A Service-Based Concept for Camera Control in 3D Geovirtual Environments

Jan Klimke, Benjamin Hagedorn and Jürgen Döllner

Abstract 3D geovirtual environments (3D GeoVEs) such as virtual 3D city models serve as integration platforms for complex geospatial information and facilitate effective use and communication of that information. Recent developments towards standards and service-based, interactive 3D geovisualization systems enable the large-scale distribution of 3D GeoVEs also by thin client applications that work on mobile devices or in web browsers. To construct such systems, 3D portrayal services can be used as building blocks for service-based rendering. Service-based approaches for 3D user interaction, however, have not been formalized and specified to a similar degree. In this paper, we present a concept for service-based 3D camera control as a key element of 3D user interaction used to explore and manipulate 3D GeoVEs and their objects. It is based on the decomposition of 3D user interaction functionality into a set of services that can be flexibly combined to build automated, assisting, and application-specific 3D user interaction tools, which fit into service-oriented architectures of GIS and SDI-based IT solutions. We discuss 3D camera techniques as well as categories of 3D camera tasks and derive a collection of general-purpose 3D interaction services. We also explain how to efficiently compose these services and discuss their impact on the architecture of service-based visualization systems. Furthermore, we outline an example of a distributed 3D

J. Klimke (✉) · B. Hagedorn · J. Döllner
Hasso-Plattner-Institut, University of Potsdam,
Prof.-Dr.-Helmert-Str. 2-3,
14482 Potsdam, Germany
e-mail: jan.klimke@hpi.uni-potsdam.de

B. Hagedorn
e-mail: benjamin.hagedorn@hpi.uni-potsdam.de

J. Döllner
e-mail: doellner@hpi.uni-potsdam.de

geovisualization system that shows how the concept can be applied to applications based on virtual 3D city models.

Keywords 3D camera control · Service-oriented architecture · 3D portrayal · Virtual 3D city models · 3D thin client

1 Introduction

3D geovirtual environments (3D GeoVEs) such as virtual 3D city models and 3D landscape models serve as integration platforms for complex 2D and 3D geospatial information. They provide a conceptual and technical framework to integrate, manage, edit, analyze, and visualize that information, facilitate use and communication of geospatial information, and represent a key functionality for IT solutions based on 3D GeoVEs. There is a growing number of application fields for 3D GeoVEs, in particular in those fields that require a true three-dimensional representation as in the case of virtual 3D city models. Because 3D GeoVEs commonly rely on massive, heterogeneous, and complex structured 3D geodata, high-quality, interactive 3D geovisualization systems usually demand for high processing power, large memory, and hardware-accelerated 3D graphics. These demands, however, make the development of robust, efficient, and compatible client applications a challenging task.

Service-oriented architectures (*SOA*) (Papazoglou et al. 2007), as a paradigm for design and development of distributed information systems, represent a common approach to address these challenges. 3D geovisualization systems, based on the SOA paradigm, encapsulate resource intensive tasks, such as management, processing, transmission, and rendering of massive 2D and 3D geodata as services that can be reused by various client applications. While 2D geovisualization systems can rely on standardized and robust services such as the Web Map Service (WMS), specified by the Open Geospatial Consortium (OGC), only first approaches and service implementations for 3D geovisualization, namely the *Web 3D Service* (W3DS) and the *Web View Service* (WVS), have been suggested (Schilling and Kolbe 2010; Hagedorn 2010). Recent developments in service-oriented architectures for interactive 3D portrayal aim at making 3D geodata, geodata management functionalities, and geospatial knowledge available even through *thin clients*, i.e., applications with low requirements concerning processing power and 3D graphics capabilities designed for lightweight platforms such as mobile phones or web browsers (Hildebrandt et al. 2011).

Besides capabilities for the presentation of 3D geodata, client applications need to offer tools to interact with 3D GeoVEs. 3D camera control, as the major 3D interaction type, enables users to explore and use a 3D GeoVE; it is crucial for its usability, as “a 3D world is only as useful as the user’s ability to get around and interact with the information within it” (Tan et al. 2001). Existing 3D portrayal services provide only rudimentary support for user interaction or camera control. Service-based approaches for 3D interaction have not been specified and formalized so far. Thus, 3D camera control functionality still needs to be designed and implemented separately for each client application.

In this paper, we present a concept for service-based 3D camera control as a key element of 3D user interaction used to explore 3D GeoVEs and their objects. It is based on the decomposition of 3D user interaction functionality into a set of services that can be flexibly combined to build automated, assisting, and application specific 3D user-interaction tools, which fit into service-oriented architectures of GIS and IT solutions based on spatial data infrastructures (*SDI*).

We discuss 3D camera control, categories of 3D camera tasks, and derive a collection of general-purpose 3D interaction services. We also explain how to efficiently compose these services and discuss their impact on the architecture of service-based visualization systems. Furthermore, we show by example how to apply this concept and to decompose a specific camera control technique into a set of services.

