LEGENDARY OF 18 WEAPONS – MOTION CAPTURE DATA ANALYSIS FOR WII REMOTE

Supervisor : Prof. Michael R. Lyu

Presented by :

Ng Kwok Ho, Alex

Tsoi Chi Hung, Tony

OUTLINE

- Introduction
  - Background information
  - Motivation & Objective
- The Wii Remote
  - How to get data from Wii Remote ?
  - What can be obtained from Wii Remote ?
- Method of Classification
  - Possible Classification Methods
  - Working Principle of Dynamic Time Warping
- How do we apply Dynamic Time Warping ?
- Tuning
- Future Work
- Q & A

INTRODUCTION

BACKGROUND INFORMATION

- Legendary of 18 Weapons
  - 18 major weapons used in traditional martial art

* Images are extracted from http://www.kenchan.com.hk

BACKGROUND INFORMATION

- Motion capture
  - Recording movements in a digital manner
  - With the use of accelerometers
- Wii Remote
  - Has built-in accelerometers
  - Capture movements of 1 hand
  - Can be extended to both hands with the use of Nunchuk

MOTIVATION

- Wii is a revolutionary console
  - A new method of playing games
  - Costly to buy a console
- Wii Remote can pair with personal computers
  - Capture data using Wii Remote and send to computer
  - Players can still enjoy with specially-designed games

OBJECTIVES

- Construct a motion classifier
- Build an interface
  - A game, or an educational software
  - Perform motions in front of computer
  - Recognize correctly and be displayed in the computer

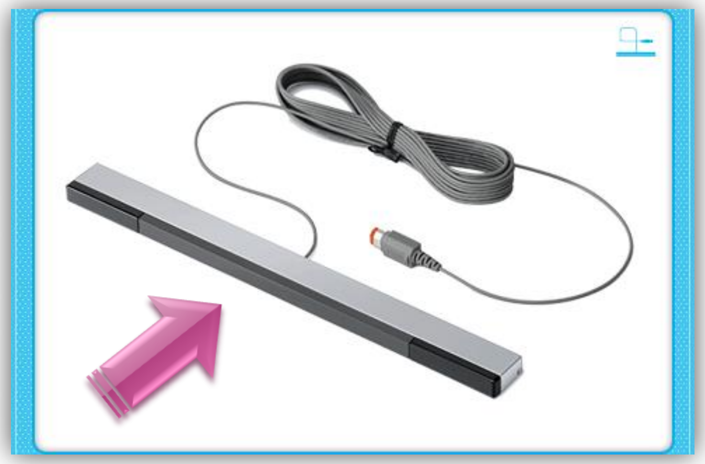
THE WII REMOTE

HOW TO GET DATA FROM WII REMOTE ?

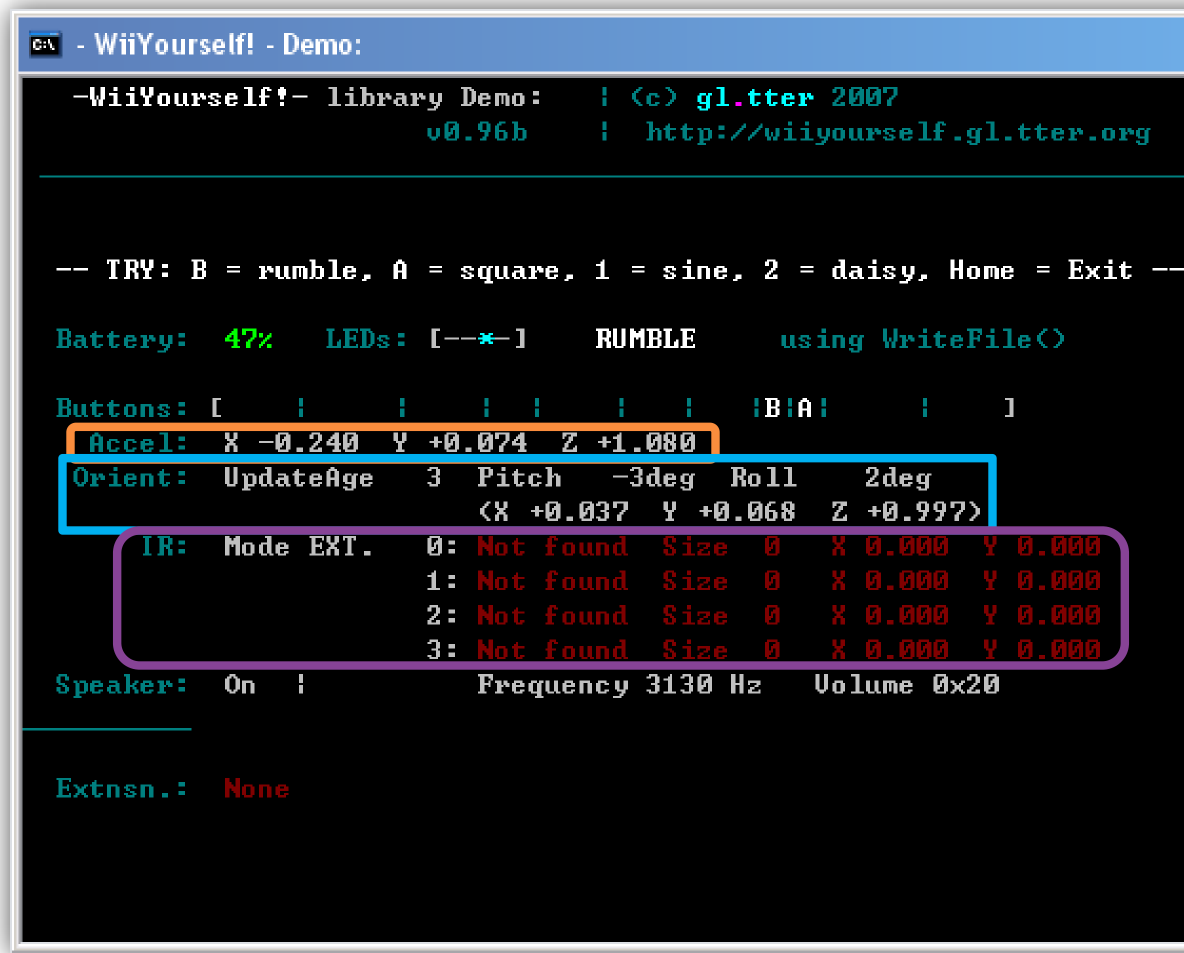
- Bluetooth as the communication channel
- Pair up with Bluetooth USB adapter to communicate
- Use Wii Remote API get the data from Wii Remote
  - *WiiYourSelf!* as the API
  - Be integrated to our classifier

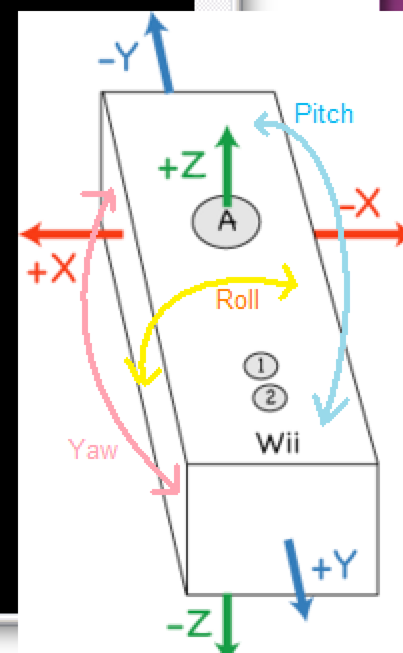
WHAT CAN BE OBTAINED FROM WII REMOTE?

