An Adaptive Tai-Chi-Chuan AR Guiding System Based on Speed Estimation of Movement

Yi-Ping Hung[1,2,3(✉)], Peng-Yuan Kao[2], Yao-Fu Jan[4], Chun-Hsien Li[3], Chia-Hao Chang[2], and Ping-Hsuan Han[2]

[1] Graduate Institute of Animation and Film Art, Tainan National University of the Arts, Tainan, Taiwan
hung@csie.ntu.edu.tw
[2] Graduate Institute of Networking and Multimedia, National Taiwan University, Taipei, Taiwan
[3] Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan
[4] Department of Computer Science, National Chengchi University, Taipei, Taiwan

Abstract. Augmented Reality (AR) headsets has become a potential device as an auxiliary tool for practicing physical activities such like Tai-Chi Chuan (TCC). Although some learning systems can display the virtual coach movement in AR headsets, the playing speed cannot be adjusted appropriately just like a real coach stand next to you. In most of the learning system, the common approach is using controller to control the playback system under a specific speed. Once user want to speed up or speed down, he has to do these commands via a controller. In this work, we propose a TCC learning system which will real time detect the delay time between current action of user and virtual coach. After obtaining the delay time, our learning system will adjust speed of virtual coach automatically. With real time speed adjustment, learners can practice TCC with their own pace and virtual coach will slow down or speed up to follow learners' movement.

Keywords: Real-time speed estimation · Physical activity learning · Mixed reality · Tai-Chi Chuan

1 Introduction

In the processing of learning Tai-Chi Chuan, users are asked to perform a sequence of body movements with highly accurate position. According to the learner is familiar with the movement or not, learner might practice in different speed. In addition, learner may not only start practicing at the first movement but also a specific movement. The way to control playback system can be generally grouped into several categories: controller, camera, voice command and wearable based. First of all, controller-based learning systems usually use hands to control system via touch the screen, push the buttons, and using other input device, e.g., mouse or keyboard [1]. However, controller-based approach will cause some problem. Once user would like to adjust the

C.-Y. Chang et al. (Eds.): ICS 2018, CCIS 1013, pp. 115–130, 2019.
https://doi.org/10.1007/978-981-13-9190-3_11

speed faster or slower, he should suspend the current movement to control the learning system. These pausing will make the practicing incoherent and inefficient, we called it practice interrupt problem. Second, camera-based learning systems [2, 3] overcome the practice interrupt problem. This approach use camera as additional sensor to detect user's gesture, but it will cause another problem that user should face to the camera with specific direction. The third approach, voice command, uses speech recognizer to control the playback system [4], and it has no problem which users are limited to specific position and direction. Nevertheless, it leads to another problem that speech recognizer not always detect the correct word successfully.

In this work, we propose a TCC learning system which will real time detect the delay time between current action of user and virtual coach. After obtaining the delay time, our learning system will adjust speed of virtual coach automatically. With real time speed adjustment, learners can practice TCC with their own pace and virtual coach will slow down or speed up to follow learners' movement.

2 Related Work

In Around Me [3], they developed a learning system with a movable robot which contains computer, monitor and camera. The robot could dynamically change the position based on user's position. Swimoid [5] support system also provides a movable robot in water, it shows user's swim posture real-time on monitor of robot. These two system both use movable robot to carry monitor, so users still need to face the display of robot although these robots would follow users automatically.

Some of these researches don't provide any way to control learning system, users can only play the whole movements from the beginning to the end. MotionMA [1] is a system that provide body movement and current speed information, it doesn't allow users make the system play faster or slower. In [6] and [7], they also focus on learning steps of dancing, but don't provide an interactive learning system for dancing learner. However, people do exercise would have different speed, the fixed speed definitely not suit for everyone.

Several researches control learning system by using gesture recognition technology. YouMove [2] is one of the interactive learning system that use Kinect to recognize user's skeleton and track the hand position to control system. However, this approach would cause practice interrupt problem, users may not have good practicing experience if they should stop current movement to control learning system. Around Me [3] is a learning system that combine gesture recognition and vision-based approach, it used gesture to send movable robot remote instructions and use camera to track user's position to follow him. Joggobot [8] and Flying Sports Assistant [9] are also using a drone as a movable robot to encourage and guide users. These approaches also cause the practice interrupt problem as mention before. My Tai-Chi coaches [4] is an interactive TCC learning system using finger gesture and voice command to control system. Finger gesture doesn't overcome the practice interrupt problem because users still need to stop and do finger gesture. Although voice command doesn't have this problem, speech recognition approach isn't work well in every environment such like noisy place. If the number of command growth, recognize correct keyword will also be more difficult.

