

EXTRACTING PROCEDURAL AND CONTEXTUAL ELEMENTS OF VERBAL COMMUNICATION TO INSTRUCT A SHAPE CONCEPTUALIZATION SYSTEM

György Kuczogi, Imre Horváth, Zoltán Rusák,
Jouke Verlinden, Johan Jansson, Joris S.M. Vergeest

Department of Design Engineering
Delft University of Technology
Delft, The Netherlands

ABSTRACT

This paper presents a framework of processing verbal communication applied in computer supported shape conceptualization. As typical form of communication between the user and the computer system is supposed the one in which the user controls the system. In the practice it means that we can suppose to have instructional sentences as input. Instructional sentences have a more or less regular grammar, but passive, compound and incomplete sentences need special handling. The approach of processing follows the context sensitive matching strategy and relies on the known levels of information communication (as applicable in conversion of the verbal instructions to executable system commands). In the interpretation process, we try to achieve the semantic level of the communication. First, the procedural elements, the so-called actions, are extracted from the user communication. Afterwards, depending on the necessary operand list of an action, we can extract the contextual elements, namely the modeling types, modeling items, indirect and direct values. Hence, we extract the requested procedural elements based on their respective category.

KEYWORDS

Verbal instructions, shape conceptualization, user interface, interpretation strategy, communication unit

1. Introduction

The most fundamental communication format of human being is verbalism (talking). However, the verbal communication, which has several advantageous features such as expressiveness, is not actually used in the recent design systems. Former research explored that the its non-deterministic nature is the source of most of the difficulties in the verbal communication between human and computer. On the other hand, the non-deterministic nature can be beneficial for conceptual design to capture and represent vagueness of human thinking. Thus, it seems to be expedient to use verbal communication to support, for instance, shape conceptualization.

Characteristics of a typical verbal input

Parties in the man-machine communication have no equal role, namely, the human is supposed to control a machine. Therefore, we can postulate that there can be instructive elements of the human communication towards machines. The instructional English has a rather strict and regular grammatical structure ("do that this way"), that is an advantageous feature from the viewpoint of the implementation of a computer aided conceptual design system.

In addition to that, there is a limited number of frequently used sentences, instruction patterns (e.g., "Stop", "Go further" or "O.K."), especially in the spoken language. They mostly carry the context of a review or a decision about a previously done action, and they most probable are of incomplete grammatical structure. They can also be defined as exceptions, when the interpretation process for regular instructions fails. We call this group of sentences "reactive instructions". The

variety of the frequently used reactive instructions is quite low. We can estimate less than a hundred of them in a general man-machine communication for shape conceptualization.

Passive and compound sentences may also cause irregularities in communicative sentences. Compound sentences contain more than one action to be executed. They often have specific parts that are mentioned only once, but concern more than one action.

Related research

Cohen *et al.* [1] employ an agent-based software architecture to interpret and process multimodal input as system commands while they operated a simulation system. McTear [2] gives an overview of spoken language technology, including natural language understanding and dialog management. Bolt *et al.* [3] pioneered in exploring the combined way of using speech, gesture, and gaze in the interface to allow people to interact in real-time with a graphics display by pointing, looking, asking questions, and issuing commands. A virtual reality-based conceptual shape design system is introduced by Dani and Gadh [4]. They use the VR hardware and software technology to create an interactive 3D environment, in which the designer can use a combination of hand gesture, voice input, and keyboard to generate and view mechanical components.

2. Overview of the input processing

System architecture

We process the verbal input based on the known levels of information communication. Our aim is to achieve the possible highest communication level, in order to speed up the conceptualization process. Application of higher level communication allows the designer to concentrate more on his particular design task, and to be bothered less by the system. Being the focus of this research, we consider extracting and interpreting the semantic elements of the user input as a key step in reaching semantic level of communicating information.

