Edge-splitting in a Cumulative Multimodal System, for a No-Wait Temporal Threshold on Information Fusion, Combined with an Under-Specified Display

Edward C. Kaiser, Paulo Barthelmess

Natural Interaction Systems, LLC.
10260 SW Greenburg Road Suite 400
Portland , OR 97223
{Ed.Kaiser,Paulo.Barthelmess}@naturalinteraction.com

Abstract

Predicting the end of user input turns in a multimodal system can be complex. User interactions vary across a spectrum from single, unimodal inputs to multimodal combinations delivered either simultaneously or sequentially. Early multimodal systems used a fixed duration temporal threshold to determine how long to wait for the next input before processing and integration. Several recent studies have proposed using dynamic or adaptive temporal thresholds to predict turn segmentation and thus achieve faster system response times. We introduce an approach that requires no temporal threshold. First we contrast current *multimodal command interfaces* to a new class of *cumulative-observant multimodal systems* that we introduce. Within that new system class we show how our technique of *edge-splitting* combined with our strategy for *under-specified*, no-wait, visual feedback resolves parsing problems that underlie turn segmentation errors. Test results show a 46.2% significant reduction in multimodal recognition errors, compared to not using these techniques.

Index Terms: multimodal, parsing, speech, handwriting, turn segmentation, under specified display.

1. Introduction

It is difficult for a multimodal system to know how to group and segment user input turns. For example, Fig. 1 shows a user performing a multimodal, "move that table … over here," command in virtual reality [1]. This particular command required (1) a combination of speech and gesture to designate the object to be moved, followed by (2) a second speech and gesture combination to designate a destination location. Suppose the command preceding this one was a *color change* command, which required pointing at an object and saying, "make that red." If unsuccessful, then representations of both the point gesture and the spoken utterance would be leftovers. With such leftovers present, the first point gesture of the current *move* command could erroneously combine with the leftover speech, and the table, which the user intended to move, would not move but instead turn red (Example 1). Even with no interference from leftover inputs, the user might pause so long before designating the destination that the system would decide that turn-input had finished, and combine a lower scoring speech alternative, (e.g. "make that gray" – rather than the correct, "move that table") with the initial deictic point gesture, so that instead of the table being moved, the floor beneath it would unexpectedly turn gray (Example 2).

Such mis-combinations exemplify the difficulty any multimodal system has in segmenting and grouping individual modal inputs into either integrative multimodal units or separate unimodal commands. To address this difficulty we will first examine the issue of turn segmentation in *command interface* systems (Sec. 2). Then we will introduce an alternative multimodal *cumulative-observant system* (Sec. 3), which by leveraging accumulated structure, and employing our technique for *edge-splitting* (Sec. 4) with an *under-specified* display (Sec. 5) offers an alternative approach to turn segmentation and grouping problems.

2. Turn Segmentation

The current approach to the multimodal turn segmentation problem, as discussed in the literature, is to wait for some fixed threshold of time before assuming the end of user turn-input [2]. Recent research has sought to reduce this fixed wait time by the use of corpus-based, probabilistic [3] or user-adaptive models [4] of input styles. The motivation for modeling this temporal threshold is to avoid on the one hand what Gupta *et al* term *under-collection* errors (in which some user turn-inputs arrive after the start of processing – Table 1: #1, #2), and on the other hand *over-collection* errors (in which users re-enter inputs due to a perception of system unresponsiveness – Table 1: #3, #4) [3]. Our Introductory Example 2 is an *under-collection* error. As

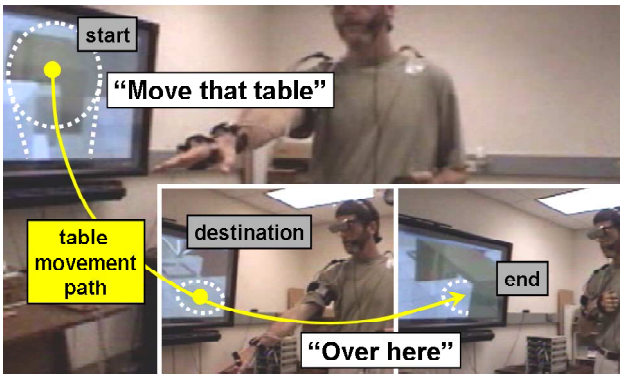

Figure 1: A 4-part multimodal command (speech/point + speech/point) to move a virtual table. The plasma screen shows what the user sees in his HMD. The white-dotted areas (added here) show pointing targets. **Turn segmentation** is critical to correct multimodal recognition.

