HEGM: A Hierarchical Elastic Graph Matching for Hand Gesture Recognition

Yu-Ting Li, Juan P. Wachs*

School of Industrial Engineering, Purdue University, West Lafayette IN 47906, U.S.A

Abstract

A hierarchical scheme for elastic graph matching applied to hand gesture recognition is proposed. The proposed algorithm exploits the relative discriminatory capabilities of visual features scattered on the images, assigning corresponding weights to each feature. A boosting algorithm is used to determine the structure of the hierarchy of a given graph. The graph is expressed by annotating nodes of interest over the target object to form a bunch graph. Three annotation techniques, manual, semi-automatic, and automatic annotation are used to determine the position of the nodes. The scheme and the annotation approaches are applied to explore hand gesture recognition performance. A number of filter banks are applied to hand gestures images to investigate the effect of using different feature representation approaches. Experimental results show that the hierarchical elastic graph matching (HEGM) approach classified the hand posture with a gesture recognition accuracy of 99.85% when visual features were extracted by utilizing the histogram of oriented gradient (HOG) representation. The results also provide the performance measures from aspect of recognition accuracy to matching benefits, node positions correlation and consistency on three annotation approaches, showing that the semi-automatic annotation method is more efficient and accurate than the other two methods.

Keyword: Elastic bunch graph, Graph matching, Feature hierarchy, Feature extraction, Hand gesture recognition

1. Introduction

Hand gesture recognition has the potential to enable natural communication between human and machines. Hand gestures have been used in human-computer interaction (HCI) for the last three decades, since it allows a level of expression and naturalness comparable to that achieved in interpersonal communication [1]. Such systems include smaller, cheaper and versatile sensors that are becoming a new trend in gaming [2, 3, 4] and in mobile communications. The recognition of hand postures and gestures involves modeling, representation, analysis and interpretation of gestures. These phases require low memory overhead so they do not affect the natural flow of the interaction. While gesture based interaction can be achieved through a number of ways (e.g. gloves, controls, magnetic trackers), vision-based interaction is the current mainstream approach, since it
allows the user to remain untethered to the machine [5]. Various application areas for gesture interaction have been studied and developed over the past 30 years [5] including: sign language interpretation [6,7], medical systems and assistive technologies [8], entertainment [9], and human-robot interaction [11, 12, 13, 14]. However, response times and accuracy lag from the interaction offered by standard interfaces such as mice and keyboard. Thus, the ideal gesture recognition technology will require to be precise (high true positives and low false positive rates) while keeping the natural feeling of interpersonal communication.

This paper describes a procedure to recognize hand gestures using graph matching where the nodes in the graph are assigned different hierarchy levels, relative to their importance in the matching process. We propose the hierarchical elastic graph matching method (HEGM) for classification of a lexicon of hand postures. The main improvement to the classic method of elastic graph matching (EGM) [15] is the use of levels of hierarchies assigned to the nodes. The nodes represent visual features on the gesture image. Those visual features with higher likelihood to be found on the target image receive a higher hierarchy level compared to those features that are less consistent with the graph model. This paper is an extension of our previous work [16]. The first contribution of this paper is the enhancement on the recognition rate by enabling higher discriminative power to the features assigned to higher levels in the node hierarchy. The concept of hierarchy is expressed by assigning more computational resources to those features in the graph that have more discriminative power. The second contribution of this work is the comparison of image representations using features. A bank of Gabor filters, Gaussian filters and Laplacian of Gaussian filters (LoG), and Histogram of Oriented Gradient (HOG) are used to describe the visual gesture and are compared. Previous works [17, 18] have showed that representing the human silhouette using HOG features can result in robust human detection. In our work, a framework using HOG feature is used, to extract the visual features over the graph’s nodes. This method shows precise recognition under a number of scenarios. The third contribution of this work is a study on efficient annotation techniques for the creation of bunch graphs. We applied four metrics to evaluate the effectiveness of the proposed annotation methods.

