Information Extraction to Generate Visual Simulations of Car Accidents from Written Descriptions

Pierre Nugues[1], Sylvain Dupuy[2], and Arjan Egges[3]

[1] Lund University, LTH, Department of Computer science,
Box 118, S-221 00 Lund, Sweden
Pierre.Nugues@cs.lth.se
[2] ISMRA, 6, boulevard du Maréchal Juin, F-14050 Caen, France
Sylvain.Dupuy@free.fr
[3] Université de Genève, MIRALab,
24, rue du Général Dufour, CH-1211 Geneva, Switzerland
egges@miralab.unige.ch

Abstract. This paper describes a system to create animated 3D scenes of car accidents from written reports. The text-to-scene conversion process consists of two stages. An information extraction module creates a tabular description of the accident and a visual simulator generates and animates the scene.
We outline the overall structure of the text-to-scene conversion and the template structure. We then describe the information extraction system: the detection of the static objects and the vehicles, their initial directions, the chain of events, and the collisions. We show snapshots of the car animation output and we conclude with the results we obtained.

1 Text-to-Scene Conversion

Images and graphics have an indisputable capacity to represent and to communicate knowledge, see [1], [2], and [3]. As it has been frequently noted, it is often easier to explain physical phenomena, mathematic theorems, or structures of any kind using a drawing than words. Images can help understand ideas or situations and realize their complexity. They are an effective means to describe and explain things.

Text-to-scene conversion consists in creating a 2D or 3D geometric description from a text and in displaying it. The scene can be static or animated. To be converted, the text must be appropriate in some sense, that is, contains explicit descriptions of objects and events for which we can form mental images. Animated 3D graphics have some advantages for the visualization of information. They can reproduce a real scene more accurately and render a sequence of events. When 3D graphics are coupled with virtual environments, they enable users to navigate in a 3D world and interact with objects.

The conversion of natural language texts into graphics has been investigated in a few projects. NALIG [4], [5], is an early example that was aimed at recreating static 2D scenes. One of the major goals of the project 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 [6] is a recent and 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 interpretation of a narrative is based on an extension of case grammars (semantic frames) and a good deal of inferences about the environment [7]. WordsEye does not address real world stories. The narratives cited as examples resemble imaginary fairy tales. In addition, all the cited texts appear to have been invented by the authors.

CogViSys [8] 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 is based on the Discourse Representation Theory [9]. 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* [10]. The authors give no further details on the text corpus and no precise description of the results.

2 CarSim

CarSim [11], [12], is a text-to-scene converter. It analyzes written reports of car accidents, creates, and animates 3D scenes from the texts. It has been applied to a corpus of 87 car accident reports written in French and provided by the MAIF insurance company. It is worth being noted that the reports are real texts transcribed without modification.

Texts are short narratives written by one of the drivers after the accident. They correspond to relatively simple accidents: There were no casualties and both drivers agreed on what happened. In spite of this, many reports are pretty complex and sometimes difficult to understand, even for a French native speaker.

The CarSim architecture is divided into two parts that communicate using a formal representation of the accident. This formalism adopts a template structure similar to that of information extraction systems [13]. The template has been designed so that it contains the information necessary to reproduce the scene and animate the accident entities. CarSim's first part is a linguistic module that extracts information from the report and fills the template slots. The second part is a virtual scene generator that takes the filled template as an input, creates the visual entities, and animates them (Figure 1).

We believe that the conversion of a text to a scene can help understand its information content as it can make it more concrete to a user. Although we don't claim that a sequence of images can replace a text, we are sure that it

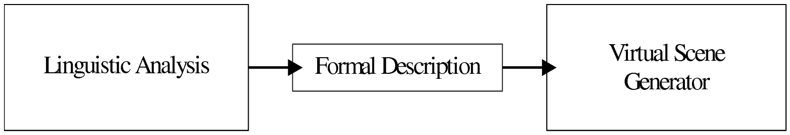

Fig. 1. The CarSim architecture.

can complement it. And automatic conversion techniques can make this process faster and easier.

