2008 American Control Conference
Westin Seattle Hotel, Seattle, Washington, USA
June 11-13, 2008

A Robust Dynamic Time Warping Algorithm for Batch Trajectory Synchronization

Yang Zhang and Thomas F. Edgar

Department of Chemical Engineering, The University of Texas at Austin, Austin, TX, 78712 USA

***Abstract*— Derivative Dynamic Time Warping (DDTW) is useful in time series alignment to avoid singularity points and reduce the bias of alignment results towards the reference trajectory compare with Dynamic Time Warping (DTW). In order to apply DDTW to an industrial environment, a robust DDTW (RDDTW) algorithm is proposed by combining a moving window least squares procedure with DDTW. The developed algorithm is tested with noisy data and results are compared with those of DDTW and DTW.**

I. Introduction

Time series data are a commonly occurrence in physical, social, and economic systems. In order to compare one sequence with another, it is always desirable to align the features (peak, valley, etc.). Dynamic Time Warping (DTW) can capture the dynamics and match patterns for different series and it was first used in speech recognition and introduced for chemical batch profile synchronization by Kassidas et al. in 1998 [1]. Subsequently, DTW method was successfully applied to spectroscopic profile analysis [2], semiconductor production monitoring [3] and other applications.

Because DTW warps the batch profile based on the Euclidean distance between corresponding points of reference and the new series, it fails when a feature (peak, valley, etc.) is higher (or lower) from one batch to another. Furthermore, a single point on one time series can map to a large number of points on the other one (called a singularity point) with DTW. In order to overcome these problems, Keogh and Pazzani [4] used point derivative instead of distance to measure the difference between two trajectories. Their results demonstrated that DDTW overcomes the singularity problem and retains the most important features of the raw trajectory to obtain a synchronized batch profile without bias.

Yang Zhang is with Department of Chemical Engineering, The University of Texas at Austin, Austin, TX, 78712 USA. (e-mail: yz@che.utexas.edu)

Thomas F. Edgar is with Department of Chemical Engineering, The University of Texas at Austin, Austin, TX, 78712 USA. (e-mail: edgar@che.utexas.edu)

Despite those attractive features, in order to apply DDTW online, one should make the numerical derivative estimation robust to process noise. There are several candidates to increase the robustness:
(1). Instead of using raw data, use piecewise average values;
(2). Use a filter for high frequency signal noise;
(3). Use moving window least squares (Savitzky-Golay method).

In this study, all three methods are tested based on simulation and experimental data. Based on the results, a robust DDTW (RDDTW) algorithm is proposed. The rest of the paper is organized as follows: DTW and DDTW algorithms will be reviewed in Section 2 and the new RDDTW algorithm is proposed in Section 3. Several application examples are shown in Section 4, and in Section 5 we give conclusions and propose future work.

II. Algorithm Review and Development

2.1 DTW

Assume x_{ref} is a reference trajectory (previously defined) and x_{new} is a new trajectory to be synchronized. The first step of DTW is to define the Euclidian distance between each point of the two trajectories as:

$$d(i(k), j(k)) = \{x_{ref}[i(k),:] - x_{new}[j(k),:]\} \cdot W \cdot \{x_{ref}[i(k),:] - x_{new}[j(k),:]\}^T \quad (1)$$

d is the local distance, W is a positive weighting matrix that reflects the repeatability of each process variables for batch synchronization. k is the number of grid points along the path (in Figure 1, k=1,2,…,7). The lengths of x_{ref} and x_{new} are t and r, respectively. Then the total distance between the two trajectories are:

$$D(t,r) = \frac{\sum_{k=1}^{K} d(i(k), j(k)) \cdot w(k)}{N(w)} \quad (2)$$

where $w(k)$ is a nonnegative weighting function for local distance (d) and $N(w)$ is a normalization factor. There are many possible point assignments and the goal is to find the

optimal path *f* that minimizes *D(t,r)*, which can be denoted as (see the blue line in Figure 1):

$$f = \{c(1), c(2), ..., c(K)\} \quad (3)$$

c(k) represents a grid point in Figure 1, where *k* can be treated as another index used to connect reference (*i*) and new trajectories (*j*), e.g. *c(1) = [1,1]*, *c(2) = [1,2]*, *c(7) = [5,6]* in Figure 1. *K* is the total number of points needed for the path (*K*=7 in Figure 1).

