Carsim: A System to Visualize Written Road Accident Reports as Animated 3D Scenes

Richard Johansson **David Williams** **Anders Berglund** **Pierre Nugues**
LUCAS, Department of Computer Science, Lund University
Box 118
SE-221 00 Lund, Sweden
{richard, pierre}@cs.lth.se, {d98dw, d98ab}@efd.lth.se

Abstract

This paper describes a system to create animated 3D scenes of car accidents from reports written in Swedish. The system has been developed using news reports of varying size and complexity. The text-to-scene conversion process consists of two stages. An information extraction module creates a structured representation of the accident and a visual simulator generates and animates the scene.

We first describe the overall structure of the text-to-scene conversion and the structure of the representation. We then explain the information extraction and visualization modules. We show snapshots of the car animation output and we conclude with the results we obtained.

1 Text-to-Scene Conversion

As noted by Michel Denis, language and images are two different representation modes whose cooperation is needed in many forms of cognitive operations. The description of physical events, mathematical theorems, or structures of any kind using language is sometimes difficult to understand. Images and graphics can then help understand ideas or situations and realize their complexity. They have an indisputable capacity to represent and to communicate knowledge and are an effective means to represent and explain things, see (Kosslyn, 1983; Tufte, 1997; Denis, 1991).

Narratives of a car accidents, for instance, often make use of space descriptions, movements, and directions that are sometimes difficult to grasp for most readers. We believe that forming consistent mental images are necessary to understand them properly. However, some people have difficulties in imagining situations and may need visual aids predesigned by professional analysts.

In this paper, we will describe Carsim, a text-to-scene converter that automates the generation of images from texts.

2 Related Work

The conversion of natural language texts into graphics has been investigated in a few projects. NALIG (Adorni et al., 1984; Manzo et al., 1986) is an early example of them that was aimed at recreating static 2D scenes. One of its major goals was to study relationships between space and prepositions. NALIG considered simple phrases in Italian of the type subject, preposition, object that in spite of their simplicity can have ambiguous interpretations. From what is described in the papers, NALIG has not been extended to process sentences and even less to texts.

WordsEye (Coyne and Sproat, 2001) is an impressive system that recreates 3D animated scenes from short descriptions. The number of 3D objects WordsEye uses – 12,000 – gives an idea of its ambition. WordsEye integrates resources such as the Collins' dependency parser and the WordNet lexical database. The narratives cited as examples resemble imaginary fairy tales and WordsEye does not seem to address real world stories.

CogViSys is a last example that started with the idea of generating texts from a sequence of video images. The authors found that it could also be useful to reverse the process and generate synthetic video sequences from texts. The logic engine behind the text-to-scene converter (Arens et al., 2002) is based on the Discourse Representation Theory. The system is limited to the visualization of single vehicle maneuvers at an intersection as the one described in this two-sentence narrative: *A car came from Kriegstrasse. It turned left at the intersection.* The authors give no further details on the text corpus and no precise description of the results.

3 Carsim

Carsim (Egges et al., 2001; Dupuy et al., 2001) is a program that analyzes texts describing car accidents and visualizes them in a 3D environment. It has been developed using real-world texts.

The Carsim architecture is divided into two parts that communicate using a formal representation of

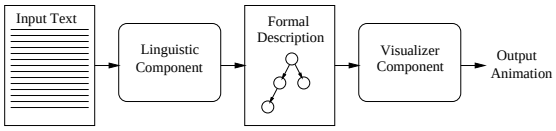

Figure 1: The Carsim architecture.

the accident. Carsim's first part is a linguistic module that extracts information from the report and fills the frame slots. The second part is a virtual scene generator that takes the structured representation as input, creates the visual entities, and animates them (Figure 1).

4 A Corpus of Traffic Accident Descriptions

As development and test sets, we have collected approximately 200 reports of road accidents from various Swedish newspapers. The task of analyzing the news reports is made more complex by their variability in style and length. The size of the texts ranges from a couple of sentences to more than a page. The amount of details is overwhelming in some reports, while in others most of the information is implicit. The complexity of the accidents described ranges from simple accidents with only one vehicle to multiple collisions with several participating vehicles and complex movements.