The rest of the paper is organized as follows: in Section 2 the framework for gesture recognition, including the fundamentals of EGM and enhanced HEGM is described. Section 3 describes feature extraction and representation using the various filter banks; and the proposed annotation methods are explained. Experimental results in Section 4 demonstrate the feasibility and efficiency of the proposed techniques. Finally the discussion and conclusions are presented in Section 5.

2. Overview of Gesture Recognition Framework

2.1. Elastic Graph Matching

Elastic graph matching (EGM) is an object recognition approach that is used to represent an object as a labeled bunch graph [19]. This object is intended to be recognized in unseen images. The target object is repre-
represented as a connected graph whereas the nodes in the graph represent salient features in the target object. The bunch graph is often built based on a group of template images (dictionary). The feature values are filter responses at each node in the graph. The similarity between the target and the template is determined by computing the node values in the template images (within the bunch graph) and a target image. To adjust the graph to fit better the target image, nodes’ positions are adjusted so a cost function is minimized. This way posture recognition is achieved on target images. Over the years, EGM was implemented for tasks such as face recognition [19, 20], face verification [21] and gesture recognition [15]. In Wiskott et al. [19], EGM was proposed to recognize facial images where features were extracted at facial key regions (e.g. the pupils, the beard, the nose, and the corners of the mouth). Those key regions were labeled as interesting nodes in individual graphs, which when grouped formed a collection (or a bunch graph). Thus the bunch graph representation has the generalization strength to cover the possible changes of individual’s face [22]. Bunch graph are also used to represent and recognize hand postures [15, 23]. Triesch et al. [15] employed EGM to develop a classification approach of hand gestures against complex background. In Ref. [15], by convolving a set of images (the dictionary set) with a Gabor-based filter bank, a Gabor jet included the responses computed on the graph nodes. The positions of the nodes were annotated a priori over the template images. The jet is a vector of complex responses which consists of a set of filter responses and it is defined as \( J(\vec{x}) = \alpha(\vec{x})e^{i\phi(\vec{x})} \) at a given pixel \( \vec{x} \).

In this paper, the objects of interest are hand postures. Thus, the classification of a given image as a gesture is obtained by measuring the likelihood of two jets (one from the bunch graph \( J \), and one from the target image \( J' \)). The similarity function using the magnitude \( \alpha_j \) and phase \( \phi_j \) of the two jets is used find a matching score between the target image and the bunch graph (obtained from dictionary images); and is stated as follow:

\[
S_{\text{pha}}(J, J') = \frac{1}{2} \left( 1 + \sum_j \frac{\alpha_j \alpha'_j \cos(\phi - \phi')}{\sqrt{\sum_j \alpha_j^2 \sum_j \alpha'^2}} \right)
\]

(1)

where \( j \) is the dimensionality of the jet. The phase information varies rapidly between continuous pixels, thus the maxima responses provide a good initial estimate about the position of the hand within the target image. Once the bunch graph is initially positioned on the image, all nodes are allowed to shift locally. The links connecting the nodes within the graph express some topological metric, such as the Euclidian distance. A penalty cost is introduced to avoid significant graph distortion:

\[
C = \frac{1}{m} \sum_i d(E_i)
\]

(2)

where \( d(E_i) \) is the cost of the difference of edge \( i \) before and after shifting the graph relative to the original positions. Considering the possible distortions of the nodes’ positions, the objective is to find the best graph matching according to the maximum total score of the matching:

\[
S_{\text{total}} = S_{\text{pha}} - \lambda C
\]

(3)
where $\lambda$ is a penalty parameter. The classification is determined by the maximum score over all the detectors (Max-Wins rule [24]).