Before minimizing Eq. 2, there are some global and local constraints that should be addressed:

Global constraints:

The two ends of both trajectories (*c(1)* and *c(K)*) should be aligned together:

$$c(1) = [1,1]$$

$$c(K) = [t, r]$$

where

$$c(k) = [i(k), j(k)]$$

The constraint on path lengths of x_{ref} and x_{new} is:

$$\max(t, r) \le K \le t + r$$

Local constraints:

Local constraints are used to make the path continuous and monotonic. If *(i,j)* is the *k*th point on *f (c(k))*, then the predecessor point on path *f* (*c(k-1)*) can be any one of the three points(the three directions in Figure 1):

$$c(k-1) = (i-1, j),\ (i-1, j-1)\ or\ (i, j-1) \quad (4)$$

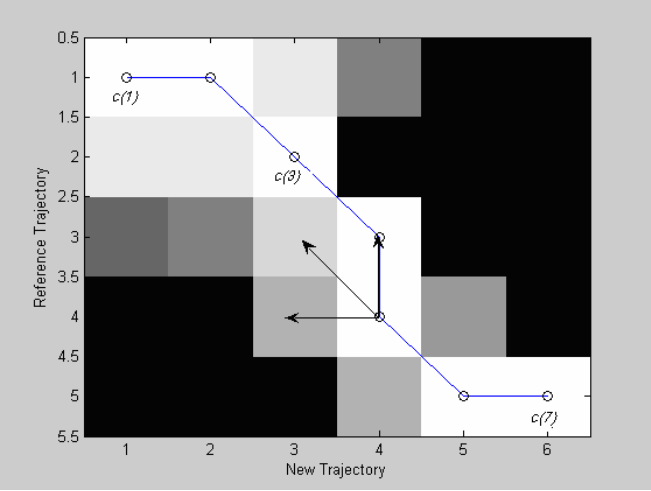

Figure 1. Warping path by DTW.

After minimizing Eq. 2 subject to the local and global constraints by dynamic programming, the optimal warping path can be found. Detailed algorithm descriptions can be found in [1, 2, 5].

As pointed out above, singularity points may occur and several methods have been proposed to alleviate this such as windowing, slope weighting and step patterns [4]. Although these methods may prevent singularity points, they can distort the warping profiles and it is not obvious how to choose the parameters needed by each algorithm. Furthermore, DTW sometimes fails to find a natural alignment because a feature (peak, valley, inflection point, etc.) in one sequence may be slightly different than the other.

2.2 DDTW

According to Keogh and Pazzani [4], the shortcomings listed in Section 2.1 can be overcome by DDTW, which measures the difference of the estimated derivatives of the reference and new trajectories:

$$d(i(k), j(k)) = \{dx_{ref}[i(k),:] - dx_{new}[j(k),:]\} \cdot W \cdot \{dx_{ref}[i(k),:] - dx_{new}[j(k),:]\}^T \quad (5)$$

where $dx_i = \frac{x_i - x_{i-1}}{2}$. DDTW can be carried out by replacing Eq. 1 by Eq. 5 and carrying out the minimization of Eq. 2. Instead of measuring Euclidean distance, DDTW considers the estimated local derivatives. According to Keogh and Pazzani [4], DDTW's time requirement is *O(tr)*, which is the same as DTW, thus both methods take approximately the same time for computation.

