UMEÅ UNIVERSITY

A NEW HANDS-FREE FACE TO FACE VIDEO COMMUNICATION METHOD

Profile based frontal face video reconstruction

Songyu LI

Supervisor: Ulrik Söderström
Examiner: John Berge

Master thesis, 30 ects

Master's Programme in Robotics and Control, 120 ects

VT 2018

A New Hands-free Face to Face Video Communication Method

Profile based frontal face video reconstruction

LI,Songyu

LI,Songyu
VT 2018
Master Thesis, 30 ects
Supervisor: Ulrik Söderström
Examiner: John Berge
Master's Programme in Robotics and Control, 120 ects

Acknowledgements

My deepest gratitude goes first and formost to my supervisor Ulrik Söderström for all the advice, guidance and help he has provided me during my thesis working time. Without his help, I hadn't been able to pursue my project.

Second, my thanks would go to my examiner John Berge for providing academic supports during the work.

I would also like to acknowledge my parents, they take good care of my life and gave me a lot of encouragement when I was working at this thesis. I feel much grateful and heartily owe my achievement to them.

Finally, thanks to everyone who gives me a hand with my thesis work, especially to my friends for their help with the result evaluation works.

LI,Songyu
Umeå, Sweden, 2018

Abstract

This thesis proposes a method to reconstruct a frontal facial video based on encoding done with the facial profile of another video sequence. The reconstructed facial video will have the similar facial expression changes as the changes in the profile video. First, the profiles for both the reference video and for the test video are captured by edge detection. Then, asymmetrical principal component analysis is used to model the correspondence between the profile and the frontal face. This allows encoding from a profile and decoding of the frontal face of another video. Another solution is to use dynamic time warping to match the profiles and select the best matching corresponding frontal face frame for reconstruction. With this method, we can reconstructed the test frontal video to make it have the similar changing in facial expressions as the reference video. To improve the quality of the result video, Local Linear Embedding is used to give the result video a smoother transition between frames.

Contents

1 Introduction

With the development of the Internet and the popularization of electronic devices such as mobile phones and computers, face-to-face video communication has become an increasingly popular part of people's daily life. Compared with voice communication or text communication, face-to-face video communication can obviously provide better user experience. Face-to-face video communication enables the user to see the changes of the communicator's expression directly, and can more intuitively feel the emotions of the communicator on the other side of the screen. But at the same time, face-to-face video communication also has some deficiencies, will be limited by equipment and use of the occasion. For example, when using a computer or fixed camera for face-to-face video communication, the device cannot move flexibly with the user, which greatly limits the user's range of activity. And when using handheld devices such as mobile phones for face-to-face video communication, the hands will be occupied so that users have to stop the work they are doing. In cases where people cannot stop working, such as when they are cooking or driving, they must continue their work and their attention cannot be disturbed, they have to stop face to face video communicating.

For facial performance capture, already have the precedent for using a headset camera to film user's frontal face [1]. But a headset sideview camera could be a better choice. Capturing the side view gives a huge benefit in usability since the user can wear a camera on head that films the side view of the face instead of the frontal view of the face, the differences are visualized in Figure 1, a frontal face setting camera may affect the user, such as attracting the user's attention or blocking the user's view.

Figure 1: The face captured by cameras filming the frontal and side view of a person.

For communication purposes, people want to see the frontal view of a face. But from a user point of view, wear a camera that films the side of the face instead of the front of the face can turns video capturing into something which is performed without any inconvenience such as a camera in front of the face would mean.

Therefore, in this thesis, a method is proposed to reconstruct the frontal face video based on the profile of the side face. Through the obtained video of side face and the existing video template of frontal face, reconstruct the frontal face video template to get a new video can follow the same frontal facial performanc as side video recording.

Chapter 2 describes related theories include edge detection, Asymmetrical Principal Component Analysis (APCA), Dynamic Time Warping (DTW) and Locally Linear Embedding (LLE). Chapter 3 explains how these theories are used for video reconstruction and for quality improvement. Chapter 4 show the results from practical experiments, the work is discussed in chapter 7 and chapter 8 is about the future works.

2 Correlation Theories

2.1 Edge Detection

The contour of a side view face can be extracted out from the background through edge detection. Several commonly used edge detection operators include Roberts cross operator, Prewitt operator, Sobel operator, laplace operator, Canny operator, Kirsch operator, etc.

2.1.1 Roberts cross operator

The Roberts cross operator was initially proposed by Lawrence Roberts in 1963 [2]. It use derivative operators to find edges. The operation principle about how Roberts cross operator perform edge detection can be expressed by the following equation:

$$G(x,y) = \sqrt{[f(x,y) - f(x+1,y+1)]^2 + [f(x+1,y) - f(x,y+1)]^2} \tag{2.1.1}$$

Where $f(x,y)$ is a point in the original image and $G(x,y)$ is the gradient.

For easy to understand, here we give a template to represent a part of the original image:

$$\begin{bmatrix} Z_1 & Z_2 & Z_3 \\ Z_4 & Z_5 & Z_6 \\ Z_7 & Z_8 & Z_9 \end{bmatrix}$$

We also have two kernels:

$$\begin{bmatrix} +1 & 0 \\ 0 & -1 \end{bmatrix} \textit{and} \begin{bmatrix} 0 & +1 \\ -1 & 0 \end{bmatrix}$$

Let the center point Z_5 be the dealt point, $G_x = Z_5 - Z_9$ be a point in an image formed by convolving with the first kernel and $G_y = Z_6 - Z_8$ be a point in an image formed by convolving with the second kernel, then the graident became to:

$$G(_{Z5}) = \sqrt{(Z_5 - Z_9)^2 + (Z_6 - Z_8)^2} \tag{2.1.2}$$

Then check if G is greater than the preset threshold to determine whether it is an edge point.

2.1.2 Prewitt operator

J.M.S. Prewitt presented the idea of Prowitt operator in 1970 [3]. This operator uses two 3×3 kernels to convolve the original image to calculate approximations of the derivatives, one for horizontal changes and one for vertical changes.

The two kernels are:

$$\begin{bmatrix} +1 & 0 & -1 \\ +1 & 0 & -1 \\ +1 & 0 & -1 \end{bmatrix}$$

for horizontal and

$$\begin{bmatrix} +1 & +1 & +1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

for vertical.

If we define A as the source image and G_x be a image formed by convolving with the horizontal kernel and G_y be a image formed by convolving with the vertical kernel, which are:

$$G_x = \begin{bmatrix} +1 & 0 & -1 \\ +1 & 0 & -1 \\ +1 & 0 & -1 \end{bmatrix} * A$$

and

$$G_y = \begin{bmatrix} +1 & +1 & +1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} * A$$

Then the gradient can be defined as:

$$G = \sqrt{G_x^2 + G_y^2} \tag{2.1.3}$$

2.1.3 Sobel operator

Sobel and Feldman presented the idea of an "Isotropic 3×3 Image Gradient Operator" in 1968, which be called Sobel operator [4]. The Sobel operator can be regarded as an improvement based on the Prewitt operator [5], this operator also uses two 3×3 kernels to convolve the original image to calculate approximations of the derivatives, one for horizontal changes and one for vertical changes.

The two kernels are:

$$\begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix}$$

for horizontal and

$$\begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

for vertical.