- 3D Acceleration data
  - x axis, y axis, z axis
- Infra-red data
  - Used with sensor bar
  - Determine pointing position



WHAT CAN BE OBTAINED FROM WII REMOTE?





METHOD OF CLASSIFICATION

POSSIBLE CLASSIFICATION METHODS

- Tree-based & rule-based
  - Compute attributes based on motion data
    - Relative Max / Min Acceleration Time
    - Initial / Average / End Acceleration
    - Rotation angle
  - WEKA
    - Open source data mining software
    - Generate decision trees / rule with different tree-based / rule-based algorithms based on values of attributes
      - C4.5 decision tree learner

POSSIBLE CLASSIFICATION METHODS

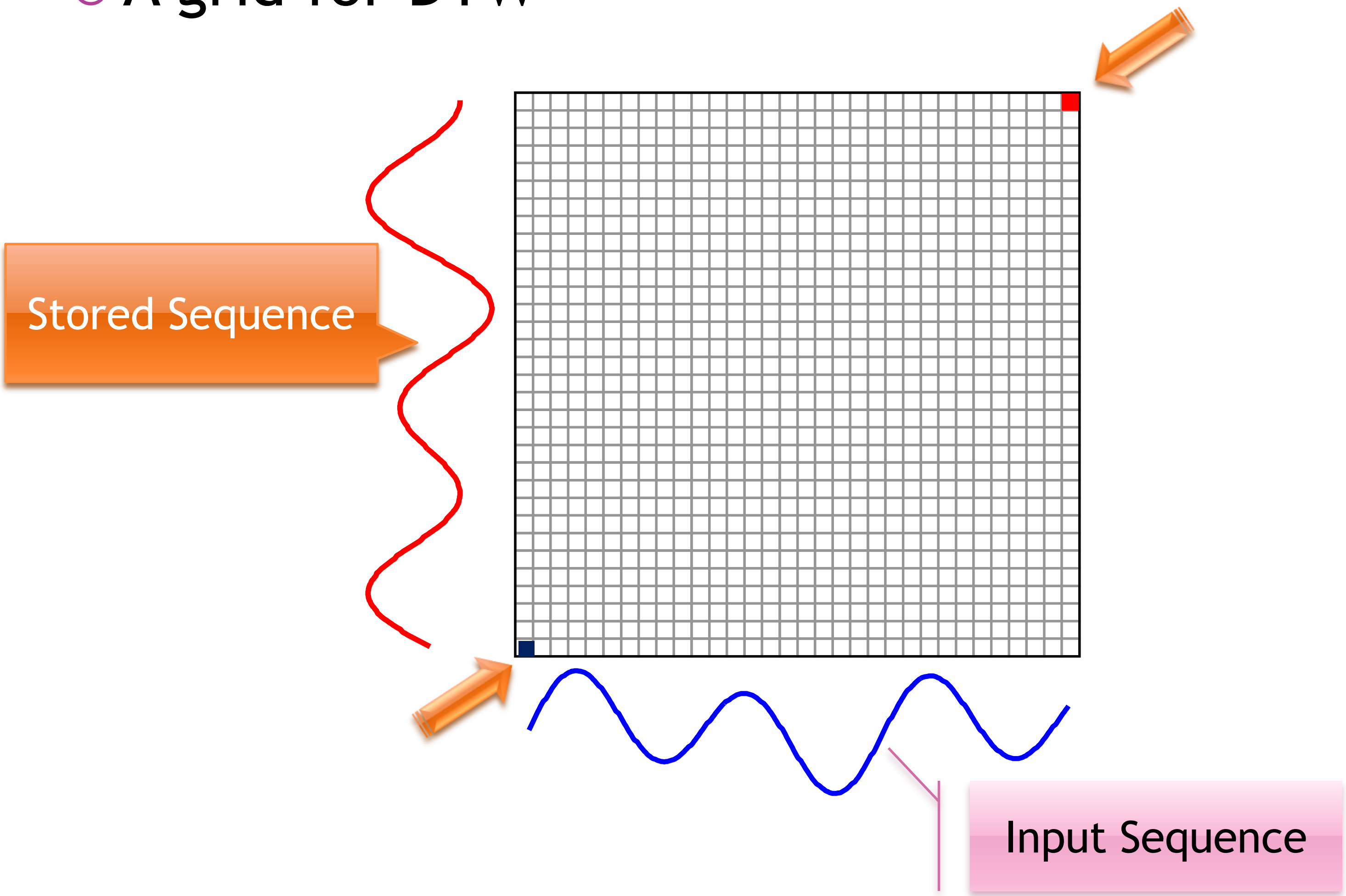
- Similarity search
  - No further computations needed
  - Compare patterns of motion data
  - Euclidean Distance Algorithm
  - Dynamic Time Warping

WORKING PRINCIPLE OF DYNAMIC TIME WARPING

- Find the best match between two given sequences.
- Sequences mapped non-linearly with the time dimension.
  - To detect similarity regardless of variations in time and speed
- To produce a matching
  - First model sequences into a 2-dimensional grid

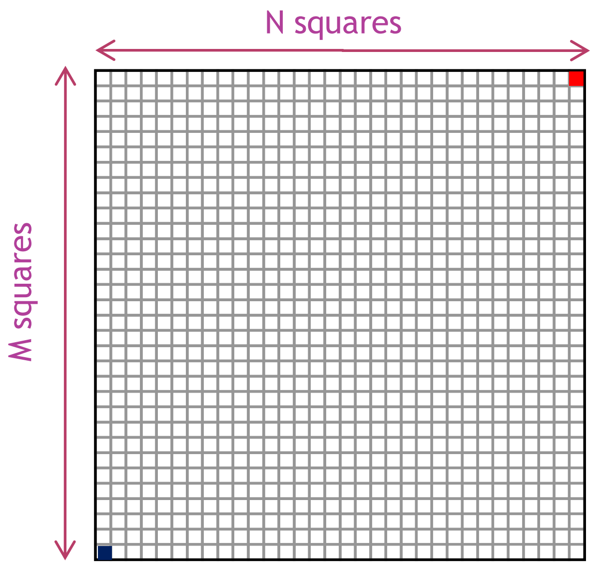
WORKING PRINCIPLE OF DYNAMIC TIME WARPING

