Meeting the Spirit of History

Ursula Kretschmer, Volker Coors
Fraunhofer-IGD, Rundeturmstr. 6, D – 64283 Darmstadt
{Ursula.Kretschmer, Volker.Coors}@igd.fhg.de

Ulrike Spierling, Dieter Grasbon
ZGDV, Rundeturmstr. 6, D – 64283 Darmstadt
{Ulrike.Spierling, Dieter.Grasbon}@zgdv.de

Kerstin Schneider, Isabel Rojas, Rainer Malaka
EML, Schloß-Wolfsbrunnenweg 33, D - 69118 Heidelberg
{Kerstin.Schneider, Isabel.Rojas, Rainer.Malaka}@eml.villa-bosch.de

1 Abstract

This paper describes a research and development project for a novel technology, making the conveyance of cultural heritage during a historic sightseeing tour a unique experience. The cornerstones of this system are mobile augmented reality, including a hybrid tracking approach, intelligent queries to pose complex questions about geographical and historical knowledge, as well as a story engine to interactively run a digital story. This system involves the user in a thrilling story while exploring the history and the attractions of the city.

Keywords:
Interactive digital storytelling, intelligent geo-temporal databases, mobile augmented reality

2 Introduction

When walking through historic, cultural places, you sometimes feel as if the spirits of the past were distinctly present in the ruins and could tell many lively stories about a building or a place – if only you could see and communicate with them. Yet all that is really required is the proper "magic equipment" designed for a pedestrian to wear.

This paper describes a research and development project for a novel technology that makes the conveyance of cultural heritage during a historic sightseeing tour a unique experience. This project is called 'GEIST' and the attractiveness of the means will surely increase – compared to traditional guided tours - the target group of persons interested in historical knowledge.

In the future, the technology presented here will redeem the age-long nostalgia to meet the spirits of the past at their very field of activity in the form of moving figures, gliding in space directly in the real scenery of a ruin or landscape, visible through semi-permeable glasses. Embedded in a historically well-founded story or a learning game, players can release a restless ghost by delving into the concerns of that time and helping him bring his story to a (hopefully) fruitful conclusion. To do this, you have to go to the actual locality of the story, because only there can you meet more ghosts who will either help with further information or make the story more fascinating. The ghosts of specific locales perceive the presence of a player and can also show him the former appearance of the location.

This technology is tested using an attractive model: Heidelberg with "olde town" and castle. Especially in this scenery, spirits of quite different epochs are being evoked: bellicose neighbors from Palatinate and France fighting for property and power and lovesick romanticists and poets suffering the heartaches and turmoil of the Thirty Years' War.

The techniques combined in the project GEIST[1] are augmented reality (AR) developed by Fraunhofer-IGD in Darmstadt, digital storytelling as a competence of ZGDV in Darmstadt, and intelligent data query made available by EML in Heidelberg.

[1] The term 'Geist' means 'ghost' in English.

The following paper presents the technology applied to the GEIST system. This includes tracking, intelligent queries, and a description of the story engine. First, we give an overview of the state-of-the-art technology involved in mobile outdoor AR systems and interactive storytelling.

3 State of the Art

3.1 Mobile Outdoor AR Systems

AR is a technique to enhance the reality a person perceives by fading in information in his line of sight. This can be done by means of a small display on which the information is presented. At the same time, the user can watch the reality either because the screen is very small and does not occlude the whole environment or because there is a see-through modus integrated in the display. To cover the reality with the information on the correct position of the screen, the position and orientation of the user have to be determined. For this reason, the user is provided with a mobile computer. There are different possibilities for solving this problem. One example of a solution in closed rooms relies on markers fixed in the environment, which are detected by a camera. Analyses of the camera pictures provide the system with information about the orientation of the camera.

Systems being used in outdoor environments cannot be based on changes in the environment. Some projects use GPS (global positioning system) and sourceless tracking sensors to determine the orientation of the user. Sourceless tracking sensors are devices for determining head rotation, which do not base on the signal from a transmitter. Both Columbia University in New York [10], [18] and the University of South Australia [27] have developed such a system. The results are not very accurate and it seems difficult to overlap the real environment precisely. To stabilize the motion of the user, the team around Azuma and Hoff at HRL laboratories in Malibu [2] uses the results of the orientation sensors to extrapolate angular position and rotation of the head by instating a linear model. The results still contain signal gaps or falsified signals. Image processing can be used to label landmarks correctly. Park, You, and Neumann [26] presented a system in 1999 that tracks natural features with a priori unknown position. To get correct results, 3D information is necessary. At Rockwell Science Center, a system was developed which includes horizon silhouettes to solve the orientation problem [3]. The horizon silhouette detected in a video image is compared with data from a digital elevation map. The results are camera azimuth and elevation, as well as the calibration parameters. Here, the tracking results of the sensors are matched with the camera picture. This system can only be used in an open environment where the hills being used are relatively far away. The University of Tokyo presented a system in 1999 where problems with GPS in cities, as well as occlusion of moving vehicles, are compensated by analyzing image sequences [6]. A requirement of the algorithm is manual georeferencing of the image.

In the GEIST project, we strive for a similar approach. We are using a 3 D model of the environment to compare it with pictures in a video stream. In our project, manual geo-referencing cannot be applied to the system, as the tracking component has to be as automatic as possible. The initial values of position and orientation are gathered from a GPS and a tracking sensor.

3.2 Interactive Storytelling

Interactive storytelling is a live experience. Therefore, we need a run-time engine taking care of the user's experience as the story keeps unfolding. Interactive drama is the class of narratives generated by story engines. In most cases, it allows the user to step into the role of the protagonist and witness the events from a first-person perspective.

