Use of Kinect Depth Data and Growing Neural Gas for Gesture Based Robot Control

Paul M. Yanik[1], Joe Manganelli[2], Jessica Merino[1], Anthony L. Threatt[2],
Johnell O. Brooks[3], Keith E. Green[2], Ian D. Walker[1]
[1]Department of Electrical and Computer Engineering
[2]School of Architecture
[3]Department of Psychology
Clemson University
Clemson, South Carolina, USA 29634
{pyanik, jmanganelli, jmerino, threatt, jobrook, kegreen, iwalker}@clemson.edu

***Abstract*—Recognition of human gestures is an active area of research integral to the development of intuitive human-machine interfaces for ubiquitous computing and assistive robotics. In particular, such systems are key to effective environmental designs which 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. The robotic response 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 which places no initial constraints on the user to perform gestures in a specific way. Skeletal depth data collected using the Microsoft Kinect sensor is clustered by GNG and used to refine a robotic response associated with the selected GNG reference node. We envision a supervised learning paradigm similar to the training of a service animal in which the response of the robot is seen to converge upon the user's desired response by taking user feedback into account. This paper presents initial results which show that GNG effectively differentiates between gestured commands and that, using automated (policy based) feedback, the system provides improved responses over time.**

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 [13]. 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 [47].

Heretofore, architects and environmental designers have attempted to accommodate those with physical impairment through the use of Universal Design Principles (UDP) 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 [18],[23],[37] 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 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 [12],[17],[22],[28]. These efforts are mostly focused on building systems since the real-time processing of occupant activity has historically been expensive and often relies on technologies considered intrusive of people's privacy (e.g. cameras). As a result, most occupant sensing in smart homes remains of low fidelity. Smart home technology would benefit greatly from the capacity to sense and interpret motion anonymously and with high fidelity. In particular, this would facilitate development of robotic components to actively support user need while preserving privacy.

Effectively implemented, such robotic components would allow for learned inference of user action and intention through persistent monitoring. Further, 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 [47]. Toward this goal, this paper presents initial research into the use of arm-scale gesture as a possible basis for a command vocabulary allowing human-machine interaction that is effective, extensible and familiar to the user.

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 which may replace words during speech. Of these, hand and arm gesticulation account for some 90% of gestured communication [34]. 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

PervasiveHealth 2012, May 21-24, San Diego, United States

DOI 10.4108/icst.pervasivehealth.2012.248610

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 which are instrumented with magnetic field tracking devices or accelerometers, or vision-based techniques involving one or more cameras [34]. Still other approaches involve IR motion or proximity sensors.

Jin et al. [24] use a glove-based orientation sensor to extract static hand positions to be used as commands. Lementec and Bajcsy [32] use wearable (arm) orientation sensors for sensing arm gesture models composed of Euler angles. These are intended for use in an unmanned aerial vehicle (UAV) and implemented as a lab simulation. Zhou et al. [48] use MEMS accelerometer data to characterize hand motions including *up*, *down*, *left*, *right*, *tick*, *circle* and *cross*. Wearable sensors are also used in [45], [48], [49], and others. Typically, however, the usefulness of wearable devices for measuring 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 [34].

IR proximity sensors are used by Cheng et al. [10] to create a reliable gesture recognition system for a touchless mobile device interface. The method uses the pair-wise 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*. Rhy et al. [39] 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. [47], 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 of cameras. Vision based approaches may suffer from disadvantages associated with latency, occlusion, or lighting. Further, since most video sequences represent a 3D to 2D projection, a loss of information is inherent in the processing of data [34]. Also, 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 [6],[16].

With the limitations of these various sensor types in mind, the research reported in this paper utilizes the Microsoft Kinect depth sensing system [2]. The Kinect provides a rich, real-time, 3D 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 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 [7]. In general, however, the problem of data representation is one of feature selection. Some vector of characterizing numerical features is selected and applied to a classifier.