- A grid for DTW



WORKING PRINCIPLE OF DYNAMIC TIME WARPING

- Dimension of grid
  - Determined by the total number of data points of the sequence
  - One data point is related with one row / column of squares



WORKING PRINCIPLE OF DYNAMIC TIME WARPING

- The next step is to compute the values for all the squares
- Each square stores a distance …
  - Defined by sum of 2 parts

$$\gamma(i,j) = d(s_i, p_j) + \min\left[\gamma(i-1,j-1), \gamma(i-1,j), \gamma(i,j-1)\right]$$

    - The Euclidean distance between the corresponding elements of the two sequences
    - Minimum distances among the 3 neighboring squares

WORKING PRINCIPLE OF DYNAMIC TIME WARPING

- Euclidean distance is defined as the ordinary distance that could be measured with a ruler
  - For two points *P and Q* in general n-dimension space, it is defined as:

$$\sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \ldots + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}$$

  - In particular for 3D space
    - For two 3D points, $P = (p_x, p_y, p_z)$ and $Q = (q_x, q_y, q_z)$, the Euclidean is defined as:

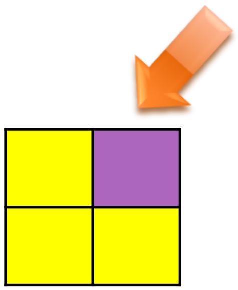
$$\sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + (p_3 - q_3)^2}$$

WORKING PRINCIPLE OF DYNAMIC TIME WARPING

- The next step is to compute the values of all the squares
- Each square stores a distance …
  - Defined by sum of 2 parts

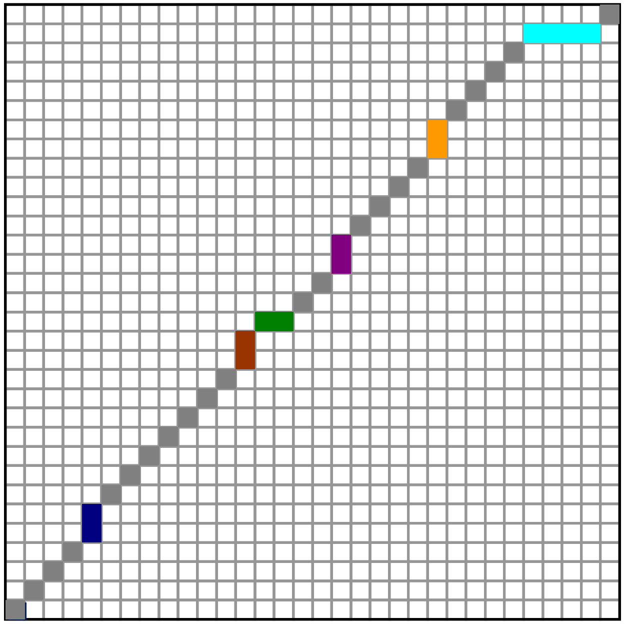
$$\gamma(i,j) = d(s_i, p_j) + \min[\gamma(i-1,j-1), \gamma(i-1,j), \gamma(i,j-1)]$$

    - The Euclidean distance between the corresponding elements of the two sequences
    - Minimum distances among the 3 neighboring squares
      - Values computed already



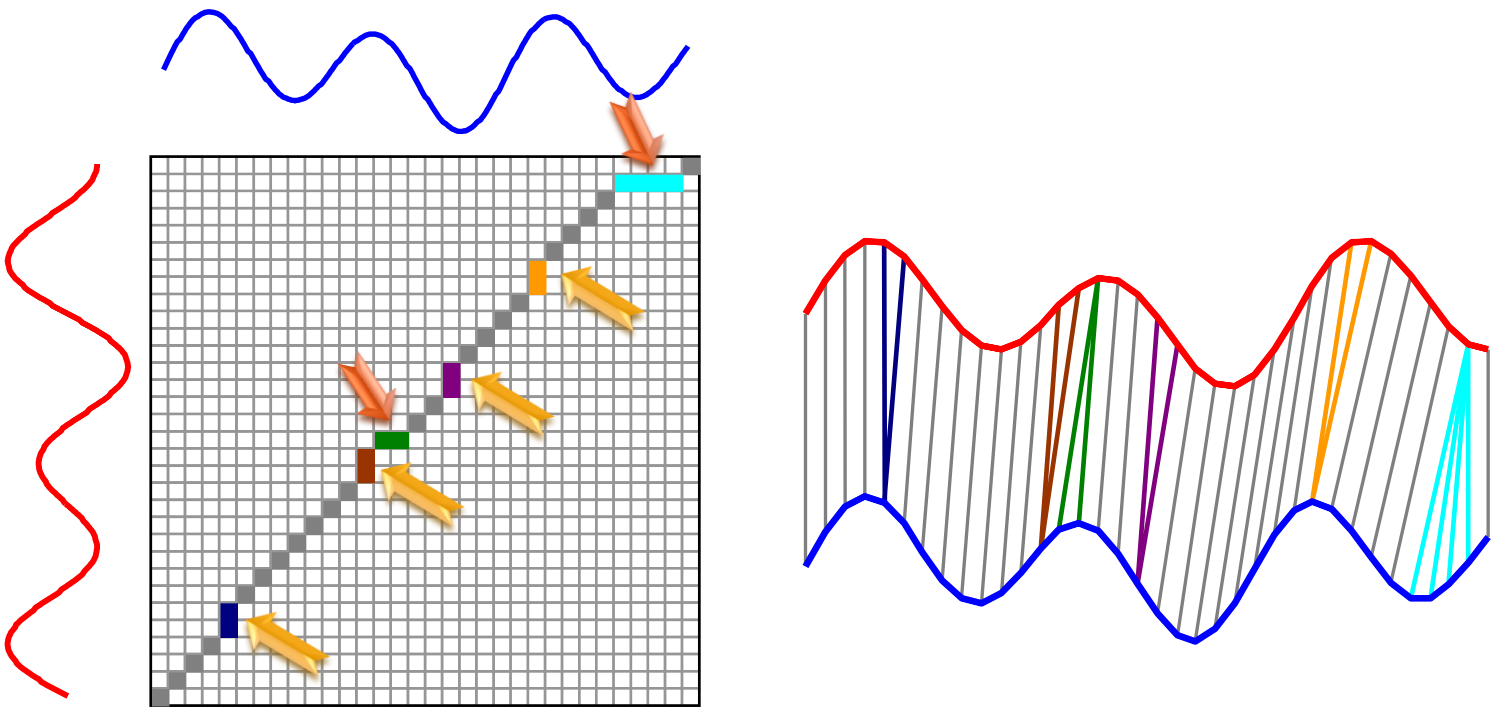
WORKING PRINCIPLE OF DYNAMIC TIME WARPING

- Next is to find a path within the grid which minimizes the total distance from the blue square to the red square



WORKING PRINCIPLE OF DYNAMIC TIME WARPING

- Warping path is found
  - Just the shortest path computed
- Defines the mapping between two sequences



HOW DO WE APPLY DYNAMIC TIME WARPING?