If we define A as the source image and G_x be a image formed by convolving with the horizontal kernel and G_y be a image formed by convolving with the vertical kernel, which are:

$$G_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * A$$

and

$$G_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * A$$

Then the gradient can be defined as:

$$G = \sqrt{G_x^2 + G_y^2} \tag{2.1.4}$$

2.1.4 Laplacian operator

The Laplace operator is a second derivative operator, it uses a 3×3 kernels to convolve the original image to calculate approximations of the derivatives [6].

$$\begin{bmatrix} 0 & +1 & 0 \\ +1 & -4 & +1 \\ 0 & +1 & 0 \end{bmatrix}$$

how Laplace operator perform edge detection can be expressed by the following equation:

$$G(x,y) = [f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1)] - 4f(x,y) \tag{2.1.5}$$

Where $f(x,y)$ is a point in the original image and $G(x,y)$ is the gradient.

2.1.5 Canny operator

The Canny edge detection operator is a multistage edge detection algorithm developed by John F. Canny in 1986 [7]. Canny edge detection is no longer a single intensity gradients calculation, but contains a series of processes include image smoothing by Gaussian filter to remove the noise, get the intensity gradients of the image, non-maximum suppression, double threshold and edge tracking by hysteresis.

Image noise may affect the edge detection result, so filter out the noise can prevent false detection caused by noise to improve the result. Use a Gaussian filter to convolve with the source image can smooth the image to reduce the effects of obvious noise on the edge detector.

Next is to give the gradient magnitude of the filtered image, the Canny algorithm uses four filters to detect horizontal, vertical and diagonal edges in the filtered image, here it can use the first derivative edge detection operator like Roberts, Prewitt, or Sobel to do the first derivative in the horizontal direction and the vertical direction. After we finish this step we can have the edge gradient and direction:

$$G = \sqrt{G_x^2 + G_y^2} \tag{2.1.6}$$

$$\Theta = atan2(G_y, G_x) \tag{2.1.7}$$

The edge direction angle is rounded to one of four angles representing vertical, horizontal and the two diagonals $(0^\circ, 45^\circ, 90^\circ and 135^\circ)$.

Non-maximum suppression can sharp the blurred edges, it can retain the maximum gradient strength on each pixel point, and delete other values. For each pixel, it do the following:

1 Compare the gradient strength of the current pixel with the gradient strength of the other pixels in its positive and negative gradient directions.

2 If the gradient strength of the current pixel is the largest in the comparison with the other pixels in the mask with the same direction, the value will be preserved. otherwise, will be set to 0.

There are still many noise points in the image after non-maximal suppression. The Canny algorithm uses a technique called double threshold. That is to say, a high threshold bound and a low threshold bound are selected. If the gradient value of an edge pixel in the image are greater than the threshold upper bound,it is marked as a strong edge pixel. if the gradient value of this pixel is smaller than the low threshold value, it will be suppressed. If the gradient value falls between the high threshold value and the low threshold value, it is marked as a weak edge pixel.

Now the strong edge pixels should be considered as true edges which will certainly be involved in the final edge image, but for weak edge pixels, there will be some debate. These weak edge pixels can be real edges, or they can be caused by noise or color changes.

For accurate results, the weak edge points caused by noise or color changes should be removed. It is generally believed that a weak edge pixel caused from true edges will be connected to a strong edge pixel, while a weak edge point caused by noise are unconnected. The so-called hysteresis edge tracking algorithm checks the 8 connected neighborhood pixels of a weak edge pixel. As long as there is one strong edge pixel exists, this weak edge point is considered to be a real edge pixel.

2.1.6 Kirsch operator

Kirsch operator is an edge detection algorithm proposed by r.k. irsch in 1971 [8]. It is a non-linear edge detector that finds the maximum edge gradient strength in 8 predetermined directions. This operator uses eight different 3×3 kernels to convolve the source image, each kernel for one direction:

$$g(1)=\begin{bmatrix}+5 & +5 & +5\\ -3 & 0 & -3\\ -3 & -3 & -3\end{bmatrix} g(2)=\begin{bmatrix}+5 & +5 & -3\\ +5 & 0 & -3\\ -3 & -3 & -3\end{bmatrix} g(3)=\begin{bmatrix}+5 & -3 & -3\\ +5 & 0 & -3\\ +5 & -3 & -3\end{bmatrix} g(4)=\begin{bmatrix}-3 & -3 & -3\\ +5 & 0 & -3\\ +5 & +5 & -3\end{bmatrix}$$

$$g(5)=\begin{bmatrix}-3 & -3 & -3\\ -3 & 0 & -3\\ +5 & +5 & +5\end{bmatrix} g(6)=\begin{bmatrix}-3 & -3 & -3\\ -3 & 0 & +5\\ -3 & +5 & +5\end{bmatrix} g(7)=\begin{bmatrix}-3 & -3 & +5\\ -3 & 0 & +5\\ -3 & -3 & +5\end{bmatrix} g(8)=\begin{bmatrix}-3 & +5 & +5\\ -3 & 0 & +5\\ -3 & -3 & -3\end{bmatrix}$$

The edge gradient of the pixel is calculated as the maximum gradient across all directions:

$$G(x,y) = max_{z=1,\ldots,8} \sum_{i=-1}^{1} \sum_{j=-1}^{1} g_{ij}(z) \cdot f(x+i, y+j) \tag{2.1.8}$$

Where $f(x,y)$ is a point in the original image and $G(x,y)$ is the gradient.

The following Figure 2 shows the edge detection results of Roberts cross operator, Prewitt operator, Sobel operator, laplace operator, Canny operator, Kirsch operator applied to a color pthtograph of a side view face.

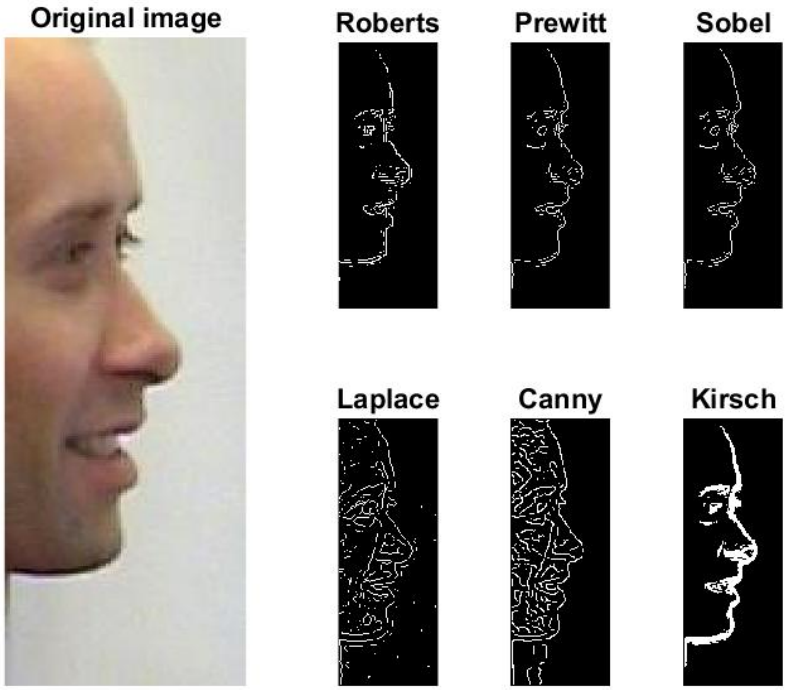

Figure 2: Edge detection results of Roberts cross operator(threshold:0.03), Prewitt operator(threshold:0.03), Sobel operator(threshold:0.03), laplace operator(threshold:0.001), Canny operator(threshold:0.03), Kirsch operator(threshold:0.05) applied to a color pthtograph of a side view face.

