A Gesture Learning Interface for Simulated Robot Path Shaping With a Human Teacher

Paul M. Yanik, *Senior Member, IEEE*, Joe Manganelli, Jessica Merino, Anthony L. Threatt, Johnell O. Brooks, Keith Evan Green, *Member, IEEE*, and Ian D. Walker, *Fellow, IEEE*

***Abstract*—Recognition of human gestures is an active area of research integral for the development of intuitive human–machine interfaces for ubiquitous computing and assistive robotics. In particular, such systems are key to effective environmental designs that facilitate *aging in place*. Typically, gesture recognition takes the form of template matching in which the human participant is expected to emulate a choreographed motion as prescribed by the researchers. A corresponding robotic action is then a one-to-one mapping of the template classification to a library of distinct responses. In this paper, we explore a recognition scheme based on the growing neural gas (GNG) algorithm that places no initial constraints on the user to perform gestures in a specific way. Motion descriptors extracted from sequential skeletal depth data are clustered by GNG and mapped directly to a robotic response that is refined through reinforcement learning. A simple good/bad reward signal is provided by the user. This paper presents results that show that the topology-preserving quality of GNG allows generalization between gestured commands. Experimental results using an automated reward are presented that compare learning results involving single nodes versus results involving the influence of node neighborhoods. Although separability of input data influences the speed of learning convergence for a given neighborhood radius, it is shown that learning progresses toward emulation of an associative memory that maps input gesture to desired action.**

***Index Terms*—Gesture recognition, human-robot interaction, machine learning, robots.**

I. Introduction

As people age, they must often deal with decreased mobility. Such reductions may ultimately impair one's ability to perform essential activities of daily living (ADLs). For those wishing to age in place, a diminished capacity to conduct ADLs is frequently an indicator for diminished quality of life, decreased independence, increased caregiver burden, or institutionalization [1]. With this population in mind, the authors envision a comprehensive system of adaptive architectural and robotic components to support independent living for individuals whose capabilities and needs are changing over potentially long periods of time [2].

Heretofore, architects and environmental designers have attempted to accommodate those with physical impairment through the use of universal design principles (UDPs) and *smart home* technologies. UDPs ensure that the environment does not confound an individual's efforts to complete tasks. UDPs aim to make the environment safe, clean, legible, and barrier-free [3]–[5] for all occupants regardless of ability. These strategies facilitate resident mobility and independence. However, the majority of current implementations are static and of low fidelity, with accommodation solely the result of the form and placement of furniture and fixtures.

Smart homes aim to extend awareness, increase control over systems, and enhance the security, healthfulness, and safety of the environment through sensing, inference, communication technologies, decision-making algorithms, and appliance control [6]–[9]. However, the real-time processing of occupant activity has historically been costly in terms of computing and memory requirements and often relies on technologies considered intrusive of people's privacy (e.g., cameras). As a result, these efforts have focused on systems associated with the built environment such as the design and placement of furniture and fixtures. Practical occupant sensing in smart homes remains of low fidelity including such ON/OFF sensor activations as room changes, door openings/closings, appliance actuations, etc.

A logical progression for the use of high fidelity sensing may be seen in its central importance to assistive robotics. As Green and Walker describe [10], the notion of assistive robotics frequently conjures images of a self-contained *humanoid servant* in which all robotic and intelligence challenges have been addressed. Finding this to be an unlikely possibility in the near term and seeking to move beyond the conventional *static* smart home, we envision an environment containing robotic components that take advantage of the capabilities and higher level thinking of the user to operate in a collaborative manner; working *with* rather than *for* the user. The authors' past investigations into possible forms and use models for assistive robotics have considered appliances such as a hospital over-the-bed table, continuum surfaces, and intelligent storage for personal items [11]–[13]. Current efforts on the *Assistive Robotic Table* project under way at Clemson University make use of high fidelity sensing to create a nonverbal communication loop between the device and a user in a health care setting (see Fig. 1).

Manuscript received August 31, 2012; revised March 25, 2013 and August 14, 2013; accepted November 2, 2013. Date of publication December 18, 2013; date of current version January 16, 2014. This work was supported by the U.S. National Science Foundation under Award IIS-SHB-116075. This paper was recommended by Associate Editor A. Howard of the former IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans (2012 Impact Factor: 2.183).

P. M. Yanik is with the Department of Engineering and Technology, Western Carolina University, Cullowhee, NC 28723 USA (e-mail: pyanik@wcu.edu).

J. Merino and I. D. Walker are with the Department of Electrical and Computer Engineering, Clemson University, Clemson, SC 29632 USA (e-mail: jmerino@clemson.edu; iwalker@clemson.edu).

J. Manganelli, A. L. Threatt, and K. E. Green are with the School of Architecture at Clemson University, Clemson, SC 29632 USA (e-mail: jmangan@clemson.edu; threatt@clemson.edu; kegreen@clemson.edu).

J. O. Brooks is with the Department of Psychology, Clemson University, Clemson, SC 29632 USA (e-mail: jobrook@clemson.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMC.2013.2291714

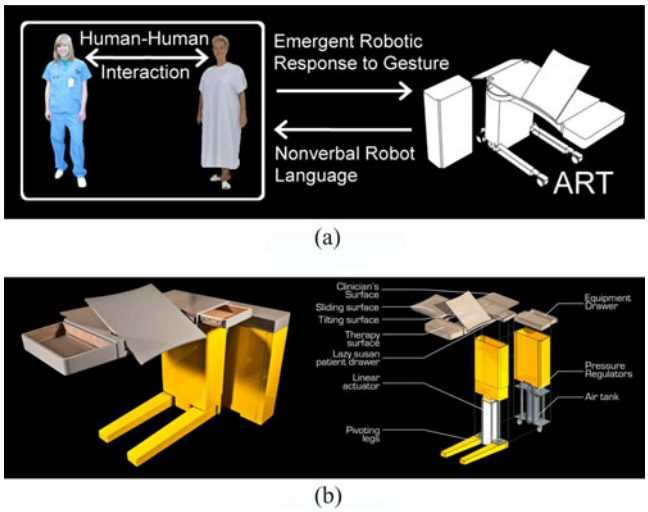


Fig. 1. (a) Nonverbal communication loop of the *Assistive Robotic Table* being developed at Clemson University. The focus of this paper is on the emergent (learned) response of this device to the user. (b) Current project artifact.

The high fidelity sensing employed in efforts such as these will allow for learned inference of user action and intention through persistent monitoring. Furthermore, degradation in the abilities of the user could be tracked over time so as to adaptively inform the robot's assistive action plans. With knowledge of typical user motion patterns, the robot could respond to gestured commands or detect infrequent needs such as assistance with reach, weight transference, or ambulation [2]. Effectively implemented, assistive robotic components would facilitate the aforementioned aims of smart home technologies.

Toward the realization of these components, this paper presents research into one possible mode of human–machine interaction: the use of arm-scale gesture as a basis for a learned command vocabulary. Given the prevalence of gesture at this scale as a means of human communication (see Section II), the system presented here offers promise as an interface that is effective, extensible, and intuitive to the user. However, due to the range of possible performance variations by the actor, the capability for such an interface to generalize across gestures with few observations is seen as essential to its success. Addressing this challenge is a key contribution of this paper. Furthermore, the ability of the proposed system to learn new gestures with no adherence to a choreographed example offers accessibility to the unskilled or impaired user. Therefore, while basic technical challenges associated with robot path shaping could be addressed by simpler mechanisms such as a joystick or push-button interface, this paper focuses on the research aspects of gesture learning that would cater to a broader user community.

This paper is structured as follows. Section II describes constituent problem areas associated with gesture recognition. Related work in each area is presented along with its respective benefits, drawbacks, and how it informs our approach. Section III discusses the specifics of the system design including data representations, algorithms, and our simulation environment. Section IV discusses our data collection fixture, experimentation scenarios, and results. Section VI presents conclusions and future work.

II. Related Work

Human gesture may occur in various forms including hand and arm gesticulation, pantomime, sign language, static poses of the hand and body, or language-like gestures that may replace words during speech. Of these, hand and arm gesticulation account for some 90% of gestured communication [14]. Hence, the exploration of gesture at this scale as a means of command interaction with robotics and computing is warranted. Efforts at automated gesture recognition generally involve a common set of considerations and problems to be addressed. These include some combination of sensor platform, data representation, pattern recognition, and machine learning. This section discusses previous approaches to these problems relative to the methods applied in this paper.

A. Sensing

In order for gestures to be detected and classified, the motion or pose of the actor must be sensed. Typical sensor strategies include wearable devices such as data gloves or body suits that are instrumented with the magnetic field tracking devices or accelerometers, or vision-based techniques involving one or more cameras [14]. Still other approaches involve IR motion or proximity sensors.

Jin *et al.* [15] use a glove-based orientation sensor to extract static hand positions to be used as commands. Lementec and Bajcsy [16] use wearable (arm) orientation sensors for sensing arm gesture models composed of Euler angles. These are intended for use in an unmanned aerial vehicle and implemented as a lab simulation. Zhou *et al.* [17] use MEMS accelerometer data to characterize hand motions including *up*, *down*, *left*, *right*, *tick*, *circle*, and *cross*. Wearable sensors are also used in [17]–[19], and others. Typically, however, the usefulness of wearable devices to measure gestured motion is accompanied by the acknowledgment that such devices may limit user motion and often require a wired connection to a computer. Thus, they present inherent impediments to practical application [14].

IR proximity sensors are used by Cheng *et al.* [20] to create a reliable gesture recognition system for a touchless mobile device interface. The method uses the pairwise time delay between a passing user's hand and two IR proximity sensors. This system detects gestures of *swipe right*, *swipe left*, *push*, and *pull*. Dongseok *et al.* [21] propose a computer control interface design using a proximity sensor to extract hand commands to a GUI. The mechanism is scaled as a mouse replacement. Such coarse assessment of motion is not sufficiently descriptive to support an extensive vocabulary of gestures. However, as shown by Yanik *et al.* [2], an array of IR motion sensors can provide sufficiently rich data to allow for accurate classification of gross motions.