DATA STRUCTURE

Motion Database

2		A	Left
Reference Motion Data	Reference Motion Data	Reference Motion Data	Reference Motion Data

EXAMPLE

Time	X	Y	Z
0	1.000	2.519	0.480
15	0.720	2.667	0.320
30	0.120	2.963	0.240
45	-0.120	3.111	0.280
60	-0.600	3.519	0.440
75	-1.120	3.852	0.520
90	-1.280	4.000	0.640
105	-1.440	4.111	1.000
120	-1.680	4.148	1.160
135	-2.400	4.185	1.400
150	-2.880	4.185	1.440
165	-3.480	4.222	1.400
180	-3.280	4.222	1.280
195	-2.960	4.222	1.200
210	-2.480	4.222	1.000
225	-2.360	4.259	0.960
240	-2.360	4.259	0.840
255	-2.640	4.259	0.960
270	-3.400	4.259	1.000
285	-3.800	4.259	0.920
300	-3.480	4.296	0.840
315	-2.320	4.222	0.760
330	-2.040	4.259	0.920
345	-2.360	4.111	1.040
360	-2.520	3.519	0.840
375	-2.440	3.185	0.680
390	-2.000	2.630	0.400
405	-1.720	2.407	0.280

Reference Motion of Same Type

Time	X	Y	Z
0	1.400	1.963	0.760
15	1.240	2.111	0.760
30	0.880	2.593	0.720
45	0.440	2.926	0.760
60	0.240	3.074	0.760
75	-0.120	3.444	0.840
90	-0.360	3.667	0.840
105	-0.920	3.963	0.960
120	-1.320	4.074	0.920
135	-2.000	4.148	1.000
150	-2.520	4.185	1.200
165	-2.880	4.222	1.240
180	-3.240	4.222	1.240
195	-4.120	4.222	1.400
210	-4.280	4.222	1.240
225	-3.600	4.259	1.000
240	-2.840	4.259	1.000
255	-2.760	4.259	1.080
270	-3.120	4.296	1.280
285	-3.720	4.259	1.320
300	-4.600	4.296	1.000
315	-3.960	3.926	0.560
330	-3.360	3.593	0.440
345	-2.240	3.519	0.600
360	-1.960	3.593	0.760
375	-1.800	3.481	0.960
390	-1.880	3.222	1.000
405	-2.000	2.630	0.760
420	-1.920	2.370	0.600
435	-1.680	1.926	0.400
450	-1.520	1.704	0.320

Motion To Classify

THE INPUT DATA

Time (Millisecond)	X	Y	Z
0	1.00	2.52	0.48
15	0.72	2.67	0.32
30	0.12	2.96	0.24
45	-0.12	3.11	0.28
60	-0.60	3.52	0.44
75	-1.12	3.85	0.52
90	-1.28	4.00	0.64
105	-1.44	4.11	1.00
120	-1.68	4.15	1.16
135	-2.40	4.19	1.40
150	-2.88	4.19	1.44

APPLYING DYNAMIC TIME WARPING

Dynamic Time Warping Cost:

$$\gamma(i,j) = d(s_i, p_j) + \min\left[\gamma(i-1, j-1), \gamma(i-1, j), \gamma(i, j-1)\right]$$

Typically:

$$d(s_i, p_j) = \left| s[i] - p[j] \right|$$

In our case:

$$d(s_i, p_j) = \sqrt{(s[i].X - p[j].X)^2 + (s[i].Y - p[j].Y)^2 + (s[i].Z - p[j].Z)^2}$$

