A MULTI-MODAL HUMAN-COMPUTER INTERFACE:
COMBINATION OF GESTURE AND SPEECH RECOGNITION

Minh Tue Vo  Alex Waibel
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213-3890, U.S.A.
E-Mail: tue@cs.cmu.edu, ahw@cs.cmu.edu

ABSTRACT
Multi-modal interfaces can achieve more natural and effective
human-computer interaction by integrating a variety of signals,
or modalities, by which humans usually convey information.
The integration of multiple input modalities permits greater
expressiveness from complementary information sources, and
greater reliability due to redundancies across modalities.

This paper describes a text editor developed at Carnegie Mel­
lon, featuring a multi-modal interface that allows users to
manipulate text using a combination of speech and pen-based
gestures. The implementation of this multi-modal text editor
also illustrates a framework on which more general joint inter­
pretation of multiple modalities can be based.

KEYWORDS: Multiple modalities, multi-modal interface, ges­
ture recognition, word spotting, semantic-fragment grammar,
natural networks.

INTRODUCTION
Human beings communicate with each other using a variety of
signals such as speech, pen, gesture, eye-contact, facial expres­
sion, etc.; it is this combination of different modalities that
gives human communication a naturalness and flexibility pre­
sently unequaled in human-computer interaction. A user study
at Carnegie Mellon University [2] has shown that in interacting
with computer systems, people prefer a combination of speech
and gestures over speech or gestures alone. Different input
modalities can complement each other, allowing greater
expressiveness than each modality on its own. For example, in
a text-editing session a user may delete a paragraph simply by
circling the text and saying “Delete” at the same time. The
modalities can also enhance each other when similar concepts
are expressed in many different ways; this redundancy can be
exploited to increase reliability. Noise may hamper the recogni­
tion of a spoken “Delete” command, but the system can
recover if it realizes that the user also drew a cross on top of
some text to emphasize the “Delete” concept. Such a system
capable of accepting and integrating information from multiple
sources would be very likely to gain user acceptance because of
its flexibility and natural feel.

Some of the human communication modalities (e.g. speech)
have been extensively investigated, but mostly in isolation.
Although researchers have been aware of the advantages of
integrating multiple modalities for some time, practical imple­
mentations of multi-modal systems have been slow to emerge
because of a lack of understanding of how to combine the dif­
f erent input signals to achieve maximum joint benefit. In the
present paper, we describe a text editor developed at Carnegie
Mellon, capable of recognizing gestural and speech inputs, and
combining these information sources to determine the action to
carry out. This joint interpretation is performed using a flexible
frame-based approach suitable for general multi-modal seman­
tic interpretation.

GESTURE RECOGNIZER
In the context of our editing task, a gesture is defined to be any
symbol or mark drawn using a stylus on a digitizing tablet. Our
text editor currently supports 8 gestures (see Table 1).

<table>
<thead>
<tr>
<th>Gesture</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Select</td>
</tr>
<tr>
<td>1</td>
<td>Paste</td>
</tr>
<tr>
<td>2</td>
<td>Begin selection</td>
</tr>
<tr>
<td>3</td>
<td>End selection</td>
</tr>
<tr>
<td>4</td>
<td>Transpose</td>
</tr>
<tr>
<td>5</td>
<td>Split line</td>
</tr>
</tbody>
</table>

Input Representation and Preprocessing
We use a temporal representation of gestures, i.e. a sequence of
coordinates tracking the stylus as it moves over the tablet's sur­
f ace, as opposed to a static bitmapped representation of the
shape of the gesture. This dynamic representation was moti­
vated by its successful use in handwritten character recognition
[1]. Results of experiments described in that work suggest that
the time-sequential signal contains more information relevant
to classification than the static image, leading to better per­
formance.

In our current implementation, the stream of data from the dig­
itizing tablet goes through a preprocessing phase patterned
after the one described in [1], consisting of normalizing and
resampling the coordinates to eliminate differences in size and
drawing speed, and extracting local geometric information
such as the direction of pen movement and the curvature of the
trajectory. These features are believed to hold discriminatory
information that could help in the recognition process.

Gesture Classification Using Neural Networks
We use a Time Delay Neural Network (TDNN) (see Figure 1)
to classify each preprocessed time-sequential signal as a ges­
ture among the predefined set of 8 gestures. Each gesture in the
set is represented by an output neuron. Details on the workings
of the TDNN can be found in [3]. The network is trained on a
set of manually-classified gestures using a modified backpropagation algorithm [3]. The output neuron with the highest activation level determines the recognized gesture.

Our gesture recognizer achieves 98.9% recognition rate on the training data set (640 samples) and 98.8% on an independent test set (160 samples).

**SPEECH PROCESSOR**
The speech processing subsystem of our multi-modal text editor consists of a word spotter and a semantic-fragment parser.

**Word Spotter**
This initial version of the text editor requires only a small vocabulary, hence a word spotter was deemed more appropriate than a full speech recognition system. Instead of trying to recognize all parts of an input utterance, the word spotter only signals occurrences of predefined keywords within the utterance. The word spotter used in our system was developed at Carnegie Mellon by Zeppenfeld, based on the Multi-State Time Delay Neural Network (MS-TDNN), an extension of the standard TDNN architecture. More details on architecture, implementation, and performance evaluation of the word spotter can be found in [5].

For our editing task, the word spotter was trained on a single-speaker speech database that includes about 45 instances of each of 11 keywords: delete, move, transpose, paste, split, character, word, line, sentence, paragraph, and selection. The word spotter achieves a recognition performance of 95.9% on the training data set.

**Semantic-Fragment Parser**
The output of the word spotter is a text string consisting of keywords occurring in the input utterance. This can be regarded as a machine-transcribed version of the input in which only essential words are retained. For instance, "Please delete this word for me" produces "delete word". This simplified version is then parsed using a semantic-fragment grammar. The parser, developed by Ward [4], matches fragments of the input text against predefined templates to find semantically useful parts of the text. It then creates a frame consisting of slots representing various components of a plausible semantic interpretation, and fills in any slot it can using semantic fragments found in the input sentence.

In the case of our text editor, the grammar defines two slots: action and scope. For the above example, the sentence "delete word" will cause the action slot to be filled with delete, and the scope slot to be filled with word.

**JOINT INTERPRETATION OF GESTURE AND SPEECH**
Figure 2 shows a block diagram of the interpreter.

We based the interpretation of multi-modal inputs on frames. As explained above, a frame consists of slots representing parts of an interpretation. In our case, there are three slots named action, source-scope, and destination-scope (the destination is used only for the move command). Within each scope slot are subslots named type and unit. The possible scope types are: point (specified by coordinates), box (specified by coordinates of opposite corners), and selection (i.e. currently highlighted text). The unit subslot specifies the unit of text to be operated on, e.g. character or word.

Consider an example in which a user draws a circle and says "Please delete this word". The gesture-processing subsystem recognizes the circle and fills in the coordinates of the box scope specified by the circle in the gesture frame. The word spotter produces "delete word", which causes the parser to fill the action slot with delete and the unit subslot of source-scope with word. The frame merger then produces a unified frame in which action=delete, source-scope has unit=word and type=box with coordinates as specified by the drawn circle. From this the command interpreter constructs an editing command to delete the word circled by the user.

One important advantage of this frame-based approach is its flexibility, which will facilitate the integration of more than two modalities. All we have to do is define a general frame for interpretation and specify the ways in which slots can be filled in by each input modality. In a general implementation, it is possible that the slots may be filled in different ways, and performing a search to find the best merging would be superior.

**REFERENCES**


