A METHOD FOR AUTOMATICALLY CREATING 3D ANIMATED SCENES FROM ANNOTATED FICTION TEXT

Kevin Glass. *Rhodes University, Grahamstown, South Africa*
kevinrobertglass@gmail.com

Shaun Bangay. *Rhodes University, Grahamstown, South Africa.*
s.bangay@ru.ac.zal

ABSTRACT

This paper describes a strategy for automatically converting fiction text into 3D animations. It assumes the existence of fiction text annotated with avatar, object, setting, transition and relation annotations, and presents a transformation process that converts annotated text into quantified constraint systems, the solutions to which are used in the population of 3D environments. Constraint solutions are valid over temporal intervals, ensuring that consistent dynamic behaviour is produced. A substantial level of automation is achieved, while providing opportunities for creative manual intervention in animation process. The process is demonstrated using annotated examples drawn from popular fiction text that are converted into animation sequences, confirming that the desired results can be achieved with only high-level human direction.

KEYWORDS

Text to scene conversion, story visualization, computer graphics, constraint optimization.

1. INTRODUCTION

Popular fiction books describe rich visual environments that contain characters, objects, and behaviour. The conversion of fiction text to a unified multi-modal animation is a difficult and subjective process which requires substantial human intervention at every step of the process (as evidenced by the lengthy list of credits at the end of any film).

Some steps in the process stand to benefit from automation, freeing the human component to concentrate on the creative aspects. Technologies already exist that reduce the effort required for the creation of animated graphics, including key-frame animation, inverse-kinematics, motion capture, and fluid and cloth simulations. Additional tasks stand to benefit from automation, including the analysis and interpretation of language and the population of corresponding virtual environments, while still allowing directorial intervention consistent with the constraints in the scene.

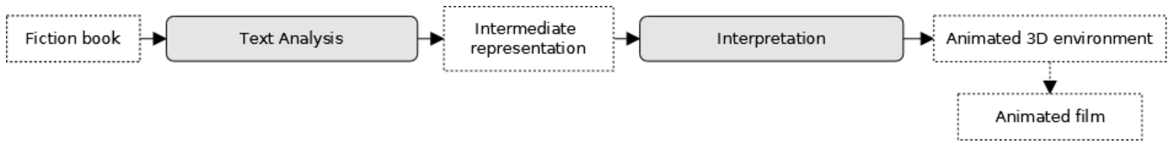


Figure 1. Overview of the tasks that can benefit from automation.

An overview of the possible tasks that can benefit from automation is provided in Figure 1. They include transforming the natural language text into an intermediate representation, and thereafter transforming this representation into a populated 3D environment.

The first task of converting fiction text into an intermediate representation is detailed by Glass (2008). The intermediate representation consists of the original text that is *annotated* in the categories summarized in Table 1. Each category is chosen and defined in a manner such that automatic information extraction processes may be used for the creation of annotations over fiction text (Glass and Bangay, 2005, 2006). The format of the annotations supports additional human manipulation to further refine the animation.

Table 1. Summary of annotation categories.

Category	Fields	Description
Avatar	-	Explicit identification of a character
Object	-	Explicit identification of an object
Setting	-	Explicit identification of the environment
Relation	*type, subject*	Explicit description of a spatial relation
Transition	*type, subject, object*	Explicit indication of entry to or exit from the scene

Existing work demonstrates that the use of a hierarchical rule-based learning system is capable of creating these annotations in a manner that matches a particular human's manually-created annotations with accuracies ranging between 51.4% and 90.4% (Glass, 2008). Experience has shown that annotations cannot be defined as categorically "incorrect", because the idea of correctness differs from one human to the next. The remainder of this paper assumes the existence of the annotations summarized in Table 1, but does not make assumptions about the completeness or correctness (with regards to any particular person) of the annotations.

They had it on the top of a hill, in a sloping field that looked down into a sunny **<setting>valley</setting>**. **<avatar>Anne</avatar>** didn't very much like a big brown **<object>cow</object>** who <**transition type='INSIDE' subject='cow'>came</transition>** up **<relation type='near' subject='cow' object='her'>close<relation>** and stared at her, but it **<transition type='OUTSIDE' subject='it'>went</transition>** away when **<avatar>Daddy</avatar>** told it to.

Figure 2. Example of annotated fiction text, sourced from *The Famous Five: Five on a Treasure Island* by Enid Blyton.

An example of annotated fiction text is presented in Figure 2. Fiction writing tends assume much implicit knowledge, and therefore any details not explicitly mentioned are not automatically annotated. As a result, there is an opportunity for manual intervention in the process during which human creativity may be used to insert further annotations.

In this paper, we present a process for transforming annotated fiction text into 3D animations. This is done by formulating time-quantified constraints with regards to objects in the scenes, the solutions to which are used to specify layout, appearance and motion within a 3D environment.

After a survey of previous approaches to text-to-scene conversion (Section 2), we describe the steps used in our approach. The first step in generating 3D animations from annotated fiction text is the automatic translation of annotations into abstract constraints (Section 3), from which a set of quantified constraints is derived and solved using interval methods (Section 4). The scene specification is used to instantiate a 3D virtual environment, which is rendered into a final animation (Section 5). We present our experimental methodology in Section 6, and assess examples of automatically produced animations in Section 7.

2. RELATED WORK

The task of converting natural language text into graphical formats is generally differentiated according to the type of textual input and the type of output (Glass, 2008). Types of textual input include *language similar* input, which resembles natural language, but has a formal structure (Clay and Wilhelms, 1996); *commands*, that explicitly target entities in the environment and describe modifications to their properties using natural language (Badler et al., 2000); *descriptive instructions*, that use short natural language statements to describe how a scene should appear (Coyne and Sproat, 2001); and *narrative*, that is running text written in story-form (Johansson et al., 2005). Types of graphical output include sequences of corresponding photographs (Zhu et al., 2007); static images rendered from 3D scenes (Coyne and Sproat, 2001; Zeng et al., 2003, 2005; Seversky and Yin, 2006); and animated 3D environments (Lu and Zhang, 2002; Johansson et al., 2005; Ma, 2006).

Fiction text presents language in narrative form and involves a temporal component. Three notable related systems exist that handle input text in a similar form, namely SWAN (Lu and Zhang, 2002), CARSIM (Johansson et al., 2005), and CONFUCIUS (Ma, 2006).

The SWAN system converts stories expressed using natural language to 3D animations, requiring that input be expressed in simplified Chinese language (Lu and Zhang, 2002). The CARSIM system converts road accident reports into corresponding 3D animations, where input is restricted to the car accident domain (Johansson et al., 2005). CARSIM and SWAN are notable in the fact that the exposition of the natural language input is in *story* form (although neither use fiction books as immediate sources of input). The CONFUCIUS system is capable of converting single sentences, possibly sourced from fiction books into short multi-modal animations (Ma, 2006).

