Perception-link behavior model : supporting a novel operator interface for a customizable anthropomorphic telepresence robot

Gu, William Yuanlong

2017

Gu, W. Y. (2017). Perception-link behavior model : supporting a novel operator interface for a customizable anthropomorphic telepresence robot. Doctoral thesis, Nanyang Technological University, Singapore.

<http://hdl.handle.net/10356/72753>

<https://doi.org/10.32657/10356/72753>

**PERCEPTION-LINK BEHAVIOR MODEL –
SUPPORTING A NOVEL OPERATOR INTERFACE
FOR A CUSTOMIZABLE ANTHROPOMORPHIC
TELEPRESENCE ROBOT**

GU YUANLONG WILLIAM

School of Mechanical and Aerospace Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirement for a
Doctor of Philosophy

2017

This page is intentionally left blank.

Abstract

The customizable anthropomorphic telepresence robot (CATR) is viewed as a potential medium for increasing the social presence in a computer-mediated communication medium towards face-to-face communication. However, there are issues in teleoperation that might deteriorate the efficiency and expressiveness of the CATR. These issues include decoupling, network condition, and limited situation awareness. Decoupling is an event where the operator forgoes part of his nonverbal modalities from a conversation. The second issue is the network condition that can be affected by many factors. For the last issue, the operator's situational awareness of the remote environment is limited, and it might pose a danger to the remote participants. The outcome is to have a compatible interface for the CATR so that it can perform effectively without degrading its expressivity and overloading the cognitive load of the operator.

From a review of the literature, the CATR interface should encapsulate the following: high expressiveness, low cognitive loads, have a deliberate safety mechanism, and ability to substitute. The CATR must express realistic nonverbal cues similar to the operator so that the CATR is representing the operator, in terms of personality and mood. In addition, the interface should not add to the operator's cognitive load, and it should be intuitive for the operator to use. Next, the CATR should not intimidate or endanger the interactants without distorting the meaning of the gestures. Lastly, the interface should promote a seamless conversation, and the solution is to substitute any missing or undesirable modalities that best fit the current contextual information. None of the existing reviewed interfaces is complete and comprehensive for CATR.

The aim of the perception-link behavior model (PLBM) is to overcome the above problems. The PLBM adopts a contactless natural interface as the acquisition approach, which allows the operator to gesticulate expressive gestures using a lower cognitive load. Furthermore, the PLBM employs an unsupervised approach to extract the distributed features, which is more consistent and complete. The PLBM transforms the input signal into a distributed output, which is more informational and expressive. The PLBM is also capable of reconstructing the input signal given the encoded data. In addition, the encoded data can be re-tuned to create a variance of the input signals. Finally, the PLBM can also associate the nonverbal cues from both the operator and audience in real time. This associating function empowers the operator to substitute any undesirable gestures with a coherence gestures retrieved from the knowledge-based. In short, the CATR with PLBM maintains a high-degree of social behavior and social cues; moreover, the PLBM also delivers a seamless communication without compromising the operator's cognitive workload and audience's safety.

By optimizing and integrating the above modules, the PLBM has numerous advantages over other existing telepresence robot interfaces, e.g. imitation approach. The PLBM can conceal and replace unwanted gesture of the operator during a conversation. Secondly, the PLBM can insert missing data when there is a bad network connection. Lastly, the PLBM can re-tune the incoming data so that the CATR can deliberative and smoothly avoid a collision. In conclusion, the PLBM is superior to existing interfaces given the above scenarios.

This page is intentionally left blank.

Acknowledgement

The author would like to express his deepest gratitude to Associate Professor Gerald Seet Gim Lee, his supervisor, for his patience and understanding; especially, when it comes to writing this report. Along this journey, he provided the author with numerous advice that made this thesis possible.

The author would also like to thank Burhan, Pang W. C., Ratmin A. K., Tju H. S., and Wong C.Y. for creating a lively and warmly working environment that keeps him going even in the darkest moment.

The author also appreciates the financial grant and the academic advice from Institute for Media Innovation.

Finally, the author would like to thank his parents for their absolute support and understanding during the last five years.

This page is intentionally left blank.

Table of Contents

Abstract	i
Acknowledgement	iii
Table of Contents	v
List of Figures	ix
List of Tables	xv
List of Symbols	xvi
List of Acronyms	xix
1 Introduction	1
1.1 Background	2
1.1.1 Benefits of face-to-face communication	2
1.1.2 Benefits of computer mediated communication	2
1.1.3 Telepresence robot and its benefits	3
1.1.4 Customizable anthropomorphic telepresence	4
1.1.5 Benefits of gesturing in CATR	5
1.1.6 Problems of teleoperating a CATR	6
1.2 Research gaps	8
1.3 Aim of the research	8
1.4 Significance of the research	10
1.5 Perception-link behavior model	12
1.5.1 Detail descriptions of the PLBM dealing with the various issues	14
1.5.2 Summary of the PLBM	19
1.6 Outline of remaining chapters	19
2 Literature review	20
2.1 Social presence	21
2.1.1 Difference between telepresence and social presence	21
2.1.2 Social presence: Appearance and behaviors	22
2.2 Existing control interfaces	25
2.2.1 Evaluation of the existing interface	25
2.2.2 Summary of the research gaps	25
2.2.3 Desirable configurations in term of the characteristic	28
2.2.4 Optimal configurations for CATR	30

3	Methods: Perception-link behavior model	32
3.1	Perception-link behavior model	33
3.1.1	Definition of the model.....	33
3.1.2	Components of the model	33
3.1.3	Achieving the four parameters.....	36
3.1.4	CATR with PLBM to handle the teleoperating issues.....	38
3.1.5	Other advantages for PLBM in CATR	40
3.2	Model for encoder	43
3.2.1	Background.....	43
3.2.2	Architecture of encoder for gestures (CNN-RBM)	48
3.2.3	Data and its preprocessors	49
3.2.4	Evaluation criteria.....	52
3.3	Model for decoder	54
3.3.1	Modified LSTM (Ratio BPTT).....	54
3.3.2	Architecture of decoder for gestures.....	58
3.3.3	Data and its preprocessors	59
3.3.4	Evaluation criteria.....	60
3.4	Model for associator.....	62
3.4.1	Adaptive resonance theory and its extension.....	62
3.4.2	Architecture of decoder for gestures (multi-channel ARAM).....	66
3.4.3	Data and its preprocessors	68
3.4.4	Evaluation criteria.....	71
4	Experiments and results.....	72
4.1	Encoder.....	72
4.1.1	Using automated feature extractor to obtain spatiotemporal distributed output for gesture.....	72
4.1.2	Comparison with other feature extractors.....	74
4.1.3	Selection of the parameters	78
4.2	Decoder	79
4.2.1	Regression with the encoded data.....	79
4.2.2	Regression with the re-tuned encoded data	81
4.2.3	Comparison of the LSTM with other feature extractors	84
4.2.4	Selection of the parameters and the effects of high neuron size.....	85

4.3	Associator.....	88
4.3.1	Multimodalities spaces and the relationships between different modalities.....	88
4.3.2	Regression with retrieved encoded data given the facial features	89
4.3.3	Selection of the parameters	92
4.4	CATR with a PLBM	94
4.4.1	Inserting a valid gesture style in decoupling situation with different sequences	94
4.4.2	Reconstructing gestures in an unstable connection situation.....	96
4.4.3	Reconstructing gestures in a tight situation	100
5	Conclusion and discussion	107
5.1	Contributions.....	108
5.1.1	Low operator's cognitive load for using the CATR	108
5.1.2	Expressive CATR's behaviors to imitate personal gestures	108
5.1.3	Handling decoupling situation with associator in PLBM	110
5.1.4	Generating expressive and smooth CATR gesture during intermittent connection.....	112
5.1.5	Collision-free and expressive CATR gesture using PLBM with evolutionary algorithm.....	113
5.1.6	Conclusion	114
5.2	Limitations	115
5.3	Future Works for some of the limitations	116
	Tables of appendices.....	117
	Appendix A Nonverbal communication	118
	Appendix B Local and distributed representation	121
	Appendix C Automated feature extraction	126
	Appendix D Existing telepresence robot interfaces.....	130
	D.1 Traditional telepresence robot with traditional interface	130
	D.2 MeBot with master-slave interface	131
	D.3 Hasegawa's bot with natural interface	132
	D.4 Park's bot with recognition-synthesis interface	132
	Appendix E Expressive gestures on virtual agent	134
	Appendix F Characteristics and effects of an CATR interface	138
	Appendix G Supporting videos for the various experiments.....	142
	Reference	143

This page is intentionally left blank.

List of Figures

Figure 1-1 Traditional telepresence robot.....	3
Figure 1-2 A - PRoP with a pointer [3], B - RP-Vita with a pan-tilt unit [4], C - MeBot with 3 DOF manipulators [5], and D - Hasegawa's bot with 4 DOF manipulators [6].....	4
Figure 1-3 Classification of telepresence robot w.r.t humanoid robot (Left), and EDGAR-1 [9] being inhabited (Right).....	5
Figure 1-4 The relationship between gesture intensives and perception of extraversion by Neff et al. [11] (left). Interactive gestures by Bavelas et al. [13] (right).	6
Figure 1-5 The three possible teleoperating issues while operating a CATR using natural interface.....	7
Figure 1-6 An obstacle avoidance situation, where there are two possible outputs: deliberative and reactive.	10
Figure 1-7 An operator was gesturing and conversing to participant "A" at t1 (first row). The same operator was gesturing a different gesture, and he was speaking to participant "B" at t2 (second row). The operator was using the keyboards and mouse while talking to participant "A" or "B" at t3 (third row).	11
Figure 1-8 Concept of perception-link behavior.....	12
Figure 1-9 The scatter distribution of nine gestures in the 1 st (y) and 2 nd (x) principal components. The principal components were formulated using principle component analysis (PCA).	14
Figure 1-10 The output of a gesture decoder. There are nine different encoded data at each interval, which is 100 time steps; as a result, the decoded gestures have nine different styles.....	15
Figure 1-11 Three overlapped skeleton points with boundary constraints. The first and second columns are the actual and the reconstructed gesture without considering the constraints. The third column is the output with constraints considered.	16
Figure 1-12 On the left, the operator was reacting to an audience in four distinct gestures style with four different facial expressions, ABCD. On the right is a simulated decoupling situation, the CATR's gesture was synthesized based on the facial expression in a different sequence, CADB.	17
Figure 1-13 Segment of the three different signals. The first signal is the actual user gesture Y-motion in respective to the right wrist. The second signal is the PLBM reconstructed gestures given an uninterrupted encoded data. The second signals are the reconstructed gesture using the previous encoded data whenever there is a missing data. The third signal is the gesture signal from spatial imitation.	18
Figure 2-1 The individual components in the presence balloon	21
Figure 2-2 MeBot [5] (partial anthropomorphic telepresence robot) and EGDAR [9] (customizable anthropomorphic telepresence robot)	22
Figure 2-3 Layout of Networked Minds social presence by Biocca et al. [21].	23

Figure 2-4 The variance between different spatial parameter for an arms motion – extending and contracting towards the chest by Hartmann [16].....	28
Figure 2-5 Various input interfaces	29
Figure 2-6 The four input (blue) and the 15 output (black or green) for a cheering motion by Lau [19].....	30
Figure 3-1 Block diagram of perception-link behavior model, and the unreliable signals are denoted as $\mathbf{v}$, while the reliable signals are denoted as $\mathbf{u}$. The blue and red coding are referring to operator's and audience's signal respectively. The green coding is the decoded signals.	33
Figure 3-2 The possible output of each component in a PLBM.	35
Figure 3-3 Three different case scenarios for the PLBM	39
Figure 3-4 Perception-link behavior model integrated with telepresence robot system....	42
Figure 3-5 Procedures to get the visual words.....	43
Figure 3-6 Three objects with their respective histogram	44
Figure 3-7 Node connections for A) Boltzmann machine and B) Restricted Boltzmann machine	45
Figure 3-8 Convolution neural network with discriminative model.....	47
Figure 3-9 Four layers encoder model	48
Figure 3-10 Selected features for the experiment	50
Figure 3-11 Visualization of inter distance and intra distance between three clusters of data.	52
Figure 3-12 Forward pass of a LSTM cell with input, output, and forget gates.....	54
Figure 3-13 Backward pass of a LSTM cell with input, output, and forget gates	56
Figure 3-14 Decoder model when rollout with time.....	59
Figure 3-15 The Q table and the optimal solution generated by the dynamic programming.	60
Figure 3-16 The DTW algorithm realigns input signal to a target signal, and the result is the realigned input.....	61
Figure 3-17 ART (left) and ARAM (right).....	65
Figure 3-18 Three main processes of the associator: a) slow recoding, b) fast learning, and c) recalling	65
Figure 3-19 The convoluted face features of eleven subjects on the first two PCA axis (left). The convoluted face features of the eleventh participant with the neutral and six different expressions (right).	69
Figure 3-20 Two sample sequence of data from each dataset: identity dataset (left) and expression dataset (right)	70
Figure 4-1 Convolution features for CNN (left) and the fifth convolution feature (right)	72

Figure 4-2 Encoded output given a segment of gesture with two different styles – 2D spatiotemporal distributed output (top) and specific features (bottom) (Refer to Appendix G for the video).....	73
Figure 4-3 The top figure shows the location of the coincided peaks of multiple classes and the multi-peaks for a specific class for the raw data. The bottom figure shows the histogram of CNN with the coincided peaks of two specific classes at PCA1, and the same classes being separated in PCA2.	75
Figure 4-4 Distribution of classes in their new vector space: Spatial transformation and CNN. Each distribution comprises of two histograms and a 2D scatter plot. The two histograms estimate the class distribution for the data on the first and second PCA components, which the x-axis and y-axis respectively.	77
Figure 4-5 A zoom in on Figure 4-7 to show the convergence of the periodic signal (Refer to Appendix G for the video).....	79
Figure 4-6 A zoom in on Figure 4-7 to show the difference in behavior between the input and the internal state.	79
Figure 4-7 Nine continuous synthesized gestures with 400 gate size. Each synthesized gesture is about 100 frames in duration, which is about 20 seconds for five fps. There are six rows of data where their x-axis represents the time. The first row shows the three of the normalized output signals, which the x-y-z signals of the right wrist. The second to fourth rows display three gates activation: input, forget, and output gates respectively. The fifth row is the cell status, while the last row is the encoded data. At the top, the detail skeleton view displays a segment of the synthesized gestures from 375 to 425, where each body encapsulates five frames of data	80
Figure 4-8 Spatial difference between two encoded data: original and additive of noise. The original encoded data (blue) has a signal with a height different of 0.269, and the modified encoded data (red) has a signal with a height different of 0.418.....	81
Figure 4-9 Temporal difference between two encoded data in term of cycle time. The original encoded data (blue) produced a signal with a cycle time of 15 steps (0.333fps), while the encoded data with noise (red) produced a signal with a cycle time of 13 steps (0.385fps).	82
Figure 4-10 Comparison between original (blue) and the neighbors (red) reconstructed-gestures with 0.3 and 0.7 gain values. The first column represents the gestures motion for the right wrist in the three two-dimensional spaces, and three subgraphs represent the front, top, and side view. The second column displays right y-axis motion with respect to time. The last column shows a set of elements of the modified (red) and original (blue) encoded data. The figure has two main rows, and each main row has a unique noise gains.	83
Figure 4-11 The DTW_MSE results gathered from a permutation of the four factors with three replicas.	87
Figure 4-12 The visualization of the features spaces from Figure 4-15, using PCA. The three axis of the figure are the first three PCA components. In this figure, the train data	

are the blue dots, the pruned weights are the red diamond, and the learnt weights are the grey circles. (Refer to Appendix G for the video)	88
Figure 4-13 The operator's gesture signal and the reconstructed gesture signal.	89
Figure 4-14 On the top are the operator's encoded gesture data and the retrieved encoded gesture data. On the bottom are the activation weight during learning and during recalling.....	90
Figure 4-15 The training and recalled sequence from an expression dataset. In learning mode, the input is all the feature fields, and output is the winner nodes. During the recall mode, the input is only the facial feature fields, and the output is the retrieved gesture encoded data and decoded gesture signals.....	91
Figure 4-16 A detail descriptions between the actual gesture and reconstructed gesture from Figure 4-17. (Refer to Appendix G for the video)	94
Figure 4-17 The synthesized gestures in a different sequence, CADB, as the training sequence, ABCD. (Refer to Appendix G for the video)	95
Figure 4-18 Segment of the five different signals at $P(\text{connection}) = 0.5$. The first signal is the actual user gesture Y-motion in respective to the right wrist. The second signal is the PLBM reconstructed gestures given an uninterrupted encoded data. The third and fourth signals are the reconstructed gesture by the first and second approaches. The fifth signal is the gesture signal from spatial imitation. (Refer to Appendix G for the video)..	97
Figure 4-19 The degree DTW_MSE (top) and jerk (bottom) with respect to the degree of connection. Each data from first three signals is computed from a median of 75 signals, which is from 15 training sequences with five replicates. Each signal is generated from different connection sequences.	98
Figure 4-20 The three overlapped skeleton points with boundary constraints. The first and second columns are the actual and the reconstructed gesture without considering the constraints. The third column is the output with constraints considered. (Refer to Appendix G for the video)	100
Figure 4-21 An example of an evolutionary algorithm, and the contours are the scores for the given features, y_1 and y_2	102
Figure 4-22 The iteration results of the evolutionary algorithm in the PLBM at one of the time step.	103
Figure 4-23 The motion points of the wrists with constraints in different spatial views for 300 time steps. The motion contains three distinct gesture styles. Two constant constraints were deliberately set on the x-axis and z-axis; as a result, the first and second columns have few of its points outside the boundary.....	104
Figure 4-24 The median of various evaluation criterions over 15 training sequences with the boundary set at $\{x\text{-min: } -0.4, x\text{-max: } 0.4, y\text{-min: } -0.6, y\text{-max: } 0.6, z\text{-min: } -0.45, z\text{-max: } 0.15\}$	105
Figure 4-25 Motion points of the wrists with constraints in different spatial views	106
Figure 5-1 Convolved features from CNN-RBM-Mean.....	108

Figure 5-2 The output of a gesture decoder. There are nine different encoded data at each interval, which is 100 time steps; as a result, the decoded gestures have nine different styles.	109
Figure 5-3 Visualization of the features spaces, from Figure 4-15, using PCA	110
Figure 5-4 Retrieving gestures motion give a similar encoded facial data as the learning mode.....	111
Figure 5-5 The three overlapped skeleton points with boundary constraints. The first and second columns are the actual and the reconstructed gesture without considering the constraints. The third column is the output with constraints considered.	113
Figure A-1 Various interpersonal spaces categorized by Hall [53].	118
Figure A-2 The relationship between gesture intensives and perception of extraversion [11].	119
Figure B-1 A sample training dataset with objects of different shapes and colors.....	121
Figure B-2 Different test input with their corresponding output represented in local and distributed representation.....	122
Figure B-3 Sorting pick and place scenario.	123
Figure B-4 Sample of the MNIST database (left), 100 hidden units in first layer of a neural network that was pre-trained using Deep Belief Network then fine-tuned using backpropagation with dropout (middle) [50], and a simple neural network configuration (right).	123
Figure B-5 Mnist dataset represented using t-sne by [61].	124
Figure C-1 A motion sequence (bottom) and its 100 hidden units (Top) [14].	126
Figure C-2 Training images (left) and reconstructed images using RBM (Right) [62]...	126
Figure C-3 The associative architecture between identity label, FACS labels, and facial image (left). The reconstructed facial images given the identity label and FACS labels (right) [62].	127
Figure C-4 The top row is the raw image sequence. The second row is the input sequence, which is the raw images sequence corrupted with noise. The third and fourth rows are the output of TRBM-VV using single hidden layer and two hidden layers respectively [36].	128
Figure C-5 On the left is some RBM features of facial expression. On the right is the positive (1-3 rows) and negative (4-6 row) correlation of the features (row) with an action units (column).	128
Figure D-1 MAVEN [65] user interface for navigation control	130
Figure D-2 MeBot [5] user interface	131
Figure D-3 Hasegawa's [6] user interface.....	132
Figure D-4 Park's [18] user interface	133

Figure E-1 The variance between different spatial parameter for an arms motion – extending and contracting towards the chest [16].	134
Figure E-2 Reanimation for two different styles given the same script [17].	135
Figure E-3 The four input (blue) and the 15 output (black or green) for a cheering motion [19].	135
Figure E-4 Reconstructing walking motion with five different styles using CRBM [37].	136
Figure E-5 Reanimating the same motion for two different styles: neutral (input top) and proud (output bottom) [15].	137

List of Tables

Table 2-1 Comparison of the existing models with their characteristics and effects, and grey boxes are the desired characteristics and effects	27
Table 4-1 Comparison between the raw data and the various encoded features with the objective functions. The CNN and K-mean have their average distances computed from five different training dataset.	77
Table 4-2 Test result using (3.15) for CNN-mean with rectified dropout neuron and Adadelta learning rate. The rows represent the number of neurons, and the columns store two different parameters: the type of decay and the decay rate.....	78
Table 4-3 Comparison between different encoded features with gate size of 300 and L2-norm. MSE is the square error of the reconstructed signals with respect to the actual signals, while the DTW_MSE is the square error of the realigned reconstructed signals with respect to the actual signals.	84
Table 4-4 The relationship between the parameter size respects with to the DTW_MSE result.....	87
Table 4-5 The median and standard deviation of the best DTW_MSE results with respect to its learning rates and pruning threshold given the vigilance thresholds at Table 4-7. The rows of tables represent the learning rates, and the columns of the tables denote the different pruning thresholds.....	93
Table 4-6 Total number of weights after pruning with respects to the DTW_MSE result in Table 4-5	93
Table 4-7 The vigilance thresholds with the best DTW_MSE result at a specific learning rate and pruning threshold.	93
Table F-1 Characteristics and its types with descriptions and examples (bold* are the preferred types)	138
Table F-2 Effects and its types with descriptions and examples (bold* are the preferred types).....	140

List of Symbols

Encoder variables	
Symbols	Description
$\mathbf{v}$	Data
$\mathbf{v}'$	Reconstructed data
$\mathbf{P}$	Set of re-orientated skeleton point from Kinect
$\hat{\mathbf{o}}_k$	k unit vector of the upper limb
$\mathbf{i}$	Concatenation of $\hat{\mathbf{o}}_k$, where $k \in \{1, 2, \dots, 6\}$
$\hat{\mathbf{i}}$	Normalized of $\mathbf{i}$ with zero-mean and unit variance
$\mathbf{h}$	Spatial encoded data or Spatiotemporal encoded data
$\mathbf{h}_t^{(1)}$	Spatial encoded data for CNN layer 1 at time t
$\mathbf{h}_t^{(2)}$	Convolved output for CNN layer 2 at time t
$\mathbf{h}_t^{(3)}$ or $\mathbf{z}_t$	Pooled output for CNN layer 3 at time t , and spatiotemporal encoded data
$\boldsymbol{\theta}_{e_s}$	Spatial encoder parameters
$\boldsymbol{\theta}_{e_t}$	Spatiotemporal encoder parameters
$\mathbf{W}_{e_s}$	Spatial features
$\mathbf{a}_{e_s}$	Spatial visual bias
$\mathbf{b}_{e_s}$	Spatial hidden bias
$\mathbf{W}_{e_t}$	Convolved features
$\mathbf{a}_{e_t}$	Spatiotemporal visual bias
$\mathbf{b}_{e_t}$	Spatiotemporal hidden bias
$\boldsymbol{\mu}(\boldsymbol{\theta})_j$	Mean of the encoded data for a class j
$d(\boldsymbol{\theta})$	Inter-class distance
$e(\boldsymbol{\theta})$	Intra-class distance

Decoder variables

Symbols	Description
θ_{dt}	Spatiotemporal decoder parameters
C_{env}	Environment Constraints
x_{in}	LSTM: input
x_{gin}	LSTM: input gate
x_{gfor}	LSTM: forget gate
x_{gout}	LSTM: output gate
y_{in}	LSTM: input activation
y_{gin}	LSTM: input gate activation
y_{gfor}	LSTM: forget gate activation
y_{gout}	LSTM: output gate activation
s_{cell}	LSTM: Cell status
y_{cell}	LSTM: output
W_{in}	LSTM: input parameter
W_{gin}	LSTM: input gate parameter
W_{gfor}	LSTM: forget gate parameter
W_{gout}	LSTM: output gate parameter
W_{out}	Output transformation parameter
y_{out}	Transformation output
ϵ	Noise gain

Associator variables

Symbols	Description
ρ	Vigilance threshold/parameter
x_a'	Estimated data
$\tilde{x}_a$	Partial data
x_a	Data
y_a	Associator output or degree of the prototypes
τ	Pruning threshold
θ_a	Associate parameters
W_a	Associate weights/prototypes
$w_{a,j}$	Associate vector/prototype

Encoder functions

Symbols	Description
$f_{e_s}(\cdot)$	Spatial encoding function
$f_{e_s}^{-1}(\cdot)$	Spatial decoding function
$f_{e_t}(\cdot)$	Spatiotemporal encoding function
$g_{e_t}(\cdot)$	Convolution function
$Pool(\cdot)$	Pooling function

Decoder functions

Symbols	Description
$f_{d_t}(\cdot)$	Spatiotemporal decoding function
$Dtw(\cdot)$	Dynamic time warping
$\sigma_{log}(\cdot)$	Logistic Function from $[0,1]$
$\sigma_{siga}(\cdot)$	Sigmoid Function from $[-a, a]$
$E_{d_t}(\cdot)$	Optimization function to release constraint

Associator functions

Symbols	Description
$s(x, y)$	Similarity function
$l_{fast}(\cdot)$	Fast Learning
$l_{slow}(\cdot)$	Slow Learning
$f_a(\cdot)$	Find winner
$r_a(\cdot)$	Find winner with missing input

List of Acronyms

Acronym	Definition
ARAM	Adaptive resonance associative map
ART	Adaptive resonance theory
BPTT	Back propagating through time
CATR	Customizable anthropomorphic telepresence robot
CMC	Computer-mediated communication
CNN	Convolution neural network
CV	Cross validation
DOF	Degree of freedom
DP	Dynamic programming
DTW	Dynamic time warping
EA	Evolutionary algorithm
EDGAR	Expression Display & Gesturing Avatar Robot
ESN	Echo state network
FALCON	Fusion architecture for learning, cognition, and navigation
FCRBM	Factored conditional RBM
FFT	1D fast Fourier transform
FFT2	2D fast Fourier transform
FPS	Frame per second
FTF	Face-to-face
K-NN	K-nearest neighbor
LSTM	Long short-term memory
MLP	Multilayer perceptron
MSE	Mean square error
NATR	Non-anthropomorphic telepresence robot
PATR	Partial-anthropomorphic telepresence robot
PCA	Principle component analysis
PLBM	Perception-link behavior model
RBM	Restricted Boltzmann machine
RBPTT	Ratio BPTT
RNN	Recurrent neural network
SIFT	Scale-invariant feature transform
SVM	Support vector machine
TRBM	Temporal RBM
TTR	Traditional telepresence robot
VSM	Vector space model

This page is intentionally left blank.

