A Machine Learning Approach to Extract Temporal Information from Texts in Swedish and Generate Animated 3D Scenes

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

Abstract

Carsim is a program that automatically converts narratives into 3D scenes. Carsim considers authentic texts describing road accidents, generally collected from web sites of Swedish newspapers or transcribed from hand-written accounts by victims of accidents. One of the program's key features is that it animates the generated scene to visualize events.

To create a consistent animation, Carsim extracts the participants mentioned in a text and identifies what they do. In this paper, we focus on the extraction of temporal relations between actions. We first describe how we detect time expressions and events. We then present a machine learning technique to order the sequence of events identified in the narratives. We finally report the results we obtained.

1 Extraction of Temporal Information and Scene Visualization

Carsim is a program that generates 3D scenes from narratives describing road accidents (Johansson et al., 2005; Dupuy et al., 2001). It considers authentic texts, generally collected from web sites of Swedish newspapers or transcribed from hand-written accounts by victims of accidents.

One of Carsim's key features is that it animates the generated scene to visualize events described in the narrative. The text below, a newspaper article with its translation into English, illustrates the goals and challenges of it. We bracketed the entities, time expressions, and events and we annotated them with identifiers, denoted respectively o_i, t_j, and e_k:

En {***bussolycka***}$_{e1}$ *i södra Afghanistan* ***krävde***$_{e2}$ {***på torsdagen***}$_{t1}$ {***20 dödsoffer***}$_{o1}$. *Ytterligare* {***39 personer***}$_{o2}$ ***skadades***$_{e3}$ *i* ***olyckan***$_{e4}$.

Bussen$_{o3}$ {***var på väg***}$_{e5}$ *från Kandahar mot huvudstaden Kabul när* ***den***$_{o4}$ *under en* ***omkörning***$_{e6}$ ***körde***$_{e7}$ *av* ***vägbanan***$_{o5}$ *och* ***voltade***$_{e8}$, ***meddelade***$_{e9}$ *general Salim Khan, biträdande polischef i Kandahar.*

TT-AFP & Dagens Nyheter, July 8, 2004

{**20 persons**}$_{o1}$ **died**$_{e2}$ in a {**bus accident**}$_{e1}$ in southern Afghanistan {**on Thursday**}$_{t1}$. In addition, {**39 persons**}$_{o2}$ {**were injured**}$_{e3}$ in the **accident**$_{e4}$.

The **bus**$_{o3}$ {**was on its way**}$_{e5}$ from Kandahar to the capital Kabul when **it**$_{o4}$ {**drove off**}$_{e7}$ the **road**$_{o5}$ while **overtaking**$_{e6}$ and {**flipped over**}$_{e8}$, **said**$_{e9}$ General Salim Khan, assistant head of police in Kandahar.

The text above, our translation.

To create a consistent animation, the program needs to extract and understand who the participants are and what they do. In the case of the accident above, it has to:

1. Detect the involved physical entities $o3$, $o4$, and $o5$.
2. Understand that the pronoun $o4$ refers to $o3$.
3. Detect the events $e6$, $e7$, and $e8$.

4. Link the participants to the events using semantic roles or grammatical functions and infer the unmentioned vehicle that is overtaken.

5. Understand that the order of the events is $e6$-$e7$-$e8$.

6. Detect the time expression $t1$ to anchor temporally the animation.

In this paper, we describe how we address tasks 3, 5, and 6 within the Carsim program, i.e., how we detect, interpret, and order events and how we process time expressions.

2 Previous Work

Research on the representation of time, events, and temporal relations dates back the beginning of logic. It resulted in an impressive number of formulations and models. In a review of contemporary theories and an attempt to unify them, Bennett and Galton (2004) classified the most influential formalisms along three lines. A first approach is to consider events as transitions between states as in STRIPS (Fikes and Nilsson, 1971). A second one is to map events on temporal intervals and to define relations between pairs of intervals. Allen's (1984) 13 temporal relations are a widely accepted example of this. A third approach is to reify events, to quantify them existentially, and to connect them to other objects using predicates based on action verbs and their modifiers (Davidson, 1967). The sentence *John saw Mary in London on Tuesday* is then translated into the logical form: $\exists\epsilon[Saw(\epsilon, j, m) \wedge Place(\epsilon, l) \wedge Time(\epsilon, t)]$.