Both SWAN and CONFUCIUS make use of detailed knowledge-bases that provide the ability for reasoning with regards to the automatic interpretation of text and the layout of a scene. A result of this is the ability to generate highly detailed animations, but require some restriction in the form of input (limited to what the knowledge base supports) as well as substantial effort in the creation of the knowledge-base. We present an alternative method in which the degree of world knowledge required for the automatic conversion process is kept to a minimum, rather leaving this aspect to human creativity.

Our work, which we term *fiction-to-animation*, is the first to transform text sourced from popular fiction into corresponding 3D animations without prior language simplification. This research examines a new perspective of the text-to-graphics conversion problem: transforming *annotated fiction* text into animations, as opposed to conventional techniques that convert knowledge-rich intermediate *semantic* representations (for example, predicate argument structures or semantic frames (Coyne and Sproat, 2001)) into animations.

The techniques described in this article cover the final phase of the larger fiction-to-animation conversion process, first described by Glass (2008). This research is the first to handle the quantity and complexity of textual input presented by unrestricted fiction text, while producing multi-modal, multi-scene representations of the text.

3. ANNOTATIONS TO STRUCTURED SCENE DESCRIPTIONS

Fiction text describes a sequence of scenes, each of which contains a number of characters and objects that perform behaviours. The text describing each of these elements is indicated by the annotations. In spite of the existence of these annotations, there is still insufficient structure and information to automatically construct corresponding 3D environments.

This section describes the knowledge-poor techniques used to convert annotated text into structured scene descriptions that are both human readable (for manual intervention where needed) but sufficiently structured for subsequent processing. High-level scene descriptions include the following elements: a list of *scenes* to be instantiated; the contents of each scene (specified using *entity descriptors*); and the behaviour of entities in each scene (specified using *abstract constraints*).

The process of deriving scene descriptions is illustrated in Figure 3, and contains a number of points at which a human can provide his or her own interpretation of the text. The different classes of annotation are used to progressively develop scene descriptions. *Setting* annotations are used to segment the input text into a number of scenes. *Avatar* and *object* annotations are used to compile a list of entities that occur in each scene. *Relation* and *transition* annotations are then used to generate a list of time quantified abstract constraints that summarize the layout and behaviour in the scene.

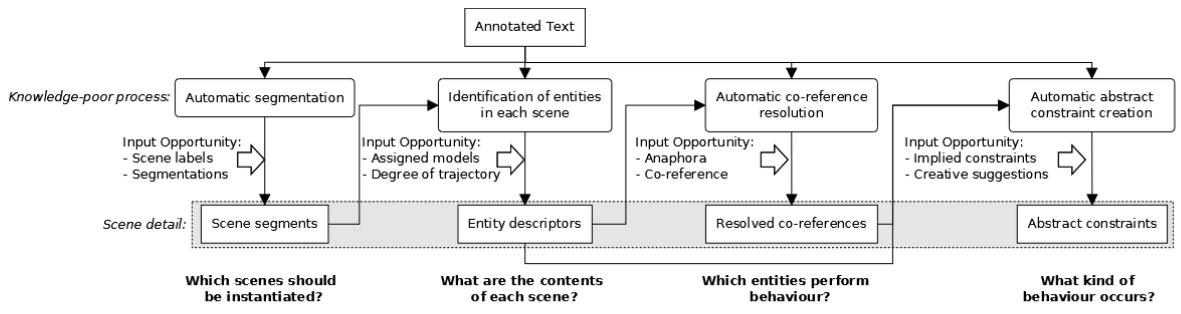


Figure 3. Illustration of the abstract constraint creation process.

We define a *scene* to correspond to a single physical location. The setting annotation (indicated in Table 1) is used to segment the text into contiguous portions that describe a single physical location. The setting is not always explicitly stated in fiction text, however, resulting in the absence of an automatically created setting annotation. Human interpretation is encouraged at this point to indicate non-explicit scene changes.

An *entity descriptor* is created for every unique avatar and object annotation that occurs in a portion of scene text. Entity descriptors assign a geometric model (see Section 5), sourced from a library of pre-built models, to each avatar or object entity. The descriptor also associates an initially unconstrained trajectory to the entity, expressed in terms of a number of uninitialized variables that represent the motion of the model through 3D space. By default, objects are static and are fixed at an initially undetermined position, while avatars are assigned trajectories that permit motion.

Before behaviours can be assigned to entities, a pointer must be created between words in the text that refer to a particular entity and the descriptor for that entity. All instances of such *co-reference* must be resolved before relation and transition annotations can be converted to constraints. Instances of personal pronominal anaphora (such as “he”) are resolved by maintaining state containing the last explicitly mentioned male and female avatars and matching the gender of the anaphora with the corresponding element in the state vector. The pronoun “it” is resolved to the last-mentioned object. Co-references in the form of explicit names in the text are resolved by matching the corresponding word with a list of keywords in the entity descriptor.

The last step of the process is the creation of *abstract constraints* from transition and relation annotations, expressed in the manner illustrated in Figure 4. Each abstract constraint is phrased in terms of the *type* field of the annotation from which it is derived, as well as the descriptors of the involved entities. For transitions the *type* may be either INSIDE or OUTSIDE, while relations may be of type NEAR, IN_FRONT_OF, BEHIND, TO_LEFT_OF, TO_RIGHT_OF, ON_TOP_OF and BELOW.

```
CONSTRAINT 1:      CONSTRAINT 2:      CONSTRAINT 3:      CONSTRAINT 4:
 Subject: MAN       Subject: MAN       Subject: MAN       Subject: MAN
 Relation: OUTSIDE  Relation: INSIDE   Relation: NEAR     Relation: NEAR
 Object: ROOM       Object: ROOM       Object: TABLE      Object: CHAIR
 Start-time: 0      Start-time: 5      Start-time: 9      Start-time: 14
 End-time: 5        End-time: 30       End-time: 14       End-time: 30
```

Figure 4. Example set of abstract constraints.

Explicit constraints are derived directly from transition and relation annotations, while *implicit* constraints are created using an automated heuristic approach to enforce world constraints. If the first constraint for an entity is of type INSIDE at a time greater than zero, then an implicit OUTSIDE constraint is inserted from time 0 to the start-time of the first constraint, to ensure that the entity is initially outside the scene. Any entities that are not constrained by a transition constraint are automatically assigned an implicit INSIDE constraint for the duration of the scene to ensure that they appear in the scene. Implicit NO_COLLIDE constraints are added for every pair of entities in the scene, lasting for the duration of the scene to ensure that entities do not interpenetrate at any point.