1 Introduction

“Many believe effective networking is done face-to-face, building a rapport with someone by looking at them in the eye, leading to a solid connection and foundational trust.”

Raymond J. A.

Face-to-face (FTF) communication is still the best form of communication tools over computer-mediated communication (CMC). Many researchers have pointed out that FTF communication is more efficient and informative than CMC because of the higher degree of social cues. Therefore, the customizable anthropomorphic telepresence robot (CATR) should be a solution to increase the degree of social cues in CMC. However, a fully anthropomorphic using traditional teleoperation might deteriorate the effectiveness of the conversation. As a result, the objective of the thesis is to propose and develop a novel module for the CATR operator interface so that the CATR can perform operator's personal and expressive nonverbal cues during teleoperation.

1.1 Background

1.1.1 Benefits of face-to-face communication

Face-to-face (FTF) communication is the de facto communication medium because it has richer social cues that make the communication process more efficient and informational. These social cues range from facial expression, vocal variety, body posture, gestures, and proximity (Appendix A). They can demonstrate the degree of intimacy and immediacy towards a person or a topic. For instance, two communication partners could exploit the physical distance between each other to show the degree of intimacy. Furthermore, social cues can regulate and facilitate a conversation. For example, delivery, citing, seeking, and turn gestures can regulate and facilitate a communication process. In addition, the deliverer can build rapport with the receiver using these cues. A deliverer could mimic the receiver's posture to increase the rapport between himself and the receiver. In conclusion, FTF communication is rich in social cues, which improves social interaction. However, FTF communication does have a few limitations.

1.1.2 Benefits of computer mediated communication

Despite the advantages in FTF communication, computer-mediated communication (CMC) is still an essential medium for the future, and CMC take the form of e-mail, video and audio, audio only, instant message, digital forum, and telepresence robot. It is essential because CMC is not bounded by the constraint of space. For instance, a video conference with parties located in two different countries, Singapore and Hong Kong. It could be set up despite the large distance between the two countries. Secondly, few of the CMC devices are also not restricted by time (asynchronous). For example, a deliverer, who located in Singapore, could send an intended message through email to his receiver, who lived in Hong Kong. The deliverer could continue with other tasks, while he would expect the receiver to get his message. Besides releasing physical constraints, CMC can also be a useful tool for a person to reveals more information about himself to others. Because the conversation between two parties could be held at familiar environments, both the parties should feel less intimidated and less defensive. On the other hand, a person can also conceal his genuine state of mind through CMC.

For an instant message, a deliverer could conceal his emotions by choosing the opposite words to express his current emotion. In conclusion, CMC is an important tool, and it has several advantages over FTF communication. One of the goals of this thesis is to combine the advantages from both FTF communication and CMC so that there is a high-quality pseudo FTF communication that is not bounded by space.

1.1.3 Telepresence robot and its benefits

Figure 1-1 Traditional telepresence robot

In the existing CMC devices, a telepresence robot (Figure 1-1) should have the highest level of social presence. Unlike most CMC tools, telepresence robot has a physical embodiment, and a physical embodiment can project the physical presence of the remote user that improves the interaction. In the experiment conducted by Bainbridge et al. [1], they observed that the participant not only enjoyed interacting with the physical robot but also gave the physical robot a personal space. Furthermore, mobility also provides the operator a means to demonstrate their presence, involvement, and attention. With mobility, the operator could control the robots to turn “left and right” to show attention and turn “towards the person” who was talking to demonstrate his involvement [2]. Besides mobility, few of the telepresence robots also have anthropomorphic functions to improve social interaction. Figure 1-2 shows A) PRoP [3] and B) RP-Vita [4] that are non-anthropomorphic telepresence robot (NATR) with a few specific anthropomorphic functions. PRoP has a laser pointer to perform the deictic gesture, and RP-Vita has a pan-tilt screen to simulate gazing. Recently, there are also telepresence robots with higher anthropomorphism to model realistic human motion. C and D of Figure 1-2

are two of the partial-anthropomorphic telepresence robots (PATR) with more freedom (DOF) in the neck and manipulators modules. With the higher DOF manipulators, the operator could gesture more expressive motions than the NATR to increase the degree of social presence. In conclusion, a telepresence robot is the only CMC tool that empowers the operator with video conferencing and mobility. In order to increase the degree of social presence in the existing telepresence robot, a customizable anthropomorphic telepresence robot was proposed and developed.

Figure 1-2 A - PRoP with a pointer [3], B - RP-Vita with a pan-tilt unit [4], C - MeBot with 3 DOF manipulators [5], and D - Hasegawa's bot with 4 DOF manipulators [6].

1.1.4 Customizable anthropomorphic telepresence

The latest telepresence robot is the customizable anthropomorphic telepresence robot (CATR) that imitates operator's nonverbal cues (Figure 1-3). The idea is to increase the anthropomorphic functions so that the system can execute realistic and nature human motion. An anthropomorphic telepresence robot is a combination of the social humanoid robot and the traditional telepresence robot. An anthropomorphic telepresence robot has similar upper body structure as a social humanoid so that it can perform realistic motions, e.g. gestures, gaze, or posturing. Besides anthropomorphic functions, the anthropomorphic embodiment also has an impact on social interaction. In the study of motor interference [7], [8], both studies concluded that beings or robot with an anthropomorphic body created an interference effect on the observing participants, and the interference effect was not significant on an industrial robot. Lastly, CATR is also easily customizable to any operator with its unique head module. Unlike a social robot, telepresence robot needs to represent the operator, so the system must be able to change its facial

identity. Two examples of these systems are the Expression Display & Gesturing Avatar Robot (EDGAR-1) [9] and the Robot Thespian [10], where both systems have a back-projection head. The high fidelity anthropomorphic appearance and functions should enrich the social interaction process.

Figure 1-3 Classification of telepresence robot w.r.t humanoid robot (Left), and EDGAR-1 [9] being inhabited (Right)

1.1.5 Benefits of gesturing in CATR

Gestures have plenty of functions and effects to an ongoing conversation, and the following are three of the functions. Gestures can demonstrate the personality of the exhibitor. For example, Neff et al. [11] conducted an experiment to show the relationship between gesture intensives and the perceptions of extraversion in a virtual agent. They [11] showed that there was a significant effect of increasing gesture intensives to increase the perceptions of extraversion (Figure 1-4 left). In other functions, punctuation gestures, also known as baton gestures [12], can highlight or consolidate the important segment of speeches. Some examples include pointing the finger into the air, pounding the fist, or oscillating both hands. For instance, a speaker could perform a numbering gesture in ascending order at the start of every point. This type of gestures allows the speaker to create a better structure for the speech by highlighting the main points. The other functions also include interactive gestures that regulate and organize a spoken dialog. Bavelas et al. [13] identified four groups of the interactive gestures (Figure 1-4 right), and the interactive gestures are delivery, citing, seeking, and turn gestures. The delivery gestures facilitate the delivery of information from the speaker to the listener. The

citing gestures cite or acknowledge listener for their contribution. Next, the seeking gestures try to obtain a response from the listener. Finally, the turn gestures manage the speaking turn between both the speaker and listener. This set of gestures can be used to facilitate interaction to obtain smooth and fluent communication. In conclusion, gestures are important to a communication process. Hence, CATR operator interface should be capable of generating personal and congruent gesture to increase the degree of social presence. The main challenge is for CATR to maintain the personal nonverbal cues during a teleoperation process.

Figure 1-4 The relationship between gesture intensives and perception of extraversion by Neff et al. [11] (left). Interactive gestures by Bavelas et al. [13] (right).

1.1.6 Problems of teleoperating a CATR

Few of the problems associated with teleoperating a CATR, especially on gestures, include decoupling, unstable network condition, and limited situation awareness (Figure 1-5). Decoupling is an event where the operator chooses to commit part of his nonverbal modalities to other tasks, instead of the interaction. For instance, an operator was conversing with a remote participant while typing on the keyboard. In most cases, the CATR would either become stationary or continue to map the operator's gesture. Both the outcomes are undesirable because those gestures are not congruent to the context. The second issue is the network condition that can be affected by many factors, and some of them include signal interference and physical distance between the devices. When the success rate of the incoming signals is uncertain, some types of data representations, such as poses information, might cause the CATR motion to be jerky and uncanny. For the last issue, the operator's situation awareness of the remote environment is limited, and it might pose a danger to the remote participants. One of the solutions is to deploy the

reactive safety mechanism on the CATR. Often, this technique distorts the original signals to fit the environment constraint; as a result, the CATR motion might not be smooth, and it might not deliver the actual intents. In short, an ideal interface should handle the above issues so that the operator can effectively operate the CATR without degrading the expressiveness and overload the operator's cognitive load.

Figure 1-5 The three possible teleoperating issues while operating a CATR using natural interface

1.2 Research gaps

From the review, in Chapter 2, we found a few research gaps, and we briefly illustrate some of them. Firstly, there is a lack of studies in generating expressive gestures for an anthropomorphic robot or virtual agent [14]–[17] without tedious manual configuration. Secondly, most of the PATR [5], [6] are using reactive obstacle avoidance that might cause jerky motion. Lastly, most of the models [5], [6], [14], [15], [18], [19] do not have a proper strategy to handle decoupling. There are only two models [16], [17] that allows the recreation of gestures using other modalities, but the strategy needs the operator to manually gather the relationship between the modalities. In short, there does not exist a framework that can handle the teleoperating issues so that the operator can effectively operate the CATR without degrading the expressiveness and overload the operator’s cognitive load.

1.3 Aim of the research

The aim of the research is to study and develop a module that is capable of solving the teleoperating issues without degrading the CATR expressiveness and overloading the operator’s cognitive load. The details of the objectives are as follows:

1. At all time, the CATR should imitate the operator’s personal gestures. (expressiveness)
2. The control for the CATR should be intuitive and easy to use for any operator. (cognitive load)
3. During a decoupling situation, the CATR should react according to the operator’s reaction as though he was still operating the system. (substitution capability)
4. If the connection was not stable, the CATR should continue to gesticulate smoothly and congruently. (spatiotemporal data structure)
5. If the operator executed an unsafe gesture, the module should tune the gesture so that the CATR does not pose any danger to the remote-audience. (deliberate safety mechanism)
6. In addition to above, the re-tuned gesture should keep the original intention and should not be jerky. (deliberate safety mechanism)

In summary, the objectives can be reduced to having high in expressiveness, low in cognitive load, a deliberate safety mechanism, and a substitution capability. Expressiveness defines the amount of variance in the robot behavior. The interface should generate high expressive nonverbal cues so that any operator could execute their personal nonverbal cues. Besides having more prototype behaviors, there should also have more variance within a prototype behavior. As a result, the operator could also adjust the distributed parameters to fit his personal gestures to his current mood. For the next parameter, cognitive load defines the amount of mental information the operator needs to execute a nonverbal behavior. Even with the high expressiveness, the interface should not overload the operator's cognitive load because the operator should focus on the primary task that is the conversation. The interface should also be intuitive for an operator to use. For the third parameter, safety concerns about the well-being of remote participants when they coexist the robot. The interface should prevent the remote system from intimidating or endangering the interactants. Besides collision avoidance, the interface should, also, preserve the expressiveness of the behavior. For instance, the robot should execute a motion that best fit the desired motion to avoid the obstacle (Figure 1-6). The deliberative output is the preferred output because it is usually less abrupt and more meaningful than the reactive output. The last parameter is the ability to substitute, and it concerns the system behavior whenever there is a need to replace specific incoming nonverbal cues. The interface should promote a seamless interaction when the wireless connection was unstable or the operator chose to compromise part of his nonverbal signals to engage in other tasks. Instead of naively substituting any operator's motion, the system should learn and retrieve a motion that is coherent to the current contextual information. Figure 1-7 shows three scenarios in sequence from $t_1 < t_2 < t_3$, and each scenario has different events. At t_3 , the operator was talking to one of the participants, but his hands were occupied by the keyboard and mouse. In this scenario, the imitation or idle outcomes was not the desirable because they disrupted the communication process with an incongruent gesture. A good solution at t_3 would be substituting the learnt motion in t_1 or t_2 onto the participant "A" or "B" respectively.

Figure 1-6 An obstacle avoidance situation, where there are two possible outputs: deliberative and reactive.

1.4 Significance of the research

The completion of this study will have several significances to CMC technologies. In a bigger picture, this research is a step to push CMC technology closer to FTF communication. From the previous study of the anthropomorphic appearance [7], [8], we understand that there is a biological effect when an audience is interacting with an anthropomorphic embodiment. However, our study is adding the behavioral components, which is also an important part of our daily social conversation. Consequently, the solution is to have a strategy to portray the CATR as though it is the operator, himself. Unlike naively imitating the operator's behavior, our study, also, considers the problems an operator and audience might experience during a remote conversation. In this study, we found three of the teleoperating problems, which might affect the expressiveness of the CATR. In addition, we, also, formulated a holistic strategy to tackle these problems without compromising the expressiveness and cognitive load. In conclusion, this research and findings would be important for the future of CMC technology.

	Operator environment	Remote environment
t_1	TOP VIEW Operator	Robot Participant A Participant B Imitation
t_2		Imitation
t_3		<div> <div> Imitation </div> <div> Idle </div> </div> <div> Substitution </div> <div> OR </div>

Figure 1-7 An operator was gesturing and conversing to participant “A” at t_1 (first row). The same operator was gesturing a different gesture, and he was speaking to participant “B” at t_2 (second row). The operator was using the keyboards and mouse while talking to participant “A” or “B” at t_3 (third row).

1.5 Perception-link behavior model

Figure 1-8 Concept of perception-link behavior

The proposed model, perception-link behavior model (PLBM), balances the four parameters, and the framework with a PLBM (Figure 1-8) contains four components: natural interface, encoders, decoders, and associator. A natural interface is the preferred acquisition technique because it needs minimum operator's cognitive load. The operator just had to gesticulate as if he was with the audience, and the capturing device could acquire the pose information. The encoders are a set of specialized modules, trained to a specific signal (different nonverbal cues), and they transform these signal into a set of spatiotemporal distributed parameters. In addition, the spatiotemporal distributed space groups gesture with the similar style closer to each other. If the operator was expressing a certain gesture style for a period, then the encoder should transform the signals into a distributed output. During this period, the distributed outputs should be similar to one another. Next, the decoders are a set of trained modules to reconstruct various signals given the encoded parameters. Instead of receiving the poses or joint information, the CATR was receiving a distributed output that encapsulated the gestures style. The decoder would then decode the distributed output so that the CATR could exhibit the gesture. Lastly, the associator builds up relationships between various modalities, facial and gesture features, because the operator might have his preference of

reaction on different audiences. For instance, the operator was expressing extrovert personality towards a participant, A, and he was expressing an introvert personality towards another participant, B. The associator would learn the operator's reaction according to the context. It could exploit this knowledge during a decoupling situation (Figure 1-5).

1.5.1 Detail descriptions of the PLBM dealing with the various issues

The PLBM adopts the natural interface as the acquisition approach because the operator can control the system with lesser cognitive workload. There is no remapping of intended motions into any sort of input devices. Unlike other input device, operator just needs to express himself without added mental information. Any operator can use the system with minimum instructions. Since the interface captures the operator's motions, the operator just need to memorize a few command postures. Some examples of the command postures might be engaging and disengaging the system.

Figure 1-9 The scatter distribution of nine gestures in the 1st (y) and 2nd (x) principal components. The principal components were formulated using principle component analysis (PCA).

For expressivity, the model transforms all its input signals into distributed output to perverse their expressivity. As opposed to local representation, distributed representation (Appendix B) encompasses more information given the same amount of memory. For a memory size of N capacity, the local representation can only represent up to N number of classes; on the contrary, the distributed representation with binary values can represent up to 2^N classes. Besides, the output of the transformation is also nearer to signals with similar dynamic characteristics. Because the distributed parameters represent the mutual features within the dataset,

similar signals tend to group closer to one another. Figure 1-9 displays nine distributions, which each distribution stands for a distinct gesture motion. The gestures were being transformed into their distributed output and further processed into a visible space by a dimension reduction technique. From the histograms of the first (x) and second (y) components, the gestures under the same category tend to be grouped closer together. In addition, the distributed features are also extracted using an automated feature extraction algorithm (Appendix C), and it is more comprehensive than manually extracted features. In a manual feature extraction process, a feature engineer would design and extract the relevant features based on the application and personal experience; this might result in biases and incompleteness in the features. On the other hand, automated feature extraction algorithm extracts a set of features that is the common underlying structures within a given dataset. This approach is more complete, and the same set of features can even be deployed for different applications with different specifications.

Figure 1-10 The output of a gesture decoder. There are nine different encoded data at each interval, which is 100 time steps; as a result, the decoded gestures have nine different styles.

Next, the decoder can reconstruct gestures and act as a deliberate safety mechanism. By combining the encoder and decoder, the PLBM can reconstruct the encoder data to its original dynamic signals. Figure 1-10 shows nine encoded data (third row) and their reconstructed signals (second row). The reconstructed signals are the spatial motion of the right wrist only, and a segment of the skeleton view is

at the first row of the figure. This shows the capability of the decoder to reconstruct a sequence of cyclic gestures given the encoded data. The system can also adjust the encoded data to suit the environment constraints. Figure 1-11 shows three different upper stick bodies executing the similar motion. If the black box is the boundary constraints, both the actual (first column) and unconstrained encoded motions (second column) are considered unsafe as a portion of their motion is outside the box. On the other hand, the constrained encoded motion (third column) has all its motion enclosed within the black box.

Figure 1-11 Three overlapped skeleton points with boundary constraints. The first and second columns are the actual and the reconstructed gesture without considering the constraints. The third column is the output with constraints considered.

In an event of decoupling, there are missing or undesirable modalities. The online association module is responsible to retrieve or replace them with a set of relevant signals. The online association module, first, learns the co-occurrence between the different modalities. Figure 1-12 (left) shows the operator's gesture style can be grouped into four different dynamics depend on the facial features from the operator and audience. In this case, the associator learns their relationships, which can be classified as A, B, C, and D. Once the training phase is over, the system can then infer the unobservable/loss user's reactions given the other observable environment signals. Figure 1-12 (right) shows the reconstructed gesture when the system is given similar facial data from the operator and audience. The facial features are in a different sequence (CADB); still, the synthesized gestures should have the similar dynamics as the actual gestures style.

Figure 1-12 On the left, the operator was reacting to an audience in four distinct gestures style with four different facial expressions, ABCD. On the right is a simulated decoupling situation, the CATR's gesture was synthesized based on the facial expression in a different sequence, CADB.

Figure 1-13 Segment of the three different signals. The first signal is the actual user gesture Y-motion in respective to the right wrist. The second signal is the PLBM reconstructed gestures given an uninterrupted encoded data. The second signals are the reconstructed gesture using the previous encoded data whenever there is a missing data. The third signal is the gesture signal from spatial imitation.

For intermittent connection issues, the system should insert a valid gesture motion whenever the gesture signal is missing. If the data structure uses spatial information, e.g. poses or joints information, then the output signals might be discontinuous and jerky at the remote-end. For example, the third row of Figure 1-13 shows the output using the imitation approach. The system always updates the output with the current input, and it will remain unchanged whenever there is a missing data. As a result, there might have an instance where there is a huge change in acceleration causing the system to be jerky. On the other hand, if the data structure captures the spatiotemporal information, then the system could insert the last received data into the current time step. For instance, the second row of Figure 1-13 shows the output using the spatiotemporal information (encoded gesture data), where missing data were replaced with previously known data. The data structure stores the spatiotemporal information, gesture style; as a result, the decoder can reconstruct a smooth and continuous gesture even when there is a discontinuity in the network.

1.5.2 Summary of the PLBM

In conclusion, the proposed model, the PLBM, is capable of solving the teleoperation issues without degrading the CATR expressiveness and overloading the operator's cognitive load. It is equipped with a set of encoders, a set of decoders, and an associator. With these modules working together, the PLBM is able to encode and associate multimodal signals. During decoupling situation, it can retrieve missing modalities. With the spatiotemporal distributed structure, earlier encoded data can be inserted to the current time step during a disconnection. Furthermore, the spatiotemporal distributed structure also allows the re-tuning of the encoded data so that the new decoded gesture could fit into a constrained environment.

1.6 Outline of remaining chapters

The remaining chapters of this thesis are organized as the following. Chapter 2 illustrates on the previous research studies in social interaction, computer-mediated communication, and gestures interface in the field of telepresence robot and virtual agent. Chapter 3 covers a description of the proposed framework, PLBM, for a customizable anthropomorphic telepresence robot, and the selected model for each of the modules within the PLBM. Chapter 4 explains the experimental setup and results for each module and the framework. Lastly, Chapter 5 reemphasizes on the achieved work so far and proposes a list of future works, which will lead to the final objectives.

2 Literature review

“Increasing anthropomorphism may raise expectations [in the remote participants] and should be done only when the interface and system can meet higher expectations.”

Nowak K. L. and Biocca F. [20]

Upgrading the existing telepresence robot to a customizable anthropomorphic telepresence robot (CATR) seems like a probable solution to replace the face-to-face (FTF) communication. CATR is equipped with a human-like and human-size upper body structure and a set of anthropomorphic functions to imitate higher degree of nonverbal cues. These enhancements should increase the degree of social presence in CATR. However, the system will not be completed if the operator interface could not represent the nonverbal cues to meet the increasing expectations. Besides solving the teleoperation problems, the module should also assist the CATR to create expressive gestures, and it should not overload the operator's cognitive load.

2.1 Social presence

2.1.1 Difference between telepresence and social presence

A telepresence robot is an interactive social robot that represents the operator and interacts with the audience in the remote environment. The first task of the telepresence robot interface brings the contexts of the remote environment to the operator, and the second task is to project the presence of the operator to audience in the remote environment. The difference between these two tasks can be classified using the terms of telepresence and social presence.

Figure 2-1 The individual components in the presence balloon

The term telepresence is defined as the “sense of being there” [21] from the viewpoint of the operator. Telepresence was first described as the study of realistic physical sensory experience on the operator. Minsky [22] first introduced this term: telepresence. He described that the concept of telepresence should focus on “high-quality sensory feedback” so that the remote operator could work as though he was there. With the introduction of virtual reality, the concept of telepresence has extended into the psychological state of the operators. Pettey et al. [23] defined telepresence as the use of technology to immerse a user’s psychological state into a mediated environment. In short, the usage of telepresence only tends to be fixated on the operator. However, a telepresence robot application involves not only the operator but also the audience in the remote environment.

Social presence is defined as the “sense of being together with others” [21]. Social presence is the degree of social aspect projected by a mediated medium. Short et al. [24] were the first to introduce theory of social presence. They believed that different mediated medium exhibited different level of immediacy and intimacy. For instance, video conferencing has a higher degree of social presence comparing to instant-messaging [25]. Subsequently, this concept was brought into the studies of virtual agents. In one of these studies, Nowak et al. [20] applied the theory of social presence to examine the degree of the anthropomorphism on the observer towards in a virtual agent. In summary, social presence encompasses the view of an observer on the mediated medium as a social agent.

2.1.2 Social presence: Appearance and behaviors

Unlike other systems like MeBot [5], CATR has a human-like and life-size appearance, and these physical characteristics might have double-edged effects (Figure 2-2). Realistic physical characteristics should match with realistic behaviors to be effective. In a telepresence application, the realistic behaviors should match the operator’s behaviors because social presence does not only depend on the appearance of the system but also the behavior of the system.

Figure 2-2 MeBot [5] (partial anthropomorphic telepresence robot) and EGDAR [9] (customizable anthropomorphic telepresence robot)

In Figure 2-3, Social presence has three main components, and they are co-presence, psychological involvement, and behavioral engagement [21]. The sense of co-presence is the experience where the observer feels that he is not alone in an environment or vice versa. An instance of the survey question is “I hardly noticed

another individual”, or a vice versa is “I think the other individual often felt alone” [21]. Next, psychological involvement is the observer’s experience of empathy, attention, and understanding towards the other or vice versa. In the Network Minds social presence questionnaires, one of the questions is “My thoughts were clear to my partner”, while the reverse is “The other individual’s thought were clear to me” [21]. The last components, behavior engagement, is when the observer believes that his actions is connected to or responded from the action of the other or vice versa. An example of such question is “What the other did [affect] what I did”, and the corresponding question is “What I did [affect] what the other did” [21]. In summary, the above components further build the understanding of social presence, and there is a correlation of social presence to nonverbal cues.

Figure 2-3 Layout of Networked Minds social presence by Biocca et al. [21].

From the components created by Biocca et al. [21], each component can be associated with either physical anthropomorphic appearance or nonverbal behaviors. Firstly, realistic physical appearance with expressive gestures can improve the sense of co-presence of the observer. For instance, Garau et al. [26] found out that a high realism agent with inferred gazing had a higher co-presence than high realism agents with random gazing. On the other hand, they also observed that the opposite trends for a low realism agent. In their conclusion, they emphasized “the importance of matching the visual and behavioral realism to increase avatar effectiveness” [26]. Secondly, nonverbal cues can increase the degree of psychological involvement in a social interaction. Argyle et al. [27]

conducted several experiments to evaluate the effect of varying nonverbal and verbal cues with different attitudes. One of experiments was a stand-alone experiment, and the subject would experience attitudes, e.g. hostile/friendly, unpleasant/pleasant, or dislike/like, from either nonverbal cues or verbal cues only. In the other experiment, it was an interaction experiment between nonverbal cues and verbal cues with varying attitude. Because of the inconsistency between the nonverbal cues and verbal cues, the scales for the experiment were changed to stable/unstable, straightforward/confusing, and sincere/insincere. From these experiments, Argyle et al. [27] concluded that subjects “attach[ed] more weight to non-verbal cues in the communication of interpersonal attitudes”. In other fields on social interaction, there are also studies of behavior engagement between observer and speaker to build rapport. For example, the chameleon effect describes the effect of subconsciously mirroring during a positive social interaction [28]. The types of mirroring are extended in form of body posture or speech pattern. In conclusion, both physical embodiment and nonverbal behaviors play an important role in improving degree of social presence. Especially in CATR, the problem is to improve the realism of the nonverbal behavior to match its customizable anthropomorphic structures.