Description of relations between time, events, and verb tenses has also attracted a considerable interest, especially in English. Modern work on temporal event analysis probably started with Reichenbach (1947), who proposed the distinction between the point of speech, point of reference, and point of event in utterances. This separation allows for a systematic description of tenses and proved to be very powerful.

Many authors proposed general principles to extract automatically temporal relations between events. A basic observation is that the temporal order of events is related to their narrative order. Dowty (1986) investigated it and formulated a *Temporal Discourse Interpretation Principle* to interpret the advance of narrative time in a sequence of sentences. Lascarides and Asher (1993) described a complex logical framework to deal with events in simple past and pluperfect sentences. Hitzeman et al. (1995) proposed a constraint-based approach taking into account tense, aspect, temporal adverbials, and rhetorical structure to analyze a discourse.

Recently, groups have used machine learning techniques to determine temporal relations. They trained automatically classifiers on hand-annotated corpora. Mani et al. (2003) achieved the best results so far by using decision trees to order partially events of successive clauses in English texts. Boguraev and Ando (2005) is another example of it for English and Li et al. (2004) for Chinese.

3 Annotating Texts with Temporal Information

Several schemes have been proposed to annotate temporal information in texts, see Setzer and Gaizauskas (2002), *inter alia*. Many of them were incompatible or incomplete and in an effort to reconcile and unify the field, Ingria and Pustejovsky (2002) introduced the XML-based Time markup language (TimeML).

TimeML is a specification language whose goal is to capture most aspects of temporal relations between events in discourses. It is based on Allen's (1984) relations and a variation of Vendler's (1967) classification of verbs. It defines XML elements to annotate time expressions, events, and "signals". The SIGNAL tag marks sections of text indicating a temporal relation. It includes function words such as *later* and *not*. TimeML also features elements to connect entities using different types of links, most notably temporal links, TLINKs, that describe the temporal relation holding between events or between an event and a time.

4 A System to Convert Narratives of Road Accidents into 3D Scenes

4.1 Carsim

Carsim is a text-to-scene converter. From a narrative, it creates a complete and unambiguous 3D geometric description, which it renders visually. Carsim considers authentic texts describing road accidents, generally collected from web sites of Swedish newspapers or transcribed from handwritten accounts by victims of accidents. One of the program's key features is that it animates the generated scene to visualize events.

The Carsim architecture is divided into two parts that communicate using a frame representation of the text. 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.

4.2 Knowledge Representation in Carsim

The Carsim language processing module reduces the text content to a frame representation – a template – that outlines what happened and enables a conversion to a symbolic scene. It contains:

- *Objects*. They correspond to the physical entities mentioned in the text. They also include abstract symbols that show in the scene. Each object has a *type*, that is selected from a predefined, finite set. An object's semantics is a separate geometric entity, where its shape (and possibly its movement) is determined by its type.

- *Events*. They correspond intuitively to an activity that goes on during a period in time and here to the possible object behaviors. We represent events as entities with a type taken from a predefined set, where an event's semantics will be a proposition paired with a point or interval in time during which the proposition is true.

- *Relations and Quantities*. They describe specific features of objects and events and how they are related to each other. The most obvious examples of such information are *spatial* information about objects and *temporal* information about events. Other meaningful relations and quantities include physical properties such as velocity, color, and shape.

5 Time and Event Processing

We designed and implemented a generic component to extract temporal information from the texts. It sits inside the natural language part of Carsim and proceeds in two steps. The first step uses a pipeline of finite-state machines and phrase-structure rules that identifies time expressions, signals, and events. This step also generates a feature vector for each element it identifies. Using the vectors, the second step determines the temporal relations between the extracted events and orders them in time. The result is a text annotated using the TimeML scheme.

We use a set of decision trees and a machine learning approach to find the relations between events. As input to the second step, the decision trees take sequences of events extracted by the first step and decide the temporal relation, possibly none, between pairs of them. To run the learning algorithm, we manually annotated a small set of texts on which we trained the trees.

