A Novel Approach to Lifelong Learning for Robotic Response to Gesture

PAUL M. YANIK
Western Carolina University
Dept. of Engineering and Technology
Cullowhee, NC
USA
pyanik@wcu.edu

ANTHONY L. THREATT[2], JESSICA MERINO[1]
JOE MANGANELLI[2], JOHNELL O. BROOKS[3]
KEITH E. GREEN[2], IAN D. WALKER[1]
Clemson University
[1]Dept. of Electrical and Computer Engineering
[2]School of Architecture
[3]Dept. of Automotive Engineering
Clemson, SC
USA
{anthont, jmerino, jmanganelli, . . .
jobrook, kegreen, iwalker}@clemson.edu

Abstract: For unskilled or impaired users, intuitive interfaces are seen as key to the wide adoption of assistive robotic agents. Interfaces based on gesture offer a natural communication modality for Human-Robot Interaction which facilitates ease of use for the target population. However, as the abilities of the user change with time, their capacity to properly perform a given command choreography may degrade. Therefore, it is critical to the long-term success of the interface that it learn the behavioral intentions of the user and adapt accordingly throughout the lifespan of the agent. This paper describes a clustering approach to an agent's *lifelong learning* based on the Growing Neural Gas algorithm. A simulated user issues reinforcement feedback so as to progressively refine an agent's actions toward a goal configuration. Several network distance metrics for neighborhood learning within connected clusters are compared for their relative burden to the user-trainer in terms of speed of convergence. It is shown that the proposed method learns new gestures while retaining past learning with few training iterations.

Key–Words: Human-robot interaction, Growing Neural Gas, Lifelong learning, Q-learning

1 Introduction

As individuals age, a motivation to remain independent in the face of diminished mobility and health will increasingly draw upon assistive technologies to support essential Activities of Daily Living (ADLs) [1]. For those wishing to *age in place*, a reduced capacity to conduct ADLs may be associated with decreased quality of life and independence, higher caregiver burden, or even institutionalization [2]. The work described in this paper is motivated by a dearth of technologies that might provide adequate support of these important functions.

With this in mind, the authors envision an intelligent class of assistive devices and services which are highly integrated into the built environment. These technologies would allow the user's surroundings to act as an adaptive partner to facilitate ADLs for individuals whose ability levels are changing so as to improve quality of life and prolong independence [3]. To this end, the Assistive Robotic Table (ART) project begun at Clemson University (Figure 1) serves as a platform for research in this area and is the application focus of the work presented in this paper. ART is a new over-the-bed table appliance which seeks to innovate in areas of the user experience with assistive technology interactions [4].

Models of nonverbal communication offer the promise of intuitive and adaptive interaction [5] with intelligent assistive agents. A command interface based on gesture presents one option for nonverbal communication which would be easily adopted by this group of users. Typically, however, Human-Robot Interaction (HRI) in any form depends on interaction modalities which are predefined by the system designers. Commands based on gesture, voice commands or hand positions must accurately emulate one of a fixed repertoire of known possibilities. Thus, the response by the robotic agent is a one-to-one mapping of user action to a predefined response. This paradigm limits the usability and adaptability of the interface and potentially reduces the rate of its adoption among users.

Further, as dexterity and mobility of the user change with time, their ability to properly perform a given command choreography may diminish. There-

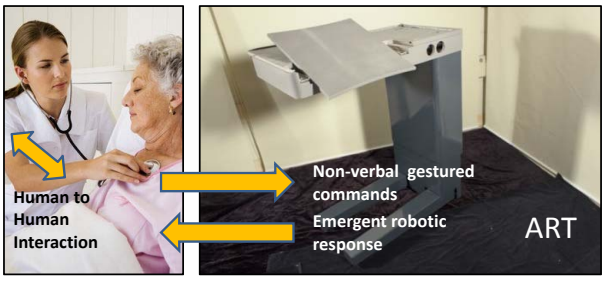


Figure 1: The non-verbal communication loop of the *Assistive Robotic Table* (ART). The focus of this work is on the emergent (learned) response of this device to the user.

fore, it is critical to the long-term success of the interface design that it learn the behavioral intentions of the user and adapt accordingly. This circumstance illustrates the need for a system which engages in *lifelong learning* [6] and is capable of discerning new gestures and interpreting the user's desired response to them. The research described in this paper targets the ART appliance and presents an approach which learns a user's preferred three-dimensional configuration of the appliance for tasks performed in a healthcare or home setting. The method uses arm-scale gesture motions collected from human participants. The intent of the system described in this paper is to engage in lifelong gesture learning by both (1) adapting to a user's changing abilities to perform gestures over time, and (2) learning new gestures without loss of past training. Human-robot interactions are conducted with a simulated human user/trainer who controls the application of a binary feedback (reward) signal to train the agent.

Extending past work by the authors [7, 1], a system based on the Growing Neural Gas (GNG) algorithm [8] and Reinforcement Learning (Q-Learning) is used in this paper to create an active mapping between performed gestures and robotic actuations. The proposed method takes advantage of the user's broad view of the problem space to selectively apply positive rewards where robot actions are tending toward the user's preferred goal configuration for a given gesture. The systems is shown to be robust in the presence of user performance degradation.

Contributions of this paper are twofold. First, a novel training/use model is presented which allows the learning system to be trained with the user as teacher. The model aims at minimizing the number of gesture samples which must be performed so as to reduce physical burden to the user. The success of the proposed approach is measured in terms of its speed of convergence to the user's preferred response through decreasing numbers of cycles of observation and reward. Second, an extension to GNG is introduced which allows for the insertion and deletion of nodes so as to simultaneously facilitate both the learning of new gestures and the retention of past learning (lifelong learning). These contributions are seen as key to the envisioned paradigm of assistive devices which support easy adoption by the user and long-term use in the face of their changing abilities.

This paper is structured as follows. Section 2 discusses the gesture recognition problem along with past research efforts in lifelong learning and describes their respective innovations and shortcomings. Section 3 discusses the new approach in this paper including data representations, algorithms and the simulation environment. Section 4 discusses the data collection fixture, and experimentation scenarios. Section 5 presents and interprets the experimental results. Finally, concluding remarks are given in Section 6.

2 Related Work

Because human gesture represents a natural and ubiquitous form of communication, its potential use as a mode of Human-Robot Interaction (HRI) is an area of ongoing research. This section gives an overview of the component problems frequently associated with gesture recognition, lifelong learning, and the approaches which are utilized for the work described in this paper.

2.1 Gesture Recognition

Human gesture may occur in various forms including pantomime, sign language (static and dynamic), full body poses or hand and arm gesticulation. Gestures involving hand and arm gesticulation comprise some 90% of gestured communication [9]. Thus, exploration of gesture at this scale is relevant to the larger HRI problem. The problem space of gesture recognition may be decomposed into the subproblems of sensing, data representation, pattern recognition (gesture classification), and machine learning (sensorimotor mapping). Much past work has been devoted to solving these problems individually and to that of gesture recognition in general. Aspects of this body of work which are applicable to the results presented in this paper are discussed in this section.

2.1.1 Sensing

Typical gesture sensing strategies involve wearable devices such as data gloves or body suits, or conventional cameras. Wearable sensors have been used to capture hand position/shape [10], arm orientation

[11], or gross motion using accelerometers [12, 13]. Although they offer accurate measurements of physiological geometry and dynamics, wearable devices present an inherent encumbrance to the user and thus may be difficult for the user to adopt with consistency [9].

