lines in the morphing area.
- However, the human eye cannot observe the deformation when the value of parameter $\alpha$ is too small, and the deformation will be very exaggerated when the value of parameter $\lambda$ is too large.
- Therefore, the values of parameter $\alpha$ and parameter $\lambda$ play an important role in our following 2D image morphing and 3D model deformation. Finally, in order to show the best deformation effect of the algorithm, we set the fixed values: $\alpha = 0.8$ and $\lambda = 500$.

**B. RESULTS ON 2D IMAGE**

1) Local deformation results on 2D image

As demonstrated in Figure 6, the first column shows the original images with all control points marked by some green points, while the rest three columns are the corresponding deformed results of MLS, our MRLS-TPS, and Adobe PhotoShop CS 6 respectively. It should be noticed that the initial control points here are marked by blue points to facilitate experimental comparison. For images in Figure 6, the deformed control points here are all marked by red circles. In terms of the first row (Animation), we complete our deformations by four points around the eyes. And we only choose two points near the bridge of the nose to control the deformation, and keep other two points closed to ears motionless. This operation on experiments generally deals with the case of fine adjustment. MLS begins to degenerate because it is a parameter model. Our method MRLS-TPS generates a natural deformation because it is robust. In contrast, Adobe PhotoShop CS 6 can not produce desired deformation, so the wrapped animation face presents asymptotic eyes. We further consider the second row (Photo). We plan to make the girl look softer by changing her eyebrows. In other words, we desire to get the local deformation without global morphing. However, MLS wraps the shape of eyes and glasses in the picture, while Adobe PhotoShop CS 6 can keep the frame of eyes and glasses the same, but it caused a strange deformation of the eyebrows. In the third row (Sketch), we mean to make the nose of western man shorter than before. Our algorithm deforms the nose without changing other parts in the sketch. However, MLS and Adobe PhotoShop CS 6 complete the deformation, causing varying degrees of distortion on the mouth at the same time. In the last row (Oil painting), the four images also demonstrate that our algorithm is capable of dealing with local strong deformation. In the second figure (the result of MLS, i.e.), we can find that the deformation lines around the mouth...
TABLE 2: Runtime of rigid deformation.

<table>
<thead>
<tr>
<th></th>
<th>Cat</th>
<th>Dog</th>
<th>Horse</th>
</tr>
</thead>
<tbody>
<tr>
<td>#ctrl pt</td>
<td>4</td>
<td>6</td>
<td>15</td>
</tr>
<tr>
<td>grid (step=10)</td>
<td>50 × 50</td>
<td>50 × 50</td>
<td>50 × 50</td>
</tr>
<tr>
<td>MLS(s)</td>
<td>0.7864</td>
<td>0.5941</td>
<td>0.7995</td>
</tr>
<tr>
<td>MRLS-TPS(s)</td>
<td>0.4821</td>
<td>0.3957</td>
<td>0.5934</td>
</tr>
</tbody>
</table>

do not look gentle, while in the fourth figure (the result of Adobe PhotoShop CS 6,i.e.), some deformed points of the mouth on the right become raised, making deformation result undesirable. Therefore, 16 pictures in Figure 1 prove that our MRLS-TPS is able to deal with both global deformation and local morphing. Especially, MRLS-TPS can control local deformation without the morphing of other parts, even when there are obstacles.

For the runtime of all experiments, we only compare MRLS-TPS with MLS, for the reason that it does not provide a tool to calculate the runtime of Adobe PhotoShop CS 6. Table 1 presents the runtime of local deformation algorithms of MLS and our MRLS-TPS. It can be seen that both methods perform quite fast and can be deformed in real-time, and our MRLS-TPS presents faster speed than MLS even when the deformation image is an animation, a sketch or an oil painting. Here we would like to emphasize two points from Table 1 to Table 2 and Table 3: firstly, in order to show good facial visual effects, the size of the pictures is cut into uniform size 500 × 550 in the first set of experiments, and we set the uniform size 500 × 500 in the last two set of images, this does not mean that our method can not handle large-scale 2D images or 3D models; secondly, setting the unified grid step value of 10 is convenient for us to facilitate the statistics, we will get satisfied experimental results as well as we set fixed grid step if we change the value.

2) Rigid deformation results on 2D image

Three sets of experimental images are shown in Figure 7, demonstrating that our algorithm can achieve rigid deformation no worse than MLS and Adobe PhotoShop CS 6. Like Figure 6, the first column shows the original images with all control points marked by some green points, while the rest three columns are the corresponding deformed results of MLS, our MRLS-TPS and Adobe PhotoShop CS 6 respectively with the initial blue control points and the deformed red control points. For images in the first row (Cat), we mean to make the ears of the cat smaller, our MRLS-TPS can generate a natural and symmetric deformation result. In contrast, the MLS has really achieved experimental results, but it changes the basic shape of its ears; while the Adobe PhotoShop CS 6 makes the cats’ ears asymmetric obviously, producing ears with two different sizes. We further consider animations in the second row (Dog), all of the three methods can swing the dog’s tail well when there are enough control points on the deformation area. While the result of our MRLS-TPS is more living since the smooth morphing lines vividly express the shape of the dog’s tail. Finally, sketches in the last row (Horse) also show that our algorithm is able to complete rigid deformation well. We can see that in the second figure (the results of MLS ) and the third figure (the results of Adobe PhotoShop CS 6) , which both achieve the natural extension of the horse’s legs. Therefore, our MRLS-TPS can deal with rigid deformation as well as MLS and Adobe PhotoShop CS 6. Besides, MRLS-TPS performs the deformation better than MLS and Adobe PhotoShop CS 6 under certain circumstances.