Although our work has concentrated on the press clippings, we also have access to accident reports from the STRADA database (Swedish TRaffic Accident Data Acquisition) of Vägverket, the Swedish traffic authority. STRADA registers nearly all the accidents that occur in Sweden (Karlberg, 2003). (All the accidents where there are casualties.) After an accident, the victims describe the location and conditions of it in a standardized form collected in hospitals. The corresponding reports are transcribed in a computer-readable format in the STRADA database. This source contains two kinds of reports: the narratives written by the victims of the accident and their transcriptions by traffic experts. The original texts contain spelling mistakes, abbreviations, and grammatical errors. The transcriptions often simplify, interpret the original texts, and contain jargon.

The next text is an excerpt from our development corpus. This report is an example of a press wire describing an accident.

> *En dödsolycka inträffade inatt söder om Vissefjärda på riksväg 28. Det var en bil med två personer i som kom av vägen i en vänsterkurva och körde i hög hastighet in i en gran. Passageraren, som var född -84, dog. Föraren som var 21 år gammal vårdas på sjukhus med svåra skador. Polisen misstänker att bilen de färdades i, en ny Saab, var stulen i Emmaboda och det ska under dagen undersökas.*
>
> Sveriges Radio, November 9, 2002
>
> A fatal accident took place tonight south of Vissefjärda on Road 28. A car carrying two persons departed from the road in a left-hand curve and crashed at a high speed into a spruce. The passenger, who was born in 1984, died. The driver, who was 21 years old, is severely injured and is taken care of in a hospital. The police suspects that the car they were traveling in, a new Saab, was stolen in Emmaboda and will investigate it today.
>
> The text above, our translation.

5 Knowledge Representation

The Carsim language processing module reduces the text content to a formal representation that outlines what happened and enables a conversion to a symbolic scene. It uses information extraction techniques to map a text onto a structure that consists of three main elements:

- A *scene* object, which describes the static parameters of the environment, such as weather, light, and road configuration.
- A list of *road objects*, for example cars, trucks, and trees, and their associated sequences of movements.
- A list of *collisions* between road objects.

The structure of the formalism, which sets the limit of what information can be expressed, was designed with the help of traffic safety experts at the Department of Traffic and Road at Lund University. It contains the information necessary to reproduce and animate the accident entities in our visualization model. We used an iterative process to design it. We started from a first incomplete model (Dupuy et al., 2001) and we manually constructed the representation of about 50 texts until we had reached a sufficient degree of expressivity.

The representation we use is a typical example of frames *à la* Minsky, where the objects in the representation consist of a number of attribute/values slots which are to be filled by the information extraction module. Each object in the representation

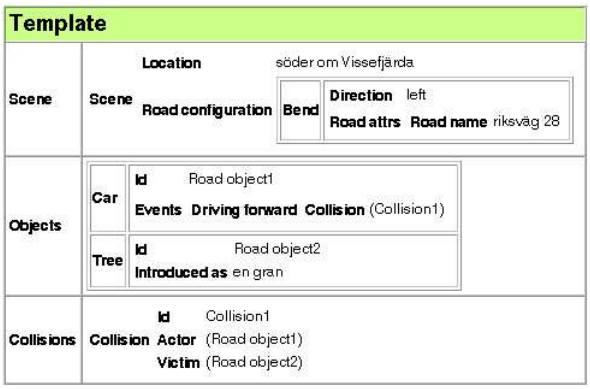

Figure 2: Representation of the accident in the example above.

belongs to a concept in a domain ontology we have developed. The concepts are ordered in an inheritance hierarchy.

Figure 2 shows how Carsim's graphical user interface presents the representation of the accident in the example above. The scene element contains the location of the accident and the configuration of roads, in this case a left-hand bend. The list of road objects contains one car and one tree. The event chain for the car describes the movements: the car leaves the road. Finally, the collision list describes one collision between the car and the tree.

6 The Information Extraction Module

The information extraction subsystem fills the frame slots. Its processing flow consists in analyzing the text linguistically using the word groups obtained from the linguistic modules and a sequence of semantic modules. The information extraction subsystem uses the literal content of certain phrases it finds in the text or infers the environment and the actions.

We use a pipeline of modules in the first stages of the natural language processing chain. The tasks consists of tokenizing, part-of-speech tagging, splitting into sentences, detecting the noun groups, clause boundaries, and domain-specific multiwords. We use the Granska part-of-speech tagger (Carlberger and Kann, 1999) and Ejerhed's algorithm (Ejerhed, 1996) to detect clause boundaries.