3 Speed Estimation of Movement

When learning process of general exercise, there is always a delay between learner and coach. According to the difficulty of actions, the delay time will be different. The more difficult or unfamiliar the movement is, the more delay will be caused. In this chapter, we propose a method to estimate the delay time of movements between the learner and the virtual coach, and to estimate the speed difference between the coach and the learner by the known delay. At the end of this chapter, we designed an experiment to verify the accuracy of the algorithm and the efficiency of execution.

3.1 Problem Definition

In this system, we except that leaner will follow the movements of the virtual coach to practice TCC. But when the user moves much more slowly than the virtual coach, the learner usually forgets what the coach was doing, and will skip to the current action of the virtual coach. So, we constraint the range of the delay time, and set the maximum delay time to 5 s. For those who have some experience of TCC, their speed of playing TCC might be faster than the virtual coach, so we have set a maximum of 5 s for the time which is faster than the virtual coach.

As you can see from formula 1, where a_u is current action of user and a_c is action of virtual coach. In our learning system, *fps* equals to 35 which means data receive rate is thirty-five frames per second. We can notify that when Δt is positive, it means that the movement of the learner is slower than the virtual coach. When Δt is negative, it means that the movement of the learner is faster than the virtual coach.

$$\Delta t = \frac{a_u - a_c}{fps} \tag{1}$$

The constraint for Δt is as following:

$$-5 \leq \Delta t \leq 5 \tag{2}$$

3.2 Movement Acquisition

Before making an estimate of speed, we must get the movement of virtual coaches and the learner first. The movement of virtual coaches are obtained in advance, and we import the movement file into unity, then set the virtual coach to play TCC which based on the movement file. So, the learner can see each detail of the movement through the AR headsets from TCC coach. We can estimate the delay time between virtual coaches and the learner by comparing the information which obtained by wrist wearable device for both of them.

3.2.1 Standard Movement Acquisition

While getting the standard movements, we used the Vicon Motion Capture System in order to record the movement which obtain high accuracy. We have invited a coach who has more than 10 years of TCC experience and we use many infrared Vicon cameras to record the movement of the coach (Fig. 1).

Fig. 1. Using Vicon cameras to record coach movement

3.2.2 Learner Movement Acquisition

In the process of practicing TCC, we ask learners to wear the smart bracelet which has built-in nine axis IMU. So, we can obtain the information of the wrist movements from the user through Bluetooth to the mobile device.

3.3 Algorithm

We propose an effective and accurate algorithm. First of all, we take the current action of the virtual coach as the center of search space. We will crop the data from the previous five seconds to the next five seconds from the center. This segment of whole data is the search space. When searching the data, we use a sliding window with size $fps \times 5$ s, and the sliding window stride equals to $fps \times 0.5$ s.

After cropping the search space, we will move sliding window through search space with every 0.5 s and the end of sliding window start from $t - 5$ s to $t + 5$ s. When we find the end of sliding window, we can take this sliding window which starts from 5 s ago as a feature (Fig. 2). Take Fig. 2 as an example, when the current point we search is at $t - 5$ s, the data which start previous 5 s than the point to $t - 10$ s will be regarded as a feature. As for learner movement, we regard data obtained from learner from five seconds previous than current action as a feature (Fig. 3).

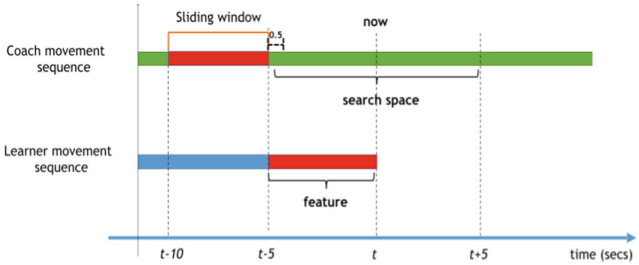


Fig. 2. Illustration of algorithm

Function SpeedEstimation()