The remainder of this paper is organized as follows: Section 2 provides an introduction to service-based 3D geovisualization and 3D camera control in 3D GeoVEs as well as related work. Section 3 presents our concept for the decomposition of 3D camera control functionalities and their provisioning as services. Section 4 gives an example of a distributed camera control system that is based on the described services. Section 5 provides a discussion of the properties of such systems. Section 6 gives conclusions and an outlook.

2 Basics and Related Work

In this paper we build onto research in the area of service-based geovisualization and user interaction in virtual environments. Service-based geodata provisioning, processing and visualization have been standardized in recent years and systems implementing this paradigm are continuously evolving. In the following we provide an introduction to service-based 3D geovisualization and provide related work in the area of 3D camera control.

2.1 Service-Based 3D Geovisualization

The interoperability of systems and applications dealing with geodata is a central issue to build systems out of interoperable software components for geodata access, processing, and visualization. Beside a common understanding on information models (Bishr 1998), definitions of service interfaces are necessary. The Open Geospatial Consortium (OGC) defines a set of standardized services, models, and formats for geodata encoding and processing. For example, a Web Feature

Service (WFS) (Panagiotis and Vretanos 2010) can provide geodata, encoded in the Geography Markup Language (GML) (Portele 2007) or City Geography Markup Language (CityGML) (Gröger et al. 2008), and processed by a Web Processing Service (WPS) (Schut 2007).

For geovisualization processes a general portrayal model is provided by the OGC that describes three principle approaches for distributing the tasks of the general visualization pipeline between portrayal services and consuming applications (Altmaier and Kolbe 2003; Haber and McNabb 1990). While the OGC Web Map Service (WMS), providing map-like representations of 2D geodata, is widely adapted and used, 3D geovisualization services have not been elaborated to a similar degree. Several approaches for 3D portrayal have been presented (Basanow et al. 2008) and are currently discussed as standard proposal in the context of the OGC (Schilling and Kolbe 2010; Hagedorn 2010). These approaches differ in the type of data that is exchanged between client and service: Either filtered feature data, graphical representations (display elements), or rendered images are transmitted. Each type of data is generated by one specific OGC service:

- A WFS provides *feature* data, encoded in standardized formats, to a service consumer. This data can be processed at the client side; for visualization a *thick client* has to derive graphical representations and to perform the rendering.
- A Web 3D Service (W3DS) provides *display elements* to a service consumer (e.g. X3D or KML, organized as 3D scene graph) (Schilling and TH 2010; Altmaier and Kolbe 2003). This representation includes, e.g. geometry information and texture data. For visualization, a *medium client* needs to be able to process and render this graphics data.
- A Web View Service (WVS) provides *images* of a 3D scene to a potentially *thin client* (Hagedorn et al. 2009). In the simplest case, a client displays finally rendered images to a user. More advanced clients may also allow for more interactive visualizations using a WVS.

These segmentations lead to different requirements regarding 3D rendering capabilities of the portrayal services and corresponding client applications and to different types of interaction techniques that can be implemented within client applications.

2.2 Camera Control for 3D GeoVEs

On a technical level, the 3D camera control process generally includes (a) recognizing navigation intentions, (b) deriving path information, and (c) adjusting the visualization. Users of a 3D GeoVE express their navigation intentions by inputs provided to a client application, such as pressing UI controls, selecting objects in the scene, or sketching paths or gestures (Hagedorn et al. 2009). User input is evaluated and camera animations, i.e., camera positions alongside with camera orientations, are derived.

2.2.1 Assisting Camera Control Techniques

3D camera control in a virtual environment is a complex task, especially for non-expert users. Therefore, techniques for camera control in virtual environments were presented that assist users to explore 3D space by avoiding confusing or disorienting viewing situations (Buchholz et al. 2005). Task-oriented camera techniques generate camera paths with respect to a high-level navigation intention, such as “go to the closest landmark”.

A virtual camera’s behavior can depend on the semantics of the underlying model data of the 3D GeoVE (Döllner et al. 2005). Such semantics-based camera interaction techniques need client-side data and computing capabilities to perform camera path computations. Due to network and computational limitations, it is hard to make such capabilities available on thin clients or for large datasets.

2.2.2 Camera Control in Service-Based 3D GeoVEs

For distributed applications using thick and medium clients (as described) the rendering stage of the visualization pipeline is implemented on client-side, so the necessary data, such as model geometry or points of interest, is generally available. Therefore various camera-control techniques can be implemented within such applications, while thin clients need service-side support for reaching corresponding results since there is usually only a very limited set of client-side information about the geometry or topology of the 3D environment.

For example, a W3DS client, running on a machine with high processing capabilities and high speed connection to the W3DS server, could provide highly interactive real-time visualization and camera control, based on the retrieved 3D graphics data.

In contrast, a WVS provides multi-layer images, including not only color images but also, e.g. depth information per pixel, which allows for implementing (a) clients that only display images and provide only a step-by-step navigation as well as (b) more complex clients that reconstruct the virtual environment from information contained in such images and could even provide real-time navigation.