6.1 Named Entity Recognition

Carsim uses a domain-specific named entity recognition module, which detects names of persons, places, roads, and car makes (Persson and Danielsson, 2004).

The recognition is based on a small database of 2,500 entries containing person names, city and region names, and car names. It applies a cascade of regular expressions that takes into account the morphology of Swedish proper noun formation and the road nomenclature. The recall/precision performance of the detector is 0.89/0.97.

6.2 Finding the Participants

The system uses the detected noun groups to identify the physical objects, which are involved in the accident. It extracts the headword of each group and associates it to an entity in the ontology. We used parts of the Swedish WordNet as a resource to develop this dictionary (Åke Viberg et al., 2002).

We track the entities along the text with a simple coreference resolution algorithm. It assumes that each definite expression corefers with the last sortally consistent (according to the ontology) entity which was mentioned. Indefinite expressions are assumed to be references to previously unmentioned entities. This is similar to the algorithm mentioned in (Appelt and Israel, 1999). Although this approach is relatively simple, we get reasonable results with it and could use it as a baseline when investigating other approaches.

Figure 3 shows an excerpt from a text with the annotation of the participants as well as their coreferences.

Olyckan inträffade när [*bilen*]$_1$ *som de fem färdades i körde om* [*en annan personbil*]$_2$. *När* [*den*]$_1$ *svängde tillbaka in framför* [*den omkörda bilen*]$_2$ *fick* [*den*]$_1$ *sladd och for med sidan rakt mot fronten på* [*den mötande lastbilen*]$_3$.
The accident took place when [the car]$_1$ where the five people were traveling overtook [another car]$_2$. When [it]$_1$ pulled in front of [the overtaken car]$_2$, [it]$_1$ skidded and hit with its side the front of [the facing truck]$_3$.

Figure 3: A sentence where references to road objects have been marked.

6.3 Resolution of Metonymy

Use of metonymy, such as alternation between the driver and his vehicle, is frequent in the Swedish press clippings. An improper resolution of it introduces errors in the templates and in the visualization. It can create independently moving graphic entities i.e. the vehicle and its driver, that should be represented as one single object, a moving vehicle, or stand together.

We detect the metonymic relations between drivers and their vehicles. We use either cue phrases

like *lastbilschaufförен* ('the truck driver') or the location or instrument semantic roles in phrases like *Mannen som färdades i lastbilen* ('The man who was traveling in the truck'). We then apply constraints on the detected events and directions to exclude wrong candidates. For example, given the phrase *Mannen krockade med en traktor* ('The man collided with a tractor'), we know that the man cannot be the driver of the tractor.

We do not yet handle the metonymic relations between parts of vehicles and the vehicles themselves. They are less frequent in the texts we have examined.

6.4 Marking Up the Events

Events in car accident reports correspond to vehicle motions and collisions. We detect them to be able to visualize and animate the scene actions. To carry out the detection, we created a dictionary of words – nouns and verbs – depicting vehicle activity and maneuvers. We use these words to anchor the event identification as well as the semantic roles of the dependents to determine the event arguments.

6.4.1 Detecting the Semantic Roles

Figure 4 shows a sentence that we translated from our corpus of news texts, where the groups have been marked up and labeled with semantic roles.

[*En personbil*]$_{Actor}$ **körde** [*vid femtiden*]$_{Time}$ [*på torsdagseftermiddagen*]$_{Time}$ [*in i ett radhus*]$_{Victim}$ [*i ett äldreboende*]$_{Loc}$ [*på Alvägen*]$_{Loc}$ [*i Enebyberg*]$_{Loc}$ [*norr om Stockholm*]$_{Loc}$.
[About five]$_{Time}$ [on Thursday afternoon]$_{Time}$, [a car]$_{Actor}$ **crashed** [into a row house]$_{Victim}$ [in an old people's home]$_{Loc}$ [at Alvägen street]$_{Loc}$ [in Enebyberg]$_{Loc}$ [north of Stockholm]$_{Loc}$.

Figure 4: A sentence tagged with semantic roles.

Gildea and Jurafsky (2002) describe an algorithm to label automatically semantic roles in a general context. They use the semantic frames and associated roles defined in FrameNet (Baker et al., 1998) and train their classifier on the FrameNet corpus. They report a performance of 82 percent.

