Implementation and Evaluation of a Constraint-Based Multimodal Fusion System for Speech and 3D Pointing Gestures

Hartwig Holzapfel
hartwig@ira.uka.de

Kai Nickel
nickel@ira.uka.de

Rainer Stiefelhagen
stiefel@ira.uka.de

Interactive Systems Laboratories
Universität Karlsruhe (TH)
Germany

ABSTRACT

This paper presents an architecture for fusion of multimodal input streams for natural interaction with a humanoid robot as well as results from a user study with our system. The presented fusion architecture consists of an application independent parser of input events, and application specific rules. In the presented user study, people could interact with a robot in a kitchen scenario, using speech and gesture input. In the study, we could observe that our fusion approach is very tolerant against falsely detected pointing gestures. This is because we use speech as the main modality and pointing gestures mainly for disambiguation of objects. In the paper we also report about the temporal correlation of speech and gesture events as observed in the user study.

Categories and Subject Descriptors

H.5.2 [**INFORMATION INTERFACES AND PRESENTATION**]: User Interfaces

General Terms

Human Factors, Experimentation, Languages

Keywords

multimodal fusion and multisensory integration; speech, vision, natural language, gesture; multimodal architectures

1. INTRODUCTION

Multimodal interfaces [23] have been in the focus of research since Bolt's [2] seminal work. Recently, multimodal interfaces have been applied to facilitate natural interaction with human-friendly robots [13], [12]. A major challenge is to understand different input modalities in changing environments with background noise and changing light condition, which complicates speech recognition and visual tracking. The work presented in this paper is part of our efforts on building technologies for multimodal human-robot interaction. An overview of the complete system, including details about the speech recognizer, person tracking and gesture recognition, can for example be found in [25]. In this work, we focus on the integration of different input modalities. We present our architecture for multimodal fusion that tries to cope with the above problems, together with an evaluation of the whole system. The presented system is tolerant against falsely detected gestures, which is important as our experiments have shown that under the given conditions, we could achieve a gesture detection rate of 87% (recall), but achieving only 47% precision (53% falsely detected gestures).

1.1 Related Work

So far, different systems for multimodal fusion and multimodal dialogue systems exist e.g. [15], [22]. In our experiments we have found out that the system has to cope with many false detections, which is different from gesture input on 2D surfaces e.g. to pen input [21], which we will address later in this paper. Different formalisms and approaches have been taken for multimodal fusion, like statistical approaches [27], salience-based [8], or rule-based [14], and biologically motivated approaches [26].

Existing work with pen input systems [21], and more recently 3D gestures [5], has shown that speech and manual gestures are correlated in time. Time correlation has also been evaluated for speech and gestures in multi party dialog [4], for gaze and speech [17], and based on prosody [18]. In this paper we will also present an indication for very close correlation in time between gesture and referring words in speech.

1.2 Overview

For multimodal fusion we have chosen a rule-based approach on the semantic level, with a separation of the system into an application independent parser and application specific fusion rules. For representation of semantics we use typed feature structures (TFS) [3]. The parser uses cons-

ICMI'04, October 13–15, 2004, State College, Pennsylvania, USA.
Copyright 2004 ACM 1-58113-954-3/04/0010 ...$5.00.

traints to determine which elements can be merged, afterward construction rules are applied to create the output. We do not only rely on direct deictic references like ’this’, since words like ’the’, ’this’, and ’that’ can easily be mixed up by the speech recognizer. Merging is rather done on an information-based approach by comparing object types that are defined in an ontology. We extend previous work on multimodal parsing [14] [16] to deal with falsely detected input, integrated processing of n-best lists and a clean separation of constraints and merging rules. Different to pen input, it is much harder to detect when a gesture has taken place for 3D pointing gestures. We thus cannot rely on this information. To obtain robustness, different form other work [14], [16], we use speech as the main modality, and gesture to resolve ambiguities, which is better suited to deal with falsely detected gestures. For redundant multimodal input, multimodal fusion improves the robustness of the system e.g. when speech recognition errors occur.

1.3 Application Scenario

The system is designed to run on a humanoid robot in a kitchen scenario. The robot’s task is to help people in the kitchen, to bring dishes, switch on lights, put objects into the dishwasher [11]. We are currently porting the system to the humanoid platform ARMAR [1], which is shown in Figure 2. In the experiment the user could instruct the robot to get or bring objects and switch on or off lamps and the air conditioner. The system disambiguates object descriptions either by exact description from speech input or by disambiguation through the interpretation of pointing gestures. The experiment was performed with a stereo camera for person tracking and a head set microphone with automatic segmentation for speech recognition.