The complexity of camera-control techniques achievable for these client classes differs: Thick clients can easily consider semantic information from underlying geodata, which is not per se available through a W3DS or WVS. However, each of these 3D portrayal approaches could benefit from providing camera-control capabilities as distributed, reusable resources.

2.3 Challenges for Camera Control in 3D GeoVEs

This paper is motivated by the goal to implement interactive 3D GeoVEs on thin clients. Compared to desktop-based, thick client 3D GeoVEs, thin client 3D GeoVEs face several challenges regarding network capabilities as well as device constraints such as computing capacity, presentation, and interaction issues.

For distributed systems implementing camera-control functionality, various requirements need to be considered for achieving effective 3D camera control for 3D GeoVEs:

- Visualization and interaction should be decoupled to facilitate reuse of camera control functionalities as independent building blocks of 3D geovisualization systems.
- The separation of camera-control functionalities should support the implementation as services, but also as integrated part of a client application. So the decision of the location of network boundaries in concrete system architectures can be made per client application.
- Feedback (about available and pending camera movements, as well as system state information) should be provided by a distributed system for 3D camera control.
- A distributed system for 3D camera control should be designed to deal with limitations of wireless communication networks in connection with mobile clients (e.g. connection loss, latency times, available bandwidth etc.).
- The system architecture should not be restricted to a special type of input. Especially mobile devices provide more than one sensor that can serve as user input device. For example, device location, orientation, speed or other data delivered by device sensors could influence the way camera control has to be performed, e.g. to support building a relation between a user's actual position and the position and orientation inside the 3D GeoVE.

2.4 Further Work

Döllner et al. (2005) present an approach for interactive visualization of 3D GeoVEs on mobile devices using server-generated video streams. A user expresses his/her navigation intention by sketches instead of specifying the parameters for and steering the virtual camera explicitly. Sketch data is transmitted to a server, which interprets the input data, depending on the semantics of underlying objects, and computes a resulting camera path. A camera path animation is rendered as video and streamed to the requesting client. This way, only minimal demands are put to the mobile device. Our approach for interactive camera control introduces a more general and more flexible model of integration of 3D camera control into service-based 3D geovisualization systems. This allows for a larger set of input and output methods and decoupling camera control functionality from visualization functionality where possible.

Nurminen et al. (Nurminen and Helin 2005; Nurminen 2008) introduce a mobile application, which heavily uses rendering and transmission optimizations for 3D city models. They provide an interactive 3D GeoVE that integrates dynamic data, provided by remote servers, into the visualization. Geodata is preprocessed at server side optimized and transmitted for rendering. Camera control in the virtual environment is implemented completely client-side on the mobile device. However, for devices with limited input capabilities a higher level camera control could improve the usability of such 3D map applications. Due to the iterative transmission of model data to the mobile client depending on the camera parameters, camera navigation techniques that use model semantics cannot operate in many situations due to the lack of data available on the user's mobile device. Efficient 3D camera control could be integrated more easily in such an environment using a service-based approach for the separation and distribution of camera control functionality.

Chen and Bowman (2009) advocate that design for 3D interaction techniques should be application domain specific. They propose to decompose the interaction tasks into subtasks that consist of universal interaction tasks (e.g. navigation, selection or manipulation). The subtasks are implemented by concrete interaction techniques. Here, Chen focuses more on the question how to design domain-specific interaction techniques. In contrast, the focus of this paper is more on system engineering. We describe how such techniques could be designed as components of a service-oriented system, which facilitates reuse of specifically designed camera interaction techniques wherever the specific application domains come into play.

3 Concept for a Service-Based 3D Camera Control System

Since network bandwidth and end-user hardware and software is very heterogeneous, the development of robust, compatible, and efficient applications that provide interactive access to 3D GeoVEs represents a complex software architecture problem. 3D geovisualization systems using thin clients can bypass such limitations by designing a software architecture that can cope with hardware and software limitations of end-user devices and platforms. With thin clients, only small parts of the overall geodata are available on client side and could therefore be considered for camera path computation. Thus, we propose to move major parts of functionality for 3D camera control away from client applications to services. This loosens the dependency of camera interaction techniques from specific client implementations. Service components can be run in a scalable, controlled server environment and can, therefore, be maintained and optimized more efficiently.

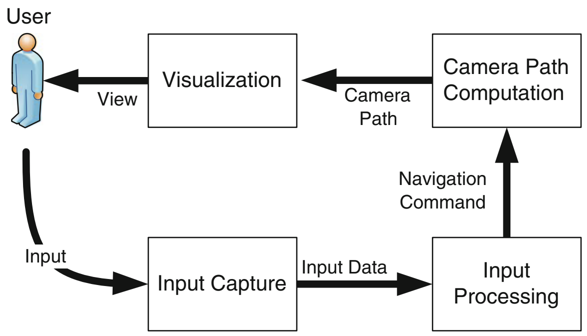

Fig. 1 Conceptional tasks of a 3D camera control process

Server-side access to geodata is usually more efficient due to lower network latencies and better performing hardware. Each of such service components is required to expose its capabilities, e.g. their operations, parameters, and effects, as well as their technical requirements. Capability information should also contain quality of service information, e.g. expected operation times such as minimum, maximum and average processing time for requests to allow raw latency estimations.