Roberts cross operator(threshold:0.03), Prewitt operator(threshold:0.03), Sobel operator(threshold:0.03), laplace operator(threshold:0.001), Canny operator(threshold:0.03), Kirsch operator(threshold:0.05)

2.2 Asymmetrical Principal Component Analysis (APCA)

2.2.1 Principal Component Analysis (PCA)

PCA was invented in 1901 by Karl Pearson [9], Is one of the most important methods of dimensionality reduction. The main idea of PCA is to find out the most important aspects in the data and replace the original data with these important data. This is done by retaining lower order principal components and ignoring higher order principal components.

Assuming there is an n-dimensional dataset that contains m pieces of data, set it as matrix $X'_{n\times m}$:

$$X' = [x'_1, x'_2, ..., x'_i, ..., x'_m] \tag{2.2.1}$$

Zero mean each colume of X' matrix (the sample mean of each column has been shifted to zero).

$$X = [x_1, x_2, ..., x_i, ..., x_m], x_i = (x'_i - x'_o) \tag{2.2.2}$$

Where x'_o is the mean of each column of X'.

$$x'_o = \frac{1}{m}\sum_{i=1}^{m} x'_i \tag{2.2.3}$$

Then find the covariance matrix of X:

$$C = \frac{1}{m}XX^T \tag{2.2.4}$$

Calculate the eigenvalues and corresponding eigenvectors of the covariance matrix C. Diagonalize the covariance matrix C: All the elements except the diagonal are reduced to zero, and the elements are arranged from big to small along the diagonal.

Compute the matrix P of eigenvectors which diagonalizes the covariance matrix C. Set $Y = PX$, matrix P is a $n \times n$ matrix represent the n eigenvectors of the covariance matrix C and D is the covariance matrix for Y, here we have:

$$\begin{aligned} D &= \frac{1}{m}YY^T \\ &= \frac{1}{m}(PX)(PX)^T \\ &= \frac{1}{m}PXX^TP^T \\ &= P(\frac{1}{m}XX^T)P^T \\ &= PCP^T \end{aligned} \tag{2.2.5}$$

Where D is the diagonal matrix of eigenvalues of C. Matrix P is the matrix consisting of the set of all eigenvectors of C, one eigenvector per column.

If we sort the columns of the eigenvector matrix P and eigenvalue matrix D in order of decreasing eigenvalue and maintain the correct pairings between the columns in each matrix, then save the first L columns of P as the $n \times L$ matrix W and multiply it with the original data matrix X, the result is the data after dimensionality reduction to L dimension.

2.2.2 Asymmetrical Principal Component Analysis (APCA) Video Coding

Based on principal component analysis, Söderström and Li proposed a new method for video coding called Asymmetrical Principal Component Analysis(APCA), which can use one part of a frame for encoding and decode the entire frame [10].

If the side view and the frontal view come from the same video and there is a correspondence between the facial features in the two views, the side view can be used for encoding and the frontal view be used for decoding. To implement this method, an eigenspace for the side view should be constructed at the beginning as:

$$\phi_j^s = \sum_i b_{ij}^s (I_i^s - I_o^s) \tag{2.2.6}$$

where I_i^s are side views and I_o^s is the mean of the side view. b_{ij}^s are eigenvalues from the eigenvectors of the covariance matrix

$$(I_i^s - I_o^s)^T (I_j^s - I_o^s)$$

Encoding of side view is performed as:

$$\alpha_j^s = (I^s - I_o^s)^T \phi_j^s \tag{2.2.7}$$

where α_j^s are coefficients extracted using information from the side view I^s.

Then use eigenvalues b_{ij}^{fr} from the eigenvectors of the covariance matrix $(I_i^{fr} - I_o^{fr})^T (I_j^{fr} - I_o^{fr})$ and frontal view of the frames to make a space consisted by pseudo principal components:

$$\phi_j^{p_{fr}} = \sum_i b_{ij}^{fr} (I_i^{fr} - I_o^{fr}) \tag{2.2.8}$$

where I^{fr} is the frontal view of the frames and I_o^{fr} is the mean image of the frontal view.

The coefficients from encoding with the side view are combined with this space for frontal view decoding.

$$\hat{I}^{fr} = I_o^{fr} + \sum_{j=1}^{M} \alpha_j^s \phi_j^{p_{fr}} \tag{2.2.9}$$

Thus, a decoded video of the frontal view can be reconstructed based on its corresponding side view video.

2.3 Dynamic Time Warping (DTW)

2.3.1 The fundamentals

Dynamic Time Warping (DTW) is a method to measure the similarity between two temporal sequences, which is mainly used in the field of speech recognition to identify whether two speech segments represent the same word [11]. This algorithm was proposed by Japanese scholar ITAKURA [12], initially for isolated words recognition, and its essence is a method to measure the similarity of two time series with different lengths.

In a time series, the lengths of two time series that need to be compared for similarity may not be equal. For example, in the field of speech recognition, the speech speed of different people is different, and the pronunciation speed of different phonemes in the same word is different. In addition, different time series may only have displacement on the time axis, that is, in the case of reducing displacement, the two time series are consistent. In these complex cases, the distance (or similarity) between two time series cannot be effectively calculated using the traditional Euclidean distance.

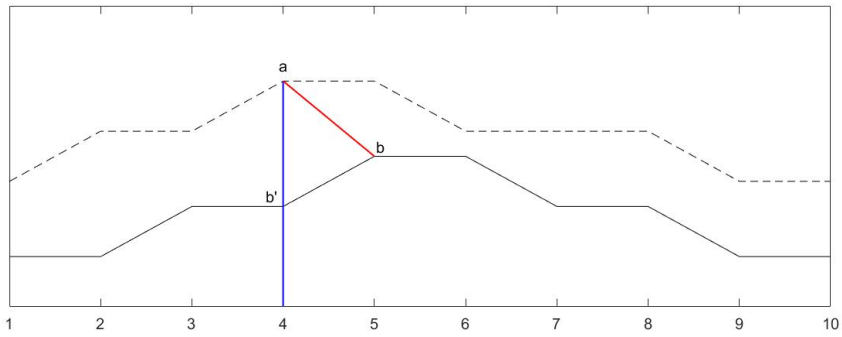

Figure 3: Two sequences comparison without dynamic time warping

For example, as shown in Figure 3, the real line and the dotted line are two similar time series (pulled apart on the Y-axis for observation).

You can see in figure that their overall waveform shape is similar, but it's not aligned on the timeline. For example, at the fourth time point, point a of the real line waveform will correspond to point b' of the dotted line waveform, so it is obviously not reliable to calculate the similarity by comparing distances. Because obviously, point a of the real line is correct for point b of the dotted line.