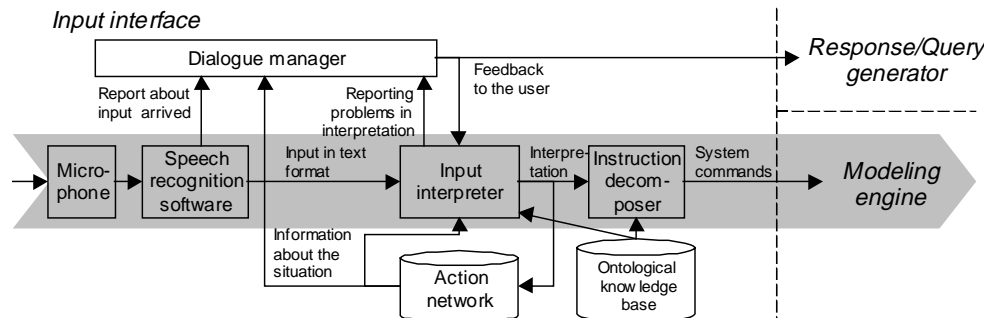


Figure 1. Architecture of the user interface

For the basic levels of communication (i.e., physical, statistical, syntactical), we use standard and commercial tools (Figure 1). A microphone captures the physical signals, and speech recognition software is applied to work on the syntactical level. The input interpreter is responsible for extracting the semantic information from the user

input. In most cases, it is not a straightforward process. The dialog manager inspects the whole interpretation process, and intervenes into the communication if the input information stream is not fully comprehensible, or the knowledge base of the system cannot provide all information that is necessary for the interpretation. The interpreted commands are decomposed to low-level system commands and forwarded to the modeling engine for execution. However, the interpreted commands are also stored in an action network, since they represent the actual situation of the design process.

Methodology of interpretation

Processing of verbal communication consists of three main phases (Figure 2). In the preparation phase we reduce the complexity of the instructions by detecting passive and compound sentences. Passive sentences have to be filtered out, since their grammar significantly differs from that of the active sentences. In the case of compound sentences, we have to forecast from the preparation process whether the sentence contains more than one action. Another task in this phase is the pre-selection of reactive instructions. Reactive instructions are collected in a dedicated, separate thesaurus. Based on their relatively low variety, they are identified based on mapping them "as they are" to the thesaurus. If a reactive in-

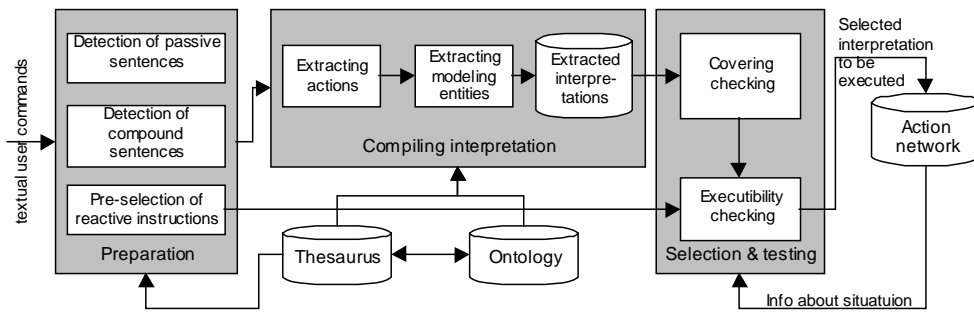


Figure 2. The scheme of processing of user instructions

them to a thesaurus. However, coordination of the search for matching instances is directed by the semantic relationships of elements in the user instructions. Therefore, it is a context sensitive matching process. The first step is the extraction of the action part of the instruction. The found action prescribes the type of metric content(s) that we should and can expect. These semantic relationships between actions and their operands are described in a shape conceptualization ontology. In the second step we recognize, or find out, the operands that most probably appear in the user instructions. A given interpretation is considered acceptable if all the required operands of the action are recognized.