3 Examples of Reports

The next two texts are examples of reports contained in the MAIF corpus:

> *Véhicule B venant de ma gauche, je me trouve dans le carrefour, à faible vitesse environ 40 km/h, quand le véhicule B, percute mon véhicule, et me refuse la priorité à droite. Le premier choc atteint mon aile arrière gauche, sous le choc, et à cause de la chaussé glissante, mon véhicule dérape, et percute la protection métallique d'un arbre, d'où un second choc frontal.*
>
> Text A4, MAIF corpus.
>
> I was driving on a crossroads with a slow speed, approximately 40 km/h. Vehicle B arrived from my left, ignored the priority from the right and collided with my vehicle. The first collision hit my rear fender on the left side and because of the slippery road. I lost control of my vehicle and hit the metallic protection of a tree, a second frontal collision.
>
> Text A4, our translation.
>
> *Je roulais sur la partie droite de la chaussée quand un véhicule arrivant en face dans le virage a été complètement déporté. Serrant à droite au maximum, je n'ai pu éviter la voiture qui arrivait à grande vitesse.*
>
> Text A8, MAIF corpus.
>
> I was driving on the right-hand side of the road when a vehicle coming in front of me in the bend skidded completely. Moving to the right of the lane as far as I could, I couldn't avoid the car that was coming very fast.
>
> Text A8, our translation.

Both texts are good examples of the contents of an accident report. The text A4 depicts a collision between two cars and then between one of the cars and a tree. This narrative is explicit but the descriptions are not always as straightforward. Many reports contain understatements, negations, cosmetic descriptions, or more simply typographical errors. It often prevents them from being understood without guessing the state of mind of the driver. The report A8 doesn't mention explicitly the collision for example.

4 Knowledge Representation

CarSim uses templates to represent the text. This means that it reduces the text content to a tabular structure that outlines what happened and enables a conversion to a symbolic scene. The structure consists of three elements:

– The static objects element identifies the type of the road and the objects that play a role in the collision (traffic lights, trees, obstacles, etc.).
– The dynamic objects element describes the vehicles, cars or trucks, involved in the accident, and
– The collisions element describes the impact sequence between the dynamic objects. A collision can also involve a static and a dynamic object.

Table 1 shows the XML template representation of the A8 report. The `staticObjects` element contains one meaningful static object: the road configuration, a curve. The road arrangement is selected amongst four possible cases: *straight road*, *left* and *right turning curves*, and *crossroads*.

The `dynamicObjects` element describes two vehicles: the narrator's and the other vehicle. For each `vehicle` – or actor – detected in the text, there will be a corresponding slot in the template, which is filled with the name of the vehicle, its initial direction, and a chain of events corresponding to its motions.

The insurance reports usually identify vehicles as A and B. When this information is not available, the `id` slot is filled with "narrator" and the other vehicle is identified as vehicle B or by the name of the driver when it is mentioned. The vehicles are positioned relatively to each other before they start moving since most of the time the text gives no information on their absolute geographic location. The initial directions are given in the `initDirection` attribute. They are selected amongst *west*, *east*, *north*, and *south*. The chain of events corresponds to the sequence of vehicle motions. The current possible actions are: *driving forward*, *turn left*, *turn right*, *stop*, *change lane left*, *change lane right*, and *overtake*.

Finally, the `collisions` element reflects the sequence of collisions. In Table 1, one single collision occurs between the narrator and vehicle B. The parties are named `actor` and `victim` although this is not an interpretation of liabilities. When possible, the template identifies the impacted parts. The `coords` field is an optional and plausible location of the collision where (0, 0) is the center of the scene.

We assessed the expressiveness of this formalism to represent the accidents described in the corpus. We extracted templates manually and we found that approximately 60 percent of the accidents could be reproduced using these settings.

The remaining 40 percent corresponds to more complex road configurations, or more complex accident scenarios.

5 Scene Synthesis

The visualizer reads its input from the template description. It synthesizes a symbolic 3D scene and animates the vehicles [11]. The scene generation algorithm positions the static objects and plans the vehicle motions. It uses rule-based modules to check the consistency of the template 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 template (Figure 2). 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 trajectories.

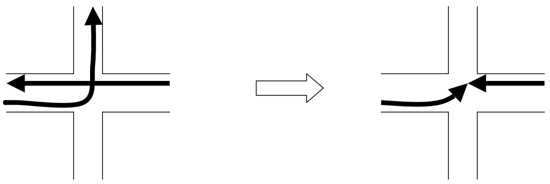
Fig. 2. Planning the trajectories.