Carsim uses a classification algorithm similar to the one described in this paper. However, as there is no lexical resource such as FrameNet for Swedish and no widely available parser, we adapted it. Our classifier uses a more local strategy as well as a different set of attributes.

The analysis starts from the words in our dictionary for which we designed a specific set of frames and associated roles. The classifier limits the scope of each event to the clause where it appears. It identifies the verb and nouns dependents: noun groups, prepositional groups, and adverbs that it classifies according to semantic roles.

The attributes of the classifier are:

- *Target word*: the keyword denoting the event.
- *Head word*: the head word of the group to be classified.
- *Syntactic class* of head word: noun group, prepositional group, or adverb.
- *Voice* of the target word: active or passive.
- Domain-specific *semantic type*: Dynamic object, static object, human, place, time, cause, or speed.

The classifier chooses the role, which maximizes the estimated probability of a role given the values of the target, head, and semantic type attributes:

$$\hat{P}(r|t, head, sem) = \frac{C(r, t, head, sem)}{C(t, head, sem)}.$$

If a particular combination of target, head, and semantic type is not found in the training set, the classifier uses a back-off strategy, taking the other attributes into account.

We annotated manually a set of 819 examples on which we trained and tested our classifier. We used a random subset of 100 texts as a test set and the rest as a learning set. On the test set, the classifier achieved an accuracy of 90 percent. A classifier based on decision trees built using the ID3 algorithm with gain ratio measure yielded roughly the same performance.

The value of the semantic type attribute is set using domain knowledge. Removing this attribute degraded the performance of the classifier to 80 percent.

6.4.2 Interpreting the Events

When the events have been detected in the text, they can be represented and interpreted in the formal description of the accidents.

We observed that event coreferences are very frequent in longer texts: A same action like a collision is repeated in several places in the text. As for metonymy, duplicated events in the template entails a wrong visualization. We solve it through the unification of as many events as possible, taking metonymy relations into account, and we remove the duplicates.

6.5 Time Processing and Event Ordering

In some texts, the order in which events are mentioned does not correspond to their chronological order. To address this issue and order the events correctly, we developed a module based on the generic TimeML framework (Pustejovsky et al., 2002). We use a machine learning approach to annotate the whole set of events contained in a text and from this set, we extract events used specifically by the Carsim template – the Carsim events.

TimeML has tags for time expressions (*today*), "signals" indicating the polarity (*not*), the modality (*could*), temporal prepositions and connectives such as *for, during, before, after*, events (*crashed, accident*), and tags that indicate relations between entities. Amongst the relations, the TLINKs are the most interesting for our purposes. They express temporal relations between time expressions and events as well as temporal relations between pairs of events.

We developed a comprehensive phrase-structure grammar to detect the time expressions, signals, and TimeML events and to assign values to the entities' attributes. The string *den tolfte maj* ('May 12th') is detected as a time expression with the attribute *value="YYYY-05-12"*. We extended the TimeML attributes to store the events' syntactic features. They include the part-of-speech annotation and verb group structure, i.e. auxiliary + participle, etc.

We first apply the PS rules to detect the time expressions, signals, and events. Let e_1, e_2, e_3, ..., e_n be the events in the order they are mentioned in a text. We then generate TLINKs to relate these events together using a set of decision trees.

We apply three decision trees on sequences of two to four consecutive events $(e_i, e_{i+1}, [, e_{i+2}[, e_{i+3}]])$, with the constraint that there is no time expression between them, as they might change the temporal ordering substantially. The output of each tree is the temporal relation holding between the first and last event of the considered sequence, i.e. respectively: adjacent pairs (e_i, e_{i+1}), pairs separated by one event (e_i, e_{i+2}), and by two events (e_i, e_{i+3}). The possible output values are *simultaneous*, *after*, *before*, *is_included*, *includes*, and *none*. As a result, each event is linked by TLINKs to the three other events immediately after and before it.

We built automatically the decision trees using the ID3 algorithm (Quinlan, 1986). We trained them on a set of hand-annotated examples, which consists of 476 events and 1,162 TLINKs.