Early approaches to interactive drama emphasized the development of autonomous believable agents. This philosophy was introduced by the Oz project led by Joseph Bates at Carnegie-Mellon University [21]. The underlying assumption was that a coherent story line would emerge from the agents' autonomous behavior, given the help of infrequent interventions from the side of a drama manager [23].

Less work was done on interactive plot [22]. In recent years, however, several systems have been developed. The Erasmatron, a story engine developed and implemented by Chris Crawford, is based on a sophisticated world model. It seeks to balance character-based and plot-based approaches by using verbs as the basic components of action. Crawford does not believe in story generation through algorithms and therefore plans for the author to create a useful set of verbs that the engine can work with [9]. When designing a story engine for the DEFACTO project, Nikitas M. Sgouros followed a different approach. Using a rule-based system, he aimed at shaping interactive plot into the structure of Aristotelian drama by modeling rising conflict between the characters [30]. Nicolas Szilas takes a similar approach [31]. Michael Mateas and Andrew Stern are working on an interactive story world. They define beats as the basic units of plot and interaction [23].

We believe that all of these approaches have a rather fine granularity of plot control in common, which in turn provides the user with frequent opportunities for changing the plot. Although this is obviously an important goal for interactive storytelling, we have decided to assign it less priority in our approach. This is because we believe technology does not yet

provide us with means to create meaningful stories under this condition. Therefore, we chose to deal with interactive plot at a higher level – the level of morphological functions as defined by Russian formalist Vladimir Propp [28].

4 The GEIST Technology

In order to allow seamless use and to free the user from the burden of using highly complex systems, it is necessary to design intelligent components that integrate and connect knowledge about the user's physical location in the environment with a story line and content in databases on city models and historical content. In the following sections, we give an overview of the technologies that will be the corner-stones of the GEIST system:

- Tracking to estimate position and line of sight of the user
- Intelligent queries in order to access and integrate multiple data sources
- Story engine to run interactive digital stories in the outdoor environment

4.1 Tracking

In the GEIST project, a combination of tracking sensors and information contained in a 3D model of the environment is planned for use. In the dense city centers where the buildings might be too close to each other and quite tall, satellite signals are hard to receive and the results of image processing are necessary to bridge reception gaps. These developments will include the use of a camera representing the viewing field of the user. This 2D information should be compared with 3D information contained in a model of the environment. This can be done by generating views around the position of the tourist and comparing these 2D images with a picture out of the video stream. Here, the number of generated views depends on the accuracy of the position and orientation trackers. At this point, the reception of a GPS signal is also required. A view will not be generated exactly at the position where the user is standing. However, there will be one view with a very high amount of the same information as in the camera image. The camera can be rotated around three axes: yaw, pitch, and roll. By only receiving information about the position of the tourist, the yaw angle can be between 0 and 360 degrees. As the tourist normally does not focus on the sky or the ground of the environment, pitch should usually be between +40 and –30 degrees (looking above or on the ground). In our first approach, a rotation around roll is neglected, as this is an unusual movement of the head.

To get real-time results, the generation of several views for a constant comparison with images of the camera is not feasible**.** In the GEIST scenario, this should only be done if the most accurate position and orientation results are necessary. This is the case when 3D reconstructions of surrounding buildings should be rendered and faded in the line of sight of the user. Then, the user will stand still for some minutes while getting information on the environment he is looking at. In other cases, when the tourist just moves around the city, walking from one interesting point to the next, the system will not have sufficient time to generate views out of the model. Here, we strive for an approach of getting position and orientation information out of successive pictures of the video stream. This technique permits getting non-scalable information of the relative orientation of a camera. In our system, we are in need of absolute results, i.e., we require information about the scale. This information is received from the 3D models at points where the tracking information is obtained by a comparison between camera image and views of the model. The features, which are compared, have to be tracked in successive pictures. Out of all these pairs, the absolute position and orientation of the real camera can be computed. This will be done until the user reaches a place where historical buildings are shown on his display. Here, accurate tracking is required.

The complete tracking system consists of different components. Each component must be checked separately. Figure 1 shows the basic components of our video tracking system.

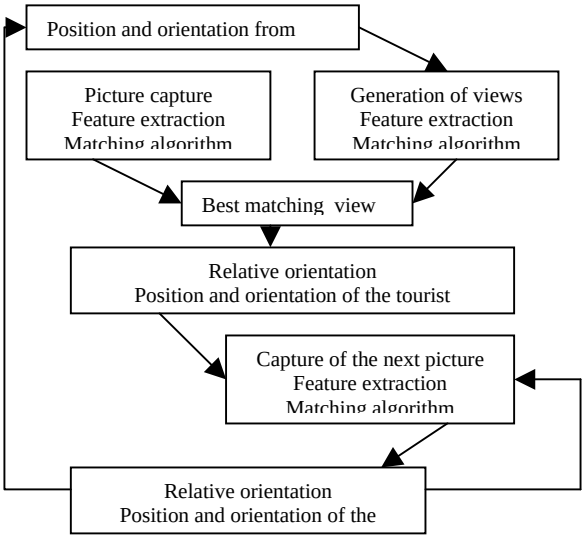

Figure 1: Components of the tracking system

We have implemented a very early version of some components of our system [8]. We realized the capture of the views, a feature detection, a matching algorithm based on the work of Gros et al. [14], which also includes the detection of the best matching view, and the evaluation of position and orientation of a camera in relation to another camera. Besides the feature extraction module [32], all the components are programmed in Java [17].

4.1.1 Capture of the views

Java3D is used for this component. At the moment, the 3D model of the environment is a 3D-Studio Max model. As it is not possible to import this format into Java3D, the model is exported in VRML. The VRMLloader of the 'Java3D and VRML working group' builds a Java3D scenegraph. The camera is defined by its internal and external camera parameters, as well as by its projection behavior. It then captures an image of the model.