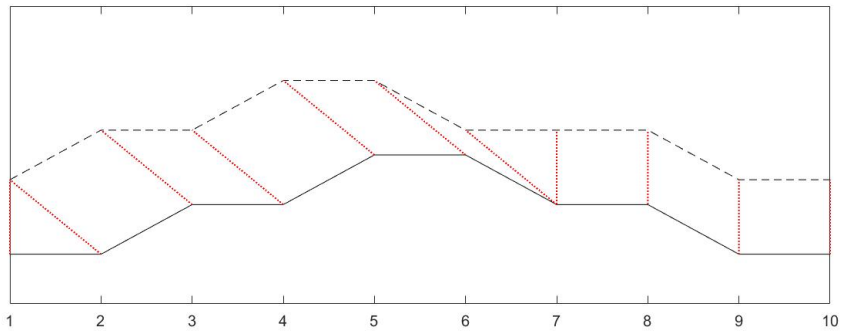

Figure 4: An example of the utility of dynamic time warping.

As shown in the Figure 4 above, the top and bottom lines represent two time series, and the dotted lines between the time series represent similar points between the two time series. That is, for the most part, the two sequences have very similar shapes overall, but these shapes are not aligned on the X-axis. So before we compare their similarity, we need to place one (or two) of the sequences under the timeline warping for better alignment.

DTW is an effective way to achieve this distortion.

DTW calculates the sum of the distances between all these similarity points, called Warp Path Distance, to measure the similarity between two time series.

2.3.2 Computational Method

Dynamic time normalization is a typical optimization problem. It uses the time planning function satisfying certain conditions to describe the time correspondence between the input sample and the reference template, and solves the regularization function corresponding to the minimum accumulated distance between two templates.

Suppose two time series are given, which are sample series Q and test series C, and the lengths of them are n and m respectively.

$$Q = q_1, q_2, \ldots, q_i, \ldots, q_n \tag{2.3.1}$$

$$C = c_1, c_2, \ldots, c_j, \ldots, c_n \tag{2.3.2}$$

Then we construct an n-by-m matrix D, the $D(i,j)$ element of the matrix contains the distance $d(q_i,c_j)$ between the two points q_i and c_j.

Here we use the Euclidean distance $d(q_i,c_j) = (q_i\text{-}c_j)^2$ and then can calculate out the distance matrix D :

$$D = \begin{bmatrix} d(q_n,c_1) & d(q_n,c_2) & \cdots & d(q_n,c_m) \\ \vdots & \vdots & \ddots & \vdots \\ d(q_2,c_1) & d(q_2,c_2) & \cdots & d(q_2,c_m) \\ d(q_1,c_1) & d(q_1,c_2) & \cdots & d(q_1,c_m) \end{bmatrix} \tag{2.3.3}$$

Determine the warping path: Find a path that passes through a number of matrix elements in the matrix, and the matrix elements that pass through the path are the aligned elements calculated for the two time series, these elements defines a mapping between Q and C.

We define this path as the warping path, expressed as W. The k^{th} element of W is defined as $w_k = (i,j)_k$ and then we can hve the warping path:

$$W = w_1, w_2, \ldots, w_k, \ldots, w_K \tag{2.3.4}$$

The warping path satisfies the following constraints:

Length conditions

$max(m,n) \leq K < m+n-1$

Boundary conditions

$w_1 = D(1,1), w_k = D(n,m)$

Continuity

$w_{k-1} = D(i,j), w_k = D(i',j'), i'-i \leq 1, j'-j \leq 1$

Monotonicity

$w_{k-1} = D(i,j), w_k = D(i',j'), i'-i \geq 0, j'-j \geq 0$

Then we define a cumulative distances: Matching the two sequences Q and C, starting at point $(0,0)$, and every time get to a new point, the distances calculated by all the previous

points add up. When arrived the final point *(n,m)*, this cumulative distance is the total distance that we talked about, which shows the similarity between the sequence Q and C.

Cumulative distances *(i,j)* can be expressed as follows:

$$\gamma(i,j) = d(q_i,c_j) + min\{\gamma(i-1,j-1),\gamma(i-1,j),\gamma(i,j-1)\} \tag{2.3.5}$$

The best path can makes the final cumulative distances have the minimum value. This is illustrated in Figure 5.

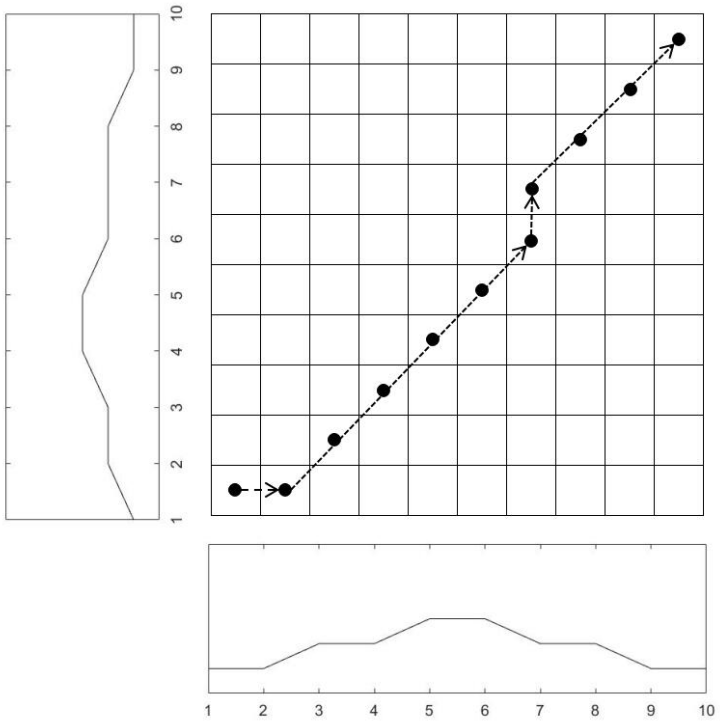

Figure 5: An example warping path.

The solution process of this path satisfies the idea of dynamic programming.

2.4 Dynamic Programming (DP)

Dynamic programming is an optimization method proposed by American scholar R.Bellman and others in 1951 to solve multi-stage decision-making problems [13].

Dynamic programming algorithms are usually based on a recursive formula and one or more starting states. The solution to the current subproblem will be derived from the solution to the previous subproblem.

2.4.1 The multi-stage decision problem

The problem that dynamic programming deals with is a multi-stage decision problem, which generally starts from the start state and reaches the end state through the choice of the

intermediate stage decision. These decisions form a sequence of decisions and determine an activity path (usually the optimal one) that completes the entire process. As shown in the Figure 6.

Figure 6: The multi-stage decision problem.

2.4.2 The basic idea of DP

Problems that can be solved by dynamic programming generally have three properties:

Optimal Substructure: if the optimal solution of the problem contains the optimal solution of the subproblem, it is said that the problem has the optimal substructure, which satisfies the optimization principle.

No Aftereffect: once a certain stage state is determined, it is not affected by the subsequent decision of this state. In other words, the process after a certain state will not affect the previous state, only related to the current state.

Overlapping Sub-problems: that is, the sub-problems are not independent, and a sub-problem may be used multiple times in the decision-making of the next stage.

