The Spoken Language Navigation Task

J. Gurney, E. Klipple and T. Gregory
Army Research Laboratory
Adelphi, Maryland 20783
gurney@arl.mil

Abstract

The spoken language navigation task requires a user to navigate through a virtual landscape (displayed on a monitor) using only natural language, that is, without the use of a mouse or keyboard. This "hands-free/eyes-free" human/computer interaction has been sought after by users who must work in moving vehicles and otherwise disruptive, busy, and distracting conditions. An additional reason for exploring this kind of task is to study how users might employ the significant expressive power of human language. The key issues we address are: (1) the representation of linguistic meaning, (2) the representation of a non-linguistic perceived scene as the current state of the virtual world, and (3) the communication between the two in order to accomplish the navigation task.

Introduction

The spoken language navigation task requires a user to navigate through a virtual landscape (see figure 1) using only natural language. One value of this project has been the realization of the importance of various types of motion through space and how these can be represented and exploited. The authors have created a fully operational speech and natural language interface (NLVR) to a real-time 3-D virtual reality system (Gurney & Klipple 1998; Gurney, Klipple, & Voss 1996). The NLVR is an interesting test bed for detailed semantic interpretation of spatial and motional language.

The key issues we address are: (1) the representation of linguistic meaning, (2) the representation of a non-linguistic perceived scene as the current state of the virtual world, and (3) the communication between (or integration of) the two in order to accomplish the navigation task. Examples of the sentences used in the navigation task include: "turn around", "drop down to the ground", "zoom northward", and "veer off thirty six degrees to your right."

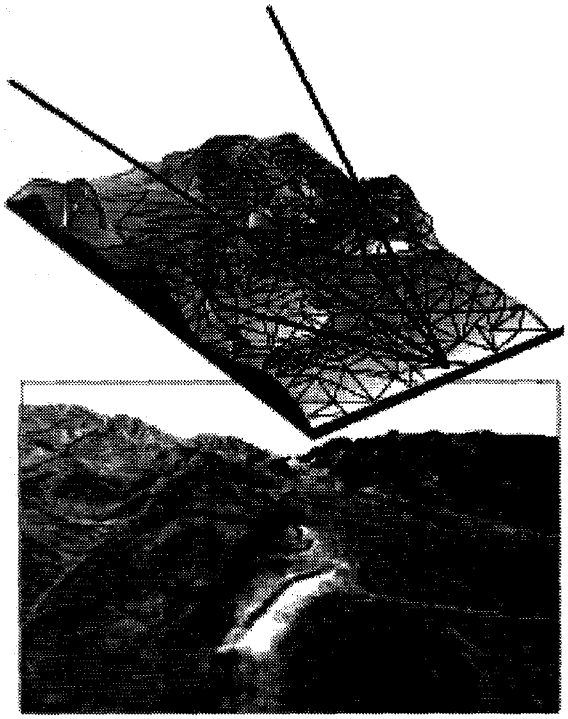

Figure 1: Terrain Data Rendered as a Landscape

We provide a representation of word, phrase, and sentence meaning that consists of lexical decomposition into primitive actions, modifiers and functions over representations of state based on our theory of fine-grained lexical meaning. This allows us to compute the communication across levels of representation. Meanings of phrases and sentences are found by coherently composing the primitives and state values specified by the words. This composition is an extension of traditional formal compositional semantics (Larson & Segal 1995).

The Two Ends of the System

At one end of our system we have a list of words spoken by the user[1]; at the other end we have the data structures that represent the virtual landscape that the virtual geographic information system (VGIS) displays on the user's monitor. Of course, these two levels of representation are not directly commensurable. Hence,

[1] ViaVoice (IBM speech recognition software) is used to generate the word list from the user's speech.

our problem is to pass appropriate and correct information between them. In the next subsection we will describe the geometrical representation at the VGIS end. In the following subsection we will discuss samples of the language spoken by the user at the word list end. Our solution to the information flow problem is to interpose three different levels of representation between the two ends, each responsible for supporting different aspects of the processing task. In the next full section we will work through an example from end to end as information flows, up and down, from one level of representation to another.

The VGIS Rendering Engine

The rendering engine is a Virtual Graphic Information System (VGIS) module (Koller *et al.* 1995) which was developed by the Georgia Tech Research Institute. As shown in figure 1, its function is to paint a picture of the virtual scene as a screen image on the user's monitor. The terrain of the VGIS world is stored in a quad tree as a large set of elevation (and other) data. For purposes of rendering, this data is first extracted from the quad tree and triangulated into a wire frame representation, as depicted in the upper part of figure 1. The rendering engine computes a view of the scene from this wire frame data which it sends to the monitor. Typical frame rates for updating the changing scene are from six to sixteen frames per second, running on a Silicon Graphics Octane.

