A novel Competitive Neural Classifier for Gesture Recognition with Small Training Sets

Facundo Quiroga, Leonardo Corbalán

III-LIDI Institute, Informatics Faculty, UNLP
La Plata, Buenos Aires, Argentina
fquiroga,corbalan@lidi.unlp.edu.ar

Abstract. Gesture recognition is a major area of interest in human-computer interaction. Recent advances in sensor technology and computer power has allowed us to perform real-time joint tracking with commodity hardware, but robust, adaptable, user-independent usable hand gesture classification remains an open problem. Since it is desirable that users can record their own gestures to expand their gesture vocabulary, a method that performs well on small training sets is required. We propose a novel competitive neural classifier (CNC) that recognizes arabic numbers hand gestures with a 98% success rate, even when trained with a small sample set (3 gestures per class). The approach uses the direction of movement between gesture sampling points as features and is time, scale and translation invariant. By using a technique borrowed from object and speaker recognition methods, it is also starting-point invariant, a new property we define for closed gestures. We found its performance to be on par with standard classifiers for temporal pattern recognition.

Keywords: gesture recognition,scale invariant,speed invariant,starting-point invariant,neural network,cpn,competitive

1 Introduction

The recent rise of gesture interaction as a practical possibility, through new devices and sensors, has made natural gesture-based software a reality, with applications ranging from web browsing and gaming to sign language interpretation and smart home interaction. A gesture recognition system usually consists of two stages: low-level feature extraction and representation based on sensor data, for example using depth images taken from a time-of-flight camera; and gesture classification employing the extracted features. Current research efforts in human-computer interaction, computer vision, motion analysis, machine learning and pattern recognition are contributing to the creation of even more robust and usable recognition systems [17] in both stages.

The Kinect SDK has been recently used as a stepping stone for doing research in the second stage of a gesture recognition system based on body joint 3D positions, for example to perform a comparison of template methods for real-time

gesture recognition [14], testing a Weighted Dynamic Time Warping algorithm [2], posture detection [15], to design a system that monitors posture ergnonomics [16], human behavior recognition by a mobile robot following human subjects [1], and allowing wheelchair-accessible gesture-based game interaction [5]. This is a recent trend in vision based gesture-recognition, since previous works mostly focused on the feature extraction stage [3], and used image based features instead of performing body joint tracking and using the resulting 3D position data to construct appropiate features as in [18] for hand-gesture recognition. Also, most hand gesture recognition research has additionaly employed finger and palm information because they tipically address sign language recognition.

In this work, we propose a speed, translation and scale-invariant method, the Competitive Neural Classifier (CNC), for recognizing hand gestures based on a time-labeled sequence of 3D hand positions, with a restricted training set size. The CNC was partially inspired by Probabilistic SOM's (ProbSOM) [4] approach to speaker recognition. The proposed methodology for the CNC (and ProbSOM) discards the sequence information contained in the extracted features computed from the sample data and thus follows an approach similar to those employed in object or speaker recognition, that is, the characterization of a sample by means of a set of distinctive features; an approach unexplored, to the best of our knowledge, in the area of gesture recognition. The architecture of each sub-classifier maps those features into a lower dimensional space by means of a competitive layer trained also in a competitive fashion, and uses the resulting activation patterns as a gesture signature employed as input for another competitive network that outputs the likelihood of each class given each sample. Finally, the use of many such sub-classifiers improves the recognition rate by combining different mappings derived from different clusterings and thus provides robustness to the method.

We focus directly on the the second stage of the gesture recognition by leveraging the Kinect SDK's recognition algorithms to obtain user joint positions and generate a gesture database to test the method and compare its performance against ProbSOM and other two known techniques: Input-Delay Feedforward Networks [7], and a modified Ordered Means Model [6] algorithm called ST-OMM.

This work is organized as follows: we introduce the gesture database in section 2, together with the preprocessing and feature extraction stage; then, we present the CNC in section 3, a brief introduction to the compared methods in section 4, and finish with experimental data and conclusions in sections 5 and 6.

2 Preprocessing and Feature Extraction

2.1 Gesture Database

We performed all of our experiments using the Arabic Numbers Hand Gesture Database [1], a small database of our creation with 10 samples of each of the arabic

[1] More information available at https://sites.google.com/site/dbanhg/

digits performed using the left hand and recorded with Microsoft Kinect's SDK, which gives a set of classes $C = 0 \dots 9$. The recording of all samples was done at an average of 28 *fps*, by the same person. In the recording of the different samples of each digit the orientation of the person with respect to the camera was the same, but the samples were performed starting from different hand and body positions and each gesture shape was drawn with different sizes.