The time interval associated with each constraint in Figure 4 specifies the period over which the constraint should hold with respect to the duration of the animation. Timing information is derived using an audio narration of the text acquired from a speech synthesizer. Each language unit in the book takes a finite interval of time to be vocalized, from which a time-line is derived for the presentation of the book. Since each scene is constructed of a sequence of tokens, the total time taken for the sounding of the corresponding audio provides a value for the scene duration, as well as a time-value for each token. Starting times for abstract constraints are derived from transition and relation annotations corresponding to the time-value of the annotated token (illustrated in Figure 5). Ending times are based on the duration of the scene by default, but for transition constraints they are set to the starting time of a subsequent transition constraint applied to the same entity. Relation constraints are also terminated by subsequent OUTSIDE transition constraints.

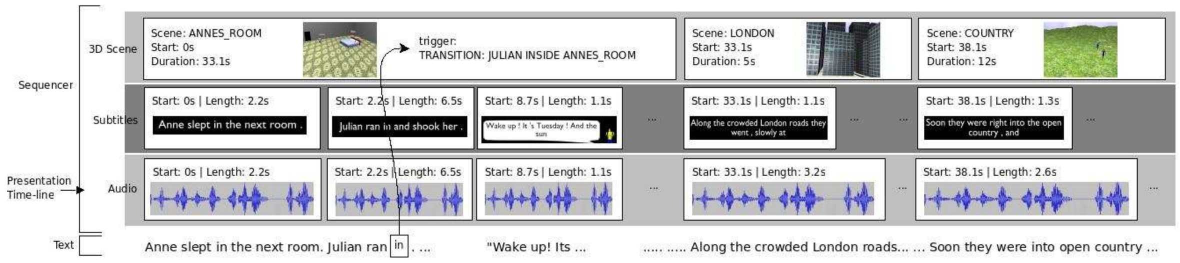

Sequencer
3D Scene
Scene: ANNES_ROOM
Start: 0s
Duration: 33.1s
trigger:
TRANSITION: JULIAN INSIDE ANNES_ROOM
Scene: LONDON
Start: 33.1s
Duration: 5s
Scene: COUNTRY
Start: 38.1s
Duration: 12s
Subtitles
Start: 0s | Length: 2.2s
Anne slept in the next room .
Start: 2.2s | Length: 6.5s
Julian ran in and shook her .
Start: 8.7s | Length: 1.1s
Wake up ! It 's Tuesday ! And the sun
Start: 33.1s | Length: 1.1s
Along the crowded London roads they went , slowly at
Start: 38.1s | Length: 1.3s
Soon they were right into the open country , and
Presentation Time-line
Audio
Start: 0s | Length: 2.2s
Start: 2.2s | Length: 6.5s
Start: 8.7s | Length: 1.1s
Start: 33.1s | Length: 3.2s
Start: 38.1s | Length: 2.6s
Text
Anne slept in the next room. Julian ran in . ...
"Wake up! Its ...
.... Along the crowded London roads... ... Soon they were into open country ...

Figure 5. Illustration of the relationship between the time-line derived from speech-synthesized text and annotations.

Once the abstract constraint list has been created, it may be modified manually to insert constraints regarding the scene that are not explicitly stated in the text. The human readable format used for expressing abstract constraints supports direct modification.

The automatically generated scene descriptions are sufficiently formal to permit the automatic creation of corresponding virtual worlds. However, before a virtual environment can be instantiated for each scene, precise values that quantify the behaviour in each virtual environment must be calculated.

4. QUANTIFIED CONSTRAINT SYSTEMS

A set of abstract constraints such as those presented in Figure 4 is converted into a system of expressions that constrain the trajectories of the entities involved in the constraints (Glass et al., 2007). Each transition and relation type is associated with a particular quantified constraint expression.

For example, MAN NEAR TABLE over interval [9,14] would be specified as a relation between the locations of the two objects' bounding spheres (of radius r_{MAN} and r_{TABLE} respectively) over a specific time interval, expressed in terms of the Euclidean distance, as follows:

$$\|R^{MAN}(t) - R^{TABLE}(t)\|^2 < (r_{MAN} - r_{TABLE} + \alpha)^2 \forall t \in [9,14]$$

where $R^{MAN}(t)$ and $R^{TABLE}(t)$ are the trajectories of MAN and TABLE as functions of time respectively. α is the minimum distance considered to verify the term "near". We derived a similar *noCollide* expression in a similar manner, and also a *directionRelation* expression for handling relation types such as IN_FRONT_OF and TO_LEFT_OF (Glass, 2008).

Each abstract constraint presented in Figure 4 is converted into a corresponding expression, and the conjunction of these constraints for a single scene forms a system of constraints.

Trajectories are expressed as curves of degree *n*. The higher the degree the more difficult the constraint solving process becomes. To counter this, the trajectory of each model is segmented into chains of lower degree curves (Christie et al., 2002). At present we find that a satisfactory trade-off between performance and quality is achieved with the use of chains of first degree curves.

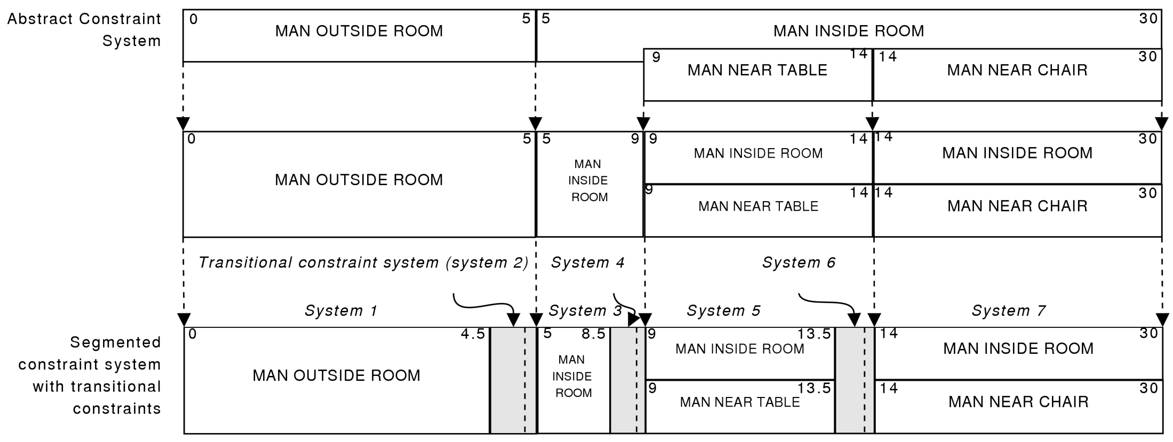
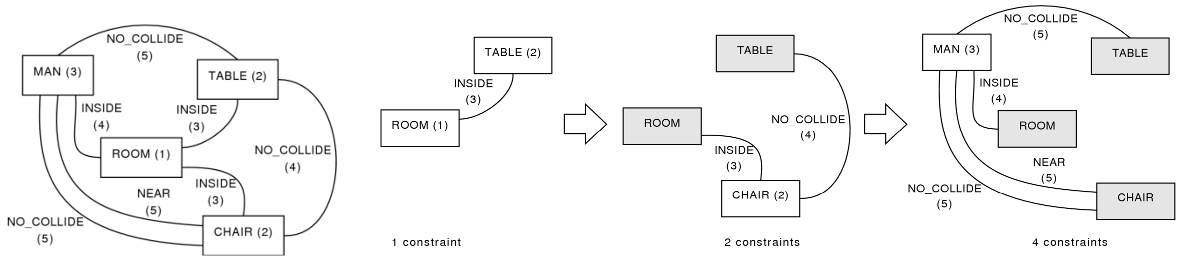