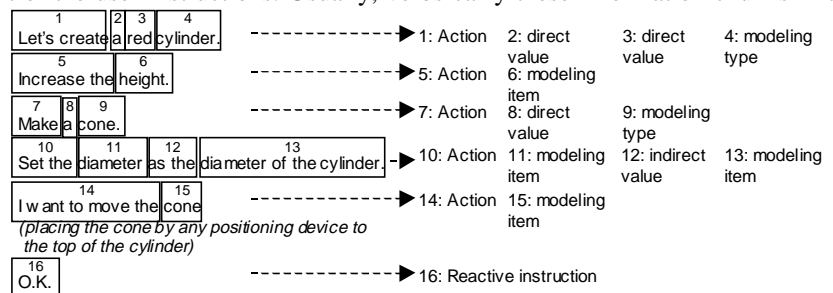
The interpretation process may produce more than one possible interpretation. To check which interpretation is correct, a syntactical test - called covering checking - is applied. The covering checking compares the interpreted parts of the user instruction to the whole sentence and examines the overlap. If the identified meaning of a particular interpretation covers the sentence completely, and the order of the acquired meaning is appropriate in the sentence, then the given interpretation is considered appropriate. The chance of finding more than one appropriate interpretation for the same instruction is quite low, but then they need further investigation. The second test - called executability checking - is therefore semantics based. It tests only instructions, which according to the covering checking, are processed successfully. The test examines whether the action can be executed in the given situation by considering the previous instructions. Since the interpretation of reactive commands is done on a direct way, obviously the covering checking can be skipped, only executability checking has to be executed. If the second test is also positive, the interpretation of the instruction is accepted. The accepted interpretation of user instructions is stored in the action network waiting for further processing and execution.

3. Elements of the input processing

Communication units of the input instructions

Figure 3 show a simple scenario on how we could create a pencil from a cone and a cylinder by verbal instruction. The example clarifies the communication units used in the verbal instructions. We differentiate three basic types of communication units in the human input.

- Actions represent the procedural part of the user instructions. Usually, verbs carry these information chunks in the input sentences. However, often not only the verbs, but also long phrases can be associated to the procedural part of the given instruction (Figure 3. fifth instruction).



The other two types describe the contextual parts of the input:

- Modeling entities, which we can categorize further to modeling types and modeling items. Model-

Figure 3. Identified communication units in an example scenario (Creating a pencil from a cone and a cylinder)

struction is detected, the appropriate meaning described in the shape conceptualization ontology is extracted directly from the thesaurus.

In the second phase, the actions and the related metric contents have to be extracted from the user instructions. The recognition of the elements of sentences is based on a syntactical pattern matching of

ing types are assumed in the background knowledge of both the system and the user. They represent the domain of the common understanding. Modeling items delineate the actual model created by the user.

- Values, which also have two further categories, which typically appear in the user instructions. The first category is formed by direct values (which can still be fuzzy), like "50 mm", "red", or "long". The second category is indirect value. Indirect values are closely related to the actions. They represent a value in an indirect way. In the fourth instruction of Figure 3, "as the diameter of the cylinder" phrase represents a number. However, the indirect value needs processing to determine its direct value.

System units to cover the communication units of the input

The implementation follows the natural separation of the communicated information to actions (the procedural part) and modeling entities (contextual part). Table 1 shows the elementary functionality categories of the actions and modeling entities. These elementary functionalities are often not suitable to carry high-level semantic information, therefore we have to use compound structures, as well. Very complex structures are typical for modeling entities, but in the case of actions, we can use elementary types in a wide range. For instance, a "move" command can be represented by a 'set' type of action very easily.

Table 2 shows the communication units of the input instructions implemented by the system units. The action classes, secondary action classes, the modeling entity classes, and the predefined modeling entity instances form the knowledge base of the system. The action instances and modeling entity instances belong to the concrete design project of the user, they represent the model to be designed. In the following part, we give a functionality description for each format of the system units.

Action classes: The actions in the input information are represented by action classes. Each action class contains a list about the possible and necessary operands. Action classes are defined in the ontological knowledge base.

Modeling entity classes: Modeling entity classes represent the structure and the main semantical associations of the contextual part of the system knowledge. Any modeling entity class can have several attributes, which are also modeling entity classes. The very basic classes symbolize only computational aspects (like integer, real number...), but we can achieve very high level semantic meanings with compound classes. Modeling entity classes are defined also in the ontological knowledge base.

Modeling entity instances: Modeling entity instances represent the geometric and semantic content of the current model. They can be instantiated from the classes directly, or they can be composed from other instances, and modified by actions. Modeling entity instances are defined in the database of the current model.

