Human Computer Interaction Based on Hand Gesture Ontology

LAWRENCE Y. DENG[1], DONG-LIANG LEE[2], YI-JEN LIU[3] and NICK C. TANG[3]
[1] Department of Computer Science and Information Engineering, St. John's University
[2] Department of Information Management, St. John's University
[3] Department of Computer Science and Information Engineering, Tamkang University
499, Sec. 4, Tam King Road, Tamsui, Taipei, Taiwan, R.O.C

Abstract: Hand gesture recognition is a new challenge to convey information such as input data or to control devices and applications such as computers, games, PDAs, browsers, cell phones, information appliances and MP3 audio players, etc. Image-based techniques detect a gesture by capturing pictures of a user's hand signs during the course of a gesture, and to recognize what user would like to do. In this paper, we proposed a vision-based gesture recognition system. This system support hand gesture interaction with referring to the hand gesture ontology for improving efficiency.
Key-Words: Gesture Recognition, Ontology, Data Mining, Case-based reasoning, Behavior Prediction

1 Introduction

Current hand gesture recognition system support retrieval using low-level features, such as color, texture, and motion. However, low-level feature often have lower meaning for users, who prefer to identify gesture sign by using high-level semantic concepts. In this paper, we proposed hand gesture ontology for shortening the semantic gap.

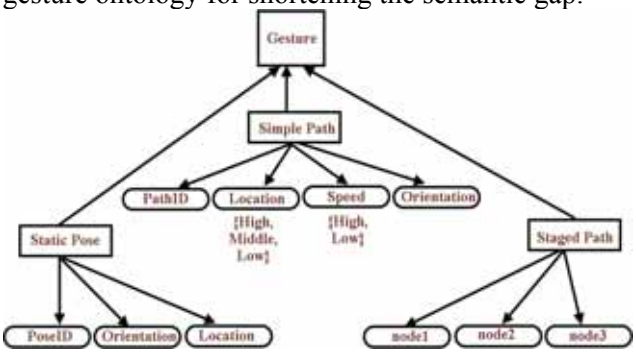

Figure. 1 Hand Gesture Ontology

There are three primitive hand gesture types in the hand gesture ontology: Static Pose, Simple Path, and Staged Path (As figure 1 showed). Static Poses are rigid hand postures that do not depend on the movement of the hand. Static Poses contains three attributes: PoseID, Orientation, and Location. Simple Paths are outlined trajectories representing a given hand gesture motion, that similar to primitive shapes, such as triangle, circle, square, or ellipse. Simple Paths contains four attributes: PoseID, Orientation, Location, and Speed. Staged paths are vector based trajectories, which represent the corresponding plotlines. The corresponding plotlines rely on localized hand postures to define the control points which are used to divide the hand gesture into a set of successive line segments. In this paper, we presented a system that provided users to issue instructions to the computer system by gesturing with hand.

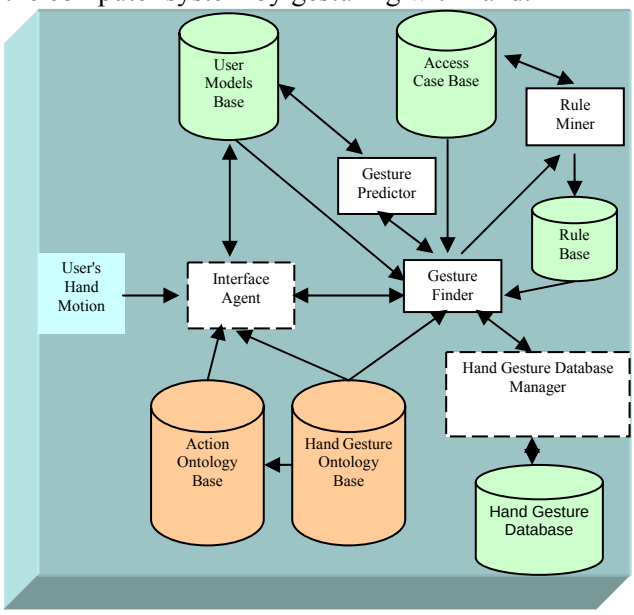

Figure. 2 System Architecture

By using the image processing technologies, the Interface Agent extracts and describes the hand contours from the raw images, which are captured from the camera (As figure 2 showed). Then it delivers this information to Gesture Finder. Gesture Finder aimed to search the related hand gesture from the hand gestures' library through several mechanisms and returns the results to Interface Agent as following steps:.