Each gesture sample $\mathbf{s_i} \in S$, where S is our gesture database, consists of a sequence $\mathbf{s_i} = \mathbf{s_i}[1], \mathbf{s_i}[2], \dots, \mathbf{s_i}[n_i], \quad \mathbf{s_i}[j] \in \mathbb{R}^3, \quad j = 1 \dots n_i$, corresponding to the hand positions in a 3D space, time-labeled $T_i = t_1, \dots, t_{n_i}, \quad t_j \in \mathbb{R}, \quad 0 = t_1 < t_2 < \dots < t_{n_i}$, and gesture class label c_i . Each sample $\mathbf{s_i}$ may have a different number of positions n_i, depending on the capture frame rate and the time used to execute the gesture.

2.2 Preprocessing

In the preprocessing stage, the first and last 3 positions of each sample are discarded because they usually contain unwanted information introduced by incorrect gesture segmentation. The samples were smoothed individually using an unweighted moving average with a window of size w in order to remove the high-frequency information from the signal, because the chosen features are based on the direction between consecutive sample points and small fluctuations in direction give too local information to characterize the overall shape of a gesture.

The nature of the feedforward and ST-OMM architectures requires the number of sample positions n_i to be constant across all samples. Also, it is desirable to obtain speed-invariant features. In order to achieve this each sample is resampled to a sequence of constant length n using cubic splines interpolation with an arc-length parameterization.

The parameterization of each sample $\mathbf{s}$ of length q gives the position of the hand in the 3D space as a function of the arc-length distance from the first position of the sample. For each position $\mathbf{s}[j]$ we calculate the arc-length distance from the first position $l_j = \sum_{k=2}^{j} ||\mathbf{s}[j] - \mathbf{s}[j-1]||$, where $||\cdot||$ is the Euclidean norm. The resampling is done at n uniformly distributed knots in the total sample arc length $L = l_q$, given by $k_j = \frac{j-1}{n-1} * L, \quad j = 1 \dots n$. We obtain the sequence of points $\mathbf{r}$ of length n such that $\mathbf{r}[j] = cubic(k_j, near_4(k_j)) \quad j = 1 \dots n$ where $cubic(x, (x_1, x_2, x_3, x_4))$ performs the cubic interpolation at distance x using distances (x_1, x_2, x_3, x_4) whose positions are known and $near_4(x)$ returns the 4 distances nearest to the distance x (ie, $min_4 = (|l_1 - x|, \dots, |l_q - x|)$) such that $x_1 < x_2 \leq x \leq x_3 < x_4$.

2.3 Features

From the smoothed, resampled sequence $\mathbf{r_i}$ we compute the normalized first difference $\mathbf{d_i}$ where $\mathbf{d_i}[j] = \frac{\mathbf{r_i}[j+1] - \mathbf{r_i}[j]}{||\mathbf{r_i}[j+1] - \mathbf{r_i}[j]||}, \quad j = 1 \dots n-1, \quad \mathbf{d_i}[j] \in \mathbb{R}^3$, which represents the relative directions between the sample positions. By computing the first difference, we obtain a translation invariant representation. By normalizing, we remove all speed information contained in the length of each direction

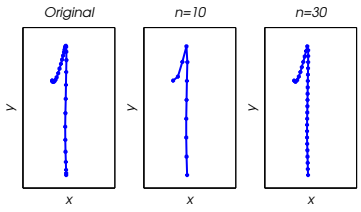

Fig. 1. Projection on the xy plane of a sample of the gesture 1, before and after resampling with $n = 10, 30$. The greater density of sampling points, shown as dots, in the upper part in relation to the lower part of the original that was caused by a difference in speed when performing the gesture has been compensated by resampling with different degrees of detail.

vector, making the signal invariant to the speed and scale with which the gesture was drawn. Note that without the resampling the normalizing would still leave a considerable amount of velocity information in the signal because the amount of sampling points in the segments (in the arc-length sense) where the user performs the gesture at high speed is lower than in those segments where the hand moves more slowly.

As an alternative to the direction vectors, we also employed the angles of the spherical representation of the direction vectors, obtaining a representation $\mathbf{a_i}$, where $\mathbf{a_i}[j] = (\theta, \phi)_j \in \mathbb{R}^2$. The z coordinate is left out because it is always 1 as a consequence of the previous normalization, and we rescaled the angles so that $\theta, \phi \in [-1, 1]$.