The key of the dynamic programming method is to write out the basic recursive relation (State transfer equation) correctly. In order to do this, we must first divide the process of the problem into several interrelated stages, correctly select the state variables and decision variables of the problem, and define the specific form of the indicator function, so as to turn a large problem into a series of homogeneous sub-problems, and then solve them one by one. That is, starting with the boundary conditions, searching for the optimal solution recursively. In the solution of each sub-problem, the optimization result of its previous sub-problem will be used. Finally, the optimal solution of the last sub-problem is the optimal solution of the whole problem.

To solve the problem using dynamic programming, the following three points should be determined:

1 Identify the stage of the problem

2 Determine the state of each stage

3 Find the recursive relationship between the previous stage and the next stage

Look back to how we find the best path of DTW in the previous chapter, the process satisfies the idea of dynamic programming very well.

2.5 Locally Linear Embedding (LLE)

Locally Linear Embedding (LLE) algorithm was presented by Sam T. Roweis and Lawrence K. Saul in 2000 [14]. It is a good method for the problem of nonlinear dimensionality

reduction. LLE can keep the original manifold structure well after dimension reduction.

2.5.1 The basic idea of LLE

Assuming the data lies on a nonlinear manifold which locally can be approximated linearly, That is, a data can be represented linearly by several samples in its neighborhood. So for example, we have a sample data x_1, and we take the closest three data x_2, x_3 , and x_4 in its original high-dimensional neighborhood. Then we assume that x_1 can be linearly represented by x_2, x_3, and x_4, which is:

$$x_1 = w_{12}x_2 + w_{13}x_3 + w_{14}x_4 \tag{2.5.1}$$

Where, w_{12}, w_{13} and w_{14} are weight coefficients.

After the dimensionality reduction by LLE, we want x'_1, the map of data x_1 in low-dimension, have basically the same linear relationship with x'_2, x'_3, and x'_4, which are the low-dimensional maps of data x_2, x_3, and x_4:

$$x'_1 \approx w_{12}x'_2 + w_{13}x'_3 + w_{14}x'_4 \tag{2.5.2}$$

2.5.2 Algorithm derivation

The first step is to determine the size of the neighborhood, which is to determine how many neighbor data are needed to represent the sample data linearly. Let's say that the number of neighboring data is k. We can select out k nearest neighbours of the sample data by some distance measurement method, such as Euclidean distance.

After the sample data x_i and its k nearest neighbours has been determined, next is to find out the linear relationship between the sample data x_i and its k neighbours, or say calculate out the reconstruction weight coefficients.

Assume a data set contains m samples $\chi = \{x_1, x_2, ..., x_m\}$, we can use the mean square deviation as the cost function:

$$J(w) = \sum_{i=1}^{m} ||x_i - \sum_{j=1}^{k} w_{ij}x_j||^2 \tag{2.5.3}$$

Normalize the weight coefficient w_{ij}, that is, the weight coefficient should meet:

$$\sum_{j=1}^{k} w_{ij} = 1 \tag{2.5.4}$$

For x_j that is not in the neighborhood of x_i, makes the corresponding w_{ij} equal to 0.

Then it gives a contribution:

$$\begin{aligned} J(W) &= \sum_{i=1}^{m} ||x_i - \sum_{j=1}^{k} w_{ij}x_j||^2 \\ &= \sum_{i=1}^{m} || \sum_{j=1}^{k} w_{ij}x_i - \sum_{j=1}^{k} w_{ij}x_j||^2 \\ &= \sum_{i=1}^{m} || \sum_{j=1}^{k} w_{ij}(x_i - x_j)||^2 \\ &= \sum_{i=1}^{m} W_i^T (x_i - x_j)^T (x_i - x_j) W_i \end{aligned} \tag{2.5.5}$$

Where $W_i = (w_{i1}, w_{i2}, ..., w_{ik})^T$.

Make $Z_i = (x_i - x_j)^T (x_i - x_j)$, then the equations can be simplified to:

$$J(w) = \sum_{i=1}^{m} W_i^T Z_i W_i \tag{2.5.6}$$

With

$$\sum_{j=1}^{k} w_{ij} = 1 = W_i^T 1_k \tag{2.5.7}$$

Where 1_k is the all 1 vector in k dimension.

Then using the Lagrange Multiplier here we can have:

$$L(w) = \sum_{i=1}^{m} W_i^T Z_i W_i + \lambda(W_i^T 1_k - 1) \tag{2.5.8}$$

Take the derivative of W and set it equal to 0, and we get:

$$2Z_i W_i + \lambda 1_k = 0 \tag{2.5.9}$$

That is

$$W_i = \lambda' Z_i^{-1} 1_k \tag{2.5.10}$$

Where $\lambda' = -\frac{1}{2}\lambda$ is a constant. Using $W_i^T 1_k = 1$ normalize the W_i, we can have:

$$W_i = \frac{Z_i^{-1} 1_k}{1_k^T Z_i^{-1} 1_k} \tag{2.5.11}$$

Now we have the weight coefficients in high-dimension, and we want that the linear relation corresponding to these weight coefficients will be maintained in the low-dimension after dimensionality reduction.

Assuming LLE maps the data set $\chi = \{x_1, x_2, ..., x_m\}$ globally to a data set $\gamma = \{y_1, y_2, ..., y_m\}$ in a low-dimension. The cost function minimised is:

$$J(y) = \sum_{i=1}^{m} ||y_i - \sum_{j=1}^{k} w_{ij}y_j||^2 \tag{2.5.12}$$

It can be seen that this function is almost the same as the cost-function in high-dimension. The only difference is that in the cost function in high-dimension, the data is known, the objective is to find the weight coefficient W corresponding to the minimum function value, while in this cost function in low-dimension, the weight coefficient W is known, and the corresponding low dimension data is what needs to be obtained.

Constraints are as follows:

$$\sum_{i=1}^{m} y_i = 0; \frac{1}{m}\sum_{i=1}^{m} y_i y_i^T = I \tag{2.5.13}$$

This gives a contribution:

$$\begin{aligned} J(Y) &= \sum_{i=1}^{m} ||y_i - \sum_{j=1}^{k} w_{ij} y_j||^2 \\ &= \sum_{i=1}^{m} ||YI_i - YW_i||^2 \\ &= tr(Y^T (I-W)^T (I-W) Y) \end{aligned} \tag{2.5.14}$$

Make $M = (I-W)^T (I-W)$, an $n \times n$ matrix, then the cost function became to:

$$J(Y) = tr(Y^T M Y) \tag{2.5.15}$$

Where Y contains the y_i's as its columns.

Finding Y by minimise $J(Y)$ with constraint $Y^T Y = mI$. Using Lagrange multipliers we can have:

$$L(Y) = tr(Y^T M Y) + \lambda(Y^T Y - mI) \tag{2.5.16}$$

And setting the derivative to 0 gives: $MY = \lambda Y$. Clearly, this is an eigenvalue problem, all eigenvectors of M corresponding to the smallest eigenvalues minimise $J(Y)$ are solutions.

As an example, Figure 7 shows a 3D data set LLE-mapped to 2D.