Figure 6. Illustration of constraint system segmentation

Figure 6 illustrates graphically the abstract constraint system of Figure 4 and indicates how the constraints are segmented into a chain of 7 constraint systems. Initially a unique system is created for each contiguous time interval by dividing the duration of the scene into intervals. For each interval, a single set of constraints is applicable over the full duration of that interval. A *transitional* system is inserted between adjacent intervals allowing a period during which both sets of constraints must be satisfied so that model locations blend smoothly from one interval to the next. To facilitate efficient constraint solving, the ending locations for each interval are used as starting locations for solving the following constraint system.

Constraint solving establishes the trajectory for each entity, which is then traced in a 3D space, defining the motion of the model through a scene. Constraints are phrased over a universally quantified time interval, and solutions or approximations are calculated using an interval-based constraint optimization algorithm (Glass, 2008) based on the universally quantified constraint solver developed by Benhamou et al. (2004). This algorithm guarantees that a solution is valid for every point during the time interval, even when the constraint systems are non-linear. A benefit of the optimization-based approach is that it provides approximate solutions even for constraints resulting from conflicting annotations (meaning that annotations do not have to be completely “correct”), and also produces useful intermediate solutions if limited processing time is available.



(a) Constraint system illustrated as weighted graph.

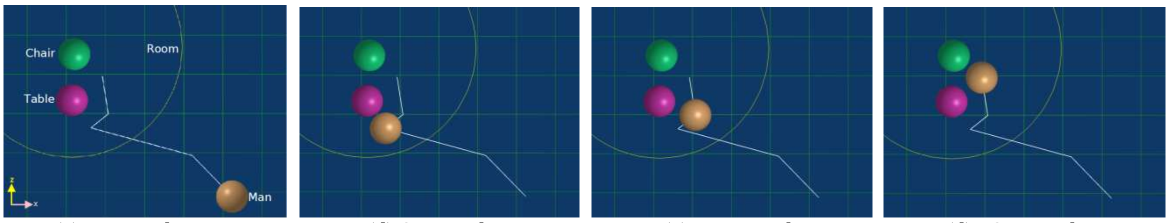
(b) Three incremental constraint systems.

Figure 7. Illustration of incremental constraint solving.

Each constraint system resulting from the segmentation process is solved in an incremental fashion. All constraints involving static objects are solved first, followed by those involving dynamic objects. This is achieved by representing the scene as a weighted graph where nodes represent entities, and edges represent constraints. Edges are given increasing weights according to whether the adjacent entities are scenes (1), objects (2) or avatars (3).

The order in which constraints are solved is determined by the weights of the corresponding edges. Figure 7 illustrates this process, where in each step the highlighted entities represent trajectories for which solutions have already been found.

We find that trajectory chaining in conjunction with incremental system solving results in constraint systems that can be rapidly solved using the quantified interval-based constraint optimization algorithm. The results of this step are well defined trajectories for all entities. A visualization of the trajectory produced for the set of constraints in Figure 4 is presented in Figure 8.



(c) 5 seconds (d) 8 seconds (c) 12 seconds (d) 18 seconds

Figure 8. Visualization of a complex trajectory as a sequence of low degree curves.

Validation of the intermediate processes described in this section and Section 3, especially regarding the interval-based constraint optimizer, is available in other work (Glass, 2008).

5. AUTOMATIC ANIMATION ASSEMBLY

Once trajectories have been quantified for a set of scenes, we automatically create a graphical representation of the annotated text. The 3D environment is created and populated and the final animation is sequenced.

Models are automatically selected for each entity from a library containing pre-textured, size- and orientation-standardized models. Humanoid models are chosen for avatars, while object models are located automatically by matching the annotation with keywords associated with each model in a library. The annotated token associated with an object is used as search term when querying the model library. If a model's keywords match the token then the model is chosen to represent the object. If no matching keyword exists, then all the synonyms of the term returned using WordNet (Fellbaum, 1998) are tried. If no match exists in this case then the immediate hypernym of the term is used as a search term. The process of generalization continues until the term can no longer be generalized, and a default placeholder object is selected (a cube).

Entity models are chosen before the constraint quantification process (described in Section 4), and the dimensions of the models are used to calculate the radius for use in each constraint expression. In addition, the trajectory type for the entity is also derived at this point, where only avatars are assigned trajectories that permit movement ($degree \geq 1$). Both the chosen model and the degree of the trajectory are specified in the entity descriptor.

We make use of the setting annotation to automatically create background geometry using procedural methods. Three types of setting are defined, namely *terrain*, *room*, and *city*. Using WordNet (Fellbaum, 1998), a setting annotation is automatically converted to one of these categories. If a setting annotation is a hyponym of the term "geological formation" or "geological area" it is classified as a *terrain*. Any term with the hypernym "urban area" is classified as a *city*, while any term with the hypernym "room" is classified as a *room*. Terrain geometry is created automatically using a method adapted from the terrain synthesis technique developed by Belhadj (2007), which allows the creation of realistic looking terrain around pre-defined points. Using this approach terrain can be specified to "support" all entities in the scene over their entire trajectory. Procedural city generation is implemented based on work by Parish and Muller (2001), but at this stage this method only produces grid-iron road patterns, with textured cubes as buildings. Rooms are generated through the creation of geometry defining four walls and a floor, textured, and containing openings where entities enter or exit the scene.