THE DTW MATRIX

0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
∞	0.7	1.3	1.6	2.3	3.3	4.8	6.6	9.1	11.9	15.3	19.3	23.6	28.2	33.7	39.3	44.3	48.5	52.7	57.3	62.4	68.3	73.4	77.9	81.3	84.5	87.5	90.5	93.5	96.4	99.1	101.8
∞	1.8	1.6	1.7	2.2	2.9	4.2	5.7	7.9	10.5	13.6	17.3	21.4	25.7	30.9	36.2	40.9	44.8	48.7	53.0	57.8	63.4	68.3	72.5	75.6	78.4	81.2	83.9	86.7	89.3	91.9	94.3
∞	3.5	3.1	2.6	2.3	2.7	3.5	4.5	6.2	8.1	10.6	13.7	17.1	20.8	25.4	30.1	34.1	37.4	40.7	44.3	48.5	53.5	57.7	61.3	63.7	65.9	68.1	70.2	72.4	74.6	76.7	78.7
∞	5.5	4.9	3.8	3.1	2.9	3.3	4.2	5.5	7.2	9.5	12.2	15.4	18.8	23.1	27.5	31.3	34.3	37.3	40.7	44.6	49.3	53.2	56.5	58.7	60.6	62.5	64.4	66.4	68.4	70.3	72.3
∞	8.0	7.2	5.6	4.3	3.9	3.6	3.8	4.6	5.6	7.3	9.4	11.9	14.8	18.5	22.3	25.5	27.9	30.3	33.0	36.4	40.5	43.9	46.6	48.3	49.7	51.0	52.4	54.1	55.9	57.8	59.8
∞	11.2	10.2	7.9	6.2	5.5	4.7	4.4	4.3	4.8	5.9	7.5	9.4	11.7	14.8	18.1	20.6	22.5	24.3	26.4	29.2	32.7	35.6	37.8	39.0	39.9	40.8	41.9	43.4	45.1	47.1	49.3
∞	14.6	13.3	10.5	8.2	7.3	6.0	5.4	4.8	4.6	5.4	6.8	8.5	10.6	13.5	16.6	19.0	20.6	22.2	24.1	26.7	30.0	32.7	34.8	35.9	36.7	37.5	38.6	40.1	41.9	44.0	46.3
∞	18.1	16.7	13.3	10.4	9.3	7.5	6.6	5.4	4.8	5.2	6.3	7.8	9.6	12.3	15.1	17.3	18.7	20.0	21.8	24.1	27.2	29.8	31.9	32.9	33.7	34.4	35.4	37.0	38.9	41.1	43.7
∞	21.9	20.2	16.3	12.9	11.5	9.2	8.0	6.2	5.2	5.1	6.0	7.2	8.7	11.2	13.8	15.7	16.9	18.0	19.4	21.5	24.4	26.8	28.7	29.7	30.5	31.2	32.1	33.7	35.6	37.9	40.5
∞	26.4	24.5	20.0	16.1	14.5	11.7	10.2	7.7	6.4	5.7	5.4	5.9	6.7	8.4	10.3	11.6	12.2	12.7	13.4	14.8	17.0	18.8	20.3	21.3	22.3	23.3	24.5	26.2	28.3	30.8	33.7
∞	31.2	29.1	24.2	19.7	17.9	14.6	12.9	9.8	8.0	6.7	5.8	5.6	6.0	7.2	8.6	9.5	9.9	10.3	10.6	11.5	13.3	14.7	15.9	17.2	18.5	19.8	21.3	23.2	25.4	28.2	31.2
∞	36.7	34.4	28.9	23.9	21.8	18.1	16.1	12.4	10.3	8.2	6.8	6.2	5.9	6.5	7.3	7.7	8.5	9.3	9.7	9.9	11.1	12.1	13.3	14.9	16.7	18.6	20.5	22.8	25.3	28.4	31.6
∞	41.9	39.4	33.4	27.9	25.5	21.4	19.1	14.8	12.3	9.5	7.5	6.6	5.9	6.7	7.5	7.7	8.3	8.8	9.0	9.4	10.8	11.8	12.9	14.3	15.9	17.5	19.3	21.4	23.8	26.7	29.9
∞	46.8	44.1	37.6	31.5	29.0	24.4	21.8	16.8	13.9	10.5	8.0	6.7	6.2	7.1	8.0	8.2	8.0	8.2	8.4	9.2	10.8	12.0	12.9	14.0	15.3	16.7	18.2	20.1	22.3	25.1	28.1
∞	51.3	48.4	41.3	34.7	31.9	26.8	24.0	18.4	15.1	11.0	8.2	7.1	7.0	7.9	8.9	9.1	8.3	8.3	8.9	9.7	11.3	12.4	13.2	13.7	14.6	15.6	16.8	18.4	20.4	22.9	25.7
∞	55.7	52.6	45.0	37.8	34.8	29.2	26.0	19.9	16.2	11.4	8.5	7.7	7.9	8.8	9.8	10.1	8.8	8.7	9.1	10.3	11.9	13.0	13.7	14.1	14.5	15.5	16.6	18.3	20.3	22.8	25.5
∞	60.1	56.8	48.6	40.9	37.6	31.6	28.1	21.4	17.2	11.8	8.9	8.4	8.7	9.8	10.8	11.1	9.3	9.2	9.6	10.5	12.6	13.6	14.2	14.5	14.8	15.5	16.6	18.3	20.3	22.7	25.5
∞	64.8	61.2	52.5	44.3	40.8	34.3	30.5	23.1	18.6	12.4	9.2	8.8	9.1	10.3	11.4	11.7	9.5	9.3	9.7	10.7	12.5	13.9	14.7	15.2	15.5	16.0	16.8	18.4	20.4	22.8	25.6
∞	70.1	66.3	57.1	48.4	44.6	37.6	33.6	25.6	20.6	13.8	10.1	9.3	9.1	9.9	10.8	11.0	10.1	10.0	9.7	10.2	11.4	12.2	13.0	14.5	16.1	17.3	17.8	18.9	20.8	23.3	26.1
∞	75.8	71.8	62.1	52.8	48.8	41.4	37.1	28.5	23.1	15.6	11.4	10.3	9.7	9.6	10.2	10.4	11.1	11.0	10.5	10.1	10.9	11.4	12.4	14.1	16.1	18.2	19.4	20.3	21.6	24.0	26.8
∞	81.2	77.0	66.8	57.0	52.7	44.9	40.3	31.1	25.3	17.1	12.4	11.0	10.2	10.5	10.5	10.4	11.1	11.8	11.1	10.7	11.3	11.6	12.3	13.8	15.4	17.3	19.2	21.5	22.7	24.6	27.3
∞	85.6	81.1	70.4	60.0	55.5	47.2	42.3	32.5	26.3	17.6	12.9	11.8	11.2	12.1	12.5	11.7	11.0	11.5	12.0	12.2	13.0	12.9	12.9	13.0	13.7	14.6	15.8	17.4	19.3	21.7	24.4
∞	89.7	85.1	73.8	62.8	58.1	49.3	44.1	33.7	27.1	17.7	13.5	12.7	12.5	13.3	14.3	13.3	11.8	11.7	12.7	13.7	14.7	14.9	14.4	13.7	13.7	14.5	15.6	17.2	19.1	21.5	24.2
∞	94.0	89.2	77.3	65.9	60.9	51.7	46.1	35.1	28.1	18.1	13.7	13.2	13.4	14.3	15.3	14.5	12.3	12.2	12.5	13.9	16.0	16.4	15.7	14.4	14.4	14.5	15.5	17.1	18.9	21.3	23.9
∞	98.3	93.2	80.9	68.9	63.7	54.1	48.3	36.8	29.4	18.9	14.5	14.1	14.3	15.2	16.2	15.9	13.1	13.0	13.2	14.0	16.2	17.5	16.6	14.8	15.0	15.1	15.3	16.3	17.6	19.5	21.6
∞	102.3	97.0	84.2	71.8	66.4	56.4	50.5	38.5	30.9	20.0	15.6	15.4	15.6	16.4	17.4	17.5	14.3	14.2	14.4	15.0	16.5	17.9	17.7	15.2	15.4	15.7	15.8	16.0	16.9	18.4	20.2
∞	105.8	100.3	87.1	74.3	68.7	58.5	52.4	40.3	32.6	21.6	17.4	17.4	17.6	18.4	19.3	19.7	16.2	16.1	16.4	16.9	18.2	18.8	19.3	16.1	16.2	16.5	16.6	16.1	16.3	17.1	18.1
∞	109.0	103.4	89.8	76.6	70.8	60.5	54.4	42.2	34.4	23.5	19.6	19.8	19.9	20.8	21.7	22.1	18.5	18.4	18.6	19.3	20.5	20.9	20.9	17.4	17.4	17.5	17.6	16.7	16.5	16.8	17.5

TAKE DURATION INTO CONSIDERATION

	Duration (Millisecond)
Reference Motion of Same Type	405
Motion to Classify	450
Average	**427.5**