Much of the work in gesture recognition is performed using video images/sequences due to their cost effectiveness and the richness of the data they provide [14]. Cameras, however, may present an intrusion of privacy in intimate settings such as bedroom or bath [15]. They also tend to suffer degraded performance in darkness or due to occlusion. Infrared cameras such as may be found with RGB-D sensors preserve user anonymity, operate in darkness, and offer the richness of data afforded by conventional cameras. For these reasons, the Microsoft Kinect RGB-D sensor (www.xbox.com/en-US/kinect) was used for this research.

2.1.2 Data Representation

From the data stream generated by a selected sensor, a compact representation (*feature vector*) of each gesture must be computed. Such representations may preserve information regarding the physiology and kinematics of the actor, or they may simply extract differential data from pixel intensities in a captured image or image sequence [16].

Visual templating of motion has been performed using Motion History Images (MHI) [17, 18]. Periodicity of motion has been shown to be a strong descriptor of motion, even allowing identification of the actor [19]. Repetitions in motion have been captured by computing the self-similarity of image sequences [16, 20]. Histograms of Gradients (HOGs) are used to create descriptors of image sequences based on the changing regional pixel intensities for single image frames. HOGs of Self-Similarity Matrices (SSMs) have been combined to produce view-invariant representations of non-periodic motions [3, 21] which facilitate robust classification. These representations, however, tend to be of high dimension and thus burdensome in computation and storage.

In this paper, a representation using Dynamic Instants (DI) proposed by Rao et al. [22] is extended to three dimensions by use of the RGB-D sensor. DIs are the extrema of acceleration in the actor's path of motion and may be used to form a visual characterization of that motion [19]. These have also been shown to be view-invariant, appearing regardless of the vantage point of the observer. A feature vector composed of dynamic instants (see Figure 5) has been shown by the authors [7] to possess useful discriminative properties for classification despite being small in dimension.

2.1.3 Gesture Classification

Classification of feature vectors typically involves sorting them into one of a gallery of types based on their proximity (by some distance metric) to clusters of similar, often labeled, vectors. Numerous classification methods have been applied to gesture recognition including Hidden Markov Models (HMM) [23, 24] , finite state machines (FSM) [25], C-means [26, 27], neural techniques such as Time Delay Neural Networks (TDNN) [28], networks based on Adaptive Resonance Theory (ART) [29], Growing Neural Gas (GNG) [30, 7] and probabalistic methods such as Most Probable Longest Common Subsequence (MPLCS) [31].

The GNG algorithm proposed by Fritzke [8] is a variant of Kohonen's self-organizing feature map (SOFM) [32]. GNG is capable of tracking a moving distribution [33], adding new reference nodes, and operating from static input parameters. Thus, it is well-suited to the task of gesture recognition and gesture learning where no labeled data is available. Indeed, since the acquisition of gesture data may be expensive in terms of the effort and time required of both the user and the researcher, such a technique that learns online is particularly desirable.

Two goals for the system described in this paper are (1) to adapt to a user's changing abilities to perform gestures over time, and (2) to be capable of learning new gestures without loss of past training. In normal operation, the receptive fields of reference nodes in the GNG cloud inherently allow for and generalize across variations in input. Mobility of reference nodes in the cloud enables persistent variations to become part of the system's learning base without focused training. These features are seen to accommodate an individual's changing ability to perform gestures over time. A GNG cloud may also grow in number of reference nodes and alter its connection topology over time to accommodate coarse variation in its input space. This feature suggests that GNG will be effective in learning new gestures as they are observed. And, as described in Section 3.7, modification to GNG is proposed which targets the strategic insertion of new reference nodes to support learning of new gestures. These features make GNG readily applicable to the two goals mentioned above and thus, it is the clustering method adopted in this paper.

2.1.4 Sensorimotor Mapping

Pattern recognition as previously described is frequently discussed under the heading of machine learning. Use of the term in this paper however, shall refer to a means of applying feedback so as to progressively refine a robotic agent's response to sensed gesture

through online training. Such *sensorimotor* mappings have been successfully used in the control of robotic agents [34, 35, 36]. Research described in these works has demonstrated that Self-Organizing Feature Maps (SOFM) [32] may be effectively used to discretize input space as cell areas surrounding reference nodes and to produce a nonlinear mapping between sensed input and robotic response.

Reinforcement Learning approaches (RL) are also frequently applied to the control of robots. Such methods use Markov Decision Processes (MDPs) to refine a mapping between an agent's state and its actions taken when in that state (a *state-action pair*). Over multiple iterations of input, action, and evaluation (via a scalar reward signal), a policy for maximizing the expected reward for taking a specific action in any given state is learned. In the limit, the policy that is developed can be seen to approach optimality [37]. One popular RL technique applied in this research is that of Q-Learning [38] for its simplicity and for its lack of need to model the environment.

Touzet [39] presents the *Q-Kohon* method which combines a Q-Learning framework with the nonlinear mapping capability of Kohonen's self-organizing map. The SOFM's topology-preserving structure allows for *neighborhood learning* wherein the action policy associated with a given node may be leveraged by nearby nodes to speed the learning process. Hence it applies well to the Q-Learning approach that underlies Touzet's method. Each node stores a tuple consisting of its state label (or *situation*), an action, and an expected reward or *Q value*. The input situation is a pointer into the map to the nearest state label having a positive Q value. The actions of nodes within a surrounding neighborhood are updated according to the reward received from taking the associated action. The approach used in this paper is a variant of Q-Kohon which makes use of Q-Learning coupled with the GNG algorithm for the benefits previously discussed. The implementation of Q-Learning in the proposed system is described in Section 3.3.

A human trainer may iteratively influence and expedite system learning through control of a reward signal [40, 41]. By allowing the user to control the application of reward, the user is not constrained by the interactions fashioned by the designer. Thus, the system is adaptable to the user rather than the user having to learn the system. Further, the selection of a simple binary reward system allows easy adoption of the system by unskilled or impaired users, giving a longer useful life to the system as the user's abilities degrade over time. And, as noted by Knox et al. [42], a human trainer possesses a high-level view of the benefit of a specific individual action. The trainer may give qualitative reward based on how a task might best be performed by an agent. It is shown in this research that, through gesture and a simple *warm/cold* evaluation of the robot's response, the agent will ultimately achieve the user's desired goal configuration.

2.2 Lifelong Learning

Learning systems have an operational lifespan that is conventionally divided into the distinct phases of learning versus recognition. This paradigm neglects the possibility that the system may need to acquire new recognition capabilities in the face of a changing input distribution from its environment. Typically, systems forced to consider new forms of input must reiterate the training phase. In so doing, these systems may suffer degradation in their ability to retain knowledge gains from the past. Thus, the stability of the system may be compromised by attempts to extend its gesture *vocabulary* [6]. This problem is termed the *Stability-Plasticity Dilemma* [43]. Toward the development of a system which can acquire and learn new gestures as needed without sacrificing past knowledge, the need for lifelong learning is considered.

The plasticity of the Growing Neural Gas (GNG) network lies in its ability to add and delete nodes during normal operation. Feature vectors (*weights*) of new nodes represent input patterns which differ from those seen in the past. As new nodes are added,the topology of the network is altered accordingly. Fritzke [8] proposed the incremental augmentation of a GNG network based on the periodic assessment of local error at each node. The node with the largest accumulated local error is the node whose receptive field is too large to adequately represent the distribution of inputs within the region and thus is most in need of a new node to reduce the total error of the network. However, in this simple form, incremental learning may result in the addition of a large number of nodes over time. In such a case, both overfitting at overlapping cluster boundaries and excessive computing time may result. Alternatively, a maximum node count may be set. However, this potentially limits network plasticity [6] as changing input may require deletion of existing nodes in favor of new clusters.