5.1 Processing Structure

We use phrase-structure rules and finite state machines to mark up events and time expressions. In addition to the identification of expressions, we often need to interpret them, for instance to compute the absolute time an expression refers to. We therefore augmented the rules with procedural attachments.

We wrote a parser to control the processing flow where the rules, possibly recursive, apply regular expressions, call procedures, and create TimeML entities.

5.2 Detection of Time Expressions

We detect and interpret time expressions with a two-level structure. The first level processes individual tokens using a dictionary and regular expressions. The second level uses the results from the token level to compute the meaning of multiword expressions.

Token-Level Rules. In Swedish, time expressions such as *en tisdagseftermiddag* 'a Tuesday afternoon' use nominal compounds. To decode them, we automatically generate a comprehensive dictionary with mappings from strings onto compound time expressions. We decode other types of expressions such as *2005-01-14* using regular expressions

Multiword-Level Rules. We developed a grammar to interpret the meaning of multiword time expressions. It includes instructions on how to combine the values of individual tokens for expressions such as {*vid lunchtid*}$_{t1}$ {*en tisdageftermiddag*}$_{t2}$ '{at noon}$_{t1}$ {a Tuesday afternoon}$_{t2}$'. The most common case consists in merging the tokens' attributes to form a more specific expression. However, relative time expressions such as *i torsdags* 'last Tuesday' are more complex. Our grammar handles the most frequent ones, mainly those

that need the publishing date for their interpretation.

5.3 Detection of Signals

We detect signals using a lexicon and naïve string matching. We annotate each signal with a sense where the possible values are: negation, before, after, later, when, and continuing. TimeML only defines one attribute for the SIGNAL tag, an identifier, and encodes the sense as an attribute of the LINKs that refer to it. We found it more appropriate to store the sense directly in the SIGNAL element, and so we extended it with a second attribute.

We use the sense information in decision trees as a feature to determine the order of events. Our strategy based on string matching results in a limited overdetection. However, it does not break the rest of the process.

5.4 Detection of Events

We detect the TimeML events using a part-of-speech tagger and phrase-structure rules. We consider that all verbs and verb groups are events. We also included some nouns or compounds, which are directly relevant to Carsim's application domain, such as *bilolycka* 'car accident' or *krock* 'collision'. We detect these nouns through a set of six morphemes.

TimeML annotates events with three features: aspect, tense, and "class", where the class corresponds to the type of the event. The TimeML specifications define seven classes. We kept only the two most frequent ones: states and occurrences.

We determine the features using procedures attached to each grammatical construct we extract. The grammatical features aspect and tense are straightforward and a direct output of the phrase-structure rules. To infer the TimeML class, we use heuristics such as these ones: predicative clauses (copulas) are generally states and verbs in preterit are generally occurrences.

The domain, reports of car accidents, makes this approach viable. The texts describe sequences of real events. They are generally simple, to the point, and void of speculations and hypothetical scenarios. This makes the task of feature identification simpler than it is in more general cases.

In addition to the TimeML features, we extract the grammatical properties of events. Our hypothesis is that specific sequences of grammatical constructs are related to the temporal order of the described events. The grammatical properties consist of the part of speech, noun (NOUN) or verb (VB). Verbs can be finite (FIN) or infinitive (INF). They can be reduced to a single word or part of a group (GR). They can be a copula (COP), a modal (MOD), or a lexical verb. We combine these properties into eight categories that we use in the feature vectors of the decision trees (see ...EventStructure in Sect. 6.2).

6 Event Ordering

TimeML defines three different types of links: subordinate (SLINK), temporal (TLINK), and aspectual (ALINK). Aspectual links connect two event instances, one being aspectual and the other the argument. As its significance was minor in the visualization of car accidents, we set aside this type of link.

Subordinate links generally connect signals to events, for instance to mark polarity by linking a *not* to its main verb. We identify these links simultaneously with the event detection. We augmented the phrase-structure rules to handle subordination cases at the same time they annotate an event. We restricted the cases to modality and polarity and we set aside the other ones.