(1) The Gesture Finder checks the User Model Base to find the related hand gesture. User's gesture behavior was stored in the User Model Base for Gesture Predictor to predict next possible hand gestures. The predict results were saved into the User Model Base for Gesture Finder to check.

(2) If there were no related hand gestures that can be found in the User Model Base, the Gesture Finder will turn to the Case-Based Reasoning process with the Access Case Base.

(3) If the Gesture Finder still can't find relative hand gestures, it will turn to the Rule-Based Reasoning process with the Rule Base.

(4) Anyway, if the Gesture Finder can't find any related hand gestures, it will trigger the Hand Gesture Database Manager to find the related hand gesture from the Hand Gesture Database directly.

The search results will be considered as some cases. The system will store each case into Access Case Base. Rule Miner extracts rules from Access Case Base and store these rules into Rule Base. At the same time, Gesture Predictor predicts possible next hand gestures, and store prediction result into User Model Base.

After searching for the related hand gesture from the hand gestures library, the Gesture Finder returns the result to the Interface Agent. Interface Agent will transform it into corresponding instructions according to the rules, which was stored in Action Ontology Base. At last, Interface Agent delivers these instructions to operating system and to trigger user-specified action. Hand Gesture Database was designed according to the hierarchy defined in Hand Gesture Ontology. It provided a robust access mechanism for gesture-instruction conversion to enhance efficiency of gesture searching. Hand Gesture Database Manager integrated the information and then stored in Hand Gesture Database efficiently for gesture searching.

Access Case Base - user issues instructions to the computer system by gesturing with the hand in front of a single video camera. Interface Agent transforms the captured images into a set of metric values and then passes these values to Gesture Finder. Gesture Finder searches for the relative hand gesture from the hand gestures library through several various mechanisms and returns the search result to Interface Agent. The set of metric values and the relative hand gesture will be considered as a case and stored in Access Case Base for case-based reasoning and rule-based reasoning.

Rule Base - rules extracted from Access Case Base that were stored in Rule Base. With Rule Base, Gesture Finder can determine if there are relative hand gestures in the hand gesture database.

User Model Base — User's profile was recorded in the User Model Base. User's gesture behavior was also stored in the User Model Base for Gesture Predictor to predict a user's possible hand gesture. The predict results were saved into the User Model Base for Gesture Finder to check.

Interface Agent is in charge of human-computer interface. In the aspects of Input, Interface Agent gets streaming media from a single video camera to capture hand motion of the user. First, Interface Agent subtracts background from captured image and deletes inadequate pixel information. Then the region of user's hand will be detected and its contour will be described. The important metrics information can be extracted from these contour characteristics. Finally, Interface Agent delivers this information to Gesture Finder to identify whether the hand motion is a hand gesture in the pre-defined hand gestures' library. In the aspects of Output, Interface Agent transforms the hand gesture into instructions according to formal instruction format, and delivers the instructions to Command Interpreter of the operating system to trigger user-specified actions.

Hand Gesture Ontology Base provides relative ontological definition for each hand gesture in the Hand Gesture Database in order to promote the efficiency of gesture searching. Action Ontology Base provides Interface Agent with standard information to transform the hand gesture into instructions, so as to trigger user-specified actions. Gesture Finder leads the whole process of gesture searching. After receiving metrics information delivered from Interface Agent, Gesture Finder searches for the relative hand gesture from the hand gestures' library through several various mechanisms. Gesture Finder check User Model Base, Access case Base and Rule Base in turn. If Gesture Finder can't find any related hand gestures, Gesture Finder will start Hand Gesture Database Manager to find the relative hand gesture in the Hand Gesture Database. Rule Miner mines rules from Access Case Base to promote efficiency of gesture searching. Gesture Predictor extracts user's gesture sequence according to user's manipulation history and store in User Model Base. Gesture Predictor then predicts next possible gesture and stores the gestures in the User Model Base.