Motion History Images (MHI) have been used to form a visual template of motion that preserves directional information [8], [27] . Histograms of Oriented Gradients (HOGs) are used in [15] to generate regional descriptors of still 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 [25]. This phenomenon is exploited by Benabdelkader et al. [7], and Cutler and Davis [14] 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 non-periodic motions in [26] and [47]. Results described in these works show that recurrences in 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 advanced by Rao et al. [38] to three dimensions. Dynamic instants are described as the extrema (or discontinuities) of acceleration in an actor's motion. The Kinect allows us to directly extract a third dimension rather than working with typical 2D video. We form our representation using the five most significant dynamic instants in (x, y, z) space along with their frame number over a 5 second 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 methods have been introduced to such time series data including Hidden Markov Models (HMM), Principal Component Analysis (PCA), Finite State Machines (FSM), clustering techniques such as Nearest Neighbor (*k*NN) and C-means, and various types of artificial neural networks including Multilayer Perceptron (MLP) networks, Time Delay Neural Networks

(TDNN) [34], Neural Gas (NG) [33], and Growing Neural Gas (GNG) [19].

Hidden Markov models have well established success in the classification of gestures and of generalized motion and are used in numerous research efforts. Notably, these include [43], [44] and others. A survey of such approaches can be found in [35]. The authors note that HMM approaches may inaccurately assume that observation parameters may be approximated by a mixture of Gaussian densities. Further, HMMs often have poorer discriminative outcomes than neural networks.

Bobick and Wilson [9] use finite state machines to classify gestures collected from video images. Lee et al. [31] seek to classify video motion sequences as whole-body gestures by mapping sequences of estimated poses to gestures. PCA is used for visualization; EM-based (Expected Maximum) Gaussian Mixture Model is used for clustering of poses. Prasad and Nandi [36] explore the effectiveness of several methods for vectorizing and clustering gesture motion data including: hierarchical, mean shift, k-means, fuzzy c-means and Gaussian mixture. Schlömer et al. [40] use k-means to determine clusters in basic hand/arm gestures generated using a wiimote controller including *square*, *circle*, *roll*, *Z*, and *tennis swing*.

Zhu and Sheng [49] use wearable sensors to detect both hand gestures and simple Activities of Daily Living (ADLs). Neural networks are used for gesture spotting. HMMs are used for classification. Varkonyi-Koczy and Tusor [42] use Circular Fuzzy Neural Networks (CFNN) 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 [46] use Time Delay Neural Networks (TDNN) to classify sequences of motion trajectories in hand motion for American Sign Language (ASL).

Stergiopoulou and Papamarkos [41] use GNG to model the topology of the hand itself (rather than more abstract features of the scene) in various finger-extended postures. 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. [5] present a probabilistic growing neural gas (A-GNG) method for tracking 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. In this way, the method is mainly that of a static analysis of hand shape.

The GNG algorithm [19] is a variant of the self-organizing feature map. Because it is capable of tracking a moving distribution [21], adding new reference nodes, and operating from static input parameters, it is well suited to the task of gesture recognition where no labelled data is available. Indeed, since the acquisition of gesture data is often expensive, such a technique which learns online is particularly desirable. Further, 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 subsection II-C may be broadly categorized as machine learning methods, our use of this term applies to the mechanism by which some manner of feedback is used to improve future outcomes. Typically, such a mechanism implies the use of training data to refine the classifier of choice off line as with conventional neural networks. However, a goal of this research is to create an online learning modality that utilizes direct interaction with the user so that a robot agent converges upon a desirable configuration.

Reinforcement learning approaches are frequently applied to such problems. In these, a learning algorithm applies an action *policy* and attempts to maximize a *reward* function. Higher level approaches utilize a *value* function which attempts to maximize long term reward despite a possible short term disadvantage. Conn and Peters use supervised remote control of a mobile robot to generate a goal seeking policy [11]. Gaskett, et al. [20] implement a wandering behavior in a mobile robot to seek interesting objects in unknown environments. A recent survey of other notable reinforcement learning approaches to robotics problems is given in [29].

Kuno et al. [30] 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. In this way, the chair's response is reinforced and the gesture is deemed registered for future use. Our initial reinforcement policy definition approximates this approach as described in a later section.

These works suggest that user generated feedback is useful in guiding the response of a robotic system. They further support our efforts to avoid gesture classification and instead apply the outcome of clustering directly to the refinement of response. The outlook of this research is thus oriented toward desirability of the generated response and no effort is made to correctly label the sensed gesture.