6.1 Generating Temporal Links

To order the events in time and create the temporal links, we use a set of decision trees. We apply each tree to sequences of events where it decides the order between two of the events in each sequence. If $e_1, ..., e_n$ are the events in the sequence they appear in the text, the trees correspond to the following functions:

$$f_{dt1}(e_i, e_{i+1}) \Rightarrow t_{rel}(e_i, e_{i+1})$$
$$f_{dt2}(e_i, e_{i+1}, e_{i+2}) \Rightarrow t_{rel}(e_i, e_{i+1})$$
$$f_{dt3}(e_i, e_{i+1}, e_{i+2}) \Rightarrow t_{rel}(e_{i+1}, e_{i+2})$$
$$f_{dt4}(e_i, e_{i+1}, e_{i+2}) \Rightarrow t_{rel}(e_i, e_{i+2})$$
$$f_{dt5}(e_i, e_{i+1}, e_{i+2}, e_{i+3}) \Rightarrow t_{rel}(e_i, e_{i+3})$$

The possible output values of the trees are: *simultaneous*, *after*, *before*, *is_included*, *includes*, and *none*. These values correspond to the relations described by Setzer and Gaizauskas (2001).

The first decision tree should capture more general relations between two adjacent events without the need of a context. Decision trees dt_2 and dt_3 extend the context by one event to the left respectively one event to the right. They should capture more specific phenomena. However, they are not always applicable as we never apply a decision

tree when there is a time expression between any of the events involved. In effect, time expressions "reanchor" the narrative temporally, and we noticed that the decision trees performed very poorly across time expressions.

We complemented the decision trees with a small set of domain-independent heuristic rules that encode common-sense knowledge. We assume that events in the present tense occur after events in the past tense and that all mentions of events such as *olycka* 'accident' refer to the same event. In addition, the Carsim event interpreter recognizes some semantically motivated identity relations.

6.2 Feature Vectors

The decision trees use a set of features corresponding to certain attributes of the considered events, temporal signals between them, and some other parameters such as the number of tokens separating the pair of events to be linked. We list below the features of f_{dt1} together with their values. The first event in the pair is denoted by a mainEvent prefix and the second one by relatedEvent:

- mainEventTense: none, past, present, future, NOT_DETERMINED.
- mainEventAspect: progressive, perfective, perfective_progressive, none, NOT_DETERMINED.
- mainEventStructure: NOUN, VB_GR_COP_INF, VB_GR_COP_FIN, VB_GR_MOD_INF, VB_GR_MOD_FIN, VB_GR, VB_INF, VB_FIN, UNKNOWN.
- relatedEventTense: (as mainEventTense)
- relatedEventAspect: (as mainEventAspect)
- relatedEventStructure: (as mainEventStructure)
- temporalSignalInbetween: none, before, after, later, when, continuing, 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 four other decision trees consider more events but use similar features. The values for the ...Distance features are of course greater.

6.3 Temporal Loops

The process described above results in an overgeneration of temporal links. As some of them may be conflicting, a post-processing module reorganizes them and discards the temporal loops.

The initial step of the loop resolution assigns each link with a score. This score is created by the decision trees and is derived from the C4.5 metrics (Quinlan, 1993). It reflects the accuracy of the leaf as well as the overall accuracy of the decision tree in question. The score for links generated from heuristics is rule dependent.

The loop resolution algorithm begins with an empty set of orderings. It adds the partial orderings to the set if their inclusion doesn't introduce a temporal conflict. It first adds the links with the highest scores, and thus, in each temporal loop, the ordering with the lowest score is discarded.

7 Experimental Setup and Evaluation

As far as we know, there is no available time-annotated corpus in Swedish, which makes the evaluation more difficult. As development and test sets, we collected approximately 300 reports of road accidents from various Swedish newspapers. Each report is annotated with its publishing date. Analyzing the reports is complex because of their variability in style and length. Their size 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.

We manually annotated a subset of our corpus consisting of 25 texts, 476 events and 1,162 temporal links. We built the trees automatically from this set using the C4.5 program (Quinlan, 1993). Our training set is relatively small and the number of features we use relatively large for the set size. This can produce a training overfit. However, C4.5, to some extent, makes provision for this and prunes the decision trees.