Fritzke [44] also proposed a utility-based approach (GNG-U) for deletion of nodes in order to allow GNG to track non-stationary input distributions while conserving computational resources. However, in terms of lifelong learning, this approach may remove nodes which represent past learning, thus leading to instability. Hamker [6] proposed a method for strategic insertion of nodes using local error thresholds developed from quality measures based on both long-term and short-term local error. The method was

shown to be effective, however, it focused on supervised learning scenarios.

Furao and Hasegawa [45] extend Hamker's approach to the insertion of nodes in unsupervised tasks. This method attempts to assign unlabelled data to clusters autonomously before applying an adaptive similarity threshold based on cluster size. An input pattern to an existing node is compared to the threshold to determine if it represents a new pattern class and is thus a candidate site for node insertion. This method, however, presupposes separable input distributions in order to place nodes in distinct clusters.

Each of the approaches mentioned above neglects the possibility of online learning and the need to accommodate a human user/trainer. The presence of a human user poses significant challenges in terms of input data separability and learning rate. As noted in [7], gesture motion data collected from human participants may be poorly separated which adversely affects the speed of convergence for a learning algorithm dealing with unlabelled input. This issue becomes especially important when considering the physical and cognitive burden to the user as they perform and apply feedback to potentially large numbers of gesture samples. Key contributions of the approach in this research include the proposal of a use model which reduces the physical burden on the user, and a method for making gesture classifications for lifelong learning with unlabelled data. These features are detailed in Section 3.

3 Method

This section describes the approach and system used for experimentation. The gesture data set and its relationship to the ART appliance are described. Toward the goals of reducing user effort and the size requirements of the input data set, an innovative use model and training paradigm is detailed. A new method for GNG node insertion which preserves network stability while promoting the rapid learning of new gestures is proposed and tested. Essential components of the system used for experimentation including data representation, simulation, reward generation and action learning based on GNG were first developed in [7]. A high-level description of this system is given here.

3.1 System Description

A block diagram of the data acquisition and learning system is shown in Figure 2. The system implements an end-to-end operational flow between a gesture performed by a human user and the learned response that is executed by a robotic agent in the form of a configuration in 3D space. Gestures were collected from human participant volunteers and presented as input patterns to the system.

Each gesture motion sample was processed to extract discriminative features (Dynamic Instants) to form motion descriptors. These descriptors were then applied to the GNG algorithm to topologically map the input data space. Each node in the resulting GNG *cloud* data structure has an associated action vector into the configuration space of the robotic agent which is selected for execution. A simulated human user observes the action of the robot and assigns reward based on whether movement was directed toward (positive reward) or away from (negative reward) the intended goal configuration. The assigned reward updates the action by adjusting the action vector's magnitude or directional orientation in configuration space (Figure 2). This update is stored for future reference. Through multiple cycles of $action \rightarrow reward \rightarrow update$, the responses of the agent are seen to converge to the user's preferred goals for each applied gesture type.

3.2 Growing Neural Gas

The Growing Neural Gas (GNG) algorithm [8] is a neural vector quantization technique in which neurons (reference nodes) represent vectors in input data space that encode a submanifold of that space. GNG forms connections between nodes so as to construct a topological representation of input space. It is capable of adding new nodes allowing for a changing input data distribution. For these characteristics, the GNG algorithm is well-suited to the goals of both stability and plasticity in the gesture learning task. The basic GNG algorithm is given by Algorithm 1. For our implementation of GNG, operating parameters were: $\epsilon_b = 0.05$, $\epsilon_n = 0.0006$, $\lambda = 100$, $\alpha = 0.5$, $\beta = 0.0005$ and $ageMax = 88$. A maximum limit of 40 nodes is imposed on the network. However, this limit can be strategically overridden for purposes of lifelong network plasticity by the proposed insertion algorithm as described in Section 3.7.

3.3 Q-Learning

In a reinforcement learning paradigm, an agent learns a policy for taking action from within any given state in its problem space. Rewards from the environment (here, the user) reinforce certain actions and thus, create mappings between current states and actions which result in the greatest rewards. The sequence of action followed by reward will, over multiple iterations, result in a policy which maximizes reward through any sequence of state-action pairs. This is accomplished through the formation of a table of state-action values (*Q values*). As the agent enters a state,

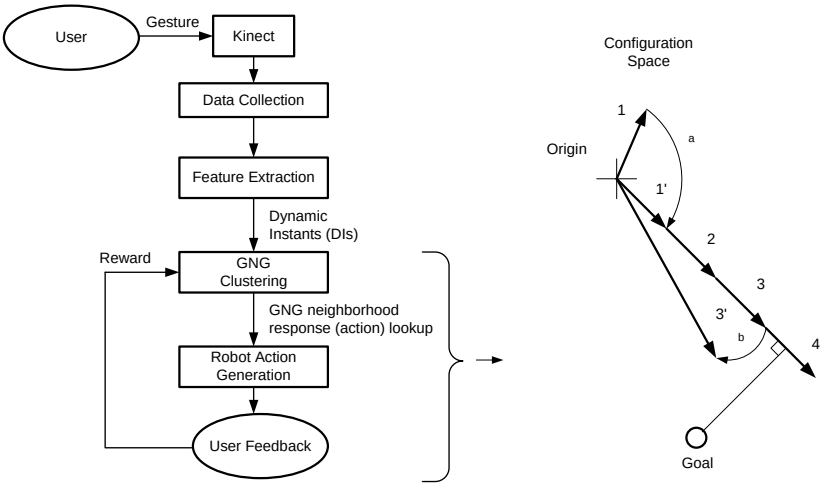
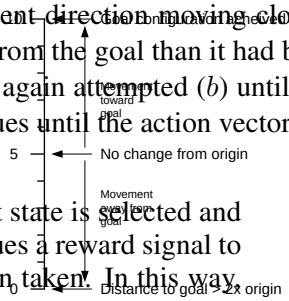


Figure 2: System block diagram showing the $action \rightarrow reward \rightarrow update$ loop with successive updates of actions toward a goal in the agent's configuration space. An initial action vector in step 1 moves the agent away from the goal (and receives a negative reward). Random angular adjustments are attempted until a results in a positively rewarded movement ($1'$) that is closer to the goal. Steps $1' - 3$ receive positive reward as the action vector is augmented in a consistent direction, moving closer to the goal. Continuing this policy at step 4 would cause the agent to move farther from the goal than it had been at step 3 and thus, would receive a negative reward. A random angular adjustment is again attempted (b) until the accumulated action vector comes closer to the goal as in step $3'$. The process continues until the action vector is within a predefined tolerance of the goal.



the highest-valued action for that state is selected and performed. The environment issues a reward signal to reinforce or discourage the action taken. In this way, the table is updated for each time step according to (1):

$$Q_{t+1}(s,a) \leftarrow Q_t(s,a) + \alpha[r + \gamma(\max_i Q_t(s',a_i)) - Q_t(s,a)] \quad (1)$$

where (s,a) is a state-action pair, (s',a_i) is a particular next state-action pair that may be chosen from the current state, α is a positive learning rate, γ is the discount factor that allows near term rewards to be valued more highly than future rewards, and r is the reward value.