Collection	Unimodal errors		Multimodal errors	
Under	***#1***	Unimodal part of multimodal input	***#2***	Partial combination of longer multimodal input
Over	***#3***	Re-input	***#4***	No feedback, re-input.
Over-under	***#5***	Left-overs combine with under-collection.	***#6***	Left-overs combine with under-collection.

Table 1: Categorization of multimodal temporal mis-combination errors by collection type and modality level.

Johnston *et al* [5] describe, avoiding these types of errors is important when mistakes have disruptive or confusing side-effects. Table 1 categorizes the types of temporal mis-combination errors that can occur in a multimodal system. Aside from under and over collection errors, we add a third category, not described by Gupta *et al* that we term *over-under collections*. These occur when left-over inputs from previous commands remain available and combine with subsequent under-collections (Introductory Example 1).

Both Gupta *et al's* [3] and Huang *et al's* [4] recent studies assume the use of *multimodal command interfaces*, which alternate between accepting turns of user input and displaying the interpreted output. They both focus on minimizing under/over collection errors by better predicting how long to wait for the end of user turn input, using Bayesian modeling techniques. However, focusing solely on turn segmentation prediction does not adequately consider the underlying parsing mechanisms at work in a multimodal system. When these are addressed, other classes of multimodal systems become possible with different strategies for turn segmentation.

3. A Cumulative-Observant System

We introduce a new class of multimodal system, a *cumulative-observant multimodal interface*. Instead of supporting a sequence of command/display turns, a cumulative interface gathers continuous input across a structured multiparty interaction, like the construction of a Gantt schedule chart [6], All sensors in the system (e.g. ink-capture whiteboard, stereo-vision-based 3D-gesture-tracker, close-talking microphones), collect information second-hand by observing what is naturally occurring between the people involved. Thus the interface is one of human-human, computer mediated interaction. In some regards, a *cumulative-observant multimodal interface* is similar to emerging dual-purpose interfaces, where inputs are dually directed to both a human and a machine, like a PDA assistant that can schedule meetings based on overhearing hallway conversations [7]. However, with a *cumulative-observant* interface there is no explicit interaction with the system.

The *cumulative-observant* system output is a record of the interaction, ranging from a searchable/browsable playback to a direct semantic interpretation, like an MS Project object representing a whiteboard planning session [6]. The record includes dynamically learned acronym and abbreviation meanings, which are built-up in the background by piggy-backing on the rich natural communication between the participants — like the redundant speech and handwriting [8] that frequently occurs in human-human communication [9]. A cumulative interface has available the evolving context of the structured interaction itself (e.g. Gantt Chart creation). This added context provides both strong spatial and temporal constraints, which implicitly segment turn-inputs.

4. Parsing and Edge-Splitting

Johnston [5] outlines the basic chart parsing algorithm as the following, where * is an operator that combines two constituents according to the rules of the grammar, constituents are designated as terminal sequences from vertex to vertex, and both the vertices and constituents are linearly ordered..

$$Chart(i,j) = \bigcup_{i<k<j} chart(i,k) * chart(k,j) \qquad (1)$$

As Johnston points out, in a multimodal context linearity is not assured, because input from different modal constituents can well be temporally overlapped. Thus he defines the basic temporal, multimodal chart parsing algorithm as:

$$multichart(X) = \bigcup multichart(Y) * multichart(Z) \qquad (2)$$
$$where\ X = Y \cup Z,\ Y \cap Z = \phi,\ Y \neq \phi,\ Z \neq \phi$$

Regardless of parser architecture, constituent edges in a multimodal parse space cannot be identified by linear spans. Instead they are identified by unique sets of identifiers (e.g. *multichart*([*s*,1,1,0],[*g*,2,2,1])), each of which specify the constituent's mode of origin, recognition sequence number, position on the list of alternate recognitions, and semantic interpretation. This identification axiom maintains the critical constraint enforced by linearity that a given piece of input can only be used once in a single parse. Commands with intersecting IDs are different interpretations of the same input, and are thus ruled out by the non-intersection constraint in equation (2) above. This means that there can only be one correct interpretation acted upon for each set of inputs. Therefore once that best scoring command is chosen and executed all constituent edges from that command are removed from the chart.