Much of the work in gesture recognition is performed using video image sequences due to the richness of information and cost effectiveness available with cameras. A recent thorough discussion of vision based and other sensor types for the purpose of gesture recognition is given by [22]. Vision-based

approaches may suffer from disadvantages associated with latency, occlusion, or lighting. Furthermore, since most video sequences represent a 3-D to 2-D projection, a loss of information is inherent in the processing of data [14]. In addition, although the presence of cameras in an individual's personal environment is becoming more common, they are often considered intrusive of privacy in certain scenarios [23], [24].

With the limitations of these various sensor types in mind, the study reported in this paper utilizes the Microsoft Kinect RGB-D depth sensing system [25]. The Kinect provides a rich, real-time, 3-D data stream that preserves user anonymity and is also functional in dark environments where conventional cameras would be ineffective.

B. Data Representation

Given an input data stream, a compact data representation must be computed. Representations may be roughly divided into feature-based (parametric) versus holistic (nonparametric) forms. Parametric representations extract features related to the physical geometry and kinematics of the actor. Spatial information is preserved.

Holistic representations utilize statistics of the motion performed (typically in (x, y, t) space). Hence, with regard to the frequently employed visual images of motion, these can also be characterized as pixel-based representations [26]. In general, however, the problem of data representation is one of feature selection. Some vector to characterize numerical features is selected and applied to a classifier.

Motion history images have been used to form a visual template of motion that preserves directional information [27], [28]. Histograms of oriented gradients (HOGs) are used in [29] to generate regional descriptors of single-frame images for human detection. Periodic motions such as walking or running may be recognizable solely from the movement of lighted feature points placed on the actor's body [30]. This phenomenon is exploited by Benabdelkader *et al.* [26] and Cutler and Davis [31] through the concept of self-similarity. In this approach, the locations of features (e.g., edges) in an image sequence are seen to generate a repeating pattern from which a motion descriptor may be generated. The set of features is tracked through the course of an image sequence. The summed distances of features between image pairs is computed. Performing this summation exhaustively across all image pairs forms a self-similarity matrix (SSM).

HOGs and SSMs are combined to produce view-invariant representations for nonperiodic motions in [2] and [32]. Results described in these works show that recurrences in a spatial sensor or video data can produce robust discriminants. Although these representations possess strong discriminative qualities, they tend to be of high dimension and require either compression or excessive computation.

In this paper, we extend the concept of dynamic instants (DIs) advanced by Rao *et al.* [33] to three dimensions. DIs are described as the extrema (or discontinuities) of acceleration in an actor's motion. This representation has also been shown to be view-invariant. The Kinect allows us to directly extract a 3-D rather than working with typical 2-D video. We form our representation using the five most significant DIs in (x, y, z) space along with their frame number over a 5-s interval at 30-Hz sampling. This is described further in Section III.

C. Pattern Recognition

In order to classify gestures, the feature vector is typically sorted into one of a known gallery of types. Numerous classification methods have been introduced including hidden Markov models (HMMs), finite-state machines (FSMs), clustering techniques such as nearest neighbor (*k*NN) and C-means, and various types of artificial neural networks including multilayer perceptron networks, time delay neural networks (TDNN) [14], neural networks based on the adaptive resonance theory (ART) [34], neural gas (NG) [35], and growing neural gas (GNG) [36].

HMMs have well-established success in the classification of gestures and of generalized motion, and are used in numerous research efforts. Notably, these include [37] and [38]. A survey of such approaches can be found in [39]. The authors note that HMM approaches may inaccurately assume that observation parameters may be approximated by a mixture of Gaussian densities. HMMs often have poorer discriminative outcomes than neural networks [40], [41] and require that the state architecture of the model and the lexicon of recognizable symbols be decided beforehand [42].

Bobick and Wilson [43] use FSMs to classify gestures collected from video images. Lee [44] seeks to classify video motion sequences as whole-body gestures by mapping sequences of estimated poses to gestures. Principal component analysis is used for visualization; an expected maximum-based Gaussian mixture model is used to cluster poses. Frolova *et al.* [45] classify planar decimal digits traced in free air with high accuracy by storing hand trajectories. The most probable longest common subsequence algorithm is used to classify trajectories by comparison with a probabilistic template based on variations within a Gaussian mixture model. Prasad and Nandi [46] explore the effectiveness of several methods to vectorize and cluster gesture motion data including: hierarchical, mean shift, k-means, fuzzy C-means, and Gaussian mixture. Schlömer *et al.* [47] use k-means to determine clusters in basic hand/arm gestures generated using a Wiimote controller including *square*, *circle*, *roll*, *Z*, and *tennis swing*. Wachs *et al.* [48] use fuzzy C-means clustering to achieve highly accurate recognition of 12 static hand gestures as the basis for a telerobotic command interface. And, although the focus of Knox's work in [49] is user-guided machine learning (see Section II-D), the author uses *k*NN to determine the probable state of a robot from sensor data in order to conduct state-action selection as the basis for a user reward function. The experimentation described in this paper is compared with a *k*NN approach. Unlike Knox, however, we assume that the sensed gesture is the state, rather than the sensed position of the robot.

Zhu and Sheng [19] use wearable sensors to detect both hand gestures and simple ADLs. Neural networks are used for gesture spotting. HMMs are used for classification. Varkonyi-Koczy and Tusor [50] use circular fuzzy neural networks (CFNNs) to classify static hand postures for their iSpace intelligent environment.

CFNNs are seen to have reduced training time. Sequences of hand postures are composed into hand gestures. Yang and Ahuja [51] use TDNNs to classify sequences of motion trajectories in hand motion for American sign language (ASL). Alexander *et al.* [52] use a neural network based on the ART to recognize static hand gestures. ART networks are seen to possess the ability to learn incrementally, thus making them effective in online learning.

Stergiopoulou and Papamarkos [53] use GNG to model the topology of the hand itself (rather than more abstract features of the scene) in various finger-extended postures. The skin color is used as the dominant feature. From this, finger directions are extracted based on the centroid of the palm. Classification is accomplished using Gaussian probability of finger angles. Angelopoulou *et al.* [54] present a probabilistic growing neural gas (A-GNG) method to track the topology of the human hand as it progresses through various gestures. A-GNG offers improved topology mapping to the basic GNG algorithm. However, the approach is chiefly video based and forms the GNG codebook vectors based on the appearance of the hand rather than on any of the movement characteristics of the action. This way, the method is mainly that of a static analysis of hand shape.

The GNG algorithm [36] is a variant of the self-organizing feature map (SOFM). Because it is capable of tracking a moving distribution [55], adding new reference nodes, and operating from static input parameters, it is well suited to the task of gesture recognition where no labeled data are available. Indeed, since the acquisition of gesture data is often 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. Furthermore, its ability to grow and alter its topology over time suggests that it may be effective in learning new gestures as they are observed. For these reasons, GNG is the clustering method explored in this paper.

D. Machine Learning

Although techniques described in Section II-C may be broadly categorized as machine-learning methods, our use of this term applies to a mechanism by which some manner of feedback is used to improve future outcomes of a robot's assistive behavior. Typically, such a mechanism implies the use of training data to refine a classifier of choice offline as with conventional neural networks. However, a goal of this paper is to create an online learning modality that utilizes direct interaction with the user so that a robot agent converges upon a desirable configuration. Hence, our goal is to iteratively create a direct mapping between sensed gestures and inferred goals.

Such *sensorimotor* mappings of sensor input to robot motor commands have been successfully used in several applications. Ritter *et al.* [56] and Martinetz *et al.* [57] showed that SOFMs [58] could be used to discretize input space into receptive fields associated with individual neurons. Each node in the network then uses an error correction rule to learn an output composed of a vector of joint angles and a Jacobian to affect a desired robot configuration. This way, the SOFM is capable of a nonlinear mapping between input and output spaces. The topology preserving nature of the SOFM allows for faster learning than conventional neural networks by taking advantage of the idea that similar inputs should yield similar outputs. Hence, topological neighbors will encode similar sensor inputs and thus, they can be made to learn desired outputs as a group. Walter and Schulten [59] use a NG mapping [35] and apply a Gaussian neighborhood function to soften learning across the discretized input space of nodes to produce smoother output control. A good survey of these and related applications can be found in [60].

Reinforcement learning (RL) approaches are frequently applied to the control of robots. Unlike supervised learning approaches, which require a set of training data with desired output values, an agent (robot) in an RL framework senses its environment and operates under some policy so as to maximize the expected future returns (evaluations) it will receive through a scalar reward signal. RL techniques use Markov decision processes (MDPs) to refine a mapping between an agent's state and its future actions. Over successive iterations of input, action, and evaluation, a policy to maximize reward is learned, which in the limit, can be seen to approach optimality [61]. Arguably, the most popular RL technique is that of Q-Learning [62] for its simplicity and for its lack of need to model the environment. We apply Q-Learning in this paper as described in Section III-E.

Within an RL framework, an agent in a particular state s of the environment, an action a is selected based on the highest available expected return (or Q *value*). The policy may be periodically modified to allow for exploration of the action space. Following each episode of state-action sequences toward a known goal, the policy is evaluated, and a table of state-action pairs is updated to reflect the actual realized returns (which may be expected in future episodes under a given policy). Typically, convergence to an optimal policy requires a large number of iterations during a training phase. In the field of robotics, this is generally impractical to achieve, given the potentially large number of state-action pairs coupled with the mechanical limitations of execution speed, reliability, and energy consumption. Hence, generalization of actions across similar states is critical [63].