For implementation of Q-Learning in the proposed system, several adaptations are made as follows [7]. Reward, $r \in \{-1, 0, 1\}$ may take values of -1 and 1 reflect user feedback of *bad* and *good* respectively. A reward of 0 reflects an outcome that requires no future adjustment (i.e. the human user is satisfied and training has been completed for a given gesture). Each gesture sample is followed by a training episode of one time step. Discounting is deemed unnecessary since each reward from the human user is seen as equally valuable evidence of movement toward or away from the goal configuration. Hence, $\gamma = 1$. Setting $\alpha = 1$ and multiplying the reward by a physical step length ($stepLen = 0.1$ units) of linear forward progress along a vector in configuration space, the update rule is reduced to the form of (2).

$$Q_{t+1}(s,a) \leftarrow r(stepLen) + \max_i Q_t(s',a_i)) \quad (2)$$

Each GNG node represents a Q-Learning state, and carries with it an action vector (output mapping) that represents a linear distance in configuration space. Topological neighbors in the GNG network may be expected to represent similar vectors in input space and thus, should elicit similar robotic actions. Given this intuition, the actions available to an agent are those of nodes within the neighborhood of a given reference node. Several neighborhood formation schemes are investigated as described in Section 3.4.

3.4 Network Neighborhood Formation

As previously stated, the connected nature of the GNG cloud allows for gesture learning to leverage previous positive outcomes for a neighborhood of nodes in a manner similar to Touzet's Q-Kohon approach [39]. This idea is based on the intuition that gestures in a

Algorithm 1 Growing Neural Gas

1: Begin with a set A of two nodes at positions w_a and w_b in $\mathbb{R}^n$: $A = \{a, b\}$.
2: Initialize a set of connections to the empty set: $C = \emptyset$.
3: **repeat**
4: Apply an input signal ξ according to $P(\xi)$.
5: Find nodes s_1 and s_2 in A closest to ξ.
6: Establish a connection between s_1 and s_2 if one does not exist: $C = C \cup \{(s_1, s_2)\}$.
7: Set the age of the connection (s_1, s_2) to zero.
8: Increment the ages of all edges connected to s_1.
9: Adjust the local error of s_1 by the square of its distance to the input: $\Delta E_{s1} = ||\xi - w_{s1}||^2$.
10: Move s_1 toward ξ by fraction ϵ_b: $\Delta w_{s1} = \epsilon_b(\xi - w_{s1})$.
11: Move the topological neighbors of s_1 toward ξ by fraction ϵ_n: $\Delta w_n = \epsilon_n(\xi - w_n)$.
12: Remove all edges having an age greater than $ageMax$. If this leaves any nodes with no connecting edges, remove them also.
13: **if** ($numGesturesSeen$ mod $\lambda = 0$) **then**
14: Determine the node q with maximum error.
15: Insert a new node r halfway between q and its neighbor f with the largest error: $A = A \cup \{r\}$ such that $w_r = 0.5(w_q + w_f)$.
16: Decrease the error of q and f by fraction α: $\Delta E_q = -\alpha E_q$ and $\Delta E_f = -\alpha E_f$.
17: Initialize the error of the new node to the interpolated error of its neighbors: $E_r = (E_q + E_f)/2$.
18: Decrease all node error variables by fraction β: $\Delta E_c = -\beta E_c \ \ \forall c \in A$.
19: **end if**
20: **until** Stopping criteria is met.

given region of the input space topology whose actions have accumulated significant positive reward (Q value) in the past are worthy of being emulated. The question of how to determine which nodes constitute a neighborhood will now be considered.

Five neighborhood scenarios, *Lone*, *Mean*, *Large*, *Floyd* and *Clumpiness*, were defined in order to explore various possibilities for exploiting the topology of the GNG cloud and to demonstrate the possible benefits to a neighborhood learning strategy afforded by GNG. Each scenario reflects a different manner for defining a neighborhood of reference nodes whose actions vectors are candidates for the agent's next actions based on past rewards. These schemes are described in Table 1. In the table, the *winner* refers the GNG reference node closest to the input feature vector by Euclidean distance.

Table 1: Neighborhood formation scenarios

Scenario	Description
Lone	Only the winner is considered.
Mean	Adjacent nodes within a mean connected distance of the winner are considered.
Large	All adjacent nodes are considered.
Floyd	The node with minimum network distance and Q value greater than the winner is considered (see Section 3.4.1).
Clumpiness	The node with the maximum clumpiness coefficient and Q value greater than winner is considered (see section 3.4.2).

3.4.1 Floyd's Shortest Distance Algorithm

Floyd's algorithm [46] determines the shortest path distances between node pairs in an undirected graph using edge lengths between individual nodes (see Algorithm 2). The algorithm returns a matrix $D_{n\times n}$ for a network with n nodes in which all entries d_{ij} represent the length of the shortest path between nodes x_i and x_j.

Algorithm 2 Floyd's shortest distance

1: Initialize $D_{n\times n}$ with all $d_{ij} = \infty$.
2: **for** $k = 1$ to n **do**
3: **for** $i = 1$ to n **do**
4: **for** $j = 1$ to n **do**
5: $d_{ij} \leftarrow \min\{d_{ik} + d_{kj},\ d_{ij}\}$
6: **end for**
7: **end for**
8: **end for**

Here, the *age* parameter of the GNG connection list (*C* in Algorithm 1) data structure is used to represent edge length. In this research, the distances between node pairs are set to ∞ (following execution of Floyd's algorithm in each time step) in cases where the action vector has yielded negative reward. This situation is indicative that the *ancestor* node (from which the current action was selected) should not have been emulated. Setting the distance to ∞ excludes that ancestor from consideration during the next time step.

3.4.2 Network Clumpiness

Frequently in classification problems, data of a given class may cluster nearer to the mean for that class than to the mean of a different class. Within a GNG graph topology, such clustering may be reflected in the

degree (number of connections) of a reference node. Given the unlabelled nature of the data used in this research, the number of gesture commands represented in a network is unknown. Hence, a metric involving the degrees of nodes and their relative distances from each other will be useful as a measure of cluster centrality among nodes associated with a given gesture type. With this motivation, Estrada's *clumpiness* metric [47] is considered for use in the formation of node neighborhoods. A node's clumpiness characteristic relates the respective degrees of a pair of nodes within a network to their distance from one another. A clumpiness coefficient Ξ_{ij} for a given pair of nodes x_i and x_j is computed by (3):

$$\Xi_{ij} = \begin{cases} \frac{k_i k_j}{(d_{ij})^2} & \text{for } i \neq j \\ 0 & \text{for } i = j \end{cases} \quad (3)$$

where k_i is the degree of node x_i and d_{ij} is the network distance between nodes x_i and x_j as computed using Floyd's algorithm (Section 3.4.1). It is shown in Section 5, that although computationally intensive, clumpiness is highly effective as a means of selecting neighborhood nodes with action vectors likely to yield positive rewards.

3.5 Gesture Types

Six gesture types are considered. These include the gesture motions for *eat*, *read*, *rest*, *take* (take an item away), *give* (bring an item closer) and *therapy* (engage in a therapy session using ART's specially designed therapy surface - see Figure 3(f)). These were selected as user-centered requests which would apply to common configurations of an over-the-bed table that might otherwise require the intervention of a caregiver. The envisioned goal configurations for these gestures will exercise the three degrees of freedom within ART: (1) the lifting column, (2) the sliding table top, and (3) the tilting work surface. Figure 3 depicts each configuration in a clinical setting. The configuration labels and their respective mappings to the three DOFs are given in Table 2.

3.6 Use Model

In this section, a use model is proposed which aims at reducing the physical burden to the user in terms of the number of training iterations required for the system to fully develop a desired actuation in response to a specific gesture. With this model, the user demonstrates a single sample of a new gesture to a system already trained to respond to some number of other gestures. The user then provides a series of consecutive rewards until the system is fully trained for that sample.