4.1. Edge-Splitting

This removal policy means that all constituent edges, which participate in an otherwise correct interpretation of partial, under-collected input, are then no longer available to participate in a subsequent interpretation of fully collected turn-inputs, because their IDs would intersect. This is the underlying issue in multimodal parsing that makes under-collection a general problem.

multichart([**id**,2,1,0]) ➔ *multichart*([**mmid**,2,1,0]) (3)

Our solution is to (1) filter all messages of an appropriate type (e.g. all ink-gestural edges), (2) clone them – changing only the input mode symbol identifier (**id➔mmid**, Eq. 3), and (3) put the cloned edges back on the chart. We then enforce the constraint that edges with the new input mode symbol identifier (e.g. **mmid**) only participate in subsequent multimodal interpretations. They can no longer be interpreted unimodally. These split-edge clones are periodically removed from the chart just as other edges are removed, based on an edge-defined time-out period. To allow for long distance associations across modes, edge time-outs are ignored until at least the next edge of the same type arrives on the chart. Thus edge-splitting, in conjunction with an under-specified display (*cf* Section 5), solves the underlying problem of under-collected, unimodal interpretations starving subsequent multimodal interpretations by removing the edges needed for multimodal integration.

Presently, our playback [10] processing time is on the order of 5-10 times real time. Using edge-splitting slows multimodal processing by about 11% in our tests. Chart parsing in general has a worst-case cubic complexity, but in practice when the rule set is regular, as in our system, then actual parsing complexity is linear. Our edge-splitting technique, in the worst case, may double the number of edges; however, the rule set remains regular so the linear order of complexity remains unchanged.

4.2. Testing Edge-Splitting in Charter

Figure 2 depicts the the use of *edge-splitting* in our Charter Suite prototype application for cumulative-observant multimodal recognition of a multiparty scheduling meeting. The scenario was planning office space and computer equipment for three new hires during a series of five three-person meetings (averaging 6 minutes each). Ink was collected from a pen-sensitive whiteboard and speech from close-talking microphones. Meeting processing was accomplished by hi-fidelity playback of recorded sensor information [10]. Five related Gantt Chart schedule diagrams were produced, one for each meeting. The upper half of the diagram in Fig. 2 shows an example task-line, labeled *office*, and diamond-shaped milestones marking the temporal availability of office space (abbreviated as *Avail*). The bottom half of the diagram shows an example task-line, labeled *buy computer*.

Without edge-splitting, ink-gestures that temporally preceded the spoken utterances with which they were associated fired unimodally producing incorrect interpretations: (middle column of Fig. 2), *trail* (for *avail*), *lay computer* (for *buy computer*). These were under-collection errors. Their respective edges were removed from the chart disabling subsequent multimodal recognition. Also, for abbreviation interpretations based solely on under-collected ink input (middle column of Fig. 2: *trail*, *Avail*), there were no semantic glosses (e.g. Fig. 2, upper right, gray text boxes containing "AVAILABLE"). These were produced only by integration with speech, via our Speech and Handwriting Recognizer (SHACER) [8]. Without edge-splitting under-collection errors disabled the multimodal integration necessary for semantic glosses.

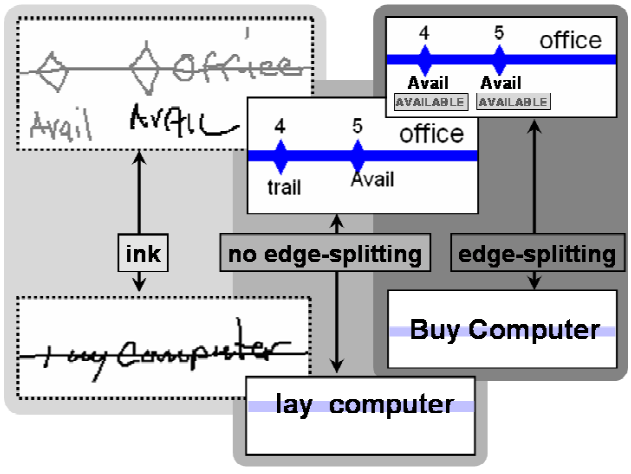

Figure 2: **Left-column**: Gantt chart ink. **Middle column**: Edge-splitting disabled, so interpretations based on Type 1 under-collection errors (*cf.* Table 1). **Right column**: Edge-splitting enabled, so errors have been corrected by subsequent multimodal interpretations. Lighter colored ink (upper left) illustrates no-wait, under-specified recognition display.

Multimodal, Redundant Labels	51	Errors	RER
Labels found: Edge-Splitting	44	7	46.2%
Labels found: No Edge-Splitting	38	13	

Table 2: Test Results for multimodal constituent labeling, with Relative Error-rate Reduction (RER) for Edge-Splitting.