Given the periodical nature of the angle representation, in all angle differences calculated for all classification algorithms we utilized the difference function $d(\alpha, \beta) = min(|\alpha - \beta|, 2 - |\alpha - \beta|)$. Although both features are mathematically equivalent and share the same desirable properties (translation, scale and speed invariance [12]), they produce slightly different results in our experiments. In the following sections we will refer to a secuence of sample features as $\mathbf{s}$, where $\mathbf{s}[j] \in \mathbb{R}^d$ could represent either feature (with $d = 2$ for angles and $d = 3$ for cartesian directions).

2.4 Starting point invariance

We define the *starting-point invariance* property for closed gestures, which are those that, ideally, start and finish in the same position. In such a case, like when recognizing the gesture corresponding to the digit 0, we would like the recognition algorithm to be able to detect the gesture without regard to where the user started performing it. Therefore, a feature given by $f :: Samples \rightarrow Features$ is starting-point invariant if $f(\mathbf{s}) = f(shift(\mathbf{s}, k)), \quad k = 1..n-1$ where n is the length of the sample, $shift(\mathbf{s}, k) = (s_{(k)\%(n+1)}, s_{(k+1)\%(n+1)}, \ldots, s_{(n+k)\%(n+1)})$ and $\%$ is the *modulo* operator.

3 Competitive Neural Classifier (CNC)

The CNC, a new gesture recognition strategy based on competitive neural networks, employs a combination of competitive neural networks and a decision mechanism that integrates information generated by multiple subclassifiers to add robustness to the recognition process and improve the recognition rate. The structure of a CNC with p subclassifiers is shown in figure 2 a).

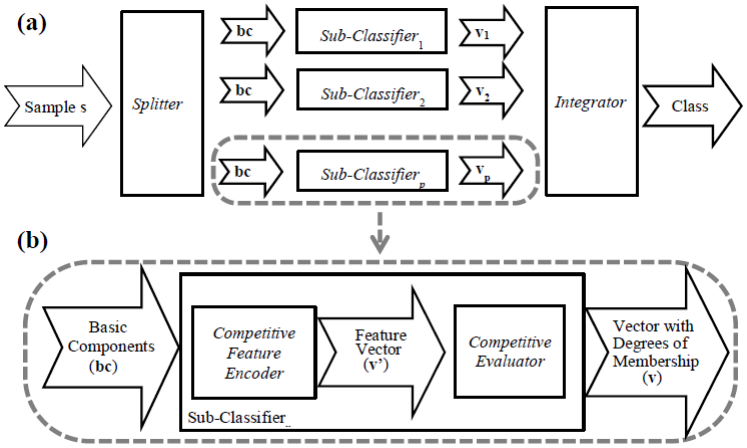

Fig. 2. a) CNC Architecture b) Subclassifier module

3.1 Classifier structure and functioning

When a sample **s** enters the classifier, the Splitter module generates the sequence of basic components **bc** and sends a copy to each of the p subclassifiers. Note that each basic component **bc**$[j]$ is simply the cartesian or angle representation of the jth direction vector described in the previous section. The Competitive Feature Encoder Module (CFEM) in each subclassifier (figure 2 b) implements a neural network nn' with m neurons $h'_1, h'_2, \ldots, h'_m$, trained with the well-known CPN learning algorithm [8]. Given a sequence of n basic components (a sample), the CFEM maps it into a characteristic vector $\mathbf{v}'$ according to:

$$\mathbf{v}' = (v'_1, v'_2, \ldots, v'_m)$$
$$v'_k = count(h'_k)/n \quad k = 1..m$$

where $count(h'_k)$ represents the number of basic components for which the neuron h'_k had an activation value greater than other neurons. Therefore, $\mathbf{v}'$ codes information about the distribution of each basic component of the sample **s**

according to the hidden space. Since the $nn's$ are trained independently, each will produce a classification based on different clusterings, hopefully complementary to each other.

The Competitive Evaluator Module (CEM) contains a neural network nn with a competitive layer composed of z hidden neurons $h_1, \ldots, h_z$ with corresponding weights $\mathbf{w_1}, \ldots, \mathbf{w_z}$, where z is determined by the learning algorithm. The neurons of this layer are stimulated with the vector $\mathbf{v'}$ to output a vector $\mathbf{v} = (v_1, \ldots, v_z)$ of scores. The network deviates from the typical competitive architecture in that instead of identifying a unique winner neuron, this vector represents the degree of membership of the sample to each neuron using the inverse of the Manhattan distance $||\cdot||$ as a similarity function, and is given by $v_k = 1/||(\mathbf{v'}, \mathbf{w_k})|| k = 1..z$.