2 Interface Agent

Interface Agent" is human-computer interface. Interface Agent gets streaming media from a single video camera to capture hand motion of the user. Firstly, Background Subtracter subtract background from captured image and delete inadequate pixel

information. Then Color Tracker detects a region surrounding the hand in each successive frame. Hand Extracter extracts the realistic hand contour by morphological smoothing. Gesture Analyzer calculates each feature value of hand contour and extracts the important metrics information from these feature values. Interface Agent delivers this information to Gesture Finder to identify whether the hand motion is a hand gesture in the pre-defined hand gestures' library and find the relative hand gesture. Finally, Gesture-Action Converter transforms the hand gesture into instructions according to formal instruction format, and delivers the instructions to Command Interpreter of the operating system to trigger user-specified actions.

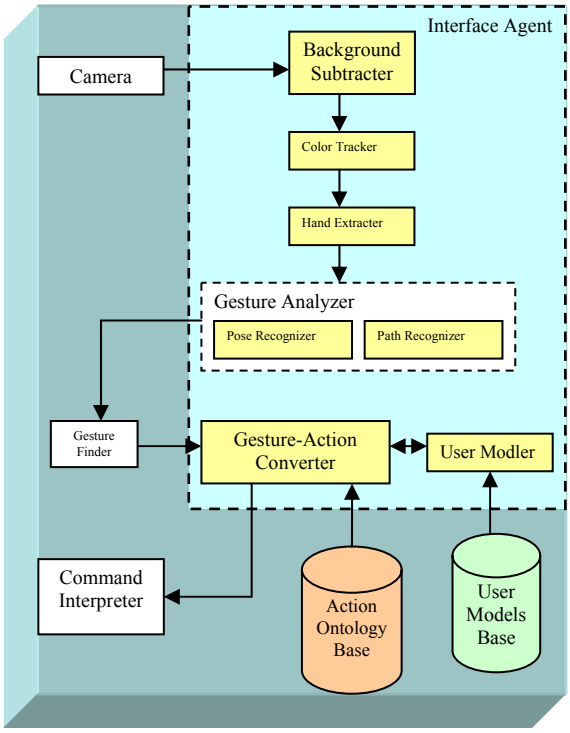

Figure 3. Interface Agent Scenario

In figure 3, the Background Subtracter calculates the variations of actual frame pixels' values for background subtraction. Color Tracker use either CamShift algorithm or Kalman Filter to detect a region surrounding the hand in each successive frame[9][10]. Hand Extracter do some further processing like morphological smoothing after color tracking in order to extract the realistic hand contour. Pose Recognizer computes each feature value of hand contour and use shape context to describe hand contour [11] [12][13]. Path Recognizer makes use of the characteristics of elementary geometric shapes, such as triangles, rectangles, circles, ellipses and lines, to determines the shape of trajectory of moving hand [14][15]. Gesture-Action Converter transforms the hand gesture into instructions according to the rules stored in Action Ontology Base and then deliver these instructions to Command Interpreter of operating system to trigger user-specified action. User Modeler determines whether or not to issue instructions to attach appropriate arguments to the instruction according to the users preference stored in User Models Base.

3 Gesture Matching

Gesture Finder searches the related hand gesture from the hand gestures library through some mechanisms and returns the results to Interface Agent. The Gesture Finder firstly checks the User Model Base to find the related hand gesture. If there are no related hand gestures that can be found in the User Model Base, the Gesture Finder will turn to the Case-Based Reasoning process within the Access Case Base. If the Gesture Finder still can't find relative hand gestures, it then turns to the Rule-Based Reasoning process with the Rule Base. Anyway, if the Gesture Finder can't find any related hand gestures, it will trigger the Hand Gesture Database Manager and to find the related hand gesture from the Hand Gesture Database directly. After searching for the related hand gesture from the hand gestures library, the Gesture Finder returns the result to the Interface Agent.