The illusion of navigating through the virtual landscape over terrain is created by moving the point-of-view (POV). The POV is the apex of the four-sided conical frustum shown in the upper part of figure 1. The POV moves relative to the wire frame terrain data by changing a set of numbers that represent the position and attitude of the POV relative to the terrain elevations, latitudes and longitudes. The frustum projects onto the terrain data set giving the segment of terrain that is rendered on the rectangular monitor. The user who is looking at the monitor will, of course, represent in his mind a visual scene of hills, mountains, gorges, etc. However, it is important for us to realize that there is no computational representation of any of this; the rendering engine merely turns on colored pixels on the monitor. The only representations we have are the data set of {elevation, latitude, longitude}, color data, and the POV position and attitude. None of this captures any any higher level concepts like "hill" or "gorge".

From the user's perspective, navigation is moving through a scene in three-dimensional space — along with turning and/or tilting one's gaze around and up and down, — all of this at various linear and angular speeds. Computationally, navigation reduces to incrementing the x, y, and z positions of the POV and rotating the view frustum around the x, y, and z axes. (The rendering engine takes care of painting the appropriate picture on the monitor for the moving POV at each frame.) Thus our ultimate goal in implementing spoken language navigation is simply to generate appropriate numerical values for the various linear and angular positions of the POV — that is, appropriate for proper and reasonable understanding of whatever the user utters while engaged in the spoken language navigation task.

Language Used for Navigating

Examples of the sentences used in the navigation task include: "turn around", "drop down to the ground", "zoom northward", and "veer off thirty six degrees to your right." Obviously, none of these sentences refers to or mentions any of the set of numbers that determine the position and attitude of the POV. Our claim is that, in the context of the navigation task, there are appropriate mappings from these and various other sentences to the POV numbers and/or sequences of these numbers. Actually, in most cases it will be streams of numbers that are required; for the sentence "turn around" we want rotation around the z axis to increment at a reasonable angular rate. This mapping depends on several factors including word meaning, sentence structure, the POV position and attitude itself, along with the linear and angular velocities of the POV. To illustrate the importance of velocity and/or speed, consider the meaning of "turn faster," which depends on some actual angular speed (if there is current turning) or an angular speed that is in some way current or relevant (if there is no turning at the moment).

The reason for emphasis on streams of POV positions is that we want almost all responses to the user to be perceptible, smooth motions. In the VGIS world sudden large changes in position are, of course, possible. But these visual discontinuities are disorienting to the user and make it difficult for him to keep track of his motion through the landscape.

The navigation expert module and the API were designed to generate the appropriate streams of POV position and attitude numbers based on the interpretations of the representations of spoken navigation task sentences. In the next section we begin by discussing the natural language parser and logical form generator which produces a structural representation of the sentence. This takes us down two levels of representation. Following that we discuss the navigation expert module which generates and uses a third lower level of representation.

An Example from Top to Bottom

In this section we will work through an example from end to end as information flows from one level of representation to another. We begin at the user's end with a string of words.

Parsing and Logical Form

We will consider the command "Look fifteen degrees to your right." The meanings of navigation task sentences depend partially, albeit importantly, on syntactic structure and closely related logical form. The list of words:

```
[look, fifteen, degrees, to, your, right]
```

is parsed (by the REAP parser (Garman ms)) into the following parse tree which displays the syntactic representation:

```
VP (cat v)
    Pro
    Vbar (cat v)
        V "look" (root "look")
        NP (cat n) (lic syn adv obj)
            NumP ((cat number)
                Numbar ((cat number)
                    Num "fifteen"
            Nbar (cat n)
                N "degrees"
        PP (cat p)
            Pbar (cat p)
                P "to" (root "to")
                NP (cat n)
                    NP (cat n)
                        Nbar (cat n)
                            N "your"
                    Nbar (cat n)
                        N "right"
```

The logical form (LF) can be generated directly from the above syntactic form:

```
[pro:X1,
     [v:look:X2:X1,
         [np:measure:X2:X3,
             [number:fifteen:X3,n:degree:X3]],
         [p:to:X2:X4,
             [n:your:X4,n:right:X4]]]]
```

Brackets in the LFs specify logical scope; thus [pro:X1, ...] specifies that X1 is a pro (null) subject of this imperative sentence, hence the addressee. The verb v:look:X2:X1 specifies an event X2 (Davidson 1967) that is a looking event, with X1 as the object that looks. The verb modifier [np:measure:X2:X3, [number:fifteen:X3,n:degree:X3]] specifies that the distance of the event X2 should measure fifteen degrees. The verb modifier p:to:X2:X4, [n:your:X4, n:right:X4] specifies that the direction of X2 be right. The multiple occurrences of X2 in this LF determine that it is the looking rather than something else that is to be fifteen degrees, etc. All of these logical facts are actually determined by the syntactic form (above) of the sentence ; the LF generator computes the LF by rewriting the syntactic for into a first-order logic form.