7.2	19.1	21.5	24.2
7.1	18.9	21.3	23.9
6.3	17.6	19.5	21.6
6.0	16.9	18.4	20.2
6.1	16.3	17.1	18.1
6.7	16.5	16.8	17.5

DTW Error(Similarity) = 17.534 ÷ 427.5 = 0.041

DTW ERROR RESULT

Motion Name	Error
Left	0.041
TopRightCircle	0.245
LeftCircle	0.196
Dash	0.273
CircleUp	0.162

Classified As: Left

TUNING

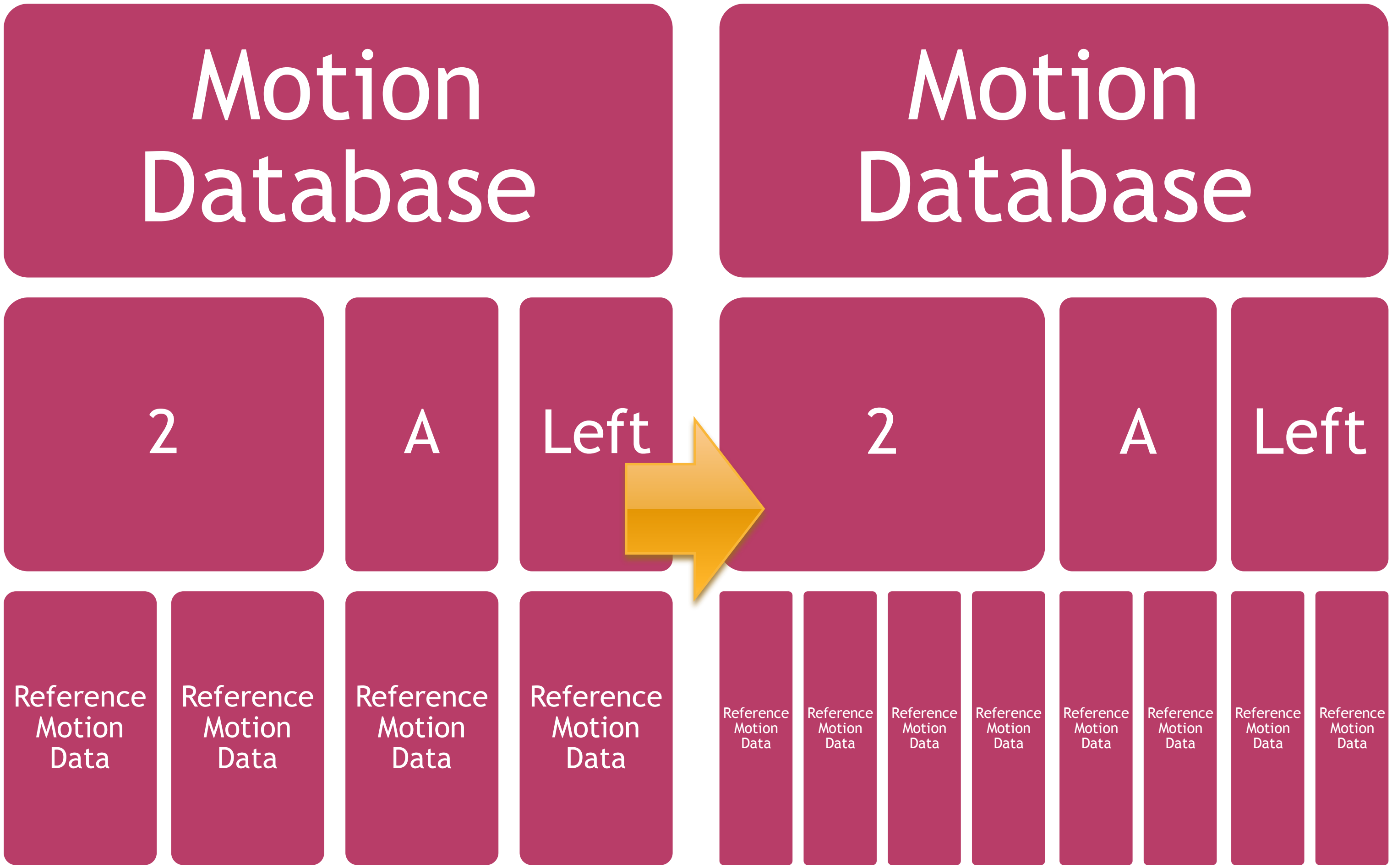
REASONS FOR LOW ACCURACY?

- Performed not in the same way
- Button pressed in wrong moment
- Similar motions
- Bad quality of reference motion

ANALYZED METHODS OF TUNING

- Number of Reference Motion
- Error Compensation
- Average of Motion Data as Reference Motion

NUMBER OF REFERENCE MOTION

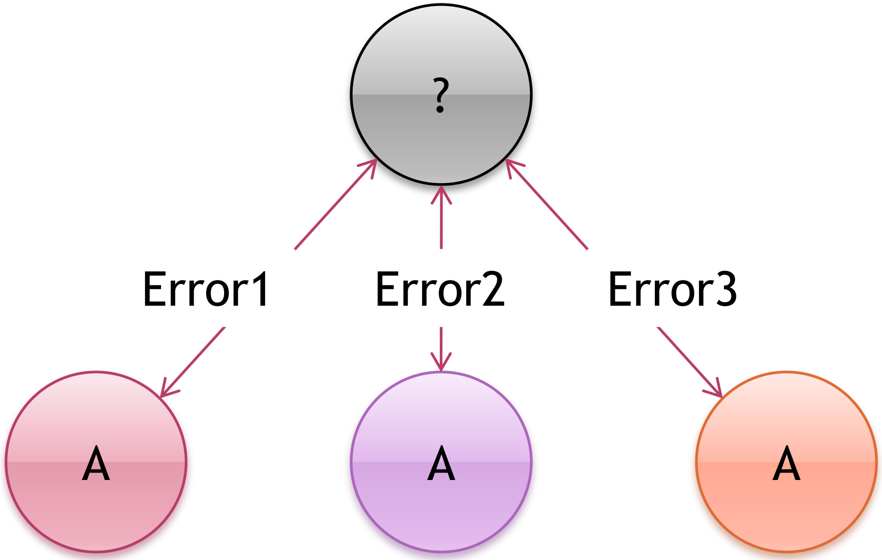


NUMBER OF REFERENCE MOTION

- Increase number of reference motions of the same type of motion
- Run Dynamic Time Warping on more reference motions