As a set of features, the decision trees use certain attributes of the events considered, temporal signals between them, and some other parameters such as the number of tokens separating the pair of events to be linked. The complete list of features with x ranging from 0 to 1, 0 to 2, and 0 to 3 for each tree respectively, and their possible values is:

- Event$_{i+x}$Tense: none, past, present, future, NOT_DETERMINED.
- Event$_{i+x}$Aspect: progressive, perfective, perfective_progressive, none, NOT_DETERMINED.
- Event$_{i+x}$Structure: NOUN, VB_GR_COP_INF, VB_GR_COP_FIN, VB_GR_MOD_INF, VB_GR_MOD_FIN, VB_GR, VB_INF, VB_FIN.
- temporalSignalInbetween: none, before, after, later, when, still, several.
- tokenDistance: 1, 2 to 3, 4 to 6, 7 to 10, greater than 10.
- sentenceDistance: 0, 1, 2, 3, 4, greater than 4.
- punctuationSignDistance: 0, 1, 2, 3, 4, 5, greater than 5.

The process results in an overgeneration of links. The reason for doing this is to have a large set of TLINKs to ensure a fine-grained ordering of the events. As the generated TLINKs can be conflicting, we assign each of them a score, which is derived from the C4.5 metrics (Quinlan, 1993).

We complement the decision trees with heuristics and hints from the event interpreter that events are identical. Heuristics represent common-sense knowledge and are encoded as nine production rules. An example of them is that an event in the present tense is after an event in the past tense. Event identity and heuristics enable to connect events across the time expressions. The TLINKs generated by the rules also have a score that is rule dependent.

When all TLINKs are generated, we resolve temporal loops by removing the TLINK with the lowest score within the loop. Finally, we extract the Carsim events from the whole set of TimeML events and we order them using the relevant TLINKs.

6.6 Detecting the Roads

The configuration of roads is inferred from the information in the detected events. When one of the involved vehicles makes a turn, this indicates that the configuration is probably a crossroads.

Additional information is provided using keyword spotting in the text. Examples of relevant keywords are *korsning* ('crossing'), *'rondell'* ('roundabout') and *kurva* ('bend'), which are very likely indicators of the road configuration if seen in the text.

These methods are very simple, but the cases where they fail are quite rare. During the evaluation described below, we found no text where the road configuration was misclassified.

7 Evaluation of the Information Extraction Module

To evaluate the performance of the information extraction component, we applied it to 50 previously unseen texts, which were collected from newspaper sources on the web. The size of the texts ranged from 31 to 459 words. We calculated precision and recall measures for detection of road objects and for detection of events. A road object was counted as correctly detected if there was a corresponding object in the text, and the type of the object was correct. The same criteria apply to the detection of events, but here we also add the criterion that the actor (and victim, where this applies) must be correct. The performance figures are shown in Tables 1 and 2.

Total number of objects in the texts	105
Number of detected objects	110
Number of correctly detected objects	94
Precision	0.85
Recall	0.90
F-measure ($\beta = 1$)	0.87

Table 1: Statistics for the detection of road objects in the test set.

Total number of events in the texts	92
Number of detected events	91
Number of correctly detected events	71
Precision	0.78
Recall	0.77
F-measure ($\beta = 1$)	0.78

Table 2: Statistics for the detection of events in the test set.

The system was able to extract or infer all relevant information correctly in 23 of the 50 texts. In order to find out the causes of the errors, we investigated what simplifications of the texts needed to be made to make the system produce a correct analysis. The result of this investigation is shown in Table 3.

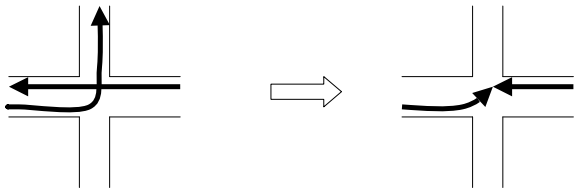

Figure 5: Planning the trajectories.

Object coreference	6
Role labeling	5
Metonymy	5
Clause segmentation	3
Representational expressivity	3
Unknown objects	2
Event detection	2
Unknown event	1
Tagger error	1
PP attachment	1

Table 3: Causes of errors.

8 Scene Synthesis and Visualization

The visualizer reads its input from the formal description. It synthesizes a symbolic 3D scene and animates the vehicles. We designed the graphic elements in the scene with the help of traffic safety experts.

The scene generation algorithm positions the static objects and plans the vehicle motions. It uses rule-based modules to check the consistency of the description and to estimate the 3D start and end coordinates of the vehicles.