This LF determines important parameters of the event referred to by the verb "look". Although there can be various aspects of any event, the structure of a natural language verb phrase (VP) impels a speaker to describe events in terms of a small set of universal parameters (Tenny 1987; Klipple 1991; Ernst 1989). Those parameters most relevant here are: direction (e.g., right), distance (e.g., fifteen degrees), theme (e.g., the addressee), manner (e.g., ten degrees per second), and goal or endpoint (e.g., southeast). In other words all spatial motion events (hence virtually all of our spoken language navigation task events) can be partially decomposed into: direction, speed, distance, and goal.

The Navigation Expert Module

We designed a next lower level of representation according to this natural language decomposition of events. This level is analogous to the conceptual level motivated by the theory of human competence (Jackendoff 1997). The navigation expert module maps the LF onto this level. We will explain this level as we follow the processing of the LF by the navigation expert module.

The first portion of the navigation expert's output log for our example is shown below. This is a sequence of diagnostic messages that we can use to track the progress of the navigation expert as it attempts to interpret the above LF.

```
MEANING of look is pitch

MEANING of measure:15.0:degree is
        set(distance,pitch,15.0)

to:right has NO MEANING
```

Here the expert tried to map the LF into a complete lower level representation by first mapping v:look:X2:X1 into **pitch**. **pitch** is one of the set of primitive motion actions that cause the various possible motions of the POV that we mentioned earlier. These motions are divided among: **translate** for linear motion in the x-y plane, **ascend** for linear motion along the z axis, and **rotate**, **pitch**, and **roll** for angular motion around each of the three axes centered in the POV. These primitive motion actions serve a dual role; they are both meanings of verbs in the LF's and VGIS functions that cause motion of the POV. Each of them is modeled on the linguistically motivated event decomposition mentioned above.

Thus for each primitive motion there is a set of parameter-setting actions that will (if used) set new values for **goal**, **distance**, **speed**, and **direction**. **set(distance,pitch,15.0)** is one of the parameter-setting actions that is possible for **pitch**.

In the current example, [np:measure:X2:X3, [number:fifteen:X3, n:degree:X3]], the piece of LF that specifies the distance of angular motion, was mapped onto the parameter setting action **set(distance, pitch, 15.0)**. Next we see that the navigation expert failed to find a meaning for the remaining piece of LF [p:to:X2:X4, [n:your:X4, n:right:X4]]. The primitive action **pitch** has no parameter (no **speed**, **direction**, **distance**, or **goal**) that can have a value corresponding to to:right. So this first attempt to interpret the LF crashed. It is worth noting that our method of interpretation here does not settle for partial interpretations as is popular in many other applications of natural language processing. It is essential to our method that interpretations be complete.

At this point the expert finds an alternative meaning for v:look:X2:X1, namely, the action **rotate**.

```
MEANING of look is rotate

MEANING of measure:15.0:degree is
        set(distance,rotate,15.0)

MEANING of to:right is
        [set(distance, rotate,90.0),
        set(direction,rotate,positive)]

MODS set(distance,rotate,15.0) and
     set(distance,rotate,90.0)
     are INCOHERENT
```

Now we see that it has selected meanings for the two PPs that are incoherent. This is because [p:to:X2:X4, [n:your:X4,n:right:X4]]]] is ambiguous between a telic (bounded) reading calling for a distance (90.0 degrees) and an atelic reading calling for a direction only. After selecting the atelic meaning (below) the expert has found a complete interpretation of the LF for "look fifteen degrees to your right".

```
MEANING of to:right is
        set(direction,rotate,positive)

ALL SECONDARY ACTS APPROVED:
        set(distance, rotate,15.0)
        set(direction,rotate,positive)

PLAN APPROVED for  rotate

SECONDARY ACTS WERE EXECUTED on assb33
PRIMITIVE ACT WAS EXECUTED on assb33
```

The expert then executes the two parameter-setting acts above. Next the primitive motion action **rotate** is executed. This finally is the level that has access to the VGIS POV. **rotate** increments the frustum of the POV around its z axis, in steps at some angular rate. Like each of the other primitive motion actions, **rotate** must cause motion at some particular rate in some particular direction, either indefinitely (if the action is atelic) or for some particular distance or to some particular goal, that is, angular position (if the action is telic).