Table 2: 3D goal configurations for ART

Gesture Type	Lift	Slide	Tilt	Mapping (cm, cm, rad)
Eat	Low	Center	Down	$(0, 0, 0)$
Read	High	Center	Up	$(0, 0, \pi/3)$
Rest	High	Center	Down	$(10, 0, 0)$
Take	High	Away from user	Down	$(0, 10, 0)$
Give	High	Toward user	Down	$(0, -10, 0)$
Therapy	Middle	Center	Down	$(5, 0, 0)$

Training (or, *path shaping*) consists of simple good/bad (or *warm/cold*) rewards assigned to incremental movements of the robotic agent in response to the gesture. Movements which place the agent closer to a user-defined goal are assigned positive rewards. Movements away from the goal are assigned negative rewards. Gestures which, through training, elicit the attainment of the user's goal are deemed *trained*. Upon completion of training for a given gesture, the Q-Learning policy for the state-action pair (the associated GNG node and its action vector) is frozen. Thus, any subsequent similar gestures whose feature vector fall into the receptive field for the same node require no further training.

3.7 Learning with a Human Trainer

The presence of a human trainer poses a key distinction between earlier methods discussed in Section 2.2 and that presented in this paper. Here, input gesture samples are unlabelled and their data representations may not be well separated. However, using the proposed use model, the user-generated reward may be viewed as a binary *in-cluster/out-of-cluster* indicator. In the case of fully trained nodes, an input pattern which receives negative rewards when executing the action vector defined by that node must be of a different class than that to which the node has been trained. The location indicated by the input feature vector is interpreted as a likely good candidate site for node insertion.

The local accumulated error of the winner in this case (the node nearest the input feature vector) is artificially inflated to the network maximum. At the same time, any node in the network whose most recent reward is negative (*cold* nodes) are considered

(a) Eat (b) Read (c) Rest

(d) Take (e) Give (f) Therapy

Figure 3: 3D Configurations for ART in a clinical setting: (a) *eat* - the table surface is lowered to a comfortable dining height (b) *read* - the work surface is inclined, (c) *rest* - the table surface is raised (and moved aside), (d) *take* - the sliding surface is extended and away from the user, (e) *give* - the sliding surface is extended toward the user, (f) *therapy* - the table surface is at medium height to accommodate a therapy session.

for deletion. The *age* field for connections within the network may loosely be thought of as being indicative of a node's nearness to a cluster center. A node with older-aged connections has previously been matched with fewer incoming patterns in those regions where its connections are oldest. When the network has reached a defined maximum number of nodes, the node with the highest sum of connection ages is targeted for deletion by the artificial *aging* of its connections to the maximum age limit. If the network is not at the maximum node count, then a new node may be added without deletion elsewhere in the network. In cases where all nodes in the network are either fully trained or are receiving positive rewards, new nodes may be added above the predefined maximum. This effectively relaxes the predefined maximum to afford plasticity when needed. This scheme for node insertion/deletion is summarized in Algorithm 3.

4 Experimentation

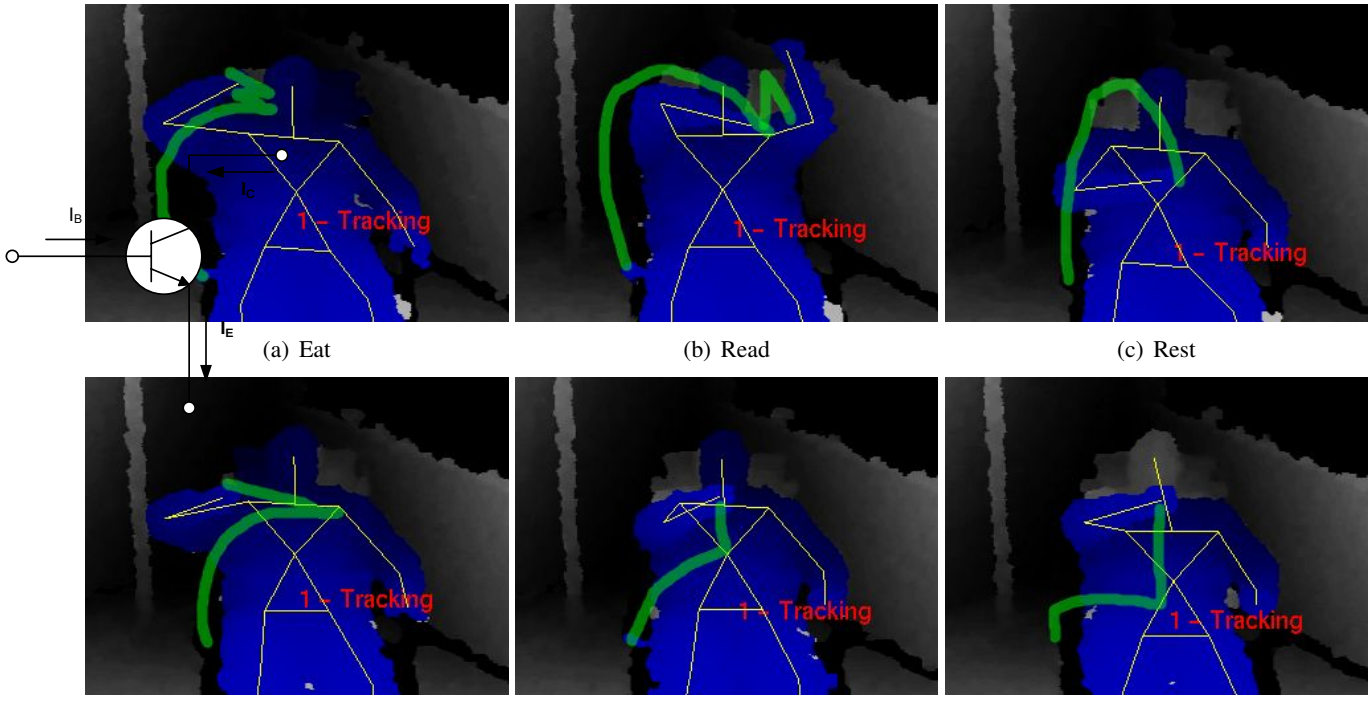
This section describes the experimental data sets and the neighborhood scenarios in which they were tested. The experimental procedure sequence and key outcome metrics are also given.

4.1 Data Collection

Gesture motion samples were collected at 30 Hz over five second intervals for a total of 150 data points per motion sample. Although RGB samples are also generated by the Kinect, these were not stored in order to preserve user anonymity. The Kinect was set at approximately 2 m above the floor with the participant in a pose a distance of 1.5 m from the Kinect The Kinect was angled so that the eleven joints of the upper body were visible in the depth image (Figure 4).

Six participants each performed 50 repetitions for each of the six candidate gestures. For the last 25 samples of each gesture, the participants were asked to wear a flexible 1.1 kg weight around their wrist. This was done in order to simulate degraded performance over time. Over the six participants, a total of 300 samples of each gesture were collected, or 1800 samples in all. Although no particular choreography was required by the proposed machine learning approach, performance models for these gestures were taken from the American Sign Language Dictionary (www.aslpro.com/cgi-bin/aslpro/aslpro.cgi) to support repeatability across the field of participants. Gestures were formed by motion of the actor's left hand as shown in Figure 4.

Dynamic instants (DIs) were computed for each



(a) Eat (b) Read (c) Rest

(d) Take (e) Give (f) Therapy

Figure 4: Kinect depth images showing skeletal tracking for an actor in a reclined pose. Trajectories of the actor's left hand sketched in green depict the candidate gesture motions.

sample. DIs are defined as the extrema of acceleration in the motion of an actor. The five DIs of greatest performance interval are concatenated to form feature vectors (Figure 5) which were presented to the system as described in Section 3.6.