Animation assembly is performed in the Blender modeling package (available at `http://www.blender.org/`) which is an open source 3D modeling and animation tool that provides a

Python scripting interface as well as a video sequencing editor. The scripting interface is used to convert trajectories and object descriptors produced by the previous stages into 3D environments.

Blender allows for the creation of multiple 3D environments, conveniently called *scenes*. As such, every scene identified in the text is created as a separate 3D environment in Blender, from which segments may be rendered and placed into the video sequencer along with subtitles and audio. Subtitles, audio narrations and foleys are also generated automatically (Glass et al., 2007).

The model library is an archive of existing Blender files containing object geometry, materials, armatures and motion capture data which can be linked into new scenes. Models in the library are modified by the scripting environment to create customized characters for each avatar (for example, changing clothing and hair colouring).

Positioning and motion within the 3D environment is controlled using Blender's interpolation (IPO) curve structure, which defines a model's translation in a scene in terms of an independent curve for each dimension. Each sampled location point from the entity's trajectory defines a point on the corresponding IPO curve. The orientation of each model is modified concurrently to always face the direction of motion. Depending on the velocity of the model at each point, the appropriate pose (for example stand, walk or run) for the models is selected automatically.

6. EXPERIMENTAL SETUP

The measurement of the success of an automatically generated virtual world is a subjective process because of different human interpretations of fiction text. The results produced by the scene creation processes are visual in nature, and it is unclear which features in a visual scene should be measured for quantitative evaluation.

The problem of evaluating automatically generated scenes is encountered in other research in the text-to-graphics domain. Most related research performs subjective evaluations through the use of visual examples (Coyne and Sproat, 2001; Lu and Zhang, 2002; Zeng et al., 2003; Joshi et al., 2004). In these cases, a small set of example images produced from the original text is provided, leaving evaluation to the discretion of the reader.

The only other evaluation method we encounter in related text-to-graphics research is user evaluation, where a group of human subjects is asked to rate the visual content produced by the automatic system (Johansson et al., 2005; Ma, 2006). However, Johansson et al. (2005) acknowledge that such studies do not provide a complete reflection on the capabilities of a system, and results vary greatly from one human to the next. We believe that this is because such studies do not objectively evaluate the system. Rather, they evaluate both the human as well as the system. Subjectivity is involved in comprehending natural language and visual images. The processes we describe are designed to improve in quality given additional time and patience of the human. These factors introduce errors into a user-evaluation that are not necessarily caused by the automated process, and are therefore not measurable.

We perform our evaluations using a subjective (but quantitative) evaluation of the manual interventions needed to ensure consistency in automatically generated content. We measure consistency as the percentage of correctly generated content in relation to the total generated content. In addition, we also follow the methods used by WORDSEYE (Coyne and Sproat, 2001), SWAN (Lu and Zhang, 2002), and other text-to-graphics systems (Zeng et al., 2003; Joshi et al., 2004) in using visual examples as evidence of the correctness of the automated content. We critically evaluate each visual example. However, we maintain that the true level of correctness varies from human to human, or according to the adage, is "in the eye of the beholder".

The primary source of experimental error for this experimentation is the complexity of the English language and its understanding. The annotations produced for the corpus of test data cannot be guaranteed to be consistent or correct, and this potentially produces extraneous or surprising artifacts in the automatically generated scenes. The subjectivity of image and film comprehension is also a source of experimental error. Similar to language comprehension, comprehension of the visual and audio modalities is influenced by individual human experience. As such, manual modifications and critical evaluations provided for each result are independent to the evaluator, and do not necessarily correlate with another human's opinion.

We draw six extracts from three different books, written by three different authors. Each extract is annotated manually with setting, avatar, object, transition and relation annotations.

We deliberately limit the produced scene quality to avoid embellishments that are customized to any particular example. Also, additional scene detail is described in the text but which is not handled by our set of annotation categories. These cases are ignored, pending the creation of the relevant annotations in future work.

7. RESULTS

For each of the six annotated extracts, we record the amount of automatically generated content and compare this with the amount of content that is modified by a human. The type and degree of manual intervention with respect to each step of the scene description process (illustrated in Figure 3) is listed in Table 2.

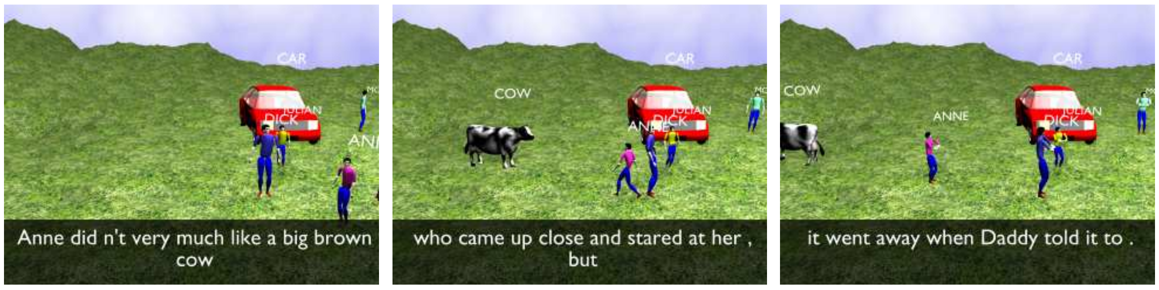
Table 2. Degree of consistency for the automatic process for creating scene descriptions over six extracts.

Extract	Scene segmentation			Model descriptors			Co-reference			Abstract constraints		
	Total	Modified	Consistency	Total	Modified	Consistency	Total	Modified	Consistency	Total	Modified	Consistency
1	1	0	100%	12	1	91.67%	12	2	83.33%	68	1	98.52%
2	1	0	100%	8	2	75.0%	52	15	71.5%	41	1	97.56%
3	1	1	0%	6	0	100%	40	8	80.0%	45	0	100%
4	6	2	66.67%	10	0	100%	23	0	100%	41	9	78.0%
5	1	0	100%	2	0	100%	11	0	100%	4	0	100%
6	2	0	100%	8	0	100%	28	1	96.43%	44	0	100%
Summary	12	3	83.33%	46	3	93.47%	166	26	84.33%	243	11	95.47%

Scene segmentation is consistent in the majority of cases, where the only modifications required are the insertion of scene segments where none are explicitly stated in the text. The creation of model descriptors is also consistent in the majority of cases, and only 3 of the 46 automatically created descriptors are modified manually. Co-reference is automatically performed to a high level of accuracy (84.33%), where the primary problems encountered are when resolving words such as “boy” to the correct avatar. Only a small number of the automatically generated abstract constraints require manual modification. The greatest number of modifications is made for extract 6. In this case, constraints are added manually to cater for details not explicitly described in the fiction text.