Input: C – a TimeSeries of coach movement with length $|C|$
U – a TimeSeries of user movement with length $|U|$
M_c – coach current movement

Output: time interval between user current movement and coach current movement

```
// Crop user movement data to let |U| equal to fps × 5(seconds)
U = crop_data_with_length( user_movement, length = fps × 5 )

// Set sliding window width and sliding window stride
sliding_window_width = fps × 5
sliding_window_stride = fps × 0.5

// Let Mc be the center and compute signal similarity between each sliding
// window data which is in search space and U
// Record the smallest DTW distance happened in which sliding window
min_distance = MAX_INFINITE
sliding_window_index = -1
FOR( each sliding window data Ci in search space)
{
    dist, path = fastDTW( U, Ci )
    IF( dist < min_distance )
    {
        min_distance = dist
        sliding_window_index = i
    }
}

// Use the last frame of sliding_window_index^th sliding window data and
// Mc to calculate Δt
last_frame_index = last_frame_in_sliding_window( C, index =
                                sliding_window_index)

deltaT = ( last_frame_index - Mc) / fps

RETURN deltaT
```

Fig. 3. Pseudocode of algorithm

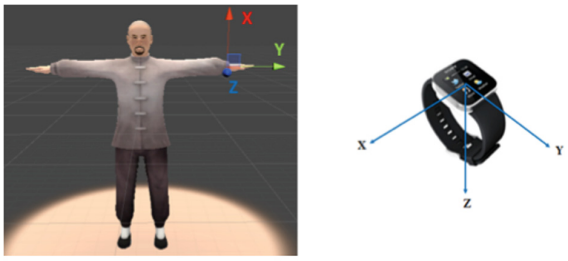
After obtaining the feature of learner and virtual coach, we will calculate signal similarity between these two features. When find out all of sliding windows under search space, we will have an estimation result which has highest similarity with learner current action. We can estimate Δt by dividing the frame number between learner current action and the last frame of sliding window which has highest similarity with learner movement by *fps* which equals to 35 in our system.

3.4 Details of Implementation

In this chapter, we will illustrate some details of our implementation of this algorithm. This contain the pre-processing after receiving data and the implementation details of comparing the similarity for two signals.

3.4.1 Space Correspondence Alignment

After collecting information from both hands of the learner and the virtual coach, we have to align the coordinate axis of the data of the left and the right hand first. We can compare the data after corresponding the coordinate axis. The corresponding coordinate axis for smart bracelet and the wrist of the virtual coach is shown in the following table (Fig. 4) (Table 1):



(a) (b)

Fig. 4. (a) Space correspondence of virtual (b) Space correspondence of Sony SmartWatch3

Table 1. Corresponding coordinate axis

Coach movement axis	Sony SmartWatch3 axis
X	−Z
Y	−X
Z	Y

3.4.2 Signal Similarity

In our algorithm, we use dynamic time warping (DTW) to compute the distance between two signals. People usually use DTW to compute the similarity of two 1D-signals. However, we consider that both of left wrist and right wrist at the same time, so we concatenate the acceleration of both wrists and the dimension of input data when computing DTW have become 6D. Suppose there are two time series R and Q, and the acceleration of i^{th} point in R can be expressed as $(acceR_x^i, acceR_y^i, acceR_z^i)$, and the acceleration of j^{th} point in Q can be expressed as $(acceQ_x^j, acceQ_y^j, acceQ_z^j)$ We can derive the distance $Dist(i,j)$ between R_i and Q_j by following formula:

$$Dist(i,j) = \sqrt{(acceR^i_{lx} - acceQ^j_{lx})^2 + (acceR^i_{ly} - acceQ^j_{ly})^2 + (acceR^i_{lz} - acceQ^j_{lz})^2} + \sqrt{(acceR^i_{rx} - acceQ^j_{rx})^2 + (acceR^i_{ry} - acceQ^j_{ry})^2 + (acceR^i_{rz} - acceQ^j_{rz})^2} \quad (3)$$

3.4.3 Computing Efficiency Improvement

In order to give users real-time feedback, computing efficiency of the algorithm is also an important consideration. To improve computing efficiency, we use FastDTW [10] in our TCC learning system instead of DTW. FastDTW uses a multilevel approach that recursively projects a solution from a coarse resolution and refines the projected solution. By using FastDTW, time complexity and space complexity can be reduced to linear time.