3.1 Case-Based Reasoning

The system presented in this paper is a computing platform to support gesture interaction. Users can issue instructions to the computer system by gesturing with the hand in front of a single video camera. Interface Agent utilizes several image processing technologies to extract and describe the contours of the hand. Interface Agent delivers this information to Gesture Finder. Gesture Finder searches for the relative hand gesture from the hand gestures' library through several various mechanisms. The search result will be considered as a case. The system will store this case into Access Case Base. The system use Access Case Base as a case base to find the relative hand gesture through Case-Based Reasoning.

Case Retriever retrieves referable cases from Access Case Base according to metrics information delivered from Interface Agent. Case Reuser returns candidate cases retrieved by Case Retriever to Gesture Finder. If there is a hand gesture exist in one of candidate cases, which matches up with user's hand gesture, Gesture Finder will return the search result to Interface Agent. Case Reuser will compare the similarity between reference case and user's hand gesture. If they give high similarity, system don't have to spend time and space to add user's hand

gesture to Access Case Base, else the search result will be stored in the Access Case Base.

Case Retainer will save the search result as a case and then save in the Access Case Base. Besides adding cases, Case Retainer should delete invalid or inaccurate cases and combine cases which give high similarity in order to avoid the number of cases being too large and slowing down the speed of case searching. Finally, the Similarity Metrics Base provides the rules to compute the similarity between cases.

3.1.1 Case Retriever

There are three steps to retrieve referable cases from Access Case Base according to the metrics information delivered from Interface Agent:

Definition:

$F = \{f_1, f_2, \cdots, f_m\}$: set of all features of hand gesture

$C = \{c_1, c_2, \cdots, c_n\}$: set of all cases in case base

G : user's hand gesture

C_r : set of cases retrieved as candidate case

Step 1: Retrieve cases which contain the same or more gesture features.

```
For all c ∈ C do begin
 if Exist f ∈ F s.t. c.f.value = G.f.value then begin
  Add c to Cr
 end
end
```

Step 2: Delete cases which contain more than one different feature.

```
For all c ∈ Cr do begin
 count = 0;
 For all f ∈ F do begin
  if c.f.value != g.f.value then do begin
   count++
   if count > 1 then do begin
    delete c from Cr
    break
   end
  end
 end
end
```

Step 3: Delete the dissimilar cases according to the type of ontology Relationship.

In Step 3, the case and user's hand gesture have only one different feature value. We select a measure according to the feature's Ontology Relationship:

(1)Mutually exclusive—delete the case.

(2)Downward-compatible—if the feature value of the case is higher than the feature value of user's hand gesture, delete the case.

(3)Upward-compatible—if the feature value of the case is lower than the feature value of user's hand gesture, delete the case.

(4)Threshold— the difference between the case and user's hand gesture must be lower than a threshold, else delete the case.

```
For all c ∈ Cr do begin
 if Exist f ∈ F s.t. c.f.value != G.f.value then
  if F.Relationship = "Mutually-exclusive" then delete c from Cr
  elseif F.Relationship = "Downward-compatible" then do begin
   if c.F.value > G.F.value then delete c from Cr
  end
  elseif F.Relationship = "Upward-compatible" then do begin
   if c.F.value < G.F.value delete c from Cr
  end
  elseif F.Relationship = "Thresholding" then do begin
   distance = ||G.F.value - c.F.value||
   if distance > threshold.upbound or distance < threshold.lowerbound then
    delete c from Cr
  end
end
```

3.1.2 Case Reuser

Case Reuser delivers candidate cases that retrieved by Case Retriever to the Gesture Finder. If there is a hand gesture exist in one of candidate cases that matches up with user's hand gesture, Gesture Finder will return the search result to Interface Agent. Case Reuser will check whether a user's hand gesture can be directly stored in Access Case Base according to the similarity of the reference case against user's hand gesture. If the reference case and user's hand gesture have high similarity, system don't have to spend time and space to add user's hand gesture to Access Case Base, else the search result will be considered as a case and stored in the Access Case Base.