III. Method

This section describes the laboratory fixture used to collect gesture data as well as the data representation, clustering technique, generation of robotic response, and user feedback. An operational flow diagram of the system is given by Figure 1. Data were collected for three essential hand gestures which 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 any particular manner, motion models for these gestures were taken from the American Sign Language Dictionary (as demonstrated at

[1]) to facilitate repeatability. The gestures included *come closer*, *go away* and *stop*. The *stop* gesture requires special consideration since it intuitively suggests that the robot is presently executing an earlier command. Hence, the problem of gesture segmentation arises. Because segmentation is a significant unsolved problem in gesture recognition, we leave it 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*.

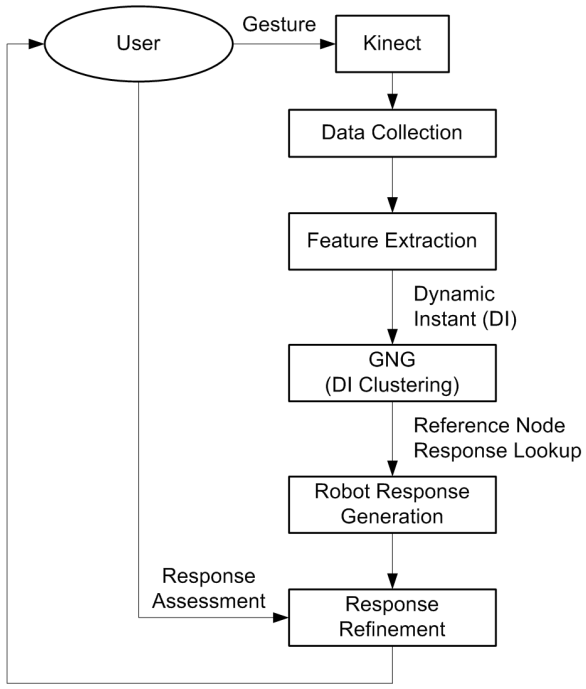


Fig. 1. System flow diagram.

A. Data Collection

Data samples were collected using the Microsoft Kinect depth sensor [2]. The Kinect generates depth maps of the user at approximately thirty frames per second. Samples were collected over five second intervals for a total of 150 data points per sample. The data collection program was developed using the Robot Operating System (ROS) [4]. ROS was selected for its open source and for its active community of research oriented users. Further it supports a variety of simulated and real world robotic platforms through a message based publisher/subscriber environment. Thus, direct migration of this research to working hardware is expected to be a viable path.

Within ROS, the Kinect data stream was accessed using the PrimeSense OpenNI Kinect package [3] to track the skeletal joints of the participant by ROS messages. An example of the Kinect depth image showing skeletal tracking is given by Figure 2. Depth data for eleven joints were collected over the sampling interval for possible future work. However only the participant's left hand joint is considered for gesture characterization. Data points consisted of (x, y, z) coordinates.

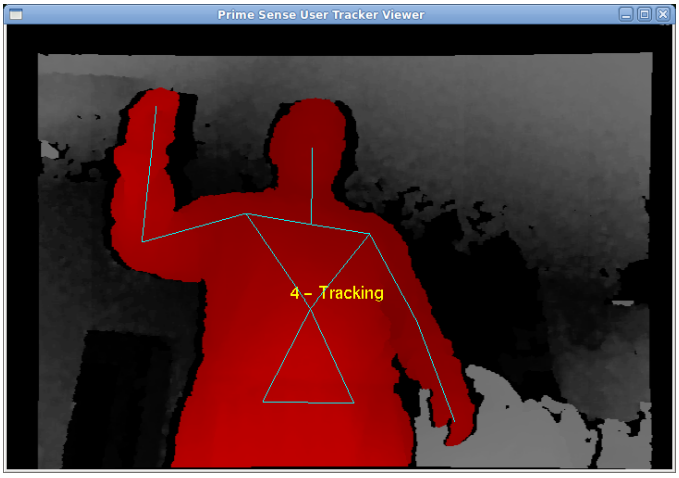


Fig. 2. The PrimeSense OpenNI depth image showing skeletal tracking.