Predefined modeling entity instances: Predefined modeling entity instances describe the values and constants. They are usually applied for low-level modeling entities such as numbers, colors etc. The other application field is the use of libraries of standardized and complex objects, such as a series of bearings or bolts, described by high-level modeling entities. Predefined modeling entity instances are defined in the ontological knowledge base.

Secondary action classes: Secondary action classes are a subset of action classes. The difference is in that the action classes can describe any type of procedural input of the user, but secondary action classes intend to cover indirect values only. Therefore, they are specified to obtain and provide a requested value (or modeling entity), and their phrases usually do not show procedural characteristics (e.g. "double of" or "as high as").

Action instances: Action instances are special elements in that sense they do not only cover a part of the user input, but the entire one.

They are the interpretation of the user input. They represent the meaning of the user input in a procedural way,

System unit	Elementary functionalities
Actions	Get
	Set
	Create
	Delete
Modeling entities	Integer
	Real number
	Text
	Link

Table 1. Elementary functionalities of system units

Communication units of the input instruction		System units
Actions		Action classes
Modeling entities	Modeling types	Modeling entity classes
	Modeling items	Modeling entity instances
Values	Direct values	Predefined modeling entity instances
	Indirect values	Secondary action classes

} Action instances

Table 2. Covering the communication units of input instructions by the system units

as a result of the interpretation process. They also differ from the other five system units, since the other elements are searched for during the process, but action instances are built during the process. A sequence of the action instances represents the action network, which is part of the database of the current model.

4. The interpretation process

Detection of communication units

Each system unit has a list of phrases attached (called thesaurus). These phrases are dedicated to be the linguistic equivalents of the represented content, and they can be detected in the input sentences. A phrase can contain only one, or, in other cases, more than one word. In general, detection of a communication unit happens to be by matching the thesauruses of the expected set of units against the sentence at hand. During the matching procedure, we may find cases where more than one phrase can fit to a particular part of the input sentence. However, none of the words of the input sentence should be covered by more than one phrase of communication units. For instance, we can find both the "create" and the "let's create" phrases in the first instruction of the example above (Figure 3). If the matching phrases belong to the same system unit, like in this case, the system assumes that the longest matching phrase (counted by letters) has the highest chance to be the intended phrase of the user. However, if the phrases belong to different system unit, each case has to be further investigated as a possible good solution. The best one can be selected based on the semantic contents and relationships of the detected communication units after the interpretation process.

Actions, values, indirect values and modeling types can be filtered out from the user instructions in the presented way. However, modeling items usually not, since they are new objects, therefore, they often do not have plausible or commonly accepted names. The system provides for each object a unique identifier, and the user can also give a name to the created objects, but in the real life people hardly use these kinds of identification of objects in verbal communication. People rather refer to an object by one of its characteristic feature like its type, color, or size. Therefore, if we search for a modeling item in the input, we can recognize it by matching the types or values of the possible items. We also have to check, whether the instruction contains pronouns, which refer to the modeling item(s) mentioned in the previous instruction.

Steps of interpretation of user instructions

The interpretation of communicated instructions and the extraction of communication units from the instructions are simultaneous processes, they influence each other and they form a loop. The output of the extraction is the input of the interpretation, the output of the interpretation determines what to extract in the next loop. In order to avoid overlapping of the extracted units in the text, all words of the input sentences are marked, which belong to an identified communication unit. In the process of extraction of a particular unit, the system does not quit after the first hit, but tries to identify all possibilities. If multiple solutions are found, then the system creates independent copies about the already interpreted units, and the interpretation of each case can continue.

The very first step is extracting the action from the instruction (Figure 4). If the action is identified, the system can generate the skeleton of the action instance, which will represent the translation of the input instruction. Based on the implemented operand list of the extracted action, we can search for direct and indirect values, modeling types or modeling items. The operand list specifies not only the type and the level of necessity of the operands, but it may give information about the assumed order of communication units in the sentence.