4.1.2 Feature extraction

As preparation for the feature extraction module of the current system, a low-pass filter helps to eliminate the secondary information out of the image. After this groundwork, edges can be detected within the image by use of the sobel filter. These edges then have to be vectorized. As a first step, a thickening and then a thinning algorithm is executed for the filtered image. An edge-tracking algorithm is used to close the gap between parts of a line. For the complete line, the thinning algorithm is used and all the pixels belonging to one line are stored in a list. An algorithm is applied to these so-called "pixel snakes" to represent them by polygons.

4.1.3 Matching algorithm

To match an image pair, the algorithm of Gros et al. [14] is applied to our system. This includes the extraction of segment configurations. These are two straight lines with a common endpoint. The configurations in one image can be transformed into a configuration in another image for all patterns visible in both images. This transformation can be approximated by a similarity. The transformation parameters are rotation, translation, and scale. The invariants of this transformation are the angle between the segments and the length ratio. These invariants are calculated for each segment configuration in each image. The results are used to detect matching configurations. To do so, they are ordered by angle for each image. Possible matching configurations are detected by restricting the search to similar invariants. For each correspondent configuration pair, the parameters of the similarity are calculated. All correct correspondences are roughly transformed by the same transformation. Similar to a Hough transformation, a region in 2D is looked for where most of the parameters can be found. These parameters are used to calculate the matching edges.

4.1.4 Best matching view

The algorithm described above was used for every image pair. A result for the match between picture and view is to get the best matching image pair. The similarity value for each pair is calculated by the use of the number of edges and endpoints in relation to the corresponding edges and endpoints.

4.1.5 Relative orientation

As a first step, rotation and translation between the two cameras have to be calculated. This process is called relative orientation. If we want to match a view with a picture, it is the orientation between the virtual and the real camera. If we match two successive pictures of a video stream, it is the relative orientation of two real cameras. The transformation can be expressed by the essential matrix E [16]. It is evaluated by taking into account the detected edges as results of the matching algorithm. The similarity is only an approximation of a 2D transformation between two images. There are non-matching edges within the number of features due to preparation for and calculation of the matching algorithm, as well as due to image noise. To detect outliers and incorrect pairs, adjustment algorithms are built into the calculation of the essential matrix. In the case at hand, the least median squares technique is applied to the relative orientation. Here, a random volume of points is gathered to calculate the essential matrix. The result is the rotation and translation matrix between the two cameras.

4.1.6 Position and orientation of the tourist

As the absolute position and orientation of one camera is known, the absolute values of the second camera can be calculated. When we are dealing with the comparison between a generated view and a picture from the video stream, the parameters of the virtual camera are known. Otherwise, the parameters of the previous image are used.

4.1.7 Test of the components

A first test was applied to a model of the residence in Darmstadt. Out of this model, views were generated. One example is shown in figure 2.

Figure 2: One generated view of the Darmstadt model

This view and a picture taken with a video camera connected to a wearable computer were loaded in the system. To do so, the size of the picture and the view had to be changed and transformed in grayscale and in tiff format. The feature extraction, as well as the matching algorithm, was then applied to both the picture and the view. The following figure (Figure 3) shows the results of the matching algorithm.

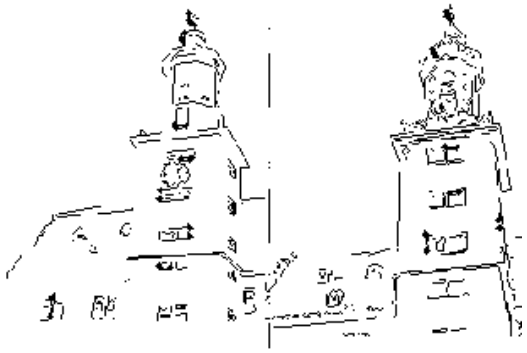

Figure 3: Matching results for comparison of a view of the 3D model (left) with a picture taken with a digital camera (right)

This result shows that the edge detection algorithm works quite satisfactorily. Many short lines were discovered, but long contours were also present. A better algorithm for closing gaps between part of the edges, as well as another solution for excluding short lines, should be applied to the system. The matching algorithm leads to some correct results. The major part, however, shows that lines were matched with non-identical lines. A solution for this problem would be to replace the edge detection with a point detection algorithm. The matching algorithm is partly based on the comparison of lengths. In fact, this is a point-based matching algorithm, as the main information of our line detection algorithm is the orientation of the edges.

4.2 Intelligent Queries

In the application domain of GEIST, we want to tell stories about the history in and of a city at the actual places where historical buildings stood or events took place. We, therefore, need repositories that know about the geography and history of the city. Moreover, story designers (authors) or users might want to ask questions or request information, such as

1. How far did Kathy have to walk to get to the well?
2. What did the Heidelberg castle look like around 1630?
3. Which buildings (or spatial objects) were related to Salomon de Caus?
4. Show me the next southerly building related to a historical event that occurred around the year 1630 (i.e., for which building historical or fictional information is available).
5. Show me a map of Heidelberg around the year 1630 highlighting the spatial objects for which there is information available.
6. Who was the Elector Palatine in 1618?
7. How many people lived in Heidelberg at the end of the 30-year religious war?
8. Show me the way to the next whiskey bar.

To find answers, the system relies on a set of interconnected databases containing information about historical facts, spatial objects (e.g. buildings and architectural objects in the city) and fictional elements (e.g. legends, hypothetical beliefs) that can be used as building blocks for stories and for connecting the story to historical facts (Figure 4). The construction of the answers to these questions may require access to one or more of these databases, and preprocessing or post-processing of the questions or answers, respectively, may also be necessary.