To structure the interaction cycle of service-based 3D visualization systems, we divide the process for 3D camera control into four core tasks (Fig. 1):

- *Input Capture*: Input provided by a user has to be captured and encoded in a way that allows for efficient evaluation.
- *Input Processing*: User input is preprocessed, e.g. converted, transformed, smoothed, or patterns are recognized and a navigation command is derived from the resulting data. This command is used to select the 3D camera service for camera path computation.
- *Camera Path Computation*: Camera positions and orientations, and transitions between them are computed. Specifications for camera paths are the result of this stage.
- *Visualization*: The computed camera specifications have to be applied for the client-side visualization of the 3D GeoVE. Visual or non visual (e.g. audible) feedback has to be generated and integrated in order to complete a 3D camera-control cycle.

While input capture has to be implemented by a client application, input processing, camera path computation and visualization can be implemented by one or multiple services. Figure 2 illustrates our concept for decomposing the core tasks for 3D camera control into functional independent *3D interaction services* and depicts their collaboration and the types of data exchanged between them. In the following, we present these major 3D interaction services: input preprocessing services, command recognition services, 3D camera services and composition services.

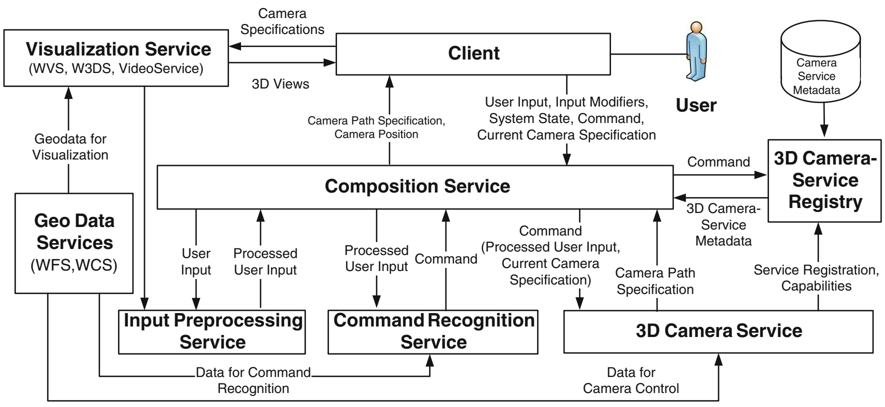
Camera Specifications
Visualization Service (WVS, W3DS, VideoService)
Client
User
3D Views
Camera Service Metadata
Camera Path Specification, Camera Position
User Input, Input Modifiers, System State, Command, Current Camera Specification
Geodata for Visualization
Command
3D Camera-Service Registry
Composition Service
3D Camera-Service Metadata
Geo Data Services (WFS,WCS)
User Input
Processed User Input
Processed User Input
Command
Command (Processed User Input, Current Camera Specification)
Camera Path Specification
Service Registration, Capabilities
Input Preprocessing Service
Command Recognition Service
3D Camera Service
Data for Command Recognition
Data for Camera Control

Fig. 2 Abstract component architecture and data flow of a geovisualization system using service-based 3D camera control

3.1 Input Capture

An end-user client-application must provide a description of user inputs. These could be, e.g. a higher-level navigation command (e.g. "look into a certain direction"), button events, or captured mouse-cursor respectively finger positions. Thus, a specification of user inputs is required that supports a variety of user inputs. Each data sample is annotated with timestamps to allow, e.g. for segmenting user input in time and space (e.g. series of sketches or device positions) and computing velocities of movements.

Additional information that is not directly originating from user actions could be required for service-based camera interaction. Client state information, such as input modifiers (e.g. pressed keys) or previous navigation commands, may affect the mapping from input parameters to navigation commands to be executed. Further, a user's current view (including, e.g. camera specification and visible objects) specifies the geospatial context of the user input, which may affect the evaluation of a user input.

3.2 Input Processing

Input processing is divided into two steps: input preprocessing and command recognition.

Depending on the type of input captured by the client, an *input preprocessing step* may be necessary (a) to improve the quality of the input, e.g. by filtering or smoothing and (b) to convert the input data to an analytic representation, e.g. recognizing geometry from a series of 2D input samples.

In a *command recognition step*, navigation commands are derived from the preprocessed input. These represent a more abstract description of a user's intention and include all the parameters required for their execution. We define three categories of navigation commands in respect of the camera-control task they describe (Hagedorn and Döllner 2008):

- *Direct Camera Manipulation*: A command directly influences the values of camera parameters, such as position or orientation vectors, which specify the current view. Commands like 'turn by 30°' or 'move 100 m into camera direction' are examples for such direct camera manipulation commands.
- *Path Oriented Navigation Command*: A command includes a path description that has to be followed by a camera path. The desired path has been computed in the input processing step or has been specified by the client directly (explicit or implicitly, e.g. by providing a target name).
- *Task Oriented Navigation Command*: A command contains a description of a task, which has to be fulfilled by a 3D camera service. Commands like "go to the next feature of class X" or "inspect feature X" belong to this command category.