2.2. Hierarchical Elastic Graph Matching

We propose to assign a hierarchy level to each node in a graph. The standard approach assumes that equal weights are given to every node in the bunch graph when determining the similarity function (Eq. (3)) when matching the graph with the target image. However, some features of the hand are more dominant than others, in terms of their discriminative power. Thus, the importance (hierarchy level) of each node should follow the similarity metric $\hat{S}_{\text{dba}}$ value. Boosting [25, 26, 27, 28] was adopted in our posture recognition algorithm to assign hierarchical values (weights) to the nodes within the bunch graph to maximize the recognition accuracy. These weights are, in practice, coefficients that maximize the discriminative function between feature vectors that are retrieved from specific positions in the hand, and positions in observations from a negative set of images (background).

Boosting is a general machine learning technique used to design, train and test classifiers by combining a series of weak classifiers to create a strong classifier. In the boosting technique, a family of weak classifiers forms an additive model in the form:

$$F(v) = \sum_{m=1}^{M} \hat{f}(v)$$

where $\hat{f}(v)$ denotes a weak detector, $v$ is a feature vector, and $M$ is the number of iterations (or number of weak detectors) to form a strong classifier, $F(v)$. When training the features, a set of weights is applied to the training samples and they are updated in each iteration. The update rule decreases the probability assigned to those features for which the current weak classifier makes a good prediction and increases the probability of the features for which the prediction is missed. The weights $\omega_i = e^{-\epsilon_i F(v_i)}$ for each training sample $i$ with class label $z_i$ are defined so the cost of misclassification is minimized by adding a new optimal weak classifier:

$$\arg\min_{\hat{f}} \sum_{i=1}^{N} \omega_i (z_i - \hat{f}(v_i))^2$$

Upon choosing the weak classifier and added to $F(v_i)$, the estimates are updated: $F(v_i) = F(v_i) + \hat{f}_m(v_i)$. Accordingly, the weights over the samples are updated by $\omega_i = \omega_i e^{-\epsilon_i \hat{f}_m(v_i)}$. In this paper, the gentleboost cost function [26] was used to minimize the error.

Each weak classifier serves as an indicative function for one feature value extracted on the node of the bunch graph. The similarity metric $\hat{S}_{\text{pH2}}$ is weighted by the coefficient vector $c$ that represents the discriminatory degree of each node (the relative proportion that the threshold is exceeded):
where $\mathcal{B}$ is the bunch graph with node index $k$, and $\hat{J}(\vec{x})$ is the jet computed from the target image taken at node vector $\vec{x}$. For different hand postures, the classifiers are trained separately. To collect the positive samples (true hits), feature vectors are extracted from the best bunch graph matching location. The region of interest in the target image is about 85% of the size of the image in the positive images. All nodes are allowed to shift their positions by four pixels in $\chi$ and $\gamma$ directions. Negative samples are feature vectors extracted by searching the best matching location of a bunch graph in the negative set of images from the training set (this method is broadly used to find negative instances that could potentially be recognized as true hits). Figure 1 shows the similarity response (Gabor-based kernel is used for this example) of a sample image when the similarity metric is computed with and without hierarchy assignment (the bunch graph is scanned over the entire image with an increment of 4 pixels).

Figure 1 illustrates the fact that the similarity response is more focused in one region when the hierarchy scheme is applied (the left image) than the response obtained without hierarchy (the right image). In other words, the similarity scores of the entire image exhibit a clear global maxima when hierarchy is applied. The focused response resulted from applying hierarchical graph matching provides fewer local maxima, thus leading to a more effective and reliable decision criteria.

$$\tilde{S}_{pha} = \sum_{k=1}^{N} c_k S_{pha}(\mathcal{B}^{(k)}, \hat{J}(\vec{x}^{(k)}))$$

Fig. 1. Similarity responses of bunch graph matched to an example image (a) with hierarchy; (b) without hierarchy
Figure 2 shows the importance of the nodes represented by a heat map (the edges are omitted to emphasize the nodes coloring system). Note that the importance (weights) of all nodes summed to one. Warm colors represent high hierarchy levels, while cold ones represent low hierarchy levels. In Figure 2, for those nodes with positions that blend with the background, lower weights are assigned (yellow color). On the other hand, those nodes over the rim of the hand are assigned higher weights (warmer colors) since they are more distinct from the background, and more descriptive of the hand shape.