With edge-splitting enabled the incorrectly interpreted unimodal ink-gesture edges were split, and their multimodal clones put back on the chart. These split edges then combined with their respective spoken utterances producing correct multimodal interpretations (shown in the right column of Fig. 2), which replaced the incorrect unimodal interpretations.

Table 2 shows the test results for using edge-splitting. In the five meeting test series there were 51 constituent labels in the finished Gantt Charts. All 51 were presented multimodally, i.e. handwritten and spoken redundantly. Without edge-splitting there were 13 multimodal label recognition errors. With edge-splitting there were only 7. Counts were determined by visual inspection of system output (e.g. Fig. 3, middle and right cols.). Therefore edge-splitting yielded a relative error rate reduction (RER) of 46.2% — significant by a McNemar test ($p = 0.03$).

We note that in two of the six error instances corrected by our edge-splitting technique speech was associated with a preceding ink-gesture that was between 3-22 unrelated utterances and 45-250 seconds earlier. Thus, edge-splitting mitigated under-collection errors for both temporally distinguished turns (Fig. 2, *buy computer*), and for input groupings that were structurally distinguished based on their redundancy relations despite long temporal/turn distances (Fig. 2, *Avail*). Turn segmentation based solely on temporal thresholds (e.g. adaptive temporal threshold prediction [3, 4]) could not address such integration errors across long-distance, structurally-distinguished groupings.

In our test set of five meetings there were 500 utterances and 183 gestural/ink inputs. Most chart constituents could only be recognized unimodally (e.g. axes, tickmarks, tasklines, milestones), while others (e.g. handwritten labels) could be interpreted either unimodally or multimodally. All gestural inputs were initially interpreted unimodally. In edge-splitting mode roughly 24% (44/183) of gestural inputs were also interpreted multimodally (Table 2), whereas in non-edge-splitting mode 21% (38/183) were interpreted multimodally. In non-edge-splitting mode the unimodal recognition rate was roughly 87%, while in edge-splitting mode it was 89%. Thus, in our test series, with competing unimodal/multimodal recognitions, edge-splitting never confounded or degraded recognition, it only significantly improved recognition.

5. Interface Under-Specification

For our Charter system to avoid disrupting user work in the presence of inevitable misinterpretations we employ *under-specification* as a strategy for interface presentation, based on the principle that the interface should at all times make as few commitments to specific interpretations as necessary. For example, in the *color-change* versus *move* scenarios from our Introduction, a delayed commitment to the initial, mistaken color change interpretation could take the form of highlighting the object without actually changing its color. As additional

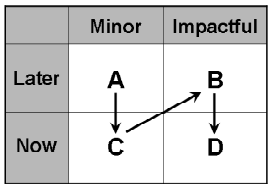

Figure 3: Design space: ***moment in time*** x ***potential impact*** of presentation. The arrow shows a path of growing potential user disruption.

evidence is collected, the system could determine whether to apply the coloring or render the *move* interpretation. In this way users would be shielded from the full impact of an imperfect interpretation. Being able to postpone specific presentation commitments allows a system to benefit from techniques like edge-splitting, which correct earlier partial or mistaken interpretations by subsequent fully collected interpretations.

Two main factors influence the user interface effects of recognition-based systems: a) the moment in time a system is required to commit to an interpretation, and b) the form in which the current state of interpretation is manifested. Figure 3 displays the interplay between these two factors, which defines a design space. Systems will range from those that require immediate responses (such as many command oriented systems), to those that produce results only after an interaction has been concluded. Cumulative-observant systems fall typically in this latter category. In between these two extremes, other systems, such as digital assistants, may choose to interact only when the system has high enough confidence on the state of the interpretation, as suggested in Neem [11].

The form in which a system manifests an interpretation – the second aspect depicted in Figure 3 – determines the impact of misinterpretations. Minor impact may result from a display that still allows users to recognize the correct interpretation, e.g. a slightly misspelled word. Highly impactful manifestations, like an incorrect navigation command (e.g. interpreting a *pan* command as a *zoom*), can leave users disconcerted and confused.

6. Discussion and Conclusion

A key insight and contribution of this paper is the notion that multimodal systems capable of revised interpretations can be designed to consider the tradeoffs between interpretation confidence, moment and impact of presentation. Under-specification takes advantage of this insight by proposing that less impactful renditions be presented earlier on, being potentially replaced by more concrete (and more impactful) renditions as the system has a chance to examine additional evidence and thus produce a more robust interpretation.