We evaluated three aspects of the temporal information extraction modules: the detection and interpretation of time expressions, the detection and interpretation of events, and the quality of the final ordering. We report here the detection of events and the final ordering.

Feature	$N_{correct}$	$N_{erroneous}$	Correct
Tense	179	1	99.4%
Aspect	161	19	89.4%
Class	150	30	83.3%

Table 1: Feature detection for 180 events.

7.1 Event Detection

We evaluated the performance of the event detection on a test corpus of 40 previously unseen texts. It should be noted that we used a simplified definition of what an event is, and that the manual annotation and evaluation were both done using the same definition (i.e. all verbs, verb groups, and a small number of nouns are events). The system detected 584 events correctly, overdetected 3, and missed 26. This gives a recall of 95.7%, a precision of 99.4%, and an F-measure of 97.5%.

The feature detection is more interesting and Table 1 shows an evaluation of it. We carried out this evaluation on the first 20 texts of the test corpus.

7.2 Evaluation of Final Ordering

We evaluated the final ordering with the method proposed by Setzer and Gaizauskas (2001). Their scheme is comprehensive and enables to compare the performance of different systems.

Description of the Evaluation Method. Setzer and Gaizauskas carried out an inter-annotator agreement test for temporal relation markup. When evaluating the final ordering of a text, they defined the set E of all the events in the text and the set T of all the time expressions. They computed the set $(E \cup T) \times (E \cup T)$ and they defined the sets $S^{\vdash}$, $I^{\vdash}$, and $B^{\vdash}$ as the transitive closures for the relations *simultaneous*, *includes*, and *before*, respectively.

If $S_k^{\vdash}$ and $S_r^{\vdash}$ represent the set $S^{\vdash}$ for the answer key ("Gold Standard") and system response, respectively, the measures of precision and recall for the *simultaneous* relation are:

$$R = \frac{|S_k^{\vdash} \cap S_r^{\vdash}|}{|S_k^{\vdash}|} \qquad P = \frac{|S_k^{\vdash} \cap S_r^{\vdash}|}{|S_r^{\vdash}|}$$

For an overall measure of recall and precision, Setzer and Gaizauskas proposed the following formulas:

$$R = \frac{|S_k^{\vdash} \cap S_r^{\vdash}| + |B_k^{\vdash} \cap B_r^{\vdash}| + |I_k^{\vdash} \cap I_r^{\vdash}|}{|S_k^{\vdash}| + |B_k^{\vdash}| + |I_k^{\vdash}|}$$

$$P = \frac{|S_k^{\vdash} \cap S_r^{\vdash}| + |B_k^{\vdash} \cap B_r^{\vdash}| + |I_k^{\vdash} \cap I_r^{\vdash}|}{|S_r^{\vdash}| + |B_r^{\vdash}| + |I_r^{\vdash}|}$$

They used the classical definition of the F-measure: the harmonic means of precision and recall. Note that the precision and recall are computed per text, not for all relations in the test set simultaneously.

Results. We evaluated the output of the Carsim system on 10 previously unseen texts against our Gold Standard. As a baseline, we used a simple algorithm that assumes that all events occur in the order they are introduced in the narrative. For comparison, we also did an inter-annotator evaluation on the same texts, where we compared the Gold Standard, annotated by one of us, with the annotation produced by another member in our group.

As our system doesn't support comparisons of time expressions, we evaluated the relations contained in the set $E \times E$. We only counted the reflexive *simultaneous* relation once per tuples (e_x, e_y) and (e_y, e_x) and we didn't count relations (e_x, e_x).

Table 2 shows our results averaged over the 10 texts. As a reference, we also included Setzer and Gaizauskas' averaged results for inter-annotator agreement on temporal relations in six texts. Their results are not directly comparable however as they did the evaluation over the set $(E \cup T) \times (E \cup T)$ for English texts of another type.