The search of operands always starts with the indirect values, since they are also a kind of actions and they can refer to other communication units of the input instruction. If indirect value is found, the search continues with the operand list of the indirect value, which can contain direct values or modeling items. Theoretically, modeling types and further indirect values could be also in the operand list of indirect

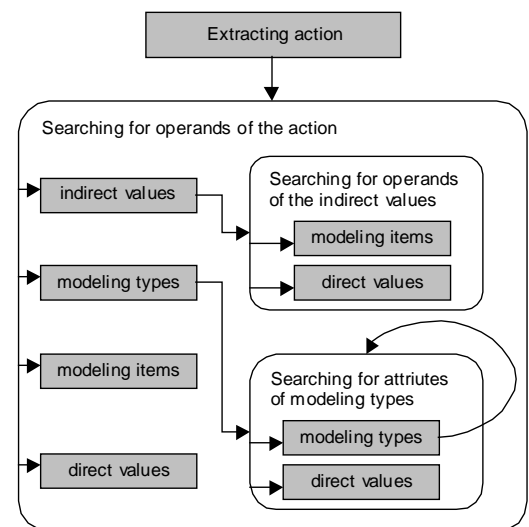


Figure 4 Strategy of extracting communication units from verbal instructions

values, but it is unusual and ambiguous in the human verbalism, and we suppose that they do not appear in the user input. When the indirect value with all its operands completely extracted, their place in the sentence is substituted by a direct value.

In the case of "create" type of actions the operand list always contains a modeling type operand. If the modeling type is identified, the attributes and values of the attributes of the modeling type have to be determined. They can be extracted either from the input instruction or from the knowledge base. In the first instruction of the example shown in Figure 3, the color of the cone is specified by the user, but the information about the height, angle etc. have to be set by default values, which have to be extracted from the ontology. These default values are not necessarily constants; they can be adapted to the particular situation. On the other hand, if no implemented modeling type is mentioned, then the user probably introduces something new. In such a case an abstract design method can be applied. The system creates a modeling item based on the most basic modeling type and all the attributes are created and filled in by the user. The modeling item can be displayed only as label, as long as the geometry is not defined.

Finally, if more than one result has been found, the system has to decide which interpretation approaches the intent of the user best. In the evaluation we consider the order of the extracted communication units in the sentence compared to the assumed orders in the operand list, and whether all the words of the sentence covered by an extracted communication unit.

5. Summary and conclusions

The paper has presented a framework about processing of verbal communication in the field of computer aided shape conceptualization, with the main focus on the interpretation of the verbal instruction. We distinguish five communication units (action, modeling item, modeling type, direct value, indirect value), with which we can describe the assumed instructional type of user input. Each unit has a thesaurus, which contain phrases. These phrases are dedicated to be the linguistic equivalents of the represented content, and they can be detected in the input sentences. The strategy of processing follows the context sensitive matching approach, namely we decide on the unit to be searched based on the content of the previously extracted units.

Conclusions:

- The presented method is above all suitable for processing instructional style of sentences. Processing of compound or passive sentences needs further investigation.
- The real benefits of verbal communication can be taken with the jointed application of hand gesture detection. The combination of modalities not only makes the interaction easier and more natural, but it usually reduces the complexity of verbal instructions, therefore it makes the interpretation process more robust.

REFERENCES

1. Cohen, P. R., Johnston, M., McGee, D., Oviatt, S., Pittman, J., Smith, I., Chen, L., and Clow, J.. QuickSet: Multimodal interaction for distributed applications, Proceedings of the Fifth International Multimedia Conference (Multimedia '97), ACM Press, pp 31-40. 1997
2. McTear, M.F.: Spoken Dialogue Technology: Enabling the Conversational User Interface http://www.infj.ulst.ac.uk/~cbdg23/survey/spoken_dialogue_technology.html
3. Bolt, R. A. and Herranz, E. "Two-Handed Gesture in Multi-Modal Natural Dialog" in Proc. of ACM Symp. on UIST, November, 1992 pp 7-14.
4. Dani T. and Gadh R., "COVIRDS: A New Paradigm for Conceptual Shape Design using Virtual Reality", accepted for publication in Computer Aided Design Journal, Elsevier Science Inc., July, 1996.