In conclusion, a CATR should portray realistic nonverbal behaviors in a conversation. However, the teleoperation issues represented in Section 1.1.6 might degrade the expressiveness to a certain level. During intermittent connection, the CATR might have a jerky and discontinuous motion, and the audience’s opinion towards the CATR as the user might be affected. Furthermore, the solution should be carefully considered so that the solution would not degrade the expressiveness of the system. For instance, some interface uses spatiotemporal local data structure that groups gesture into finite and mutually exclusive set of gestures. This method will truncate the personal gestures into a generic gesture, and this truncation might alternate the intention of the conversation. As a result, the module should ensure that the CATR creates and maintains the expressive gestures, in addition to solving the teleoperation issues.

2.2 Existing control interfaces

2.2.1 Evaluation of the existing interface

Table 2-1 displays the different interfaces with their characteristics and effects; in addition, the greyed boxes indicate the desirable characteristics and effects for a CATR interface. The rows are the various interfaces for controlling CATR. The first five rows are the interfaces from telepresence robot applications (Appendix D). They are interfaces from the traditional telepresence robot (TTR), MeBot [5], Hasegawa’s bot [6], and Park’s bot [18]. The next three rows have the five generative models (Appendix E) for a virtual agent to generate expressive gestures. They are the model from Hartmann [16], Neff [17], Lau [19], Taylor [14], and Xia [15]. The characteristic covers the module or architecture of the existing reviewed models (Appendix F), and the effects are the capabilities when using that specific configuration (Appendix F).

2.2.2 Summary of the research gaps

In summary, we observed that all the models have their pros and cons, but none of them is capable of satisfying our aims, which we set in Section 1.3. The following are some research gaps that we had found by comparing the existing models.

Firstly, there is a lack of studies in generating expressive gestures for an anthropomorphic telepresence robot with a deliberative safety mechanism. For instance, the MeBot [5] uses reactive obstacle avoidance, and their approach is to abruptly stop/reroute the motion before it hit the obstacle. This abruptly stopping or rerouting might cause the motion to be jerky. We make the above observation because MeBot uses a passive-model controller, where the MeBot only receives the joint information, spatial information. As a result, it is impossible for the robot to perform deliberate gesture motion.

Secondly, there is a lack of studies in generating expressive gestures for an virtual agent [14]–[17] without tedious manual configuration. For example, the Hartmann’s model [16] generates expressive gesture by tuning the six different distributed parameters. Three of these parameters are spatial extent, temporal

extent, and fluidity. In this model, the operator must select the desired prototypical gestures and tune these six parameters to his desired preference. The whole process would be too tedious for the operator especially when he was concurrently conversing with his audience.

Lastly, most of the models [5], [6], [14], [15], [18], [19] do not have a proper strategy to handle decoupling. For example, the strategy adopted by MeBot [5] was to let go of the manipulator controller while the operator attended to his remote tasks. During this period, the remote robot will be motionless while the conversation is still going on. This is not a desirable outcome because the modalities, e.g. gesture and vocal cue, are not congruent to the operator's behavior. Fortunately, there are two models [16], [17] that allow the recreation of the missing modalities using other observable modalities, but this strategy needs the operator to manually gather the relationship between the modalities.

In short, there does not exist a framework that can handle the teleoperating issues so that the operator can effectively operate the CATR without degrading the expressiveness and overload the operator's cognitive load.

Table 2-1 Comparison of the existing models with their characteristics and effects, and grey boxes are the desired characteristics and effects

		Characteristics					Effects				
		Input interface	Representation	Encoding	Decoding	Associating	Expressivity	Cognitive load	Decoupling	Obstacle avoidance	Avoid Speech collision
Telepresence robot	TTR (>100 predefined motions)	Traditional input device	Spatiotemporal local (>100)	Manual	Auto	-	High	High	Yes (idle)	Deliberative or Reactive	No
	TPR (<10 predefined motions)	Traditional input device	Spatiotemporal local (<10)	Manual	Auto	-	Low	Low (require remapping)	Yes (idle)	Deliberative or Reactive	No
	MeBot [5]	Master-slave	Spatial distributed (Auto)	Direct	Direct	-	High	Low (require remapping)	Yes (idle)	Reactive	No
	Hasegawa [6]	Natural interface	Spatial distributed (Auto)	Direct	Direct	-	High	Low	No	Reactive	Yes
	Park [18]	Natural interface	Spatiotemporal local (13 predefined motions)	Auto	Auto	-	Low	Low	Yes (idle)	Deliberative or Reactive	Yes
Virtual agent	Hartmann [16] and Neff [17]	Traditional input device	Spatiotemporal distributed (Manual)	Manual	Auto	Manual	High	High	Yes (associate)	Deliberative or Reactive	No
	Lau [19]	Natural interface	Spatiotemporal distributed (Auto)	Auto	Auto	-	High	Low	Yes (idle)	Deliberative or Reactive	Yes (assume)
	Taylor [14] (FCRBM) and Xia [15]	Natural interface and Traditional input device	Spatiotemporal distributed (Manual)	Manual	Auto	-	High	High	Yes (idle)	Deliberative or Reactive	Yes (assume)

2.2.3 Desirable configurations in term of the characteristic

The interface should allow the operator to express expressive gestures. A local spatiotemporal represented interface with many generic gestures (>100) might generate expressive gestures. In a TTR interface, each key or combination of keys (>100) could be a unique gesture. This abundant amount of gestures increases the degree of expressiveness in this interface. Next, a direct transmission interface with spatial representation can also reconstruct expressive gestures. For instance, the Hasegawa's interface adopts the direct transmission with spatial representation approach. Using a KINECT sensor, an array of joint information could be acquired, which directly maps onto a set of servos on the remote system. During operation, the operator's pose would be directly transmitted and projected onto the remote robot. In an ideal condition, the robot would be as expressive as the operator since robot was imitating the operator. Finally, an interface with spatiotemporal distributed representation can synthesize expressive gestures too. For example, if a system has N prototypical gestures and M binary distributed parameters, then the total number of gestures will be $N \cdot 2^M$. Figure 2-4 displays a variation of the spatial parameter on a prototypical gesture.

Figure 2-4 The variance between different spatial parameter for an arms motion – extending and contracting towards the chest by Hartmann [16].

In addition, the interface should not overwhelm the operator's cognitive load. An interface using the traditional input device (Figure 2-5) can only have limited predefined gestures (<10) to lower operator's cognitive load. For a TTR interface, if a limited number of keys (<10) is one-to-one mapped with a unique gesture, then an

operator just need to memorize a few keys. This limited amount of gestures should not be overwhelming to the operator. An interface with master-slave devices (Figure 2-5) also uses lesser operator's cognitive load. For MeBot, the operator just needed to remap his intended motion onto a primary device, and the motion would be replicated onto the remote robot. This process does not need the operator to memorize any keys. An operator interface with the natural interface (Figure 2-5) consumes lesser cognitive load than other input interfaces. Unlike other interfaces, the natural interface acquires the natural motion of the operator without memorization or remapping process. In the Hasegawa's interface, the operator just needed to express his gesture, and the remote robot would move accordingly to the operator's motion.

Figure 2-5 Various input interfaces

Next, the remote system should also have a safety mechanism, so the safety of the remote subjects and objects is ensured. If the safety mechanism tunes the original motion, then the tuned motion should preserve the intention of the original motion. As a result, the deliberative safety mechanism is preferred because it can alter the path of the motion, and the motion keeps its original intention. Deliberative safety mechanism can only be achieved by interface with spatiotemporal representation. For instance, Lau's model could output many variances of an input signal, and all the output signals kept the same shape and dynamic characteristics as the input (Figure 2-6). Furthermore, a deliberate output is also less abrupt than

reactive output. For instance, MeBot had range sensors to detect any obstacle in its surrounding, and the system would abruptly crop off any motions that were too close to the obstacles.

Figure 2-6 The four input (blue) and the 15 output (black or green) for a cheering motion by Lau [19].

Lastly, the interface should also handle decoupling situation by substituting a coherence behavior for the missing modalities. For most of the existing works, the interface handles decoupling by allowing the remote-system to go into an idling state. For example, an operator, who was operating the MeBot, could stop his actions and continue with other tasks simply by letting go of the primary system. Fortunately, there are interfaces with associating function that can create rules to relate different modalities; moreover, the interface can infer behaviors to replace or fill in any missing modalities given other observable modalities. For the model in Hartmann's and Neff's work, the gesture behaviors were manually tagged with the speech structures. When a speech was executed, the corresponding gestures would also be executed. In the context of telepresence robot, the operator could talk with the remote-partner, while he was typing on his keyboard. The remote robot would continue to express his gestures as though he was gesturing them.

2.2.4 Optimal configurations for CATR

The desirable effects can be summarized into four parameters, and they are high expressiveness, low in cognitive load, deliberate safety mechanism, and the ability to substitute. For deliberate safety mechanism, the interface should ensure the safety

of any objects and persons in the remote environment; in addition, it should not distort the meaning of the gestures. From most of the review works, the interface with spatiotemporal representation and automatic decoding function are the only pairs of characteristics that suit these objectives. For instance, a remote system could re-tune the spatiotemporal output for the gesture to suit the environment; thereafter, the motion could be generated without posing any threat and distorting the meaning of the motion. For high expressiveness and smooth re-tuning process, the interface signals should adopt distributed representation (Appendix B), and the features should be extracted by an automated feature extractor (Appendix C). Unlike local representation, distributed representation has higher expressiveness and can generate new signals. In addition, with an automated feature extractor, the distributed features will be more consistent and comprehensive. An interface with the above conditions and a low cognitive load must then adopt natural interface with automatic encoding functions. For this case, the operator just expressed his gestures to the natural interface, and the captured information should be automatically encoded into a set of spatiotemporal distributed signals. These spatiotemporal distributed signals would then be sent to the remote system for re-tune if necessary. Finally, the interface should have automatic associating functions to create rules between different modalities. This function will allow the remote system to substitute coherence and congruent modality when that modality is undesirable or missing. For example, an operator was talking to a specific audience, and he was gesturing with a specific style. The gesture with the same style should be displayed on the remote system if the operator's upper limb was occupied on the keyboard when he was talking to the same audience.

3 Methods: Perception-link behavior model

“Social interaction is impossible without a subtle and unspoken – in other words, nonverbal – negotiation of the respective roles to be played by the two parties in an interaction.”

Knapp M. L. and Hall J. A. [29]

In a teleoperation, a customizable anthropomorphic telepresence robot (CATR) must behave expressively to be an effective medium. Besides expressiveness, a CATR interface must also have other desirable effects, low cognitive load, deliberate safety mechanism, and ability to substitute, to counter the issues posed during teleoperation. As a result, the architecture of the module should consist of the automatic encoding, decoding, and associating capability to achieve the desirable effects. In addition, the data structure for the modalities should be spatiotemporal distributed representation in order to preserve expressiveness with spatiotemporal information. Thus, the perception-link behavior model (PLBM) was designed and implemented with the desirable functions and representation; furthermore, the PLBM considers the signals from both the operator and audiences because social interaction is between two or more parties.

3.1 Perception-link behavior model

3.1.1 Definition of the model

The name of this model is the perception-link behavior model (PLBM) because the behaviors between the conversation partners are often depended on what they see during a conversation. Role negotiation [30] is an example where two partners' behaviors depend on their perceived roles between them. The examples of the roles include friend-friend, mother-child, teacher-student, or interrogator-suspect. The two partners must subtly negotiate their roles. The conversation will not be successful unless “the two people tacitly agree on a common understanding” [29]. For perception related behaviors, chameleon effect is also a popular phenomenon where people consciously or unconsciously imitate the communication partner. In the experiment conducted by Chartrand and Bargh [28], they observed that subjects experienced smoother and more positive interaction when the communication partner mirrored their movements. In summary, the PLBM aims to associate both the communication partners' behaviors, so the PLBM can better model the relationship between two communication partners.

3.1.2 Components of the model

Figure 3-1 Block diagram of perception-link behavior model, and the unreliable signals are denoted as v , while the reliable signals are denoted as u . The blue and red coding are referring to operator's and audience's signal respectively. The green coding is the decoded signals.

The PLBM is a real-time associative multimodal modal with encoding and decoding capability (Figure 3-1), and it has three main components: encoders, decoders, and associator. The encoder transforms the signal into a distributed output, and the signals are grouped nearer together if they have the similar characteristic. Figure 3-2 (first row) shows a motion generator with two degrees of freedom: linear l_t and rotation θ_t . During the encoding process (second row of Figure 3-2), the motion generator created four different signals, A, B, C, and D, by varying l_t and $\frac{\partial \theta_t}{\partial t}$. The encoder would then learn the distributed features α and β . Subsequently, the signals could be represented in the new distributed space as shown in the second row of Figure 3-2, and signals with similar dynamics should be grouped nearer together. In addition, the distributed space is also capable of representing unseen signals, e.g. $(A + B) / 2$ in Figure 3-2. The next component is the decoder, and the decoder regenerates distributed output back to its original signals without losing any expressiveness. Once the signals were represented into their distributed output, the decoder could take in the distributed output and transformed it back to its original signals (third row of Figure 3-2). Besides regenerating on trained data and signals, the decoder could also generate unseen data and signal. For the last components, associator learns relationships between multiple modalities; eventually, it can also retrieve any missing modalities by searching the best candidate to match the observable modalities. In the fourth row of Figure 3-2 (left) shows that the relationship of the generated motion v_1 and the two other signals u_1 and u_2 . If these relationships were consistent, then the associator could use this knowledge and create a list of rules on these relationships. In a different situation (fourth row in Figure 3-2), if there was a missing signal v_1 , the associator could exploit that learnt knowledge and retrieve a synthesized signal v'_1 given the observable signals u_1 and u_2 .

Figure 3-2 The possible output of each component in a PLBM.

3.1.3 Achieving the four parameters

The PLBM has an encoding technique that preserves expressiveness, and it has a decoding technique that generates expressive motion given the encoded output. The encoder adopts an unsupervised technique so that the extracted features are more complete and comprehensive. For instance, the second row of Figure 3-2 shows the motion generator of a signal $v_1(t) = \alpha(t) \sin(\beta(t) + \varphi)$, where α is the amplitude, β is the angular frequency, and φ is the phase. In an unsupervised approach, e.g. Fourier transformation, the distributed features will be α and β . Instead, manually extracted features might not contain both the features. The extracted features represent the underlying structures of the data, and they are mutually inclusive. Oppositely, a local representation represents signals into mutually exclusive categories, e.g. A, B, C, and D. This representation truncates an expressive signal into a generic category. The PLBM, on the contrary, uses the distributed representation. The distributed features, α and β , are the amplitude and the angular frequency of any signal. Instead of using a library of synthesized gestures, the decoder is a contextual generative model, and the reconstructed signals have a same expressiveness as the original signals. The third row of Figure 3-2 shows the desired outcomes when the decoder regenerated the signal for the encoded data, A, B, C, and D. In addition, the decoder should also generate any encoded data in the distributed space without losing expressiveness.

The PLBM also adopts techniques and interface that require a lesser cognitive load. Firstly, the natural interface is the acquisition technique for the gesture in the PLBM. With a minimum amount of memorization or remapping, the natural interface only requires the operator to perform his gestures as if it is a face-to-face conversation. Furthermore, most of the modules are automatic. For instance, the operator just needed to gesticulate. The encoder would automatically encode the gesture signals into its corresponding distributed output, and the interface would transmit the distributed output of the gesture and other modalities to the remote system. Subsequently, the associator would automatically update its parameters. Once the decoder receives the encoded data, it would automatically synthesize the

current poses of the signals. Except for the gesticulation process, the rest of the processes do not require any human involvement.

The PLBM can also be implemented with a deliberative safety mechanism. The decoder can simulate the upcoming motion so that the system can deliberately respond to the constraints. For instance, the left of the third row of Figure 3-2 shows a list of synthesized signals progress ahead of time by several time steps from the initial condition. Hence, the simulated signal can be tested with the current environment constraint before the robot executes the motion, and this testing will continue for every time step. If the simulated signal does not satisfy the environmental conditions, then the next process will be executed. Since the encoder groups similar signals in the distributed space, it is also feasible to find a similar encoded data to replace the incoming encoded data so that the reconstructed signal can deliberately avoid any constraints. The right of the third row of Figure 3-2 shows a reconstructed signal with a combination distributed output from cluster A and cluster B. This example demonstrates that if a list of distributed outputs is extracted along cluster A to cluster B, then they will share the same amplitude but the angular frequency will increase until it is the same as B signal. From the above analogy, if an incoming distributed data cannot satisfy the environmental conditions, then the system can search the neighbor region of the incoming distributed data to find the best candidate.

The PLBM also has the ability to substitute coherence modalities to replace any missing modalities. The associator empowers the PLBM to learn the relationship between different modalities. In the fourth row of Figure 3-2 (left) shows that the relationship of the generated motion v_1 and the two other signals u_1 and u_2 . If these relationships were consistent, then the associator could use this knowledge and create a list of rules on these relationships. Besides learning, the associator can also retrieve any missing modalities given the observable modalities. In a different situation (fourth row in Figure 3-2), if there was a missing signal v_1 , the associator could exploit that learnt knowledge and retrieve a synthesized signal v'_1 given the observable signals u_1 and u_2 .

3.1.4 CATR with PLBM to handle the teleoperating issues

By achieving the four parameters, the PLBM can also be incorporated into a CATR to handle the three teleoperation issues (Figure 3-3). In the first scenario, the PLBM can conceal and replace unwanted gesture of the operator during a conversation. The PLBM can automatically associate multimodal signals and generate congruent modality on the concealed modality. For the second scenario, the PLBM can insert missing data when there is a bad network connection, and data are losing at a rate of $P(connect) < X$. In this case, the PLBM can reconstruct a smooth and expressive gesture. Any lost data can be replaced by the last known data since the received data encapsulates a gestures style rather than the pose information. For the last scenario, the PLBM can re-tune the incoming data so that CATR can deliberately and smoothly avoid a collision. Since the encoded data are grouped closer to their dynamics, a searching algorithm in PLBM can search in the surrounding region to find a good candidate, which gives zero-collision and minimizes distortion in respect to the original signal.

Figure 3-3 Three different case scenarios for the PLBM

3.1.5 Other advantages for PLBM in CATR

Besides having the ideal configuration of the four parameters in the review chapter, the PLBM also has other properties that make it superior over the other existing interfaces for a CATR operation.

The PLBM is the only interface with a real-time associator. The existing interface requires an offline acquisition of parameter before it can operate. For instance, Neff's model [17] requires a statistical model of the speaker's gestures style and the relations of the gestures style with the speech structures. All of this information must be gathered before the system can be launched. On the other hand, the PLBM has a real-time associator. The interface can create knowledge of the operator's behaviors and audience's behaviors in runtime. Therefore, any operator could operate the system, and his personal behaviors towards a specific audience would be recorded. All of this can be done without any prior knowledge of the operator or the audience.

Besides, the PLBM also takes into the consideration of the audience since a conversation is between two communication parties. Since most of the existing interface does not have an associator, they do not consider modeling the audience's behavior or identity. Bottom of Figure 3-4 shows the data flow in a MeBot operation. The interface did not store any information from either the operator's behaviors or audience's behaviors. As a result, if the operator decided to decouple from the master system, then the slave system would enter into an idling state. On the contrary, the PLBM can form rules between the operator's behaviors and audience's behaviors so that the audience can experience personal conversation if the associator is activated. Top of Figure 3-4 shows the placement of the PLBM in a CATR operation. During the operation, the modalities of the operator and audience would be acquired and associate by the associator. These modalities include gestures, facial expression, identity, and variety. If the operator decided to decouple from the motion sensor, the associator could retrieve the best gestures to fit into the contexts based on the observable modalities. Therefore, the audience would still experience the operator's presence.

The PLBM is also expandable to take in more modalities. Most existing interfaces do not consider modeling other modalities, so none of the interface has considered the above issues. Most of the interface is designed to perform a one-to-one mapping from a sensor signals to an output devices. From the example in bottom of Figure 3-4, the MeBot only transmits neck, pseudo-gestures, vocal content, and raw facial signals. In the remote environment, the incoming signals are directly mapped onto neck servos, slave system, microphone, and display screen respectively. On the other hand, the PLBM adopts the approach of completing the contextual information. In top of Figure 3-4, the PLBM does not only model the gazes, gestures, vocal cues, and facial expression, it also models the user's identity and other audience's modalities to have a better understanding of the current context. In the future, if there is more information, e.g. scene understanding, then the PLBM can also accommodate the addition modalities.

Figure 3-4 Perception-link behavior model integrated with telepresence robot system

3.2 Model for encoder

3.2.1 Background

The basic concepts of encoder were derived from different existing and robust models. They are the vector space model, restricted Boltzmann machine, and convolution neural network. The next few sections show the fundamental concept and the advantages of these existing models.

3.2.1.1 Vector space model

Vector space model (VSM) is a model for remapping/transforming any object into a vector. These objects in their vector forms are nearer to their counterparts with the similar trait. This concept is very useful, and it is the backbone of the concept in the encoder. For instance, an encoder can convert any multivariate time signals into the new vector space, and the new vector space keeps those signals with similar dynamic characteristics together. VSM is a set of techniques to represent any data into a vector. The data can be any objects, e.g. text document or images. The elements of the vector can be a set of unique words gathered from a set of documents or a set of generalized cropped images from a group of heterogeneous images. Once the data are represented in their vector form, a system can perform various operations, such as querying, indexing, or retrieval. The next section demonstrates a VSM example in the computer vision application.

Figure 3-5 Procedures to get the visual words

Figure 3-5 shows an example on one of the computer vision application, known as bag of words model. The first step of this model is to extract local features using a feature extractor, e.g. Harris detector. The next step is to adopt and apply a feature descriptor, e.g. Scale-invariant feature transform (SIFT), to the local images. Finally, the last step is to perform a clustering algorithm, e.g. K-mean, to create a bag of visual words. At this point, the system has K-words in its cookbook. If it is a classification system, the system will require a few more offline procedures.

Figure 3-6 illustrates a by-product of a simple classification system to categorize humans, dogs, and boats. The first step extracts the local patches from the raw images. The subsequent step transforms them into the new vector space based on the feature descriptor. At this point, every image has a different number of local patches, and each local patch is represented by vectors. The next step is to count the occurrence of K-words in the cookbook, given an image and its corresponding vectors. Every image will have a corresponding histogram with K-dimension, and the last step is to train a supervised model, for example a support vector machine (SVM) or K-nearest neighbor (K-NN), to categorize the necessary classes, which is humans, dogs, and boats.

Figure 3-6 Three objects with their respective histogram

In summary, VSM represents an object with similar trait together, and it is an important idea for the encoder. It is the fundamental concept for encoder because it allows further development for the decoder and associator.

3.2.1.2 Restricted Boltzmann machine

Restricted Boltzmann machine (RBM) is a generative model, which models the joint probability distribution between the observations and their label. Generative modeling is an important learning strategy for the encoder, for it is a difficult task to manual extract out a list of valid and correct distributed labels. A generative modeling also reduces human error in the label selection process.

A RBM [31], [32] is a special version of Boltzmann machine (Figure 3-7), where there is no lateral connection. The energy function of a binary RBM is

$$\begin{aligned} -E(\mathbf{v}, \mathbf{h}; \boldsymbol{\theta}) &= \sum_i v_i a_i + \sum_j h_j b_j + \sum_{i,j} v_i h_j W_{ij} \\ &= \mathbf{v}^T \mathbf{a} + \mathbf{h}^T \mathbf{b} + \mathbf{v}^T \mathbf{W} \mathbf{h}, \end{aligned} \quad (3.1)$$

where $\mathbf{v} \in [0,1]^n$ is the binary input, $\mathbf{h} \in [0,1]^m$ is the binary output, and $\boldsymbol{\theta} = \{\mathbf{W}, \mathbf{a}, \mathbf{b}\}$ are the parameters for the RBM model.

Figure 3-7 Node connections for A) Boltzmann machine and B) Restricted Boltzmann machine

Using such configuration, the conditional probability of the hidden node can be calculated given the visible node. This is possible because the hidden nodes are independent of each other. This also applies while reconstructing the visible node given the hidden nodes. The sigmoid function is the typical inferring and the reconstructing function for binary RBM. They can be expressed as

$$p(h_j = 1 | \mathbf{v}; \boldsymbol{\theta}) = \frac{1}{1 + e^{-b_j - \sum_i W_{i,j} v_i}} = \sigma \left(b_j + \sum_i W_{i,j} v_i \right), \quad (3.2)$$

$$p(v_i = 1|\mathbf{h}; \boldsymbol{\theta}) = \frac{1}{1 + e^{-a_i - \sum_j W_{i,j}h_j}} = \sigma\left(a_i + \sum_j W_{i,j}h_j\right). \quad (3.3)$$

The next advantage of RBM is the fast learning algorithm known as Contrastive Divergence [31] that estimates the maximum likelihood of the network. The concept of the Contrastive Divergence assumes that the energy at the training state should always be lower than the region surrounding the training state.

There are two phases the Contrastive Divergence algorithm. They are the positive and negative phase. During the positive phase, the hidden unit $\mathbf{h}$ is inferred via (3.2) given $\mathbf{v}$. The positive gradient is formulated by the outer product of $\mathbf{v}$ and $\mathbf{h}$. Subsequently, the surrounding region is estimated through Gibbs sampling. In Gibbs sampling, the first step is to take $\mathbf{h}$ and reconstruct $\mathbf{v}'$ via (3.3). Next, the second stage is to use $\mathbf{v}'$ and infer $\mathbf{h}'$ via (3.2). This negative phase will carry on until the predefined sampling parameter is met. Finally, the negative gradient is computed by the outer product of the last $\mathbf{v}'$ and $\mathbf{h}'$. The weight update function is

$$\Delta W_{ij} = \epsilon \left(\langle v_i h_j \rangle_{data} - \langle v'_i h'_j \rangle_{gibb} \right), \quad (3.4)$$

where ϵ is the learning rate, $\langle \dots \rangle_{data}$ is the expected positive gradient, and $\langle \dots \rangle_{gibb}$ is the expected negative gradient.