3.5 Sliding Window Size Selection

The size of sliding window is an important part of our algorithm. As sliding window size too small, we only can view little part of whole signal. When there is no significant feature or some significant features which are divided into two segments, learning system will misjudge the matching feature and cause bizarre speed adjustment. When sliding window size is big enough, it can reduce the influence of misjudgment caused by incomplete feature. However, sliding window size is not the bigger the better. When sliding window size is too big, it will significantly reduce the computing efficiency. To design a real-time interactive learning system, we must make a tradeoff between accuracy and computing efficiency.

To select an appropriate sliding window size, we design an experiment. We invited four TCC coaches who have experience of TCC for several years. Two of them have more than 10 years' experience and one of coaches has more than 6 years' experience and the other coach has more than 4 years' experience. We ask coaches to wear smart bracelets on both wrists and wear AR HEADSETS. We choose a TCC movement which is consist of eight key postures (Fig. 5) and ask coaches to follow the virtual coach. Each coach practiced this movement for five times.

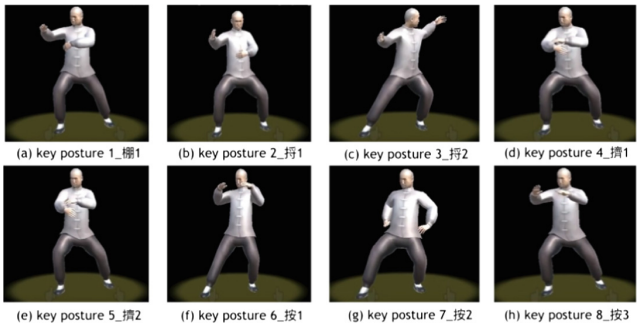


Fig. 5. Eight key postures in one TCC movement

When coaches practicing this movement, we record the coach's movement with a camera at the same time. After practicing, we label the timestamps of each key posture occurs in coach's movement manually.

We defined the ground truth of delay time difference of i^{th} key posture Δt_g^i as timestamps of i^{th} key posture labelled manually minus timestamps of i^{th} key posture virtual coach played. And We defined estimation delay time difference of i^{th} key posture Δt_e^i as timestamps of i^{th} key posture estimated by our algorithm minus timestamps of i^{th} key posture virtual coach played. After deriving ground truth and estimation of delay time difference, we can calculate average estimation error of each coach by formula 4. Then we can get total average estimation error by calculating the average of four average estimation error of coaches.

$$Error_{avg} = \frac{\sum_{n=1}^{5} \sum_{i=1}^{8} |\Delta t_{g,n}^i - \Delta t_{e,n}^i|}{5 \times 8} \tag{4}$$

We defined $\Delta t_{g,n}^i$ as delay time difference of i^{th} key posture Δt_g^i as timestamps of i^{th} key posture labelled manually minus timestamps of ith key posture virtual coach played at nth times coach playing this movement.

In this experiment, we limit the sliding window size to multiple of *fps* which equals to 35 in our learning system and constraint the multiple from 1 to 10. We will record the accuracy and total computing time of different sliding window size using in our algorithm. We found that when sliding window size equals to $fps \times 5$, the algorithm had lowest estimation error and the total computing time is acceptable. So, we use

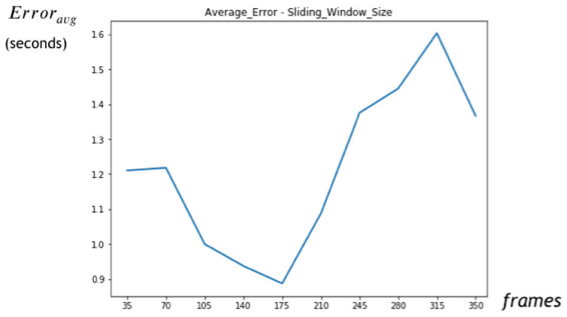


Fig. 6. Average estimation error with different sliding window sizes

$fps \times 5$ as sliding window size in our algorithm (Fig. 6).