Table 2 presents the runtime of rigid deformation approaches on both MLS and MRLS-TPS. The closed-form solution just involves simple matrix operations of addition and multiplication, though we deal with rigid deformation problems, the application of non-rigid model in MRLS-TPS will not bring efficiency decreases. The runtime of MLS and MRLS-TPS shows that our non-rigid deformation methods MRLS-TPS performs no worse than the rigid deformation algorithm MLS.

3) Non-rigid deformation results on 2D image

Unlike Figure 7, this section validates our algorithm from the overall deformation effect in the following four examples in
TABLE 3: Runtime of non-rigid deformation.

<table>
<thead>
<tr>
<th></th>
<th>Candle</th>
<th>Fire</th>
<th>Petal</th>
<th>Moon</th>
</tr>
</thead>
<tbody>
<tr>
<td>#ctrl pt</td>
<td>2</td>
<td>1</td>
<td>11</td>
<td>34</td>
</tr>
<tr>
<td>grid (step=10)</td>
<td>50 × 50</td>
<td>50 × 50</td>
<td>50 × 50</td>
<td>50 × 50</td>
</tr>
<tr>
<td>MLS(s)</td>
<td>0.3864</td>
<td>0.3832</td>
<td>0.7645</td>
<td>0.9624</td>
</tr>
<tr>
<td>MRLS-TPS(s)</td>
<td>0.2822</td>
<td>0.2651</td>
<td>0.5449</td>
<td>0.7458</td>
</tr>
</tbody>
</table>

Figure 8. Similarly the first column demonstrates the original images with the green control points, and the rest three columns are the corresponding deformed results of MLS, our MRLS-TPS and Adobe PhotoShop CS 6 respectively with initial blue control points and deformed red control points. To begin with the first row (Candle), we try to simulate the shape of the flame when the candle is blown by the left wind in three ways. It is clear that our MRLS-TPS produces a soft and realistic deformation. However, the deformed lines of MLS are too curved, and the morphing result of Adobe PhotoShop CS 6 is too stiff. Compared with the second row (Fire), we make the flame tip of the fire lean to the left. We get a natural morphing image by our MRLS-TPS. However, the deformed lines of MLS in the second picture becomes a bit thinner and longer, and the morphed flame of Adobe PhotoShop CS 6 in the fourth image looks strange. Considering the third row (Petal), we select five points to control the deformation, and make other six points motionless to get a natural bigger petal. This generally deals with the ease of fine adjustment. While the deformed lines of Adobe PhotoShop CS 6 are too straight to get a better result. What’s worse, MLS starts to degenerate as a parameter model, leading to a white hole in the second column. Finally, the deformation examples in the last row (Moon) also demonstrate that our method can deal with non-rigid deformation better than MLS and Adobe PhotoShop CS 6. The same as the third row; MLS still generates dark hole in the process of deformation, and Adobe PhotoShop CS 6 is failed to make deformed lines soft and smooth.

Table 3 calculates the runtime of non-rigid deformation algorithms on MLS and MRLS-TPS. Though both two methods can complete deformation quite fast in no more than 1 second, out MRLS-TPS perform better than MLS no matter in deformation results or morphing runtime. Especially, we can conclude that our MRLS-TPS shows better performance than MLS, no matter when we deal with a few or a large number of increasing control points in the process of morphing.

C. RESULTS ON 3D SURFACE

Since the dimension of input data will not influence the result of our deformation method MRLS-TPS, we experiment on 3D models to demonstrate the performance of our method. Like 2D cases, we also set the same parameters here. We test MRLS-TPS on the surface of a 3D sculpture with 538,338 points. After a large number of experiments, we conclude that the average runtime costs 1.3429 + 0.0169 seconds, the precomputed time takes 1.3429 seconds, the deformation time spends 0.0169 seconds. Our algorithm performs quite fast, because the original control points are all fixed and only very few deformed control points have to be adjusted. The deformed images are shown in Figure 9: we obtain natural deformations of our method for 3D deformation on a statue without complex operations. For the sculpture in left column, we stretch her left ear and deform her mouth, it seems that we make her looks like elves with a smile face. For the sculpture in middle column, we close her eyes and open her mouth a little bigger than before by controlling deformed lines on the eyes and lips, presenting a cry facial expression with eyes closed and mouth opened; in the right figure, we make the nose shorter and rise the upper lip, giving the sculpture an expression of sluggish. Therefore, our MRLS-TPS shows good performance on 3D deformation results.
stage, several directions for the future work are possible:

1. Automatic feature extraction. In this paper, we just realize automatic feature extraction for face deformation by using a cascade regression tree based face alignment algorithm, which has been implemented in dlib library to extract 68 accurate face landmark points. And it is still a difficult work to get automatic feature points for different contents of images. Therefore, we are dedicated to the research of automatic feature extraction for all types of deformations in the future work.