The visualizer uses a planner to generate the vehicle trajectories. A first module determines the start and end positions of the vehicles from the initial directions, the configuration of the other objects in the scene, and the chain of events as if they were no accident. Then, a second module alters these trajectories to insert the collisions according to the accident slots in the accident representation (Figure 5).

This two-step procedure can be justified by the descriptions found in most reports. The car drivers generally start the description of their accident as if it were a normal movement, which is subsequently been modified by the abnormal conditions of the accident.

Finally, the temporal module of the planner assigns time intervals to all the segments of the trajec-

tories.

Figure 6 shows two screenshots that the Carsim visualizer produces for the text above. It should be noted that the graphic representation is intended to be iconic in order not to convey any meaning which is not present in the text.

9 Conclusion and Perspectives

We have presented an architecture and a strategy based on information extraction and a symbolic visualization that enable to convert real texts into 3D scenes. We have obtained promising results that validate our approach. They show that the Carsim architecture is applicable to Swedish and other languages. As far as we know, Carsim is the only text-to-scene conversion system working on non-invented narratives.

We are currently improving Carsim and we hope in future work to obtain better results in the resolution of coreferences. We are implementing and adapting algorithms such as the one described in (Soon et al., 2001) to handle this. We also intend to improve the visualizer to handle more complex scenes and animations.

The current aim of the Carsim project is to visualize the content of a text as accurately as possible, with no external knowledge. In the future, we would like to integrate additional knowledge sources in order to make the visualization more realistic and understandable. Geographical and meteorological information systems are good examples of this, which could be helpful to improve the realism. Another topic, which has been prominent in our discussions with traffic safety experts, is how to reconcile different narratives that describe a same accident.

In our work on the information extraction module, we have concentrated on the extraction of data which are relevant for the visual reconstruction of the scene. We believe that the information extraction component could be interesting in itself to extract other relevant data, for example casualty statistics or traffic conditions.

Acknowledgements

We are very grateful to Karin Brundell-Freij, Åse Svensson, and András Várhelyi, traffic safety experts at LTH, for helping us in the design the Carsim template and advising us with the 3D graphic representation.

This work is partly supported by grant number 2002-02380 from the Språkteknologi program of Vinnova, the Swedish Agency of Innovation Systems.

References

Giovanni Adorni, Mauro Di Manzo, and Fausto Giunchiglia. 1984. Natural language driven image generation. In *Proceedings of COLING 84*, pages 495–500, Stanford, California.

Douglas E. Appelt and David Israel. 1999. Introduction to information extraction technology. Tutorial Prepared for IJCAI-99. Artificial Intelligence Center, SRI International.

Michael Arens, Artur Ottlik, and Hans-Hellmut Nagel. 2002. Natural language texts for a cognitive vision system. In Frank van Harmelen, editor, *ECAI2002, Proceedings of the 15th European Conference on Artificial Intelligence*, Lyon, July 21-26.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet Project. In *Proceedings of COLING-ACL'98*, pages 86–90, Montréal, Canada.

Johan Carlberger and Viggo Kann. 1999. Implementing an efficient part-of-speech tagger. *Software Practice and Experience*, 29:815–832.

Bob Coyne and Richard Sproat. 2001. Wordseye: An automatic text-to-scene conversion system. In *Proceedings of the Siggraph Conference*, Los Angeles.

Michel Denis. 1991. Imagery and thinking. In Cesare Cornoldi and Mark A. McDaniel, editors, *Imagery and Cognition*, pages 103–132. Springer Verlag.

Sylvain Dupuy, Arjan Egges, Vincent Legendre, and Pierre Nugues. 2001. Generating a 3D simulation of a car accident from a written description in natural language: The Carsim system. In *Proceedings of The Workshop on Temporal and Spatial Information Processing*, pages 1–8, Toulouse, July 7. Association for Computational Linguistics.

Arjan Egges, Anton Nijholt, and Pierre Nugues. 2001. Generating a 3D simulation of a car accident from a formal description. In Venetia Giagourta and Michael G. Strintzis, editors, *Proceedings of The International Conference on Augmented, Virtual Environments and Three-Dimensional Imaging (ICAV3D)*, pages 220–223, Mykonos, Greece, May 30-June 01.