The degree to which automatically produced content matches the text is justified through the use of visual examples. We present snapshots from automatically generated animation sequences using the process described in this paper. The corresponding films are available for download from the following site: `http://www.cs.ru.ac.za/research/g05g1909/`.

Figure 9 presents snapshots based on the annotated text example of Figure 2. Examples of the changes that have been applied in each stage of the process include: re-annotating the cow as an avatar rather than a static object; manual resolving of the pronoun “it” to the cow object; and removing extraneous models in the scenes created in Blender. In comparison to the amount of effort that would be otherwise involved in creating this scenes manually, these changes are accomplished quickly and easily.



Anne did n't very much like a big brown cow

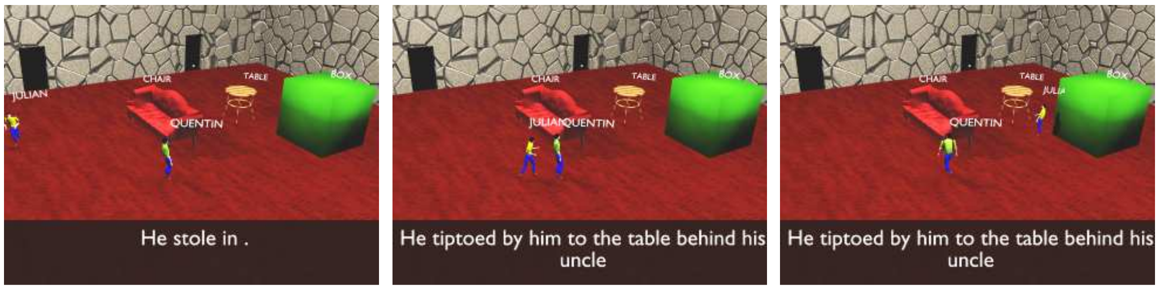
who came up close and stared at her , but

it went away when Daddy told it to .

Figure 9. Cow scene from *The Famous Five: Five on a Treasure Island* by Enid Blyton (extract 1) - 3 models manually deleted from the 3D scene.

The cow enters and exits the scene at the correct moments according to the concurrent subtitles, indicating the successful conversion of the transition annotations to visual behaviour. The setting is interpreted correctly in providing a background suitable for the description “valley”, and the appropriate geometric models appear in the virtual environment.

Figure 10 presents an example from the same book, in which a *room* setting is created, and which requires the correct layout of objects in the scene. Notice that the table is behind the chair as specified, and how the model representing Julian moves to the correct locations according to the annotations. This example demonstrates that complex behaviour is visualized in the form of correct movement through multiple way-points.



He stole in .

He tiptoed by him to the table behind his uncle

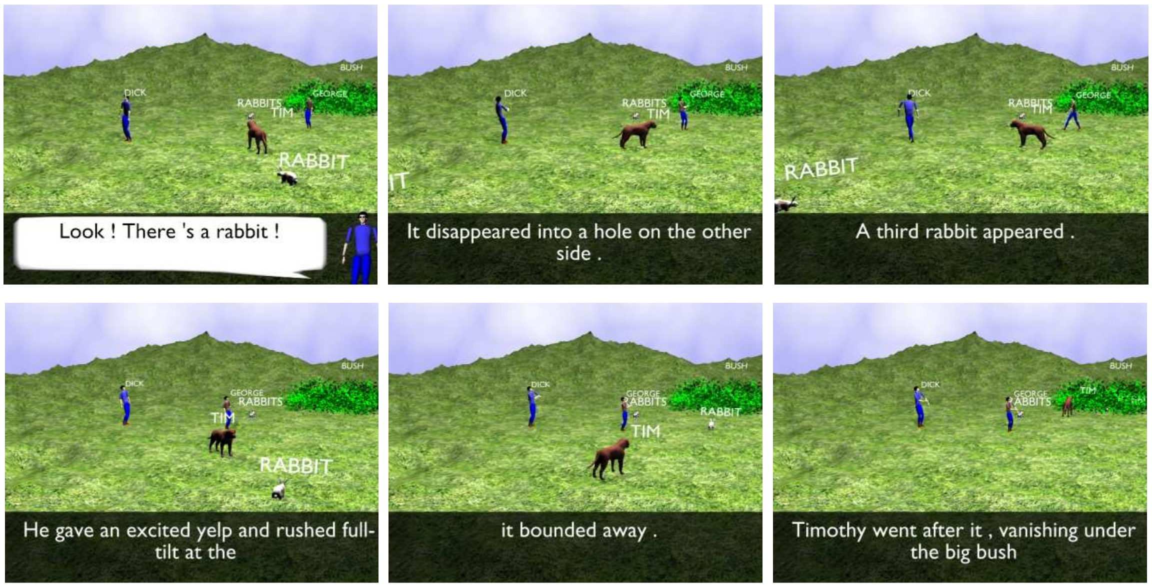
He tiptoed by him to the table behind his uncle

He stole **<transition type='INSIDE' subject='He'>in</transition>**. His **<avatar>uncle</avatar>** still snored. He tiptoed by him **<relation type='NEAR' subject='he' object='table'>to</relation>** the **<object>table</object>** **<relation type='BEHIND' subject='table' object='chair'>behind</relation>** his uncle's **<object>chair</object>**.

Figure 10. Study scene from *The Famous Five: Five on a Treasure Island* by Enid Blyton (extract 3) - 1 model manually deleted from the 3D scene.

We identify problems with the results in Figure 10, such as the fact that the box is not placed on the table. This fact is never explicitly stated in the text however, and the created scene is correct according to the annotations. One flaw is the initial location of Quentin, who should be in the chair throughout the scene. This constraint is only implemented from the point at which the annotation is encountered in the text, resulting in the movement of Quentin to his chair only midway through the scene.

The extract presented in Figure 11 contains a number of transition annotations describing the behaviour of the Rabbit entity. A number of different behaviours are also specified by Relation annotations: Timothy rushes towards the Rabbit; the Rabbit moves under a bush; and Timothy follows the rabbit to the bush. The combination of these different behaviour types results in complex motion in the virtual environment. The sequence of snapshots illustrates the corresponding motion of Timothy towards the rabbit and the subsequent motion of the rabbit towards the bush. The images visualize Timothy following the rabbit into the bush.