In short, the RBM is a generative model, and it learns the underlying features in the data. It remaps the data into a set of distributed output label based on the learnt features. This reduces the amount of manual work and human error for labeling the labels. The produced labels are naturally distributed because the captured features are the mutual features among the data. These properties are important for encoder because the encoded data need to be a set of vivid labels. If those vivid labels are manually predefined, those vivid labels might be biased due to the labeler's experience and culture.

3.2.1.3 Convolution neural network

Convolution neural network (CNN) [33] is an architecture that follows the concept of VSM, and it is possible to implement RBM into this architecture. In

addition, CNN has a few advantages that improve the remapping performance as compared to a typical bag of words model.

A typical CNN contains a few functions depending on its design, and one of the instances of CNN contains convolution layers, pooling/sub-sampling layers, and a multilayer perceptron. The convolution layer can have numerous features depend on the applications; in Figure 3-8, the first convolution layer has k_1 features/weights. These k_1 features will slide along input image and produce k_1 feature maps. The next layer is the sub-sampling/pooling layer. The sub-sampling/pooling operation groups the neighbor data together. This operation not only reduces the number of elements in the layer but also makes the data invariant to translation. The convolution and pooling layer can continue for a few times. However, a fully connected multilayer perceptron (MLP) with softmax function is always at the end, if the CNN adopts the supervised learning model, back-propagation.

Figure 3-8 Convolution neural network with discriminative model

Similar to VSM, CNN represents any object into a vector. It has a sequence of convolution and pooling layers to transform and subsample an object into a vector of the new space. The final layer is the multilayer perceptron model, which classifies the input images to their respective class. Unlike the classical bag of words model where its local spatial information is lost, the local spatial information for CNN is only truncated.

Besides training using the discriminative model, the CNN can also adopt a generative model. For instance, a RBM can be used to train the network instead of back-propagation, which requires labeling.

3.2.2 Architecture of encoder for gestures (CNN-RBM)

Inspired by the VSM, RBM, and CNN, the motivation was to purpose and build an encoder model, and it should have the capability to remap a non-stationary signal to a generalized, distributed, and phase-free vector space.

Figure 3-9 Four layers encoder model

The proposed encoder model (Figure 3-9) has four layers. The first layer is the input layer. The data for the input layer is a list of upper limb vectors $\hat{\mathbf{i}}$. Each data contains six unit vectors. Each limb is rescaled to a unit so that it is invariance to the length of the limb. The second layer is a spatial transformation layer. This layer transforms input data $\hat{\mathbf{i}}$ into a spatial distributed data

$$\mathbf{h}^{(1)} = f_{e_s}(\hat{\mathbf{i}}; \boldsymbol{\theta}_{e_s}) = \mathbf{W}_{e_s} \cdot \hat{\mathbf{i}} + \mathbf{b}_{e_s}. \quad (3.5)$$

where $\boldsymbol{\theta}_{e_s} = \{\mathbf{W}_{e_s} \in \mathbb{R}^{N^{(1)} \times 18}, \mathbf{b}_{e_s} \in \mathbb{R}^{N^{(1)}}\}$ are the model parameters, and $N^{(1)}$ is the number of feature maps in the second layer. The third layer is a convolution layer. The model first extracts a window of T size from the spatial

data $\mathbf{h}_{(t-T+1):t}^{(1)} \forall t$. Next, a smaller window of c size slides across the T window at every k , where k ranges from $0, \dots, T - c$. For every k , the model executes the following operation:

$$\begin{aligned} \mathbf{h}_{t,k}^{(2)} &= g_{e_t} \left(\mathbf{h}_{(t-k-c+1):(t-k)}^{(1)}; \boldsymbol{\theta}_{e_t} \right) \\ &= \sigma_{rec} \left(\mathbf{W}_{e_t} \cdot \mathbf{h}_{(t-k-c+1):(t-k)}^{(1)} + \mathbf{b}_{e_t} \right), \end{aligned} \quad (3.6)$$

where $\mathbf{h}_{t,k}^{(2)} \in \mathbb{R}^{N^{(2)}}$ is the convoluted output, the parameters are $\boldsymbol{\theta}_{e_t} = \{\mathbf{W}_{e_t} \in \mathbb{R}^{N^{(2)} \times cN^{(1)}}, \mathbf{b}_{e_t} \in \mathbb{R}^{N^{(2)}}\}$, and $\sigma_{rec}(z) = \max(0, z)$ is a rectified function. Because it is a generative model, the reconstructed spatial data

$$\mathbf{h}'_{(t-k-c+1):(t-k)}^{(1)} = g_{e_t}^{-1} \left(\mathbf{h}_{t,k}^{(2)}; \boldsymbol{\theta}_{e_t} \right) = \mathbf{W}_{e_t}^T \cdot \mathbf{h}_{t,k}^{(2)} + \mathbf{a}_{e_t}, \quad (3.7)$$

where $\boldsymbol{\theta}_{e_t} = \{\mathbf{W}_{e_t} \in \mathbb{R}^{N^{(2)} \times cN^{(1)}}, \mathbf{a}_{e_t} \in \mathbb{R}^{N^{(1)}}\}$ are the parameters for the inverse function. The last layer is the pooled layer. This layer pools the sequence of convoluted data $\mathbf{h}_{t,0:(T-c)}^{(2)}$ elementwise to get

$$\mathbf{h}_t^{(3)} = \text{pool} \left(\mathbf{h}_{t,0:(T-c)}^{(2)} \right), \quad (3.8)$$

where the $\text{pool}(\cdot)$ can either be mean or max operation. Finally, the encoded data $\mathbf{h}_t^{(3)}$ is a noise-free, distributed, and phase-free vector. To simplify the processes from (3.5), (3.6), (3.7), and (3.8), the spatiotemporal encoder function can be rewritten as

$$\mathbf{z}_t = \mathbf{h}_t^{(3)} = f_{e_t} \left(\hat{\mathbf{t}}_{(t-T+1):t}; \boldsymbol{\theta}_{e_s}, \boldsymbol{\theta}_{e_t} \right), \quad (3.9)$$

3.2.3 Data and its preprocessors

Gathering the data for the encoder is not an easy task. Firstly, the selected dataset should cover large variety gestures; in addition, the gestures within the same class should have some variance. Secondly, the data would come from different individuals with different physical characteristics, which might affect the accuracy of the result. Lastly, a high parameters network might find itself over-fitted, and it

also have a long training time; hence, the dataset should have addition pre-processes to improve the efficiency of the training.

The encoder needs a large amount of dataset from a wide diversity of gestures because it is a data-driven model. As a result, the dataset was made out of three sources. The first source is the Microsoft action gesture [34], and it contains three sets of 12 gaming gestures from ten individuals. Only nine out of the 12 gestures were selected because they have more expressive upper limb motion. The second source is the 3D iconic gesture dataset [35]. These gestures are speech-dependent, and it is extremely useful for the telepresence application. The dataset covers 29 subjects gesturing 20 different virtual objects. These objects range from primitive objects, such as sphere and cube, to complex objects, such as bottle and vase. The third dataset is an in-house dataset, and it encompasses deictic gestures and a few speech-dependent gestures. Lastly, the combined dataset undergoes a mirroring operation to double up the number of training data to 268296 for five frames per second (FPS).

Figure 3-10 Selected features for the experiment

Next, the raw data (Figure 3-10 left) not only contains motion of the upper limb but also entangles with other unwanted information. The unwanted information can be variances in subject's height or subtle body movement. These variances might alternate the similarity index among similar gestures. To prevent such issues, the raw skeleton data went through two operations. The first operation normalized pose data so that the normalized poses $\mathbf{P} = \{\mathbf{p}_1, \dots, \mathbf{p}_{20}\} \in \mathbb{R}^{3 \times 20}$ are independent of the torso's movement. The next operation extracted the eight poses $n =$

(5,6,7,9,10,11) that govern the upper-limbs. It computed the six relatively unit vectors $\hat{\mathbf{o}}_j \in \mathbb{R}^3$ by

$$\hat{\mathbf{o}}_j = \frac{\mathbf{o}_j}{\|\mathbf{o}_j\|} = \frac{\mathbf{p}_{n_{j+1}} - \mathbf{p}_{n_j}}{\|\mathbf{p}_{n_{j+1}} - \mathbf{p}_{n_j}\|}. \quad (3.10)$$

The final vector is a set of upper-limb orientations $\mathbf{i} = [\hat{\mathbf{o}}_1^T, \dots, \hat{\mathbf{o}}_6^T]^T$ as shown in (Figure 3-10 right).

Lastly, the dataset underwent two more procedures, and they are feature rescaling and model validation. In this report, the majority of the normalization or feature rescaling is standardization. Standardization is used in this situation because the features have different statistical parameters or units. Standardization rescales all the features in the data to have zero-mean and unit variance. Standardized feature $\hat{\mathbf{i}}$ can be computed by

$$\hat{\mathbf{i}} = \frac{\mathbf{i} - \bar{\mathbf{i}}}{\sigma}, \quad (3.11)$$

where $\bar{\mathbf{i}}$ is the mean of the upper-limb vector, and σ is the standard deviation. In many studies for machine learning, normalization has improved the experiment results and training speed. The cross-validation (CV) is a model validation technique for checking how well the trained model performs on an unseen dataset. The procedure starts by separating the dataset into three portions. They are known as training, cross-validating, and testing dataset. During training, the trained parameters are tested with the CV dataset at a specific interval. The training phase will stop when the CV results start to deteriorate. Lastly, the trained model is examined using the test dataset to get the final score. To reduce inconsistency, multiple subsets of the training and CV dataset will be created while keeping a consistent test dataset. The test results will be averaged to obtain a consistent result. The above process prevents overfitting by testing the trained parameters with the unseen data during training.

3.2.4 Evaluation criteria

The clustering evaluation provides the information of the distributions within the new vector space. There are two components in this evaluation. They are the inter-distance and intra-distance as shown in Figure 3-11. Inter-distance $d(\boldsymbol{\theta})$ is the average of the distance between the mean of all the classes $\{\boldsymbol{\mu}(\boldsymbol{\theta})_1, \dots, \boldsymbol{\mu}(\boldsymbol{\theta})_g, \dots, \boldsymbol{\mu}(\boldsymbol{\theta})_G\}$ to one another, and it is

$$d(\boldsymbol{\theta}) = \frac{\sum_{x \neq y}^{x, y \in \{1, \dots, G\}} \|\boldsymbol{\mu}(\boldsymbol{\theta})_x - \boldsymbol{\mu}(\boldsymbol{\theta})_y\|}{\beta}, \quad (3.12)$$

where G is the number of known clusters, and the denominator is $\beta = G^2 - G$, which is the normalization term. The inter-distance should be as large as possible so that individual classes are far away from one another. Intra-distance $e(\boldsymbol{\theta})$ is the average distance of all classes $\boldsymbol{\mu}(\boldsymbol{\theta})_g$ to their data points $\{\mathbf{z}_1^{(g)}, \dots, \mathbf{z}_n^{(g)}, \dots, \mathbf{z}_{N_g}^{(g)}\}$, and it is

$$e(\boldsymbol{\theta}) = \frac{\sum_g^G \left(\frac{\sum_{n=1}^{N_g} \|\boldsymbol{\mu}(\boldsymbol{\theta})_g - \mathbf{z}_n^{(g)}\|}{N_g} \right)}{G}, \quad (3.13)$$

where N_g is the number of data in class g . The intra-distance $e(\boldsymbol{\theta})$ measures the spread within a class g . It should be as small as possible so that each class is as tight as possible.

Figure 3-11 Visualization of inter distance and intra distance between three clusters of data.

Since the concept is maximizing inter-distance while minimizing intra-distance, the objective function can be rewritten as

$$\boldsymbol{\theta} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \left(\frac{e(\boldsymbol{\theta})}{d(\boldsymbol{\theta})} \right). \quad (3.14)$$

For evaluating a CNN model, the objective function for optimizing $\boldsymbol{\theta}$ is as follow:

$$\begin{aligned} \boldsymbol{\theta} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \left(\frac{e(\boldsymbol{\theta})}{d(\boldsymbol{\theta})} + \operatorname{MSE} \left(g_{e_t}^{-1} \left(\mathbf{h}_t^{(2)}; \boldsymbol{\theta} \right), \mathbf{h}_{(t-c+1):(t)}^{(1)} \right) \right. \\ \left. + \zeta \left(\mathbf{h}_t^{(3)} \right) \right), \end{aligned} \quad (3.15)$$

where $\operatorname{MSE}(\mathbf{x}, \mathbf{y})$ is the mean square error between $\mathbf{x}$ and $\mathbf{y}$, $\zeta(\cdot)$ measures the percentage non-zero elements in $\mathbf{h}_t^{(3)}$, and t represents a set of data index from the CV dataset. The mean square error $\operatorname{MSE}(\mathbf{x}, \mathbf{y})$ is

$$\operatorname{MSE}(\mathbf{x}, \mathbf{y}) = \frac{1}{MN} \sum_{m=1}^M \sum_{n=1}^N (x_{n,m} - y_{n,m})^2, \quad (3.16)$$

where $\mathbf{x}$ is the reconstructed data, $\mathbf{y}$ is the target data, N is the total number of features, and M is the total number of training data. The nonzero counting function is

$$\zeta(\mathbf{y}) = \frac{1}{MN} \sum_{m=1}^M \sum_{n=1}^N [y_{n,m} > 0], \quad (3.17)$$

where $[...] is the Iverson bracket, and it returns 1 if the condition is true.$

3.3 Model for decoder

Given the encoded data, the aim of this section is to find a suitable generative model for gestures. Many models, such as temporal RBM (TRBM) [36], factored conditional RBM (FCRBM) [37], long short-term memory (LSTM) [38], [39], and Echo state network (ESN) [40], have proven to generate signals, such as motion, voice, and writing. However, only the FCRBM and ESN generate different contextual signals. Hence, a preliminary experiment was conducted to select a suitable model. In that experiment, the models were evaluated based on the error generated between the reconstructed signal and the actual signal. The preliminary experiments showed that LSTM (Figure 3-12) performed more consistent and better in terms of the evaluation score.

Figure 3-12 Forward pass of a LSTM cell with input, output, and forget gates

Long short-term memory (LSTM) [38], [39] is an extension of the recurrent neural networks (RNN). Unlike an RNN, it has the gating mechanism to prevent vanishing/exploding gradient issue while performing back propagating through time (BPTT). In addition, the introduction of forget gates [39] gives the LSTM an extra mechanism to reset its internal state to prevent an indefinite increase of its state when the input is a continuous signal. In recent works, many researchers have applied LSTM and its extension to create many generative applications, such as generating handwriting, human dynamic, and voice phonetic.

3.3.1 Modified LSTM (Ratio BPTT)

Training a LSTM is similar in training a MLP. The first step is to determine and update the input values $\mathbf{x}_{i,t} \in \mathbb{R}^{N^{(i)} \times N^{trg}}$, where $i \in \{in, g_{in}, g_{for}, g_{out}\}$. Each

activation channel has two parameters: $\mathbf{W}_i \in \mathbb{R}^{N^{gate} \times N^{(i)}}$ and $\mathbf{b}_i \in \mathbb{R}^{N^{gate}}$, where N^{gate} is the size of the cell. The feedforward pass function computes the activation for the input, gates, internal states, cell output, and output. The activation for the input $\mathbf{y}_{in,t}$ and gates $\mathbf{y}_{i,t}$ are

$$\mathbf{y}_{in,t} = \sigma_{sig_{\alpha_{in}}}(\mathbf{W}_{in} \cdot \mathbf{x}_{in,t} + \mathbf{b}_{in}), \quad (3.18)$$

$$\begin{aligned} \mathbf{y}_{i,t} &= \sigma_{log}(\mathbf{W}_i \cdot \mathbf{x}_{i,t} + \mathbf{b}_i), \\ i &\in \{g_{in}, g_{for}, g_{out}\}, \end{aligned} \quad (3.19)$$

where $\sigma_{sig_{\alpha}}(z) = \frac{2\alpha}{1+e^{-z}} - \alpha$ is a sigmoid function ranges from $[-\alpha, \alpha]$, and $\sigma_{log}(z) = \frac{1}{1+e^{-z}}$ is a logistic function ranges from $[0, 1]$. The internal state $\mathbf{s}_{cell,t}$ is

$$\mathbf{s}_{cell,t} = \mathbf{y}_{in,t} \circ \mathbf{y}_{g_{in},t} + \mathbf{y}_{g_{for},t} \circ \mathbf{s}_{cell,t-1}. \quad (3.20)$$

The cell output $\mathbf{y}_{cell,t}$ is

$$\mathbf{y}_{cell,t} = \mathbf{y}_{g_{out},t} \circ \sigma_{sig_{\alpha_{cell}}}(\mathbf{s}_{cell,t}). \quad (3.21)$$

The last step transforms the cell output $\mathbf{y}_{cell,t}$ into the actual output $\mathbf{y}_{out,t}$:

$$\mathbf{y}_{out,t} = \sigma_{sig_{\alpha_{out}}}(\mathbf{W}_{out} \cdot \mathbf{y}_{cell,t} + \mathbf{b}_{out}). \quad (3.22)$$

For training the parameters, there were two variations of backward pass through time (BPTT), and they are the truncated BPTT [38], [39] and BPTT with clipping [41]. In the following, a new version was proposed, known as BPTT with a ratio (RBPTT). The different between RBPTT with others is the addition ratio parameters γ_t and γ_{t+1} . They are located at the point of merger between the current error derivatives and the future error derivatives.

The RBPTT can easily transform to either the truncate BPTT or BPTT. For an equivalent truncated BPTT in RBPTT, the parameters are $\gamma_t = 1$ and $\gamma_{t+1} = 0$. For an equivalent BPTT in RBPTT, the parameters are $\gamma_t = 1$ and $\gamma_{t+1} = 1$. In our experiment, the parameters are $\gamma_t = (1 - \gamma)$ and $\gamma_{t+1} = \gamma$, where $0 \leq \gamma \leq 1$, to prevent the explosion gradient problem.

Figure 3-13 Backward pass of a LSTM cell with input, output, and forget gates

For the backward pass algorithm (Figure 3-13), it starts with the SE objective function and its derivatives function

$$\mathbf{E}_t = \frac{1}{2} \|\mathbf{y}_t - \mathbf{y}_{out,t}\|^2, \quad (3.23)$$

$$\frac{\delta E_t}{\delta \mathbf{y}_{out,t}} = \mathbf{e}_t = \mathbf{y}_{out,t} - \mathbf{y}_t, \quad (3.24)$$

where $\mathbf{y}_t$ is the ground truth. The error derivative for the output $\delta_{out,t}$ is

$$\begin{aligned}\boldsymbol{\delta}_{out,t} &= \frac{\delta \mathbf{E}_t}{\delta \mathbf{z}_{out,t}} = \left[\frac{\delta \mathbf{E}_t}{\delta \mathbf{y}_{out,t}} \frac{\delta \mathbf{E}_{t+1}}{\delta \mathbf{y}_{out,t}} \right] \frac{\delta \mathbf{y}_{out,t}}{\delta \mathbf{z}_{out,t}} \\ &= \left[(1 - \gamma) \mathbf{e}_t + \gamma \left(\sum_i \mathbf{w}_i^{(out)T} \cdot \boldsymbol{\delta}_{i,t+1} \right) \right] \\ &\quad \circ \sigma_{sig\alpha_{out}}^{-1}(\mathbf{z}_{out,t}),\end{aligned}\tag{3.25}$$

where $\gamma = 0$ when $t \geq T$, $(\sum_i \mathbf{W}_i^{(out)T} \cdot \boldsymbol{\delta}_{i,t+1})$ are the future error derivatives, and $i \in \{in, g_{in}, g_{for}, g_{out}\}$. The error derivatives for cell output $\boldsymbol{\delta}_{cell,t}$ is

$$\begin{aligned}\boldsymbol{\delta}_{cell,t} &= \frac{\delta \mathbf{E}_t}{\delta \mathbf{y}_{cell,t}} = \frac{\delta \mathbf{E}_t}{\delta \mathbf{z}_{out}} \frac{\delta \mathbf{z}_{out,t}}{\delta \mathbf{y}_{cell,t}} + \frac{\delta \mathbf{E}_{t+1}}{\delta \mathbf{y}_{cell,t}} \\ &= (1 - \gamma) \mathbf{W}_{out}^T \cdot \boldsymbol{\delta}_{out,t} + \gamma \left(\sum_i \mathbf{W}_i^{(cell)T} \cdot \boldsymbol{\delta}_{i,t+1} \right), \quad (3.26) \\ i &\in \{in, g_{in}, g_{for}, g_{out}\}.\end{aligned}$$

The error derivatives for the gate output $\delta_{g_{out},t}$ is

$$\begin{aligned}\delta_{g_{out},t} &= \frac{\delta E_t}{\delta \mathbf{z}_{g_{out},t}} = \frac{\delta E_t}{\delta \mathbf{y}_{cell,t}} \frac{\delta \mathbf{y}_{cell,t}}{\delta \mathbf{y}_{g_{out},t}} \frac{\delta \mathbf{y}_{g_{out},t}}{\delta \mathbf{z}_{g_{out},t}} \\ &= \delta_{cell,t} \circ \sigma_{sig_{\alpha_{cell}}}(\mathbf{s}_{cell,t}) \circ \sigma_{log}^{-1}(\mathbf{z}_{g_{out},t}).\end{aligned}\quad (3.27)$$

The error derivatives for the internal state output $\delta_{s,t}$ is

$$\begin{aligned}\delta_{s,t} &= \frac{\delta E_t}{\delta \mathbf{s}_{cell,t}} = \frac{\delta E_t}{\delta \mathbf{y}_{cell,t}} \frac{\delta \mathbf{y}_{cell,t}}{\delta \mathbf{s}_{cell,t}} + \frac{\delta E_{t+1}}{\delta \mathbf{s}_{cell,t+1}} \frac{\delta \mathbf{s}_{cell,t+1}}{\delta \mathbf{s}_{cell,t}} + \frac{\delta E_{t+1}}{\delta \mathbf{s}_{cell,t}} \\ &= (1 - \gamma) \delta_{cell,t} \circ \left(\sigma_{sig_{\alpha_{cell}}}^{-1}(\mathbf{s}_{cell,t}) \circ \mathbf{y}_{g_{out},t} \right) \\ &\quad + \gamma \left[\delta_{s,t+1} \circ \mathbf{y}_{g_{for},t+1} + \sum_i \mathbf{w}_i^{(s)T} \cdot \delta_{i,t+1} \right], \\ &\quad i \in \{g_{in}, g_{for}, g_{out}\}.\end{aligned}\quad (3.28)$$

To get to the error derivatives $\delta_{i,t}$ for $i \in \{in, g_{in}, g_{for}\}$, (3.20) need to be differentiated with respect to their $\mathbf{z}_{i,t}$. The results are

$$\begin{aligned}\phi_{in,t} &= \frac{\delta \mathbf{s}_{cell,t}}{\delta \mathbf{z}_{in,t}} = \mathbf{y}_{g_{for},t} \frac{\delta \mathbf{s}_{cell,t-1}}{\delta \mathbf{z}_{in,t-1}} + \frac{\delta \mathbf{y}_{in,t}}{\delta \mathbf{z}_{in,t}} \mathbf{y}_{g_{in},t} \\ &= \mathbf{y}_{g_{for},t} \circ \phi_{in,t-1} + \sigma_{sig_{\alpha_{in}}}^{-1}(\mathbf{z}_{in,t}) \circ \mathbf{y}_{g_{in},t},\end{aligned}\quad (3.29)$$

$$\begin{aligned}\phi_{g_{in},t} &= \frac{\delta \mathbf{s}_{cell,t}}{\delta \mathbf{z}_{g_{in},t}} = \mathbf{y}_{g_{for},t} \frac{\delta \mathbf{s}_{cell,t-1}}{\delta \mathbf{z}_{g_{in},t-1}} + \mathbf{y}_{in,t} \frac{\delta \mathbf{y}_{g_{in},t}}{\delta \mathbf{z}_{g_{in},t}} \\ &= \mathbf{y}_{g_{for},t} \circ \phi_{g_{in},t-1} + \mathbf{y}_{in,t} \circ \sigma_{log}^{-1}(\mathbf{z}_{g_{in},t}),\end{aligned}\quad (3.30)$$

$$\begin{aligned}\phi_{g_{for},t} &= \frac{\delta \mathbf{s}_{cell,t}}{\delta \mathbf{z}_{g_{for},t}} = \mathbf{y}_{g_{for},t} \frac{\delta \mathbf{s}_{cell,t-1}}{\delta \mathbf{z}_{g_{for},t-1}} + \frac{\delta \mathbf{y}_{g_{for},t}}{\delta \mathbf{z}_{g_{for},t}} \mathbf{s}_{cell,t-1} \\ &= \mathbf{y}_{g_{for},t} \circ \phi_{g_{for},t-1} + \sigma_{log}^{-1}(\mathbf{z}_{g_{for},t}) \circ \mathbf{s}_{cell,t-1}.\end{aligned}\quad (3.31)$$

where $\phi_{i,t-1} = \mathbf{s}_{cell,t-1} = 0$ when $t = 1$. Error derivatives $\delta_{i,t}$ for $i \in \{in, g_{in}, g_{for}\}$ are

$$\delta_{i,t} = \frac{\delta E_t}{\delta \mathbf{z}_{i,t}} = \frac{\delta E_t}{\delta \mathbf{s}_{i,t}} \frac{\delta \mathbf{s}_{cell,t}}{\delta \mathbf{z}_{i,t}} = \delta_{s,t} \circ \phi_{i,t} \quad (3.32)$$

The backpropagation weight changes $\Delta \mathbf{W}_i$ for $i \in \{in, g_{in}, g_{for}, g_{for}, g_{out}\}$ are

$$\Delta \mathbf{W}_i = \frac{\delta E_t}{\delta \mathbf{W}_i} = \delta_{i,t} \cdot \mathbf{x}_{i,t} \quad (3.33)$$

3.3.2 Architecture of decoder for gestures

The proposed decoder architecture has four layers (Figure 3-14). The first layer is the input layers, and it is

$$\mathbf{x}_{i,t} = (\hat{\mathbf{i}}_{(t-p):(t-1)}, \mathbf{z}_t, \tilde{\mathbf{y}}_{i,t-1}), \quad (3.34)$$

where $\hat{\mathbf{i}}_{(t-p):(t-1)}$ is the previous upper-limb states, $\mathbf{z}_t$ is the encoded gesture signal, and $\tilde{\mathbf{y}}_{i,t-1} = (\mathbf{y}_{cell,t-1}, \dots)$ is the previous LSTM states. The second layer is the LSTM layers. The third layer is the transformation layer, and it transforms the LSTM output vector $\mathbf{y}_{cell,t}$ into the upper limbs vector $\mathbf{y}_{out,t}$. Lastly, the final layer normalizes the upper limbs vector $\mathbf{y}_{out,t}$ to the constraint upper-limb vector $\hat{\mathbf{i}}_t$. The final output is then placed at the input of the next time step. To simplify the feed-forward functions from (3.18) to (3.22) and normalization process, the decoder function can be rewritten as

$$\hat{\mathbf{i}}'_t = f_{d_t}(\hat{\mathbf{i}}_{(t-p):(t-1)}, \mathbf{z}_t; \boldsymbol{\theta}_{d_t}), \quad (3.35)$$

where $f_{d_t}(\cdot)$ is decoder function, p is the number of previous states, $\mathbf{z}_t$ is the encoded data, and $\boldsymbol{\theta}_{d_t}$ is the optimized parameters for the model.