Figures 2 a and b compare the synchronization results of DTW and DDTW. The new trajectory follows $x=\cos^2(t)+\sin(t)$ $1<t<10$ (in red) and the valley between time points 6 to 12 in the reference trajectory is moved downward by 2 (in blue). According to DTW (Figure 2a), point 8 in the new trajectory is aligned to five points (7 to 11) in the reference trajectory. This result is consistent with the shortcomings of DTW described earlier. After replacing point distance by point derivative, DDTW results are shown in Figure 2b. It can be seen that point 8 in the new trajectory is only aligned to point 8 in the reference. Comparing Figures 2 a and b, one can see that the new trajectory is aligned much better by DDTW than DTW. Figure 4 shows the alignment path by DDTW. As expected, a perfect alignment between these two trajectories should be a diagonal line and Figure 2c is very close to this. From this simple example, we can see DDTW seems to correct the problems with DTW. However, DDTW may not be as robust as DTW as a result of local numerical derivative estimation. The left point estimation ($dx_i = \frac{x_i - x_{i-1}}{2}$) is the simplest way to estimate a local derivative but it amplifies any process noise dramatically.

For this noise-free case (Figure 2), DDTW works fine but will have difficulty in real industrial applications where the noise level is normally not low. When left point estimation is used to estimate point derivatives of the process: $x_n(t)=\sin(t)+norm(0,0.04)$, the quality of estimation becomes unacceptable. As a result, a more robust version of the DDTW algorithm is needed.

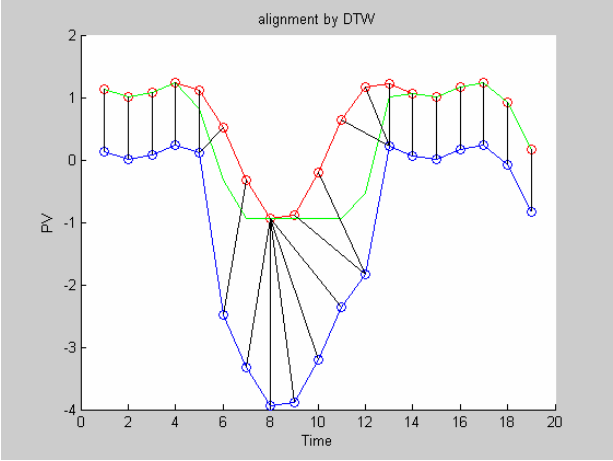

Figure 2a. DTW alignment of two trajectories. Blue: the reference trajectory. Red: the trajectory to be synchronized. Green: the synchronized trajectory suggested by DTW.

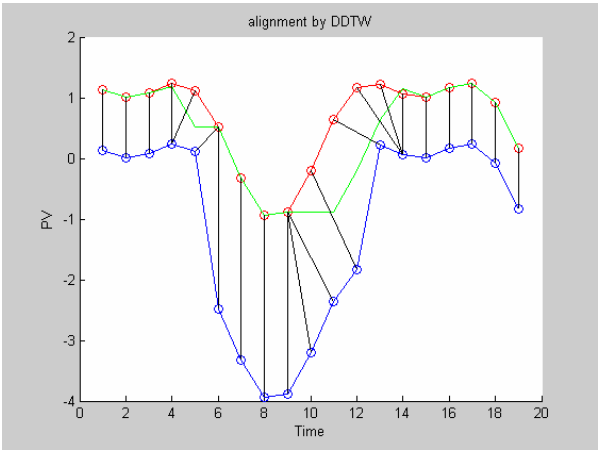

Figure 2b. DDTW alignment of two trajectories. Blue: the reference trajectory. Red: the trajectory to be synchronized. Green: the synchronized trajectory suggested by DDTW.