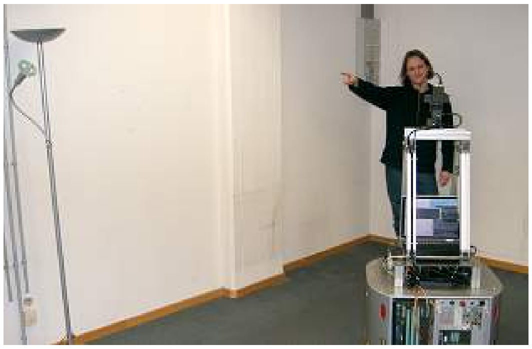

Figure 1: Interaction with our development system. Software components include: speech recognition, speech synthesis, person and gesture tracking, dialogue management and multimodal fusion of speech and gestures.

The following example dialogues show possible interaction scenarios with speech only or with speech and gestures.
example dialog 1: (speech only)
User: ”please switch on the light”
System: ”which light?”
User: ”the big lamp”
System: ”switching on the big lamp”

example dialog 2: (speech + gest)
User: ”please switch on the light”

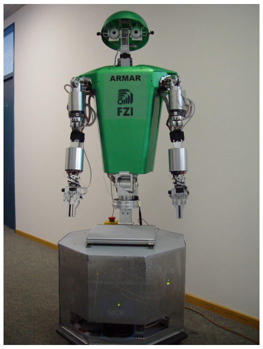

Figure 2: Some components have already been integrated in a humanoid robot with two arms.

System: ”which light?”
User: ”this lamp” (+ pointing gesture)
System: ”switching on the big lamp”

The remainder of the paper is organized as follows: Section 2 describes (pre-)processing of the input. Section 3 describes the parsing and fusion algorithms, for which application specific rules are created in section 4. Section 5 finally presents an evaluation of the described system and a small user study that has been performed to create fusion rules.

2. INPUT PROCESSING AND INFORMATION REPRESENTATION

An input event (in its semantic representation) is referred to as an input token. Fusion is done by applying a single, best fitting rule to a set of input tokens. Each rule has a left hand side that defines how to combine multiple tokens; and a right hand side that defines preconditions and constraints for the input tokens, to test if the rule can be applied. The right hand side defines constraints for the number, modality, time properties and the semantic content of the input tokens.

The gesture input to the system is provided by a gesture recognizer working with stereo cameras [20]. By using stereo cameras, 3D depth information is available that allows to detect pointing directions in 3D space. A gesture event contains two 3-dimensional vectors. One vector contains the position of the user’s hand. The other vector contains the pointing direction. The 2 vectors are parsed into a feature structure of the following type:

$$\begin{bmatrix} \textit{gst\_pointing\_3d} \\ HX \begin{bmatrix} 0.1 \end{bmatrix} \\ HY \begin{bmatrix} 0.2 \end{bmatrix} \\ HZ \begin{bmatrix} 0.1 \end{bmatrix} \\ PX \begin{bmatrix} 0.1 \end{bmatrix} \\ PY \begin{bmatrix} 0.2 \end{bmatrix} \\ PZ \begin{bmatrix} 0.1 \end{bmatrix} \end{bmatrix}$$

This information is used to retrieve object information from the environment model database. The result is a list of objects that are close to the pointing direction within an error cone. From the resolved database request a feature structure of the following type is created:

$$\left[\begin{array}{l} \mathit{gst\_pointing\_3d\_resolved} \\ \mathit{REF}\left[\, < \mathit{object-type} > \,\right] \end{array}\right]$$

For example:

$$\left[\begin{array}{ll} \multicolumn{2}{l}{\mathit{gst\_pointing\_3d\_resolved}} \\ \mathit{REF} & \left[\begin{array}{l} \mathit{obj\_lamp} \\ \mathit{NAME}[\,''\mathit{littlelamp}''\,] \\ \mathit{SCORE}[\,''168.16''\,] \end{array}\right] \end{array}\right]$$

The feature "SCORE" contains a score computed by the object resolving algorithm. It defines, how well the object fits to the pointing gesture. The maximum (best) score is 180. The resolving algorithm computes a score for each object in the environment model. It returns a (sorted) list of objects that is limited by a predefined threshold that must be exceeded by the score. We will refer to this list as the n-best list. Note that with this definition n is not fixed and changes with each result.

The system creates one feature structure for each object in the n-best list. The n-best list is wrapped in an object structure that allows to access only the best element or to access the whole set to handle each feature structure in the list.