Each neuron h_k is associated with a gesture class c by means of a function $f :: Neuron- > Class$, and therefore such scores $\mathbf{v}$ also represent the degree of membership of the sample to each class. Finally, the Integrator module receives the outputs $\mathbf{v_i}, \quad i = 1..p$ of every classifier, and calculates the corresponding class as $class = f(max_k(scores))$, where $scores = \sum_{i=1}^{p} \mathbf{v_i}$ and max_k returns the index of the vector component (hidden neuron) with the maximum value.

3.2 Learning algorithm

The subclassifiers are trained independently in two stages. First, the nn' of each CFEM is trained with the classical CPN iterative learning algorithm using the basic components of all samples and the Manhattan distance as a similarity function.

After the CFEM's training is finished, each h'_i corresponds to the centroid of a cluster of basic components. Given the nontraditional use of each nn' in the generation of characteristic vectors $\mathbf{v'}$on the training algorithm can be stopped very quickly (as early as two iterations in our tests) while still obtaining good results.

The neural network nn of the CEM requires no training. Once the CFEM's training is complete, the nn is built with $z = |C| \times u$ neurons h_i, where u is the number of samples of each class and $|C|$ the number of classes. To each h_i corresponds a weight vector $\mathbf{w_i} = \mathbf{v'_i}$ where $\mathbf{v'_i}$ is the characteristic vector generated by the CFEM when presented with the basic components of sample $\mathbf{s_i}$. The mapping function f is simply $f(i) = c_i$, that is, given sample index i, it returns the class label of sample i.

4 Compared Methods

4.1 Probabilistic SOM (ProbSOM)

ProbSOM combines the well know Self Organizing Map (SOM) clustering technique with a probabilistic decision model which has been succesfully applied to solve speaker identification problems. CNC has been partially inspired by

ProbSOM's approach of extracting basic components of a signal to stimulate a competitive layer, but differs substantially in the mapping approach and decision process. For an accurate comparison with CNC, we tested an ensemble of p independently trained ProbSOMs used as subclassifiers, and use as final output the mean of the subclassifiers scores for each class.

4.2 Input Delay Feedforward Network (ID-FF)

Input Delay Feedforward Networks (ID-FF) are a class of feedforward networks adapted for temporal sequence processing. Applied to this type of problems, their main distinctive feature is the grouping of the sequence of feature vectors that characterize a gesture into a single vector by simply aggregating all the features while maintaining their temporal ordering. This results in a representation that simply but effectively models the dynamics of the gesture and has the advantage that the neural network can be designed and trained in a traditional way.

We chose the standard 2-layered feedforward network, trained with resilient backpropagation, a well-known, fast and proven first-order training algorithm [9]. We employed the transfer function *tansig* for both the hidden and output layer. The output layer contains 10 neurons, one for each gesture class c, with output values $o_c \in -1, 1, c = 1 \dots 10$.

4.3 Small-Training-Set Ordered Means Model (ST-OMM)

The Ordered Means Model (OMM) is a simplified Hidden Markov Model (HMM) which has been successfully applied for solving gesture recognition problems in a variety of settings. Moreover, HMMs are the de-facto standard for generative classifier models, (and, arguably, gesture recognition [10]) and thus a good choice for comparison.

While most model building methods for both the OMM and HMM commonly employ some variant of the Expectation-Maximization (EM) algorithm to find optimal values for their parameters, such a choice is ill-suited for small training sets as required for our problem. Therefore, we have created the Small-Training-Set-OMM (ST-OMM), an adaptation of the OMM approach to comply with this requirement.