For example, consider our Distributed Charter system [12], which is a collaborative front-end for MS-Project. It allows co-located and distributed users to jointly create a Gant Chart via sketching and speaking. Earlier versions of the system presented users with a beautified version of their sketches (e.g. Fig. 2, middle column). Being able to see the system's interpretation in real-time resulted on many occasions in users interrupting project-related discussion, and shifting their efforts to correct the system's mis-interpretations. This in turn introduced a level of noise that confused the system, harming the re-evaluation mechanisms that allowed the system to recover from early errors by subsequent revisions as more evidence was accumulated. To counter this adverse effect, we re-engineered the interface to reduce the impact of the rendition of interpretations. The current interface is based on the original sketched elements rather than on their beautified counterparts. Recognition is indicated by changes in color of the sketched elements (e.g. Fig. 2, upper left ink-example). Users therefore still have general access to the state of recognition, but can (and in practice do) ignore it and continue interacting. The beautified version can be displayed on demand, mitigating any user sense of unresponsiveness that could lead to over-collection errors. The system now delays its commitment to interpretations by shifting the impact of presentation. In terms of Figure 3, Charter's interface was moved from quadrant D to A. The now minor impact represented by color changes of the original ink allows the system to safely delay final decisions with respect to an interpretation.

We have shown that our *edge-splitting* technique results in a significant 46.2% relative reduction in multimodal recognition error rate by avoiding the underlying parsing problems related to incorrect turn segmentation and supporting long-distance groupings beyond the scope of temporal-based turn segmentation predictors. Combined with our strategy for *under-specified* display, this approach provided better user satisfaction with enhanced final recognition.

7. Acknowledgements

This work was supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. NBCHD030010. Opinions, findings, conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect the views of the DARPA or the Department of Interior-National Business Center (DOINBC).

8. References

[1] Kaiser, E., A. Olwal, D. McGee, H. Benko, A. Corradini, X. Li, P. Cohen, and S. Feiner. *Mutual Disambiguation of 3D Multimodal Interaction in Augmented and Virtual Reality*. *ICMI,* 2003.

[2] Oviatt, S.L., A. DeAngeli, and K. Kuhn. *Integration and synchronization of input modes during multimodal human-computer interaction*. *CHI '97*. New York: ACM Press.

[3] Gupta, A.K. and T. Anastasakos. *Dynamic time windows for multimodal input fusion*. *INTERSPEECH-2004*. Jeju Is., Korea.

[4] Huang, X., S. Oviatt, and R. Lunsford. *Combining User Modeling and Machine Learning to Predict Users' Multimodal Integration Patterns*. MLMI, 2006. Washington, DC, USA.

[5] Johnston, M., P.R. Cohen, D. McGee, S.L. Oviatt, J.A. Pittman, and I. Smith. *Unification-based Multimodal Integration*. *ACL '97*.

[6] Kaiser, E., D. Demirdjian, A. Gruenstein, X. Li, J. Niekrasz, M. Wesson, and S. Kumar. *Demo: A Multimodal Learning Interface for Sketch, Speak and Point Creation of a Schedule Chart*. *ICMI,* 2004. State College, PA.

[7] Lyons, K., C. Skeels, and T. Starner. *Providing Support for Mobile Calendaring Conversations: A Wizard of Oz Evaluation of Dual–Purpose Speech*. in *Mobile HCI*. 2005.

[8] Kaiser, E.C. *SHACER: a Speech and Handwriting Recognizer*. *ICMI, MM Meeting Processing Workshop,* 2005. Trento, Italy.

[9] Anderson, R., C. Hoyer, C. Prince, J. Su, F. Videon, and S. Wolfman. *Speech, Ink and Slides: The Interaction of Content Channels*. in *ACM Multimedia*. 2004.

[10] Kaiser, E.C., P. Barthelmess, and A. Arthur. *Multimodal Play Back of Collaborative Multiparty Corpora*. *ICMI, Workshop on Multimodal, Multiparty Meeting Processing*. 2005. Trento, Italy.

[11] Barthelmess, P. and C.A. Ellis, *The Neem Platform: an Evolvable Framework for Perceptual Collaborative Applications.* Journal of Intelligent Information Systems, 2005. **25**(2207-240).

[12] Barthelmess, P., E.C. Kaiser, X. Huang, and D. Demirdjian. *Distributed Pointing for Multimodal Collaboration Over Sketched Diagrams*. *ICMI*. 2005. Trento, Italy