3.6 Evaluation

We design an experiment to evaluate our algorithm. We invited four TCC coaches who have experience of TCC for several years. Two of them have more than 10 years' experience and one of coaches has more than 6 years' experience and the other coach

has more than 4 years' experience. We ask coaches to wear smart bracelets on both wrists and wear AR headsets. We choose a TCC movement which is consist of eight key postures (Fig. 5) and ask coaches to follow the virtual coach practicing this movement for five times and we record the coach's movement with a camera at the same time. After deriving ground truth and estimation of delay time difference, we can calculate average estimation error equals to 0.8875 s and cost 0.05 s for each estimation.

There is estimation error distribution of each key posture (Fig. 7). We can find that the estimation error of key posture 1 is lowest and key posture 6, 7, 8 will cause higher

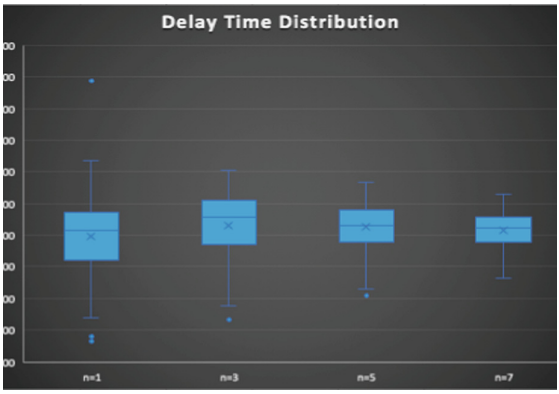


Fig. 7. Algorithm estimation error distribution

estimation error and the range of estimation error is wider, because each coach has different habits when practicing this movement.

4 Delay Time Normalization

In real life, when the speed of the user is not able to keep up with the coach, the coach will stop or slow down to wait for the student to follow. And we propose a method to adjust the speed of the virtual coach by using the algorithm described in previous chapter. After calculating the time difference between virtual coach and the learner, we can estimate the difference of speed between learner and virtual coach. When the learner is slower than the virtual coach, the virtual coach will slow down. Whereas when the speed of the learner is faster than the virtual coach, the virtual coach will speed up.

4.1 Delay Time Distribution

We hope to understand the distribution of delay time for each key posture of both leaner and the coach. After knowing the time difference between coach and learner, we hope to design a TTC learning system to shorten the delay between the novice and the coach, so we designed an experiment. We Invited four coaches with many years of TCC experience and five novices without TCC experience. We asked the learner to wear smart

bracelet and head-mounted monitor and follow the virtual coach in the head-mounted monitor to do one style with eight key postures of TTC movements (Fig. 5). We will use camera to record the movements and analyze the distribution of Δt_g.

We can find three things from the following figure. First, when the delta t (Formula 1) is less than 0, it means that the speed of the user is much faster than the virtual coach. Due to the coach is more familiar with the movement, the average of a lot of key posture appear earlier than the coach. Second, the delay time of the coach is more concentrated, and due to the reason that novice is not familiar to the movements, so the delay time is more dispersed. Third, there will be a delay time whether it is coach or novice playing TCC, and the trend of delay time will be the same. For example, the

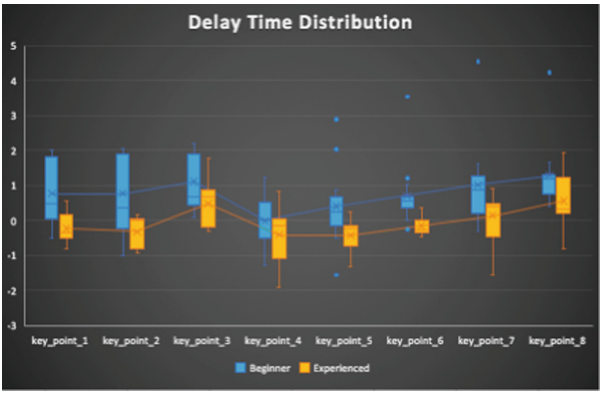


Fig. 8. Delay time distribution

movement of the user will be much slower than the virtual coach when the key posture 2 moves to key posture 3. And the movement of the user will be faster when key posture 3 moves to key posture 4 (Fig. 8).