B. Feature Extraction

Using an approach similar to [38], Dynamic Instants (DI) were extracted from each 150 point data sample for motion of the left hand joint. Position data for each of the three dimensions was first smoothed by convolution with the discrete Gaussian kernel given by (1) with $\sigma^2 = 1.0$.

$$G = [1, 4, 6, 4, 1]/16 \tag{1}$$

Velocity and acceleration data were then computed from position data for each dimension. As a further smoothing step, an *evolution time* of seven time steps was used for velocity and acceleration computation so that short term *jitter* of the actor could be filtered and longer term trends could be captured. The five highest occurrences of peak acceleration were selected as the dynamic instants. As discussed in [38], 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 ability of the Kinect to represent these peaks in 3D space and with the frame number accounting for discrete time, a spatial graph 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 Figure 3. 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 applied to the GNG algorithm.

C. The Growing Neural Gas Algorithm

The Growing Neural Gas (GNG) algorithm proposed by Fritzke [19] 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 Neural Gas (NG) algorithm proposed by Martinetz and

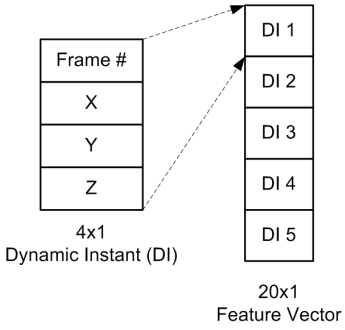


Fig. 3. Feature vector format for a depth-sampled gesture.

Schulten [33]. GNG differs from NG in its ability to form connections between nodes, and to continue adding new nodes so as to effectively map the topology of the input data distribution. The process continues on until a specified performance criteria has been satisfied. The basic GNG algorithm is given by Algorithm 1 [19]. For our implementation of GNG, the following parameters were used: $\epsilon_b = 0.05$, $\epsilon_n = 0.0006$, $\lambda = 100$, $\alpha = 0.5$, $\beta = 0.0005$ and $a_{max} = 88$.

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.

In our approach, the A 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 2D mobile robot. Therefore, as the GNG algorithm updates the cloud of reference nodes with each input vector, the nearest neighbor in the GNG cloud already holds a learned robotic response based on the history of the system. In this way, the GNG algorithm avoids the task of correctly labelling the input in favor of generating a desirable response. Using feedback from the user to gauge the quality of response, the algorithm attempts to improve the response outcome even as it quantizes the input space.

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. For simplicity, this research limits the goal response to movement in 1D (along the line $y = x$) with no change in the final angle of approach (θ). The Turtlesim environment and goal responses can be seen in Figure 6. Movement with higher degrees of freedom is left to future work.

D. User Feedback

A key aspect of our approach is the use of user supplied feedback on the relative success of a robotic response to gesture. User feedback is utilized to effectively supervise online system learning in real time and with no initial training data. However, as previously stated, obtaining gesture data and user feedback may be expensive. For this early work, user feedback was automated programmatically according to predefined goals. These defined goal configurations represent relative translations (x, y, θ) from the starting position of the robot and were chosen so as to be easily distinguished:

- *come closer* $= (3.95, 3.95, 315^o)$
- *go away* $= (-3.95, -3.95, 315^o)$
- *stop* $= (-2.00, -2.00, 315^o)$

Feedback was generated as an integer value in $\{0 \dots 10\}$ as shown in Figure 4. Feedback values less than 5 indicate a response that moved toward a configuration that was worse than where it began. Values greater than 5 indicate movement toward a desired goal. For example, a response which moved the robot 20% farther from the goal than where it started would cause a feedback of 4 to be generated. A response which moved the robot 20% closer to the goal would cause a feedback of 6 to be generated.

E. Response Refinement

The system receives the feedback value and uses it to refine and update the generated response. The portion of the system responsible for this update is isolated from the generation of the feedback. This is to emulate a future scenario when an actual user is providing the feedback. Currently, this update is the simple policy based approach given by Algorithm 2.

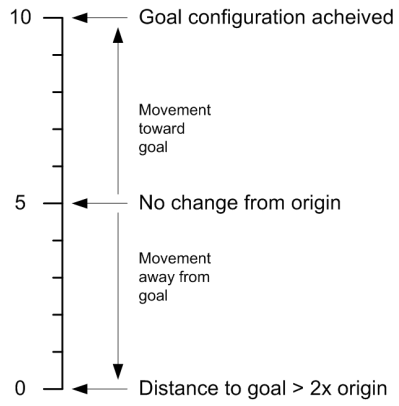


Fig. 4. User feedback scale.