For speech recognition, we are using the Janus Recognition Toolkit (JRTk) [9] with the Ibis single pass-decoder [24]. We use the option of Ibis to decode with context free grammars (CFG) instead of statistical n-gram language models (LM). The dialogue manager uses the same grammars as the speech recognizer to convert the resulting parse tree into typed feature structures. This is performed by conversion rules that are defined in the grammar as semantic tags of the grammar nodes. The spoken input "please switch on the lamp" is transformed to the TFS

$$\left[\begin{array}{l} \mathit{act\_switch} \\ \mathit{ONOFF}\left[\begin{array}{l} \mathit{prp\_onoff} \\ \mathit{BOOL}[\,\mathit{true}\,] \end{array}\right] \\ \mathit{OBJ}[\,\mathit{obj\_lamp}\,] \end{array}\right]$$

The grammar for speech recognition and language understanding contains 164 non-terminal grammar nodes and $\approx$ 1000 terminals. A complete roll-out of the grammar generates roughly 232 million input sentences.

The multimodal fusion component synchronizes and combines the output of the recognition components and sends the result to the dialogue manager. Using a rule-based system fits into the approach of the dialogue manager to be well suited for rapid prototyping [7]. It doesn't require a preexisting corpus for training the system.

3. MULTIMODAL FUSION

3.1 Architecture

The fusion component runs in a multi-threaded architecture. On the input side tokens are added asynchroneously to the input set. During fusion, the tokens are read from this set by the parser, and combined into one output stream representing the fusion result.

Each modality has its own channel for receiving modality specific events that are then transformed into semantic structures, according to the previous section. After transformation, the tokens are added to the input set. The input set is used to synchronize the different threads that are adding and reading from the set. During one parser run, new inputs to the set are delayed until the parser run is finished. The input set thus remains constant for the parsing algorithm.

To merge the input streams, constraint-based parsing is applied to the input tokens in the set. Parsing rules include constraints to determine whether a subset of tokens can be merged, and instructions (creation rules) to construct the merge result.

3.2 Parsing

Different from parsing algorithms normally used in natural language processing, a linear order of the elements is not appropriate for multimodal integration. Parsing is rather performed on a pool of elements, where new elements can be added and elements can be removed. This mode of operation is similar to the multichart parser [16]. Different from the multichart approach, this parser can skip tokens that remain in the pool for different iterations until a fusion rule can be applied, or a time expiration rule can be applied. This is important to deal with different arrival times of the various modality specific recognizers, see also section 3.3.

The parsing algorithm observes each possible subset of the input set. One parser run first observes all subsets with the greatest size possible, and then step by step all smaller subsets. It is thus able to parse longer matches first.

Our system needs to be robust against falsely detected gestures. They lead to tokens that should be ignored by the parser. This is accomplished by removing tokens, for which no fusion rules can be applied, from the input set after a predefined amount of time. Runtime characteristics of the system are described in section 5.5.

3.3 Online Characteristics

Parsing of the input set must obey some characteristics of online algorithms. For example, the arrival time of the input events can change for different delays in the speech and gesture recognizers. It is possible that an event B arrives before event A, though A occurred before B. In this case it is possible that - at the time of parsing - the input set contains only B, because A has not arrived yet. This has to be considered when creating parsing rules for a system.

We have observed different time delays in various setups, i.e. different hardware and CPU resources available for speech recognizer and gesture recognizer, in which the arrival times of the gesture event ranged between 400 ms before the speech event and 200 ms after the speech event. To cope with this, we extended the constraints of the fusion rules. The decision to implement this by fusion rules is appropriate, since they are designed to take care of application specific peculiarities. The interpretation of this behavior would be to wait for a certain amount of time for a gesture event, if the speech content suggests that there might be information to disambiguate.

3.4 Multimodal Fusion Rules

To allow the parsing algorithm to be independent of the rule definition, a multimodal fusion rule consists of constraints and result construction rules. The parsing algorithm uses the constraint definitions to determine which rules can be merged. After that, the construction rules are applied to create the fusion results. For the same purpose, it is not predefined, when constraints are checked, so they may not change or modify the input structures. The execution of construction rules may not influence the decision whether input tokens can be merged or not.

The system knows some predefined basic constraint ty-

$$\text{CONTENT} \quad \begin{bmatrix} \$1 = act\_switch \\ ONOFF \begin{bmatrix} prp\_onoff \\ BOOL \begin{bmatrix} \$2 = bool \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Figure 3: CONTENT constraint

pes, including content, time and modality constraints. In addition, script constraints are allowed to offer more flexibility. Like predefined constraints, script constraints return a boolean value. Script constraints can be defined "in-place" which means directly in the code, or they can be extracted and used in the same way as predefined constraints.

The *CONTENT* constraint (e.g. Figure 3) defines the minimum of information that needs to be represented by the matching token. The constraint is evaluated in the same manner as underspecified feature structures, where the typed feature structure nodes are elements in an ontology, e.g. [6].

The *MODALITY* constraint defines the input channel. Possible values are for example "speech" or "gesture".