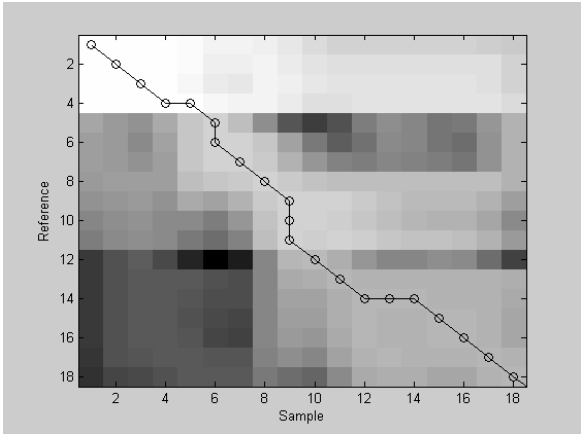

Figure 2c. DDTW alignment path of two trajectories.

2.3 Moving Window Least Square (SG filter)

In 1964, Savitzky and Golay [6] combined the idea of least squares and moving window together, which can be used to smooth raw data and predict derivatives simultaneously. Errors contained in [6] were later corrected by Steinier et al.[7].

The smoothing problem with a moving window can be reformulated as follows: $2h+1$ measurements ($X = [x_{-h}, x_{-h+1}, \ldots x_h]$) are to be fit into an n^{th} order polynomial ($n<2h+1$) which is defined by:

$$f_i = \sum_{k=0}^{n} a_{nk} i^k; \quad i = -h, -h+1, \ldots, 0, \ldots, h-1, h \qquad (6)$$

Thus for the window length we have $F = [f_{-h}, f_{-h+1}, \ldots f_h]$ and in order to find the parameter value, a least square objective function is applied:

$$J = \min\{(X-F)^T (X-F)\} \qquad (7)$$

In order to solve this least squares problem, the first derivative of Eq. 7 ($\frac{\partial J}{\partial a_{nr}}$) is set to zero (note: there is no constraint on the parameters (a)):

$$\frac{\partial J}{\partial a_{nr}} = \sum_{i=-h}^{h} [(\sum_{k=0}^{n} a_{nk} \cdot i^k) - x_i] \cdot i^r = 0 \qquad (8)$$

By solving Eq. 8, we have:

$$\sum_{k=0}^{n} a_{nk} S_{r+k} = F_k \qquad (9)$$

where

$$S_{r+k} = \sum_{i=-h}^{h} i^{r+k}$$

$$F_k = \sum_{i=-h}^{h} i^k x_i$$

and r is the equation number that runs from 0 to n. Eq. 9 can be used to determine the value of polynomial coefficients a_{nk} derived in [6, 7]. If the polynomial order or the number of observations used in the estimation is so large that it is not listed in [6, 7], Eq. 9 can be used. In real application, the first step is to calculate S_{r+k} and substitute it into Eq. 9. Together with F_k which is related to real observations, all parameters a_{nk} can be expressed by a weighting matrix in terms of observations.

With a_{nk}, one can calculate the smoothed value at current time ($i=0$) and its corresponding derivatives. The jth order derivative of the smoothed polynomial trajectory at $i = 0$ can be expressed as:

$$\left(\frac{d^j f_i}{d_i^j}\right)_{i=0} = j!a_{nj} \qquad (10)$$

As an example, seven observations are used to determine the first derivative by fitting a quadratic polynomial: $n = 2$; $h = (7-1)/2 = 3$:

$$dx_0 = a_{21} = \frac{-3x_{-3} - 2x_{-2} - 1x_{-1} + 0x_0 + 1x_1 + 2x_2 + 3x_3}{28}$$

Taking Eq. 7 into account, $j = 1$ so that $j! = 1$, thus a_{21} is the robust quadratic estimation function with seven observations.

By combining the Savitzky-Golay (SG) filter with DDTW, a robust DDTW algorithm can be proposed. The noisy sinusoid process described in Section 2.2 is tested by the SG filter method. A piecewise average (every ten observations) is taken first as the process dynamics change much slower than the sampling rate. Both five point and seven point SG are closer to the true dynamics compare with exponential filter approach (in blue) used in original DDTW algorithm, thus the SG filter is a robust method for derivative estimation.