Touzet [64] presents a method for generalization among state-action pairs in a Q-Learning framework using Kohonen's self-organizing map (Q-Kohon). As previously mentioned, the SOFM's topology preserving structure allows for *neighborhood learning*. Hence, it applies well to the Q-Learning approach that underlies Touzet's method. Q-Kohon uses the SOFM as an associative memory. Each node stores a tuple consisting of its state label (or *situation* in Touzet's terminology), an action, and a Q value. The input situation probes the map for the nearest state label having a positive Q value. The neighborhood actions are updated according to the reward received from taking the action associated with this node.

The approach used in this paper is an adaption of Q-Kohon. As previously stated, we employ the GNG algorithm so as to avoid extensive parameter tuning. In addition, the capability of the GNG topology to add nodes in the presence of new gesture forms or significant distribution error is seen as key. However, the strengths of the SOFM paradigm remain available to us.

Usually, reinforcement learning utilizes an automated, internally generated reward function. As previously mentioned, the number of trials required to learn an optimal policy in this case is large. Furthermore, the reward function is typically sparse in nature. For example, Tesauro's implementation of a backgammon player taught by reinforcement learning assigned a reward of 1 (one) for a winning game and a reward of 0 (zero) otherwise. Given the huge state space of the backgammon game, the player required hundreds of thousands of games to become proficient [61].

For our application to assistive robotics, and in particular, to robotic agents capable of learning gestured human commands, such lengthy training phases are not feasible. As such, several variants of knowledge transfer between human teachers and robots have been devised. A summary of these approaches was produced by Knox and Stone [65] that covers advice-taking agents, learning by example, and human-generated reward signals. The authors note that the advising of agents in a meaningful way may involve expertise beyond that of a typical user. Learning by example, in which the user demonstrates a desired response, may place a burden on the user to observe the outcome or require that they possess expertise to generate an adequate example (as with simulated aircraft operation).

One straightforward way for a human teacher to influence the learning of a robotic agent is by allowing them to control a simple *good/bad* reward indicator. Kaplan *et al.* [66] proposed the use of the animal training technique known as *clicker training* to teach an AIBO dog robot to learn complex actions. Blumberg *et al.* [67] extend this idea to use reinforcement learning to instruct virtual characters. Breazeal and Thomaz [68] use RL in a simple environment with a small number of states to show that tasks within this space can be learned through clicker-like guidance. Kartoun *et al.* [69] create an extension to Q(λ) (Q-Learning with mulitple step eligibility tracing) to switch between fully autonomous and semiautonomous operation (accepting human guidance) in a bag emptying task. This approach, called CQ(λ) allows for levels of collaboration with a human observer. It is shown that the influence of human guidance speeds the learning process.

Similarly, Kuno *et al.* [70] use face identification and hand gesture recognition to control an intelligent wheelchair. The system makes an initial assumption of an appropriate direction and speed response for the wheelchair based on a best guess at the user's gesture. If the user approves of the response, it is assumed that they will repeat the gesture. This way, the chair's response is reinforced and the gesture is deemed registered for future use.

Since this paper assumes an unskilled human user who is attempting to train an assistive robot, we feel that clicker training style approach is appropriate. And, as noted by Knox *et al.* [71], a human trainer has a broader view of the benefit of a specific individual action than is considered by MDPs. Rather, the trainer may give reward based on a qualitative view of how a task *should* be performed by an agent. These authors suggest that this observation indicates that using a human teacher is more akin to a supervised learning approach. However, for simplicity and to facilitate the incorporation of other reward modalities in the future, this paper makes use of the Q-Learning method.

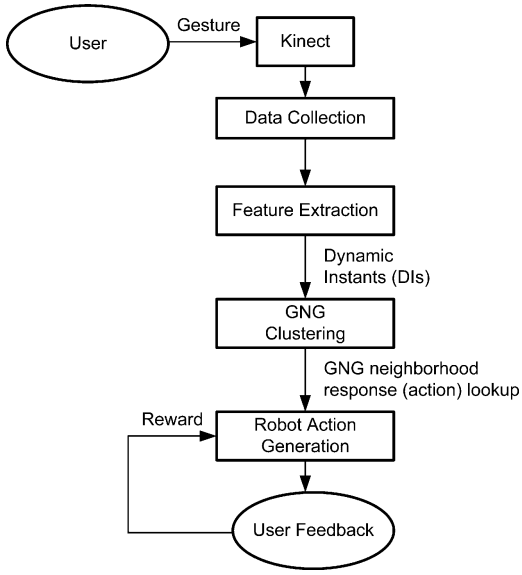


Fig. 2. System flow diagram. User feedback is automated for the experiments described here. The human user would generate the reward in the eventual implementation.

We assume that a goal configuration is known by the user, and that through gesture and a simple reward or punishment evaluation of the robot's response, the robot will eventually achieve this configuration. Certainly, higher dimensional configurations will be more challenging to attain with binary feedback. Furthermore, some trajectories toward the final configuration may allow the robot to pass through undesirable configurations. We leave the assessment of trajectory to future work. Hence, our learning algorithm essentially undertakes the problem of developing the user's goal rather than modeling the environment to obtain ever higher rewards. The shortest path to the goal is understood to yield the greatest benefit.

III. Method

This section describes the laboratory fixture used to collect gesture data, the system by which gesture samples are processed, and the response learning algorithm. Included are descriptions of modules that generate data representation, perform clustering, generate robotic response action, and issue user feedback (reward).

A. System Overview

An operational flow diagram of the system is shown in Fig 2. The system implements an end-to-end solution between the performance of gesture by a human user and the learned response to gesture by a robotic agent. Modules within the system address each of the constituent challenges described in Section II (sensing, data representation, pattern recognition, and machine learning).

Gesture samples performed by human participants are presented to the system. The samples captured using the Kinect

RGB-D sensor are processed and features (DIs) are extracted to form motion descriptors. Descriptors are applied to a GNG-based clusterer which maps the topology of the input space. A reference node within the GNG *cloud* is selected for each motion and one of its available responses is executed by the robot agent. A (simulated) human user evaluates the response and provides feedback (or *reward*). The reward is used to update the action response for future reference. This way, responses are shown to converge according to the user's preference for a given gesture type.

The approach described in this paper seeks to overcome certain limitations of prevailing methods by using nonintrusive sensing, learning with the human as teacher through a relatively short-training phase, and learning goal configurations with no prior knowledge of the user's preferences. The remainder of this section details the fixture and algorithms used in pursuit of these contributions.

B. Data Collection

Data were collected for three essential hand gestures that were deemed a baseline command set for the eventual operation of an assistive robot. Although our approach places no expectation on the user to perform gestures in a particular manner, motion models for these gestures were taken from the ASL dictionary (as demonstrated in [72]) to facilitate repeatability. The gestures included *come closer*, *go away*, and *stop*. These three gesture types were seen as a baseline essential collection of commands for an assistive robot. Although the gesture command vocabulary will be increased in future work, these were considered sufficient for this proof-of-concept research. The *stop* gesture requires special treatment since it intuitively suggests that the robot is presently executing an earlier command. Hence, the problem of gesture segmentation arises. Since segmentation is not the focus of this paper, we leave its consideration to future work. Instead, *stop* will not be interpreted in its literal sense, but rather as having a specific goal configuration similar to that of *come closer* and *go away*.

Data samples were collected using the depth sensing feature of the Microsoft Kinect RGB-D camera [25]. The Kinect generates depth maps of the user at approximately 30 frames/s. Samples were collected over 5-second intervals for a total of 150 data points per sample. The data collection program was developed using the robot operating system (ROS) [73]. ROS was selected for its open source and for its active community of research-oriented users. Furthermore, it supports a variety of simulated and real-world robotic platforms through a message-based publisher/subscriber environment. Thus, direct migration of this paper to working hardware is expected to be a viable path.

Within ROS, the Kinect data stream was accessed using the PrimeSense OpenNI Kinect package [74] to track the skeletal joints of the participant by ROS messages. Example Kinect depth images showing skeletal tracking are given in Fig. 3. Hand trajectories overlaid on the subfigures highlight the candidate gesture motions. Depth data for 11 joints were collected over the sampling interval for possible future work. However, only the participant's left-hand joint is considered for gesture characterization in this experiment. Data points consist of (x, y, z) coordinates.

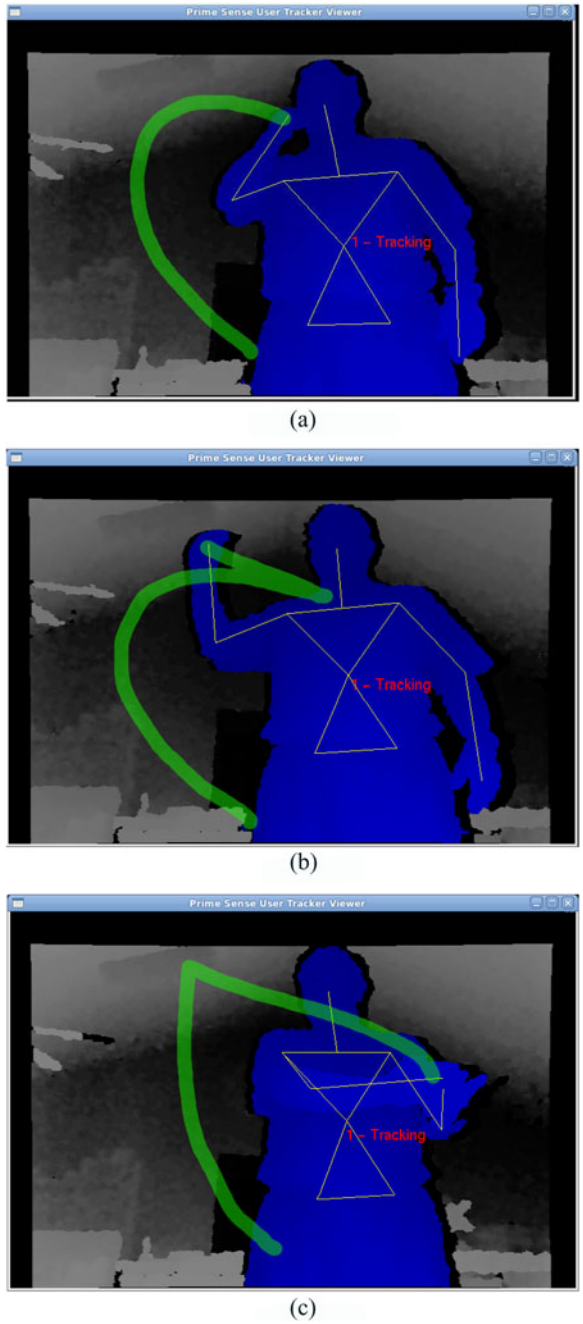


Fig. 3. PrimeSense OpenNI depth images showing skeletal tracking. Trajectories sketched in green depict the candidate gesture motions: (a) come closer, (b) go away, and (c) stop. Note that the PrimeSense interface displays the mirror image of the actor.