Additionally, an input token has properties like timestamps that are not represented in its typed feature structure. They can be checked by script constraints. Script constraints can be used to test if events overlap in time or follow each other within a maximum amount of time. Script constraints are defined in Python and have the same power as a full programming language. This is shown in SCRIPT1, see Figure 4. SCRIPT1 first checks by type inheritance if the found object is of type lamp and then sets a variable that will later be used for constructing the result TFS.

Constraints allow the definition of variables. The definition of a variable can take place in *CONTENT* constraints or in script constraints. The variables are then used to during result construction.

The left hand side of a fusion rule describes how to construct results and is represented in the form of a typed feature structure.

4. EXEMPLARY RULE DEFINITION

This section describes some rules from the system, that are used to resolve deixis in general [19], and n-best list processing of recognizer output.

4.1 Rules for Gestures and Speech

The rule in Figure 6 performs resolution of a deictic reference. The information that is required for the speech event is that the user wants to switch on or off something. The constraints for the gesture token require the referenced object to be a lamp, see SCRIPT1 in Figure 4.

4.2 Speech only

Figure 7 shows a rule to parse only a single speech token without merging it with a gesture token. As already described in section 3.2, longer rules are processed before shorter rules. Long at this refers to the number of tokens that are matched by the right hand side. Hence the rule in Figure 7 only applies to a speech token, if this token cannot be merged with other tokens.

The *CONTENT* constraint only requires the element base:bot which is the most general node in the ontology. Thus, any typed feature structure that is not undefined matches the constraint.

```
import jarray
#simple example: uses only best element
obj_lamp = tfs.getType( "generic:REF" )
if onto.isMoreGeneral("obj_lamp", obj_lamp.typeName):
    # set var $lamp
    vars.addVarType("$3", obj_lamp )
    constraint = 1
else:
    constraint = 0
```

Figure 4: SCRIPT1 (Python) - if the referenced object has the correct type, set variable $3 and return true

In the given example the variable $1 transfers the top level node from the speech token to the top level node of the result TFS. The content of the input typed feature structure remains unchanged.

The time constraint TIME causes the system to wait 200 milliseconds for incoming gesture events that can be merged with the speech token. The constraint is evaluated as a python expression. 'tfs' is a predefined variable in the script environment that points to the input token object which is matched by this right hand side definition. The property 'tfs.time' contains the time stamp at which the token has been added to the input set.

4.3 N-Best Lists

The resolution of pointing gestures in the environment model results in an n-best list. This is first because of the impreciseness of the gesture recognizer, that cannot resolve with an accuracy below a few centimeters, second because recognition errors are possible. Third, gestures can be ambiguous, even with perfect recognition. Consider for example a window and a lamp that stands in front of this window. The pointing gesture to the lamp could refer to the lamp or to the window. This ambiguity can only be resolved within the spoken context.

Section 2 already describes how the n-best list is created from a database request. SCRIPT2 in Figure 5 is a modification of SCRIPT1 (Figure 4). It checks for all retrieved elements from the n-best list, if they fit to the speech token.

```
for tfsi in tfs.all:
   obj_lamp = tfsi.getType( "generic:REF" )
   if onto.isMoreGeneral("obj_lamp", obj_lamp.typeName):
      # set var $lamp
      vars.addVarType("$3", obj_lamp )
      constraint = 1
      break
   else:
     constraint = 0
```

Figure 5: SCRIPT2 (Python) - iterate over nbest result of gesture resolution, if the referenced object has the correct type, set variable $3 and return true

$$\begin{bmatrix} lhs & \\ CONTENT & \begin{bmatrix} act\_switch & \\ ONOFF & \begin{bmatrix} prp\_onoff \\ BOOL[\,\$2\,] \end{bmatrix} \\ OBJ[\,\$3\,] & \end{bmatrix} \end{bmatrix}$$

$$->$$

$$\begin{bmatrix} rhs1speech & \\ CONTENT & \begin{bmatrix} \$1A = act\_switch & \\ ONOFF & \begin{bmatrix} prp\_onoff \\ BOOL[\,\$2 = bool\,] \end{bmatrix} \end{bmatrix} \\ MODALITY[\,''speech''\,] & \\ PATH[\,''generic: NAME'' == None\,] & \end{bmatrix}$$

$$\begin{bmatrix} rhs2gest & \\ CONTENT & \begin{bmatrix} gst\_pointing\_3d\_resolved \\ generic: REF[\,generic: object\,] \end{bmatrix} \\ MODALITY[\,''gesture''\,] & \\ SCRIPT\_M[\,SCRIPT1\,] & \\ TIME[\,\{tfs1.tstop <= tfs.tstart + 100\}\,] & \end{bmatrix}$$

Figure 6: Rule to merge a speech token with a gesture token to resolve a lamp object

$$\begin{bmatrix} lhs \\ CONTENT[\,\$1\,] \end{bmatrix}$$

$$->$$

$$\begin{bmatrix} rhs1 \\ CONTENT[\,\$1 = base: bot\,] \\ MODALITY[\,''speech''\,] \\ TIME[\,tfs.time + 200 <= currtime\,] \end{bmatrix}$$

Figure 7: Single speech token parse rule

For example: if the user says "please bring me this cup", and the gesture tokens contains the following elements (ordered by confidence) { lamp1, cup, window }, SCRIPT2 will pick the cup from the list.