Algorithm 2 Response Update Policy

1: **if** $feedback < 5$ **then**
2: Move in the opposite direction by a fraction of the distance indicated by the feedback.
3: **else if** $feedback > 5$ **then**
4: Move in the current direction by a fraction of the distance indicated by the feedback.
5: **else**
6: Move in the direction indicated by signs of (x, y) in the present response (i.e. make a guess).
7: **end if**

IV. Experimentation

The Kinect 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 eleven tracked joints were fully visible in the depth image. Participants were invited to occasionally shift their weight or angle of approach slightly so as to introduce a nominal variation in the collected data. Five volunteers were asked to perform fifty 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.

These 750 samples were randomized and presented to the system as a single *epoch*. For each sample, feature vectors were computed and passed to the GNG algorithm, a response was issued, feedback was automatically generated and the response was updated accordingly. 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, sixty epochs were executed. Results are shown in Figure 5(a). Average error can be seen to trend downward with typical error of less than $1\ m$ within approximately 15 epochs. Typical goal seeking results (in Turtlesim) using the mature GNG cloud can be seen in Figure 6.

Dissimilarity among computed DIs was seen to effect the smoothness of convergence: Some samples of a given gesture differed significantly the majority. For comparison with the original dataset, a filtered subset of samples was also generated. Those samples having fewer than twenty data points farther than 1.5 standard deviations from the mean for the gesture type were retained. This reduced the data set to an average 191 samples per gesture type for a total of 573 samples. These results are shown in Figure 5(b). Although the downward trend is smoother for the subset, the rate of convergence is similar. In a real world setting, users would be expected to exhibit natural variation in the performance of gestures. These results suggest that our system would be robust to such variation.

Perturbations within the GNG cloud can also be seen as the error curves do not descend smoothly. This may be explained again by samples within the data set which remain poorly separable despite filtering. Samples implicitly mistaken for the wrong gesture type would find their generated response to be far from desirable. However, despite such cases, the algorithm reliably reconverges toward goal configurations and average error continues to trend downward.

V. Conclusions and Future Work

In this paper, we have presented early work toward development of a gesture based human-machine interface. It has been shown that 3D data from the Kinect depth camera can be used to generate a useful descriptor of gesture in the form of prominent dynamic instants. Further, the GNG algorithm is capable of differentiating between these descriptors. Most interestingly, the goal of gauging the success of our learning algorithm based on the desirability of response rather than on a classifier label is shown to be practical. Clearly, the policy based update method we employ in this initial experiment is a simplistic approach to reinforcement learning. Development of a longer term value function to maximize user satisfaction will be of central focus in our work.

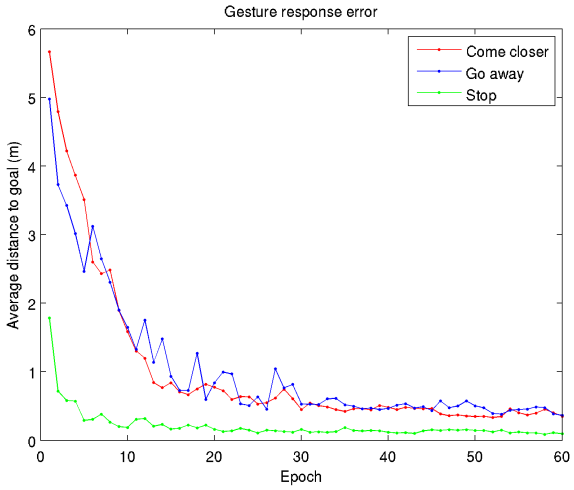
Segmentation of gestures is a typical problem in gesture recognition. Future work may include investigation of the segmentation problem.

The use of DIs as a data representation will likely be revisited. In addition to the need for improved separability in the data set, DIs present concerns regarding both spatial scale and speed of execution of the performed gesture. Progress in this area could be expected to increase the speed of convergence by the GNG algorithm, thereby reducing the expense of data collection.