6 Information Extraction

The information extraction (IE) subsystem fills the template slots. Its processing flow consists in parsing the text using a partial parser and framing the accident into a prototype situation using the word groups obtained from the parser and a sequence of semantic modules. The IE subsystem uses the literal content of certain phrases it finds in the text or infers the environment and the actions.

The first module of the IE subsystem is a partial parser. It uses DCG rules to detect noun groups, verb groups, and prepositional groups together with some multiword patterns. Vehicles names identifying actors as *vehicle A*, *vehicle B*, car makes or models are examples of these multiwords.

Table 1. The template representing text A8.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE accident SYSTEM "accident.dtd">
<accident>
  <staticObjects>
    <road kind="turn_left"/>
  </staticObjects>
  <dynamicObjects>
    <vehicle id="enonciateur" kind="car" initDirection="east">
      <eventChain>
        <event kind="driving_forward"/>
      </eventChain>
    </vehicle>
    <vehicle id="vehicule" kind="car" initDirection="south">
      <eventChain>
        <event kind="driving_forward"/>
      </eventChain>
    </vehicle>
  </dynamicObjects>
  <collisions>
    <collision>
      <actor id="enonciateur" side="unknown"/>
      <victim id="vehicule" side="unknown"/>
      <coords x="-1.5" y="1.5"/>
    </collision>
  </collisions>
</accident>
```

6.1 Extracting the Static Objects

The static objects correspond to the road configuration, road signs, and road obstacles. They are extracted using a lexical ontology of the domain, a pattern matcher, and the result of the parse.

The most meaningful static object is the road configuration. The detection module tries to match a set of words to the text corresponding either to crossroads, with words such as *croisement*, *carrefour*, or *intersection* or to right or left curves with words like *tournant* or *virage*, etc. The lexical ontology has been partly acquired using the Integral dictionary [14]. The detection module also uses simple inference rules with the action described by the driver. Thus, when no bend is mentioned in the report, an occurrence of the verb *tourner* “turn” creates a crossroads object.

When they are mentioned in the report, other objects such as road signs, traffic lights, or road obstacles participate most of the time in the accident. They are detected using the same method.

6.2 Detecting Actors

Actors in CarSim are the vehicles or persons playing a role in the accident. In most reports, the accident description is presented from the viewpoint of the narrator who describes what happened, using first person pronouns. In some other reports, the description is impersonal and uses sentences like *Vehicle A collided with Vehicle B* as:

> *Circulant Avenue du Sud, le véhicule B a brusquement coupée la voie au véhicule A.* (Constat B58)
>
> Driving on Avenue du Sud, vehicle B cut off the path of vehicle A abruptly. (Report B58)

The actors of an accident are associated to noun groups. The actor detection is then basically a co-reference resolution. One of the assumptions of the algorithm is that the probable number of actors is most frequently two and sometimes one or three.

By default, a first actor named the narrator is created. Then, the actor detection creates as many actors as it can find different head nouns in the noun groups. These nouns must be semantically compatible with vehicles or drivers: proper nouns, vehicle identifiers such as *vehicle A* or *vehicle B*, vehicle models such as *Fiat Punto*, *Renault Clio*, nouns depicting vehicles, person names, etc. In case of failure, the actor detector searches static objects that can be involved in a collision. In this latter case, one single actor is created. And finally, if none of the two previous cases is verified, the actor detector creates a second default vehicle.

The co-reference resolution associates all the first person pronouns and possessive determiners to the narrator and the third person pronouns and possessive determiners to the other actors using a recency criterion. If no first person pronouns are found, the narrator is deleted.

6.3 The Initial Directions