3. Feature Extraction and Representation

3.1. Node Annotation Techniques

The bunch graph was created by selecting a set of nodes for each instance (image) of each posture in the dictionary set. Each chosen node represents the same landmark in the same gesture class in that set. The process of selecting nodes constituting the bunch graph is called “annotation”. Two types of nodes were annotated: edge nodes (nodes lying on the contour of the hand) and inner nodes (nodes lying inside the contour). In this paper, three methods to perform the annotation task were compared: manual, semi-automatic, and automatic. Semi-automatic and automatic approaches are compared with the standard manual annotation approach. In the manual annotation approach, nodes are selected manually by the designer. Therefore, the designer has to make sure that every landmark corresponds roughly to the same point in all the images in the dictionary set of that gesture class. On the other hand, the other annotation approaches suggested allow a less supervised node alignment regime. In the automatic method, the inner nodes are automatically selected by picking those positions where the Harris corner detector [29] responds more sharply, (e.g. highly textured regions within the hand). The semi-automatic approach follows the same procedure as the automatic approach except that the user is allowed to correct manually those detected landmark points by an offset. All the methods rely on the fact that the contour in every image was first annotated manually for precise alignment.
The rationale for automatic and semi-automatic annotation methods is to enable a more efficient and expeditious nodes’ alignment in the dictionary set. To this end, the main difference among these three approaches is the manner on which the nodes are selected within the hand region. For the two methods (automatic and semi-automatic), one reference graph is chosen and the remaining five graphs are aligned with respect to it. A linear assignment problem (LAP) is applied to find the points in each graph in the bunch that better correspond to those points in the reference graph. The objective is to find the least displacement pairs of nodes from a larger set of candidates of the current graph. This is a minimization problem which formulation is provided in Eq. (7 and 8):

\[
\min_x \sum_{j=1}^{N} \sum_{j=1}^{N} d_{ij} z_j
\]

s.t. \[\sum_{j=1}^{N} z_j = N_1, z_j = 0 \text{ or } 1\]

where \(d_{ij} = \| (x_i^1,y_i^1) - (x_j^2,y_j^2) \| \) is the Euclidian distance between the nodes \((i = 1...N_1, j = 1...N_2)\). \((x_i^1,y_i^1)\) is the node of the reference graph, and \((x_j^2,y_j^2)\) is the node of the graph to be matched. The detailed process is summarized in following Algorithm table.

<table>
<thead>
<tr>
<th>Algorithm: Node Annotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input: Edge nodes (\mathcal{X}) of images from dictionary set (B);</td>
</tr>
<tr>
<td>for all (I \in B) do</td>
</tr>
<tr>
<td>// given (I) as the reference graph with outer nodes (\mathcal{X}^i)</td>
</tr>
<tr>
<td>(\mathcal{P}_{\text{ref}} \leftarrow \text{HarrisCornerDetector}(I))</td>
</tr>
<tr>
<td>(B_r \leftarrow B - {I})</td>
</tr>
<tr>
<td>for all (J \in B_r) do</td>
</tr>
<tr>
<td>// (J) as the graph to be aligned with outer nodes (\mathcal{X}^j)</td>
</tr>
<tr>
<td>(\mathcal{P}_{j} \leftarrow \text{HarrisCornerDetector}(J))</td>
</tr>
<tr>
<td>([t, r, s] \leftarrow \text{Alignment}(\mathcal{X}^i, \mathcal{X}^j)) // translation, rotation, and scaling</td>
</tr>
<tr>
<td>(\mathcal{P}<em>{\text{transf}} \leftarrow \text{PointTransformation}(\mathcal{P}</em>{j}, t, r, s))</td>
</tr>
<tr>
<td>([\mathcal{P}, \mathcal{C}] \leftarrow \text{LAP}(\mathcal{P}<em>{\text{ref}}, \mathcal{P}</em>{\text{transf}})) // Linear assignment problem returns the assigned nodes and optimized costs</td>
</tr>
<tr>
<td>end for</td>
</tr>
<tr>
<td>(C_i \leftarrow \text{sum}(\mathcal{C})) // summation of total cost when (I) as the reference graph</td>
</tr>
<tr>
<td>end for</td>
</tr>
<tr>
<td>(k \leftarrow \text{argmin}_{i} {C_i}) // best reference graph with minimum total cost</td>
</tr>
<tr>
<td>(I_k) = optimal reference graph, save all the annotations.</td>
</tr>
</tbody>
</table>