4.2 Delay Time Normalization Implementation

When we get the Δt from calculating algorithm, the delta t will be a consistent value ideally. However, in reality, there will have different delay time delta Δt^k at different time k. We propose a method that we will add the delay time Δt^k which obtained at the timing t^k to the next n seconds. In other words, we change to play the film of n seconds to $n + \Delta t^k$ after t^k (Fig. 9). The updated speed $speed_{adjust}$ is calculated as follows:

$$speed^k_{adjust} = speed_{normal} \times \frac{n}{n + \Delta t^{k-1}_e} \tag{5}$$

We can see that when Δt^k is a positive, it means that the movement of the user is slower than the virtual coach, so speed adjust will be a value less than 1. On the other

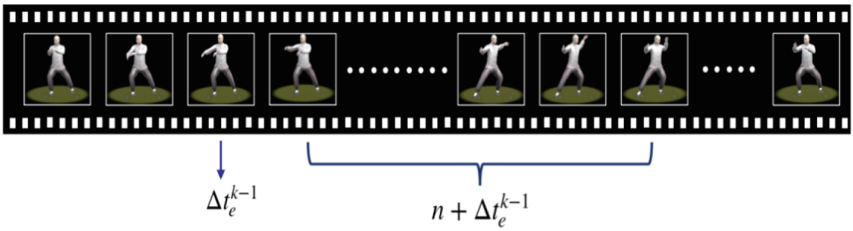


Fig. 9. Delay time normalization

hand, when Δt^k is a negative, it means the movement of the user is faster than the virtual coach, so $speed_{adjust}$ will be a value more than 1.

4.3 Experiment

Before using Δt^k to adjust the speed of virtual coach, we need to determine the value of n first. In order to find out the best n value for users, we design an experiment. In this experiment, we limit n to 1, 3, 5, 7. We invited two TCC coaches who have experience of TCC for several years and two beginners. We ask participants to wear smart bracelets on both wrists and wear AR HEADSETS. We choose a TCC movement which is consist of eight key postures (Fig. 5) and ask participants to follow the virtual coach practicing TCC for three times under each mode with different n. When participants practicing this movement, we record their movement with a camera at the same time.

As the following result (Fig. 10), we can find that when n is equal to 7, average delay time between learner and virtual coach almost equal to zero. Although average delay time is almost zero when n is equal to 1, the distribution of delay time is very divergence. Because when n equals to 1, learning system is too sensitive for adjusting the speed of virtual coach. When the speed of virtual coach is changed intensely,

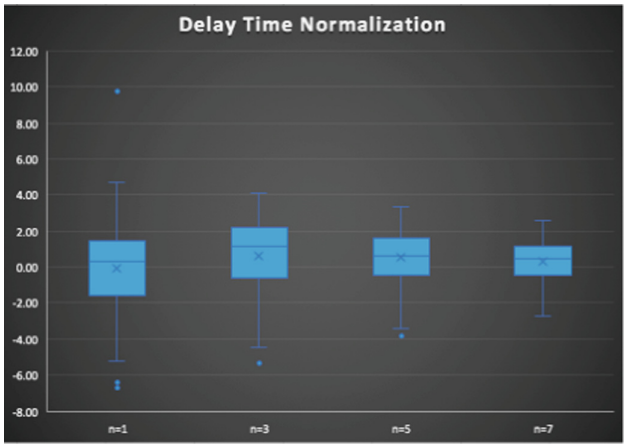


Fig. 10. Delay time normalization

learners cannot follow the virtual coach movement. According to the result, we choose the value of n as seven and normalize the speed of virtual coach in our TCC learning system.

5 Adaptive Tai-Chi-Chuan Learning System

In this chapter, we will introduce the intelligent TCC learning system. We can know how to calculate the delay time of the current action between user and virtual coach in this system, and we will use the result to adjust playback speed of the virtual coach.