A query engine is a core part of the system that breaks down complex questions into particular database queries and than reassembles the particular answers of the databases into proper answers. In the authoring system, the query engine supports the author in the task of building the story by supplying historical facts or fictional elements that can be inserted to build the story line. Architectural objects can be inserted to construct the scenario of the story. The query engine will allow the author to extract information on architectural elements relevant to the historical period, person, historical event, etc. contained in the story. With this information, plus information on the geographical location of these architectural objects, the author can plan the routing of the story and synchronize it with the

story contents. All of this information is then stored in the story engine.

At runtime, the query engine will support the system by supplying information about the location and 3D models of the architectural elements that have been defined as relevant for the story. The renderer will use these 3D models in order to present them in the AR system.

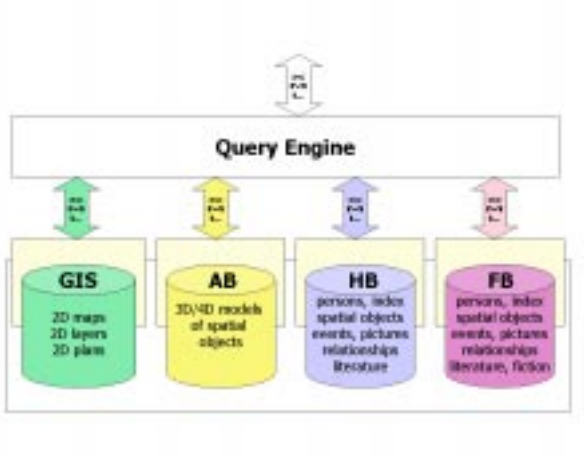

Figure 4: The main databases of the system: Geographical Information System (GIS), Architecture Base (AB), History Base (HB) and Fiction Base (FB)

The system must permit the querying of temporal and spatial data where the temporal attributes are not necessarily described by timestamps. When talking about historical periods, it is possible that the beginning and end of events are not time-points, but general intervals (e.g. we say a historical event happened during or around a certain time point; this is not exactly defined).

To answer queries concerning the current location of the user, e.g. if the user located in a story-relevant place or queries related to the current scene of the story or to the general contents of the story, the query engine will require information from other components, namely the story engine, the observer and the synchronizer (Figure 5).

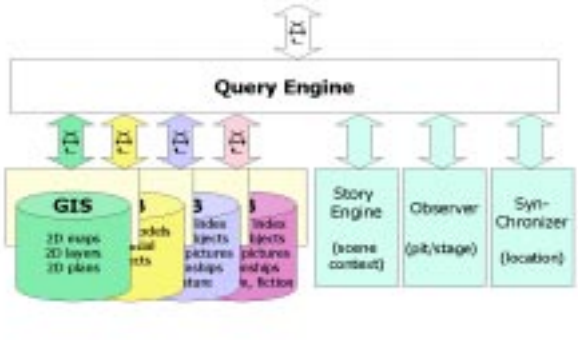

Figure 5: Additional sources for obtaining state and location information

As a starting point, we can use existing databases that have been developed in a number of projects at EML and that already provide content on buildings and historical sights [20] [34] [33] [35]. However, these databases have been implemented employing different types of database systems (i.e., object-oriented and object relational database systems). The query engine must be able to handle this heterogeneity. To address this issue, the query engine is designed in 3 layers: the query-processing layer, the federation (or integration) layer and the foundation layer (Figure 6). The query-processing layer is responsible for dividing the original query into sub-queries (when necessary), coordinating the processing sequence of the sub-queries and building the final answer. In the federation layer, the distribution of the queries or sub-queries is done together with the integration of the partial answers provided by the difference databases. The foundation layer contains the databases.

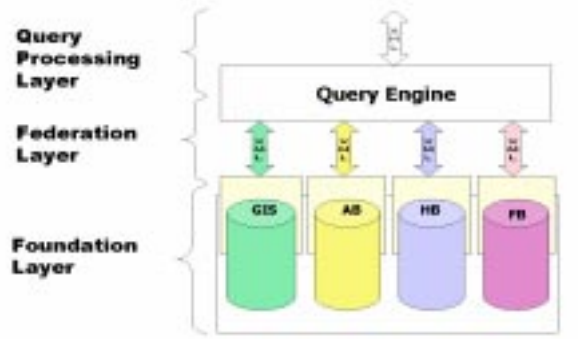

Figure 6: The 3 layers

There are some types of queries that require access to all the databases mentioned. For example, the first query (How far did Kathy have to walk to get to the well?) requires data about Kathy, who is a fictional person, with information stored in the Fiction Base. It additionally requires data about the well, which is a real historical object; so, the requested historical information can be found in the History Base. The 3D model of the well is stored in the Architecture Base. Finally, information from the GIS (Geographical Information System) provides geographical data including the routing information to identify the path and the distance that Kathy must walk in order to get to the well. Query 4 requires data about the visitor's location (Synchronizer), the next southerly buildings (GIS), historical events (HB, FB), as well as the selected building (AB, HB, FB).

4.2.1 3D data model

A query-oriented data model for the architecture base should support efficient query handling. It must be easy to extract information from the database about thematic and geometric aspects of spatial features and their topological relationships. It is essential to store objects beyond those with a 3D spatial extension. Relevant objects may also include point, line, or planar geometry.

Molenaar has proposed a 3D formal data structure (3D-FDS) as an extension of his *single vector value map* concept in [24]. He uses a vector approach and stores 3D topology based primarily as boundary relationships. Zlatanova and Tempfli pointed out that the data organization in this model is not efficient at supplying data for visualization purposes [36]. Flick proposes an extension of the 3D-FDS, explicitly storing adjacency, incidence and inclusion between all geometric objects [11], which leads to an even larger amount of data. In our opinion, the main disadvantage in these approaches is the explicit storage of edges. For polyhedrons, Euler's formula gives the relation between numbers of vertices V, edges E and Faces F as V-E+F=2. This means that in a typical polygonal model, there are as many edges as there are nodes and faces added together. A proof of this formula is given in [15]. However, most topological queries for a city model are not based on edges, but on adjacent faces.