4.4 Ellipsis in Answering

The example in Figure 6 only applies to typed feature structures where the root node is a speech act. This is intended, because the system described here is used to merge multimodal input for a dialogue system.

During conversation, users utter elliptical constructions, e.g. in answers. The utterance "this light" is used in reply to the disambiguation question of the system "which lamp do you want to switch on". The user utterance above can be used together with a pointing gesture. During input processing, the dialog manager applies a resolution algorithm to the input (see section 2). The resolution is currently quite simple. The system generates a list of transformations for each dialogue state that map object definitions to a complete representation within a speech act. The utterance "this light" with the semantic representation

$$\begin{bmatrix} obj\_lamp \end{bmatrix}$$

is converted to the complete representation

$$\begin{bmatrix} act\_switch & \\ ONOFF & \begin{bmatrix} prp\_onoff \\ BOOL[\,true\,] \end{bmatrix} \\ OBJECT[\,obj\_lamp\,] & \end{bmatrix}$$

The converted representation is then compatible to the parsing rule in Figure 6.

5. EVALUATION

The evaluation was performed with a human-robot interaction task in the kitchen scenario. The user can use multimodal commands instruct the robot with speech and gesture.

Within that task we have evaluated speech recognition and gesture recognition results, the overall success rates of the system and its runtime performance.

We first describe the data sets created for evaluation, their annotation and relationship. Based on these sets we have conducted experiments to measure the time correlation between speech signal and gestures / gesture recognition. After that we show the recognition rates and system performance on the described data sets.

5.1 Annotation Scheme & Evaluation Data

For evaluation, different data sets are created that contain annotated gestures, recognized gestures, annotated speech input and recognized speech input. Additional annotation files, here referenced as functions on the data sets, describe the correspondence of the data, including transcription (recognized to annotated data) and the combination for multimodal fusion. The first set is called G_{real} and contains the manually transcribed gestures. Set G_{rec} contains the automatically recognized gestures. Set S, with subsets, contains all speech utterances with transcriptions and timestamps.

The partial function $\varphi_G : G_{real} \rightarrow G_{rec}$ maps a manually transcribed gesture to the corresponding recognized gesture. The function is only partial since not all annotated gestures do have a recognized correspondent. All elements $g_l \in G_{rec}$ for which $\neg\exists i : \varphi_G(g_i) = g_l$ are false detections.

The partial function $\varphi_M : G_{real} \rightarrow S$ maps a manually transcribed gesture to the speech utterance to which it belongs. The set S contains all speech utterances that are relevant for multimodal fusion. The set $S_d \subset S$ contains all speech utterances that start with deictic references. It is specifically relevant to measure time alignment between a gesture and the deictic reference word in speech.

The data used throughout the evaluation was collected from seven persons interacting with the robot, and contains around 500 user inputs. Among these 500 user inputs there are 102 multimodal inputs with speech and gesture. A set of 89 gestures that corresponds to 87% recall on the set of 102 gesture inputs is used to evaluate gesture resolution on the given task. The pointing gestures were used to disambiguate between three lamps. Five other objects, four cups and one fan, were placed into the scene but not referenced by the user, the distance between these objects and the lamps was within the range of 10cm to 100cm.

5.2 Time Alignment

To define realistic time constraints, we have conducted a small user study. Based on the collected data we have analyzed the time correlation between gestures and speech. Therefore we have built two gesture sets, one containing manually transcribed gestures (G_{real}) and the second one containing the corresponding automatically detected gestures (G_{rec}). And we have created two speech sets, one containing any deictic reference (S) and the second one containing only speech utterances that start with deictic referring words (S_d).