Figure 3-14 Decoder model when rollout with time

3.3.3 Data and its preprocessors

The upper limbs dataset went through the same process as the dataset in the encoder. Firstly, the dataset is a combination of the three datasets. They underwent the same extraction and normalization processes. These processes include extracting the relative unit vector for the upper-limb and normalizing the body orientation.

The gesture encoder transformed the gestures signal to form the encoded information. The encoded information has 200 features, which contains 25 frames (5 FPS) of information. They also went through the normalization process, zero-mean unit-variance, which has proven to give better results than the non-normalize encoded data.

The final step aligns the encoded information with the corresponding upper limb data. Since the encoded data encompass 25 frames of information, encoded data were aligned with 25 frames of upper limb data. The previous data was set to five; the decoder will train to generate the remaining 20 frames.

3.3.4 Evaluation criteria

Dynamic time warping (DTW), a form of dynamic programming algorithm, is a non-linear mapping technique for calculating the minimum distance between two strings or signals. Given a target signal $(\mathbf{t}_1, \dots, \mathbf{t}_n)$ and an input signal $(\mathbf{s}_1, \dots, \mathbf{s}_m)$ with $\mathbf{t}_i, \mathbf{s}_j \in \mathbb{R}^f$, the concept is to find a good monotonic sequence of the input signal that minimizes the distance between the target and input signals.

Figure 3-15 The Q table and the optimal solution generated by the dynamic programming.

Dynamic programming (DP) is the procedure to find the optimum solution and the best value. Given the similarity matrix C , DP solves the problem by breaking it into sub-problems. Orderly from the start to the end of the target signals, the algorithm compares each time slices with the local search space of the input signals $\{j \geq \max(i - w, 1) \text{ and } j \leq \min(i + w, m)\}$. As a result, DP fills up a table

$$Q_{i,j} = \begin{cases} \infty & j < \max(i - w, 1) \text{ and } j > \min(i + w, m) \\ C_{i,j} & i = 1 \text{ and } j = 1 \\ \min(Q_{i-1,j-1}, Q_{i-1,j}, Q_{i,j-1}) + C_{i,j} & \text{otherwise} \end{cases}, \quad (3.36)$$

where C is the similarity matrix, in term of cosine distance, and it is computed by

$$C_{i,j} = \frac{\mathbf{t}_i' \cdot \mathbf{s}_j}{\|\mathbf{t}_i\| * \|\mathbf{s}_j\|}. \quad (3.37)$$

Once the program fills up the table, the optimum value is in $Q_{n,m}$, and the optimum solution requires back tracing of minimum decision from the end to the start of the element (Figure 3-15).

Figure 3-16 The DTW algorithm realigns input signal to a target signal, and the result is the realigned input.

During the evaluation process, the aim is finding the best parameters to minimize the error between the aligned decoded signals $\mathbf{v}'_{dtw}$ and the actual signals $\mathbf{v}_{(t+1):(t+T)}$. The first step generates the decoded signals $\mathbf{v}'_{(t+1):(t+T)}$, and the decoded signal at each time step is

$$\mathbf{v}'_{t+i} = f_{d_t}(\mathbf{v}_{(t-p+i):(t+i-1)}, \mathbf{z}_t; \boldsymbol{\theta}_{d_t}), \quad (3.38)$$

$$i \in \{1, 2, \dots, T\}$$

where $f_{d_t}(\cdot)$ is the LSTM forward pass function eq. (3.35), p is the number of previous state, $\mathbf{z}_t$ is the encoded data, and $\boldsymbol{\theta}_{d_t}$ is the optimized parameters for the model. The subsequent procedure realigns decoded signal $\mathbf{v}'_{(t+1):(t+T)}$ to form the aligned decoded signals $\mathbf{v}'_{dtw}$ (Figure 3-16). Subsequently, the optimum parameter is

$$\boldsymbol{\theta}_{d_t} = \operatorname{argmin}_{\boldsymbol{\theta}_{d_t}} \operatorname{MSE}(\mathbf{v}'_{dtw}, \mathbf{v}_{(t+1):(t+T)}). \quad (3.39)$$

3.4 Model for associator

3.4.1 Adaptive resonance theory and its extension

The adaptive resonance theory (ART) [42] is a stable online clustering algorithm. Like k-mean algorithm, both algorithms estimate the location of the clusters in an unknown distributed dataset. Unlike k-mean algorithm that handles a batch of offline data, ART does them online. Besides online learning, ART can increase its knowledge base, the number of clusters, while maintaining a stable network. These are possible through its unique architecture.

There are many versions of ART [43], but all of them have three basic functions. The basic functions are the choice, match, and learning function. The first step in ART is to prepare the feature fields so that they satisfy a few requirements. In case of an ART 2A-E, the requirements involve normalizing or transforming the input field $\mathbf{v}$ such that

$$\mathbf{x}_a = f_{norm}(\mathbf{v}) \in [0, 1], \quad (3.40)$$

where $\mathbf{x}_a \in \mathbb{R}^M$ is the feature field for ART. During learning, the next function is the choice function. The choice function compares the feature field $\mathbf{x}_a$ with all the existing knowledge $\mathbf{W}_a = (\mathbf{w}_{a,1}, \dots, \mathbf{w}_{a,N})$ to compute the choice activation $\mathbf{y}_a \in \mathbb{R}^N$ that is approximated by

$$y_{a,j} = 1 - \sqrt{\frac{1}{M} \sum_{i=1}^M (x_{a,i} - w_{a,ji})^2}. \quad (3.41)$$

Next, the matching function finds the most similar knowledge $\mathbf{y}_{a,J}$ relative to the feature field $\mathbf{x}_a$ by

$$J = \underset{j}{argmax} (y_{a,j}), \quad (3.42)$$

and it compares $\mathbf{y}_J$ with the vigilance threshold ρ . If $\mathbf{y}_J > \rho$, then the learning model is the slow recoding model. The weight $\mathbf{w}_{a,J}$ is updated by

$$\mathbf{w}_{a,J} := (1 - \eta) \cdot \mathbf{w}_{a,J} + \eta \cdot \mathbf{x}_a, \quad (3.43)$$

where η is the learning rate, and it is kept between $0 < \eta < 1$. On the other hand, if $\forall j \mathbf{y}_j \leq \rho$, then an uncommitted knowledge $\mathbf{w}_{a,N+1}$ is recruited (fast learning) such that

$$\mathbf{w}_{a,N+1} = \mathbf{x}_a. \quad (3.44)$$

With these three functions, ART is a real-time stable fast and slow recoding unsupervised model. To generalize ART, the multi-channel adaptive resonance associative map was introduced.

A multi-channel adaptive resonance associative map (multi-channel ARAM) [44] is an extension of ART, and multi-channel ARAM allows multi-ARTs to work together to form a collective and associative memory. Figure 3-17 shows a multi-channel ARAM with k channels, and each channel is an ART. The k channel can work together to update the category field, or they can serve as observable modalities to estimate the missing modalities.

An instance of multi-channel ARAM is the Fusion architecture for learning, cognition, and navigation (FALCON) [45], FALCON is a three channels ARAM for a reinforcement learning model. The three channels are state, action, and rewards. In the exploration stages, three channels work together to fill the category field with rules by randomly generating a valid action. These rules not only encompass the relationship between a state and an action but also the reward, which is the feedback for taking that selected action during that state. In theory, FALCON would eventually store all the optimized rules, given enough time and training. During the exploitation stages, the system exploits the category field to find the best action given the current state. It can find the best action by setting the reward to the maximum. With the two observable modalities, the system could search through its category fields to find its best action given the state and rewards.

In conclusion, the concept is to leverage the online learning and retrieving strategy of a multi-channel ARAM in a mediated interaction process. The multi-channel ARAM can act as an associative model to record the relationships across

different nonverbal modalities. As result, the retrieval process can recover or correct any missing or corrupted modalities, when necessary. In addition, the multi-channel ARAM can capture all these rules in runtime. Therefore, any new operator can operate the system even when the system does not have any prior knowledge of that operator.

Figure 3-17 ART (left) and ARAM (right)

Figure 3-18 Three main processes of the associator: a) slow recoding, b) fast learning, and c) recalling

3.4.2 Architecture of decoder for gestures (multi-channel ARAM)

The associator model is a modification of the multi-channel ARAM. The channels in this model are the nonverbal cues captures from both the operator and the audience. The idea is to associate the different channels into a set of rules for future querying. In the experiment and result section, the features are user face features, audience face features, and user gestures feature.

The training strategy is a modified ART 2A-E. Instead of bounding the feature field $\mathbf{x}_a \in [0, 1]$ in (3.40), the encoded data are implicitly bounded. As a result, the encoded data is the feature fields

$$\mathbf{x}_a = (\mathbf{x}_a^{(f_{user})}, \mathbf{x}_a^{(f_{aud})}, \mathbf{x}_a^{(g_{user})}) = (\mathbf{z}_t^{(f_{user})}, \mathbf{z}_t^{(f_{aud})}, \mathbf{z}_t^{(g_{user})}), \quad (3.45)$$

where $\mathbf{z}_t^{(f_{user})}$, $\mathbf{z}_t^{(f_{aud})}$, and $\mathbf{z}_t^{(g_{user})}$ are the encoded data for the user's face features, audience's face features, and user's gestures features respectively. Since the feature field $\mathbf{x}_a$ is not explicitly bounded, the distance between two nodes are not bounded. Hence, new choice activation is

$$y_{a,j}^k = e^{-\sqrt{\frac{1}{M^k} \sum_{i=1}^{M^k} (x_{a,i}^k - w_{a,ji}^k)^2}}, \quad (3.46)$$

where $y_{a,j}^k$ ranges from $[0,1]$, and M^k is the number of elements in the $k \in \{f_{user}, f_{aud}, g_{user}\}$ feature fields. Subsequently, the next step is to find the best choice activation index J by

$$J = \underset{j}{argmax} \left(\prod_{k \in \{f_{user}, f_{aud}, g_{user}\}} y_{a,j}^k \right). \quad (3.47)$$

In order to determine rather to execute fast learning or slow recoding, it compares the best choice activation $y_{a,J}^k$ with each channel's vigilance threshold ρ^k . If $\forall j \exists k$ such that $y_{a,j}^k \leq \rho^k$, then fast learning function (Figure 3-18b) inserts a new weight $\mathbf{w}_{a,N+1} = \mathbf{x}_a$. On the other hand, if $\exists j$ such that $\forall k$ satisfies $y_{a,j}^k > \rho^k$, then the slow recoding function (Figure 3-18a) updates the winner node $\mathbf{w}_{a,J}$ by

$$\mathbf{w}_{a,J}^{(new)} := (1 - \eta(\varphi_J)) \cdot \mathbf{w}_{a,J}^{(old)} + \eta(\varphi_J) \cdot \mathbf{x}_a, \quad (3.48)$$

where $\eta(\varphi_J) = (\varphi_J + 1)^{-1}$ is the adaptive learning rate. Adaptive learning rate $\eta(\varphi_J)$ decreases as the weight occurrence φ_J increases. This decrease in learning rate ensures the winner weight keep within the convergence region; on the contrary, constant learning rate might cause the winner weight to move away from the convergence region when the data moves into the transition region.

The recall strategy (Figure 3-18c) is executed when there are one or more missing features. In the experimental scenario, the user's gestures feature $\mathbf{z}_t^{(guser)}$ is the missing modality such that the new features field $\tilde{\mathbf{x}}_a$ is

$$\tilde{\mathbf{x}}_a = (\mathbf{x}_a^{(fuser)}, \mathbf{x}_a^{(faud)}, \emptyset). \quad (3.49)$$

During the initialization of the recall mode, the associator prunes away any the weight j with a frequency $\varphi_j < \tau$, where τ is the pruning threshold. Next, the associator computes the choice activation using (3.47); however, it only calculates the choice activation for the observable feature fields $\mathbf{x}_a^{(fuser)}$ and $\mathbf{x}_a^{(faud)}$. The best choice activation J is selected by

$$J = \underset{j}{argmax} \left(\prod_{k \in \{fuser, faud\}}^K y_{a,j}^k \right). \quad (3.50)$$

After retrieving the best weight $\mathbf{w}_{a,J} = \{\mathbf{w}_{a,J}^{(fuser)}, \mathbf{w}_{a,J}^{(faud)}, \mathbf{w}_{a,J}^{(guser)}\}$, the estimated encoded data $\mathbf{z}'_t$ is

$$\mathbf{z}'_t = \mathbf{x}_a'^{(guser)} \approx \mathbf{w}_{a,J}^{(guser)}. \quad (3.51)$$

The final process decodes the encoded data $\mathbf{z}'_t$ through the decoder function $f_{d_t}(\cdot)$ to estimate upper-limb unit vector $\hat{\mathbf{i}}'_t$

$$\hat{\mathbf{i}}'_t = f_{d_t}(\hat{\mathbf{i}}_{(t-p):(t-1)}, \mathbf{z}'_t), \quad (3.52)$$

where $\hat{\mathbf{i}}_{(t-p):(t-1)}$ is a p sequence of previous upper-limb unit vectors.

3.4.3 Data and its preprocessors

The facial expressions and gestures are the two features for the experiment. The facial expression dataset is the CK+ dataset [46] where it contains 97 subjects expressing various facial expressions from the frontal view. The gestures dataset is the same combination datasets from decoder and encoder model.

There are two simulated datasets, and both of them contain sequences of operator's facial features, audience's facial features, and operator's gestures. The first set of data, identity dataset, simulates event where an operator is talking to three different persons in series. The second set, expression dataset, simulates operator talking to the same individual.

The final products are eleven and four sequences for the identity dataset and expression dataset respectively. Each sequence in the identity dataset (Figure 3-20 left) contains the neutral facial expression throughout the sequence for both the operator and the audiences, and the different audience has a respective gesture. On the other hand, the expression dataset (Figure 3-20 right) contains a mixture of expressions from the operator and audience. Each sequence has four portions, and each portion has a corresponding gesture. The first portion comprises of the operator and audience making a neutral face. The second and third have one of them making a random expression while the other remains neutral. The final portion covers both of them expressing a random expression.

For facial expression, they were randomly selected from a few individuals with a specific expression, and they were extracted with a short sequence of images. From the short clips, they were mirrored and extended to form a 200 frames clip. For every training and testing set, they were randomly selected and sorted into two sets of 100 frames from the 200 frames. Lastly, they were randomly paired with another person by their identity or expressions depend on which dataset.

Once the two facial expressions were paired, the paired facial expressions were randomly matched with a gesture. In this process, the selected gestures must exhibit the same style for about 100 frames so that the unique gesture matches the corresponding paired facial expression.

Finally, there are two encoder models: the gesture and facial encoder models. The gesture encoder model convoluted the gestures features in every five frames; subsequently, it pools 21 frames of the convoluted signals. On the other hand, the facial encoder model convoluted the facial feature within the same period, and it pools 25 frames of the convoluted signal to form the encoded face data (Figure 3-19).

Figure 3-19 The convoluted face features of eleven subjects on the first two PCA axis (left). The convoluted face features of the eleventh participant with the neutral and six different expressions (right).

Figure 3-20 Two sample sequence of data from each dataset: identity dataset (left) and expression dataset (right)

3.4.4 Evaluation criteria

DTW is the evaluation method. The experiments primarily focus on reconstructing the user's gestures given the various user and audience identities and expression. During the learning process, the associator captures the relationship between those three modalities. Subsequently, the associator retrieves the missing modalities, the encoded gesture, given the facial information. Therefore, the experiments were setup to prove the capable of online generalizing the relationship between multimodalities, and the capability to decode a retrieved encoded modality.

During the evaluation process, the aim is finding the best parameters to minimize the error between the aligned decoded signals $\mathbf{v}'_{dtw}$ and the actual signals $\mathbf{v}_{(t+1):(t+T)}$. The first step generates the decoded signals $\mathbf{v}'_{(t+1):(t+T)}$, and the decoded signal at each time step is

$$\begin{aligned} \mathbf{v}'_{t+i} &= f_{d_t}(\mathbf{v}_{(t-p+i):(t+i-1)}, \mathbf{z}'_t(\boldsymbol{\theta}_a); \boldsymbol{\theta}_{d_t}), \\ i &\in \{1, 2, \dots, T\} \end{aligned} \quad (3.53)$$

where $\mathbf{z}'_t(\boldsymbol{\theta}_a)$ is the retrieved encoded data given the associator parameters, and $f_{d_t}(\cdot)$ is the forward pass for the decoder. The next procedure rearranges decoded signal $\mathbf{v}'_{(t+1):(t+T)}$ to get the aligned decoded signals $\mathbf{v}'_{dtw}$. The optimum parameter is

$$\boldsymbol{\theta}_a = \operatorname{argmin}_{\boldsymbol{\theta}_a} \operatorname{MSE}(\mathbf{v}'_{dtw}, \mathbf{v}_{(t+1):(t+T)}). \quad (3.54)$$

4 Experiments and results

4.1 Encoder

4.1.1 Using automated feature extractor to obtain spatiotemporal distributed output for gesture

The automated feature extractor and spatiotemporal distributed output are two of the important characteristics in an encoder. The automated feature extractor should extract the underlying structures of the dataset. Using the extracted features, the encoder should transform the gestures signal into a spatiotemporal distributed output, which is phase-free. The following figures will demonstrate the expected outcomes.

Figure 4-1 Convolution features for CNN (left) and the fifth convolution feature (right)

Figure 4-1 shows the convoluted features learnt by the RBM for a CNN in a 15 FPS dataset. The left figure shows 30 features in two dimensions, where the x-axis is the time, the y-axis is the input, and the intensive (white is higher) represents the correlation between time and the input. To a certain extent, the 30 represented features are different from one another. With more of these features, they can better represent the input signal. On the right, it shows the three-dimensional view of the fifth convoluted feature, and this view provides a better understanding of features respect to time. The learnt features are smooth and continuous with respect to time.

Figure 4-2 shows the effect of a spatiotemporal distributed output in a segment of gesture with two gesture styles. The boundary, which separates the two gesture styles, can be easily spotted. For the input layer, the different in the frequency and intensive separates the two gestures. In contrast, the different in the intensive of the feature activations divide the two gestures for $h^{(2)}$ and $h^{(3)}$. However, there is a delay in the gestures transition in $h^{(3)}$ as compared with $h^{(1)}$ and $h^{(2)}$. From the figure, the encoder output has a time delay, about 25 frames, and it is equivalent to the T window in (3.8).

Figure 4-2 Encoded output given a segment of gesture with two different styles – 2D spatiotemporal distributed output (top) and specific features (bottom) (Refer to Appendix G for the video)

4.1.2 Comparison with other feature extractors

Instead of a local representation, the encoder transforms the input signals into a distributed output, and the output is phase-free. An implicit characteristic from this approach is that the transformed output with similar trait tends to group nearer to one another. Using the objective criteria at (3.14) and (3.15), the aim of this experiment is to find a good encoder model from a list of feature extractors.

The five features, from the various feature extractors, are the raw data, 1D fast Fourier transformation (FFT), 2D fast Fourier transformation (FFT2), CNN, and K-mean. Raw data is the spatial distributed features. These features $f_e: \hat{\mathbf{t}} \rightarrow \mathbf{h}^{(1)}$ were acquired by transforming the upper limb data $\hat{\mathbf{t}}$ with a spatial gesture encoder. The parameters of the spatial encoder were trained using a RBM. The FFT and FFT2 are the 1D and 2D Fourier transformation of the upper limb data $\hat{\mathbf{t}}$ in a period of five second about 75 frames for 15 fps and 25 frames for 5 fps. In both features, only a portion of the transformation results was extracted. The FFT holds the amplitude of the first ten Fourier components for each feature, while the FFT2 stores the amplitude of the first ten Fourier components in the time domain and all the components in the spatial domain. In the end, both the features will have the same number of features, 180, for each time stamp. CNN is a mean pooled network with its local features trained by a RBM or Autoencoder. Using Figure 3-9, T window is 25 frames (five seconds), c window is five frames (one second), and the number of convoluted features is 200. Lastly, the K-mean has its local features trained using fuzzy K-mean, and mean pooling is the pooling strategy. Similarly, the K-mean has a T window and c window of 25 frames (five seconds) and five frames (one second) respectively. The number of clusters is 200.

Figure 4-4 displays the real distributions, unlike the ideal distributions in Figure 3-2 (encoder). The first observation is that the distributions from those classes are overlapping between another. Unlike the distribution in Figure 3-2 (encoder), where the cluster of each class is isolated in their own spaces, the distributions in Figure 4-4 tend to overlap each other to some extent. Besides the overlapping, the raw data also produces distributions with multi-peaks within a single class. Figure 4-3 (top) shows a specific class from the raw data with multi-peaks. On the contrary, Figure

4-3 (bottom) shows that CNN only produces distributions with a central peak. The multi-peaks distribution in a single class is an undesirable distribution for encoder because it implies that there is more than one cluster for the class of data. The last observation is that there is coincided peak in both the raw data and CNN. For the raw data, the coincided peaks occurred in both the principal components (Top of Figure 4-3). For the CNN, the coincided peaks tend to occur on one of the principal components (PCA1 of Figure 4-3 bottom), and the coincided peaks entangle themselves on the other components (PCA2 of Figure 4-3 bottom). In short, the visual result demonstrates the superiority of the spatiotemporal output over the spatial output.

Figure 4-3 The top figure shows the location of the coincided peaks of multiple classes and the multi-peaks for a specific class for the raw data. The bottom figure shows the histogram of CNN with the coincided peaks of two specific classes at PCA1, and the same classes being separated in PCA2.

From Table 4-1, the CNN200-RBM-Mean is the selected model for the encoder. Among other neural approaches that use the (3.15) as the objective function, the CNN200-RBM-Mean is the second best. Both the CNN200-RBM-Mean and CNN200-Autoencoder-Mean have a very close score of 1.770(0.001) and 1.769(0.001) respectively. The t-test gives a P-value of 0.1525, which is not statistically significant. However, the CNN200-RBM-Mean has a lower score with (3.14). The CNN200-RBM-Mean scores about 0.776(0.001), while the CNN200-Autoencoder-Mean scores about 0.799(0.001). The t-test gives a P-value less than 0.0001, which is statistically significant. In terms of (3.15), the neural networks that have the spatial layer scored a lower value as compared to the neural networks without the spatial layer. The RBM and Autoencoder without spatial layer score 1.770(0.001) and 1.769(0.001) respectively. These values are lower than their counterpart with the spatial layer, which is 1.781(0.007) and 1.775(0.000) respectively.

Figure 4-4 Distribution of classes in their new vector space: Spatial transformation and CNN. Each distribution comprises of two histograms and a 2D scatter plot. The two histograms estimate the class distribution for the data on the first and second PCA components, which the x-axis and y-axis respectively.

Table 4-1 Comparison between the raw data and the various encoded features with the objective functions. The CNN and K-mean have their average distances computed from five different training dataset.

Objective Function	Raw data	FFT (15/5 FPS)	FFT2 (15/5 FPS)	CNN200- RBM-Mean	CNN200- RBM-Mean- Spatial	CNN200- Autoencoder- Mean	CNN200- Autoencoder- Mean-Spatial	Fuzzy-K- mean
(3.14)	1.578	1.307 / 1.394	1.387 / 1.396	0.776(0.001)	0.782(0.000)	0.799(0.001)	0.799(0.001)	1.430(0.022)
(3.15)	--	--	--	1.770(0.001)	1.781(0.007)	1.769(0.001)	1.775 (0.000)	--

4.1.3 Selection of the parameters

There are many hyperparameters for a CNN model. The hyperparameters include number of neuron, type of neuron (e.g. rectified [47], [48] or binary neuron), sparsity [49], dropout [48], [50], pooling function, and many more [51]. Due to the large of parameters, this experiment only shows the results on a few selected parameters. The aim of the experiment is to find the optimal configuration across three factors: number of neurons, decay rate constant, and types of decay.

The descriptions of the three factors are as follows. The number of neurons is the number for of convoluted features (Figure 4-1). The higher number of neurons increases the performance, but it might overfit towards the training data. Hence, regulator, sparsity, and dropout are one of the tools to generalize the model. As mentioned, the regulator is one of the devices to generalize the model. The regulator periodically decays the weight given a decay rate, and this prevents any feature from dominating. L1-norm and L2-norm are two different ways of regulating the weights. Both have their own characteristic, L2-norm produces a more distributed weight while L1-norm produces a sparser weight.

Table 4-2 Test result using (3.15) for CNN-mean with rectified dropout neuron and Adadelata learning rate. The rows represent the number of neurons, and the columns store two different parameters: the type of decay and the decay rate.

		Regulator Parameters			
		L1-norm		L2-norm	
		1e-5	1e-4	1e-5	1e-4
Number of Neurons	50	1.808(0.001)	1.815(0.001)	1.810(0.001)	1.810(0.000)
	100	1.783(0.002)	1.788(0.003)	1.785(0.001)	1.785(0.001)
	200	1.770(0.001)	1.779(0.001)	1.771(0.002)	1.772(0.002)

From Table 4-2, the best performance comes from CNN with 200 neurons, 1e-5 decay rate, and L1-norm. The higher number of neurons improves the performance of the model. The L2-norm produces consistent results given the two decay rates; on the contrary, L1-norm results have a higher difference between the two decay rates. However, the L1-norm yields a better performance against L2-norm when the decay rate is 1e-5 at all number of neurons. As a result, CNN with 200 neurons, 1e-5 decay rate, and L1-norm is the chosen CNN for the future experiment.