Our proposed model consists of three fundamental levels: a *feature* represents a real world entity and is the link between geometry and semantic information. The four elementary objects (Point, Line, Surface, and Body) are used to represent the geometry of an n-dimensional feature (n=0, 1, 2, 3) and the four construction objects (Node, Arc, Face, and Solid) build the geometry and store topological information between the construction objects. A Solid is a polyhedron and is defined by its bounding faces. The orientation of a face is given implicitly by the order of bounding nodes. Face adjacency is implicitly given by shared nodes. Two convex polygons are adjacent if they share at least two nodes. To use this property, we restrict the face geometry to planar convex polygons as in [36]. Arcs are only used to present line objects. See Figure 7 for more details on the query data model.

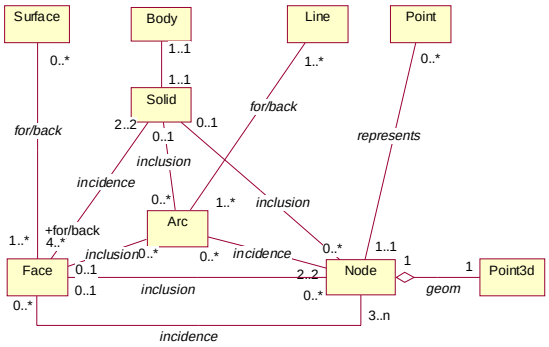

Figure 7: Geometry and topology in the query-oriented data model

Visual attributes, mainly used for presentation purposes are stored independently from the geometry. In order to support several degrees of details in visualization, a view concept as introduced by [11] is used to store different presentations for each feature. Due to the observation that topological query are not normally concerned with the small details of a 3D building model, these views are not stored in the query data model as done by Flick, but in an output-oriented model, such as triangle meshes or compressed geometry. However, some constraints between the feature geometry and its views should be established, e.g. the footprint of a building has to be consistent in all views. The view concept was used to realize a level-of-detail concept in [7]. Another advantage of the view concept is that the feature geometry and its views do not have to be of the same dimension. A three-dimensional feature can be represented by a textured plane or a panoramic view, for example. This may be very useful for the visualization on a small mobile device without 3D rendering acceleration. On the other hand, point features can be represented in a realistic manner using a 3D view. Representing a tree with full 3D geometry in the query model does not have any benefit. In most applications, it is much more efficient to represent it as a point feature. A 3D model of a tree can be stored as a view.

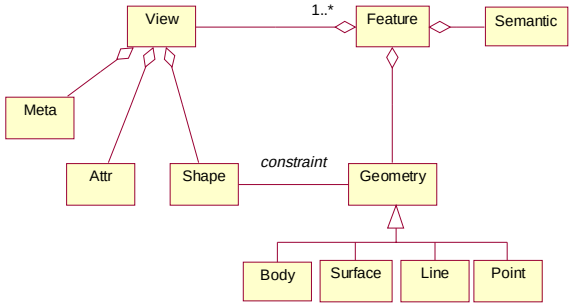

Figure 8: View concept: distinguish between the feature geometry and its presentation

4.3 Story Engine

Story engines are tools that tell interactive stories. The main goal of the previously described system "GEIST" is not only to provide mere information about historical facts, but to entertain and to tell stories. Like in a historical novel, real history is explained best by example stories of individual characters. These can be fictional, but they should appear coherent to actual historical facts.

The goal of the project is for users not to perceive the devices as technology. We prefer to view the augmented reality equipment the users interact with as “magic equipment” that hides all technological issues and that can be used to “summon” virtual characters (ghosts) that allow for interaction with historical content.

4.3.1 Authorship

The most critical part of the design is defining the borderline between machine generation and human authorship. Most story-generating systems use a model or grammar of what stories are supposed to be like. It appears to us that no single model can describe all possible narratives, however. As far as we can tell, this has not even been accomplished for any subset of stories. We agree with Janet Murray who observes that most of these story models have a very reductive quality [25]. In our view, the creative parts of story generation must be performed by human authors alone.

On the other hand, if the author has to create everything single-handedly, the story space remains very limited. This is the case for any static structure, such as trees or a string of pearls – they limit the number of different plots to just a few, which makes the ability to interact essentially meaningless. Thus, it is evident that any interactive system should work with a dynamic representation of what the author created. Since we do not believe any predefined model can satisfy the author’s needs, our main concern is to make its representation accessible. The author should continuously shape the story model to meet expectations as they develop during the authoring process. This has been suggested by Janet Murray, as well [25].

Phoebe Sengers points out that narratives are always particular [29]. It is in the details that the shaping influence of the author becomes visible. We do not believe that machines are able to create convincing details. Therefore, we have decided to put the responsibility of generation at scene level and below entirely into the hands of the author.

The GEIST storytelling system does not generate the details of the story. It relies on plot elements that are originally created by authors, and presents them interactively during runtime.

Interactivity is a live experience. Occurrences, topics covered and phenomena are subject to change, depending on interactions and on environmental issues, as in GEIST. However, an emergent narrative approach cannot be used with the proposed application for a number of reasons. The main point being we do not want to change history. We *do* want to interactively experience historical events *now* and they are therefore described in a background story.

So, interactive story *telling* in GEIST *relies on* a predefined story space, a specific variety of plot elements concerning facts and occurrences. Only the *telling* of the story is done interactively.