C. Feature Extraction

Using an approach similar to [33], DIs were extracted from each 150 point data sample for motion of the left-hand joint. Position data for each of the three dimensions were first smoothed by convolution with the discrete Gaussian kernel given by (1)

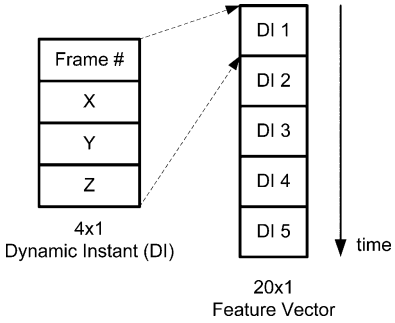


Fig. 4. Feature vector format for a depth-sampled gesture. DIs are concatenated in chronological order by the frame number.

with $\sigma^2 = 1.0$

$$G = [1, 4, 6, 4, 1]/16. \quad (1)$$

As a further smoothing step, an *evolution time* of seven time steps was applied to position data so that short-term *jitter* of the actor could be filtered and longer term trends could be captured.

Velocity and acceleration data were then computed from position data for each dimension. The five highest occurrences of peak acceleration were selected as the DIs. As discussed in [33], such peaks occur at sharp changes of direction or speed, and starts/stops. For our DIs, the (x, y, z) coordinates and the frame number were recorded. Given the Kinect's capability to represent the positions of these peaks in 3-D space and with the frame number accounting for discrete time, the spatial trajectory of gesture execution is effectively generated. Hence, DIs did not require the extra dimensions of velocity and acceleration to be stored for useful discrimination between gesture types.

Feature vectors for each sample were constructed by the concatenation of the five DIs to yield a 20×1 descriptor as shown in Fig. 4. Both frame numbers and coordinate values were scaled to $[0, 1]$, based on the range of values of their respective types so as to prevent any given field from dominating the feature vector. Feature vectors were then clustered using GNG (see below).

D. Growing Neural Gas Algorithm

The GNG algorithm proposed by Fritzke [36] is a vector quantization technique in which neurons (*nodes*) represent *codebook* vectors that encode a submanifold of input data space. In this regard, GNG is similar to the NG algorithm proposed by Martinetz and Schulten [35]. GNG differs from NG in its ability to form connections between nodes and, thus, preserves a topological representation of input space in a manner functionally similar to the SOFM. Furthermore, it is capable of adding new nodes so as to effectively map the topology of a changing input data distribution. The basic GNG algorithm is given by Algorithm 1 [36]. 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 $a_{\max} = 88$. We also impose a maximum limit of 100 nodes on the network.

In our approach, the A (node list) data structure consists of a C++ vector class of reference nodes. Each reference node carries its feature vector w, its node label, and of key importance, the response configuration (x, y, θ) for a 2-D mobile robot. Therefore, as the GNG algorithm updates the cloud of reference nodes with each input vector, the *k*NN in the GNG cloud already holds a learned robotic response (or *action* as described below) based on the history of the system. This way, the GNG algorithm avoids the task of correctly labeling the input in favor of generating a desirable response. Rather, the action associated with the node serves as its label. Using a reward signal from the user to gauge the quality of response, the algorithm attempts to improve the response outcome as it quantizes the input space.

Algorithm 1 Growing Neural Gas

1: Begin with a set A of two nodes at positions w_a and w_b in 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 a_{max}. If this leaves any nodes with no connecting edges, remove them also.
13: **if** $(numInputs \ mod \ \lambda = 0)$ **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.

E. Q-Learning

Within a reinforcement learning framework, an agent attempts to learn an optimal policy to map its set of possible states to future actions that are likely to be encouraged (or *reinforced*) through a reward signal from the environment. This way, the total reward received throughout a sequence of state-action pairs may be maximized. Typically, a table of the state-action values (*Q values*) is maintained. As the agent encounters a state, the highest valued action for that state is selected and performed. The reward signal is observed and the table is updated according

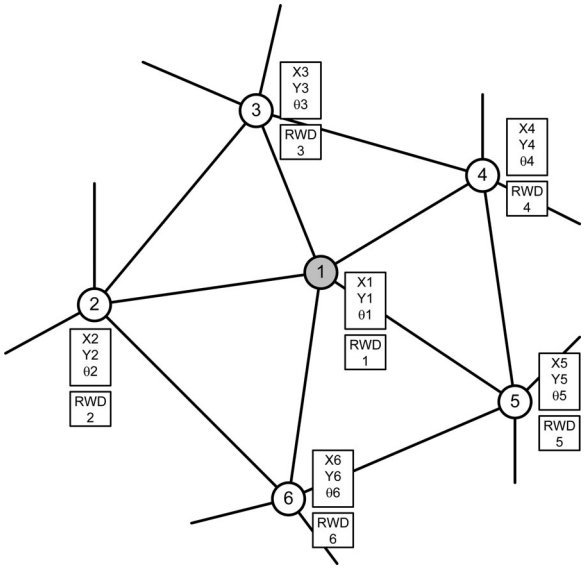


Fig. 5. Example, (2-D) GNG neighborhood with associated action vectors and most recent rewards. Gestures that fall closest to node 1 will elicit the highest valued action from nodes 1–6.

to the following equation:

$$Q_{t+1}(s,a) \leftarrow Q_t(s,a) + \alpha[r + \gamma(\max_a Q_t(s',a)) - Q_t(s,a)] \tag{2}$$

where (s,a) is a state-action pair, (s',a) 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 $r \in [-1, 0, 1]$ is the reward value. Reward 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). For our implementation, each gesture sample is followed by a training episode of a single time step. Discounting is unnecessary since each reward from the human user is equally important as evidence of movement toward or away from the goal configuration. Hence, $\gamma = 1$. With $\alpha = 1$ and multiplying the reward by a step length (*stepLen*) of linear forward progress, the update rule is reduced to the form

$$Q_{t+1}(s,a) \leftarrow r(stepLen) + \max_a Q_t(s',a)). \tag{3}$$

For this paper, $stepLen = 0.1$. This quantity is the same as the final error tolerance for the robot to achieve the goal configuration.

As previously mentioned, topological neighbors in the network may be expected to represent similar vectors in input space (sensed gestures) and should, therefore, produce similar output actions. Thus, we attempt to utilize the topology of the GNG cloud to accelerate learning by taking into account the rewards obtained by neighboring nodes (see Fig. 5). A network-structural interpretation of (3) can be stated as selection of the highest valued action vector from the neighborhood of a reference node since that vector has the richest history of positive reward. The selection and update process is given by Algorithm 2 (adapted from Touzet [64]).

Algorithm 2 Q-Learning

1: Initialize Q values of all states s (nodes) in the set of states S: $Q(s,a) = 0$.
2: Initialize action vectors for all nodes: $\{x, y, \theta\} = \{0, 0, 0\}$.
3: **repeat**
4: Apply an input signal gesture vector ξ.
5: Find node s closest to ξ.
6: Find the set of nodes N that includes s and its neighbors.
7: **if** $(r = 0)$ **then**
8: The node is fully trained. Select the associated action.
9: **else**
10: Examine past rewards $r\ \forall s \in N$.
11: **if** $(r = 1)$ for any $a \in N$ **then**
12: Select and extend an action a to be performed according to (3).
13: **else**
14: $(r = -1)$
15: Select the action (with angular correction).
16: **end if**
17: **end if**
18: Perform the action.
19: Observe and record the reward.
20: **until** Training complete: $r = 0\ \forall s \in S$

Each node in the GNG network represents a state-action tuple consisting of an input (gesture) feature vector (which is the state label), an output action vector, and a Q value. As each gesture is sensed, the GNG network is scanned for the node with the closest input vector by Euclidean distance. The set of available actions is taken from those of its topological neighbors. For our implementation, the Q value is the length of the action vector. Since action vectors pointing to locations in configuration space that are farthest from the origin must have experienced the greatest number of positively reinforced episodes, they represent actions that promise the greatest likelihood of future reward. Given this intuition, (2) becomes a simple search for the longest vector in a node's immediate neighborhood. For positive reinforcements $(r = 1)$, the node's action vector is updated with that of its highest Q-valued neighbor increased by a uniform step length along its current trajectory.

An action vector, whose reinforcement value is negative $(r = -1)$ indicates an action that would move the agent farther from the user's goal. If no neighboring node possesses a higher valued, positively reinforced action, exploration is called for and the node's action vector is updated with a small randomized angular correction. Repeated application of randomized correction will eventually yield an action that will be positively rewarded. This scenario is depicted in Fig. 6.