Initial directions are detected using the prepositions and motion verbs contained in the report and inference rules [15]. When they start moving, vehicles are either on a same axis or have orthogonal directions. As Carsim does not use absolute references, the narrator is set to come from the east. Hence, cars coming from the right will come from the north in the symbolic orientation. Cars coming from the left will be located on the south. Facing cars will come from the west. Cars behind or in front moving in the same direction will also come from the east.

Some reports indicate geographical locations and directions such as

> *Fort trafic 17 h 15 Bd Sébastopol...* (Constat A3)
>
> Heavy traffic 5.15 pm Sébastopol blvd... (Report A3)

When possible, names such as that of *boulevard de Sébastopol* in Paris are extracted to fill a `startSign` sub-element in the `vehicle` element of the template. They are visualized as arrow-shaped direction signposts.

To detect the initial directions, the algorithm searches couples of verb groups and directions. Verbs correspond to motions such as *venir de*, *surgir de*, *arriver sur*, ("come from", "appear suddenly") etc. Directions are typically adjuncts of locations: *sur la droite*, *en sens inverse*, ("to the right" or "from the right", "in the opposite direction"), etc. The sentence subject is then unified with one of the actors detected previously. This enables to initialize the motion of the actors.

6.4 Detecting Collisions

The collision detection is carried out using a subset of collision verbs restricted to those we observed in the accident reports: *percuter*, *heurter*, *casser*, *accrocher*, etc. ("smash", "hit", "break", "bump"). We classified them according to their meaning and whether they entail a second collision as with *pousser*, *projeter*, ("push", "cast"). The system also considers negations such as in *je n'ai pas pu éviter...* (I could not avoid...)

The module identifies the subject and the object of each collision verb. It checks whether the verb group is in the passive or active form and detects subject-verb, verb-object, verb-agent, and verb-target patterns. The verb-target pattern corresponds to certain verb-preposition constructions in French such as *cogner sur quelque chose* ("hit something") where the prepositional group would be an object in another language as in English.

The system creates as many collisions as there are collision verbs except if two collisions involve exactly the same actors. The system removes these duplicate collisions then. In effect, some people tend to repeat themselves and mention twice a same collision in their report.

6.5 Detecting the Chain of Events

Each vehicle carries out a succession of one or more different actions before the collision. These actions are encoded as a list of `event` elements contained in the `eventChain` element of the dynamic objects. The list of actions that the visualizer can animate are `driving_forward`, `turn_left`, `turn_right`, `overtake`, `change_lane_right`, `change_lane_left`, and `stop`.

In the reports, these actions correspond to motion verbs such as *démarrer*, *rouler*, *circuler*, *avancer*, *tourner*, *virer*, etc. ("start", "drive", "move", "turn", etc.). The event detection module extracts these verbs and creates a new event for each verb occurrence. It assigns it to the event chain of the verb subject provided that it corresponds to a valid actor. Some verbs entail multiple events such as for example *redémarrer* ("start driving again") that indicates that the driver stopped before. Then, the event is transcribed into a sequence of two actions: `stop` and `driving_forward`.

The extraction mechanism appends the events to the chain in the order they are written in the text. This doesn't always correspond to the real chronology. However in almost all the texts, the order of events in the text corresponds to their real order.

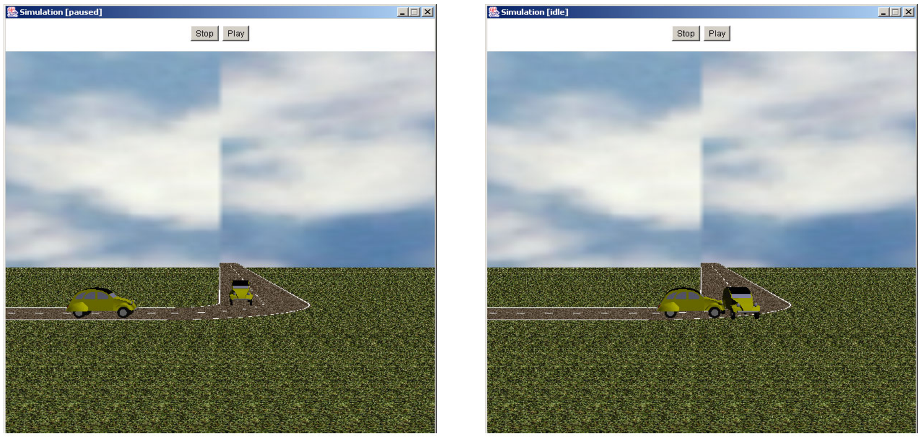

Fig. 3. Visualization of text A8.

7 Results and Perspectives

We evaluated the CarSim system on all the reports of the MAIF corpus. The method we adopted is simpler than the classical assessment of information extraction systems that use precision and recall. We assessed the whole processing chain by comparing the visual output that CarSim produces from a text with the understanding and the mental images a person can have of it. Figure 3 shows two snapshots of the scene syntheses for the report A8. As we can view, the evaluation entails a part of subjectivity.