4.3.2 The role of the user

When we say that our system features interactive storytelling, we do not mean that the user has free will to do whatever he wants in the story without destroying its coherence. Though this free reign might seem attractive, we do not see how it could be realized. Espen Aarseth considers it impossible in general when evaluating Brenda Laurel’s work and suggests an approach based on emergence instead [1]. Unfortunately, we do not believe emergent narrative will yield the high-level plot structures we are looking for. This leaves us with just one alternative: we cannot allow the user to randomly change the remaining plot. Individual scenes of the story can be highly interactive, but user interaction cannot change their function for the remaining story. This does not restrict interaction very much; the user can navigate freely and behave in a wide variety of ways. However, consequences for the plot are limited to the comparatively small number of decisions that our engine makes (when choosing which scene to show next). Thus, our system takes into account user interaction when adapting the future narrative, but it does so infrequently.

Since we cannot constrict user movement in our augmented reality scenario, we have to allow free navigation. This has implications for the stories that can be told, as well as for the design of the engine. If the user takes over the role of the protagonist, he cannot be kidnapped or arrested. Instead, we will send the player out on a search of some kind. The engine has to ensure that the story keeps unfolding wherever the user goes.

4.3.3 The story model

The story model of our prototype is based on Vladimir Propp's "Morphology of the folktale" [28], which was written in 1928. As the basis of his study, Propp used an arbitrary sample of 100 Russian fairy tales. As mentioned by Selmer Bringsjord and David Ferrucci [5], Vladimir Propp's system of classification inspired research on story grammars, one of the fundamental approaches to story generation. Since our goal is not story generation, but rather guiding interactive drama, we use Propp's work to create rules and algorithms for the run-time engine instead of defining a grammar. The application of Propp's morphological approach to interactive storytelling was suggested and sketched by Janet Murray.

Propp's classification is continuously aware of the storyteller's branching possibilities between morphological variants. Arthur Berger points out that Propp's functions constitute the core of all narratives to this day [4]. Propp defines function as follows: "Function must be taken as an act of dramatis personae, which is defined from the point of view of its significance for the course of action of a tale as a whole." [28]

Thus, functions are independent of the specific characters, as well as of the particular actions that perform them. Propp's analysis revealed that the number of functions to be encountered in a large body of narratives is rather limited. Furthermore, they tend to appear in one and the same order in the fairy tales examined. Stories differentiate themselves in regard to which functions have been left out. Many functions exist in different variants that are in part correlated with variants of other functions. Functions themselves often appear in pairs. For example, any misfortune caused by villainy should be followed by a liquidation of that misfortune. Propp suggested grouping functions according to the dramatis personae that perform them. By *dramatis personae*, he was referring to different types of roles in the abstract plot, not to specific characters. Among those types, he counted the villain, the donor, the helper, the person sought after and her father, the dispatcher, the false hero and the hero. Most of these types could take the form of any particular character, a group of characters, an animal or even a thing, depending on the morphological variant. Propp also discovered that several narrative sequences (in his terminology, '*moves*') could follow each other or even overlap.

We have chosen to implement Propp's system for a variety of reasons. Bride Linane-Mallon and Brian Webb argue that stories cannot be divided into discrete modules, since their elements are highly interrelated [19]. Phoebe Sengers points out that the cause and effect of narrative events are more important than the events themselves. We believe Propp has solved these problems by defining functions in the context of the surrounding story.

4.3.4 Functions and scenes

The story engine works with two levels of abstraction. At the upper level, a sequence of functions is selected in real-time. We use the term *function* in the same way as Propp does, as an abstraction from characters, setting and action while emphasizing the function's role in the surrounding plot. User interaction and other constraints can result in functions being skipped, as well as in the choice of a different variant of a specific function. At the lower level, a particular scene is executed for each function. They are either being generated or have been authored in advance.

We see functions as classes, and scenes as instances of those classes. A scene possesses unity of time, space and action, thus facilitating temporal, spatial and emotional immersion of the user. Between one scene and the next, leaps in time are possible and even desirable, since we do not want to bother the player with less exciting events.

User interaction cannot change the morphological function of most scenes after their beginning. We have implemented a few polymorphic functions, however. Our conception of polymorphic functions was inspired by the notion of polymorphic beats, introduced by Mateas and Stern at a different level of detail. They refer to the most basic elements of action, which can have different outcomes depending on user interaction [23]. We are referring to scenes composed of a number of actions at a specific location. When the user enters a scene that corresponds to a polymorphic function, its outcome is not yet determined. It is important to note that by *outcome*, we do not mean the content of the scene (which we allow to be flexible in case of non-polymorphic functions, as well), but rather its function for the overall story. Thus, user interaction and other dynamic factors influence which morphological variant is chosen for the scene after its execution.

We have implemented polymorphic functions for the outcome of attempts of villainy and for the reaction of the hero to riddles and tests. User interaction decides if these functions succeed or fail, which has direct consequences for the remaining plot. If we

need these functions to succeed (as in the case of villainy), we either repeat them with different scenes or choose a nonpolymorphic variant with the desired outcome.

4.3.5 Scene selection

Propp uses capital letters to denote functions. In function A, the villain causes harm or injury. Later, the protagonist is tested by a potential donor (D). Depending on his reaction (E), he is given a magical agent (F), or not. Propp discovered that any D can follow an A, but each D requires a specific E. Further, D and E can be repeated in some cases or even be skipped altogether.

Regarding the selection of the next function, Propp gave an informal description of an algorithm that we have implemented: "It is possible to artificially create new plots of an unlimited number. All of these plots will reflect a basic scheme, while they themselves may not resemble one another. In order to create a folktale artificially, one may take any A, one of the possible B's, then a C, followed by absolutely any D, then an E, then one of the possible F's, then any G, and so on. Here, any elements may be dropped (apart, actually, from A or a), or repeated three times, or repeated in various aspects." [28]

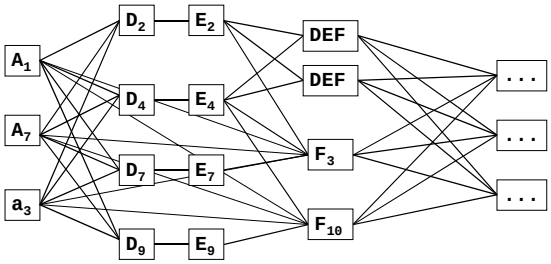

Figure 9: A section of the story space

Figure 9 illustrates this algorithm. The section is traversed from left to right while each square represents a specific scene. The scenes depicted in the diagram fulfill the functions A, D, E and F. It is important to note that the connections between them are not fixed, but rule-based.