The ST-OMM takes as input a sample, which is a fixed length sequence $\mathbf{s}$ of n features, and outputs the likelihoods of the sample belonging to each gesture class c. The ST-OMM is built with n competitive Gaussian Mixture Models (C-GMM), and each C-GMM $G_j, j = 1 \dots n$ is composed of a set of states $S_{j,c}, c \in C$, with constant mixture coefficients $\omega_c = 1/|C|$. To every state $S_{j,c}$ corresponds to a set of Gaussian pdfs with means $\mu_{j,c,k}$ and covariance matrices $\mathbf{\Sigma}_{j,c,k}, k = 1 \dots |Sc|$, where Sc is the set of samples of class c, which models the probability of an emission of gesture part j by the gesture class c in a competitive fashion. This parameters are estimated as $\mu_{j,c,k} = \mathbf{s}_{ck}[j]$ and $\mathbf{\Sigma}_{j,c,k} = cov(I_{j,c})$, $k = 1 \dots |Sc|$, where $\mathbf{s}_{ck}$ is the kth sample of class c and $I_{j,c}$ is a matrix whose columns are $\left[\, \mathbf{s}_{c1}[j] \; \mathbf{s}_{c2}[j] \; \cdots \; \mathbf{s}_{c|Sc|}[j] \;\; {\scriptstyle k=1 \dots |Sc|} \,\right]$, that is, $I_{j,c}$ is the matrix that contains the j^{th} feature of every sample of class c.

We are therefore using each part of every sample of the training set - the best likelihood estimators for that set - as a C-GMM mean, which does not yield a computationally demanding model because we are specifically targeting a very small training sets. For the classification of a new sample $\mathbf{s}$, we calculate the likelihood of emission for each state $S_{j,c}$ as:

$$P(S_{j,c}|s[j]) = \frac{P(s[j]|S_{j,c})P(S_{j,c})}{P(s[j])} = \frac{p_{j,c}P(S_{j,c})}{\sum_{k\in C} p_{j,k}P(S_{j,k})} = \frac{p_{j,c}}{\sum_{k\in C} p_{j,k}}$$

where $P(S_{j,k}) = \frac{1}{|C|}$ is the same for all classes k, and $p_{j,k} = P(s[j]|S_{j,c}) = max(\mathcal{N}(s[j]; \mu_{j,c,k}, \mathbf{\Sigma}_{j,c,k}))$ is the maximum of the scores that model the likelihood of the j^{th} feature of the sample belonging to class c.

Then, the likelihood of the whole sample belonging to class c is:

$$P(c|s) = \frac{P(s|c)P(c)}{\sum_{k\in C} P(s|k)P(k)} = \frac{P(s|c)}{\sum_{k\in C} P(s|k)}$$

where we define $P(k) = \frac{1}{|C|}$ and $P(s|k) = \frac{\sum_{j=1}^{n} P(S_{j,k}|s[j])}{n}$

We can thus picture the ST-OMM as a $|C| \times n$ state HMM, with one left-to-right submodel for each gesture that does not allow a transition from a state to itself, that is:

$$P(S_{j,c} \to S_{j',c}) = \begin{cases} 1 & \text{if } j' = j+1 \\ 0 & \text{otherwise} \end{cases}$$

This restriction avoids doing a dynamic programming search over state combinations and, although it is obviously of lower computational capacity than a full HMM, works well given a large enough n, since any desynchronization between a novel performance of the gesture and the model produces only small mismatches.

5 Experimentation

We compared the recognition rate of the four methods on the gesture database, using the same preprocessing parameters for all. The resampling size n was set to 60 and the smoothing window size w to 5. Each algorithm was tested 500 times using random subsampling cross-validation, with 3 samples per class for the training set (30 total) and 7 for the test set (70 total). In the case of the feedforward network, 2 samples of each class were taken from the test set to be used as the validation set, leaving 5 for the test set. For the ID-FF, CNC subclassifiers and ProbSOM networks, we show results for hidden neurons $m = 30$, $m = 70$ and $m = 100$ respectively, which gave the best results in our experiments (we tested $m \in 5, 10, \ldots, 150$ to determine those values). Both the CNC and ProbSOM had $p = 5$ sub-classifiers. In all experiments, for ProbSOM, ST-OMM

and ID-FF the class assigned to a sample gesture was the one that gave the best score for that class, irrespective of its actual value; that is, when presented with a sample, the method calculates the outputs o_c for each class normally and the corresponding class is assigned according to the rule $class = max_c(o_c)$.

Method / Feature	Direction	Angles
CNC	98.91 (1.20)	95.32 (2.20)
ProbSOM	62.69 (5.94)	43.57 (6.86)
ID-FF	91.54 (9.06)	80.88 (8.81)
ST-OMM	95.54 (2.72)	96.40 (2.09)

Table 1. Sample mean and standard deviation of recognition rates over 500 experiments, with training and test cases chosen randomly, $w = 5$, $n = 60$.

We also show the performance without resampling for the ProbSOM and CNC which do not require a fixed length input vector, although in this case they are not truly speed independent.