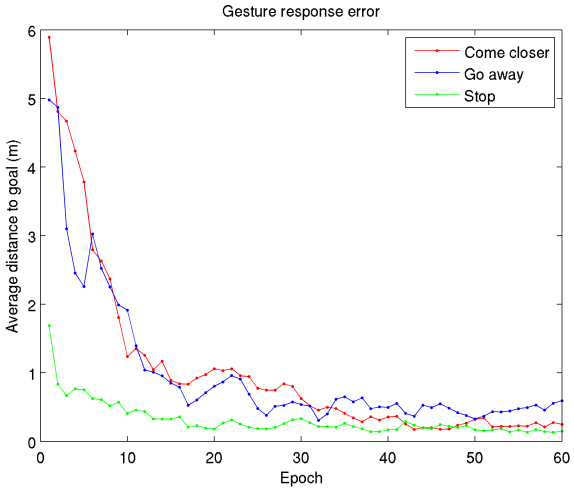
Finally, the online addition of new gestures will be explored. Certainly, for the envisioned system to effectively assist the user, the vocabulary of known commands must be open to amendment as needed. Indeed, this is a key facet of the gesture recognition problem that frames our ongoing research.

Acknowledgment

This research was supported by the U.S. National Science Foundation under award IIS-SHB-116075.

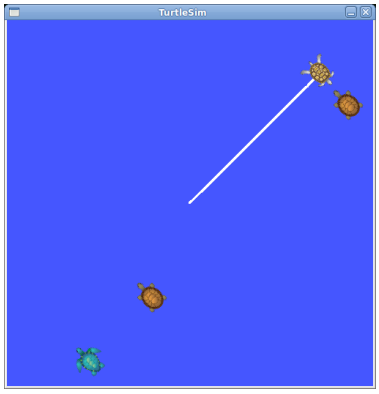


(a) Results using unfiltered samples.

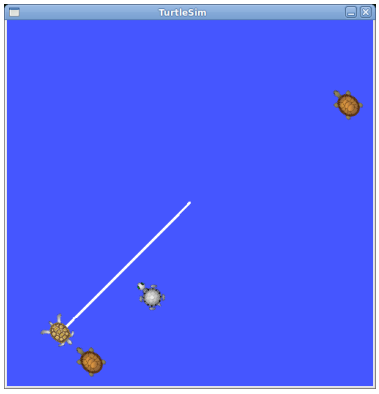


(b) Results using filtered samples.

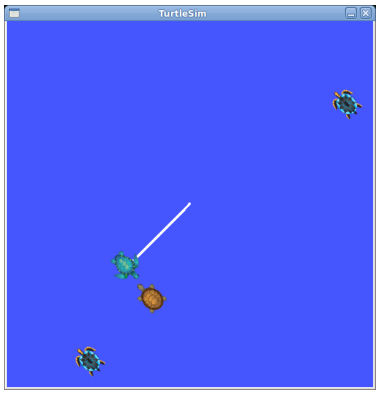
Fig. 5. Average gesture response error per epoch.



(a) Come closer.



(b) Go away.



(c) Stop.

Fig. 6. Motion paths for Turtlesim with a mature GNG cloud. The goal seeking turtle begins at the center of the frame and traces its trajectory (white line) as it moves along the diagonal line $y = x$ toward its 1D goal configuration. Markers are shown at the goal positions for each candidate gesture. Trajectories for each response are given in their respective subfigure as noted. The turtle can be seen to align with the appropriate marker for each gesture type. In all cases, the error is less than $0.1\ m$

REFERENCES

[1] ASL Pro Website. http://www.aslpro.com/cgi-bin/aslpro/aslpro.cgi.

[2] Microsoft Xbox 360 + Kinect Website. http://www.xbox.com/en-US/kinect.

[3] ROS OpenNI Website. http://www.ros.org/wiki/nite.

[4] ROS Website. http://www.ros.org.

[5] A. Angelopoulou, A. Psarrou, J. Garcia-Rodriguez, and G. Gupta. Tracking gestures using a probabilistic self-organising network. In *Proceedings of the 2010 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE, 2010.

[6] 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)*, 2(1):1–21, 2009.

[7] C. BenAbdelkader, R.G. Cutler, and L.S. Davis. Gait recognition using image self-similarity. *EURASIP Journal on Applied Signal Processing*, 2004:572–585, 2004.

[8] A.F. Bobick. Movement, Activity and Action: The Role of Knowledge in the Perception of Motion. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 352(1358):1257–1266, 1997.

[9] A.F. Bobick and A.D. Wilson. A state-based approach to the representation and recognition of gesture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(12):1325–1337, Dec. 1997.