Eva Ejerhed. 1996. Finite state segmentation of discourse into clauses. In *Proceedings of the 12th European Conference on Artificial Intelligence (ECAI-96) Workshop on Extended Finite State Models of Language*, Budapest, Hungary.

Daniel Gildea and Daniel Jurafsky. 2002. Auto-

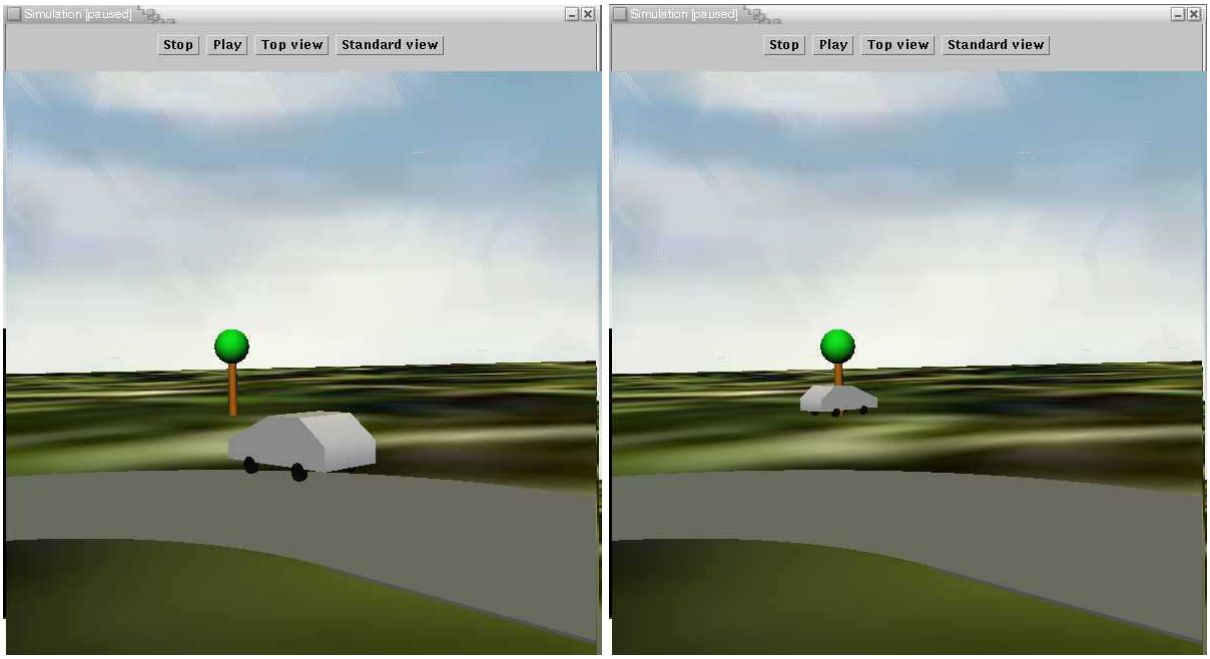

Figure 6: Screenshots from the animation of the text above.

matic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Nils-Olof Karlberg. 2003. Field results from STRADA – a traffic accident data system telling the truth. In *ITS World Congress*, Madrid, Spain, November 16-20.

Stephen Michael Kosslyn. 1983. *Ghosts in the Mind's Machine*. Norton, New York.

Mauro Di Manzo, Giovanni Adorni, and Fausto Giunchiglia. 1986. Reasoning about scene descriptions. *IEEE Proceedings – Special Issue on Natural Language*, 74(7):1013–1025.

Lisa Persson and Magnus Danielsson. 2004. Name extraction in car accident reports for Swedish. Technical report, LTH, Department of Computer science, Lund, January.

James Pustejovsky, Roser Saurí, Andrea Setzer, Rob Gaizauskas, and Bob Ingria. 2002. TimeML Annotation Guidelines. Technical report.

John Ross Quinlan. 1986. Induction of decision trees. *Machine Learning*, 1(1):81–106.

John Ross Quinlan. 1993. *C4.5: Programs for Machine Learning*. Morgan Kauffman.

Åke Viberg, Kerstin Lindmark, Ann Lindvall, and Ingmarie Mellenius. 2002. The Swedish WordNet project. In *Proceedings of Euralex 2002*, pages 407–412, Copenhagen.

Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4):521–544.

Edward R. Tufte. 1997. *Visual Explanations: Images and Quantities, Evidence and Narrative*. Graphic Press.