NUMBER OF REFERENCE MOTION

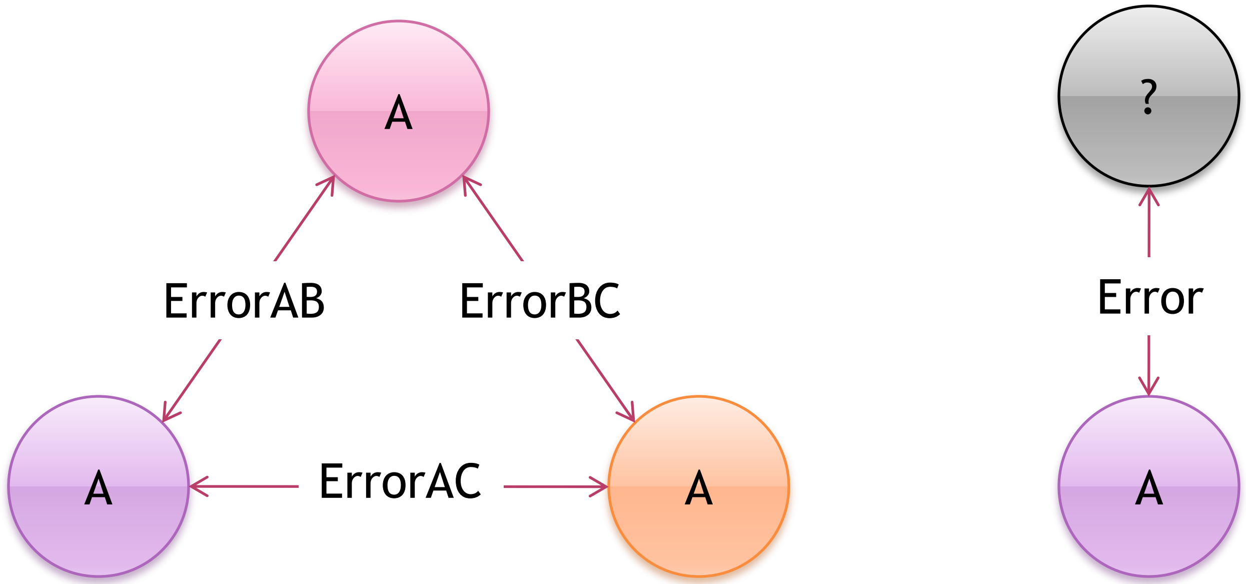
- float dtwError(Motion A, Motion B);



- Similarity = (Error1+Error2+Error3) ÷ 3

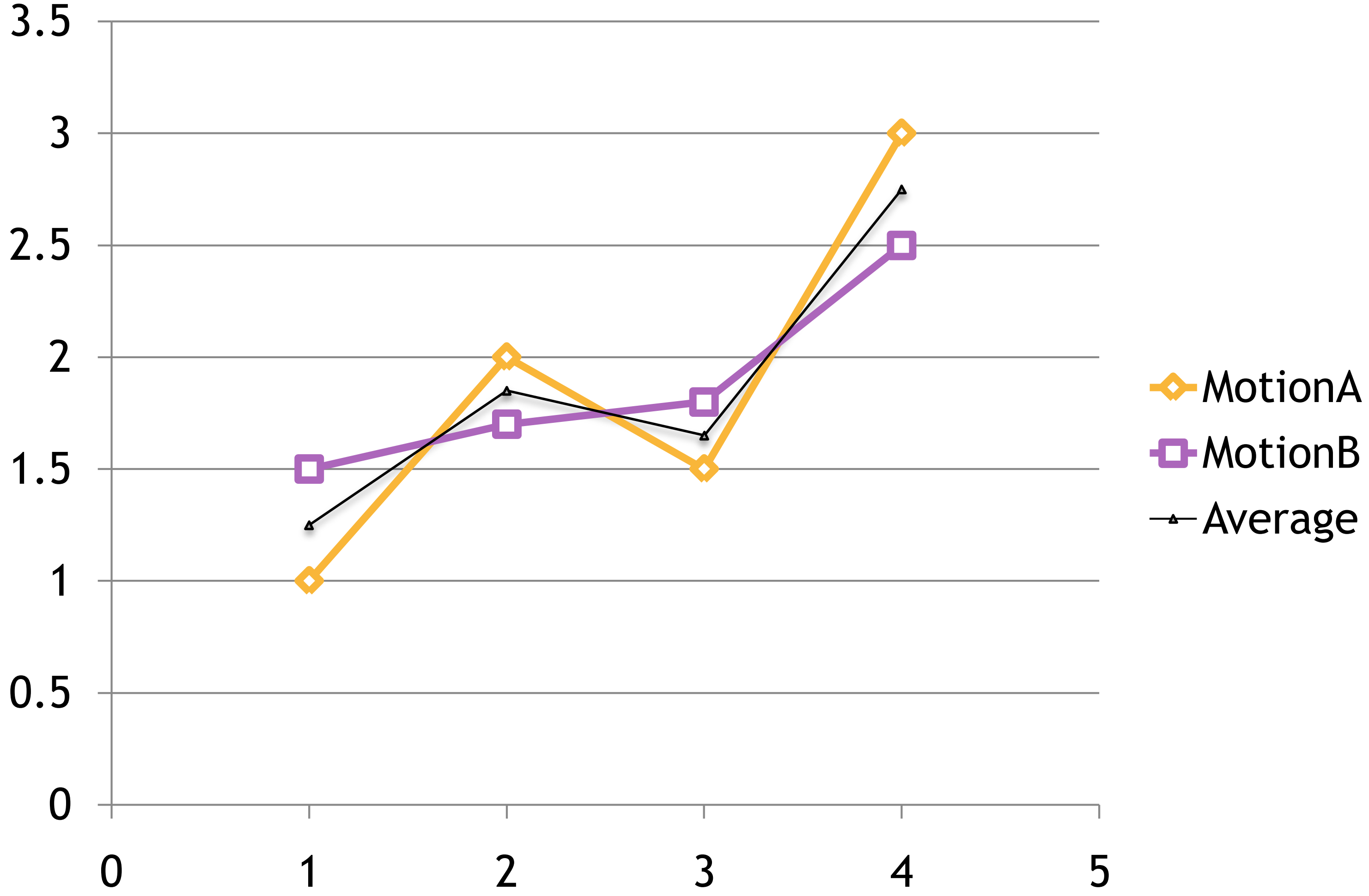
ERROR COMPENSATION

- float dtwError(Motion A, Motion B);