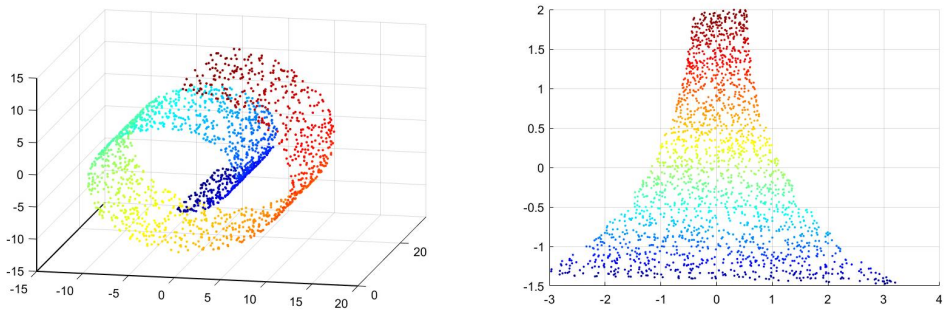

Figure 7: A data (d = 3, n = 2000) sampled from an swissroll-shaped manifold, and LLE-mapped Y (m = 2, k = 12).

3 Methodology

3.1 Profile Detection

Changes in the frontal facial expressions will give rise to corresponding changes in the profile view of the side face. Therefore, the matching of the frontal facial expressions can be simplified as the matching of the profile of the side face. The profile of the side view, or say the edge between the face and the background in side view is extracted from the side view frames through edge detection and be calculated as the pixel positions.

The Kirsch operator has be used here for edge detection, from the gradient magnitude image we got by Kirsch operator edge detection, we can see all the edges, contains the contour of the face that we need and other edges that we don't need like the edges of background stuff are all included in this new binary image.

By using Morphological image processing to deal the binary edge image we got after the edge detection, clear the outline, remove the noise and any unnecessary parts that are not needed, we can make in this binary edge image only the part shows the profile of the side face be retained (Figure 8).

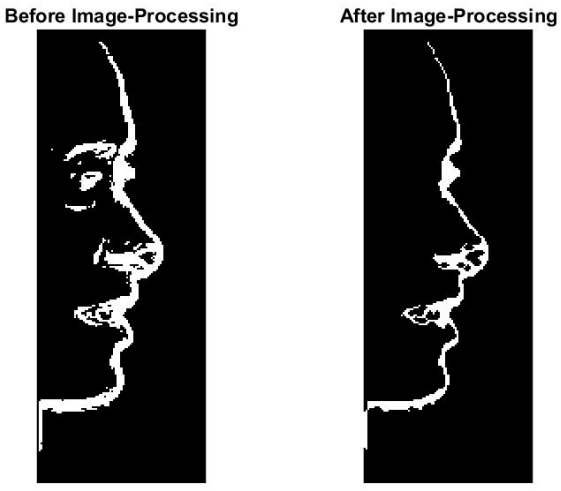

Figure 8: The comparison of before and after image processing.

But the width of this profile linealways be several pixels, to make the profile information in the image be represented as an array, we only keep the single layer pixel on the outermost edge of the profile to save and represent it as an array. The length of this array is equal to the height of the source frame image. As in Figure 9.

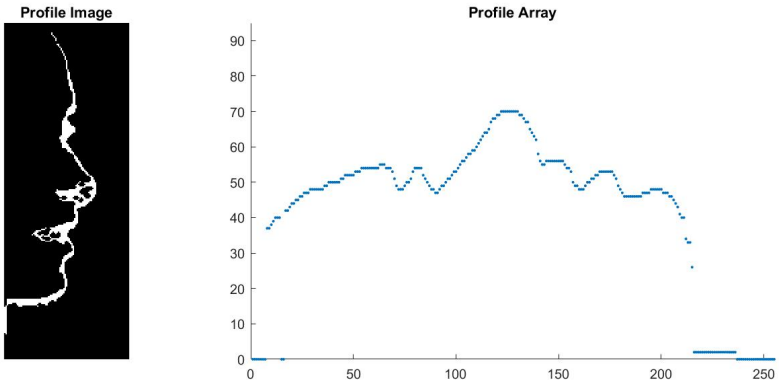

Figure 9: An example of the profile and its array.

This array that we have represents the contour of the side view face in one frame image. Finally, the number of arrays we have will equal to the number of frames that the source video have.

We can save it to a $M \times N$ matrix where M equals to the number of contour points and N equals to the number of frames of the source video.

3.2 APCA Based Video Reconstruction

Instead of the side view, here we use the profile of the side view for encoding. The profile of the side view for encoding has been calculated as the position fo each edge pixel. Then the eigenprofile for encoding can be calculated as:

$$\phi_j^{p_p} = \sum_i b_{ij}^p (X_i^p - X_o^p) \tag{3.2.1}$$

Where X^p are the profile of the side view, X_o^p is the mean image of the profile positions and b_{ij}^p are eigenvalues from the eigenvectors from the covariance matrix $(X_i^p - X_o^p)^T (X_j^p - X_o^p)$.

Encoding of profile is performed as:

$$\alpha_j^p = (X^p - X_o^p)^T \phi_j^{p_p} \tag{3.2.2}$$

And we have the eigenspace for the frontal video:

$$\phi_j^{p_{fr}} = \sum_i b_{ij}^{fr} (I_i^{fr} - I_o^{fr}) \tag{3.2.3}$$

Where I^{fr} is the frontal view of the frames and I_o^{fr} is the mean image of the frontal view.

The decoding can be performed as:

$$\hat{I}^{fr} = I_o^{fr} + \sum_{j=1}^{M} \alpha_j^p \phi_j^{p_{fr}} \tag{3.2.4}$$

By using the profile for encoding, the encoding process has been further simplified, but it also makes the high accuracy of edge detection became more important.

3.3 Contour Similarity Matching

The similarity between the reference contour series and the test contour series can be calculated by the dynamic time warping algorithm. After the profile detection, we can have the contour matrix for reference side-face video and test side-face video:

$$R_{M\times X} = [r_1, r_2, ..., r_i, ..., r_X]$$

Where M equals to the number of contour points and X equals to the number of frames of the reference video.

$$T_{M\times Y} = [t_1, t_2, ..., t_j, ..., t_Y]$$

Where M equals to the number of contour points and Y equals to the number of frames of the test video.

Then use dynamic time warping to make each column in the reference matrix be compared in turn with all the columns in the test matrix and calculate out the cumulative distance for each pair, as in Figure 10, find out the most similarity column pair and save the positions of them in reference matrix and test matrix.

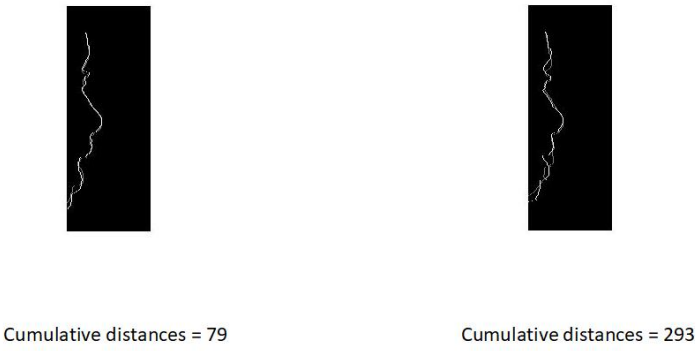

Figure 10: The comparison of two profile array and their cumulative distance.

After contour similarity matching has finished, each column in reference matrix have its most similar column in test matrix. Rearrange all the columns in test matrix to make each column in test matrix have the same order with its most similar column pair in reference matrix.