4.2 Decoder

4.2.1 Regression with the encoded data

Figure 4-5 A zoom in on Figure 4-7 to show the convergence of the periodic signal (Refer to Appendix G for the video)

In this visual demonstration (Figure 4-7), the decoder synthesized nine distinctive encoded data. The model demonstrates consistency in the reconstructed gestures over time. From Figure 4-5, the generated output converges to their periodic dynamic in less 1/10 of the duration, which about two seconds, and the generated output is very consistency in terms of its cycles and amplitude pattern throughout the duration. Moreover, the model also displays smoothness in the transition between two gestures. The transitions are smooth between the two gestures in the detail skeleton view. Intrinsically, the model might have learnt the transition dynamic during the training process. The model also captures contextual dynamics even though the inputs are only the encoded data and previous states. From Figure 4-6, the encoded data is a constant throughout the same gesture style (interval of 100), yet the internal states store the dynamics encapsulated by encoded data. In conclusion, the selected decoder model has shown promising results as it can reconstruct various gestures given the encoded information in this scenario.

Figure 4-6 A zoom in on Figure 4-7 to show the difference in behavior between the input and the internal state.

Figure 4-7 Nine continuous synthesized gestures with 400 gate size. Each synthesized gesture is about 100 frames in duration, which is about 20 seconds for five fps. There are six rows of data where their x-axis represents the time. The first row shows the three of the normalized output signals, which the x-y-z signals of the right wrist. The second to fourth rows display three gates activation: input, forget, and output gates respectively. The fifth row is the cell status, while the last row is the encoded data. At the top, the detail skeleton view displays a segment of the synthesized gestures from 375 to 425, where each body encapsulates five frames of data

4.2.2 Regression with the re-tuned encoded data

In this section, it displays the capability of decoder model to synthesize gestures given random encoded data, which might not be in the training data. In order to get a comparison, the new encoded data $\mathbf{z}_{n_i}$ is

$$\mathbf{z}_{n_i} = \mathbf{z}_{n_0,t} + \epsilon |\mathbf{z}_{n_0}| \circ \mathbf{U}_i, \quad (4.1)$$

where $\mathbf{z}_{n_0}$ is the original encoded data, $\mathbf{U}$ is a uniform random vector ranges from $[-1, 1]$, ϵ is a gain value, and $|\mathbf{z}_{n_0}|$ is the absolute value of $\mathbf{z}_{n_0}$.

Figure 4-8 Spatial difference between two encoded data: original and additive of noise. The original encoded data (blue) has a signal with a height different of 0.269, and the modified encoded data (red) has a signal with a height different of 0.418.

Figure 4-10 demonstrated the capability of the decoder to recreate gestures of similar trait given an encoded data. Comparing the spatial information, there is a slight difference between the two signals. For instance, n_2 of $\epsilon = 0.7$ is also gesturing an inverted triangle in the front view; however, the vertical height of the triangle is about 1.55 times larger than the original gestures (Figure 4-8). Besides spatial information, there is also a slight different in the periodic time between the two signals. The frequency of that same gesture is also 15.6% higher the original gestures. The noise within the gestures can be manipulated by varying the noise gain ϵ . Figure 4-10 shows the noise gain of $\epsilon = 0.3$ and $\epsilon = 0.7$. When $\epsilon = 0.3$, the synthesized gestures are closer to the original signal. On the other hand, the synthesized gestures inherit more variance when noise gain $\epsilon = 0.7$. In summary,

the gestures with noise contain trait of the original gestures, but they are not exactly the same. These differences can be beneficial if it is controllable.

Figure 4-9 Temporal difference between two encoded data in term of cycle time. The original encoded data (blue) produced a signal with a cycle time of 15 steps (0.333fps), while the encoded data with noise (red) produced a signal with a cycle time of 13 steps (0.385fps).

Figure 4-10 Comparison between original (blue) and the neighbors (red) reconstructed-gestures with 0.3 and 0.7 gain values. The first column represents the gestures motion for the right wrist in the three two-dimensional spaces, and three subgraphs represent the front, top, and side view. The second column displays right y-axis motion with respect to time. The last column shows a set of elements of the modified (red) and original (blue) encoded data. The figure has two main rows, and each main row has a unique noise gains.

4.2.3 Comparison of the LSTM with other feature extractors

One objective of this experiment is to check whether the obtained result in Section 4.1.2 is corrected. The next objective is to check the decoder performance as a gesture generator. From Table 4-3, CNN-RBM-Mean provided the best signal for the LSTM, which matches the encoder results. With a mean score of 0.257 for DTW_MSE, CNN-RBM-Mean has a higher performance of 17.628% when compared with respect to the Fuzzy-K-mean, which is in second place. In terms of the MSE score, CNN-RBM-Mean also has the lowest score. However, there is a score different between the result of MSE and DTW_MSE. This difference implies that the synthesized gestures most likely match the actual gestures spatially, but the cycle time might be off by a maximum of five frames, which correspond to the DTW local constraint w at (3.36).

Table 4-3 Comparison between different encoded features with gate size of 300 and L2-norm. MSE is the square error of the reconstructed signals with respect to the actual signals, while the DTW_MSE is the square error of the realigned reconstructed signals with respect to the actual signals.

	FFT	FFT2	CNN-RBM-Mean	Fuzzy-K-mean
BPTT Ratio	0.3	0.3	0.2	0.3
Lambda	1e-6	5e-6	5e-6	1e-6
MSE	0.547(0.004)	0.629(0.002)	0.473(0.004)	0.528(0.003)
DTW_MSE	0.342(0.002)	0.401(0.004)	0.257(0.004)	0.312(0.002)

4.2.4 Selection of the parameters and the effects of high neuron size

This section contains two results to show the effect of various factors affecting the decoder training strategy. Figure 4-11 displays the DTW_MSE results gathered from four factors, which give 24 different decoder parameters. The four factors are the connecting LSTM internal state $\tilde{\mathbf{y}}_t$, size of the gate N^{gate} , weight decay constant λ , and type of backpropagation γ . The connecting LSTM internal state is the set of internal state pass to the next time step as input. The size of the gate governs the number of elements in all the gate and cell. The weight decay constant regulates the model by periodically reducing the weight. Lastly, the type of back propagation alternates the training strategies from truncated BPTT to ratio BPTT. Table 4-4 provides the link between the parameter size and DTW_MSE result. There are three main columns in the table. Each column contains the parameter size from each weight, not including bias. The parameter size is calculated by multiplying the input to output, where the input includes the five previous upper-limb state, the encoded data, and previous LSTM internal state. Each column also holds the total parameter size and its difference with respect to the last configuration; in addition, it also stores the DTW_MSE result and its changes with respect to the last configuration.

From Figure 4-11, the best configuration is (2, 2, 2, 2) where $\tilde{\mathbf{y}}_t = \mathbf{y}_{cell,t}$, $N^{gate} = 300$, $\lambda = 5e^{-6}$, and $\gamma = 0.2$. The connecting LSTM internal state $\tilde{\mathbf{y}}_t \in \{\emptyset, \mathbf{y}_{cell,t}\}$ plays an important role in bringing down the median and ranges. Based on the t-test result on the internal state, the P-value is >0.0000 , which is significant. Given $\tilde{\mathbf{y}}_t = \mathbf{y}_{cell,t}$, the size of the gate N^{gate} provides an additional push downwards; however, this effect is not as significant when $\tilde{\mathbf{y}}_t = \emptyset$. The P-value on the size of gate given $\tilde{\mathbf{y}}_t = \mathbf{y}_{cell,t}$ is >0.0000 , which is significant. On the other hand, the P-value is 0.0242 when the t-test only based on the size of the gate. As of remaining factors, there is no consistency significance between their values.

Increasing parameters improves the performance of the model is a well-known concept. However, naively increasing the size of the parameter might not be efficiency to improve the performance. From Table 4-4, increasing the size of the parameter does not effectively improve the performance. Using $\tilde{\mathbf{y}}_t = \emptyset$ and

$N^{gate} = 300$ as the benchmark, the performance increases by 7.24% and the size of parameter increases by 11.94% when there is an addition input $\tilde{\mathbf{y}}_t = \mathbf{y}_{cell,t}$ and a reduction of gate size to $N^{gate} = 200$. It is equivalent of expanding 1.629% of the parameter size to gain 1% of accuracy. However, the efficiency deteriorates when $\tilde{\mathbf{y}}_t = \mathbf{y}_{cell,t}$ and $N^{gate} = 300$. The efficiency decreases to 8.951% of parameter expansion to yield 1% of accuracy. There is an increase in parameter expansion by 5.494 times to yields the same result. As a result, the above experiments stop investigating on adding more input or increasing the gate size because higher parameter size will affect the computation time during runtime.

Figure 4-11 The DTW_MSE results gathered from a permutation of the four factors with three replicas.

Table 4-4 The relationship between the parameter size respects with to the DTW_MSE result

	$\tilde{\mathbf{y}}_t = \mathbf{y}_{cell,t}$ and $N^{gate} = 300$	Size of $\mathbf{W}_i$	$\tilde{\mathbf{y}}_t = \mathbf{y}_{cell,t}$ and $N^{gate} = 200$	Size of $\mathbf{W}_i$	$\tilde{\mathbf{y}}_t = \emptyset$ and $N^{gate} = 300$	Size of $\mathbf{W}_i$
$\mathbf{W}_{in,gingforout}$	$(90+200+300) * 300$	177,000	$(90+200+200) * 200$	98,000	$(90+200) * 300$	87,000
$\mathbf{W}_{out}$	$300 * 18$	5,400	$200 * 18$	3,600	$300 * 18$	5,400
Total Parameters		713,400		395,600		353,400
Ratio		↑ 101.87%		↑ 11.94%		-
DTW_MSE		0.257(0.003)		0.269(0.006)		0.290(0.008)
Ratio		↑ 11.38%		↑ 7.24%		-

4.3 Associator

4.3.1 Multimodalities spaces and the relationships between different modalities

Figure 4-12 The visualization of the features spaces from Figure 4-15, using PCA. The three axis of the figure are the first three PCA components. In this figure, the train data are the blue dots, the pruned weights are the red diamond, and the learnt weights are the grey circles. (Refer to Appendix G for the video)

Given a data in Section 3.4.3, Figure 4-12 displays an output of associator using that training data. The first observation is that there are two possible states in the training data. From Figure 4-12, there are four distinct clusters, convergence zone, and four distinct trails leading to a cluster, transition phase. The convergence zone holds more training data as compared to the transition phase. Given an interval of 100 frames (Figure 4-15) at each state, most of the training data are in the convergence zone forming a region of interest. Finally, the associator has captured the region of interest while kept the knowledge light. In this instance, one or more final weights are in the convergence zone, while the pruned weights are mostly on the transition region. The size of the final weight is 55.6% smaller than the total weight.

4.3.2 Regression with retrieved encoded data given the facial features

The above experiment has demonstrated the associator capability to learn multimodal signals, and the following experiment demonstrates the associator ability to retrieve missing modality given the observable modalities. In general, Figure 4-15 shows the associator capabilities to learn and retrieve the operator's encoded gestures so that the gesture decoder can reconstruct the motion.

Figure 4-13 The operator's gesture signal and the reconstructed gesture signal.

The following is a breakdown of the observations in this particular sequence (Figure 4-15) when comparing the actual gesture with the reconstructed gesture. During recall mode, the associator synthesized the full sequence of gestures given the facial features and the weights. The reconstructed gestures have the similar dynamic as the actual gesture. From Figure 4-13, both the signals are similar but not the same. The actual gesture has mostly three pointed peaks when the time step is <100 , but the reconstructed gesture has only two peaks in all instances. Similar to encoder model, there is a transition gap between states. From Figure 4-13, the actual gesture always has a new gesture style at the 100th time interval; however, both the reconstructed gestures have the start of the new gesture styles slightly after the 100th time interval (estimated 25 frames). The reconstructed signals are less noisy than actual signal. From Figure 4-13, the actual gesture is noisy when the values more than 0.5; however, this noise is not observed in the reconstructed gesture.

The subsequent breakdown looks into the encoded data from both learning and recall mode. There is a huge size reduction of knowledge. The training data has a time step of 400, and the initial learning process has captured the modalities relationships with only 18 weights. With pruning, the size of the knowledge is further reduced to eight. It is a 50-time reduction from the original time step to the final knowledge base. From this perspective, the raw encoded data for the gesture is noisier than the retrieved encoded data. From Figure 4-14, the actual encoded data are very noisy even within the same state; on the other hand, the retrieved encoded data are more consistent.

Figure 4-14 On the top are the operator's encoded gesture data and the retrieved encoded gesture data. On the bottom are the activation weight during learning and during recalling.

Figure 4-15 The training and recalled sequence from an expression dataset. In learning mode, the input is all the feature fields, and output is the winner nodes. During the recall mode, the input is only the facial feature fields, and the output is the retrieved gesture encoded data and decoded gesture signals.

4.3.3 Selection of the parameters

The results in this section are gathered from both the dataset with the permutation of three factors. The factors are vigilance thresholds, pruning threshold, and learning rate. The vigilance thresholds contain two factors face and gesture. The range of the vigilance thresholds is across 0.7 to 0.9 with an interval of 0.1. The pruning threshold is set at no pruning, 0.04, and 0.05. Lastly, the learning rate is set to be adaptive, 0.05, 0.1, and 0.2. The aim of the experiment is to find the best configuration that gives a good accuracy with the least amount of weights.

There are a few observations from this experiment. Pruning drastically reduces the number of weight. From Table 4-6, the total number of weight drastically reduces from “more than 40” to “less than 10” with pruning. The number weight does not reduce as much with the increase of pruning threshold. Secondly, there might have a crossover interaction between learning and pruning threshold. From Table 4-5, no pruning threshold with adaptive learning rate yields 0.092 for DTW_MSE. The values increased to 0.106 when the learning rate is increased to $\eta = 0.2$. The trend was not too significant when the pruning threshold $\tau = 0.01$, and the trend was opposite when the pruning threshold $\tau = 0.05$. Lastly, the optimum parameters are adaptive learning rate, pruning threshold at 0.04, face vigilance threshold is 0.9, and gesture vigilance threshold is 0.7. The best DTW_MSE was not chosen because the chosen model yields better balance in the other results. The second best DTW_MSE was selected because its number of weight is seven times lesser than the best DTW_MSE.

Table 4-5 The median and standard deviation of the best DTW_MSE results with respect to its learning rates and pruning threshold given the vigilance thresholds at Table 4-7. The rows of tables represent the learning rates, and the columns of the tables denote the different pruning thresholds

	$\tau = 0.00$	$\tau = 0.04$	$\tau = 0.05$
$\eta(\varphi_j)$	0.092(0.051) ₁	0.096(0.048)₂	0.109(0.046)
$\eta = 0.05$	0.103(0.058)	0.101 (0.049)	0.106(0.042)
$\eta = 0.1$	0.102 (0.053)	0.103(0.046)	0.103(0.046)
$\eta = 0.2$	0.106(0.053)	0.101(0.047)	0.099(0.036)

Table 4-6 Total number of weights after pruning with respects to the DTW_MSE result in Table 4-5

	$\tau = 0.00$	$\tau = 0.04$	$\tau = 0.05$
$\eta(\varphi_j)$	49(10.2881)	7(1.3211)	6(1.5395)
$\eta = 0.05$	223(34.0363)	6(1.8141)	5(1.5725)
$\eta = 0.1$	221(34.2489)	5(1.385)	5(1.1015)
$\eta = 0.2$	56(12.1873)	5(1.5507)	4(1.461)

Table 4-7 The vigilance thresholds with the best DTW_MSE result at a specific learning rate and pruning threshold.

	$\tau = 0.00$	$\tau = 0.04$	$\tau = 0.05$
	$\rho^{(f_*)} \rho^{(g_{user})}$	$\rho^{(f_*)} \rho^{(g_{user})}$	$\rho^{(f_*)} \rho^{(g_{user})}$
$\eta(\varphi_j)$	0.7 0.8	0.9 0.7	0.8 0.7
$\eta = 0.05$	0.9 0.9	0.8 0.7	0.7 0.7
$\eta = 0.1$	0.8 0.9	0.7 0.7	0.7 0.7
$\eta = 0.2$	0.7 0.8	0.9 0.7	0.9 0.7

4.4 CATR with a PLBM

4.4.1 Inserting a valid gesture style in decoupling situation with different sequences

To better simulate a decoupling situation, the following experiment will retrieve the encoded gesture data from a different sequence. Figure 4-17 shows the sequence during the training and the recall mode. If the training sequence is ABCD, then the recall sequence is CADB. The aim of this experiment is to demonstrate the flexibility of the PBLM.

Figure 4-16 A detail descriptions between the actual gesture and reconstructed gesture from Figure 4-17. (Refer to Appendix G for the video)

The following are some of the observations obtained from this demonstration. The recall output signal follows the rearranged sequence. From Figure 4-17, the training sequence is ABCD, and the facial features were re-inserted into the model with a sequence of CADB. The synthesized gesture was reconstructed in aligned with the new sequence CADB. The input and the reconstructed sequence can be visually recognized and matched. From Figure 4-16 (extracted from Figure 4-17), the difference in dynamic between the various states are visible in both the gesture signals. The top and bottom sequence can be easily matched by comparing the dynamic in each state. However, the reconstructed signal is slightly different from the original signal. Especially for state “C”, there is clearly difference in its cycle time.

Figure 4-17 The synthesized gestures in a different sequence, CADB, as the training sequence, ABCD. (Refer to Appendix G for the video)

4.4.2 Reconstructing gestures in an unstable connection situation

An inserting scenario is a situation when the network connection is unstable. In this scenario, the system needs to insert a valid gesture motion whenever the gesture signal is missing at a probability of $P(connect) < X$. This experiment compares the two proposed approaches and imitation approach with the actual gesture and the decoded gesture without missing connection. The objective of the experiment is to understand the effects of the connection on the accuracy and smoothness of the various output signals.

In this experiment, there are two experimental criteria to measure the degree of accuracy and degree of jerkiness. For degree accuracy, the experiment compares the new results with the input gesture using the DTW_MSE equation from eq. (3.39).

On the other hand, the degree of jerkiness $J = \frac{(\sum_{t=1}^T \sum_{m=1}^M \frac{\partial^3 y_{m,t}}{\partial t^3})}{TM}$ is the average jerk measurement across a multivariate signal, where $y_{m,t}$ is the magnitude of the m feature at t time.

There are two proposed approaches to the inserting scenarios. Like the concealment scenario, the first approach is to retrieve the encoded gesture data using the remaining modalities from the associator. In this scenario, the remaining observable modality was the audience facial expression. The PLBM was required to infer the encoded gesture data whenever there was a disconnection. As for the second approach, it inserts the previous known encoded gestures into the missing time steps. This approach works because of the encode module groups similar data together. For instance, if the operator was executing a specific gesture style before the disconnection, the previously known encoded gesture data should be a valid data for the insertion.

From Figure 4-18, these are some observations on the PLBM output signals. The first observation is that three PLBM outputs are very similar to one another. From the second to the fourth rows, the three PLBM output are like one another in terms of their dynamic characteristics. All their signals have the same gesture style at various point of time. For all PLBM output, there is a transition gap from one gesture style to the other. Like the other experiments, all the PLBM output also

have an estimated 25 timesteps transition gaps. The degree of smoothness in all the PLBM output is better than the user gesture. When comparing them with the user gesture in the first row, the signals from the PLBMs seem to have less discontinuity.

From Figure 4-18, there are also some observations on the signal from previous known user gestures. Firstly, the previous known user known is in-phase with the user gesture. Comparing the signals from first and last rows, both the signals have their peak and off-peak timing nearly coincide with each other. This should be the case since the signal follows exactly as the user gestures unless there is a disconnection. However, there is more discontinuity in its signal compared with the user gesture. The discontinuity often occurs during the transition from disconnection to connection, and some examples are at time step 170 and 237.

Figure 4-18 Segment of the five different signals at $P(\text{connection}) = 0.5$. The first signal is the actual user gesture Y-motion in respect to the right wrist. The second signal is the PLBM reconstructed gestures given an uninterrupted encoded data. The third and fourth signals are the reconstructed gesture by the first and second approaches. The fifth signal is the gesture signal from spatial imitation. (Refer to Appendix G for the video)

Figure 4-19 The degree DTW_MSE (top) and jerk (bottom) with respect to the degree of connection. Each data from first three signals is computed from a median of 75 signals, which is from 15 training sequences with five replicates. Each signal is generated from different connection sequences.

Unlike Figure 4-18, Figure 4-19 shows the overall performance of the various signals across different degree of connection. The following are some observations on the PLBM results. The PLBM with previous known encoded data is more accurate than the PLBM with associator even when there is a high drop rate. From Figure 4-19 (top), The PLBM with previous known encoded data has a lower DTW_MSE value than the PLBM with associator. The trends continue from 0.1 to 0.8 of the connection index, then they converge with the uninterrupted encode data at 0.9. All the PLBM signals are a smoother than the other two signals. From Figure

4-19 (bottom), all the PLBM signals have their degree of jerk around 0.45 to 0.50, and the user gesture (uninterrupted) has a degree of jerk at 0.88. The imitation using previous poses has its degree jerk above 0.6, eight out of nine times.

Similarity, there are some observations on imitation with previous known posed data. There is an inverse non-linear relationship between the accuracy and smoothness of the signal. The imitation approach has an incremental degree of jerk from 0.1 to 0.7, and its degree of jerks slowly decrease towards the input gesture signals at 0.88. On the other hand, the degree of DTW_MSE decreases non-linearly towards zeros. The degree of jerk is above the user gestures (uninterrupted). When the degree of connection is between 0.5 and 0.9, the degree of jerk is above the user gestures (uninterrupted), which is 0.88.

4.4.3 Reconstructing gestures in a tight situation

In a re-tuning scenario, the original gesture signal does not fit into the constraint in posed by the remote environment; as a result, the PLBM re-tunes the encoded gesture data to fit the reconstructed gesture into constraint while preserving the expressiveness, as shown in Figure 4-20. In this experiment, there are two approaches to be compared. The first the approach is a reactive mechanism. A reactive mechanism replaces any out-of-bound points with its previous data. The second approach is the deliberative mechanism. The following will describe its procedures and outcome. The aim of the experiment is to compare the difference between the reactive and deliberative safety planner in terms of accuracy and smoothness.

Figure 4-20 The three overlapped skeleton points with boundary constraints. The first and second columns are the actual and the reconstructed gesture without considering the constraints. The third column is the output with constraints considered. (Refer to Appendix G for the video)

4.4.3.1 PLBM with evolutionary algorithm

The current PLBM uses an evolutionary algorithm (EA) [52] to find a good candidate in the collision avoidance scenario. The first procedure in an EA is to populate a population of candidates, and it involves mutation and crossover from the selected samples. During first iteration (Figure 4-21), the old winner (x) would populate its surrounding through mutation and crossover. These processes produced numerous candidates (o and w). The second step is to calculate the score of each candidate and find the best few candidates. Depending on the objective functions, each candidate would score differently, and only a few good candidates (w) were chosen to continue to the next iteration. The third step is to repeat steps one and two until a good candidate is found. In second iteration (Figure 4-21), the previous winner (x) would continue to populate a new population. Subsequently, each sample in the population would sort by their score. A list of good candidates would be brought over to the next iteration. This process would continue until one of the candidate scores within an acceptable threshold.

At every time step, the PLBM with EA uses two objective functions to examine the candidates; subsequently, the model selects a batch of good candidates and progresses until the candidates converge. The first function is the DTW_MSE that measures the closeness of the new projected motion with respect to the projected motion from old encoded data. In Figure 4-22, the second row shows that the 25 candidates eventually converge, and the DTW_MSE median and minimum are decreasing during the process. From observation, the new projected motions were growing closer to the old projected motion. Next, the second objective function is the total collision counts in the projected motion. In Figure 4-22 (right wrist), the old encoded data produced a few collision points at x-axis and z-axis, and the re-tuned encoded data immediately resolved the constraint in the second iteration. In short, the PLBM needs to optimize every incoming encoded data based on the two objective functions at every time step so that the model can find a collision-free and expressivity gestures.

Figure 4-21 An example of an evolutionary algorithm, and the contours are the scores for the given features, y_1 and y_2 .

Figure 4-22 The iteration results of the evolutionary algorithm in the PLBM at one of the time step.

4.4.3.2 Visualization of the output for PLBM with EA

Figure 4-23 The motion points of the wrists with constraints in different spatial views for 300 time steps. The motion contains three distinct gesture styles. Two constant constraints were deliberately set on the x-axis and z-axis; as a result, the first and second columns have few of its points outside the boundary

Based on Figure 4-23, these are some observations on the PLBM with EA. The PLBM with EA can produce collision-free gestures. From the figure, the PLBM with EA re-tuned the gestures so that all the points were within the boundary (third column of Figure 4-23). In addition, the gestures might have similar expressivity as the other two models. Given the spatial motion points, there is similar pattern generated across the three different gestures. The black arrows in Figure 4-23 show the general direction of the patterns.

4.4.3.3 Comparison between reactive and deliberative safety mechanisms

Figure 4-24 The median of various evaluation criterions over 15 training sequences with the boundary set at {x-min: -0.4, x-max: 0.4, y-min: -0.6, y-max: 0.6, z-min: -0.45, z-max: 0.15}.

These comparisons aim to find the difference between the reactive and deliberative safety planner in terms of accuracy and smoothness. Unlike reactive planner, the deliberative planner produces a smoother output. From the bar chart in Figure 4-24, the encoded data with constraints (deliberate) has a lower amount of jerk in its signals as compared to the actual motion with constraints (reactive). It is even more obvious in the fourth row of Figure 4-25. The actual motion with constraints (reactive) would crop off any new data points that fell outside the boundary to prevent a collision. As a result, the signals from actual motion with constraints (reactive) might be lack of continuity and smoothness. Besides smoother signals, the PLBM with EA also preserves the expressiveness of the gestures. In comparison with the signals from encoded data without constraints, which scores 0.559 in DTW_MSE, the encoded data with constraints has a DTW_MSE result on average of 0.645. This implies that both the reconstructed signals are relatively similar to the actual motion since the encoded data without constraints is similar to the actual motion.