Method / Feature	Direction	Angles
CNC	98.37 (1.63)	83.30 (4.40)
ProbSOM	56.83 (4.67)	39.57 (6.86)

Table 2. Mean and standard deviation of recognition rates without resampling over 500 trials, $w = 5$.

As we can see, the recognition rates are slightly lower but not significantly, which shows that in principle the method could be used without a proper normalization in cases where the need for reduced computation outweighs that of high recognition rates.

6 Conclusion

A novel approach for gesture recognition with small training samples has been presented that is time, scale and translation invariant, and for closed gestures, starting-point invariant as well, while achieving high recognition rates comparable to other approaches and with a learning algorithm that requires few iterations to converge.

In our experiments, the CNC, ST-OMM and the ID-FF all perform reasonably well, especially first two. In addition, the CNC has the advantage of being quick to train and starting-point invariant without any modification, although at a greater intrinsic computational cost. For the other two methods, the same effect can be achieved by shifting the sample in a circular way in all n possible combinations which also scales up by a factor n their execution order.

In future work, we hope to determine the method's performance on larger gesture databases, and apply it in a real-time setting to test its ability to recognize gestures in a unsegmented stream of hand positions. Finally, we intend to improve and extend our current database with 3D gesture data to provide a reference point for future comparisons and benchmarkings.

References

1. Human behavior recognition by a mobile robot following human subjects. In: Evaluating AAL Systems Through Competitive Benchmarking (2013)
2. Celebi, S., Aydin, A.S., Temiz, T.T., Arici, T.: Gesture recognition using skeleton data with weighted dynamic time warping. Computer Vision Theory and Applications. Visapp (2013)
3. Chaudhary, A., Raheja, J.L., Das, K., Raheja, S.: Intelligent approaches to interact with machines using hand gesture recognition in natural way: a survey. arXiv preprint arXiv:1303.2292 (2013)
4. Estrebou C., Lanzarini L., H.W.: Voice recognition based on probabilistic som. In: Latinamerican Informatics Conference. CLEI 2010. Paraguay. October 2010. (2010)
5. Gerling, K.M., Kalyn, M.R., Mandryk, R.L.: Kinectwheels: wheelchair-accessible motion-based game interaction. In: CHI '13 Extended Abstracts on Human Factors in Computing Systems. CHI EA '13, ACM (2013)
6. Grossekathofer, U., Wohler, N.C., Hermann, T., Kopp, S.: On-the-fly behavior coordination for interactive virtual agents: a model for learning, recognizing and reproducing hand-arm gestures online. In: Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems - Volume 3. AAMAS '12 (2012)
7. Haykin, S.: Neural Networks: A Comprehensive Foundation. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2nd edn. (1998)
8. Hecht-Nielsen, R.: Counterpropagation networks. Applied optics 26(23), 4979–4983 (1987)
9. Igel, C., Husken, M.: Empirical evaluation of the improved rprop learning algorithms (2003)
10. Just, A., Marcel, S.: A comparative study of two state-of-the-art sequence processing techniques for hand gesture recognition. Comput. Vis. Image Underst. 113(4), 532–543 (Apr 2009), http://dx.doi.org/10.1016/j.cviu.2008.12.001
11. Kindratenko, V.V.: On using functions to describe the shape. J. Math. Imaging Vis. 18(3) (May 2003)
12. Lai, K., Konrad, J., Ishwar, P.: A gesture-driven computer interface using kinect. In: Image Analysis and Interpretation (SSIAI), 2012 IEEE Southwest Symposium on. pp. 185–188 (2012)
13. Le, T.L., Nguyen, M.Q., Nguyen, T.T.M.: Human posture recognition using human skeleton provided by kinect. In: Computing, Management and Telecommunications (ComManTel), 2013 International Conference on (2013)
14. Martin, C., Burkert, D., Choi, K., Wieczorek, N., McGregor, P., Herrmann, R., Beling, P.: A real-time ergonomic monitoring system using the microsoft kinect. In: Systems and Information Design Symposium (SIEDS), 2012 IEEE (2012)
15. Sarkar, A.R., Sanyal, G., Majumder, S.: Article: Hand gesture recognition systems: A survey. International Journal of Computer Applications (2013)
16. Shen, X., Hua, G., Williams, L., Wu, Y.: Dynamic hand gesture recognition: An exemplar-based approach from motion divergence fields. Image and Vision Computing 30(3), 227 – 235 (2012), http://www.sciencedirect.com/science/article/pii/S0262885611001193, <ce:title>Best of Automatic Face and Gesture Recognition 2011</ce:title>