To describe a command and to support command recognition, a generic, structured command schema is required that specifies, e.g. command parameters (types and possible values). The command recognition step can involve retrieval of additional geoinformation, e.g. from geodata or geovisualization services such as WFS or WVS.

Input preprocessing as well as command recognition are optional steps in the 3D camera control process. Simple camera-control tasks can be transmitted by a client as navigation command, e.g. "move one meter to north" for a stepwise camera control. Such commands may be handled directly by an appropriate 3D camera service.

The functionalities of the input preprocessing and command recognition steps are encapsulated by respective service types, *input preprocessing services* and *command recognition services*.

3.3 Camera Path Computation

Camera path computation is the core task for camera-control in 3D GeoVEs. *3D camera services* compute camera paths from navigation commands and their parameters. Conceptually, one 3D camera service implements one technique for camera path computation. This includes the generation of camera path components, e.g. camera positions, orientations, or other information that can be associated with a camera transition, e.g. textual annotations. Each of those can be computed by distinct functional components that apply specific algorithms and navigation constraints per path component. For example, a specific position component could determine camera positions only along a street network, while a specific orientation component could aim to keep nearby landmarks visible.

A 3D camera service may request additional geodata, e.g. from a WFS, W3DS or WVS to provide, e.g. a higher-level, semantics-based camera control or to fulfill constraints for camera parameters. To ensure a consistent behavior, these additional services have to be based on the same geodata as the visualization services themselves.

Based on a navigation command, a 3D camera service is selected using a 3D camera-service registry, which holds information about the available 3D camera service instances, the navigation commands they support, and additional metadata.

3.3.1 3D Camera Service

In order to be managed in a service registry and allowing consumers to bind correctly to their operations, 3D camera services are required to express general service information as well as functional and additional non-functional metadata (ISO 2003, 2005). 3D camera service capabilities should include metadata regarding the following aspects:

- *Service identification*: Type and version of the service, service description.
- *Camera control metadata*: Available path description formats, covered geospatial region, supported spatial reference systems, available navigation commands including command parameters.
- *Quality of service metadata*: Information such as expected computation time, result accuracies, available collision avoidance (e.g. guaranteed, best-effort, or no avoidance).
- *Application context*: Information regarding, e.g. user information, usage information, network conditions, and device properties.

3.3.2 Camera Path Specification

Camera paths, generated by 3D camera services, represent transitions from one set of camera parameters to another. Parameters required for the definition of a view of a 3D virtual environment are the camera position, its orientation in 3D space and its projection parameters. Additionally, a camera specification can provide annotations that can be used to enrich the 3D GeoVE with thematic information (e.g. distances or relevant objects) using overlays generated by specialized visualization services.

We distinguish two types of representations of camera paths (see Fig. 3): sampled and analytical representations:

Sampled representations of camera paths include of a series of camera-specification samples. Those can be created either using fixed or variable sample times. An adaptive sampling rate of camera specifications allows for more efficient representations of camera paths. This means more dense sampling for time periods where camera parameters change more rapidly, e.g. because of increased movement speed or sharp camera turns.

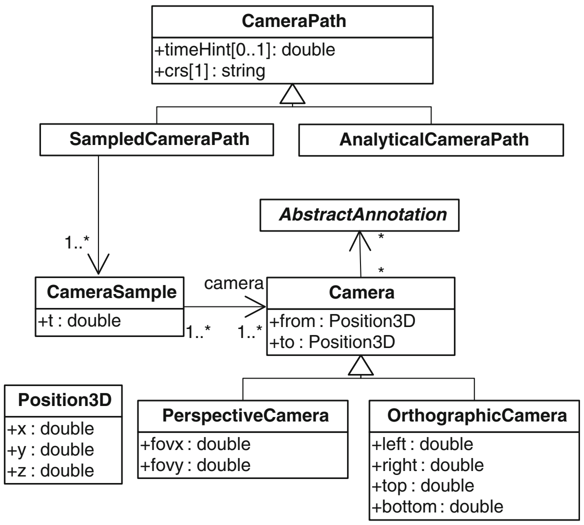

Fig. 3 Camera path specification. A `Camera Path` can be described either by samples of camera parameters or analytically using functional representations. Camera definitions can be associated with annotations containing additional information for user feedback

Analytical representations provide a separate function definition for each camera parameter to compute their values for an arbitrary point in time during a camera transition (Fig. 3). Each of those functions can be expressed as piecewise function, which eases the definition of camera paths by different kinds of functions per time slice, like Bézier curves, splines, linear or even constant functions. Furthermore, this enables the definition of story-board-like camera transitions. The overall time for the complete camera path animation is normalized. A camera path specification contains a recommended overall animation time, which would produce a comfortable camera motion.