If the reinforcement is 0 (zero), the action vector is deemed *trained*. In our implementation, such a node's action vector is removed from consideration by its neighbors in subsequent queries. This is to avoid the possibility of assigning action

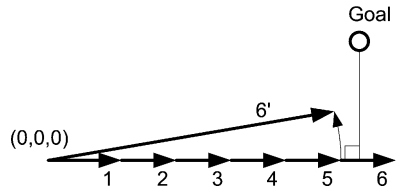


Fig. 6. Q-Learning exploration for successive approximation of actions toward a goal. Steps 1–5 receive positive reward and proceed in a consistent direction moving closer to the goal. Continuing this policy at step 6 would cause the robot to move farther from the goal than it had been at step 5 and would receive a negative reward. Random angular adjustments are attempted until the accumulated action vector comes closer to the goal as in step $6'$.

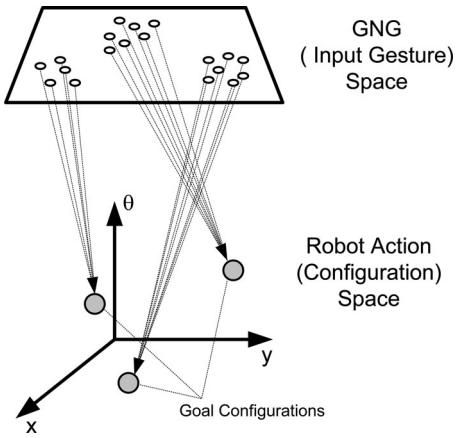


Fig. 7. Input gesture to robot action mapping. Input gestures are clustered by GNG and mapped through successive reinforcement to desired robot action vectors.

vectors repeatedly that may be incorrect for similar but different gestures. This way, the finally trained network will form an associative memory mapping between gestures and actions as shown in Fig. 7.

F. Simulation Environment

As a simulated proxy for our mobile robot, the ROS Turtlesim environment was used. Turtlesim is a basic ROS tutorial construct capable of accepting and attaining successive (x, y, θ) configuration goals. The problem of movement in higher degrees of freedom (DOF), although significant, is expected to be feasible using our approach. We consider potential issues of doing so as future work in Section VI.

G. Reward Generation

A key aspect of our approach is the use of a simple (binary) user-generated reward on the relative success of a robotic response to gesture. Reward is utilized to effectively guide online system learning in real time and with no initial training data. However, as previously stated, obtaining gesture data and the associated reward may be expensive. For this paper, generation of a reward signal was automated in the software according to predefined goal configurations. These configurations (see Table I) represent relative translations (x, y, θ) from the starting position of the robot $(0, 0, 0)$ and were chosen to be easily distinguishable.

TABLE I
GOAL CONFIGURATIONS

Gesture Type	(x, y, θ)
Come closer	$(3.95,\ 3.95,\ 45^o)$
Go away	$(3.95,\ -3.95,\ 315^o)$
Stop	$(-3.95,\ -3.95,\ 225^o)$

IV. EXPERIMENTATION

A. Data Collection

The Kinect RGB-D sensor was set at desk height (75 cm) with the participant standing at a distance of 1.3 m. The Kinect was angled so that the 11 tracked joints were fully visible in the depth image. Participants were invited to shift their weight or angle of address occasionally so as to introduce a nominal variation in the collected data. Five volunteers were asked to perform 50 repetitions for each of the three candidate gestures: *come closer*, *go away*, and *stop*. This yielded 250 samples for each gesture type for a total of 750 samples. DIs computed for each gesture type are shown in Fig. 8(a)–(c). We will refer to this as our *real* dataset so as to distinguish it from the idealized dataset discussed below. It can be seen that the *real* data are not well separated and thus, might not be expected to yield GNG neighborhoods that are readily clustered by gesture type.

For the purposes of comparison, a second (*ideal*) dataset was created based on a single exemplar gesture of each type. A collection of 750 samples was generated by the application of uniformly distributed noise within a margin of 5% of each DIs in each exemplar. DIs for the ideal dataset are shown in Fig. 8(d)–(f).

B. Neighborhood Radius

Three scenarios were performed for each dataset to explore our system's sensitivity to vary the radius of the GNG neighborhood. The respective radii dictated which nodes were considered neighbors for the purposes of action selection. These scenarios were:

1) (*Large*) All neighbors connected to the reference node by a single link were considered.
2) (*Medium*) All neighbors within the mean distance from the reference node were considered.
3) (*Small*) Only the reference node itself was considered.

C. Data Processing

For each dataset, (*real* and *ideal*), the 750 samples were randomly divided into two groups. One group consisted of 300 samples (100 of each gesture type) and was applied in *epochs* to train the GNG network to map the topology of the input space with a low degree of error. No action learning occurred during this time. Application of this first group was not a necessary step, although it facilitated smoother action learning of actions in the second step.

A second group of 450 samples (150 of each gesture type) was then applied in epochs and learning of actions was allowed to proceed. For each of the 450 samples, feature vectors were

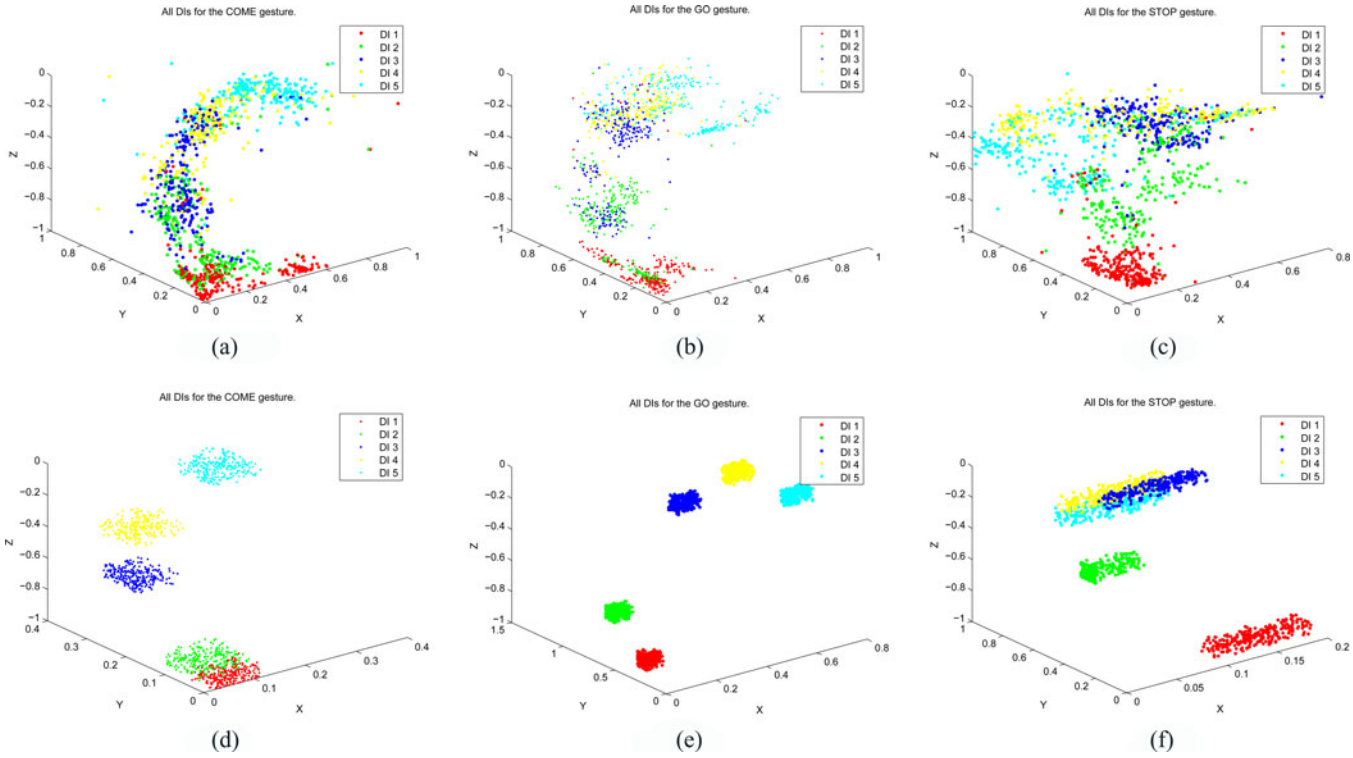


Fig. 8. DIs for *real* (a)–(c), and *ideal* (d)–(f) datasets for gestures: Come closer (a) and (d), Go away (b) and (e), and Stop (c) and (f). Note: apparent differences in data spread for *ideal* data are due to scaling within the individual plots.

computed and passed to the GNG algorithm, an action was selected and performed, reward was automatically generated, and the reference node was updated accordingly.

V. Results

The per sample error was calculated between the updated goal configuration and the known goal for that sample's gesture type. Following each epoch, the average error per gesture type was also computed. In this manner, 250 epochs were executed. This process was repeated for each of the three neighborhood radius scenarios. Resulting average error curves for the real and ideal datasets using GNG and for the real dataset using *k*NN are shown in Fig. 9.

As previously stated, once the reward signal is given a value of 0 (zero), the near reference node effectively becomes an associative memory entry, capable of producing a generalized outcome for all inputs that fall within its receptive field. Once trained, the GNG-to-action mapping strategy of our system performed well. Again, we employ TurtleSim to fully *close the loop* between a gestured command from a human user and a final learned robotic actuation. Furthermore, the relative size of the TurtleSim utility matches that of our envisioned environment. Hence, the scale and accuracy of the generated actions might help the reader to better visualize the effectivess of our approach. Typical learned action trajectories can be seen in Fig. 10.