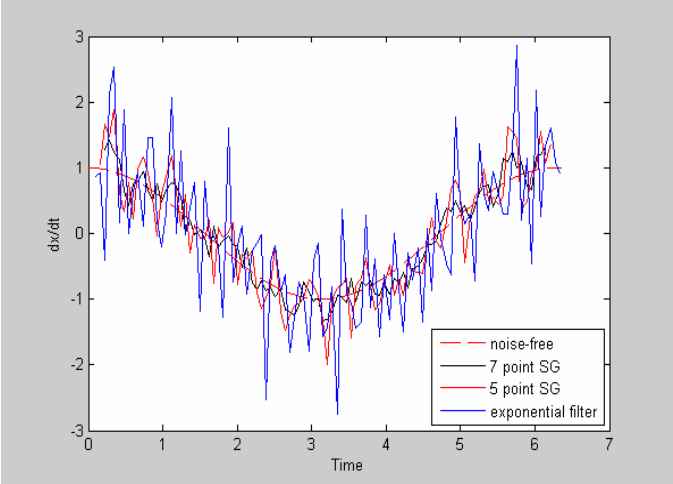

Figure 3. Five and seven point SG filter estimation compared with exponential filter approach based on piecewise averaging (average is taken every ten observations).

2.4 RDDTW algorithm

RDDTW can be treated as a combination of DDTW and robust numerical derivative estimation. The DDTW algorithm includes following steps:

1. Evaluate the raw trajectory. If the raw trajectory has high frequency noise, it is necessary to use piecewise averaging before performing further calculations.
2. Use SG method to estimate numerical derivatives. Use a quadratic polynomial although a higher order polynomial can be chosen together with a larger number of observations. Usually the number of observations should be at least twice as many as the order of the polynomial in order to get good results. The number of observations indicates the rate of change in the dynamics. For example, if a second order polynomial is chosen and four observations are used to fit the polynomial, the polynomial can be used to represent the raw data in future calculations.
3. Use the estimated derivative and the algorithm described in Sections 2.1 and 2.2 to perform the alignment robustly.

III. Examples and Results

3.1 Industrial NIR Data

In 2002, the International Diffuse Reflectance Conference (IDRC) published a "Shootout" data set consisting of spectra from 654 pharmaceutical tablets from two spectrometers. Here, two spectra were selected to test our algorithm (Figure 4). One can see the raw trajectories are fairly well-synchronized and only a minor alignment is needed. The main differences are in the magnitudes of the trajectory features.

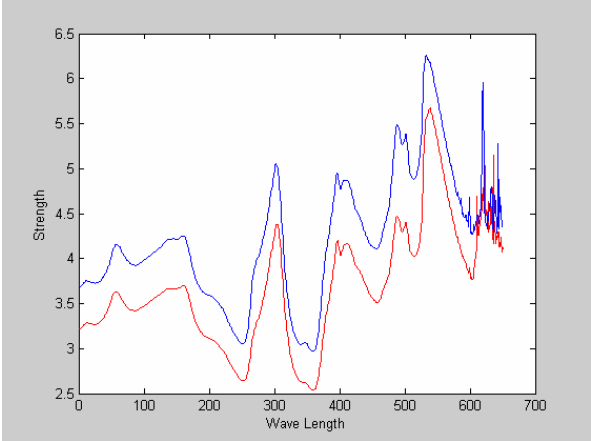

Figure 4. Raw NIR trajectory from IDRC dataset. Blue: reference trajectory. Red: trajectory to be synchronized.