Comments. The computation of ratios on the transitive closure makes Setzer and Gaizauskas' evaluation method extremely sensitive. Missing a single link often results in a loss of scores of generated transitive links and thus has a massive impact on the final evaluation figures.

As an example, one of our texts contains six events whose order is $e_4 < e_5 < e_6 < e_1 < e_2 < e_3$. The event module automatically detects the chains $e_4 < e_5 < e_6$ and $e_1 < e_2 < e_3$ correctly, but misses the link $e_6 < e_1$. This gives a recall of $6/15 = 0.40$. When considering evaluations performed using the method above, it is meaningful to have this in mind.

8 Carsim Integration

The visualization module considers a subset of the detected events that it interprets graphically. We

Evaluation	Average n_{words}	Average n_{events}	P_{mean}	R_{mean}	F_{mean}
Gold vs. Baseline	98.5	14.3	49.42	29.23	35.91
Gold vs. Automatic	"	"	54.85	37.72	43.97
Gold vs. Other Annotator	"	"	85.55	58.02	68.01
Setzer and Gaizauskas	312.2	26.7	67.72	40.07	49.13

Table 2: Evaluation results for final ordering averaged per text (with P, R, and F in %).

call this subset the Carsim events. Once the event processing has been done, Carsim extracts these specific events from the full set using a small domain ontology and inserts them into the template. We use the event relations resulting from temporal information extraction module to order them. For all pairs of events in the template, Carsim queries the temporal graph to determine their relation.

Figure 1 shows a part of the template representing the accident described in Section 1. It lists the participants, with the unmentioned vehicle inferred to be a car. It also shows the events and their temporal order. Then, the visualization module synthesizes a 3D scene and animates it. Figure 2 shows four screenshots picturing the events.

Template

Scene	**Scene**	**Location** i södra Afghanistan **Environment** rural
Objects	**Bus**	**Id** RoadObject1 **IntroducedAs** den
	Car	**Id** RoadObject2
Events	**Overtake**	**Id** Event1 **Actor** (RoadObject1) **Victim** (RoadObject2)
	LeaveRoad	**Id** Event2 **Actor** (RoadObject1) **Times** **SameTime** **RelativeTo** (Event1)
	Overturn	**Id** Event3 **Actor** (RoadObject1) **Times** **StrictlyAfter** **RelativeTo** (Event1) **StrictlyAfter** **RelativeTo** (Event2)

Figure 1: Representation of the accident in the example text.

9 Conclusion and Perspectives

We have developed a method for detecting time expressions, events, and for ordering these events temporally. We have integrated it in a text-to-scene converter enabling the animation of generic actions.

The module to detect time expression and interpret events performs significantly better than the baseline technique used in previous versions of Carsim. In addition, it should to be easy to separate it from the Carsim framework and reuse it in other domains.

The central task, the ordering of all events, leaves lots of room for improvement. The accuracy of the decision trees should improve with a larger training set. It would result in a better overall performance. Switching from decision trees to other training methods such as Support Vector Machines or using semantically motivated features, as suggested by Mani (2003), could also be sources of improvements.

More fundamentally, the decision tree method we have presented is not able to take into account long-distance links. Investigation into new strategies to extract such links directly without the computation of a transitive closure would improve recall and, given the evaluation procedure, increase the performance.

References

James F. Allen. 1984. Towards a general theory of action and time. *Artificial Intelligence*, 23(2):123–154.

Brandon Bennett and Antony P. Galton. 2004. A unifying semantics for time and events. *Artificial Intelligence*, 153(1-2):13–48.

Branimir Boguraev and Rie Kubota Ando. 2005. TimeML-compliant text analysis for temporal reasoning. In *IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*, pages 997–1003, Edinburgh, Scotland.

Donald Davidson. 1967. The logical form of action sentences. In N. Rescher, editor, *The Logic of Decision and Action*. University of Pittsburgh Press.

David R. Dowty. 1986. The effects of aspectual class on the temporal structure of discourse: Semantics or pragmatics? *Linguistics and Philosophy*, 9:37–61.