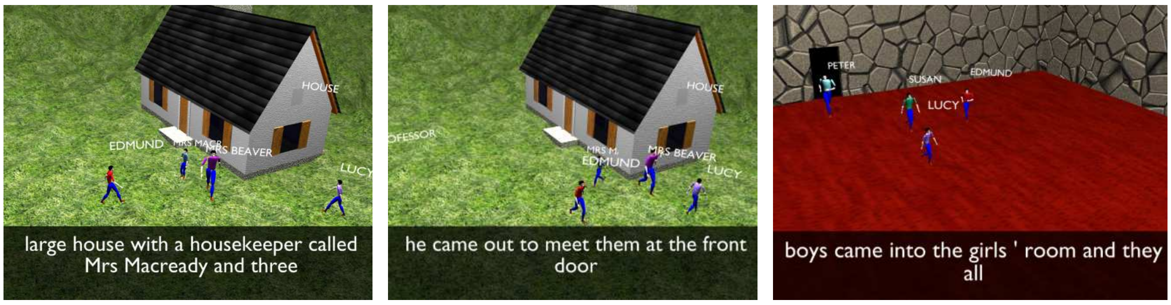


DICK
RABBITS
TIM
GEORGE
BUSH
RABBIT
Look ! There 's a rabbit !
It disappeared into a hole on the other side .
A third rabbit appeared .
He gave an excited yelp and rushed full-tilt at the
it bounded away .
Timothy went after it , vanishing under the big bush

"Look! There's a rabbit!" cried **<avatar>Dick</avatar>**, as a big sandy **<object>rabbit</object>** lolloped slowly across the **<setting>yard</setting>**. It **<transition type="OUTSIDE" subject="It">disappeared</transition>** into a hole on the other side. …
A third rabbit **<transition type="INSIDE" subject="rabbit">appeared</transition>**. It was a small one with absurdly big ears. …
But this was too much for **<avatar>Timothy</avatar>.** ... . He gave an excited **<foley>yelp</foley>** and rushed full-tilt **<relation type="NEAR" subject="He" object="rabbit">at</relation>** the surprised rabbit. ...
It disappeared **<relation type="UNDER" subject="It" object="bush">under</relation>** a gorse **<object>bush</object>** near the children**. <avatar>Timothy</avatar>** went after it, vanishing **<relation type="UNDER" subject="Timothy" object="bush">under</relation>** the big **<object>bush</object>** too. ...

Figure 11. Rabbit scene (1) from *The Famous Five: Five on a Treasure Island* by Enid Blyton (extract 2) - 0 modifications on the final 3D scene.

The results presented in Figure 9, Figure 10, and Figure 11 demonstrates that the behaviour specified by transition and relation annotations is successfully converted into visual behaviour that corresponds to the text.



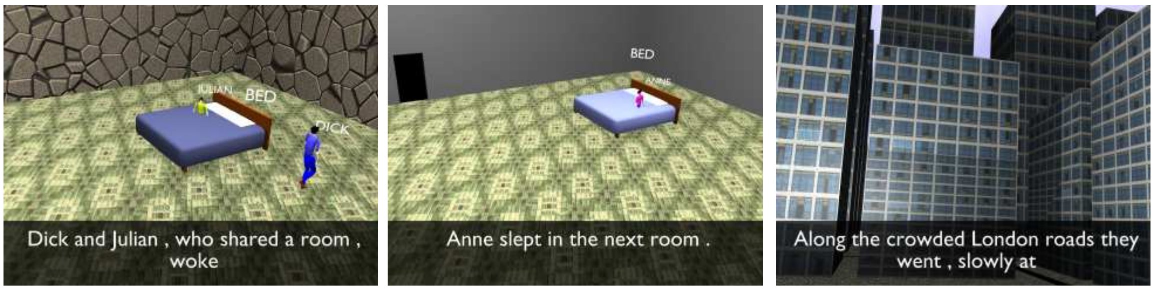
HOUSE
EDMUND
MRS MACR
MRS BEAVER
LUCY
PROFESSOR
PETER
SUSAN
large house with a housekeeper called Mrs Macready and three
he came out to meet them at the front door
boys came into the girls ' room and they all

They were sent to the **<object>house</object>** of an old **<avatar>Professor</avatar>** who lived in the heart of the **<setting>country</setting>** … He had no wife and he lived in a very large house with a housekeeper called **<avatar>Mrs Macready</avatar>** and three servants. …
As soon as they had said good night to the **<avatar>Professor</avatar>** and gone upstairs on the first night, the boys **<transition type="INSIDE" subject="boys">came</transition>** into the girls' **<setting>room</setting>** and they all talked it over.

Figure 12. House sequence from *Narnia: The Lion, the Witch and the Wardrobe* by C.S. Lewis (extract 6) - 2 objects deleted from the 3D scene representing the room.

The extract depicted in Figure 12 is drawn from a different book than the previous extracts, and begins by describing the location of the Professor's house and the initial meeting of the characters outside it. The setting then changes to inside the house, in one of the rooms. It is evident from the images that scene changes are correctly handled according to the setting annotation, demonstrating that our system is capable of producing multi-scene animations from fiction text.

An additional extract is presented in Figure 13 that demonstrates the automatic creation of an animation containing multiple scenes, including two distinct *rooms* and a *city*. This extract is an example in which human creativity is applied in the form of an additional abstract constraint specifying ANNE INSIDE BED, since this fact is not explicitly stated in the text. This demonstrates that world knowledge may be easily applied to an automatically created scene through manual intervention, removing the requirement for a complex knowledge-base.


JULIAN
BED
DICK
Dick and Julian , who shared a room , woke
BED
ANNE
Anne slept in the next room .
Along the crowded London roads they went , slowly at

<avatar>Dick</avatar> and <avatar>Julian</avatar>, who shared **a <setting>room</setting>**, woke up at about the same moment, and stared out of the nearby window. … **<avatar>Anne</avatar>** slept in the next **<setting>room</setting>**. **<avatar>Julian</avatar>** ran **<transition type='ARRIVAL' subject='Julian'>in </transition>** and shook her.…Along the crowded **<setting>London</setting>** roads they went, slowly at first…

Figure 13. Travel sequence from *The Famous Five: Five on a Treasure Island* by Enid Blyton (extract 4) - 1 object deleted from the room scene.

We identify some limitations with the process described in this article. Automatic annotations often result in the creation of extraneous objects in a scene (evidence of which is indicated by the number of deleted objects in each example presented above). For example, the process is unable to relate collective annotations such as “children” to a collection of entity descriptors, and instead creates a new avatar. Also, as evident in some images, certain semantic conditions are not handled correctly by the automated process (for example, ANNE INSIDE BED literally places the ANNE model inside the BED model, and not on top of it). However, the only mechanism to counter this problem is the use of encoded world knowledge similar to that used by Ma (2006), which is contrary to our knowledge-poor paradigm.