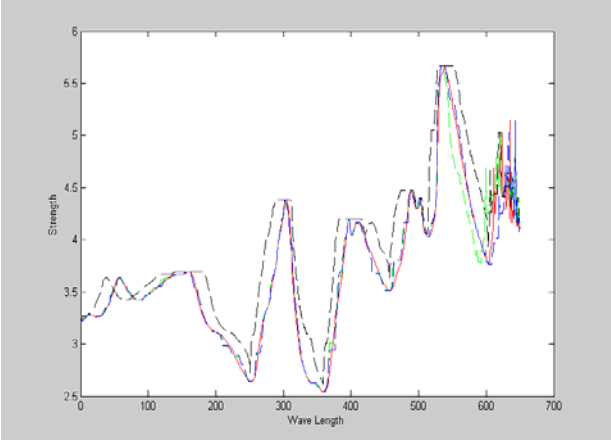

Figure 5. Synchronized NIR trajectory by different approaches. Red: trajectory to be synchronized; Black dash: DTW; Green dash: DDTW; Blue dash: RDDTW.

Figure 5 compares the alignment results. One can see that DTW shifted the raw trajectory between wavelength 30 to 80, which is unnecessary (Figure 5). Furthermore, flat periods appear at nearly every peak as anticipated. For this low noise level case, DDTW and RDDTW give similar results and only minor differences exist around wavelength 550, which means RDDTW and DDTW are equivalent in this case. Thus, we can conclude that DTW causes singularity points due to feature differences and for low noise level cases, while DDTW and RDDTW give satisfactory results that avoid singularity points.

3.2 Dynamic Trajectory Simulation

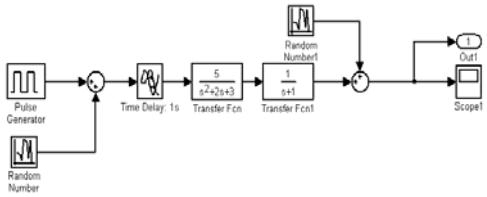

Figure 6. Simulation example developed by SIMULINK.

Figure 6 shows a process developed by SIMULINK. The system input is composed of a pulse sequence and a random number generator. After a one second transport delay, the input signal passes through a second order and a first order system. Finally, the output from first order system contains random noise. Reference and new trajectories are generated with different pulse magnitudes (0.7 for reference and 0.5 for new trajectory, respectively). The random number variance for input and output variable is 0.05 and 0.01, respectively. The simulated trajectories are shown in Figure 7 with a sampling rate is of seconds. Both trajectories behave similarly but the new trajectory (in blue) is neither synchronized nor does it have the same magnitude as the reference (in red).

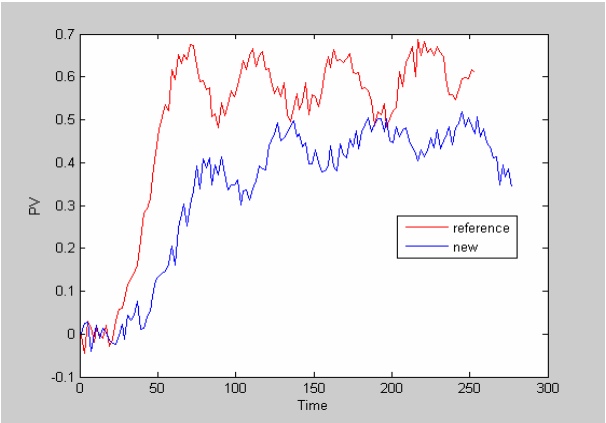

Figure 7. Simulated reference and new trajectories.

Figure 7 indicates the noise frequency is not very high, so the piecewise averaging step is bypassed. The piecewise averages are used for DTW, DDTW and RDDTW calculations. For all three methods, the local constraint is set to 40. For RDDTW, seven points with a third order polynomial is used to estimate the numerical derivative.