Results shown in Fig. 9 are typical with the exception of Fig. 9(c). This scenario is susceptible to erratic convergence due to randomizations that occur during exploration compounded by its larger neighborhood of possibility. For the well-separated ideal dataset, convergence occurs quickly and the speed of convergence varies directly with neighborhood size. Conversely, for the poorly separated *real* dataset, convergence is slower and varies inversely with neighborhood size. However, use of GNG routinely showed faster convergence over the *k*NN algorithm for similar sized neighborhoods of consideration as can be seen in Fig. 9. It is noted that this approach may see slower learning rates and higher DOF. Previous work by the researchers [75] with 1-D goals and nonbinary (*rich*) reward showed convergence in approximately one-tenth the number of iterations.

TABLE II
Average Cumulative Error for 100 Runs

Neighborhood Radius	Real Data	Ideal Data
Large	2124.34	584.14
Medium	977.49	615.66
Small	825.25	743.67

The *come closer* gesture is qualitatively different in choreography than the other two gestures used here. It is performed primarily in a plane that is approximately perpendicular to the image plane of the RGB-D sensor, while *go away* and *stop* are performed in a parallel plane. Because of this dissimilarity, more rapid convergence for *come closer* is frequency observed, although exploration by our learning algorithm may occasionally elicit other outcomes.

Because the exploratory aspects of Q-Learning utilize randomization to choose possible future outcomes, the general relationships shown may fail to hold from run to run. Table II

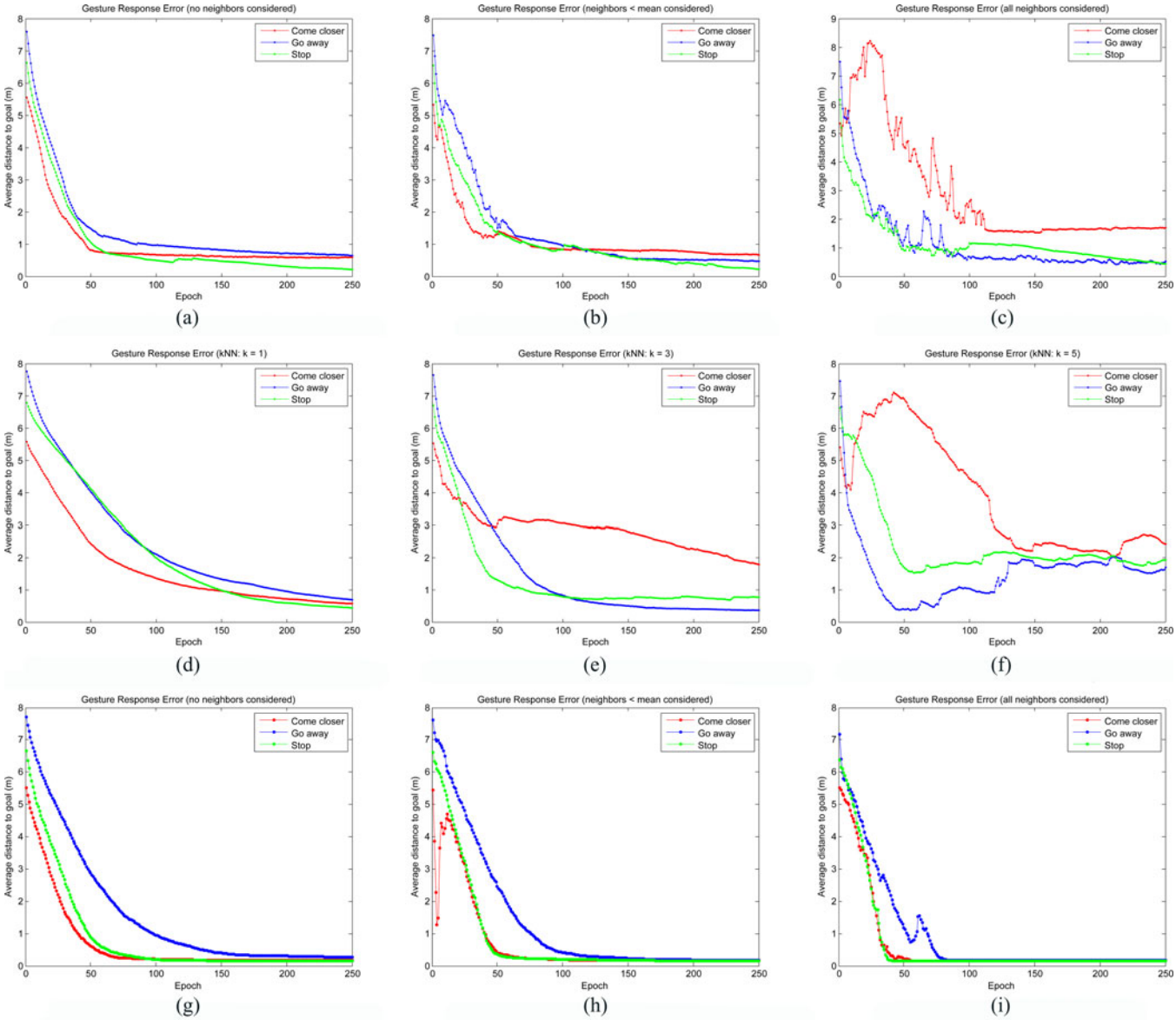


Fig. 9. Average error curves for varying neighborhood radii. Each row of subfigures shows results for a given scenario: (a)–(c) show results for the *real* dataset using GNG; (d)–(f) show results for the *real* dataset using kNN; (g)–(i) show results for the *ideal* dataset using GNG. Within each row, neighborhood radius increases from left to right. It can be seen that for poorly separated (*real*) data, GNG converges more rapidly than *k*NN for all neighborhood radii. Furthermore, with well separated (*ideal*) data, the performance of GNG improves with increasing neighborhood size. (a) Real data, small neighborhood. (b) Real data, medium neighborhood. (c) Real data, large neighborhood. (d) Real data, small ($k = 1$) neighborhood. (e) Real data, medium ($k = 3$) neighborhood. (f) Real data, large ($k = 5$) neighborhood. (g) Ideal data, small neighborhood. (h) Ideal data, medium neighborhood. (i) Ideal data, large neighborhood.

shows average outcomes for 100 such runs. In general, the expected outcomes are qualitatively produced. Again, the data show that, for well-separated data, the GNG algorithm allows faster learning with increasing neighborhood size. For poorly separated data, the inverse relationship is seen.

VI. Conclusions and Future Work

In this paper, we have presented a new approach toward the development of a gesture-based human–machine interface. An end-to-end approach is presented which maps arm-scale gesture by a human user to a learned response by a robotic agent through repeated applications of user-provided reward. Between these two end points, the constituent challenges are addressed in the areas of sensor selection, data representation, pattern recognition, and machine learning. As a composite approach, the proposed system overcomes many of the shortcomings of previous efforts.

It has been shown that 3-D data from an RGB-D camera can be used to generate a useful descriptor of gesture in the form of prominent DIs. Furthermore, the GNG algorithm utilization herein is capable of differentiating between these descriptors more effectively than a conventional *k*NN approach. Additionally, although Q-Learning typically requires a large number of iterations to produce effective results, our value function and GNG generalization technique shows favorable results relatively

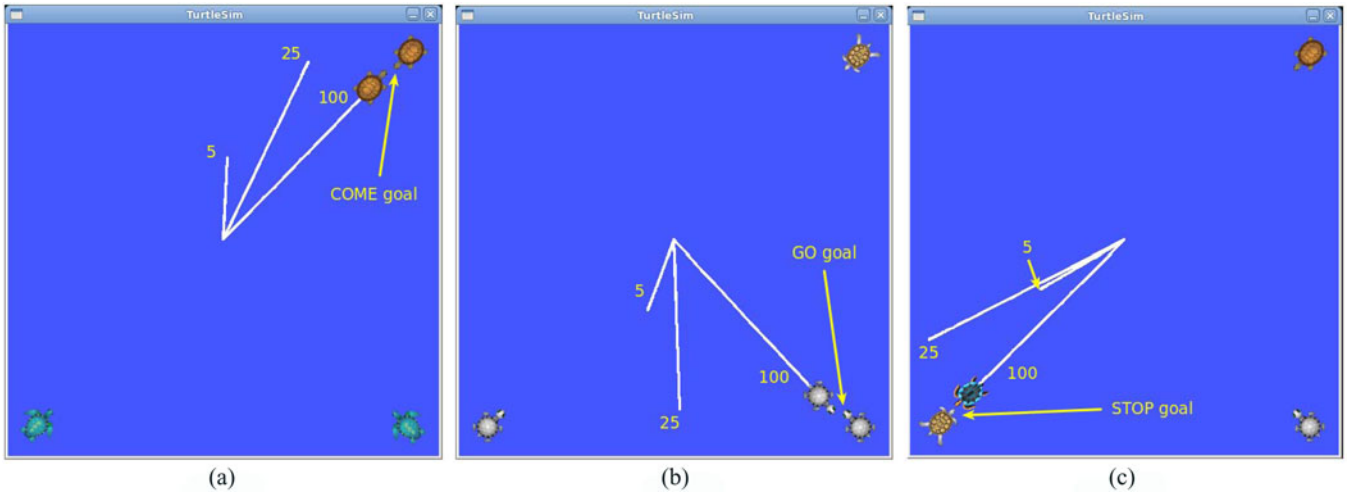


Fig. 10. Learned action trajectories in TurtleSim for gestures: (a) come closer, (b) go away, and (c) stop. Markers placed in upper right and bottom corners represent goal positions as shown. Trajectory development after 5, 25, and 100 epochs is depicted by white trace lines. As the GNG-to-action mapping data structure matures, learned trajectories and final angles of approach are seen to be progressively more accurate in the attainment of goal configurations.

quickly and with a small dataset. We envision a use model in which the user performs a single gesture and provides repeated reward input so as to allow a GNG reference node to train fully. Our near-term investigations will examine the possibility of using this approach to further reduce the number of gesture motion samples that are required.