A client may specify which type of path representation it requests. Analytical descriptions are more favorable for clients that are capable of interactive 3D rendering themselves, instead of using service-based image synthesis. In contrast, image-based clients may prefer sampled representations of a camera path, as it allows them to request images from portrayal services with a minimum of implementation effort and computational requirements.

A set of utility services can provide functionalities that can be used by several services of the distributed 3D camera control system. Functionality implemented by utility services may include, e.g. path manipulation (conversion, transformation, smoothing, composition), camera orientation computation, sketch recognition, and overlay creation (creation of image overlays as additional user feedback). Further, existing standards-based implementations of services, e.g. for geocoding of locations or conventional 2D routing using street networks may be used for camera path generation.

3.4 Visualization

The final task of the distributed, service-based 3D camera control process is to adjust the visualization of the 3D GeoVE according to a generated camera path, and to display the result to a user. Depending on the type and capabilities of a client application, several processing and visualization services could be involved for this:

- The retrieval of camera specifications from a camera path (e.g. by interpolating camera samples) could be implemented by a client application itself or could be provided by additional utility services.
- A client application that implements the 3D rendering itself needs to process the camera path specification, adjust the visualization accordingly, and generate new visual representations of the 3D GeoVE; utility services could support this process. Graphics data (e.g. X3D scene-graph) could be requested, e.g. from a W3DS.
- A client that is not capable of high-quality 3D rendering would incorporate a visualization service (e.g. a WVS) for creating visual representations of the 3D GeoVE, which could be served as image, set of images, or video to a client application.

Besides generating views of a 3D GeoVE, there are several possibilities for providing additional feedback about the camera control process. For example, textual or graphical annotations can be included in a camera path specification (Fig. 3).

3.5 Service Composition

Deploying the functionalities of the 3D camera control process as independent services enables assembling user input and camera control functionalities aligned to the requirements and capabilities of a specific client application as well as client device and network.

For a 3D camera control system, service composition includes (a) the composition of relevant input processing, command recognition, and 3D camera services and (b) the combination of multiple 3D camera services for reaching a specific camera dramaturgy.

3D camera services themselves could compose other services for implementing higher-level camera-control functionalities, e.g. task-oriented camera control.

4 Example

In this section we provide an example of a service-based 3D geovisualization system that supports 3D camera control for a mobile client applications. Images of a virtual 3D city model, are generated by 3D portrayal service (WVS). Figure 4

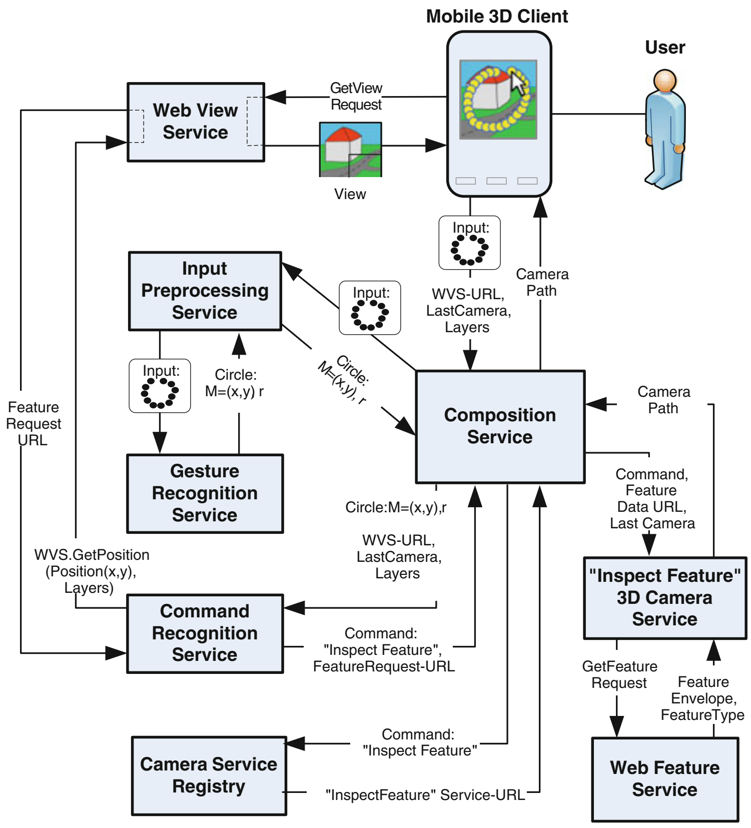
Mobile 3D Client
User
Web View Service
GetView Request
View
Input:
Input Preprocessing Service
Input:
WVS-URL, LastCamera, Layers
Camera Path
Input:
Circle: M=(x,y) r
Circle: M=(x,y), r
Composition Service
Camera Path
Feature Request URL
Gesture Recognition Service
Circle:M=(x,y),r
Command, Feature Data URL, Last Camera
WVS.GetPosition (Position(x,y), Layers)
WVS-URL, LastCamera, Layers
"Inspect Feature" 3D Camera Service
Command Recognition Service
Command: "Inspect Feature", FeatureRequest-URL
GetFeature Request
Feature Envelope, FeatureType
Camera Service Registry
Command: "Inspect Feature"
"InspectFeature" Service-URL
Web Feature Service

Fig. 4 Example of a distributed, service-based camera control system. The call sequence and concrete input and output data per service are depicted for one type of sketch-based navigation command

illustrates the services involved, the sequence of service calls and the data exchanged between them. As one purpose, the application allows users to inspect features of the 3D GeoVE. A user specifies a desired camera control task by performing gestures on a tangible display, which are captured as series of 2D positions. The gesture data, together with the client's current camera specification, a description of currently visible features (layers), and information about the visualization service used for image generation, is passed to the composition service, which manages the workflow for camera path generation.