5.1 System Overview

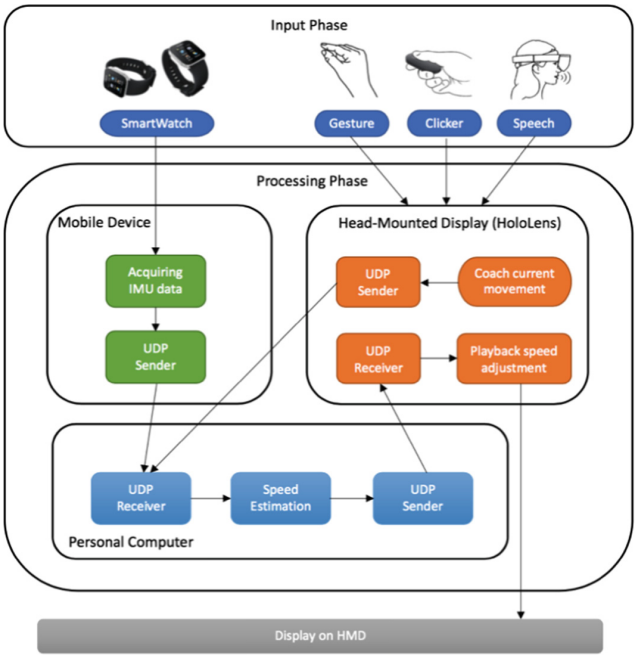


Fig. 11. System overview

Our TCC learning system can be separated into three phases (Fig. 11). First, this TCC learning system needs a control module to interact with the system. We can use Gesture, Clicker and Speech to achieve this goal. Besides, in order to acquire user's wrist rotation information, we have two smart bracelets in our input module.

In processing phase, the IMU in smart bracelet will transfer information to mobile device through Bluetooth. After receiving data, mobile device will transfer information through UDP to PC. At the same time, AR HEADSETS will transfer the current action of the coach through UDP to PC. When obtain the information from the wrist of the

user and the current action of the coach, PC will calculate delta t by the algorithm in the third chapter. PC will calculate the updated speed after calculating Δt, and then transfer the updated speed through UDP to AR headsets.

5.2 Hardware Configuration

5.2.1 Head-Mounted Display

(a) (b)

Fig. 12. Hardware devices (a) Microsoft HoloLens (b) Sony SmartWatch3

The TCC learning system in this research was developed on an augmented reality device, Microsoft HoloLens shown in Fig. 12(a). HoloLens is a see-through head-mounted display worn over the user's eyes because the eye-piece component is transparent.

5.2.2 Smart Bracelet

The smart bracelet that we use is Sony SmartWatch3 which has built-in nine-axis IMU shown in Fig. 12(b). After Sony SmartWatch3 receive information from the wrist of the user, it will transfer the IMU information through Bluetooth to mobile device. When we receive the information from the current action of the user, we can determine the time discrepancy between the user and the virtual coach immediately.

5.3 Speed-Adaptive Mode

In Speed-Adaptive Mode, the learning system will calculate the delay time Δt of the user and the coach immediately. According the result in the fourth chapter, our learning system will add Δt to next seven seconds and normalize the playing speed of virtual coach. When the learning system detect the difference of playing speed between the learner and the virtual coach. When the learner is slower than virtual coach, learning system will slow down the playing speed.

5.4 Waiting Mode

We also consider about another teaching scenario in reality. Once learners in class are slower too much than the coach, instead of playing slow down and moving on next action, the coach may stop at some important pose and let learners to capture more details. For this scenario, we design the Waiting Mode. In Waiting Mode, if the learner

is slower than the virtual coach more than 3 s, the virtual coach will stop at current action and wait for the learner until the learner is also stop at the same action.

6 User Study

We designed a user study to compare the efficiency of Speed Adjustment Mode and Waiting Mode. We invited two TCC coaches who have experience of TCC for several years and two beginners to wear smart bracelets on both wrists and wear AR headsets. We ask participants to follow AR headsets virtual coach and practice a TCC movement which is consist of eight key postures (Fig. 5) and play under three modes which are Without Speed-Guided Mode, Speed-Adaptive Mode and Waiting Mode. Participants were asked to practice five times of each mode. We will use camera to record the movement and calculate the average delay time. After deriving ground truth and estimation of delay time difference, we analysis the distribution of delay time under these three modes.

As we can see the figure below, we can find that if learners practice TCC with Speed-Adaptive Mode, their delay time will be closer to zero which means the movement of learners is more similar with the virtual coach. However, when learners practice TCC under Waiting Mode, delay time will be bigger than Without Speed-Guided Mode. Because when the virtual coach stop, learners cannot detect immediately.