The problem of achieving goal configurations in more than three DOF is seen as feasible using the proposed approach. Each DOF could be isolated so that movement could proceed along each one in turn until such motion begins to increase (rather than decrease) overall error toward the goal. However, it is foreseen that this process would be difficult for an actual human trainer to entertain, since visualization of a robot's configuration as a Cartesian space may be difficult, if not impossible, for higher dimensions. Certainly, a path-planning component would be called for which considers the robot's configuration in light of the geometry of the environment and the social sensibilities of the user (speed, angle of approach, visibility, etc.). As it is implemented here, the proposed approach is seen as practical for mapping of a robot's end effector and, thus, useful for common applications.

Segmentation of gestures (*gesture spotting*) is a typical problem in gesture recognition. Although our future work will address spotting, our focus here has been on validation of a real-time learning technique that produces desirable outcomes using a human teacher and a simple, binary reward signal.

Finally, online learning of new gestures will be explored. Certainly, for the envisioned system to effectively assist the user, the vocabulary of known commands must be open to amendment as needed.

REFERENCES

[1] 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," *J. Amer. Geriatr. Soc.*, vol. 51, no. 4, pp. 451–458, 2003.

[2] 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," *Int. J. Circuits, Syst. Signal Process.*, no. 5, pp. 279–286, 2011.

[3] A. Friedman, *The Adaptable House: Designing Homes for Change*. New York, NY, USA: McGraw-Hill, 2002.

[4] S. Iwarsson and A. Stahl, "Accessibility, usability and universal design—Positioning and definition of concepts describing person-environment relationships," *Disabil. Rehabil.*, vol. 25, no. 2, pp. 57–66, 2003.

[5] W. Preiser and E. Ostroff, *Universal Design Handbook*. New York, NY, USA: McGraw-Hill, 2001.

[6] D. Cook, M. Youngblood, E. Heierman, K. Gopalratnam, S. Rao, A. Litvin, and F. Khawaja, "MavHome: An agent-based smart home," in *Proc. IEEE Int. Conf. Pervas. Comput. Commun.*, 2003, pp. 521–524.

[7] B. DeRuyter and E. Pelgrim, "Ambient Assisted-Living Research in CareLab," *Interactions*, vol. XIV, no. 4, pp. 30–34, 2007.

[8] S. Intille, K. Larson, and E. Tapia, "Designing and evaluating technology for independent aging in the home," in *Proc. Int. Conf. Aging, Disabil. Independ.*, 2003, pp. 1–15.

[9] C. Kidd, R. Orr, G. Abowd, C. Atkeson, I. Essa, B. MacIntyre, E. Mynatt, T. Starner, and W. Newstetter, *The Aware Home: A Living Laboratory for Ubiquitous Computing Research*. New York, NY, USA: Springer-Verlag, 1999, pp. 191–198.

[10] I. Walker and K. Green, "Architectural Robotics: Unpacking the Humanoid," presented at the *ARCHIBOTS Worksh. Architect. Robot.*, Orlando, FL, USA, 2009.

[11] 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, and P. Srikanth, P. Xu, "Robotic alternatives for bedside environments in healthcare," *Int. J. Syst. Appl., Eng. Develop.*, vol. 6, no. 3, pp. 308–316, 2012.

[12] A. Threatt, J. Merino, K. Green, I. Walker, J. Brooks, S. Ficht, R. Kriener, M. Mossey, A. Mutlu, D. Salvi, G. Schafer, S. Pallavi, P. Xu, J. Manganelli, and P. Yanik, "A vision of the patient room as an architectural-robotic ecosystem," presented at the *IEEE/RSJ Int. Conf. Robots Syst.*, Vila Moura, Algarve, Portugal, 2012.

[13] I. Walker, J. Brooks, K. Green, J. Manganelli, L. Smolentzov, A. Threatt, P. Yanik, and J. Merino, "Interactive robotic environments in healthcare," in *Proc. Workshop Interact. Syst. Healthcare*, 2011, pp. 1–5.

[14] S. Mitra and T. Acharya, "Gesture recognition: A survey," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 37, no. 3, pp. 311–324, May 2007.

[15] S. Jin, Y. Li, G. Lu, J. Luo, W. Chen, and X. Zheng, "Som-based hand gesture recognition for virtual interactions," in *Proc. IEEE Int. Symp. VR Innovat.*, Mar. 2011, pp. 317–322.

[16] J. Lementec and P. Bajcsy, "Recognition of arm gestures using multiple orientation sensors: Gesture classification," in *Proc. IEEE 7th Int. Conf. Intell. Transport. Syst.*, Oct. 2004, pp. 965–970.

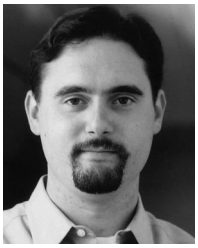
[17] S. Zhou, Q. Shan, F. Fei, W. Li, C. Kwong, P. Wu, B. Meng, C. Chan, and J. Liou, "Gesture recognition for interactive controllers using MEMS