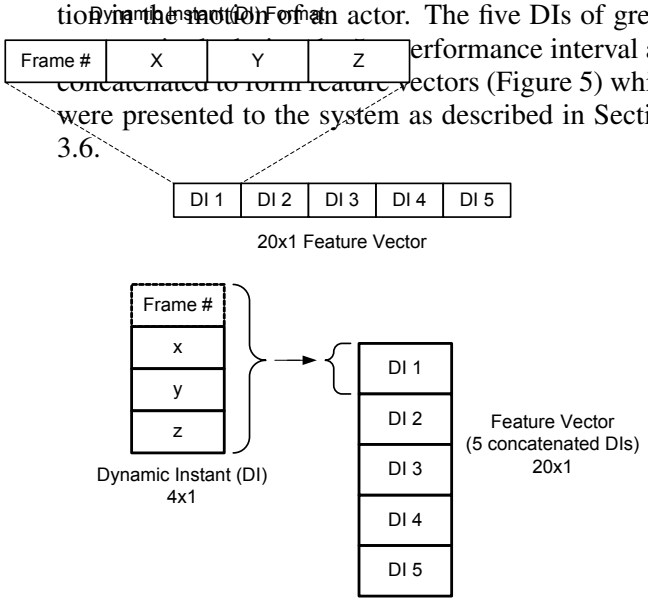


Figure 5: Feature vector format for a depth-sampled gesture. Coordinates (x, y, z) of the five DIs of largest magnitude are concatentated in chronological order by frame number.

4.2 Data Sets

The 1800 gesture samples for the six candidate gestures were divided into five data sets as given in Table 3. Data samples within the constrains shown in the table were randomized across the range of participants. This was done to exhibit the system's ability to generalize across significant variations in user body type and performance styles.

Table 3: Experimental data sets

Data set #	# samples	Gesture types included	Weights worn
1	100	*eat*, *read*, *rest*	No
2	350	*eat*, *read*, *rest*	No
3	450	*get*, *take*, *therapy*	No
4	450	*eat*, *read*, *rest*	Yes
5	450	*get*, *take*, *therapy*	Yes

4.3 Procedure

Data set 1 was used to initialize the GNG topology to include a base set of 40 nodes. No action training was conducted during application of data set 1. This was done using the *Lone* neighborhood forma-

Algorithm 3 Node insertion/deletion

1: Apply a gesture input sample.
2: Determine the $winner$ reference node $refNode$.
3: Perform the winner's associated action vector.
4: Observe the user-generated $reward$.
5: **if** $winner$ is trained and $reward$ is $cold$ or $warm$ **then**
6: Inflate local error: $winner.E = \max(refNode[i].E) + 1$.
7: **if** $numNodes < maxNodeCnt$ **then**
8: A node will be inserted near $winner$.
9: **else**
10: Locate a $cold$ node having the greatest sum of connection ages.
11: **if** A $cold$ node exists **then**
12: Target it for deletion by inflating connection ages: $C[i].age = ageMax + 1$.
13: **else**
14: $numNodes$ is allowed to increase beyond $maxNodeCnt$.
15: **end if**
16: A node will be inserted near $winner$.
17: **end if**
18: **end if**
19: GNG will perform node insertion and deletion in the next time step.

tion scheme (although, the choice of neighborhood formation method is not material at this stage of experiment).

Data set 2 was applied to perform initial action training upon which to judge plasticity and stability as described below. This allowed the system to set up the A (node list) and C (connection list) data structures (see Algorithm 1). Once trained, these structures fully define a mature GNG network for the *eat*, *read* and *rest* gestures contained in the training set. Data sets 3 through 5 were then applied in sequence to demonstrate the following system capabilities, respectively:

- Plasticity in learning actions for new gestures (data set 3)
- Stability of past learning, with degraded performance (weights worn) (data set 4)
- Stability of new learning with degraded performance (weights worn) (data set 5)

Two complete passes through a given data set is defined as an *epoch*. Using two passes allows nodes to train or, for new nodes to be first inserted as described in 3, and then trained. For each data set, a single epoch was applied one sample at a time according to the use model described in Section 3.6. Upon each presentation of a sample to the system, a simulation sequence was performed which included execution of GNG, generation of robot action, and assignment of reward. This sequence was repeated for the sample until one of three terminating conditions was reached:

1. The reference node closest to the input gesture sample became fully trained.
2. The input gesture sample received a negative reward in the receptive field of a fully trained node. In this case, the sample was immediately ignored and an additional node was inserted near the trained node according to Algorithm 3. Typically, an inserted node is trained on the following pass through the data set.
3. A *confusion threshold* of 300 training iterations was reached. This indicates that the formed neighborhood is issuing conflicting action advice and the input sample is near an inter-cluster boundary. In this case also, the sample was ignored.

Using this procedure, a 3-epoch sequence was conducted as described below. Following each epoch, performance metrics were recorded. These included the total number of nodes in the GNG network, the number of fully trained nodes, the percentage of samples ignored, and the average number of training iterations per sample. These metrics were chosen for their relationship to level of effort required by the user in training the system. The sequence was conducted for each of the five neighborhood formation methods described in Table 1. The 3-epoch sequence is as follows.

Epoch 1: Demonstration of plasticity. With the system initially trained using the training data set, a single epoch using data set 3 is applied. Execution of this epoch is intended to demonstrate the plasticity of the GNG network to learn the *take*, *give* and *therapy* gestures.

Epoch 2: Demonstration of stability of past learning. Data set 4 is applied in a single epoch. Execution of this epoch is intended to demonstrate the stability of the system learning implementation while adding variation for degraded performance. If the implementation is indeed stable, the outcome would be expected to reflect a network that retains past exposure to gesture *eat*, *read*, and *rest*. That is, the performance metrics would be expected to show iteration counts which remain tolerable to a human trainer.

Epoch 3: Demonstration of stability of new learning. Data set 5 is applied. This step reinspects the

network for the stability of the *take*, *give* and *therapy* gestures introduced by the test data set in Epoch 1, with the added variation due to degraded performance. Results for this procedure are given in Section 5.

5 Results

Typical results for execution of Epoch 1 in which data set 3 was applied is given in Table 4. Given that the GNG network was initially trained to the *eat*, *read* and *rest* gestures, application of data set 3 shows that the network exhibits plasticity in learning new gesture types *get*, *take* and *therapy* under the proposed use model.

Two metrics in particular are important in evaluating the model: (1) the percentage of samples ignored and (2) the average number of training iterations per sample. As previously mentioned, samples may be ignored by either falling into the receptive field of a node that is already trained, or by exceeding the confusion threshold through excessive training iterations. Ignoring such *problem* samples is justified based on the assertion that non-action on the part of the robot is preferred to persisting with training and ultimately performing a potentially undesirable action. Further, alteration of a previously trained action would negatively affect the stability of past learning. This assertion is predicated on the user making an accurate first attempt at performing the gesture. Thus, the priority for alteration of the system behavior is set in favor of stability over attempting to adjust to a rapidly changing input distribution. It can be seen from Table 4 that the percentage of samples ignored is relatively small, averaging less than 10%.

The *Clumpiness* scenario ignores the fewest samples. This is attributable the clumpiness metric's ability to assign actions based on cluster membership. With well-defined clusters in the GNG network, the proximity of any given gesture input to the cluster center for its class is likely to be closer. Given the variability of human performance, the cluster separability of the employed data set is poor. This can be seen in the large average number of iterations required by the *Large* neighborhood scheme. For well separated data, larger neighborhoods could be expected to yield the best results. Here, instead, increasing neighborhood size shows decreased performance. The performance of the clumpiness computation is, nonetheless superior. For more clearly separated data sets, the clumpiness metric would be more likely to form its neighborhood from nodes within its own class and thereby facilitate even faster learning. The average numbers of iterations is manageable, if still somewhat burdensome to the user. This is attributable to variations in participant body type and performance style.