The annotation of a gesture contains the time, (1) when the user starts to raise the arm, (2) when the hold phase

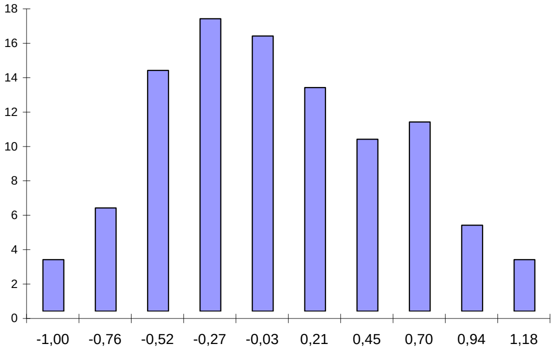

Figure 8: Time Correlation between start of gesture and start of speech signal

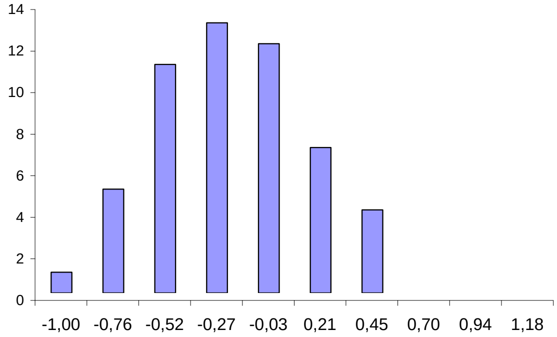

Figure 9: Time Correlation between start of gesture and start of deictic words

begins, (3) when the hold phase ends and the user lowers the arm, (4) the end phase of the movement. Figure 10 shows statistics on the recorded duration of the hold phase, which is between (2) and (3). In the following, when speaking about the start point of a gesture, we refer to the start of the hold phase (2).

Figure 8 shows the time correlation between the manually tagged start point of the gesture and the start of the speech signal. This gives information about the left bound for the time constraint for merging. The major part of the start points are between 0.52 seconds before the start of the speech signal and 0.7 seconds after the start of the speech signal. We have found that 1.0 seconds (gesture before speech) is a good boundary, also for the timestamps of the recognized gestures that are usually located between (2) and (3).

The above analysis is important to define time constraints, given the timestamps of a gesture and start and stop timestamps for the speech signal. However, to get a good understanding, how gestures are correlated, not only to the complete utterance, but more detailed to deictic words, the same data analysis has been performed on the subset $S_d \subset S$. The set S_d is a little bit smaller and contains 53 utterances. S_d contains only of those speech utterances, where the deictic word is located at the beginning of the utterance, like ”this lamp”. The result is shown in Figure 9. By estimating the data with a normal distribution we obtain a mean value of -0.3, which means that the gesture is detected 0.3 seconds before speech starts, and a variance of 0.14. It can be seen that in this analysis, the start point of a gesture is very closely correlated to the start time of the deictic word.

5.3 Gesture Recognition

In this evaluation, we used the pointing gesture recognizer described in [20]. Based on video images provided by a stereo camera, the system tracks the 3D-positions of the user's head and hands in real-time. The trajectories of the hands are classified in a run-on manner by a set of HMMs[1] in order to detect the typical motion pattern of a pointing gesture. A non-gesture HMM that represents any natural hand motions except pointing gestures is used as a threshold for the gesture models. Scaling the the output of this HMM influences the sensitivity of gesture detection. By decreasing the threshold, the percentage of successfully detected gestures (recall) increases as well as the number of false detections. Once a gesture has been detected, the line of sight between the centroids of the head and the pointing hand is used as an estimate for pointing direction.

The video data was collected with a static camera against cluttered background. The test persons were free to move within the camera's field of view at a distance of 2-4m. Given a comparatively low threshold setting, we could achieve a recall of 87%, with a precision of only 47% in the evaluated data set. This means that 13% of the gestures could not be detected, whereas the system reported 53% false detections. In our system, and in the presented evaluation, the false detections do not harm the overall performance, since the false detections could be sorted out by the described constraints.

In addition to the detection rates, the result of the resolution in the environment model is relevant for fusion success. Therefore 89 gestures were evaluated. The result of the resolution process is an n-best list of objects that match the pointing direction of the gesture well enough. In 94% of all cases the desired object was resolved as an element of the n-best list. However, only in 44% of successful resolutions, the best hypothesis was the correct result. In 56% of the successful resolutions, the correct hypothesis was found in the n-best list. These results emphasize the importance to use n-best lists for gesture results and not to rely on the best hypothesis. The best fitting element is determined by its ontology type.

[1] The HMMs were trained on 206 pointing gestures performed by 10 subjects. The training set is different from the set that is used for the multimodal evaluation.