Using the algorithms we described, we have been able to obtain a plausible simulation of 29 reports that represent a third of the corpus. This should also be related to the upper-limit of the template modeling and visualization system of CarSim that is currently of 60 percent. Although these results are far from being perfect, they demonstrate that a strategy based on information extraction and a symbolic visualization enables to convert real texts into 3D scenes.

The wrong simulations have different causes:

– The imprecise spatial descriptions. Some reports are difficult to interpret even for human readers.
– The system has no model of some 3D static objects mentioned in the reports. They include the roundabouts, pavements, motorway accesses, fences, etc.
– Some vehicle actions are not modeled by CarSim such as backing off.

Many aspects of Carsim could be improved in the language processing part as well as in the visualizer. A second version of CarSim is under development and one of the authors is currently verifying the validity of the approach with other types of accident summaries. He uses two corpora: a set of road accident summaries available from the National Transportation Safety Board and accident descriptions from Swedish newspapers. The NTSB is a US government agency that investigates aviation accidents and significant road accidents. Results already obtained show that the CarSim architecture is applicable to other languages than French and to more complex accidents. As far as we know, CarSim is the only text-to-scene conversion system working on non-invented narratives.

Acknowledgments. This work is partially supported by the Vinnova Språkteknologi program under grant 2002-02380.

References

1. Kosslyn, S.M.: Ghosts in the Mind's Machine. Norton, New York (1983)
2. Tufte, E.R.: Visual Explanations: Images and Quantities, Evidence and Narrative. Graphic Press, Cheshire, Connecticut (1997)
3. Denis, M.: Imagery and thinking. In Cornoldi, C., McDaniel, M.A., eds.: Imagery and Cognition. Springer Verlag, New York (1991) 103–132
4. Adorni, G., Di Manzo, M., Giunchiglia, F.: Natural language driven image generation. In: Proceedings of COLING 84, Stanford, California (1984) 495–500
5. Di Manzo, M., Adorni, G., Giunchiglia, F.: Reasoning about scene descriptions. IEEE Proceedings – Special Issue on Natural Language **74** (1986) 1013–1025
6. Coyne, B., Sproat, R.: Wordseye: An automatic text-to-scene conversion system. In: Proceedings of the Siggraph Conference, Los Angeles (2001)
7. Sproat, R.: Inferring the environment in a text-to-scene conversion system. In: Proceedings of the K-CAP'01, Victoria, BC (2001)
8. Nagel, H.H.: Toward a cognitive vision system. Technical report, Universität Karlsruhe (TH), http://kogs.iaks.uni-karlsruhe.de/CogViSys (2001)
9. Kamp, H., Reyle, U.: From Discourse to Logic. Kluwer, Dordrecht (1993)
10. Arens, M., Ottlik, A., Nagel, H.H.: Natural language texts for a cognitive vision system. In van Harmelen, F., ed.: ECAI2002, Proceedings of the 15th European Conference on Artificial Intelligence, Lyon (2002) 455–459
11. Egges, A., Nijholt, A., Nugues, P.: Generating a 3D simulation of a car accident from a formal description. In Giagourta, V., Strintzis, M.G., eds.: Proceedings of The International Conference on Augmented, Virtual Environments and Three-Dimensional Imaging (ICAV3D), Mykonos, Greece (2001) 220–223
12. Dupuy, S., Egges, A., Legendre, V., Nugues, P.: 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, Toulouse, Association for Computational Linguistics (2001) 1–8
13. Hobbs, J.R., Appelt, D., Bear, J., Israel, D., Kameyama, M., Stickel, M., Tyson, M.: Fastus: A cascaded finite-state transducer for extracting information from natural-language text. In Roche, E., Schabes, Y., eds.: Finite-State Language Processing. MIT Press, Cambrigde, Massachusetts (1997) 383–406
14. Dutoit, D., Nugues, P.: A lexical network and an algorithm to find words from definitions. In van Harmelen, F., ed.: ECAI2002, Proceedings of the 15th European Conference on Artificial Intelligence, Lyon (2002) 450–454
15. Le Gloannec, S., Aubeuf, P., Métais, C.: Extraction d'informations pour la simulation en 3D de textes de constats d'accidents. Technical report, ISMRA Caen (2001) Rapport de projet.