Spoken language need not be this complete. Our example fails to specify a speed of rotation which, therefore, must be available to **rotate** from some other source. In fact **rotate** operates by consulting the navigational state of the system. This state includes values for all parameters for all motion actions. This is why the secondary acts, the value setting acts, were executed before **rotate**. Deployment of this complete set of state values for all of the relevant parameters raises further questions about proper or at least reasonable interpretation in some cases. In our example, **rotate** queried the state for its angular speed. It turns out that in our implementation the current speed is always left over from the last **set(speed, ACT, X)** action. So in the following sequence the speed of the last **translate** act (which is the meaning of "go north") will be faster than the first **translate** act (which is the meaning of "start moving to your left").

```
start moving to your left
speed up
stop
go north
```

In other words, there is a persistence of expectation built into our system. This is just one possible pragmatic strategy, of course.

The state also includes current values for the POV numbers. And rotate simply increments one of these — the z axis angular position value — until the distance fifteen degrees is reached.

Comments on the Levels

We will now comment on some of the differences among the levels of the NLVR system.

The action level has access to all navigation state values including speeds, positions, goals, and so on. The primitive motion actions can query these values and reset them. At the next higher level, the navigation expert can use LFs to gain access only to the *representations* of the actions at the action level. This access depends on a lexicon that records meanings of the LF words as actions and action parameter values. The typical non-logical or lexical meaning constraints for sentences are, therefore, enforced indirectly -- by appeal to which parameter settings are possible, with which actions. Going up one more level, the syntactic level represents logical constraints on meaning. This was why the first try at [p:to:X2:X4, [n:your:X4, n:right:X4]] had no meaning; X2 had been incorrectly chosen as a pitching action.

The overall picture is one in which interpretation of spoken language has new constraints applied at each deeper level down to the bottom, where a fully specified primitive action can be performed as a computation over a numerical data structure. At higher levels, we appear to have a much richer array of concepts: north, further, zoom, around, etc. A great deal of this can be made to cash out in terms of brute settings of parameters — but according to a reasonable plan from the user's perspective. Our ontology of primitive actions and their parameters, which derives from previous work on lexical semantics, makes the task of integrating spoken language with virtual reality more straightforward than it might have been otherwise.

Acknowledgments

The authors would like to thank Joe Garman for the REAP parser and Robert Winkler for solving some software problems.

References

Davidson, D. 1967. The logical form of action sentences. In Rescher, N., ed., *The Logic of Decision and Action.* Pittsburgh, PA: Pittsburgh University Press.

Ernst, T. 1989. Chinese Adjuncts and Phrase Structure Theory. *Linguistic Society of America.*

Garman, J. ms. *REAP: A Right-Edge Adjunction Parser.*

Gurney, J., and Klipple, E. 1998. Natural language and virtual reality demonstration. In *Federated Laboratories Display Symposium.*

Gurney, J.; Klipple, E.; and Voss, C. 1996. Talking about What We Think We See: Natural Language Processing for a Real-Time Virtual Environment. In *Proceedings of the IEEE International Joint Symposia on Intelligence and Systems.* Washington, DC: IEEE.

Jackendoff, R. 1997. *The Architecture of the Language Faculty.* Cambridge, MA: MIT.

Klipple, E. 1991. *The Aspectual Nature of Thematic Relations.* Ph.D. Dissertation, MIT, Cambridge, MA.

Koller, D.; Lindstrom, P.; Hodges, W. R. L. F.; Faust, N.; and Turner, G. 1995. Virtual GIS: A Real-Time 3D Geographic Information System. In *Proceedings of Visualization.*

Larson, R., and Segal, G. 1995. *Knowledge of Meaning.* Cambridge, MA: MIT Press.

O'Keefe, J. 1996. The Spatial Prepositions in English, Vector Grammar, and the Cognitive Map Theory. In Bloom, P., ed., *Language and Space.* Cambridge, MA: MIT Press.

Tenny, C. 1987. *Grammaticalizing aspect and affectedness.* Ph.D. Dissertation, MIT, Cambridge, MA.

Tversky, B. 1996. Spatial perspective in descriptions. In Bloom, P., ed., *Language and Space.* Cambridge, MA: MIT Press.