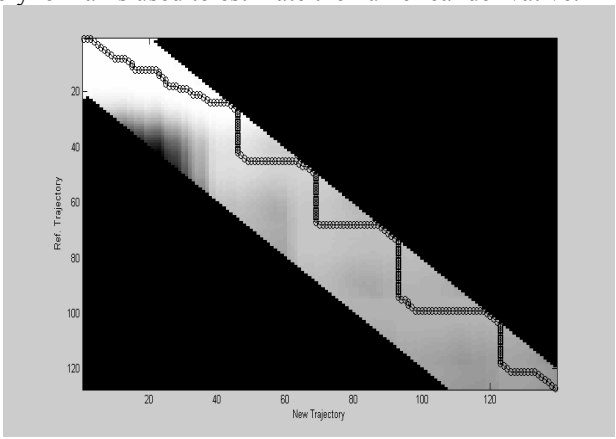

Figure 8a. DTW alignment path with local constraint equal to 40.

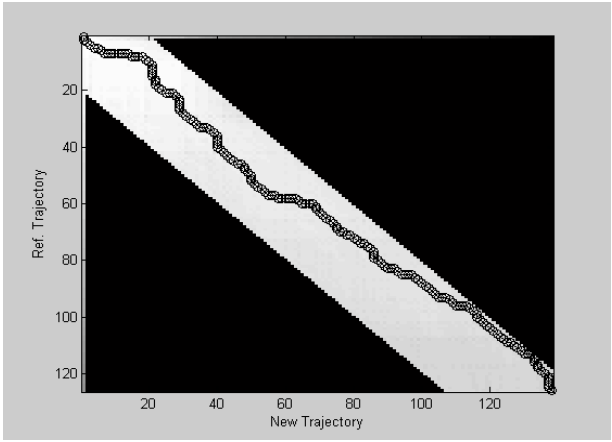

Figure 8b. DDTW alignment path with local constraint equal to 40.

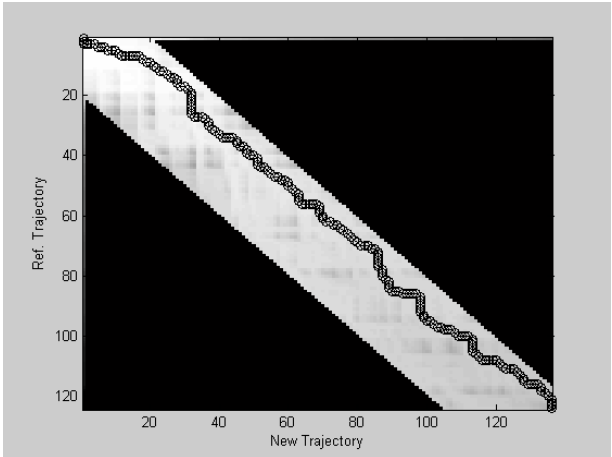

Figure 8c. RDDTW alignment path with local constraint equal to 40.

Figure 8 shows the alignment paths for each method. It is clear that Figure 8a has many perpendicular and horizontal moves indicating singularity points. Furthermore, the path reaches the boundary at all times. In comparison, DDTW and RDDTW have fewer singularity points and do not reach the boundary. After a closer look at Figures 8b and c, DDTW and RDDTW give different alignment paths. Aligned trajectories are shown in Figure 9. DTW (in blue) shows many flat periods caused by singularity points. Comparing the reference (in red) trajectory with both DDTW (in green) and RDDTW (in black) results, neither method has many singularity points, and both methods have similar results after observation 150. However, during the first 150 observations, especially the rising period from 20 to 70, RDDTW was synchronized better than DDTW since DDTW was somewhat closer to the raw trajectory (Figure 9b). Figure 9 also suggests RDDTW and DDTW do not have an obvious bias toward the reference trajectory and retain the important features of the original data (peak height, valley depth, etc).