[10] H.T. Cheng, A.M. Chen, A. Razdan, and E. Buller. Contactless gesture recognition system using proximity sensors. In *Proceedings of the 2011 IEEE International Conference on Consumer Electronics (ICCE)*, pages 149–150, Jan. 2011.

[11] K. Conn and R.A. Peters. Reinforcement learning with a supervisor for a mobile robot in a real-world environment. In *Proceedings of the 2007 International Symposium on Computational Intelligence in Robotics and Automation (CIRA 2007)*, pages 73–78. IEEE, 2007.

[12] D.J. Cook, M. Youngblood, E. Heierman, K. Gopalratnam, S. Rao, A. Litvin, and F. Khawaja. MavHome: An Agent-Based Smart Home. In *Proceedings of the IEEE International Conference on Pervasive Computing and Communications*, pages 521–524. Citeseer, 2003.

[13] K.E. Covinsky, R.M. Palmer, R.H. Fortinsky, S.R. Counsell, A.L. Stewart, D. Kresevic, C.J. Burant, and C.S. 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*, 51(4):451–458, 2003.

[14] R. Cutler and L.S. Davis. Robust real-time periodic motion detection, analysis, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):781–796, 2002.

[15] N. Dalal, B. Triggs, I. Rhone-Alps, and F. Montbonnot. Histograms of Oriented Gradients for Human Detection. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. CVPR 2005*, pages 886–893, 2005.

[16] G. Demeris, B.K. Hensel, M. Skubic, and M. Rantz. Senior residents' perceived need of and preferences for "smart home" sensor technologies. *International Journal of Technology Assessment in Health Care*, pages 120–124, 2008.

[17] B. DeRuyter and E. Pelgrim. Ambient Assisted-Living Research in CareLab. *Interactions*, XIV(4):30–34, 2007.

[18] A. Friedman. *The Adaptable House: Designing Homes for Change*. McGraw-Hill Professional, 2002.

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

[20] C. Gaskett, L. Fletcher, and A. Zelinsky. Reinforcement learning for a vision based mobile robot. In *Proceedings of the 2000 IEEE/RJS International Conference on Intelligent Robots and Systems (IROS 2000)*, volume 1, pages 403–409, Takamatsu, Japan, Oct. 2000. IEEE.

[21] J. Holmström. Growing Neural Gas: Experiments with GNG, GNG with utility and supervised GNG. Master's thesis, Upsala University, Department of Information Technology, 2002.

[22] S.S. Intille, K. Larson, and E.M. Tapia. Designing and Evaluating Technology for Independent Aging in the Home. In *International Conference on Aging, Disability and Independence*, 2003.

[23] S. Iwarsson and A. Stahl. Accessibility, Usability and Universal Design - Positioning and Definition of Concepts Describing Person-Environment Relationships. *Disability & Rehabilitation*, 25(2):57–66, 2003.

[24] S. Jin, Y. Li, G. Lu, J. Luo, W. Chen, and X. Zheng. Som-based hand gesture recognition for virtual interactions. In *Proceedings of the 2011 IEEE International Symposium on VR Innovation (ISVRI)*, pages 317–322, Mar. 2011.

[25] G. Johansson. Visual perception of biological motion and a model for its analysis. *Perception and Psychophysics*, 14(2):201–211, 1973.

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

[27] A. Karahoca and M. Nurullahoglu. Human motion analysis and action recognition. In *Proceedings of the 1st WSEAS International Conference on Multivariate Analysis and its Application in Science and Engineering*, pages 156–161. World Scientific and Engineering Academy and Society (WSEAS), 2008.

[28] C.D. Kidd, R. Orr, G.D. Abowd, C.G. Atkeson, I.A. Essa, B. MacIntyre, E. Mynatt, T.E. Starner, W. Newstetter, et al. The Aware Home: A Living Laboratory for Ubiquitous Computing Research. *Lecture Notes in Computer Science*, pages 191–198, 1999.

[29] J. Kober and J. Peters. Reinforcement learning in robotics: A survey. In *Reinforcement Learning: Stae of the Art*, pages 579–610. Springer, 2012.

[30] Y. Kuno, T. Murashima, N. Shimada, and Y. Shirai. Interactive gesture interface for intelligent wheelchairs. In *Proceedings of the IEEE International Conference on Multimedia and Expo 2000 (ICME 2000)*, volume 2, pages 789–792, 2000.