motion sensors," in *Proc. IEEE 4th Int. Conf. Nano/Micro Eng. Molecul. Syst.*, Jan. 2009, pp. 935–940.
[18] Y. Yamazaki, H. Vu, P. Le, Z. Liu, C. Fatichah, M. Dai, H. Oikawa, D. Masano, O. Thet, Y. Tang, N. Nagashima, M. L. Tangel, F. Dong, and K. Hirota, "Gesture recognition using combination of acceleration sensor and images for casual communication between robots and humans," in *Proc. IEEE Congr. Evolution. Comput. (CEC)*, Jul. 2010, pp. 1–7.
[19] C. Zhu and W. Sheng, "Wearable sensor-based hand gesture and daily activity recognition for robot-assisted living," *IEEE Trans. Syst., Man Cybern. A, Syst. Humans*, vol. 41, no. 3, pp. 569–573, May 2011.
[20] H. Cheng, A. Chen, A. Razdan, and E. Buller, "Contactless gesture recognition system using proximity sensors," in *Proc. IEEE Int. Conf. Consum. Electron.*, Jan. 2011, pp. 149–150.
[21] R. Dongseok, U. Dugan, P. Tanofsky, H. Do, S. Young, and K. Sungchul, "T-less: A novel touchless human-machine interface based on infrared proximity sensing," in *Proc. IEEE/RJS Int. Conf. Intell. Robots Syst.*, Oct. 2010, pp. 5220–5225.
[22] S. Berman and H. Stern, "Sensors for gesture recognition systems," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 42, no. 3, pp. 277–290, May 2012.
[23] 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 Trans. Access. Comput.*, vol. 2, no. 1, pp. 1–21, 2009.
[24] G. Demeris, B. Hensel, M. Skubic, and M. Rantz, "Senior residents' perceived need of and preferences for "smart home" sensor technologies," *Int. J. Technol. Assessment Health Care*, vol. 24, pp. 120–124, 2008.
[25] Microsoft Xbox 360 + Kinect Website, (2013). [Online]. Available: http://www.xbox.com/en-US/kinect
[26] C. BenAbdelkader, R. Cutler, and L. Davis, "Gait recognition using image self-similarity," *EURASIP J. Appl. Signal Process.*, vol. 2004, pp. 572–585, 2004.
[27] A. Bobick, "Movement, activity and action: The role of knowledge in the perception of motion," *Philosoph. Trans. Roy. Soc. B: Biol. Sci.*, vol. 352, no. 1358, pp. 1257–1266, 1997.
[28] A. Karahoca and M. Nurullahoglu, "Human motion analysis and action recognition," in *Proc. 1st WSEAS Int. Conf. Multivariate Anal. Appl. Sci. Eng.*, 2008, pp. 156–161.
[29] N. Dalal, B. Triggs, I. Rhone-Alps, and F. Montbonnot, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recog.*, 2005, pp. 886–893.
[30] G. Johansson, "Visual perception of biological motion and a model for its analysis," *Percept. Psychophys.*, vol. 14, no. 2, pp. 201–211, 1973.
[31] R. Cutler and L. Davis, "Robust real-time periodic motion detection, analysis, and applications," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 781–796, Aug. 2000.
[32] I. Junejo, E. Dexter, I. Laptev, and P. Pérez, "Cross-view action recognition from temporal self-similarities," in *Proc. Eur. Conf. Comput. Vis.*, 2008, pp. 293–306.
[33] C. Rao, A. Yilmaz, and M. Shah, "View-invariant representation and recognition of actions," *Int. J. Comput. Vis.*, vol. 50, no. 2, pp. 203–226, 2002.
[34] S. Grossberg, "Adaptive pattern classification and universal recoding: I. Parallel development and coding of neural feature detectors," *Biolog. Cybern.*, vol. 23, no. 3, pp. 121–134, 1976.
[35] T. Martinetz and K. Schulten, in *A "Neural-Gas" Network Learns Topologies*, T. Kohonen, K. Makisara, O. Simula, and J. Kangas, Eds. Amsterdam, The Netherlands: Elsevier, 1991.
[36] B. Fritzke, "A growing neural gas network learns topologies," *Adv. Neural Inf. Process. Syst. 7*, vol. 7, pp. 625–632, 1995.
[37] A. Wilson and A. Bobick, "Realtime online adaptive gesture recognition," in *Proc. IEEE 15th Int. Conf. Pattern Recog.*, 2000, vol. 1, pp. 270–275.
[38] J. Yamato, J. Ohya, and K. Ishii, "Recognizing human action in time-sequential images using hidden Markov model," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recog.*, 1992, pp. 379–385.
[39] M. Moni and A. Ali, "HMM based hand gesture recognition: A review on techniques and approaches," in *Proc. IEEE Int. Conf. Comput. Sci. Inf. Technol.*, 2009, pp. 433–437.
[40] G. Fang, W. Gao, and J. Ma, "Signer-independent sign language recognition based on SOFM/HMM," in *Proc. IEEE Workshop Recog., Anal., Track. Faces Gest. Real-Time Syst.*, 2001, pp. 90–95.
[41] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Trans. Acoust., Speech Signal Process.*, vol. 37, no. 3, pp. 328–339, Mar. 1989.
[42] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989.
[43] A. Bobick and A. Wilson, "A state-based approach to the representation and recognition of gesture," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 12, pp. 1325–1337, Dec. 1997.
[44] S. Lee, "Automatic gesture recognition for intelligent human-robot interaction," in *Proc. IEEE 7th Int. Conf. Autom. Face Gest. Recog.*, 2006, pp. 645–650.
[45] D. Frolova, H. Stern, and S. Berman, "Most probable longest common subsequence for recognition of gesture character input," *IEEE Trans. Cybern.*, vol. 43, no. 3, pp. 871–880, Jun. 2013.
[46] J. Prasad and G. Nandi, "Clustering method evaluation for hidden Markov model based real-time gesture recognition," in *Proc. Int. Conf. Adv. Recent Technol. Commun. Comput.*, Oct. 2009, pp. 419–423.
[47] T. Schlömer, B. Poppinga, N. Henze, and S. Boll, "Gesture recognition with a Wii controller," in *Proc. 2nd Int. Conf. Tangible Embedded Interact.*, 2008, pp. 11–14.
[48] J. Wachs, U. Kartoun, H. Stern, and Y. Edan, "Real-time hand gesture telerobotic system using fuzzy c-means clustering," in *Proc. IEEE 5th Biannu. World Autom. Congr.*, 2002, vol. 13, pp. 403–409.
[49] W. Knox, "Learning from human-generated reward," Ph.D. dissertation, Dept. Comput. Sci., Univ. Texas at Austin, Austin, TX, USA, 2012.
[50] A. Varkonyi-Koczy and B. Tusor, "Human–computer interaction for smart environment applications using fuzzy hand posture and gesture models," *IEEE Trans. Instrum. Meas.*, vol. 60, no. 5, pp. 1505–1514, May 2011.
[51] M. Yang and N. Ahuja, "Recognizing hand gesture using motion trajectories," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recog.*, Jun. 1999, vol. 1, pp. 466–472.
[52] T. Alexander, H. Ahmed, and G. Anagnostopoulos, "An open source framework for real-time, incremental, static and dynamic hand gesture learning and recognition," *Human-Comput. Interact. Novel Interact. Methods Tech.*, vol. 5611, pp. 123–130, 2009.
[53] E. Stergiopoulou and N. Papamarkos, "A new technique for hand gesture recognition," in *Proc. Int. Conf. Image Process.*, Oct. 2006, pp. 2657–2660.
[54] A. Angelopoulou, A. Psarrou, J. Garcia-Rodriguez, and G. Gupta, "Tracking gestures using a probabilistic self-organising network," in *Proc. Int. Joint Conf. Neural Netw.*, Jul. 2010, pp. 1–7.
[55] J. Holmström, "Growing neural gas: Experiments with GNG, GNG with utility and supervised GNG," Master's thesis, Dept. Inf. Technol., Uppsala Univ., Uppsala, Sweden, 2002.
[56] H. Ritter, T. Martinetz, and K. Schulten, "Topology-conserving maps for learning visuo-motor-coordination," *Neural Netw.*, vol. 2, no. 3, pp. 159–168, 1989.
[57] T. Martinetz, H. Ritter, and K. Schulten, "Three-dimensional neural net for learning visuomotor coordination of a robot arm," *IEEE Trans. Neural Netw.*, vol. 1, no. 1, pp. 131–136, Mar. 1990.
[58] T. Kohonen, "The self-organizing map," *Proc. IEEE*, vol. 78, no. 9, pp. 1464–1480, Sep. 1990.
[59] J. Walter and K. Schulten, "Implementation of self-organizing neural networks for visuo-motor control of an industrial robot," *IEEE Trans. Neural Netw.*, vol. 4, no. 1, pp. 86–96, Jan. 1993.
[60] G. Barreto, A. Araújo, and H. Ritter, "Self-organizing feature maps for modeling and control of robotic manipulators," *J. Intell. Robot. Syst.*, vol. 36, no. 4, pp. 407–450, 2003.
[61] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*, vol. 1. Cambridge, U.K.: Cambridge Univ. Press, 1998, no. 1.
[62] C. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, no. 3, pp. 279–292, 1992.
[63] C. Touzet, 2003.
[64] C. Touzet, "Neural reinforcement learning for behaviour synthesis," *Robot. Auton. Syst.*, vol. 22, no. 3, pp. 251–281, 1997.
[65] W. Knox and P. Stone, "Tamer: Training an agent manually via evaluative reinforcement," in *Proc. IEEE 7th Int. Conf. Develop. Learn.*, 2008, pp. 292–297.
[66] F. Kaplan, P. Oudeyer, E. Kubinyi, and A. Miklósi, "Robotic clicker training," *Robot. Auton. Syst.*, vol. 38, no. 3, pp. 197–206, 2002.
[67] B. Blumberg, M. Downie, Y. Ivanov, M. Berlin, M. Johnson, and B. Tomlinson, "Integrated learning for interactive synthetic characters," *ACM Trans. Graphics*, vol. 21, no. 3, pp. 417–426, 2002.
[68] A. Thomaz and C. Breazeal, "Teachable robots: Understanding human teaching behavior to build more effective robot learners," *Artif. Intell.*, vol. 172, no. 6–7, pp. 716–737, 2008.
[69] U. Kartoun, H. Stern, and Y. Edan, "A human-robot collaborative reinforcement learning algorithm," *J. Intell. Robot. Syst.*, vol. 60, no. 2, pp. 217–239, 2010.

[70] Y. Kuno, T. Murashima, N. Shimada, and Y. Shirai, "Interactive gesture interface for intelligent wheelchairs," in *Proc. IEEE Int. Conf. Multimedia Expo.*, 2000, vol. 2, pp. 789–792.
[71] W. Knox, I. Fasel, and P. Stone, "Design principles for creating human-shapable agents," in *Proc. AAAI Spring 2009 Symp. Agents Learn Human Teachers*, 2009, pp. 79–86.
[72] ASL Pro Website, (2013). [Online]. Available: http://www.aslpro.com/cgi-bin/aslpro/aslpro.cgi
[73] ROS Website, (2013). [Online]. Available: http://www.ros.org
[74] ROS OpenNI Tracker Website, (2013). [Online]. Available: http://ros.org/wiki/openni_tracker
[75] P. Yanik, J. Manganelli, J. Merino, A. Threatt, J. Brooks, K. Green, and I. Walker, "Use of kinect depth data and growing neural gas for gesture based robot control," in *Proc. 6th Int. Conf. Pervas. Comput. Technol. Healthcare*, La Jolla, CA, USA, 2012, pp. 283–290.

Paul M. Yanik (SM'05) received the B.S.E.E. and M.S. degrees in computer engineering from North Carolina State University, Raleigh, NC, USA, in 1989 and 1995, respectively, and the Ph.D. degree in computer engineering from Clemson University, Clemson, SC, USA, in 2013.

He is currently an Assistant Professor with the Department of Electrical and Computer Engineering Technology, Western Carolina University, Cullowhee, NC, USA. His research interests include robotics, machine learning and pattern recognition.



Joe Manganelli received the Ph.D. degree in planning, design, and the built environment (PDBE) from Clemson University, Clemson, SC, USA, in 2013.

Prior to graduate school, he spent six years in architectural practice, focusing on K-12, university buildings, and industrial architecture.

Jessica Merino received the B.S.E.E. and M.S. degrees in electrical engineering from Clemson University, Clemson, SC, USA.

She is currently a systems engineer at Disney Parks and Resorts in Orlando, FL, USA. Her research interests include intelligent systems and continuum robotic surfaces.

Anthony L. Threatt received the B.A. degree in architecture and the Ph.D. degree in planning, design, and the built environment (PDBE) from Clemson University, Clemson, SC, USA, in 2004 and 2013, respectively, and the M.Arch degree from Louisiana State University, Baton Rouge, LA, USA, in 2008.

He is currently a Postdoctoral Research Fellow in the Center for Research and Innovation in Systems Safety at Vanderbilt University Medical Center, Nashville, TN, USA. He conducts research in human factors focusing on usability, clinical decision making, and human-centered design of health information technology.

Johnell O. Brooks received the B.A. degree in psychology, the M.S. degree in human factors psychology, and the Ph.D. degree in industrial-organizational psychology from Clemson University, Clemson, SC, USA, in 1998, 2002, and 2005, respectively.

She is currently an Assistant Professor in the Department of Psychology, Clemson University. She conducts research in human factors and studies aging drivers within the driving and living environments.

Keith Evan Green (M'08) received the B.A. degree in experimental Psychology from the University of Pennsylvania, Philadelphia, PA, USA, the Master's degree in architecture from the University of Illinois at Chicago, Chicago, IL, USA, and the Ph.D. degree in architecture from the University of Pennsylvania.

He is currently a Professor with the Department of Architecture and Electrical and Computer Engineering, and serves as the Director of the Clemson University Institute for Intelligent Materials, Systems and Environments [iMSE] (www.CU-iMSE.org), a novel research unit partnering Architecture, Materials Science and Engineering, and Electrical and Computer Engineering. His research interests include developing, prototyping, and testing intelligent "architectural robotics." He is also an award-winning practicing Architect.

Ian D. Walker (F'06) received the B.Sc. degree in mathematics from the University of Hull, Hull, U.K., and the M.S. and Ph.D. degrees in electrical engineering from the University of Texas at Austin, Austin, TX, USA, in 1983, 1985, and 1989, respectively.

He is currently a Professor with the Department of Electrical and Computer Engineering, Clemson University, Clemson, SC, USA. His research interests are in robotics, particularly kinematically redundant robots, robot reliability and fault detection, and biologically inspired robots.