To evaluate the effectiveness of the proposed annotation methods with the existing method (which is manual), four metrics considering different aspects of graph similarity are used: (1) benefits entailed in nodes’ matching. Relative displacements of the nodes with respect to each other in the different graphs result in a
matching ‘cost’. The benefit function (oppose to the cost function) is the difference of maximum likely cost and the actual matching cost. The Euclidean distances of each pair of nodes are summed up as the total matching costs; (2) recognition accuracy: the bunch graph is annotated, built and used to classify the postures; (3) normalized cross-correlation: the cross-correlation is computed between the matching node pairs of the reference graph and the ones aligned to; (4) transformation consistency: errors resulting from affine transformation disparities between the reference graph and the ones aligned to it [30] (see Eq. (9) below). Thus consistency is proportional to the amount of alignment errors.

\[ E = [\Omega^1 - R^* \Omega^2 - t^T]^T [\Omega^1 - R^* \Omega^2 - t^T] \]  

where \( R^* \) is the optimal rotational (\( \Theta \)) and scaling matrix (\( s \)) (least-square minimization approach is used to reach the optimal) applied to \( \Omega \):

\[ R^* = \begin{bmatrix} s \cos \Theta & -s \sin \Theta \\ s \sin \Theta & s \cos \Theta \end{bmatrix} \]  

where \( \Omega \) is the vector representation of the coordinates the points in each image \( (x_i^l, y_i^l) \), \( i \in \{1, 2\} \). Also, \( t^* \) is the optimal translation parameter.

3.2. Gaussian-Based Filter Bank

Gabor-based filter bank was used in our previous work on hand posture classification [16]. In this paper, a battery of different filter banks are adopted and compared to assess recognition accuracy of each filter bank. The first filter bank consists of 3 Gaussians, 4 Laplacian of Gaussians (LoG), and 4 first derivatives of Gaussians, thus producing a 11–dimensional filter bank. The three Gaussian kernels scaled as \( \sigma = 1, 2, 4 \). The four LoGs are scaled as \( \sigma = 1, 2, 4, 8 \). The four first order derivatives of Gaussians are divided into the two \( x \) and \( y \) directions, each with two different values of \( \sigma (\sigma = 2, 4) \) [31]. Therefore, each node in each image produces a 11–dimensional feature vector. Another filter bank variant is obtained by replacing the three Gaussian kernels with three Laplacians (with \( \alpha = 0.2, 0.4, 0.8 \)). To this end, we can create a feature vector \( v \in R^n \) for each annotated node by convolving the image \( I \) with the each filter bank \( \{F_1, F_2, ..., F_n\} \) as follows:

\[ v(\vec{x}) = \{F_1 * I(\vec{x}), F_2 * I(\vec{x}), ..., F_n * I(\vec{x})\}; \forall \vec{x} \in B \]  

where \( B \) is the bunch graph.