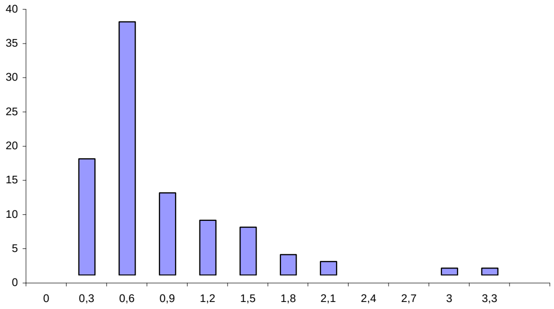

Figure 10: Time duration of a gesture (hold phase)

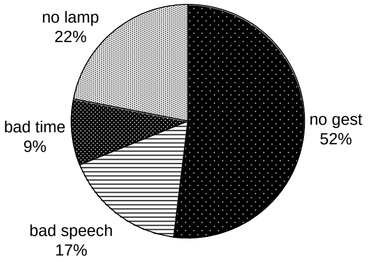

Figure 11: Contribution of the different components and constraints to failure of multimodal fusion. no gest - 52% (gesture not correctly recognized), no lamp - 22% (gesture could not be resolved to the correct pointing goal), bad time - 9% (incorrect timestamps), bad speech - 17% (speech recognition failed)

5.4 Fusion Evaluation

To evaluate the performance of the fusion approach, we evaluate the complete system from user input to fusion output.

On the 102 multimodal user inputs the overall success rate was 74% with an error rate of respectively 26%.

An observation of the errors shows that 52% (relative) of the errors are caused by failing to detect a gesture. In 22% of the error cases fusion could not be correctly performed because the gesture could not be resolved with high confidence. In 9% of the error cases gesture and speech could not be merged due to time constraints. 17% of the errors were caused by speech recognition errors, either due to incorrect segmentation or due to false recognition. See also Figure 11.

The failure of the time constraints is due to the fact that most of the gestures have been detected with incorrect time stamps.

5.5 Runtime Performance

The experiments have been performed on two PCs, one (3 GHz) camera PC with person tracking and gesture recognition, the second (2,2 GHz) PC with speech recognizer, communication architecture, database, multimodal fusion and dialog manager. The system is not yet optimized and still contains some delays within the communication architecture and polling cycles.

The speech recognizer runs in ≈ 0.8 time realtime [10] with runon decoding (recognition is started while user still talks).

The delays of the gesture recognizer, from the detection of the gesture until the arrival at the fusion component, are in most cases between 0.6 and 1.1 seconds. The multimodal parser uses a loop delay of 50 milliseconds for each parsing iteration, parsing time is below 20 milliseconds.

Dialog processing is also quite fast and takes on average around 50 milliseconds. In addition, a few milliseconds have to be added for database retrieval and gesture resolution as well as for conversion of the speech recognizer's output into semantic structures. Since the recognizer already sends parse trees, no extra time is lost by parsing the input with context free grammars, only precompiled conversion rules have to be applied that convert tree nodes to semantic structures.

The response time of the complete system mainly depends on the runtime of the speech recognizer. The output is generated in most cases between 1 and 2 seconds after segmentation stopped recording.

6. CONCLUSIONS AND OUTLOOK

We have presented an architecture for multimodal integration together with an evaluation of the system. The system runs on a humanoid robot to facilitate natural multimodal interaction. It is used for fusion of speech and pointing gestures in three dimensional space. We have described the parsing algorithm, the usage of constraints to find input tokens that can be merged, and construction rules for generating the merge result. We have shown a close correlation in time of speech and gesture on a small evaluation set. The fusion system is robust against false detection that occur in the vision based tracking of 3D gestures. Using n-best lists in the resolution process has helped to improve the results of the system.

In the future we want to evaluate our system on larger data sets and improve the response time and fusion robustness. Also, we want to extend the approach to merge speech with n-best gesture results, and observe different methods to merge n-best lists from various modalities, with score functions for comparing different combinations of elements.

7. ACKNOWLEDGMENTS

We would like to thank Petra Gieselmann for help in the data collection, Christian Fuegen for help with speech recognition, and the reviewers for helpful comments on this work.

This research has partially been funded by the German Research Foundation (DFG) as part of the SFB 588 - Humanoid Robots, within the FAME project by the European Union as IST-2000-28323, and by the European Commission under project CHIL (contract #506909).

8. REFERENCES

[1] T. Asfour, A. Ude, K.Berns, and R. Dillmann. Control of armar for the realization of anthropomorphic

motion patterns. In *The second IEEE-RAS International Conference on Humanoid Robots (HUMANOIDS 2001)*, pages 22–24, 2001.

[2] R. A. Bolt. "put-that-there": Voice and gesture at the graphics interface. In *Proceedings of the 7th annual conference on Computer Graphics and Interactive Techniques*, pages 262–270. ACM Press, 1980.

[3] B. Carpenter. *The Logic of Typed Feature Structures*. Cambridge University Press, 1992.

[4] P. R. Cohen, R. Coulston, and K. Krout. Multimodal interaction during multiparty dialogues: Initial results. In *Proceedings of the International Conference On Multimodal Interfaces*, 2002.

[5] A. Corradini, R. M. Wesson, and P. R. Cohen. A map-based system using speech and 3d gestures for pervasive computing. In *Proceedings of the 4th IEEE International Conference on Multimodal Interfaces*, 2002.