2. Large scale programs. Currently, MRLS-TPS is mainly used for image morphing in MATLAB or Android application. As future work, MRLS-TPS will be applied in other large scale programs to demonstrate its usability.

V. CONCLUSIONS AND SUGGESTIONS FOR THE FUTURE WORK

Within this article, we present a vector-field interpolation method called MRLS-TPS. Our MRLS-TPS can deal with morphing problems of rigid deformations and non-rigid deformations well in our daily life. In our algorithm, TPS parameterizes the spatial mapping related that can be decomposed into linear and nonlinear components clearly.

Moreover, TPS minimizes the bending energy with a specific physical explanation. It may be beneficial for image deformation under the circumstance of non-rigid motions. And the regularized parameters in our approach perform better local deformation lines than MLS and Adobe photoshop CS 2016 with the stability of global deformation. The results from Figure 3 to Figure 6 aboved show us the proposed method can generate natural non-rigid deformation results, especially in the case of objects with coherent motions. Furthermore, we can conclude from Table 1 to Table 3 that it is very fast and can be performed in real-time, without limiting the image categories and types of deformation objects.

Although we have done a lot of hard work at the current stage, several directions for the future work are possible:

- Automatic feature extraction. In this paper, we just realize automatic feature extraction for face deformation by using a cascade regression tree based face alignment algorithm, which has been implemented in dlib library to extract 68 accurate face landmark points. And it is still a difficult work to get automatic feature points for different contents of images. Therefore, we are dedicated to the research of automatic feature extraction for all types of deformations in the future work.

- Large dataset. The experimental data set in this paper consists of animations, sketches, photos, oil paintings and statues from Google Chrome. The authors will further study to use GAN or many other deep learning methods to train big data set by using our MRLS-TPS.

REFERENCES


HUABING ZHOU received the Ph.D. degree in control science and engineering from the Huazhong University of Science and technology, Wuhan, China, in 2012. He is currently an Associate Professor with the School of Computer Science and Engineering, Wuhan Institute of Technology, Wuhan. His research interests include remote sensing image analysis, computer vision, and intelligent robotics.

YUYU KUANG was born in ChongQing, China in 1993. She received the B.S degree in computer science and technology from the School of Computer Science and Engineering, Wuhan Institute of Technology, Wuhan, China, in 2016. She is currently studying computer vision, image process and machine learning for the M.S. degree in the School of Computer Science and Engineering, Wuhan Institute of Technology, Wuhan, China.

ZHENGHONG YU received the B.S. and M.S. degree in computer science from Wuhan Institute of Technology, Wuhan, China, in 2005 and 2008 respectively. He received the Ph.D. degree in control science and engineering from Huazhong University of Science and Technology in 2014. He is currently an assistant professor in the College of Mechanical and Electrical Engineering, Guangdong Polytechnic of Science and Technology, Zhuhai, China. Meanwhile, he has been invited as a Guest Professor at Fujian Agriculture and Forest University. His research interests include computer vision, intelligent robot and agriculture automation.

SHIQIANG REN was born in Fangxian, Hubei, China in 1991. He received B.S degree in computer science and technology from Wuhan Institute of Technology in 2015 and M.S degree in software engineering from Wuhan Institute of Technology, in 2018. He is currently working for TCL Group Industrial Research Institute, Wuhan research and Development Center. His research interests include image processing and meachine learning.

YANDUO ZHANG received the Ph.D. degree in School of Astronautics in Harbin Institute of Technology, China, in 1999. He is currently a Professor in the Hu Bei Key Laboratory of Intelligent Robot, Wuhan, China. His main research interests include artificial intelligence, intelligent robotics, and computer vision.

TAO LU received the B.S and M.S degrees from School of Computer Science and Engineering, Wuhan Institute of Technology, Wuhan, China in 2003 and in 2008 respectively, and the Ph.D. degree from National Engineering Research Center For Multimedia Software, Wuhan University, in 2013. Now he is an associate professor in School of Computer Science and Engineering, Wuhan Institute of Technology and a research member in Hubi Provincial Key Laboratory of Intelligent Robot. He was a Post-Doc in Department of Electrical and Computer Engineering, Texas A&M University from 2015 to 2017. His research interests include image processing, computer vision and artificial intelligence.
JAYI MA received the B.S from the Department of Mathematics and the PH.D. degree from the School of Automation, Huazhong University of Science and Technology, Wuhan, China, in 2008 and 2014, respectively. From 2012 to 2013, He was an Exchange Student with the Department of Statistics of California at Los Angeles, Los Angeles, CA, USA. From 2014 to 2015, he was a Post-Doctoral Researcher with the Electronic Information School, Wuhan University, Wuhan, where he is currently an Associate Professor. He has authored and co-authored more than 80 scientific articles. His research interests include computer vision, machine learning and pattern recognition.

* * *