3.3. Gradient-Based Representations: Histogram of Oriented Gradient

Histogram of Oriented Gradient (HOG) was first described in [17] to detect pedestrians in static images. HOG has strong capability to characterize the local object appearance and shape. The implementation of HOG encodes the image pixels to construct a histogram according to the pixels’ gradient magnitude and orientation
within a small spatial region (cells). To account for illumination and contrast variation, the cells are grouped into larger spatial regions (blocks) and normalized. Once the Histogram of Oriented Gradient (HOG) features are created, discriminative methods can be used to classify those features. In this paper, a patch \( w(x) \) is extracted at the annotated node from the gradient image (Sobel operator was used). A patch is defined as a window centered on the annotated node. Within a patch, the gradient magnitudes \( G(w(x)) \) and gradient orientations \( \alpha(w(x)) \) for each pixel are computed, and weighted voting is performed for each orientation. Each pixel within the patch votes for a given orientation (a histogram bin) based on its gradient magnitude. The histogram is evenly divided into 8 bins so that orientation ranges between 0 to 360 degrees in spaces of 8 regions.

The histogram values are normalized by the total energy within the patch to avoid sensitivity to illumination. Therefore, each node in each image produces a 8–dimensional feature vector. The HOG feature vector \( v \in \mathbb{R}^n \) for each annotated node is created as follows:

\[
v(x) = H(G(w(x), b), \alpha(w(x), b)) \quad \forall x \in B
\]

where \( H(\cdot) \) is the operator assigning the corresponding bin on the histogram for each pixel in the gradient patch, \( B \) is the bunch graph including each annotated node, and \( b \) is the vector representing the bin index. The features are extracted at the annotated nodes, which are from the silhouette of the hand and the interior parts of the hand. Thus effect of the parts in the background can be excluded. The process of extracting histogram features on the target posture is illustrated in Figure 3.

![Fig. 3. Extracting the orientation-based histogram features on hand posture (example nodes)](image)

4. Experimental Results and Analysis
The dataset used to validate the proposed algorithm is the Triesch online hand posture dataset [32]. The dataset consists of 10 different hand gestures against three different types of backgrounds; performed by 24 subjects. A total of 710 128x128 pixels grey-scale images (210 light background, 239 dark background, and 261 complex background images) are used. Each bunch graph was created by selecting two instances of a given posture performed by three subjects against light and dark backgrounds, resulting in six instances in each posture. Thus, the dictionary of a bunch graph includes 60 images. The bunch graph geometry was averaged from the six instances for each posture. The remaining 650 images were used for the training and testing phases. The following experimental results show the classification performance of the various approaches discussed earlier, when applied to the full dataset of 650 images. Examples of the bunch graphs matching performance, for each of the 10 hand postures is presented in Figure 4. Note that each image was scanned in increments of 4 pixels in the horizontal and vertical directions.

![Fig. 4. 10 classes of sample hand gesture images after the matching process](image)

The cyan lines connecting inner nodes and edge nodes are used to measure the allowed distortion cost of each the node. The colors (from warm to cold) were used to represent nodes’ hierarchy levels.

4.1. Comparison of using different Filter banks on Hand Gesture Classification

In Figure 5, the Receiver Operating Characteristic (ROC) curves using four different filter banks in the HEGM classification procedure are shown. The standard manual annotation approach was used to create the bunch graph for the four scenarios. Each ROC curve represents the performance of a gesture class detector when applied to the whole dataset. This ROC curve was generated using 5-fold cross-validation and a range of thresholds on the classification scores. The true positive rates are determined by comparing the classification score with a threshold. False positives were found when observations was classified as a gesture, when, in fact, they belonged to a different class. The displayed ROC curves show the relationship
between the true positives and false alarms among the 10 gesture classes. The highest average recognition accuracy (99.85%) when the HOG feature representation method was used, while the lowest average recognition accuracy (85.38%) when the Gaussian-based filter bank was applied. The experimental results also show that the use of the Laplacian-based filter bank achieves better results (95.85%) than using the state-of-art Gabor-based filter bank (92.62%), while the false negative rate is higher (16.42% vs. 12.32%) for the LoG.