3.1.3 Case Retainer

Gesture Finder searches for the relative hand gesture from the hand gestures' library. The search result will be considered as a case. Case Retainer will save the case in the Access Case Base. Besides adding cases, Case Retainer must delete invalid or inaccurate cases in order to avoid the number of cases being too large and slowing down the speed of case searching. When Case Retainer saves a new case to the Access Case Base, the survival value of the case will be calculated at the same time.

Survival Value is defined as

Sim(c,a) = $|S_c \cap S_a| / |S_a|$
Survival Value of New Case = ***Sim***(c,a) × Survival Value of reference case

Each case in Access case base has its own Survival Value to represent its importance in the base. If a case is referred frequently and useful for efficiency of gesture searching, survival value of the case will be increased, else the value will be decreased. A case will be deleted from Access case base when its survival value decreased to certain value.

- **Case Aging**

```
For all c ∈ C do begin
  if c ∈ Cr then
    c.Survival += α;
  else
    c.Survival -= α;
end
```

- **Case Deletion**

```
For all c ∈ C do begin
  if c.Survival < κ then delete c from C
end
```

- **Case Addition**

```
count = sum = 0;
For all c ∈ Cr do begin
  sum += c.Survival × sim(G,c);
  count ++;
end
if count > 0 then
  G.Survival = sum / count;
else
  G.Survival = κ + α × β;
Add G to C
```

4 Association Mining

4.1 Rule Mining

Rule Miner extract the association among the set of features from Access Case Base. Each case is considered as a transaction. Association among the items can be extracted through algorithms for Mining Association Rules. Because the source of data mining is from Access Case Base, when the base is accessed, we should consider whether the job of data mining must be started. When the number of new cases increases to certain value, Rule Miner will be started to do the job of data mining. Because old cases can't provide update information, Rule Miner's start time has nothing to do with case deletion.

Three step of rule mining:

Mining Frequent Itemsets - A frequent Itemset is an itemset which have support greater than the user-specified minimum support (called minsup). The support for an itemset is the number of transactions that contain the itemset[16]. Mining association rules - A association rule is an itemset which have support and confidence greater than the user-specified minimum support (called minsup) and minimum confidence (called minconf) respectively. The rule X => Y holds in the transaction set D with confidence c if c% of transactions in D that contain X also contain Y. Add association rules to rule base - After mining association rules, we can use SQL language to decrease rules.

4.2 Gesture Prediction

We use the concept of Sequential Pattern to predict user's hand gesture[17]. Hand gestures that user performed before, and perform time are recorded in the User Model Base. Gesture Predictor mines sequential patterns from these records.

Gesture-based Mining - Each user's hand gesture is considered as an item. Different hand gesture represents different item. Hand gestures and their time information can be provided for prediction rule mining. Assume that (A，B，C) is a sequential patterns, where A, B, C is different hand gestures. We can calculate the confidences of A → B and B → C. If confidence is greater than minimum confidence, the rule can be considered as a legal rule. During data mining process, we consider the whole hand gesture and the search result as a transaction. When Gesture Predictor heard that there is a hand gesture A, Gesture Predictor will inform Gesture Finder that hand gesture B is considered as next possible hand gesture. Gesture Finder will get the index of hand gesture B from the hand gestures' library through Hand Gesture Database Manager and store the index to User Model Base. If there is another rule A → D, in order to prevent too much load for Gesture Finder, we don't predict that B and D are next gesture at the same time. As Case Retainer determine whether a case must be removed from access case base by checking its survival value, User Modler determine whether a prediction rule must be removed by checking its prediction accuracy.

Feature-based Mining - Each feature composed the hand gesture is considered as an item.