Since the engine is still left with a variety of scenes to choose from, additional selection criteria are taken into account. The user chooses a certain time frame for the overall experience that has to be met. If the player's pace of interaction is very slow, many functions will be skipped. User timing, therefore, has an indirect effect on the remaining plot. Before selecting a scene, the engine checks whether it can be played in the remaining time. If the function of the scene requires other functions to follow, the durations of their corresponding scenes have to be taken into account, as well. With each scene description, the author has to encode an interval of its shortest and longest possible duration.

If the player appears unchallenged and if there is enough time remaining, a second *move* (Propp's term for a self-contained sequence of action) could be started. Before doing so, the engine has to make sure that there are enough scenes available for the additional plot without duplicating or contradicting previous events.

In relation to our augmented reality scenario, the number of possible scenes is further reduced by the current location of the user in physical space (unless scenes can be generated for each setting from templates). The user will feel immersed in the story, because it keeps unfolding wherever he goes.

Yet another criterion is the current context of the story. The author encodes a list of context requirements with each scene description. Scenes can create new context (i.e. if they introduce characters to the story). They can require context to be present (i.e. a certain type of misfortune) and they can remove context (i.e. if that misfortune is liquidated). If the system is still left with a set of choices after processing these criteria, a random decision is made.

We have implemented a prototype of the story engine in Prolog [12]. A detailed description of the system is given in [13].

5 Future work

In a first prototype, which will be available in February 2002, we will present a short scenario at the castle of Heidelberg. The main emphasis of this prototype is on the presentation of the inclusion of all components: AR, digital storytelling, access to the databases and first presentation by way of the head-mounted display. The AR system will consist of the Differential GPS device geo-kombi 12 B plus with additional software to reduce the influence of multipath, as well as an InterTrax head tracker II. In the future, these tracking results will be improved by the video-based approach described in this paper. The system for this first prototype will be run on a DELL Inspirion 8000 with an Nvidia GeForce2 Go graphic card due to 3D rendering performance. We hope that 3D graphic cards will be built in wearable computers very soon so that we can use either the Xybernaut MA V or the VIA II. The data will be stored in a small local database (Oracle Light or Cloudscape of Informix). Access to a larger database will be enabled by wireless LAN. In the scenario, there will be just one stage for the story. In future versions of the system, it will be possible to run the story at

different locations. Then, while involved in a thrilling story, the user can explore the history and the attractions of the city.

6 Acknowledgements

The GEIST project is funded by the BMBF (01 IRA 12A, 01 IRA 12B, 01 IRA 12C) and supported by the Klaus Tschira Foundation (KTS). The authors would also like to thank Reinhold Weinmann and Uwe Schwarting for their inspiring ideas and our fruitful discussions.

7 References

[1] Aarsetz, E.: *Cybertext: Perspectives on Ergodic Literature*, The Johns Hopkins University Press, Baltimore & London, 1997.

[2] Azuma, R., Hoff, B., Neely III, H., and R. Sarfaty: "A Motion-Stabilized Outdoor Augmented Reality System". *Proceedings of IEEE Virtual Reality '99.*

[3] Behringer, R.: "Registration for Outdoor Augmented Reality Applications Using Computer Vision Techniques and Hybrid Sensors". *Proceedings of IEEE VR '99.*

[4] Berger, A. A.: *Arthurs Computer- (Geschichten-) Abenteuer*, in Televizion 13/2000/1, Internationales Zentralinstitut für das Jugend- und Bildungsfernsehen, München, 2000.

[5] Bringsjord, S., and D. Ferrucci: *Artificial Intelligence and Literary Creativity: Inside the Mind of Brutus, A Storytelling Machine*, Lawrence Erlbaum, Mahwah, NJ, 1999.

[6] Chen, T., and R. Shibasaki: "A Versatile AR Type 3D Mobile GIS Based on Image Navigation Technology". *Int. Archives of Photogrammetry and Remote Sensing 1998.*

[7] Coors, V., and S. Flick (1998): "Integrating Levels of Detail in a Web-based 3D-GIS", *6th ACM Symposium on Geographic Information Systems (ACM GIS 98)*, Washington D.C., USA, 1998.

[8] Coors, V., Huch, T., and U. Kretschmer: "Matching buildings: Pose estimation in an urban environment". *ISAR 2000*.

[9] CRAWFORD, C.: "Assumptions underlying the Erasmatron interactive storytelling engine", in *Proceedings of the AAAI Fall Symposium on Narrative Intelligence*, 1999.

[10] Feiner, S., MacIntyre, B., Höllerer, T., and A. Webster: "A Touring Machine: Prototyping 3D Mobile Augmented Reality Systems for Exploring the Urban Environment". *Proceedings ISWC 1997.*

[11] Flick, S. (1996): "An object-oriented framework for the realization of 3D Geographic Information Systems", *Proceedings of 2nd joint European conference and exhibition on Geographical Information*, Barcelona, Spain, pp 187-196.

[12] Grasbon, D., and N. Braun: "A Morphological Approach to Interactive Storytelling", poster presented at *cast01: Living in Mixed Realities*, Bonn, September 21-22, 2001.