The hand gesture classification performance was further evaluated using another metric. The classification class with the maximum score over the 10 classifiers was chosen when classifying an arbitrary gesture (Max-
Wins rule). This metric always result in a single classification (correct or incorrect), and no false positive cases. If the maximum score points to the incorrect class, then we said that the gesture was misclassified, (accounted as confusion). The confusion matrix (see Figure 6) was created by comparing the scores obtained by each classifier applied to a given testing image, and selecting the maximum score from all 10 classifiers. Following the same procedure, the results are evaluated by comparing the use of the four different filter banks. The average accuracy of correct classification over the confusion matrix using the Gabor-based kernel reached 97.08%, which is higher than the use of Laplacian-based filter and Gaussian-based filter banks. The highest average recognition accuracy among the four filter banks was 100% for the HOG feature representation method. It is also shown that the classification results using HOG feature representation is not affected by the image background type (uniform light background, uniform dark background, and complex background). Additionally, the recognition accuracy is decreased to 98.77% when the patch size used for HOG feature extractions is increased by twice. Since the patch is the window centered on the annotated node, increasing the patch size could result in windows overlapped among the nodes.
4.2. Performance on Different Annotation Techniques

To assess the performance of each annotation technique used to create the bunch graph, we adopted four metrics: recognition accuracy, normalized cross-correlation, matching benefit, and transformation consistency, as the performance measures. The recognition performance was affected by the position of the candidate nodes (highly texted regions) inside the hand since the detection scheme was used in the automatic and semi-automatic methods. Additionally, only the semi-automatic method allowed nodes to be adjusted manually after detection. The four metrics that indicate the performance measures when using the three different methods to annotate the nodes in the bunch graph are illustrated in Figure 5. First, the classification method used followed the procedure described in [15]. When the semi-automatic approach was adopted to test with light and dark background images, the recognition accuracy (92.12%) was higher than the other two methods (90.91%, and 89.26% for manually and automatically, respectively). Second, the normalized cross-correlation of the annotated nodes between the images was highest for the manual method (a score of 0.99), while the value of semi-automatic method achieved a score of 0.97. Third, the normalized matching benefit was the lowest for the automatic technique. In other word, higher costs incurred for the automatic technique due to the inconsistency of the nodes’ position among the graphs. Finally, for the same reason, the normalized transformation consistency was also the lowest for the automatic technique. Despite the fact that matching benefits and transformation consistency of semi-automatic approaches are marginally higher than those using the manual method, these measures are considerably lower than those using the automatic method. Therefore, a trade-off exists when the recognition accuracy and the speed of creating the annotation are considered simultaneously. The former is also expressed by the high matching cost and consistency. The proposed semi-automatic technique can be an efficient annotation method for creating the bunch graph for a given recognition accuracy.
5. Conclusion

This paper has presented the proposed enhanced graph-based approach incorporating the concept of nodes hierarchy (HEGM) levels to successfully implement the hand gesture recognition task. The HEGM algorithm was further tested with several feature extraction and representation approaches to explore its flexibility and robustness. The use of HOG feature representation when applied to the nodes combined with the HEGM algorithm resulted in hand posture successful classifications, with a gesture recognition accuracy of 99.85% on average. The HEGM algorithm has shown a significant improvement on classification performance compared to the other methods for feature creation. The merit of this approach is to utilize the discriminatory capabilities of the nodes for each gesture with respect to the remaining gestures. The hierarchy based approach improved the recognition and computation performance by allocating most of the computational resources on those more important features.

Semi-automatic and automatic annotation techniques were proposed to allow convenience and flexibility in the nodes’ selecting process on the posture image. The goal was to achieve consistency within annotated images that are part of the dictionary set of the same posture. Among the three annotation techniques, semi-annotation approach outperformed the other two with the highest recognition accuracy. Although the remaining metrics show that manual annotation technique has the highest consistency between the different images of the same posture, semi-automatic annotation delivers a comparable performance.

Future work includes exploring the feasibility of extending the HEGM algorithm to incorporate depth and color information. One possible direction is using the depth information to have a better initial approximation of the region of interest so that the HEGM matching process is computed faster. The overall computation time

![Fig. 6. Performance measures for different annotation techniques.](image_url)
will be reduced for smaller search regions. Additionally, implementation of this algorithm in parallel processing is another focus area.

References