[6] M. Denecke. Object-oriented techniques in grammar and ontology specification. In *The Workshop on Multilingual Speech Communication*, pages 59–64, Kyoto, Japan, 2000.

[7] M. Denecke. Rapid prototyping for spoken dialogue systems. In *Proceedings of the 19th International Conference on Computational Linguistics*, Taiwan, 2002.

[8] J. Eisenstein and C. M. Christoudias. A salience-based approach to gesture-speech alignment. In *Proceedings of the Human Language Technology conference / North American chapter of the Association for Computational Linguistics annual meeting*, 2004.

[9] M. Finke, P. Geutner, H. Hild, T. Kemp, K. Ries, and M. Westphal. The karlsruhe-verbmobil speech recognition engine. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing, ICASSP-97*, Munich, Germany, 1997.

[10] C. Fuegen, H. Holzapfel, and A. Waibel. Tight coupling of speech recognition and dialog management dialog-context dependent grammar weighting for speech recognition. In *Proceedings of the International Conference on Spoken Language Processing*, 2004.

[11] P. Gieselmann, C. Fuegen, H. Holzapfel, T. Schaaf, and A. Waibel. Towards multimodal communication with a household robot. In *Proceedings of the International Conference on Humanoid Robots*, 2003.

[12] *Third IEEE International Conference on Humanoid Robots - Humanoids*, Karlsruhe, Germany, 2003.

[13] *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, 2004.

[14] M. Johnston. Unification-based multimodal parsing. In *COLING-ACL*, pages 624–630, 1998.

[15] M. Johnston, S. Bangalore, G. Vasireddy, A. Stent, P. Ehlen, M. Walker, S. Whittaker, and P. Maloor. Match: An architecture for multimodal dialogue systems. In *Proceedings of ACL*, 2002.

[16] M. Johnston, P. R. Cohen, D. McGee, S. L. Oviatt, J. A. Pittman, and I. Smith. Unification-based multimodal integration. In *Proceedings of the Thirty-Fifth Annual Meeting of the Association for Computational Linguistics*, pages 281–288, 1997.

[17] M. Kaur, M. Tremaine, N. Huang, J. Wilder, Z. Gacovski, F. Flippo, and C. S. Mantravadi. Where is "it"? event synchronization in gaze-speech input systems. In *Proceedings of the 5th International Conference on Multimodal Interfaces*, pages 151–158. ACM Press, 2003.

[18] S. Kettebekov, M. Yeasin, and R. Sharma. Prosody based co-analysis for continuous recognition of coverbal gestures. In *Proceedings of the 4th International Conference on Multimodal Interfaces*, Pittsburgh, USA, 2002.

[19] S. C. Levinson. *Pragmatics*. Cambridge, England: Cambridge University, 1983.

[20] K. Nickel and R. Stiefelhagen. Pointing gesture recognition based on 3d-tracking of face, hands and head orientation. In *Proceedings of the Fifth International Conference on Multimodal Interfaces*, Vancouver, Canada, Nov. 5-7 2003.

[21] S. L. Oviatt, A. DeAngeli, and K. Kuhn. Integration and synchronization of input modes during multimodal human-computer interaction. In *CHI*, pages 415–422, 1997.

[22] N. Reithinger, J. Alexandersson, T. Becker, A. Blocher, R. Engel, M. Lckelt, J. Mller, N. Pfleger, P. Poller, M. Streit, and V. Tschernomas. Smartkom - adaptive and flexible multimodal access to multiple applications. In *Proceedings of the Fifth International Conference on Multimodal Interfaces*, 2003.

[23] R. Sharma, V. Pavlovic, and T. Huang. Toward multimodal human-computer interface. In *Proceedings of the IEEE*, volume 86, pages 853 – 869, May 1998.

[24] H. Soltau, F. Metze, C. Fuegen, and A. Waibel. A one pass- decoder based on polymorphic linguistic context assignment. In *Proceedings of the Automatic Speech Recognition and Understanding Workshop, ASRU-2001*, Madonna di Campiglio, Trento, Italy, December 2001.

[25] R. Stiefelhagen, C. Fügen, P. Gieselmann, H. Holzapfel, K. Nickel, and A. Waibel. Natural human-robot interaction using speech, gaze and gestures. In *Proceedings of the International Conference on Intelligent Robots and Systems*, Sendai, Japan, 2004.

[26] I. Wachsmuth. Communicative rhythm in gesture and speech. In *Gesture-Based Communication in Human-Computer Interaction: International Gesture Workshop, GW'99*, Gif-sur-Yvette, France, March 1999.

[27] L. Wu, S. L. Oviatt, and P. R. Cohen. Multimodal integration - a statistical view. *IEEE Transactions on Multimedia*, 1(4):334–341, 1999.