We can set an array to save the information about the original position of the corresponding column in test matrix like:

$$Order : [p_1, p_2, ..., p_x]$$

The number of each element shows the original position of the corresponding column in test matrix, for example the first number p_1 means the p_1th column in test matrix is the most similar column to the first column in reference matrix.

3.4 Video Reconstruction

We already know that changes in facial expressions can make the contour of the side-face be difference. By contour similarity matching, each frame of the reference profile video

have matched to its most similar frame in test video.

Each pair of the frontal video and the profile video have the same number of frames, and frame to frame correspondence. Rearrange all the frames in test profile video with the reference profile video as reference, for the new proflie video we get can say it has the most similar expression changing as the reference profile video, then if we rearrange all the frames in test frontal video with this new permutation, the new frontal video we got can be thought of haveing the most similar expression changes to the reference frontal video.

3.5 Video Improvement

The new reconstructed video we got looks quite well in most of the time, but still have some frames are not consistent with the rest of the video, these problem frames makes the video looks jerky. Local linear embedding (LLE) algrothemcan be used here to handle this issue. By using LLE to reduce the dimension of video, it is possible to find out the single problem frame which has substantially different with its neighbor frames and ensure that the other frames share similarities with the surrounding frames. As shown in Figure 11.

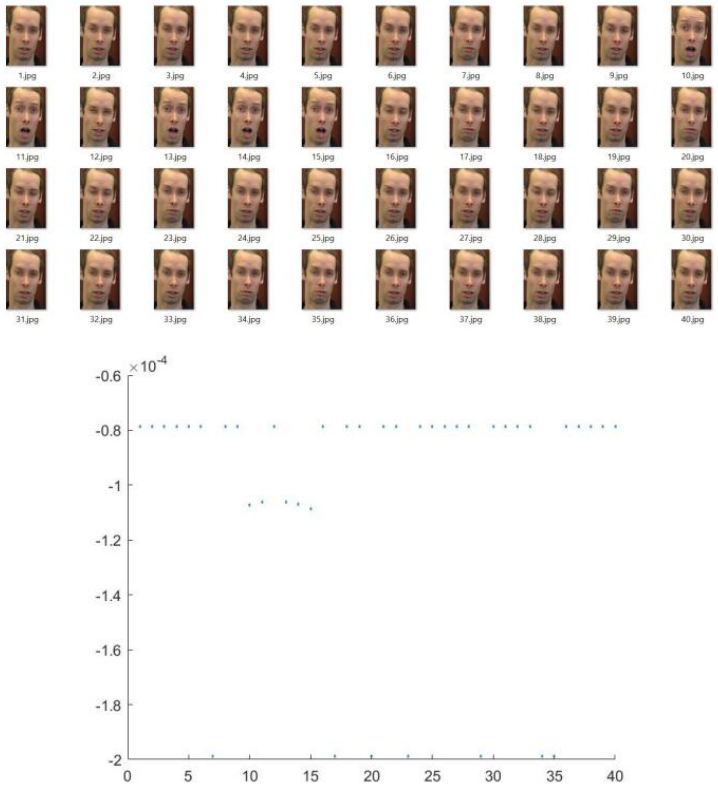

Figure 11: First 40 frames of a reconstructed video and their distribution after dimensionality reduction by LLE.

After the problem frames were found out, some methods can be used to create a video with smooth transitions between the frames.

There are usually two situations:

First one: A single problem frame suddenly appears in the video, which is quite different to its previous and next frames. As in Figure 12.

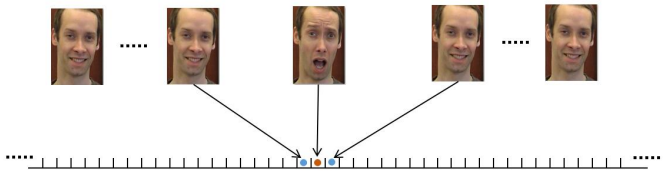

Figure 12: One single problem frame.

For this situation, we can choose to remove this problem frame and use a new frame made by the combination of its previous frame and its next frame to instead. As in Figure 13.

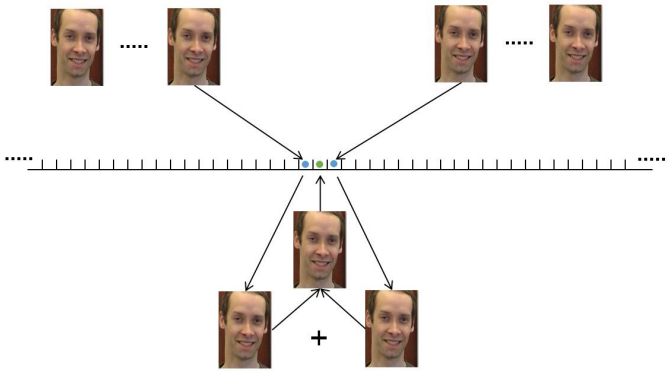

Figure 13: Frame replacement.

Second one: Starting from a frame, the subsequent frames are significantly different to the previous ones. As in Figure 14.

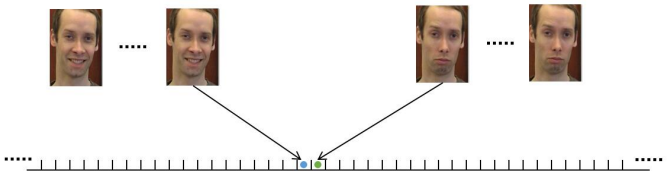

Figure 14: Different subsequent frames.

For this situation, we can artificially create a gradient from previous frames to subsequent frames. As in Figure 15.

With these two methods, the final video can looks smoother.

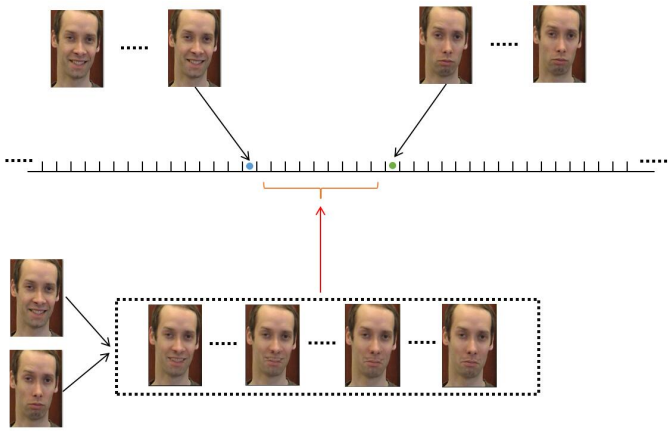

Figure 15: Gradient from previous frames to subsequent frames.

4 Results

4.1 APCA Based Video Reconstruction by Profile Encoding

We have two reconstructed videos which are “red” profile encoding “sweater” and “red” profile encoding “cap”. As in Figure 16.

Figure 16: Reconstructed videos (APCA).

To make the quality of the reconstructed videos can be measured, an evaluation standard called “Peak Signal to Noise Ratio (PSNR)” is used here, higher PSNR value means the reconstructed video have higher quality. The results are shown in Table 1 and Table 2.

The quality of each reconstructed video was measured in YUV color space because the video sequences are coded in this color space.