Table 4: Results for application of new gesture types to a trained network

Neighborhood Formation	# Nodes	# Trained Nodes	Samples Ignored (%)	Average Iterations
Lone	52	49	12.0	11.05
Mean	51	48	8.2	21.58
Large	50	50	17.6	78.61
Floyd	79	64	7.3	47.99
Clumpiness	51	50	4.7	9.51

These results show that the fully trained network which existed before the test data was first applied is capable of learning new gestures in a relatively small (tolerable to a human user) number of time steps.

Table 5 shows results from an Epoch 2 in which data set 4 is applied to the network. In data set 4, new samples for gestures *eat*, *read* and *rest* are reapplied with degraded performance by the actor. These results reflect the stability of the GNG implementation. A network which had lost previous learning would show less than tolerable average numbers of iterations and a higher percentage of samples ignored. These results, however, show that both the average number of iterations and the percentage of samples ignored from the training set are relatively small for all neighborhood formation schemes excluding *Large* and *Floyd*. With the exception of the *Lone* case, the number trained nodes has increased which reflects the ability of GNG to continuously adapt to changing input. Having previously learned many of the gestures in data set 3, the topology of the network has been altered to accommodate wider interpretations of gestures that have incurred degraded performance.

It can be seen that *Clumpiness* still shows the fewest samples ignored. This results from the fact that fully trained nodes are not selected for action emulation during neighborhood formation. This feature of the system allows the Q-Learning to avoid what Knox calls the *Positive Circuits* problem [48]. In the attempt to maximize reward, Q-Learning may be expected to select its next state from those with the greatest expected return (a trained node). However, in cases where a neighborhood contains trained nodes which are of a different gesture type from the input, selecting them would be counterproductive. Thus, as the number of trained nodes increases, the available neighbor-

Table 5: Results for re-application previously traned gestures (degraded)

Neighborhood Formation	# Nodes	# Trained Nodes	Samples Ignored (%)	Average Iterations
Lone	43	42	38.2	7.28
Mean	51	49	22.9	33.20
Large	79	53	33.8	162.97
Floyd	70	66	26.7	43.68
Clumpiness	59	58	15.6	7.71

Table 6: Results for re-application of new gesture types (degraded)

Neighborhood Formation	# Nodes	# Trained Nodes	Samples Ignored (%)	Average Iterations
Lone	48	47	7.3	2.64
Mean	60	58	8.2	7.67
Large	86	65	8.2	26.42
Floyd	75	71	5.1	4.09
Clumpiness	64	62	2.9	2.58

hood from which a node may learn shrinks. Here, the *Mean* case chooses with similar productivity to *clumpiness* by choosing form a small neighborhood rather than seeking a cluster center.

Results from Epoch 3 indicate the ability of the system to remain stable through the application of data set 5 (*get*, *take* and *therapy* with degraded performance). Table 6 shows results from this scenario. Both the average numbers of iterations and the number of samples ignored have decreased from the first application of this data set under all neighborhood formation schemes. However, multiple executions of this experimental procedure do not show a clear best strategy for selecting nodes under the applied learning paradigm and use model. The *Clumpiness* method, however, is frequently seen to ignore the fewest samples. Although not reported quantitatively here, subsequent epochs for the available data sets typically result in convergence toward zero iterations per sample: the entire network having become fully trained.

The fact that the network may become, effectively, an associative memory with perfect recall of all input samples may be seen as problematic. The node insertion/deletion scheme of Algorithm 3 may insert nodes between clusters in regions where class decision boundaries overlap. Our method, in its present form, will ultimately be guilty of *overfitting* the problem. It will have placed too fine-grained a generalization on the input space, causing it to behave poorly in situations where gesture choreographies closely resemble one another between classes. A more discriminating method for node insertion (possibly considering inter-node distance) would need to be considered in order to better temper the system when handling poorly separated data.

6 Conclusions

In this paper, a robot control model based on arm-scale gesture has been proposed which considers the effort required by a human user to train a ART or other robotic agent to a desired behavior. The model aims at reducing the number of training iterations which must be performed by a human trainer in order for the robot to robustly learn a collection of gestured commands in the presence of changing/degrading performance. It is shown that sensorimotor mapping using GNG with Q-learning offers human-tolerable training durations. The *clumpiness* distance metric is shown to perform best as a means of neighborhood formation in the GNG cloud. Modifications to GNG are also presented which facilitate both plasticity and stability of learning.

Acknowledgements: This work was supported by the U.S. National Science Foundation under Award IIS-SHB-116075.

References:

[1] P. Yanik, A. Threatt, J. Merino, J. Manganelli, J. Brooks, K. Green, and I. Walker, "A Method for Lifelong Gesture Learning Based on Growing Neural Gas," in *Proc. of the 2014 Intl. Conference on Human-computer Interaction (HCII2014)*, pp. 191–202, Springer, Jun. 2014.

[2] K. Covinsky, R. Palmer, R. Fortinsky, S. Counsell, A. Stewart, D. Kresevic, C. Burant, and C. Landefeld, "Loss of Independence in Activities of Daily Living in Older Adults Hospitalized with Medical Illnesses: Increased Vulnerability

with Age," *Journal of the American Geriatrics Society*, vol. 51, no. 4, pp. 451–458, 2003.

[3] P. Yanik, J. Merino, J. Manganelli, L. Smolentzov, I. Walker, J. Brooks, and K. Green, "Sensor Placement for Activity Recognition: Comparing Video Data with Motion Sensor Data," *Intl. Journal of Circuits, Systems and Signal Processing*, vol. 5, pp. 279–286, 2011.

[4] J. Brooks, I. Walker, K. Green, J. Manganelli, J. Merino, L. Smolentzov, T. Threatt, P. Yanik, S. Ficht, R. Kriener, M. Mossey, A. Mutlu, D. Salvi, G. Schafer, P. Srikanth, and P. Xu, "Robotic Alternatives for Bedside Environments in Healthcare," *Intl. Journal of Systems Applications, Engineering and Development*, vol. 6, no. 3, pp. 308–316, 2012.

[5] A. Threatt, K. Green, J. Brooks, J. Merino, I. Walker, and P. Yanik, "Design and Evaluation of a Nonverbal Communication Platform between Assistive Robots and their Users," in *Proc. of the 2013 15th Intl. Conference on Human-computer Interaction (HCI 2013)*, pp. 505–513, Jul. 2013.

[6] F. Hamker, "Life-long Learning Cell Structures - Continuously Learning without Catastrophic Interference," *Neural Networks*, vol. 14, no. 4, pp. 551–573, 2001.

[7] P. Yanik, J. Merino, A. Threatt, J. Manganelli, J. Brooks, K. Green, and I. Walker, "A Gesture Learning Interface for Simulated Robot Path Shaping with a Human Teacher," *IEEE Transactions on Human-Machine Systems*, vol. 44, no. 1, pp. 41–54, 2014.

[8] B. Fritzke, "A Growing Neural Gas Network Learns Topologies," *Advances in Neural Information Processing Systems 7*, vol. 7, pp. 625–632, 1995.

[9] S. Mitra and T. Acharya, "Gesture Recognition: A Survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 37, no. 3, pp. 311–324, 2007.

[10] S. Jin, Y. Li, G. Lu, J. Luo, W. Chen, and X. Zheng, "SOM-based Hand Gesture Recognition for Virtual Interactions," in *Proc. of the 2011 IEEE Intl. Symposium on VR Innovation (ISVRI)*, pp. 317–322, Mar. 2011.