Figure 4-25 Motion points of the wrists with constraints in different spatial views

5 Conclusion and discussion

The customizable anthropomorphic telepresence robot (CATR) is viewed as a potential solution for increasing the degree of social presence in a computer-mediated communication medium towards face-to-face communication. However, there are issues in teleoperation that might deteriorate the efficiency and expressiveness of the CATR. These issues involve decoupling, network condition, and situation awareness. Decoupling is an event where the operator chooses to commit part of his nonverbal modalities to other tasks, instead of the interaction. For instance, an operator was conversing with a remote participant while typing on the keyboard. In most cases, the CATR would either become stationary or continue to map the operator's gesture. Both the outcomes are undesirable because those gestures are not congruent to the context. The second issue is the network condition that can be affected by many factors, and some of them include signal interference and physical distance between the devices. When the success rate of the incoming signals is uncertain, some types of data representations, such as poses information, might cause the CATR motion to be jerky and uncanny. For the last issue, the operator's situation awareness of the remote environment is limited, and it might pose a danger to the remote participants. One of the solutions is to deploy the reactive safety mechanism on the CATR. Often, this technique distorts the original signals to fit the environment constraint; as a result, the CATR motion might not be smooth, and it might not deliver the actual intents. As a result, the perception-link behavior model was proposed and developed as a supporting module for a novel CATR operator interface to mitigate the above issues without compromising the CATR expressivity and operator's cognitive load.

5.1 Contributions

The aim of the research is to study and develop a module that is capable of solving the teleoperating issues without degrading the CATR expressiveness and overloading the operator's cognitive load. The details of the objectives and the discussions are as follows:

5.1.1 Low operator's cognitive load for using the CATR

The PLBM adopts techniques and interface that require a lower cognitive load. Firstly, the natural interface is the acquisition technique for the gesture in the PLBM. With a minimum amount of memorization or remapping, the natural interface only requires the operator to perform his gestures as if it is a face-to-face conversation. Furthermore, most of the modules are automatic. For instance, the operator just needed to gesticulate. The encoder would automatically encode the gesture signals into its corresponding distributed output, and the interface would transmit the distributed output of the gesture and other modalities to the remote system. Subsequently, the associator would automatically update its parameters. Once the decoder receives the encoded data, it would automatically synthesize the current poses of the signals. Except for the gesticulation process, the rest of the processes do not require any human involvement.

5.1.2 Expressive CATR's behaviors to imitate personal gestures

Figure 5-1 Convolved features from CNN-RBM-Mean.

For expressivity with the natural interface and deliberate safety mechanism, the encoder of the PLBM transforms all its input signals into a spatiotemporal

distributed data to perverse their expressivity, then the decoder can reconstruct the time signals with the encoded data without much distortion. The CNN-RBM-Mean is the selected model because it can represent the input signals than other representations, e.g. FFT, CNN-Autoencoder-Mean, and Fuzzy K-mean. The result of the experiments shows that the CNN-RBM-Mean has the lowest intra-to-inter ratio when compared with rest of the representations. In the first place, the CNN-RBM-Mean scores 0.776 ± 0.001 , and the second is CNN-Autoencoder-Mean that obtains 0.799 ± 0.001 . The third, FFT2, has 1.394. Besides, the features of the CNN-RBM-Mean capture relationships between the time and gestures signals. Figure 5-1 shows the convoluted features extracted by RBM. The features display the ability of RBM capturing the relationships of multivariate signals with time. By combining the encoder and decoder, the PLBM can reconstruct the encoder data to its original dynamic signals. The CNN-RBM-Mean provided the best data for the LSTM, which matches the encoder results. The CNN-RBM-Mean has a higher performance of 17.628% when compared with respect to the Fuzzy-K-mean, which is in second place. Visually, Figure 5-2 shows nine encoded data (third row) and their reconstructed signals (second row). This demonstrates the capability of the decoder to reconstruct a sequence of cyclic gestures given its corresponding encoded data.

Figure 5-2 The output of a gesture decoder. There are nine different encoded data at each interval, which is 100 time steps; as a result, the decoded gestures have nine different styles.

5.1.3 Handling decoupling situation with associator in PLBM

Figure 5-3 Visualization of the features spaces, from Figure 4-15, using PCA

The associator model is a multi-channel adaptive resonance associative map (multi-channel ARAM) that is capable of handling decoupling situation. Firstly, the ARAM can learn relationships between different observable modalities. A conversation is between two or more interactive partners, and their behaviors and status play a role in the conversation. In the experiment, a simulated dataset involving operator's facial features and gestures with audience's facial features was created. The experiment shows that the ARAM is capable of locating the cluster centers within each data sequence (Figure 5-3), and the cluster centers are the location where the operator's behaviors converge with the audience's behaviors. During the decoupling situation, the ARAM can retrieve the missing modal given the observable modalities. For instance, the model can exploit the learnt rules between different modalities, and it can retrieve the congruent and coherence behaviors to replace the missing modalities. In Figure 5-4, the reconstructed gesture is on right the figure, and the original gesture is on the left of the figure. Both of them have a similar set of facial features from an operator and audience. Because of the similar observable modalities, the regenerated gestures are also similar to the original gestures. In conclusion, the associator model provides the substitute capability in a CATR interface. It can learn the associative rules between three

modalities, and it recalls the missing encoded gestures signals given the operator's facial features and audience's facial features.

Figure 5-4 Retrieving gestures motion give a similar encoded facial data as the learning mode.

5.1.4 Generating expressive and smooth CATR gesture during intermittent connection

During an intermittent connection condition, the PLBM still can produce expressive and smooth gesture using its previous known encoded data. Using previous known encoded data, the PLBM can generate signals that are similar to the PLBM without interruption since the encoder module groups similar data together. For instance, if the operator was executing a specific gesture before the disconnection, the previously known encoded gesture data should be a valid data for the insertion. As shown in Figure 4-19 (top), it yielded the best results in comparison to the PLBM with associator. Even though the imitation using previous known pose has a lower DTW_MSE after 0.2, the PLBM with previous known encoded data is less jerky than imitation approach. From Figure 4-19 (bottom), the PLBM with previous known encoded data has a consistent degree of jerk at about 0.47. On the other hand, the imitation approach has an incremental degree of jerk from 0.1 to 0.7, and its degree of jerks slowly decrease towards the input gesture signals at 0.88.

5.1.5 Collision-free and expressive CATR gesture using PLBM with evolutionary algorithm

Figure 5-5 The three overlapped skeleton points with boundary constraints. The first and second columns are the actual and the reconstructed gesture without considering the constraints. The third column is the output with constraints considered.

With an evolutionary algorithm (EA), the PLBM can produce a synthesized gesture that is collision-free and expressive. In general, the PLBM with EA can re-tune any gestures into the constraints, and its expressiveness matches the encoded data without constraint. Unlike imitation approach (Figure 5-5 first column) and the encoded gesture data without constraints (Figure 5-5 second column), the EA algorithms with the right objective function can find a good encoded data to fit the constraint. From the third column of Figure 5-5, the reconstructed gesture fits into the black boundary (collision-free); in addition, the reconstructed gesture is also like the actual motion in terms of the object function (DTW_MSE). Even though reactive planner scores a better result in terms of the DTW_MSE score, the PLBM with EA is a deliberative planner that ensures the smoothness of the signals. The reactive planner has accuracy about 0.125, while the deliberate planner has accuracy about 0.625. Most of the time the reactive planner will have the actual motion as input; as a result, its DTW_MSE measure will be higher. However, the output of the reactive planner will be jerky whenever there is an obstacle. In this case, the deliberate planner has a jerky index about 0.625, and the reactive planner has a jerky index about 0.900.

5.1.6 Conclusion

In conclusion, the aim of the research is to study and develop a module that can solve the teleoperating issues without degrading the CATR expressiveness and overloading the operator's cognitive load. The summaries of the PLBM to obtain the goals are as follows:

1. The PLBM automatically encodes gestures into a spatiotemporal distributed output, and it can decode the distributed output back to time signals with minimum distortion.
2. For minimum cognitive load, the PLBM adopts the natural interface to acquire the gesture motion. The PLBM is a combination of the automatic encoders, decoders, and an associator, which require no human intervene.
3. With the associating capability, operator can also delegate the gesturing generation to the PLBM if the operator's hand is occupied by other tasks.
4. The PLBM can create a variation of the decoded gestures. With this capability, it can improve the safety of the interactant without compromising the expressiveness.
5. If the connection was not stable, the CATR can continue to gesticulate smoothly and congruently by inserting the pervious known encoded state.

5.2 Limitations

In the current state, there are a few assumptions made on the PLBM. The first assumption is that the PBLM is a user-assistance model, and the primary goal is to improve on the operator's experience when the operator is using the system. For instance, the operator should be the person talking to the audience, as the system would not be intelligent to make a proper conversation. The second assumption is the consistency of the operator's mood towards the audience during a conversation. Since the PLBM was designed to allow any operator to operate without prior-knowledge of the operator and audience, the PLBM will need to learn the behaviors of the operator and audience on the fly. As a result, system has to treat the knowledge acquire (during the calibrating period) as the genuine behaviors.

Besides the few assumptions, there are also a few limitations for the PLBM. Firstly, the completeness of the encoded features and correctness of the decoded features are dependent on the training data. The encoder and decoder are unsupervised models, and their features are only governed by the three datasets. This amount of data might not be adequate for a wide range of conversations. The second limitation is the time consumption to find an optimal candidate to suit the environment is not consistent and occasionally high. The current approach adopted the evolutionary algorithm that might require a significance of iterations, which involve simulating and evaluating the projected signals. In the worst case, the time needed might disrupt a conversation. Lastly, there is an averaging component for the time signal with phase, and the averaging component is to remove the phase information. However, the averaging component will also form a transition phase. This transition phase might become a significant lag if the averaging time is large.

5.3 Future Works for some of the limitations

One of the future experiments will focus on increasing the complexity of the database. A data driven model is as good as their trained data; hence, increasing the variety of the data could further generalize the model. The current datasets were taken from existing databases with specific purpose; for example, Microsoft action dataset only has 12 skeleton motions for gaming. Hence, it will be beneficial if the encoder and decoder modules can build its features from more databases to improve their generalization. Secondly, it is to tackle more modalities. Since nonverbal cues cover a wide spectrum of components, it is not comprehensive to restrict the PLBM to only a few modalities. Hence, some efforts can be invested into other modalities, like vocal cues or body poses. Finally, it is to collect a valid dataset to test the associating effects between different modalities. The current experiments used a set of simulated datasets that combined with various modalities, so the relationships between the modalities might not be genuine. Instead, an experiment can be conducted to collect data from a control group, which contains pairs of friend. The data is collected in two different occasions. In the first case, the subject will converse with their friend using a scripted dialog. For the second case, the subject will converse with a stranger using a scripted dialog.

The second future experiments will concentrate on improving and incorporating the PLBM into telepresence applications. The first task is to establish the relationships between the encoded gesture signals to decoded gesture. Currently, the CNN-RBM-Mean has 200 convoluted features. There is no documentation of these features and their effects on the decoder signals. Thus, some efforts can be invested to find a systematic approach to find the effect of these features. By knowing the effects of each feature, it should be much easier to find a searching algorithm to optimize the encoded data to suit the environment constraints. The current approach is to deploy an evolutionary algorithm to find the best individual to fit the environment constraints. That process needs huge amount of time that might not be feasible in a telepresence application; hence, a better searching algorithm will be desirable to improve this particular process.

Tables of appendices

Appendix	Name	For Section
A	Nonverbal communication	1.1.1
B	Local and distributed representation	1.5 & 2.2.4
C	Automated feature extraction	1.5 & 2.2.4
D	Existing telepresence robot interface	2.2.1
E	Expressive gestures on virtual agent	2.2.1
F	Characteristics and effects of an CATR interface	2.2.1
G	Video for the various experiments	4.1.1, 4.2.1, 4.3.1 & 4.4

Appendix A Nonverbal communication

Like an old saying, “It’s not what you say, but it’s how you say it”. If a sentence were delivered differently, it would produce a different effect or intends. In a salesperson scenario, he could either close or lose a deal. Thus, nonverbal communication is an important component for an effective social interaction, and it is not limited only to nonverbal behaviors. Based on Knapp [29] classification, it can be classified as communication environment, communicator physical characteristics, and communicator behaviors.

Figure A-1 Various interpersonal spaces categorized by Hall [53].

Communication environment looks into the physical environment and the spatial environment aspect in a communication process. Placement attachment is the emotion attachment between people to a specific or general environment. For instance, people, who love nature, might find comfort in the forested environment and anxiety in crowded shopping complex. Next, the presence or absence of a specific or general person can also affect a social interaction. For example, an introvert person might react more openly towards a mixture of friends and strangers. On the other hands, the same person might remain shut when there were only strangers. Lastly, proxemics behavior is a person usage of space in social interaction. One of these behaviors is the usage of space between two communicators, and these spaces can be categorized into intimate, personal, social, and public space [53]. Figure A-1 illustrates the relationship between the various spaces. These spaces are sensitive to personal and environmental factors; therefore, the physical space of each category will vary [53].

Next, communicator physical characteristics describe the physical aspect of a person. The primary feature is the physical characteristics, and they include height, body shape, facial features, and more. In the experiment to examine the relationship between facial width and trustworthiness, Stirrat [54] found that participants were more prompt to trust male with narrow than wide face. The second feature is the artifacts on a person, and they can range from spectacles, clothes, tattoo, and much more. Roach-Higgins [55] believed that dress, which includes body modifications and supplements, could express ranking within the social structure.

Figure A-2 The relationship between gesture intensives and perception of extraversion [11].

Lastly, the communicators' behaviors are the response from the people in an ongoing social interaction, and these responses express a degree of intimacy and immediacy towards the topic, object, or person. Firstly, one of these behaviors is the gesturing. Gesturing can demonstrate the personality of the exhibitor. For example, Neff [11] conducted an experiment to show the relationship between gesture intensives and the perceptions of extraversion in a virtual agent. Neff [11] showed that there was a significant effect of increasing gesture intensives to the increase the perceptions of extraversion (Figure A-2). Secondly, facial behaviors are also other well-known nonverbal features. In one of the earlier work, Ekman [56] believed that facial expression could deceive or represent the person emotional state. If a person was deceiving his emotional state, Ekman [57] also believed that the micro-expression could be the mechanism to detect possible lying cases. Lastly, there is also an important effect of touching in social interaction. Effect of touching can either be positive and negative. For instance, touch and gaze can be a compliance mechanism; on the contrary, self-touch can be exhibited as a nervous mechanism.

In conclusion, nonverbal communication covers a wide spectrum of features. All these features affect the outcome in an ongoing conversation. In addition, these features also give us additional information of conversation partner.

Appendix B Local and distributed representation

Up to this point, we have been openly discussing the advantages and disadvantage of predefined gestures and manually distributed gesture parameters. We have argued that distributed gestures parameters have a higher degree of expressivity over prototype gestures. Furthermore, manually distributed gesture parameters might not cover the whole spectrum of parameters. The following section will illustrate the difference between prototype gestures and distributed gestures parameters; follow by, the issue with manual distributed gestures parameters.

Figure B-1 A sample training dataset with objects of different shapes and colors.

Let begin with an example, it was to represent a list of objects, and there were three features engineers to design the new feature space. All of them were only given the eleven classes of objects in their samples training set (Figure B-1). The first feature engineer applied the local representation. The new features space was

$$\mathbf{h} = [\text{blue diamond}, \text{blue circle}, \dots, \text{orange square}], \quad (\text{B.1})$$

where $\mathbf{h} \in \{0,1\}^M$, $\sum_i^M h_i = 1$, and $M = 11$ was the total number of observable distinct classes. On the other hand, the second feature engineer used the distributed representation. The new features space was

$$\mathbf{h} = [h_{\text{shape}}, h_{\text{color}}], \quad (\text{B.2})$$

where $h_{\text{shape}} \in \{\text{diamond}, \dots, \text{square}\}$ and $h_{\text{color}} \in \{\text{blue}, \text{purple}, \text{orange}\}$. The last feature engineer also adopted the distributed representation; however, he had a

different intuition, which he replaced the h_{shape} with h_{corner} . His new features space was

$$\mathbf{h} = [h_{corner}, h_{color}], \quad (\text{B.3})$$

where $h_{corner} \in \{0,3,4,5\}$ and $h_{color} \in \{blue, purple, orange\}$. Next, they were given three testing data, and they were to transform them into their features space.

Figure B-2 Different test input with their corresponding output represented in local and distributed representation.

Figure B-2 shows the three test data and their corresponding transformation from each features space. The local features space has no problems representing the blue circle and orange square; however, there is no solution for the purple pentagon. Oppositely, both the distributed features spaces have represented all the test data into their respective spaces. However, there is an ambiguity between their correctness as they have a different perceptive for diamond, which can be diamond or 445 square.

In short, the example shows the fundamental concept and the issues for the two representations. The local representation is the equivalent of the prototype gestures or predefined gestures. The distributed representation is the same as distributed gesture parameters. Next, there will be an in-depth discussion about the advantages and disadvantages of local and distributed representation, and the problem with manual labeling.

Local representation has a few advantages over distributed representation. Firstly, it is easier to interpret by another human. Since the network “mirrors the structure of the knowledge” [58], other person can easily differentiate the different category within the local space. Next, it is easier to integrate with other system. An example can be a pick and place system that has a classification system follows by a

behavior generator (Figure B-3). The system can easily react to the activation output since the output is mutually exclusive to the rest of the output. For the second example, Figure B-4 shows a handwritten numbers recognition system. The left of the figure is the possible input, while the right of figure is the 100 hidden units of the first layer. It is nearly impossible to use the distributed representation information without a supervised classifier.

Figure B-3 Sorting pick and place scenario.

Figure B-4 Sample of the MNIST database (left), 100 hidden units in first layer of a neural network that was pre-trained using Deep Belief Network then fine-tuned using backpropagation with dropout (middle) [50], and a simple neural network configuration (right).

On the other hand, distributed representation also has a few advantages over local representation. Firstly, it is possible to generalize new input [59]. For instance, Figure B-2 shows that both the distributed features spaces can represent the unseen object, purple pentagon. Secondly, it encompasses more information and expressive [59]. Let assume there is a binary output vector $\mathbf{h} \in \{0,1\}^N$. If the vector is distributed represented, then it can categorize 2^N classes. On the other hand, the local representation can only store N classes. Finally, it groups objects with similar features closer to one another [60]. For instance, the third feature transformation in Figure B-2 shows that the objects with four corners, square and diamond, are grouped closer together. For MNIST example in Figure B-5, Maaten et al. [61]

demonstrated that the MNIST data could be grouped into their respective class using the T-SNE algorithm. Alternatively, the T-SNE algorithm created two dimensions or distributed features for the MNIST data so that the similar data were grouped closer together.

Figure B-5 Mnist dataset represented using t-sne by [61].

Next, this section discusses the issues with manual feature extraction. Firstly, there might be missing category for local representation. In the shape and color example, the first feature engineer did not take into consideration of the existence of the purple pentagon, and there might be a lot of other combination of shape and color that were not observed. With manual local features extraction, the feature engineering will usually design the features based on the application requirement and personal experiences. Secondly, there might have ambiguity in the selection of the parameters for the distributed representation. For example, the second and the third feature engineers disagreed on one of the features. Both h_{shape} and h_{corner} are both valid features, and their correctness might depend on the applications. However, this implied the lack of consistency in manual feature extraction. Lastly, different tasks for the same input might require a different set of features. For the current sorting example in Figure B-3, h_{shape} is better over h_{corner} since h_{corner} classifies diamond and square in the same category. However, if the bin was changed to categorization by the number of corners, then the h_{corner} might be superior over h_{shape} .

In conclusion, the distribution representation has its advantages and disadvantage over local representation and automated over manual features extraction. For a telepresence robot application, the distribution representation is preferred because it has a higher expressiveness. In addition, the automated feature extraction is preferred because it provides more consistent and complete features. However, there are problems like interpreting and integrating distributed information.

Appendix C Automated feature extraction

Automated feature extraction provides more complete and consistent distributed features. One of the automated feature extraction is the restricted Boltzmann machine (RBM) [31]. Over the past decade, many researchers have implemented and improved RBM to many applications. There are couples of the characteristics in RBM that are useful in the telepresence application; however, there are also characteristics that are not desired.

Figure C-1 A motion sequence (bottom) and its 100 hidden units (Top) [14].

Figure C-2 Training images (left) and reconstructed images using RBM (Right) [62].

This section only lists few of the characteristics that are useful in the telepresence application. For expressiveness, RBM is capable of extracting not only spatial distributed features but also the spatiotemporal distributed features. For spatiotemporal features, Taylor et al. proposed the conditional RBM (CRBM) [14] to model the walking dynamic with different styles. Figure C-1 shows 100 of the activated hidden units representing different styles. Not only encoding, it can also decode input from the distributed parameters. Susskind et al. [62] demonstrated this capability by reconstructing 60 random facial images from the training data (Figure C-2). Besides handling only one modality, it is also capable of associating different

modalities and recovering missing modalities. In the facial expression generative model (Figure C-3 left) proposed by Susskind et al. [62], it models three different modalities, and they are users identities, facial action coding system (FACS) [63], and the facial images. The experiments not only demonstrated the associative and extraction capability but also the ability to reconstruct missing facial images given the user identity and FACS parameters (Figure C-3 right). Next, RBM can implicitly remove noises in the input data. In the experiment conducted by Sutskever and Hinton [36], they explicitly added noises into the training data for training the model. The result was a sequence of images that represented not only the dynamic of the raw sequences but also a sequence with the noise reduced. Lastly, it can be shift-invariant with a convolutional network. Instead of training the whole image, Norouzi et al. [64] suggested to train the RBM on the sample patches of the training images to extract a set of local features. If a system was to detect human in an images, the patches could range from a whole human to a partial human. Once the local features were extracted using RBM, the system could detect the human in any part of the images and keep the spatial information. The process was to convolute the local features across the input images.

Figure C-3 The associative architecture between identity label, FACS labels, and facial image (left). The reconstructed facial images given the identity label and FACS labels (right) [62].

Unfortunately, RBM possesses few limitations that are not suitable for a telepresence operation. Firstly, RBM uses iterative approach to extract its features. In the telepresence application, any user can operate the interface without prior training the systems. However, associative capabilities in RBM required the new behaviors and information of the operator and audience prior to this conversation in order to generate the associative rules. Hence, other online associative approach needs to be considered. Secondly, there is a phase issue in spatiotemporal

distributed representation. For a periodic motion like walking, we would expect the output of the extracted features to be constant throughout the same motion. If the distributed features were walking speed and stride length, then the parameters for both the features would be constant until the motion changed. Unfortunately, Figure C-1 shows that features extracted by RBM do not have this behavior, and it is periodic oscillating as if it contains the phase information of the motion. This phase information can pose an issue to certain online associative approaches. Unlike local and distributed manual features extraction, the RBM features are difficult to interpret because RBM extracts them based on the distribution of the dataset. For instance, Figure C-5 (left) shows some RBM features learned from dataset of facial images. It is impossible to comprehend the relationship of each feature with the different expression. If it is the associative model in Figure C-3, then the relationships between FACS and the RBM features are clearer as shown in Figure C-5 (right).

Figure C-4 The top row is the raw image sequence. The second row is the input sequence, which is the raw images sequence corrupted with noise. The third and fourth rows are the output of TRBM-VV using single hidden layer and two hidden layers respectively [36].

Figure C-5 On the left is some RBM features of facial expression. On the right is the positive (1-3 rows) and negative (4-6 row) correlation of the features (row) with an action units (column).

In conclusion, RBM is a self-feature extractor that provides not only extracting features but also a lot other capabilities to enhance telepresence operation. However, this approach does have its shortfall, and additional modules would be required to create a better telepresence operator interface.

Appendix D Existing telepresence robot interfaces

D.1 Traditional telepresence robot with traditional interface

Unlike other mediated mediums, telepresence robot has a unique capability that is mobility. Mobility not only empowers the operator with the ability to explore a remote environment but also the power to show the operator's focus and attention. For the high quality communication between the operator and the partner, a good navigation interface is required to reduce the operator's cognitive load and ensure the safety of the communication partner at the remote-end.

Figure D-1 MAVEN [65] user interface for navigation control

The navigation interface can be divided into low and high-level behaviors controls. The low level behaviors control is a set of primitive movements that the operator can select, and the primitive movements can include forward, backward, clockwise, counter clockwise, strafe right, and strafe left. Many researchers [2], [66]–[70] had designed their navigation control unit using either a simple keyboard-mouse interface or joystick interface to control the primitive movement of the platform. On the other hand, the high-level behavior is a set of deliberate behaviors, and deliberate behaviors could be “point A to B navigation”, “rear people following”, and “side people following”. In a typical scenario, the operator wanted to travel to a destination. In order to reach his destination, the operator could select the “point A to point B navigation” behavior by entering the coordinate of the destination. Upon receiving the behavior and the goal, the robot generated a collision-free path leading to the goal.

D.2 MeBot with master-slave interface

A telepresence robot should also exhibit “expressive gestures, body pose, and proxemics” so that there is an “engaging and enjoyable interaction” [5]. MeBot was one of its kind when it was first designed, by Breazeal [5], to meet the above objectives. MeBot has two three DOF manipulators and a three DOF neck. They empower the operator with social cues like gesturing and gazing. In order to control these new modules, Breazeal [5] proposed and developed a few control strategies for the gesture and the gaze.

The two proposed strategies are passive model controller and natural interface for the gesture and the neck module respectively. The passive model controller is a master-slave model, which comprises of two identical systems locate at different environment. For replicating an arms gesture, a master system is at the operator-end, while the slave system is at the remote-end. The remote-system will replicate the gesture signals when the operator manipulates the master system to meet his intention. For natural interface, the operator can control a system through natural actions. For the neck module, a specialized facial system observes and updates the operator necks status, e.g. yaw, pitch, and roll. Therefore, the operator can naturally rotate his head, and the status will be sent to the neck module.

Figure D-2 MeBot [5] user interface

D.3 Hasegawa's bot with natural interface

Hasegawa's telepresence robot is an extension of the MeBot. With a higher DOF, Hasegawa's telepresence robot has two four DOF upper-limbs, two DOF waist, and three DOF neck. The proposed control strategy is natural user interface for both the arms and neck modules.