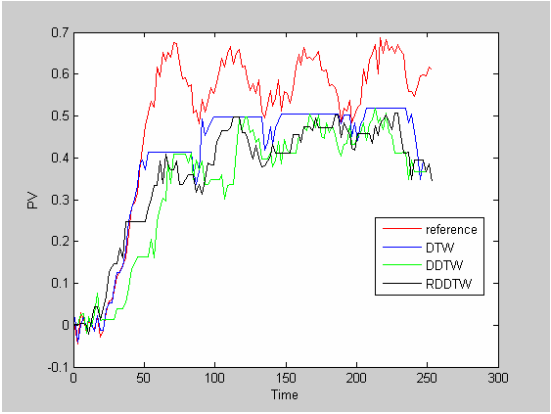

Figure 9a. Alignment results comparison with reference trajectory.

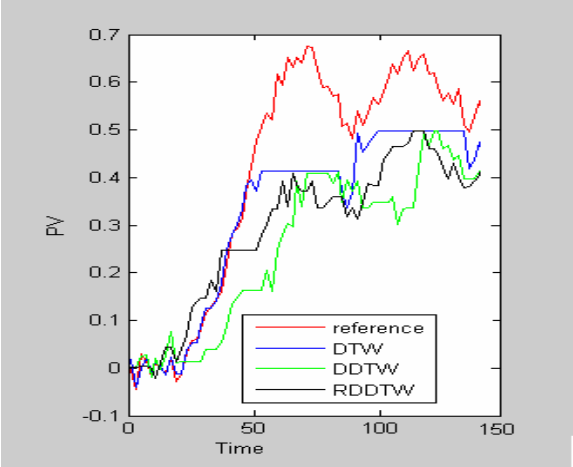

Figure 9b. Alignment results comparison with original trajectory.

IV. CONCLUSIONS

In this research, we proposed a new algorithm (RDDTW) to perform data alignment for batch processes that is comprised of three steps: 1. piecewise averaging; 2. SG filter to estimate local numerical derivatives; 3. dynamic optimization to calculate the alignment path. Two examples are used and synchronization results show that RDDTW can significantly reduce singularity point number, retain the most important feature of raw trajectory and avoid bias to obtain a good batch profile. The proposed method will avoid singularity points and may not be affected by changing features compared with DTW. In addition, RDDTW is robust to process noise compared with DDTW.

Although the examples shown in this research are mainly chemically related, RDDTW can be used in other time series analysis such as gesture recognition, manufacturing, speech processing and medicine.

In our future research, the derivative and Euclidian-based methods will be combined and RDDTW will be tested with real industrial data.

ACKNOWLEDGMENT

The authors thank Emerson Process Management for their support of this research.

REFERENCES

[1] A. Kassidas, J. F. MacGregor, and P. A. Taylor, "Synchronization of Batch Trajectories Using Dynamic Time Warping," *AIChE Journal*, vol. 44, pp. 864-875, 1998.

[2] V. Pravdova, B. Walczak, and D. L. Massart, "A Comparison of Two Algorithms for Warping of Analytical Signals," *Analytica Chimica Acta*, vol. 456, pp. 77-92, 2002.

[3] G. A. Cherry, "Methods for improving the reliability of semiconductor fault detection and diagnosis with Principal Component Analysis," PhD dissertation: The University of Texas at Austin, 2006.

[4] E. Keogh and M. Pazzani, "Derivative Dynamic Time Warping," in *First SIAM International Conference on Data Mining*. Chicago, Illionis, 2001.

[5] H.-J. Ramaker, E. N. M. v. Sprang, J. A. Westerhuis, and A. K. Smilde, "Dynamic Time Warping of Spectroscopic BATCH Data," *Analytica Chimica Acta*, vol. 498, pp. 133-153, 2003.

[6] A. Savitzky and M. J. E. Golay, "Smoothing and differentiation of data by simplified least squares procedures," *Analytical Chemistry*, vol. 36, pp. 1627-1639, 1964.

[7] J. Steinier, Y. Termonia, and J. Deltour, "Smoothing and differentiation of data by simplified least square procedure," *Analytical Chemistry*, vol. 44, pp. 1906-1909, 1972.