Input data is handed over to an input preprocessing service, which performs gesture recognition. The gesture recognition results in geometric primitives that serve as basis for the command recognition. For example, performing a circle gesture around an object on the display could mean to inspect that specific feature. In this case, the gesture recognition service detects a circle primitive from the input positions.

The command recognition service matches the recognized geometry (circle) to the corresponding “inspect feature”-command. The object to be inspected is determined by requesting the object identifier for the center pixel of the circle from a WVS using its “GetFeatureInfo” operation and used as command parameter.

The 3D camera service registry is used to identify the 3D camera service that is able to process the “inspect feature“ command. The service is invoked with this command, the current camera parameters, and the reference to the feature data. The service resolves the reference and retrieves the feature data from a WFS. Depending, e.g. on the type or the size of an feature an appropriate technique for camera positioning and alignment can be chosen by the 3D camera service. For example, the HoverCam (Khan et al. 2005) might be better applicable for the exploration of buildings, while area-like features, such as green spaces, may be explored more efficiently through a flyover from a bird’s eye perspective.

The 3D camera service delivers the generated camera path specification to the calling client. To adjust its view of the 3D GeoVE, the client interprets the camera path, adjusts the virtual camera parameters accordingly, and requests new image data from the 3D portrayal service.

5 Discussion

The decomposition of interaction functionality into independent 3D interaction services requires to decide which kind of functionality to implement at the client side and which functionality to provide by services. On the one hand, a client-side implementation allows for application, user, and task specific interaction techniques, which are typically developed in detailed knowledge of a concrete use case of an application. On the other hand, reusable services permit to provide uniform interaction mechanisms for a number of different client applications and configurations. For example, the externalization of techniques for camera path computation can enable interaction techniques that need a global view of model data even for thin clients that are unable to deal with the amount and the complexity of massive 3D models.

The system provided as example in the previous section represents a quite complete implementation of the camera interaction functionality using external services. As central element for camera interaction, camera path computation, which includes determination of good views regarding various criteria, is most likely to be usefully implemented as external service. Input preprocessing services and command recognition services on the other hand are mostly useful for ultra thin, purely image-based clients (e.g. browser-based clients that displays rendered images, received from WVS instances). Such clients pass user input to an interaction service chain and receive a new camera specification to request new views from the portrayal service. This relieves clients of all the complexity of 3D computations. Though, such a client is easy to implement and has low hardware and software requirements.

Compared to thick client applications implementing user interaction functionality completely on client-side, the use of interaction services for camera control raises challenges concerning response times and bandwidth as well as their effect on the perceived responsiveness and interactivity of a consuming client application. However, these effects get attenuated by the reduction of the amount of data that has to be transferred over the network due to possible server-side preprocessing and aggregation of complex data needed for the computation of camera paths. Furthermore, 3D camera services can use precomputed camera positions or paths as well as caching strategies to reduce server-side processing, communication effort, and response times.

Dynamic geodata, relevant for the computation of camera paths, can be integrated efficiently by 3D interaction services, since the service encapsulates retrieval and evaluation of data at a single point in the system. Again, clients are relieved from data access and processing. For example, a 3D camera service could integrate sensor or other live data, e.g. extreme temperatures inside a building indicating a possible fire, to derive situation dependent camera paths. In general, the decoupling of the complexity of model geodata, model management, access and usage from camera computations enables to develop general purpose client applications. These implement only a set of core visualization and interaction functionalities that are relatively domain independent. This reduces the complexity of applications that are deployed on end-user devices and platforms and, therefore, raises the compatibility of such application.

6 Conclusions and Outlook

In this paper, we presented a concept for a distributed 3D camera control system for 3D GeoVEs, which integrates into existing service-based geovisualization approaches. This concept provides categories of services and their interconnection so that functionalities for 3D camera control can be decomposed into and deployed as independent services. Principal service classes for input processing, camera path computation, and 3D visualization are introduced; relevant data such as 3D camera paths and service metadata is modeled. By an example, we demonstrate how to compose these services to support the exploration of a mobile 3D GeoVE.

With our concept, 3D camera control functionalities become available through the Internet as distributed resources rather than being hard-coded in specific client applications. Services for input processing and 3D camera control form major building blocks for the implementation of interactive service-based 3D GeoVEs. Using these services, client applications can be relieved from complex computation tasks (e.g. for generating high-quality 3D camera positions and orientations) or implementing adapters to services providing geodata relevant for computations (e.g. WCS or WFS); they only need to interact with the distributed camera control system. Thus, 3D camera control functionalities become available even for thin clients, e.g. running on mobile phones or in web-browser environments.