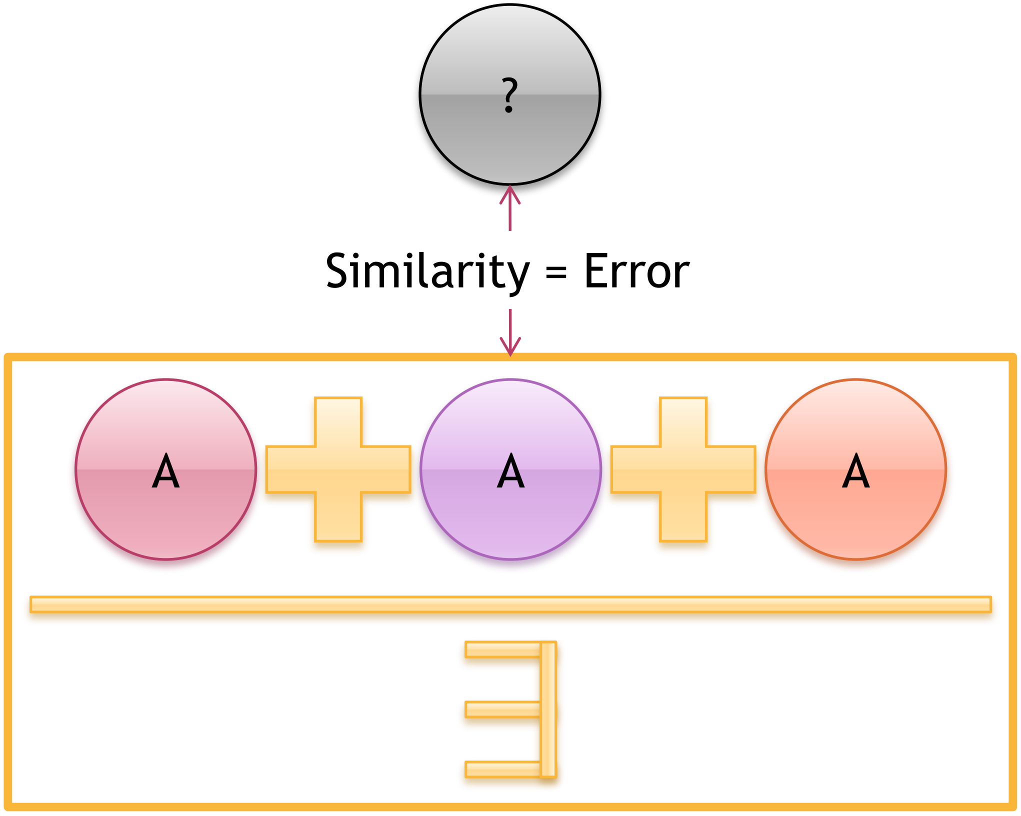
- AllowedError = (ErrorAB+ErrorBC+ErrorAC)÷3
- Similarity = Error – AllowedError

AVERAGE OF MOTION DATA AS REFERENCE MOTION



AVERAGE OF MOTION DATA AS REFERENCE MOTION

- float dtwError(Motion A, Motion B);



MOTIONS USED

- Five Directions
- Cursive Writing Letters
- Discrete Motions
- Numbers
- Dao Motions

CONFIGURATIONS USED

	Reference Motion (R)	Error Compensation (E)	Average Motion (A)
Configuration 1	1		
Configuration 2	1	3	
Configuration 3	3		
Configuration 4	3	3	
Configuration 5	5		
Configuration 6	5	3	
Configuration 7			3
Configuration 8		3	3

CLASSIFICATION ACCURACY

Configuration	Five Directions (By 3 People)	Letter	Discrete	Dao Motions	Number (By 3 People)	Average
R=1	92.57%	100%	99.51%	100%	85.69%	95.55%
R=1,E=3	95.59%	100%	98.52%	100%	90.41%	96.90%
R=3	91.06%	100%	99.50%	100%	95.47%	97.21%
R=3,E=3	91.92%	100%	99.50%	100%	98.13%	97.91%
R=5	94.18%	100%	99.50%	100%	97.07%	98.15%
R=5,E=5	94.90%	100%	99.50%	100%	98%	98.48%
A=3	91.52%	100%	99.50%	100%	97.47%	97.70%
E=3,A=3	92.66%	100%	99.50%	99.29%	90.41%	96.37%

EFFECT OF DIFFERENT TUNING

	Preparation?	Time?	Space?
Reference Motion	No	Largely	Largely
Error Compensation	Yes	Slightly	Slightly
Average Motion	Yes	No Effect	No Effect

CONCLUSION

- Depend on requirement in choosing
- Hybrid can be considered

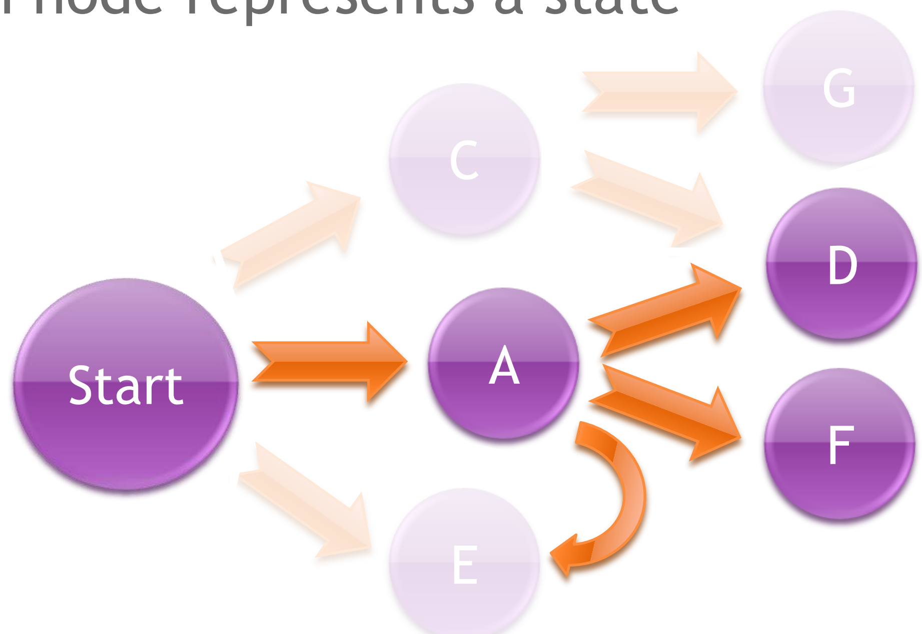
FUTURE WORK

FUTURE WORK

- A game demo
- Educational software of Chinese Martial Art
- Motion Finite State Machine
  - Making use of game rules to minimize the amount of possible candidates in classification
    - Limit the type of motions that can be followed

FUTURE WORK

- Motion Finite State Machine
  - Each node represents a state



  - Allow players to perform motions in any sequence but still maintain high accuracy

FUTURE WORK

- A game demo
- Educational software of Chinese Martial Art
- Motion Finite State Machine
- Early Recognition
- Two hand weapons using Nunchuk
- “Button-less”

Q & A