[13] Grasbon, D.: „Konzeption und prototypische Implementation einer Storyengine: Dynamisch-reaktives System zum Erzählen nichtlinear-interaktiver Geschichten bei größtmöglicher Spannung, gedanklicher Immersion, Identifikation und Motivation des Spielers", Diploma Thesis, Technical University of Darmstadt, 2001.

[14] Gros, P., Bournez, O., and E. Boyer: "Using geometric quasi-invariants to match and model images of line segments". *Rapport de recherche n^o 2608*, Institut national de recherche en informatique et en automatique (INRIA). Grenoble 1995.

[15] Henle, M. (1979): *A Combinatorial Introduction to Topology*. W.H. Freeman and Company San Francisco, 1979.

[16] Hildebrand, A.: *Bestimmung computer-graphischer Beschreibungsattribute für reale 3D-Objekte mittels Analyse von 2D-Rasterbildern*. TU Darmstadt 1996.

[17] Huch, T.: "Videobasiertes mobiles Trackingsystem in einer urbanen Umgebung unter Nutzung eines 3D-Stadtmodells". Diploma Thesis, TU Darmstadt 2000.

[18] Julier, S., Lanzagorta, M., Baillot, Y., Rosenblum, L., Feiner, S., and T. Höllerer: "Information Filtering for Mobile Augmented Reality". *ISAR 2000.*

[19] Linane-Mallon, B., and B. Webb: "Evaluating Narrative in Multimedia", in *DSV-IS'97, 4th International Eurographics Workshop*, Granada, Spain, 4-6 June, 1997, pp. 83-98.

[20] Malaka, R., and A. Zipf (2000): „Deep Map – challenging IT research in the framework of a tourist information

system", in D. R. Fesenmaier, S. Klein, D. Buhalis (eds), *Information and communication technologies in tourism 2000*, pp. 15-27, Springer-Verlag, Wien, New York.

[21] Mateas, M.: "An Oz-Centric Review of Interactive Drama and Believable Agents", in M. Wooldridge und M. Veloso, (Eds.), *AI Today: Recent Trends and Developments*, Lecture Notes in AI 1600, Berlin & New York, Springer, 1999.

[22] Mateas, M., and P. Sengers: "Introduction to the Narrative Intelligence Symposium", in *Proceedings of the AAAI Fall Symposium on Narrative Intelligence*, 1999.

[23] Mateas, M., and A. Stern: "Towards Integrating Plot and Character for Interactive Drama", in K. Dautenhahn, (Ed.), *Proceedings of the 2000 Fall Symposium: Socially Intelligent Agents: The Human in the Loop*, AAAI Press, Menlo Park, CA, pp. 113-118.

[24] Molenaar, M. (1992): "A topology for 3D vector maps", *ITC Journal* 1992-1, pp. 25-33.

[25] Murray, J.: *Hamlet on the Holodeck: The Future of Narrative in Cyberspace*, MIT Press, Cambridge, MA, 1998.

[26] Park, J., You, S., and U. Neumann: "Natural Feature Tracking for Augmented Reality". *IEEE Transactions on Multimedia 1999.*

[27] Piekarsi, W., Thomas, B., Hepworth, D., Gunter, B., and V. Demczuk: "An Architecture for Outdoor Wearable Computers to Support Augmented Reality and Multimedia Applications". *International Conference on Knowledge-Based Intelligent Information Engineering Systems 1999.*

[28] Propp, V.: "Morphology of the Folktale", in *International Journal of American Linguistics*, Vol. 24, Nr. 4, Part III, Bloomington, IN, 1958.

[29] Sengers, P.: "Narrative Intelligence", in K. Dautenhahn, (Ed.), *Human Cognition and Social Agent Technology*, Advances in Consciousness Series, John Benjamins Publishing Company, Philadelphia, PA, 2000.

[30] Sgouros, N. M.: "Dynamic Generation, Management and Resolution of Interactive Plots", in *Artificial Intelligence* 107(1), 1999, pp. 29-62.

[31] Szilas, N.: "Interactive Drama on Computer: Beyond Linear Narrative", in *Proceedings of the AAAI Fall Symposium on Narrative Intelligence*, 1999.

[32] Vögler, T.: „Impementierung einer Vektorzuordnung für die Auswertung von Stereobildpaaren in einer Videosequenz". Diploma Thesis. FH Mainz 2000.

[33] Weinmann, R., Malaka, R., Flechtner, I., Freiwald, N., and B. Vogel: "Architecture Base for 3D-Models of Buildings and Architectural Elements". *Sixteenth Annual Computers and the History of Art Conference* in London.

[34] Weinmann, R., Häußler, J., Zipf, A., and R. Malaka (2000): „Die Besucher Heidelbergs informieren - die multimediale historische Deep Map Datenbank". In: *HGG-Journal*. 2000/1 Heidelberg.

[35] Zipf, A., and R. Malaka: On the "Creation of User-Friendly Mobile Services Personalized for Tourism" - The service providers' view, in: *Proceedings of ENTER 2001, 8th International Congress on Tourism and Communications Technologies in Tourism*. Montreal, Canada. April 24-27, 2001.

[36] Zlatanova, S., and K. Tempfli (1998): "Data Structuring and Visualization of 3D Urban Data", in: *Proceedings of AGILE conference*, Enschede, The Netherlands, 1998.

Navigation Interface.

Augmented Reality presentation: original and augmented image.

Avatar athletes competing in the Stadium.

Vlahakis, Karigiannis, Almeida, Stricker, Gleue, Christou, Carlucci, Ioannidis: **ARCHEOGUIDE: First results of an Augmented Reality, Mobile Computing System in Cultural Heritage Sites,** pp. 131-140.

One generated view of the Darmstadt model.

Kretschmer, Coors, Spierling, Grasbon, Schneider, Rojas, Malaka: **Meeting the Spirit of History,** pp. 141-152.