Our approach enables to flexibly select from alternative 3D camera services for adapting the interaction process to specific application requirements (e.g. user tasks), device properties (e.g. input devices), and network capabilities (e.g. latencies or network bandwidth). Additionally, the proposed interaction services can be reused for achieving a consistent system behavior of the same quality across various applications. Service reuse also could enable a faster development of service-based 3D geovisualization systems and client applications. The approach allows for systematically accessing additional geoinformation provided as distributed services and using this information for 3D camera control (e.g. dynamic sensor data for camera path computation). Thin client applications for 3D geovisualization will become more popular in future due to increasing computing and 3D rendering capabilities of mobile devices (flanked by developments towards browser-based 3D rendering) and their increasing distribution. Such applications would directly benefit from a distributed, service-based 3D camera control system.

References

Altmaier A, Kolbe TH (2003) Applications and solutions for interoperable 3d geo-visualization. In: Fritsch D (ed) Proceedings of the photogrammetric week 2003. Wichmann, Stuttgart, pp 251–267

Basanow J, Neis P, Neubauer S, Schilling A, Zipf A (2008) Towards 3D spatial data infrastructures (3D-SDI) based on open standards–experiences, results and future issues. In: Advances in 3D geoinformation systems, Springer, Berlin, Lecture notes in geoinformation and cartography, pp 65–86. http://www.springerlink.com/content/u45062mr4hh54547

Bishr YA (1998) Overcoming the semantic and other barriers to gis interoperability. Int J Geograph Inf Sci 12(4):299–314

Buchholz H, Bohnet J, Döllner J (2005) Smart and physically-based navigation in 3D geovirtual environments. In: 9th international conference on information visualisation (IV'05), IEEE, pp 629–635.

Chen J, Bowman D (2009) Domain-specific design of 3D interaction techniques: an approach for designing useful virtual environment applications. Presence: Teleoper Virtual Environ 18(5):370–386.

Döllner J, Hagedorn B, Schmidt S (2005) An approach towards semantics-based navigation in 3D city models on mobile devices. In: Proceedings of the 3rd symposium on LBS& teleCartography, Springer, Vienna.

Gröger G, Kolbe T, Nagel C, Häfele K (2008) OpenGIS city Geography Markup Language (CityGML) encoding standard version 1.0.0. http://www.opengeospatial.org/standards/citygml

Haber RB, McNabb DA (1990) Visualization in scientific computing, IEEE Computer Society Press, chap visualization idioms: a conceptual model for scientific visualization systems, pp 74–93

Hagedorn B (2010) OGC web view service. In: OGC discussion paper

Hagedorn B, Döllner J (2008) Sketch-based navigation in 3D virtual environments. In: Proceedings of the 9th international symposium on smart graphics. Lecture notes in computer science, vol 5166. Springer, Berlin, pp 239–246.

Hagedorn B, Hildebrandt D, Döllner J (2009) Towards advanced and interactive web perspective view services. In: Developments in 3D geo-information sciences, Springer, Berlin, pp 33–51

Hildebrandt D, Klimke J, Hagedorn B, Döllner J (2011) Service-oriented interactive 3d visualization of massive 3d city models on thin clients. In: 2nd international conference on computing for geospatial research and application cOMGeo 2011.

ISO (2003) ISO 19115. Geographic information–metadata, international standard.

ISO (2005) ISO 19119. Geographic information–services, international standard.

Khan A, Komalo B, Stam J, Fitzmaurice G, Kurtenbach G (2005) HoverCam: interactive 3D navigation for proximal object inspection. In: Proceedings of the 2005 symposium on interactive 3D graphics and games, ACM, vol 1, pp 73–80

Nurminen A (2008) Mobile 3D city maps. IEEE Comp Graph Appl 28(4):20–31

Nurminen A, Helin V (2005) Technical challenges in mobile real-time 3D city maps with dynamic content. In: Kokol P (ed) Software engineering

Panagiotis P, Vretanos A (2010) OGC web feature service implementation specification. http://www.opengeospatial.org/standards/wfs

Papazoglou MP, Traverso P, Dustdar S, Leymann F (2007) Service-oriented computing: state of the art and research challenges. Computer 40(11):38–45. doi:10.1109/MC.2007.400. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4385255

Portele C (2007) OpenGIS geography markup language (GML) encoding standard. http://www.opengeospatial.org/standards/gml

Schilling A, Kolbe TH (2010) Draft for candidate openGIS web 3D service interface standard. http://portal.opengeospatial.org/files/?artifact_id=36390

Schut P (2007) OGC Web Processing Service. http://www.opengeospatial.org/standards/wps

Tan D, Robertson G, Czerwinski M (2001) Exploring 3D navigation: combining speed-coupled flying with orbiting. In: Proceedings of the SIGCHI conference on human factors in computing systems. ACM New York, pp 418–425