After ANOVA analysis, we obtain that there is a significant difference between Without Speed-Guided Mode and Speed-Adaptive Mode in the group of beginners. It means that Speed-Adaptive Mode is significantly helpful for beginners to decrease the delay time toward the virtual coach. However, for those who already have several years' experience of TCC, there is no significantly difference under Speed-Adaptive

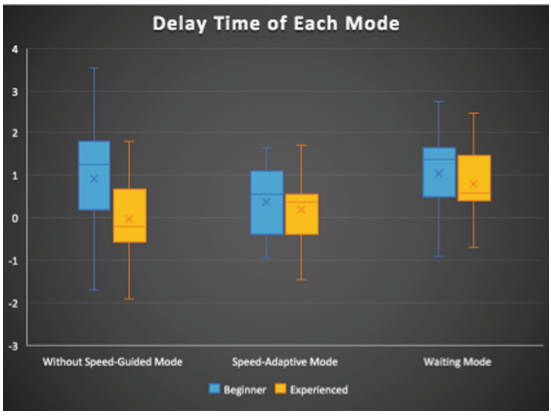


Fig. 13. System overview

Mode. As the analysis result, there is no significant difference between Without Speed-

Guided Mode and Waiting Mode. In our opinion, Waiting-Mode is more appropriate to practice one static action and Speed-Adaptive Mode is more appropriate to practice a serial of sequential movements (Fig. 13).

7 Conclusions and Future Works

In this work, we propose a TCC learning system which will real time detect the delay time between current action of user and virtual coach. After obtaining the delay time, our learning system will adjust speed of virtual coach automatically. With real time speed adjustment, learners can practice TCC with their own pace and virtual coach will slow down or speed up to follow learners' movement. With this learning system, users can learn TCC better and more efficiency.

In the future, we will try to combine the techniques of deep learning. Consider about processing efficiency, we may try SSD or YOLO. Use an image as input data and find out the wrists position at real time. By acquiring this additional information, we expect that we can increase the accuracy of movement recognition. Once the accuracy of movement recognition is increased, we can modify our learning system to become better and help those who want to learn Tai-Chi Chuan.

Acknowledgements. This study is partially supported by Ministry of Science and Technology, Taiwan (R.O.C.), under grant MOST 107-2221-E-369 -001 -MY2 and supported by the "III Innovative and Prospective Technologies Project" of the Institute for Information Industry which is subsidized by the Ministry of Economy Affairs of the Republic of China.

References

1. Velloso, E., Bulling, A., Gellersen, H.: MotionMA: motion modelling and analysis by demonstration. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 1309–1318. ACM, April 2013
2. Anderson, F., Grossman, T., Matejka, J., Fitzmaurice, G.: YouMove: enhancing movement training with an augmented reality mirror. In: Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology, pp. 311–320. ACM, October 2013
3. Tominaga, J., Kawauchi, K., Rekimoto, J.: Around me: a system with an escort robot providing a sports player's self-images. In: Proceedings of the 5th Augmented Human International Conference, p. 43. ACM, March 2014
4. Han, P.H., Chen, Y.S., Zhong, Y., Wang, H.L., Hung, Y.P.: My Tai-Chi coaches: an augmented-learning tool for practicing Tai-Chi Chuan. In: Proceedings of the 8th Augmented Human International Conference, p. 25. ACM, March 2017
5. Ukai, Y., Rekimoto, J.: Swimoid: a swim support system using an underwater buddy robot. In: Proceedings of the 4th Augmented Human International Conference, pp. 170–177. ACM, March 2013
6. Drobny, D., Borchers, J.: Learning basic dance choreographies with different augmented feedback modalities. In: CHI 2010 Extended Abstracts on Human Factors in Computing Systems, pp. 3793–3798. ACM, April 2010
7. Chan, J.C., Leung, H., Tang, J.K., Komura, T.: A virtual reality dance training system using motion capture technology. IEEE Trans. Learn. Technol. **4**(2), 187–195 (2011)

8. Graether, E., Mueller, F.: Joggobot: a flying robot as jogging companion. In: CHI 2012 Extended Abstracts on Human Factors in Computing Systems, pp. 1063–1066. ACM, May 2012
9. Higuchi, K., Shimada, T., Rekimoto, J.: Flying sports assistant: external visual imagery representation for sports training. In: Proceedings of the 2nd Augmented Human International Conference, p. 7. ACM, March 2011
10. Salvador, S., Chan, P.: FastDTW: toward accurate dynamic time warping in linear time and space. Intell. Data Anal. **11**(5), 561–580 (2007). ISBN:978-1-4503-0000-0/18/06