5 Conclusion

In this paper, we have proposed a solution for hand gesture recognition system. We have used hand gesture ontology to construct a human-computer interface that support hand gesture interaction. We have introduced several search mechanisms to improve the efficiency of gesture matching. Consequently, the number of accessing hand gesture database can be decreased, and also the workload of

the database can be graded down. While the strategies presented in this paper are specified to recognize the hand gesture is an essential ideal we want to convey here. From this point of view, future research conducted on widely information appliances, industry equipment or even military facilities.

References:

[1] Geer, D. "Will gesture recognition technology point the way?" IEEE Computer Volume 37, Issue 10, Oct. 2004 Page(s):20 - 23

[2] Xingquan Zhu, Xindong Wu, Ahmed K. Elmagarmid, Zhe Feng, Lide Wu. Video Data Mining: Semantic Indexing and Event Detection from the Association Perspective - Appendices. IEEE Trans.Knowl. Data Eng. 17(5): (2005)

[3] Staab, S., Gomez-Perez, A., Daelemana, W., Reinberger, M.L. and Noy, N.F., "Why evaluate ontology technologies? Because it works!", Intelligent Systems, IEEE, vol. 19, no. 4, pp. 74-81, 2004.

[4] Cavazza, M., Charles, F., Mead, S.J., Martin, O., Marichal, X., Nandi, A.: Multimodal actingin mixed reality interactive storytelling. IEEE-Multimedia 11 (2004) 30–39

[5] J.M.S. Dias P. Nande N. Barata A. Correia. "OGRE - open gestures recognition engine". IEEE Computer Graphics and Image Processing, 2004. Proceedings. 17th Brazilian Symposium on 17-20 Oct. 2004 Page(s):33 - 40

[6] H.K. Lee and J. H. Kim. "An hmm-based threshold model approach for gesture recognition". In IEEE Transaction on Pattern Analysis and Machine Intelligence,volume 21, pages 961–973, 1999.

[7] M. C. Su, 2000, May, "A Fuzzy Rule-Based Approach to Spatio-Temporal hand Gesture Recognition," in IEEE Transactions on Systems, Man and Cybernetics, Part C, on Volume 30, Issue 2, May 2000 Page(s):276 - 281

[8] Larry S. Davis Thanarat Horprasert, David Harwood. "A statistical approach for real-time robust background subtraction and shadow detection". Technical report, Computer Vision Laboratory University of Maryland, 1999.

[9] Gary R. Bradski. "Intel open source computer vision library overview", 2002.

[10] N. Liu and B.Lovell. "MMX-Accelerated Real-Time Hand Tracking System". Proceedings of IVCNZ 2001. pp. 381-385

[11] S. Belongie, J. Malik, and J. Puzicha. "Shape context: A new descriptor for shape matching and object recognition". In NIPS, pages 831–837, 2000.

[12] S. Belongie, J. Malik, and J. Puzicha, "Shape Matching and Object Recognition Using Shape Contexts,"IEEE Trans. Pattern Analysis and Machine Intelligence, vol.24,no. 4, pp. 509-522,Apr. 2002.

[13] Eng-Jon Ong; Bowden, R. "A boosted classifier tree for hand shape detection". Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on 17-19 May 2004 Page(s):889 - 894

[14] Manuel J. Fonseca Joaquim A. Jorge. "A simple approach to recognize geometric shapes interactively". Technical report, Departamento de Engenharia Informática, IST/UTL, 1999.

[15] Ajay Apte, Van Vo, and Takayuki Dan Kimura. Recognizing Multistroke Geometric Shapes: An Experimental Evaluation. In Proceedings of the ACM (UIST'93), pages 121{128, Atlanta, GA, 1993.

[16]R. Agrawal and R. Srikant, "Fast Algorithm for Mining Association Rules," Proc. of the 20th International Conference on Very Large Databases, pp.487-499, 1994.

[17] R. Agrawal and R. Srikant, "Mining Sequential Patterns," Proc. of the 11th International Conference on Data Engineering, pp.3-14, 1995