Figure D-3 Hasegawa's [6] user interface

D.4 Park's bot with recognition-synthesis interface

Park's user interface was not a telepresence operation; instead, it was demonstrated for a teleoperation situation. Both the operations have similar outcomes, except the telepresence operation is used in a social interaction while the teleoperation is for a general case. As a result, telepresence operation presents more nonverbal cue, e.g. facial expression. Nevertheless, Park's user interface could also be applied into a telepresence operation.

Park's user interface is a recognition-synthesis model [18]. In the operator-end, there is a sensor suite to capture the operator's verbal cues, e.g. motion capture devices. The system recognizes and classifies the captured signals into one of the predefined classification, which encapsulates operator's intentions. Upon receiving the classification, the system will decode and synthesize the motion based on its previous state and operator's intention.

Figure D-4 Park's [18] user interface

Appendix E Expressive gestures on virtual agent

Expressive gesturing has been one of the importance components in virtual agent. Besides rendering high-fidelity facial expression and facial features of a character in an animation, the motions generated by the individual character should also be unique to represent their personality. Hence, the following section illustrates some of the models that regenerate expressive gestures.

In some of the works [11], [16], their models can regenerate a variety of gesture with a set of manual defined parameters. For Hartmann's model [16], the parameters are overall activation, spatial extent, temporal extent, fluidity, power, and repetition. $[-1,1]$ are the ranges of value for each attribute. Figure E-1 shows the effect of spatial extent on a given input gesture. In the figure, the virtual agent was executing the same gesture, but the spatial extent was set at different values. From the figure, we can notice that the spatial coverage goes from smaller to larger area when the spatial extent values increases.

Figure E-1 The variance between different spatial parameter for an arms motion – extending and contracting towards the chest [16].

In other experiment, Neff [17] presented a system for recreating animation with realistic individualistic gesture given a script. Firstly, the system acquired the statistical model of the speaker's gesture style given a training video. With the model and the theme, rheme and focus, for the input text, their system created a stream of coherence gestures. The gestures script was passed to the animation model, where more detail would be embedded into the information. Finally, the system generated physical simulated motion based on the enhanced gestures script. Figure E-2 shows one of their results where two individuals, top and bottom,

gesturing on the same script. In his experiment, two individual agents expressed different gestures even though the same words were spoken.

Figure E-2 Reanimation for two different styles given the same script [17].

Figure E-3 The four input (blue) and the 15 output (black or green) for a cheering motion [19].

In other studies, Lau et al. [19] proposed and presented a generative model to regenerate variants of motion given an input motion. In this experiment, a probabilistic model, dynamic Bayesian network, was trained to model a segment of a sequence input. Once the probability distribution was learnt, the systems could sample using the learnt distribution. Figure E-3 shows the output of the Lau's model synthesizing an input motion. The first column shows the temporal variant between the input and output motion, while the second and third columns demonstrate the

spatial variant between both motions. From this figure, we can see that the output motions are similar to the input motions.

In the next works, Taylor et al. [37] proposed two generative models: conditional restricted Boltzmann machine (CRBM) and factored CRBM (FCRBM). Firstly, they demonstrated the capabilities of the CRBM by reconstructing different style of walking (Figure E-4). Unlike the previous works from Lau et al. [19], the CRBM were trained with different styles, and it was capable of generating various walking styles given the initial condition. However, CRBM cannot reconstruct motion given a context, so Taylor et al. also proposed FCRBM to address that issue. In a three-way FCRBM, there is an additional input, known as the contextual input. The contextual input is either the ten distinct styles or the two distributed parameters. The ten distinct styles are label like cat, dinosaur, or sexy, and they are mutually exclusive to each other. On the hand, the two distributed styles are walking speed and stride length. Each label has three discrete states, which produces nine combinations. Besides the nine combinations, the model also produced realistic motions between the interpolation and extrapolation of the speed and stride.

Figure E-4 Reconstructing walking motion with five different styles using CRBM [37].

In the last work, Xia et al. [15] developed a system to recreate motion into different styles given an input motion. They proposed the three use cases for this model, and they were input with known action and style, input with unknown labels, and transfer of style given an unlabeled input. In this section, we briefly illustrate on the last use case. Firstly, they annotated the training data with behavior and style attributes. The behaviors attributes include walking, running, punching, kicking, jumping, and their transition between them. The style attributes include depressed, childlike, angry, neutral, proud, old, sexy, and strutting. Given an

unlabeled input, the system uses K nearest neighbor to find the closest training data in its database for each class. With the closest training data at each class, the model can estimate the parameter for the autoregressive model for each class. Using the minimum average distance from each class, the proportional weights for each parameter can also be computed. If the goal is to change the style, the weights for each class can be adjusted to the desired style. Figure E-5 illustrates the systems transforming a set of neutral motions into a set of proud motions.

Figure E-5 Reanimating the same motion for two different styles: neutral (input top) and proud (output bottom) [15].

In short, this section has represented a few works that recreates or synthesizes variation of gestures given the contextual information. In the above works, the contextual information can be a set of predefined distributed parameters, output styles, or a set of input gestures.

Appendix F Characteristics and effects of an CATR interface

Table F-1 Characteristics and its types with descriptions and examples (bold* are the preferred types)

Characteristic	Type	Description	Example
Input interface (Figure 2-5)	Master-slave	Operator manipulates a primary system that is synchronized with a secondary system in the remote-environment.	For MeBot [5], the operator had a pair of three DOF manipulators. He could express his desired gestures on his manipulators, and the remote system would imitate the motion.
	Natural interface*	Operator can express his nonverbal cues by acting them naturally. The system will acquire the information and send them to the remote system. The remote system then executes the motion given the information.	For Hasegawa [6], the operator's poses information was acquired by a KINECT sensor, and information was sent to the remote-system. The remote system would imitate the operator by moving its joints based on the posed information.
	Traditional input devices	Operator needs to use traditional input devices to operate the various nonverbal cues.	In a traditional telepresence robot, the operator needed to select the intended gestures from the keyboard. For instance, each key or a combination of keys could represent a distinct gesture.
Representation	Auto* or Manual	Auto represents a set of features that is extracted by an unsupervised algorithm.	For Lau's model [19], a probabilistic model, dynamic Bayesian network, was trained to model a segment of an input sequence. The parameters or probability distributions were the features for this model.
		Manual represents a set of features that is extracted by a human.	In Park's work [18], 13 distinct gestures were manually extracted, and they included lift-right, lift-left, lift-both, and many more.
	Distributed* or local	Distributed are a set of features that represents the underlying structure of the information.	For Hartmann's model [16], Hartmann identified six different distributed parameters for each prototypical gesture. The parameters included spatial extent, temporal extent, fluidity, and more. The six different parameters redefined the prototypical gestures, and they are not mutually exclusive to one another.
		Local are a set of features that are mutually exclusive to one another.	For Park's work [18], the 13 distinct gestures were mutually exclusive from one another. Lift-right and lift-left could not occur at the same time. If the operator lifted both his arms, then lift-both would be selected.

Continue from previous table

Characteristic	Type	Description	Example
Representation (continue)	Spatial or Spatiotemporal*	Spatial features do not capture any spatiotemporal information	For MeBot [5], the posed or joint information was directly transmitted to the remote system.
		Spatiotemporal features encompass the temporal information.	For Park's model [18], the operator's motion was encoded into an output signals, and the output signal represents a particular motion, like lift-right.
Encoding	Auto*	An automated system encodes the intended information before the information is transmitted.	For Lau's model [19], the probabilistic model automatically generated a distributed output given a segment of a sequential input.
	Direct	The intended information is directly transmitted to the remote system without any transformation.	For MeBot [5] and Hasegawa [6], both the systems directly transmitted the joints information to the remote system without transformation.
	Manual	The intended information needs to be manually encoded by the operator.	For Hartmann's model [16], the operator must select the value for the six distributed parameters and a prototypical gesture for every intended gesture.
Decoding	Auto*	A remote automated system decodes the encoded information before the signals can be executed by the remote system.	For Park's model [18], the encoded data was decoded by a motion generator. In this case, the motion generator had a set of predefined motions, and it would execute one of the motions depend on the encoded data.
	Direct	The received information can be directly executed by the remote system without any transformation.	For MeBot [5] and Hasegawa [6], both remote systems executed their motion based on the received joints information.
Associating	Auto*	For multimodal situation, the system automatically maps relationship between different modalities.	<i>Not available in the existing interfaces</i>
	Manual	For multimodal situation, the operator will map or indicate the relationships between different modalities.	For Neff's model [17], a human coder annotated and aligned the gestures behavior and speech structure to create the rules for the system database.
	None	The system deals with each modality independently.	For instance, MeBot [5] has multimodal like neck, arms, video, and audio. However, each modality worked independently from one another.

Table F-2 Effects and its types with descriptions and examples (bold* are the preferred types)

Effect	Type	Description	Example
Expressivity	High*	The high degree of variance within a modality	For Hartmann's model [16], the operator had six distributed parameters to create a personal prototypical gesture for every prototypical gesture.
	Low	The low degree of variance within a modality	For Park's model [18] , it had 13 gestures only. It was not appropriate to be an operator interface for telepresence robot because the remote system should express personal gestures and not generic gestures.
Cognitive load	High	The system requires high amount of cognitive load to execute the nonverbal cues.	For instance, Hartmann's model [16] required the operator to select a prototypical gesture and fill in the intensity, [-1, 1], for the six distributed parameters in order to execute a gesture.
	Low*	The system requires low amount of cognitive load to execute the nonverbal cues.	In Lau's model [19], the operator just needed to express his intended gestures, and the system would automatically handle the rest of the processes.
	Low (require remapping)	The system requires low amount of cognitive load, but the system still consume some degree of operator's cognitive load for remapping.	For MeBot [5], the operator must remap his intended gestures onto a primary system before he could execute his action. During the remapping process, unconscious gestures might be filtered away.
Decoupling	No	The system will continue to send the nonverbal information even when the nonverbal information might be undesirable	For Hasegawa's bot [6], the remote system would execute any gestures the operator's gestures. Undesirable gestures, e.g. typing on keyboard, might be not congruent to the going conversation.
	Yes (idle)	The system can stop any undesirable nonverbal information from transmitting, but the system will be in an idling state	For MeBot [5], the operator could stop his gesturing by stop controlling the primary system. At this point, the operator could continue with his remote-task, but the remote system would be in an idling state.
	Yes* (associate)	The system can stop any undesirable nonverbal signal from transmitting. The system can also reproduce coherence nonverbal signal to conceal the missing signal.	For Neff [17], the system could automatically generate the intended the gestures that was congruent with the linguistic structure of the speech.

Continue from previous table

Effect	Type	Description	Example
Obstacle avoidance	Deliberative*	A deliberative planner generates motion and keeps its overall intentions. It tends to be slower when compares with reactive planner.	For Lau's model [19], the output could be a variance of the input, which preserved the overall shape of the input.
	Reactive*	A reactive planner is a reflexive behavior system that responds to the real-time external stimulus.	The MeBot [5] had a set of range sensors for obstacle avoidance. Since MeBot adopted the directly transmitted the joint information, it could only spontaneously avoid the obstacle when the obstacle was detected by the range sensor.
Speech collision prevention	No	Missing of unconscious gestures that can reduce the events where "two or more participants to start speaking at the same time"[6].	For MeBot [5], the operator consciously remapped his intended motion onto the primary system, and this remapping of gestures increased the average counts of speech collisions [6].
	Yes*	The system acquires the unconscious gestures that can reduce the speech collision.	Hasegawa's operator system [6] captured the natural gestures of the operator through a natural interface, and more unconscious gestures were projected because direct natural interface had lesser remapping.
	Yes* (assumed)	If a system with encoding is using natural interface and has high expressiveness, it is assumed that the system can capture unconscious gestures.	Lau's model [19] acquired the distributed output via a natural interface. Since it could represent any gestures given the input, it could also represent any unconscious gestures.

Appendix G Supporting videos for the various experiments

Title:	Encoding and decoding in PLBM
Description:	In this video, it illustrates effects of encoding, which clusters similar motion closer to one another. In addition, it shows that the generated motion is similar to the original input motion.
Video URL:	https://youtu.be/yUII3-31sc0

Title:	Generating congruent and expressive gesture using PLBM during decouple
Description:	In this video, it shows the process of the PLBM using its associating module. During learning, the PLBM learns and creates rules across all the modalities, e.g. facial features and gesture. When decoupling situation, the PLBM can insert congruent gestures given observable modalities.
Video URL:	https://youtu.be/VRKo_r303h0

Title:	Generating smooth and expressive gesture using PLBM during intermittent connection
Description:	In this video, it illustrates the disadvantages of using spatial information during teleoperation transmission; furthermore, it demonstrates the advantage of using spatiotemporal distributed information during teleoperation.
Video URL:	https://youtu.be/qB3Xx80oEXs

Title:	Collision avoidance using PLBM with EA
Description:	In this video, it shows the result of the PLBM re-tuning the motion to avoid collision. In addition, it shows the procedure to finds the best solution in a collision situation using evolutionary algorithm.
Video URL:	https://youtu.be/HLtWGTYhRUA

Reference

- [1] W. A. Bainbridge, J. Hart, E. S. Kim, and B. Scassellati, "The effect of presence on human-robot interaction," in *IEEE International Symposium on Robot and Human Interactive Communication*, 2008, pp. 701–706.
- [2] K. M. Tsui, M. Desai, and H. A. Yanco, "Exploring Use Cases for Telepresence Robots," in *ACM/IEEE International Conference on Human-Robot Interaction*, 2011, pp. 11–18.
- [3] E. Paulos, "Personal Tele-Embodiment," University of California, 2002.
- [4] "RP-Vita." [Online]. Available: <http://www.irobot.com/For-Business/RP-VITA.aspx>. [Accessed: 14-Sep-2016].
- [5] S. O. Adalgeirsson and C. Breazeal, "MeBot : A robotic platform for socially embodied telepresence," in *ACM/IEEE International Conference on Human-Robot Interaction*, 2010, pp. 15–22.
- [6] K. Hasegawa and Y. Nakauchi, "Preliminary Evaluation of a Telepresence Robot Conveying Pre-motions for Avoiding Speech Collisions," in *HAI-Conference.net*, 2013, pp. 5–8.
- [7] E. Oztop, D. W. Franklin, and T. Chaminade, "Human-Humanoid Interaction: Is a Humanoid Robot Perceived as a Human?," *Int. J. Humanoid Robot.*, vol. 2, no. 4, pp. 537–559, Dec. 2005.
- [8] J. Kilner, Y. Paulignan, and S. Blakemore, "An interference effect of observed biological movement on action," *Curr. Biol.*, vol. 13, no. 6, pp. 522–525, Mar. 2003.
- [9] W. C. Pang, C. Y. Wong, and G. Seet, "Design and development of edgar - a telepresence humanoid for robot-mediated communication and social applications," in *Proceeding of the IEEE International Conference on Control and Robotics Engineering*, 2016, pp. 1–4.
- [10] "Robot Thespian." [Online]. Available: <https://www.engineeredarts.co.uk/robotthespian/>. [Accessed: 06-Aug-2016].
- [11] M. Neff, Y. Wang, R. Abbott, and M. Walker, "Evaluating the Effect of Gesture and Language on Personality Perception in Conversational Agents," in *International Conference on Intelligent Virtual Agent*, 2010, pp. 222–235.
- [12] P. Ekman, "Emotional And Conversational Nonverbal Signals," in *International Colloquium on Cognitive Science*, 2004, pp. 39–50.
- [13] J. Bavelas, N. Chovil, and L. Coates, "Gestures specialized for dialogue," *Personal. Soc. Psychol.*, vol. 21, no. 4, pp. 394–405, Apr. 1995.
- [14] G. W. Taylor, G. E. Hinton, and S. T. Roweis, "Modeling human motion using binary latent variables," in *Advances in Neural Information Processing Systems*, 2006, p. 1345.

- [15] S. Xia and C. Wang, “Realtime Style Transfer for Unlabeled Heterogeneous Human Motion,” *ACM Trans. Graph.*, vol. 34, no. 4, p. 119, Jul. 2015.
- [16] B. Hartmann, M. Mancini, and C. Pelachaud, “Implementing expressive gesture synthesis for embodied conversational agents,” in *Gesture in human-Computer Interaction and Simulation*, 2006, pp. 188–199.
- [17] M. Neff, M. Kipp, I. Albrecht, and H. Seidel, “Gesture modeling and animation based on a probabilistic re-creation of speaker style,” *ACM Trans. Graph.*, vol. 27, no. 1, Mar. 2008.
- [18] H. Park, E. Kim, S. Jang, and S. Park, “HMM-based gesture recognition for robot control,” in *Iberian Conference on Pattern Recognition and Image Analysis*, 2005, pp. 607–614.
- [19] M. Lau, Z. Bar-Joseph, and J. Kuffner, “Modeling spatial and spatiotemporal variation in motion data,” *ACM Trans. Graph.*, vol. 28, no. 5, p. 171, Dec. 2009.
- [20] K. L. Nowak and F. Biocca, “The effect of the agency and anthropomorphism on users’ sense of telepresence, copresence, and social presence in virtual environments,” *Presence Teleoperators Virtual Environ.*, vol. 12, no. 5, pp. 481–494, Oct. 2003.
- [21] F. Biocca, C. Harms, and J. Gregg, “The networked minds measure of social presence: Pilot test of the factor structure and concurrent validity,” in *International Workshop on Presence*, 2001.
- [22] M. Minsky, “Telepresence,” *OMNI Mag.*, pp. 45–51, Jun. 1980.
- [23] G. Pettey, C. C. Bracken, B. Rubenking, M. Buncher, and E. Gress, “Telepresence, soundscapes and technological expectation: putting the observer into the equation,” *Virtual Real.*, vol. 14, no. 1, pp. 15–25, Jan. 2010.
- [24] J. Short, E. Williams, and B. Christie, *The social psychology of telecommunications*. 1976.
- [25] K. M. Tsui, M. Desai, and H. A. Yanco, “Towards Measuring the Quality of Interaction: Communication through Telepresence Robots,” in *Performance Metrics for Intelligent Systems Workshop*, 2012, pp. 101–108.
- [26] M. Garau, M. Slater, V. Vinayagamoorthy, A. Brogni, A. Steed, and M. Sasse, Angela, “The Impact of Avatar Realism and Eye Gaze Control on Perceived Quality of Communication in a Shared Immersive Virtual Environment,” in *SIGCHI Conference on Human Factors in Computing Systems*, 2003, pp. 529–536.
- [27] M. Argyle, V. Salter, H. Nicholson, M. Williams, and P. Burgess, “The Communication of Inferior and Superior Attitudes by Verbal and Non-verbal Signals*,” *Br. J. Soc. Clin. Psychol.*, vol. 9, no. 3, pp. 222–231, Sep. 1970.

- [28] T. L. Chartrand and J. A. Bargh, "The chameleon effect: The perception-behavior link and social interaction," *J. Pers. Soc. Psychol.*, vol. 76, no. 6, pp. 893–910, Jun. 1999.
- [29] M. L. Knapp and J. A. Hall, *Nonverbal Communication in Human Interaction*. Holt, Rinehart & Winston New York, 2002.
- [30] K. Danziger, *Interpersonal communication*. Pergamon Press, 1976.
- [31] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A Fast Learning Algorithm for Deep Belief Nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, Jul. 2006.
- [32] Y. Bengio, "Learning Deep Architectures for AI," *Found. Trends® Mach. Learn.*, vol. 2, no. 1, pp. 1–127, Nov. 2009.
- [33] Y. LeCun and Y. Bengio, "Convolution Networks for Images, Speech, and Time-Series," in *The handbook of brain theory and neural networks*, 1995, pp. 255–258.
- [34] S. Fothergill, H. Mentis, P. Kohli, and S. Nowozin, "Instructing people for training gestural interactive systems," in *SIGCHI Conference on Human Factors in Computing Systems*, 2012, p. 1737.
- [35] L.-P. Morency and A. Sadeghipour, "3D Iconic Gesture Dataset," 2011. [Online]. Available: <http://projects.ict.usc.edu/3dig/>. [Accessed: 28-Jan-2016].
- [36] I. Sutskever and G. E. Hinton, "Learning multilevel distributed representations for high-dimensional sequences," in *International Conference on Artificial Intelligence and Statistics*, 2007, pp. 548–555.
- [37] G. W. Taylor, G. E. Hinton, and S. T. Roweis, "Two Distributed-State Models For Generating High-Dimensional Time Series," *Journal Mach. Learn. Res.*, vol. 12, pp. 1025–1068, Mar. 2011.
- [38] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.
- [39] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual Prediction with LSTM," *Neural Comput.*, vol. 12, no. 10, pp. 2451–2471, Oct. 2000.
- [40] H. Jaeger, "A tutorial on training recurrent neural networks, covering bppt, rtrl, ekf and the 'echo state network' approach.," GMD-Forschungszentrum Informationstechnik., 2002.
- [41] A. Graves, "Generating sequences with recurrent neural networks," *arXiv*, pp. 1–43, 2013.
- [42] G. A. Carpenter and S. Grossberg, "ART 2: self-organization of stable category recognition codes for analog input patterns.," *Appl. Opt.*, vol. 26, no. 23, pp. 4919–30, Dec. 1987.

- [43] T. Frank, K. F. Kraiss, and T. Kuhlen, “Comparative analysis of fuzzy ART and ART-2A network clustering performance,” *IEEE Trans. Neural Networks*, vol. 9, no. 3, pp. 544–59, Jan. 1998.
- [44] A.-H. Tan and H. S. Soon, “Concept hierarchy memory model: a neural architecture for conceptual knowledge representation, learning, and commonsense reasoning,” *Int. J. Neural Syst.*, vol. 7, no. 3, pp. 305–19, Jul. 1996.
- [45] A.-H. Tan, “FALCON: a fusion architecture for learning, cognition, and navigation,” in *IEEE International Joint Conference on Neural Networks*, 2004, pp. 3297–3302.
- [46] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, I. Matthews, and F. A. A. Ave, “The Extended Cohn-Kanade Dataset (CK+): A complete dataset for action unit and emotion-specified expression,” in *IEEE Conference on Computer Vision and Pattern Recognition-Workshops*, 2010, pp. 94–101.
- [47] V. Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” in *International Conference on Machine Learning*, 2010, pp. 807–814.
- [48] G. E. Dahl, T. N. Sainath, and G. E. Hinton, “Improving deep neural networks for LVCSR using rectified linear units and dropout,” in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 8609–8613.
- [49] H. Lee, C. Ekanadham, and A. Y. Ng, “Sparse deep belief net model for visual area V2,” in *Advances in Neural Information Processing Systems*, 2008, pp. 873–880.
- [50] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, “Improving neural networks by preventing co-adaptation of feature detectors,” *arXiv*, pp. 1–18, 2012.
- [51] G. E. Hinton, “A practical guide to training restricted Boltzmann machines,” *Momentum*, vol. 9, no. 1, Aug. 2010.
- [52] R. O. Duda, P. E. Hart, and D. G. Stork, “Evolutionary Methods,” in *Pattern Classification*, Second Edi., John Wiley & Sons, 2012, p. 373.
- [53] E. T. Hall, *The hidden dimension*. Doubleday & Co, 1966.
- [54] M. Stirrat and D. I. Perrett, “Valid facial cues to cooperation and trust: male facial width and trustworthiness,” *Psychol. Sci.*, vol. 21, no. 3, pp. 349–54, Mar. 2010.
- [55] M. E. Roach-Higgins and J. B. Eicher, “Dress and Identity,” *Cloth. Text. Res. J.*, vol. 10, no. 4, pp. 1–8, Jun. 1992.
- [56] P. Ekman, “Facial Expression and Emotion,” *Am. Psychol.*, vol. 48, no. 4, pp. 384–392, Apr. 1993.

- [57] P. Ekman, "Darwin, Deception, and Facial Expression," *Ann. N. Y. Acad. Sci.*, vol. 1000, no. 1, pp. 205–221, Dec. 2003.
- [58] G. E. Hinton, "Distributed representations," 1984.
- [59] G. E. Hinton, "Connectionist learning procedures," *Artif. Intell.*, vol. 40, no. 1, pp. 185–234, Sep. 1989.
- [60] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A Neural Probabilistic Language Model," *J. Mach. Learn. Res.*, vol. 3, pp. 1137–1155, Feb. 2003.
- [61] L. J. P. Van Der Maaten and G. E. Hinton, "Visualizing high-dimensional data using t-sne," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.
- [62] J. M. Susskind, G. E. Hinton, J. R. Movellan, and A. K. Anderson, *Generating Facial Expressions with Deep Belief Nets*. INTECH Open Access Publisher, 2008.
- [63] P. Ekman and W. V. Friesen, "Measuring facial movement," *J. Nonverbal Behav.*, vol. 1, no. 1, pp. 56–75, Jul. 1976.
- [64] M. Norouzi, M. Ranjbar, and G. Mori, "Stacks of Convolutional Restricted Boltzmann Machines for Shift-Invariant Feature Learning," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 2735–2742.
- [65] G. Seet, W. C. Pang, C. I-ming, V. Iastrebov, W. Gu, and C. Y. Wong, "A Design for a Mobile Robotic Avatar - Modular Framework," in *3DTV-Conference: The True Vision-Capture, Transmission and Display of 3D Video*, 2012, pp. 1–4.
- [66] L. M. Ellison, M. Nguyen, M. D. Fabrizio, A. Soh, S. Permpongkosol, and L. R. Kavoussi, "Postoperative robotic telereounding: a multicenter randomized assessment of patient outcomes and satisfaction," *Arch. Surg.*, vol. 142, no. 12, pp. 1177–1181, Dec. 2007.
- [67] D. I. Fels, J. K. Waalen, S. Zhai, and P. Tamar, "Telepresence under exceptional circumstances: enriching the connection to school for sick children," in *Interact*, 2001, pp. 617–624.
- [68] A. Kristoffersson, S. Coradeschi, and A. Loutfi, "Towards Evaluation of Social Robotic Telepresence based on Measures of Social and Spatial Presence," in *Workshop on Social Robotic Telepresence*, 2011.
- [69] O. Kwon, S. Koo, Y. Kim, and D. Kwon, "Telepresence robot system for English tutoring," in *IEEE Workshop on Advanced Robotics and its Social Impacts*, 2010, pp. 152–155.
- [70] E. Paulos and J. Canny, "Designing personal tele-embodiment," in *IEEE International Conference on Robotics and Automation*, 1998, pp. 3173–3178.