[31] S.W. Lee. Automatic gesture recognition for intelligent human-robot interaction. In *Proceedings of the 7th International Conference on Automatic Face and Gesture Recognition (FGR'06)*, Apr. 2006.

[32] J.C. Lementec and P. Bajcsy. Recognition of arm gestures using multiple orientation sensors: gesture classification. In *Proceedings of the 7th IEEE International Conference on Intelligent Transportation Systems, 2004*, pages 965–970, Oct. 2004.

[33] T. Martinetz and K. Schulten. *A "Neural-Gas" Network Learns Topologies*. Elsevier Science Publishers, Amsterdam, The Netherlands, 1991.

[34] S. Mitra and T. Acharya. Gesture recognition: A survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 37(3):311–324, 2007.

[35] MA Moni and A.B.M.S. Ali. Hmm based hand gesture recognition: A review on techniques and approaches. In *Proceedings of the 2nd IEEE International Conference on Computer Science and Information Technology (ICCSIT 2009)*, pages 433–437, 2009.

[36] J.S. Prasad and G.C. Nandi. Clustering method evaluation for hidden markov model based real-time gesture recognition. In *Proceedings of the International Conference on Advances in Recent Technologies in Communication and Computing, 2009 (ARTCom'09)*, pages 419–423, Oct. 2009.

[37] W.F.E. Preiser and E. Ostroff. *Universal Design Handbook*. McGraw-Hill Professional, 2001.

[38] C. Rao, A. Yilmaz, and M. Shah. View-Invariant Representation and Recognition of Actions. *International Journal of Computer Vision*, 50(2):203–226, 2002.

[39] Dongseok Ryu, Dugan Um, P. Tanofsky, Do Hyong Koh, Young Sam Ryu, and Sungchul Kang. T-less : A novel touchless human-machine interface based on infrared proximity sensing. In *Proceedings of the 2010 IEEE/RJS International Conference on Intelligent Robots and Systems (IROS)*, pages 5220–5225, Oct. 2010.

[40] T. Schlömer, B. Poppinga, N. Henze, and S. Boll. Gesture recognition with a wii controller. In *Proceedings of the 2nd International Conference on Tangible and Embedded Interaction*, pages 11–14, 2008.

[41] E. Stergiopoulou and N. Papamarkos. A new technique for hand gesture recognition. In *Proceedings of the 2006 IEEE International Conference on Image Processing*, pages 2657–2660. IEEE, 2006.

[42] A.R. Varkonyi-Koczy and B. Tusor. Human–computer interaction for smart environment applications using fuzzy hand posture and gesture models. *IEEE Transactions on Instrumentation and Measurement*, 60(5):1505–1514, 2011.

[43] A.D. Wilson and A.F. Bobick. Realtime online adaptive gesture recognition. In *Proceedings of the 15th International Conference on Pattern Recognition, 2000*, volume 1, pages 270–275. IEEE, 2000.

[44] J. Yamato, J. Ohya, and K. Ishii. Recognizing human action in time-sequential images using hidden markov model. In *Proceedings of the 1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'92)*, pages 379–385, 1992.

[45] Y. Yamazaki, HA Vu, PQ Le, Z. Liu, C. Fatichah, M. Dai, H. Oikawa, D. Masano, O. Thet, Y. Tang, et al. Gesture recognition using combination of acceleration sensor and images for casual communication between robots and humans. In *2010 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–7, July 2010.

[46] M.H. Yang and N. Ahuja. Recognizing hand gesture using motion trajectories. In *Proceedings of the 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, June 1999.

[47] P.M. Yanik, J. Merino, J. Manganelli, L. Smolentzov, I.D. Walker, J.O. Brooks, and K.E. Green. Sensor placement for activity recognition: comparing video data with motion sensor data. *International Journal of Circuits, Systems and Signal Processing*, (5):279–286, 2011.

[48] S. Zhou, Q. Shan, F. Fei, W.J. Li, C.P. Kwong, P.C.K. Wu, B. Meng, C.K.H. Chan, and J.Y.J. Liou. Gesture recognition for interactive controllers using mems motion sensors. In *Proceedings of the 4th IEEE International Conference on Nano/Micro Engineered and Molecular Systems, 2009 (NEMS 2009)*, pages 935–940, Jan. 2009.

[49] 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*, 41(3):569–573, May 2011.