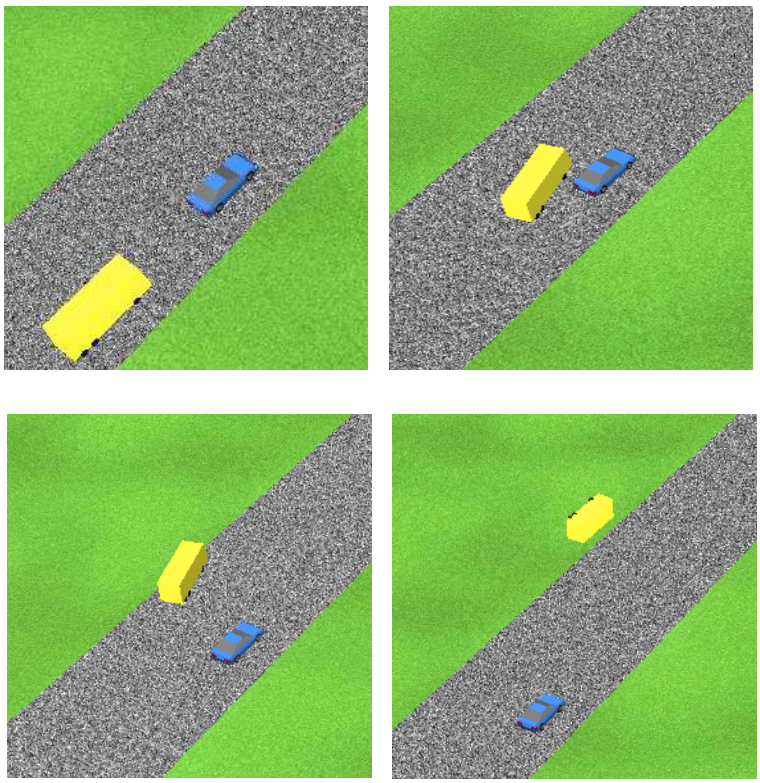

Figure 2: Animation of the scene and event visualization.

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 *ACL 2001, Workshop on Temporal and Spatial Information Processing*, pages 1–8, Toulouse, France.

Richard Fikes and Nils J. Nilsson. 1971. Strips: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189–208.

Janet Hitzeman, Marc Noels Moens, and Clare Grover. 1995. Algorithms for analyzing the temporal structure of discourse. In *Proceedings of the Annual Meeting of the European Chapter of the Association of Computational Linguistics*, pages 253–260, Dublin, Ireland.

Bob Ingria and James Pustejovsky. 2002. Specification for TimeML 1.0.

Richard Johansson, Anders Berglund, Magnus Danielsson, and Pierre Nugues. 2005. Automatic text-to-scene conversion in the traffic accident domain. In *IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*, pages 1073–1078, Edinburgh, Scotland.

Alex Lascarides and Nicholas Asher. 1993. Temporal interpretation, discourse relations, and common sense entailment. *Linguistics & Philosophy*, 16(5):437–493.

Wenjie Li, Kam-Fai Wong, Guihong Cao, and Chunfa Yuan. 2004. Applying machine learning to Chinese temporal relation resolution. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04)*, pages 582–588, Barcelona.

Inderjeet Mani, Barry Schiffman, and Jianping Zhang. 2003. Inferring temporal ordering of events in news. In *Human Language Technology Conference (HLT'03)*, Edmonton, Canada.

Inderjeet Mani. 2003. Recent developments in temporal information extraction. In Nicolas Nicolov and Ruslan Mitkov, editors, *Proceedings of RANLP'03*. John Benjamins.

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

Hans Reichenbach. 1947. *Elements of Symbolic Logic*. Academic Press, New York.

Andrea Setzer and Robert Gaizauskas. 2001. A pilot study on annotating temporal relations in text. In *ACL 2001, Workshop on Temporal and Spatial Information Processing*, pages 73–80, Toulouse, France.

Andrea Setzer and Robert Gaizauskas. 2002. On the importance of annotating temporal event-event relations in text. In *LREC 2002, Workshop on Annotation Standards for Temporal Information in Natural Language*.

Zeno Vendler. 1967. *Linguistics in Philosophy*. Cornell University Press, Ithaca, New York.