[11] J. Lementec and P. Bajcsy, "Recognition of Arm Gestures Using Multiple Orientation Sensors: Gesture Classification," in *Proc. of the 7th IEEE Intl. Conf. on Intelligent Transportation Systems, 2004*, pp. 965–970, Oct. 2004.

[12] S. Zhou, Q. Shan, F. Fei, W. Li, C. Kwong, P. Wu, B. Meng, C. Chan, and J. Liou, "Gesture Recognition for Interactive Controllers Using MEMS Motion Sensors," in *Proc. of the 4th IEEE Intl. Conf. on Nano/Micro Engineered and Molecular Systems, 2009 (NEMS 2009)*, pp. 935–940, Jan. 2009.

[13] C. Zhu and W. Sheng, "Wearable Sensor-Based Hand Gesture and Daily Activity Recognition for Robot-Assisted Living," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 41, pp. 569–573, May 2011.

[14] S. Berman and H. Stern, "Sensors for Gesture Recognition Systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 42, no. 3, pp. 277–290, 2012.

[15] S. Beach, R. Schulz, J. Downs, J. Matthews, B. Barron, and K. Seelman, "Disability, Age, and Informational Privacy Attitudes in Quality of Life Technology Applications: Results from a National Web Survey," *ACM Transactions on Accessible Computing (TACCESS)*, vol. 2, no. 1, pp. 1–21, 2009.

[16] C. BenAbdelkader, R. Cutler, and L. Davis, "Gait Recognition Using Image Self-Similarity," *EURASIP Journal on Applied Signal Processing*, vol. 2004, pp. 572–585, 2004.

[17] A. Bobick, "Movement, Activity and Action: The Role of Knowledge in the Perception of Motion," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 352, no. 1358, pp. 1257–1266, 1997.

[18] A. Karahoca and M. Nurullahoglu, "Human Motion Analysis and Action Recognition," in *Proc. of the 1st WSEAS Intl. Conf. on Multivariate Analysis and its Application in Science and Engineering*, pp. 156–161, WSEAS, 2008.

[19] G. Johansson, "Visual Perception of Biological Motion and a Model for Its Analysis," *Perception and Psychophysics*, vol. 14, no. 2, pp. 201–211, 1973.

[20] R. Cutler and L. Davis, "Robust Real-time Periodic Motion Detection, Analysis, and Applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 781–796, 2002.

[21] I. Junejo, E. Dexter, I. Laptev, and P. Pérez, "Cross-View Action Recognition from Temporal Self-Similarities," *European Conf. on Computer Vision–ECCV 2008*, pp. 293–306, 2008.

[22] C. Rao, A. Yilmaz, and M. Shah, "View-Invariant Representation and Recognition of Actions," *Intl. Journal of Computer Vision*, vol. 50, no. 2, pp. 203–226, 2002.

[23] A. Wilson and A. Bobick, "Realtime Online Adaptive Gesture Recognition," in *Proc. of the 15th Intl. Conf. on Pattern Recognition*, vol. 1, pp. 270–275, IEEE, 2000.

[24] M. Moni and A. Ali, "HMM Based Hand Gesture Recognition: A Review on Techniques and Approaches," in *Proc. of the IEEE Intl. Conf. on Computer Science and Information Technology*, pp. 433–437, 2009.

[25] A. Bobick and A. Wilson, "A State-Based Approach to the Representation and Recognition of Gesture," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 1325–1337, Dec. 1997.

[26] J. Prasad and G. Nandi, "Clustering Method Evaluation for Hidden Markov Model Based Real-time Gesture Recognition," in *Proc. of the Intl. Conf. on Advances in Recent Technologies in Communication and Computing, 2009 (ARTCom'09)*, pp. 419–423, Oct. 2009.

[27] T. Schlömer, B. Poppinga, N. Henze, and S. Boll, "Gesture Recognition with a Wii Controller," in *Proc. of the 2nd Intl. Conf. on Tangible and Embedded Interaction*, pp. 11–14, 2008.

[28] M. Yang and N. Ahuja, "Recognizing Hand Gesture Using Motion Trajectories," in *Proc. of the 1999 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*, vol. 1, pp. 466–472, Jun. 1999.

[29] T. Alexander, H. Ahmed, and G. Anagnostopoulos, "An Open Source Framework for Real-time, Incremental, Static and Dynamic Hand Gesture Learning and Recognition," *Human-Computer Interaction. Novel Interaction Methods and Techniques*, pp. 123–130, 2009.

[30] E. Stergiopoulou and N. Papamarkos, "A New Technique for Hand Gesture Recognition," in *Proc. of the Intl. Conf. on Image Processing*, pp. 2657–2660, Oct. 2006.

[31] D. Frolova, H. Stern, and S. Berman, "Most Probable Longest Common Subsequence for Recognition of Gesture Character Input," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 43, no. 3, pp. 871–880, 2013.

[32] T. Kohonen, "The Self-organizing Map," *Proc. of the IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.

[33] J. Holmström, "Growing Neural Gas: Experiments with GNG, GNG with Utility and Supervised GNG," Master's thesis, Uppsala University – Department of Information Technology, 2002.

[34] H. Ritter, T. Martinetz, and K. Schulten, "Topology-conserving Maps for Learning visuo-motor-coordination," *Neural Networks*, vol. 2, no. 3, pp. 159–168, 1989.

[35] T. Martinetz, H. Ritter, and K. Schulten, "Three-dimensional Neural Net for Learning Visuomotor Coordination of a Robot Arm," *IEEE Transactions on Neural Networks*, vol. 1, no. 1, pp. 131–136, 1990.

[36] G. Barreto, A. Araújo, and H. Ritter, "Self-organizing Feature Maps for Modeling and Control of Robotic Manipulators," *Journal of Intelligent & Robotic Systems*, vol. 36, no. 4, pp. 407–450, 2003.

[37] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*. Cambridge Univ Press, 1998.

[38] C. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3, pp. 279–292, 1992.

[39] C. Touzet, "Neural reinforcement learning for behaviour synthesis," *Robotics and Autonomous Systems*, vol. 22, no. 3, pp. 251–281, 1997.

[40] F. Kaplan, P. Oudeyer, E. Kubinyi, and A. Miklósi, "Robotic clicker training," *Robotics and Autonomous Systems*, vol. 38, no. 3, pp. 197–206, 2002.

[41] A. Thomaz and C. Breazeal, "Teachable Robots: Understanding Human Teaching Behavior to Build More Effective Robot Learners," *Artificial Intelligence*, vol. 172, no. 6-7, pp. 716–737, 2008.

[42] W. Knox, I. Fasel, and P. Stone, "Design Principles for Creating Human-shapable Agents," in *AAAI Spring 2009 Symposium on Agents that Learn from Human Teachers*, pp. 79–86, 2009.

[43] S. Grossberg, "Nonlinear Neural Networks: Principles, Mechanisms, and Architectures," *Neural Networks*, vol. 1, no. 1, pp. 17–61, 1988.

[44] B. Fritzke, "A Self-organizing Network That Can Follow Non-stationary Distributions," in *Proc. of the 1997 Intl. Conf. on Artificial Neural Networks (ICANN'97)*, pp. 613–618, Springer, 1997.

[45] S. Furao and O. Hasegawa, "An Incremental Network for On-line Unsupervised Classification and Topology Learning," *Neural Networks*, vol. 19, no. 1, pp. 90–106, 2006.

[46] A. Tucker, *Applied Combinatorics*. Wiley, 6th ed., 2007.

[47] E. Estrada, *The Structure of Complex Networks: Theory and Applications*. Oxford, 2012.

[48] W. Knox, *Learning from Human-Generated Reward*. PhD thesis, The University of Texas at Austin – Department of Computer Science, 2012.