Table 1 Reconstruction quality for encoding with profile compared to original video

Reconstruction quality (PSNR) RED-SWEATER			
ϕ	Y	U	V
$\phi_{all}/4$	36.6	37.6	45.8

Table 2 Reconstruction quality for encoding with profile compared to original video

Reconstruction quality (PSNR) RED-CAP			
ϕ	Y	U	V
$\phi_{all}/4$	36.9	38.1	45.5

From Table 1 and Table 2 it can be seen that the quality of the reconstructed frontal view is slightly lower for profile encoding compared to the origional video, but the encoding complexity is reduced a lot.

4.2 DTW Based Video Reconstruction

Here we have two frontal face video. Take one video as the benchmark and reconstruct another one, we finally can have two reconstructed videos. As in Figure 17.

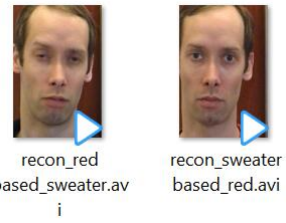

Figure 17: Reconstructed videos (DTW).

In order to evaluate the quality of the reconstructed videos, we asked some people to watch the reconstructed videos, compare each video with its reference video, and rate it to give out the final score. The range of the score for each video is from 1 to 5, 5 means the changing of the facial expressions can meet the reference video very well. We can have a table as follow:

Table 3 Evaluation for reconstructed videos

Score / Person / Video	1	2	3	4	5	Average
1	4	4	3	3	3	3.4
2	3	4	3	3	3	3.2

Most of the testers gave similar feedback: the tendency of facial expression changes in the reconstructed video was similar to that in the reference video, but the transitions of different facial expressions changes were not that natural. And the video jitter caused by some problem video clips affects the perception of the video.

From Table 3 it can be seen that most people feel the reconstructed video can generally shows the similar expression changes as the reference video have, the results can be accepted.

5 Discussion

We have shown how to use edge detection to detect the profile of a side view of a face image, and use this profile for video encoding.

We have both shown how Asymmetrical Principal Component Analysis (APCA) is used for encoding and decoding of different frontal facial and how to use Dynamic Time Warping (DTW) for similarity matching and video reconstruction.

DTW enables the reconstructed video to have the similar expression changes as its reference video. The reason we use the side view for video encoding is because the side view is easier to film since a camera can be placed closely to the side of the face instead of in front of the face.

By using only the position of the profile makes the encoding of APCA and the DTW based similarity matching be much easier. We find that the videos reconstructed by APCA have problems, e.g. it loses its natural transition between frames, it haves objective quality drops and cannot have the same expression changes as the reference, so we use another way to do the video reconstruction.

The videos reconstructed by DTW can show the similar expression changes as the reference video, but still do not have a natural transition between frames. This can be improved by incorporating local linear embedding (LLE). By using LLE, the frames which do not have a natural transition to their neighbors can be found and dealt with. Reconstructing videos via position of the profile is a new kind of video coding.

The entire procedure enables the video acquisition to became more user friendly since the frontal view does not need to be recorded and the side view can be recorded with less user impact.

6 Ethics of the Work

This thesis presents a new video communication method to facilitate people's daily life. The implementation of this method will simplify the existing video communication mode, and users can free their hands to be busy with other work when communicating with video.

We were devoted to considering the social and ethical implications of our work at all times. The results of the work in this thesis are beneficial to people's life, comply with code of ethics [15] and will not have adverse social and ethical impacts.

When it comes to social impact, a simplified way for people to communicate can lead to less travel and use of natural resources. Ethically, it must be ensured that the person in the video actually is the one that is communicating. Only the profile of the transmitting person is needed for encoding and it is possible to decode another person's face with the coding scheme. The security of such a scheme is not addressed in this thesis but it is an important issue that needs to be addressed in the future.

7 Future work

We can improve the whole method from the following aspects:

Improve the accuracy of profile capture: The accuracy of the extracted profile will directly affect the accuracy of subsequent processing. A more accurate profile would further improve the quality of reconstructed video, the result of similarity matching will be more accurate, and it will make the change of expression in the reconstructed video more natural and smooth. In order to obtain more accurate contour data, better algorithms other than edge detection can be used. For example, a large number of contour data can be learned by using artificial intelligence algorithm so that contour edge can be quickly and accurately obtained.

Improve the accuracy of profile matching: The order of each frame in a reconstructed video is determined by similarity matching based on DTW method. This similarity comparison method can reduce the interference caused by the difference in signal phase, but this method cannot eliminate the interference caused by the difference in facial orientation. If the profiles are oriented differently, even if the expressions of the two profiles are exactly the same, they will still be judged that there is a significant difference between two profiles. And because of the large amount of calculations needed, the run time for this DTW based similarity comparison consumes a lot of time. In future work, we can use better algorithms to achieve this function.

Optimize the reconstruction process of the video: The reconstruction of the video is based on frame by frame similarity matching and reordering of the source video. But, this may cause problem like making the reconstructed video to lose its natural transition between adjacent frames. A possible solution to this is to find the key frames in the video that determine the change of expression, divide the video into video fragments based on these key frames, and then rearrange these video fragments based on similarity matching to realize video reconstruction.

The key point is to reduce the computational complexity and increase the accuracy and naturalness of the restored expression in the video.

References

[1] Hao Li, Laura Trutoiu, Kyle Olszewski, Lingyu Wei, Tristan Trutna, Pei-Lun Hsieh, Aaron Nicholls, and Chongyang Ma. Facial performance sensing head-mounted display. *ACM Transactions on Graphics (ToG)*, 34(4):47, 2015.

[2] Lawrence G Roberts. *Machine perception of three-dimensional solids*. PhD thesis, Massachusetts Institute of Technology, 1963.

[3] Judith MS Prewitt. Object enhancement and extraction. *Picture processing and Psychopictorics*, 10(1):15–19, 1970.

[4] Irwin Sobel. History and definition of the so-called" sobel operator", more appropriately named the sobel-feldman operator. *Sobel, I., Feldman, G.," A 3x3 Isotropic Gradient Operator for Image Processing", presented at the Stanford Artificial Intelligence Project (SAIL) in*, 2015, 1968.

[5] Jules R. Dim and Tamio Takamura. Alternative approach for satellite cloud classification: Edge gradient application. *Advances in Meteorology,2013,(2013-12-11)*, 2013(11):1–8, 2013.

[6] Pietro Perona and Jitendra Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on pattern analysis and machine intelligence*, 12(7):629–639, 1990.

[7] John Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698, 1986.

[8] R Kirsch. Computer determination of the constituent structure. *Computers and biomedical research*, 4:315–328, 1971.

[9] Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[10] Ulrik Söderström and Haibo Li. Asymmetrical principal component analysis theory and its applications to facial video coding. In *Effective Video Coding for Multimedia Applications*. InTech, 2011.

[11] Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing*, 26(1):43–49, 1978.

[12] Fumitada Itakura. Minimum prediction residual principle applied to speech recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 23(1):67–72, 1975.

[13] Richard E Bellman and Lotfi Asker Zadeh. Decision-making in a fuzzy environment. *Management science*, 17(4):B–141, 1970.

[14] Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326, 2000.

[15] Brandon Ingram, Daniel Jones, Andrew Lewis, Matthew Richards, Charles Rich, and Lance Schachterle. A code of ethics for robotics engineers. In *Proceedings of the 5th ACM/IEEE International Conference on Human-robot Interaction*, pages 103–104. IEEE Press, 2010.