We also observe that the constraint optimizer, while functioning correctly for curves of low degree (≤ 2), is not able to locate solutions to systems containing curves of higher degree within a feasible time. While still producing representative behaviour, the currently produced motion could be improved. This can be achieved in future by providing additional heuristic optimizations to the constraint solving process.

In spite of the above limitations, we conclude that corresponding multi-modal animated 3D virtual environments and films are successfully created using the processes described in the previous sections. This conclusion is supported by the following observations from the experiments conducted in this section:

1. The creation of high-level scene descriptions (including the identification of scenes, their content, and behaviour) is performed consistently, requiring minimal human modification.
2. Virtual environments are populated in a manner that requires little significant modification.
3. Automatically generated virtual environments are representative of the input text, specifically with regards to the following aspects:
   (a) Behaviour specified by annotation categories such as transition and relation is visualized correctly in a virtual environment.
   (b) Visual sequences containing multiple scenes are created that correctly represent descriptions in the input text.
   (c) The automated process supports the creation of multi-modal animated 3D virtual environments and films across different types of books.

8. CONCLUSION

We present examples of 3D animated scenes that are automatically constructed from annotated fiction text. The process readily allows for creative input from human directors to add artistic vision and external world knowledge. Abstract constraints are derived from suitably annotated fiction text and these are used to restrict the form of object trajectories within the scene. We outline a process for determining these trajectories using a quantified interval-based constraint optimization algorithm that produces solutions valid over continuous time intervals while bounding the amount of processing required. Annotations and trajectories are used in combination with a detailed model library for the instantiation of 3D environments.

The interpretation of annotations is performed successfully in the creation of high-level structured scene descriptions using knowledge-poor approaches. Virtual environments are successfully instantiated and populated from scene descriptions using automated techniques. There is no need for repetitive 3D modeling and animation, and rich virtual environments are created rapidly.

To our knowledge the research presented here is the first to describe a system that converts extensive extracts from popular fiction into corresponding animations. We use a knowledge-poor approach that allows human intervention in the conversion process. This is different to existing text-to-graphics research which focuses on providing fully automatic knowledge-centric processes. The use of corresponding audio files for deriving temporal information for quantifying behaviour is innovative in the domain of text-to-graphics. We present the first use of interval-based constraint optimization for determining quantified scene behaviour.

The methods described in this article contribute to the field of virtual reality by providing a mechanism for automatically instantiating and populating virtual environments. Our method creates virtual environments without the need for extensive manual effort in 3D modeling, environment design and motion quantification.

REFERENCES

Badler, N.I., Bindiganavale, R., Allbeck, J., Schuler, W., Zhao, L., and Palmer, M., 2000. Parameterized action representation for virtual human agents. In *Embodied conversational agents.* Vol. 1, pp. 256–284.

Belhadj, F., 2007. Terrain modeling: a constrained fractal model. In *AFRIGRAPH '07: Proceedings of the 5th international conference on Computer graphics, virtual reality, visualization and interaction in Africa.* Grahamstown, South Africa, pp. 197–204.

Benhamou, F., Goulard, F., Languénou, E., and Christie, M., 2004. Interval constraint solving for camera control and motion planning. In *ACM Transactions on Computational Logic*. Vol 5, No. 4, pp. 732–767.

Christie, M., Languénou, E., and Granvilliers, L., 2002. Modeling camera control with constrained hypertubes. In *CP '02: Proceedings of the 8th International Conference on Principles and Practice of Constraint Programming*. London, United Kingdom, pp. 618–632.

Clay, S.R. and Wilhelms, J., 1996. Put: Language-based interactive manipulation of objects. *IEEE Computer Graphics and Applications*. Vol. 16, No. 2, pp. 31–39.

Coyne, B. and Sproat, R., 2001. Wordseye: an automatic text-to-scene conversion system. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. Los Angeles, USA, pp. 487–496.

Fellbaum, C. editor, 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, USA.

Glass, K. and Bangay, S., 2006. Hierarchical rule generalisation for speaker identification in fiction books. In *Proceedings of SAICSIT '06*. Cape Town, South Africa, pp. 31–40.

Glass, K. and Bangay, S., 2005. Evaluating parts-of-speech taggers for use in a text-to-scene conversion system. *Proceedings of SAICSIT '05*. White River, South Africa, pp. 20–28.

Glass, K., Bangay, S., and Alcock, B., 2007. Mechanisms for multimodality: taking fiction to another dimension. In *AFRIGRAPH '07: Proceedings of the 5th international conference on Computer graphics, virtual reality, visualisation and interaction in Africa*. Grahamstown, South Africa, pp. 135–144.

Glass, K., 2008. *Automating the Conversion of Natural Language Fiction to Multi-Modal 3D Animated Virtual Environments*. PhD thesis, Rhodes University, Grahamstown, South Africa.

Johansson, R., Bergland, A., Danielson, M., and Nugues, P., 2005. Automatic text-to-scene conversion in the traffic accident domain. In *IJCAI-05: Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*. Edinburgh, Scotland, pp. 1073–1078, Joshi, D., Wang, J.Z., and Li, J., 2004. The story picturing engine: finding elite images to illustrate a story using mutual reinforcement. In *Proceedings of the 6th ACM SIGMM international workshop on Multimedia information retrieval.* New York, USA, pp. 1073–1078.

Lu, R. and Zhang, S., 2002. Automatic Generation of Computer Animation: using AI for movie animation. *Lecture Notes in Computer Science*, Vol 2160. Springer.

Ma, M., 2006. *Automatic conversion of natural language to 3D animation*. PhD thesis, University of Ulster, Ulster, Ireland.

Parish, Y.I.H. and Muller, P., 2001. Procedural modeling of cities. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques.* Los Angeles, USA, pp 301–308.

Seversky, L.M. and Yin, L., 2006. Real-time automatic 3D scene generation from natural language voice and text descriptions. In *Proceedings of the 14th annual ACM international conference on Multimedia.* London, United Kingdom, pp 61–64.

Zeng, X., Mehdi, Q.H., and Gough, N.E., 2003. Shape of the story: Story visualization techniques. In *IV'03: Proceedings of the Seventh International Conference on Information Visualization*. pp. 144–149.

Zeng, X., Mehdi, Q.H., and Gough, N.E., 2005. From visual semantic parameterization to graphic visualization. In *IV'05: Proceedings of the Ninth International Conference on Information Visualisation*. London, United Kingdom, pp 488–493.

Zhu, X., Goldberg, A., Eldaway, M., Dyer, C., and Strock, B., 2007. A text-to-picture synthesis system for augmenting communication. In *AAAI'07: The Integrated Intelligence Track of the 22nd AAAI Conference on Artificial Intelligence*. Vancouver, Canada, pp 1590–1596.