

Automated Generation of Operations Level Construction Animations in Outdoor Augmented Reality

Amir H. Behzadan, A.M.ASCE¹; and Vineet R. Kamat, M.ASCE²

Abstract: Three-dimensional (3D) visualization is an effective tool for communicating, verifying, and validating the results of a simulated operation. Traditional visualization tools used for this purpose are typically based on the paradigm of virtual reality. Augmented reality (AR) is a relatively newer visualization paradigm whose engineering applications have been explored by a limited number of researchers. In this paper, the problem of generating smooth and continuous AR animations from the results of running discrete event simulation models and a general purpose methodology to overcome this challenge are discussed. The structure of an AR animation authoring language developed by the writers to create a logical link between a running simulation model and its corresponding 3D visualization in AR is described. In order to validate the functionality and effectiveness of the designed methods and animation language, an AR-based visualization application was developed and the designed algorithms were successfully tested using different simulation scenarios of varying visual and operational complexity.

DOI: 10.1061/(ASCE)0887-3801(2009)23:6(405)

CE Database subject headings: Construction sites; Graphic methods; Computer applications; Simulation; Computer graphics; Automation.

Introduction

Operation planning is a critical component of managing and controlling the different aspects of an ongoing construction project. A comprehensive operational level work plan and corresponding site layout that provides easy accessibility to different locations of interest are key to significant savings in project time and costs, and at the same time results in reduction of several unwanted resource and physical space conflicts (Halpin and Riggs 1992). Working in a well-organized operational environment with minimum amount of time spent on resolving conflicts and doing tasks not directly related to the scope of the project is an important factor in meeting the project schedule. This is a major incentive for using computer applications to model, simulate, and visualize operations beforehand in order to detect any potential project-related events that may cause unexpected delays or conflicts during the course of the real operations, and plan ahead of time to avoid such scenarios during actual construction (Rojas 2000). For a relatively small operation, reviewing, interpreting, verifying, and validating the results of a simulation model can be done manually using statistical data, flowcharts, flow diagrams, and other numerical tools. However, as the size of the operation increases and with the introduction of more resources and activities

within the operation, automating the process of communicating the results of a simulation model for verification and validation purposes turns into a crucial need that has to be done in a timely and effective manner. One of the effective methods of verifying and validating the results of a simulation model is to visualize the flow of activities in a chronologically and spatially accurate manner. This is typically done in a fully computer generated environment referred to as virtual reality (VR) (Kamat 2003).

Several researchers have recently focused on the applications of VR in verifying and validating simulated operations (Op Den Bosch 1994; Barnes 1997; Bishop and Balci 1990; Rohrer 2000; Rohrer and McGregor 2002; Kamat 2003). Although VR provides an environment in which a computer generated representation of a simulated operation can be viewed and studied, there are a number of disadvantages to its application. The significant amount of time and effort invested in computer-aided design (CAD) model engineering [i.e., creating, rendering, and managing three-dimensional (3D) CAD models of the simulation entities] is a major challenge in VR, especially as the size and complexity of the operation grows (Brooks 1999). Further, as there is no notion of the real world in a VR-based visualization, the observer of the VR scene has a relatively low level of interaction with, and involvement in the scenes that are usually defined by a limited maneuvering area (e.g., immersive VR laboratory rooms) or hand motions (e.g., using a pinch glove).

A relatively newer visualization paradigm called augmented reality (AR) is based on the idea of creating a mixed environment of virtual and real objects that coexist in a shared space (Behzadan and Kamat 2005). Unlike VR, the application of AR as a general purpose visualization technique in many scientific and engineering fields is very promising as it can potentially provide a visual environment in which the observer of the scene can completely interact with both real and virtual objects. In particular, the introduction of real existing objects into the visualization can potentially lead to more thoughtful insights. It can also lead to a reduction of time and effort otherwise invested in CAD model

¹Ph.D. Candidate, Dept. of Civil and Environmental Engineering, Univ. of Michigan, 2340 G.G. Brown, 2350 Hayward, Ann Arbor, MI 48109. E-mail: abehzada@umich.edu

²Assistant Professor, Dept. of Civil and Environmental Engineering, Univ. of Michigan, 2340 G.G. Brown, 2350 Hayward, Ann Arbor, MI 48109. E-mail: vkamat@umich.edu

Note. This manuscript was submitted on October 2, 2007; approved on March 11, 2008; published online on October 15, 2009. Discussion period open until April 1, 2010; separate discussions must be submitted for individual papers. This paper is part of the *Journal of Computing in Civil Engineering*, Vol. 23, No. 6, November 1, 2009. ©ASCE, ISSN 0887-3801/2009/6-405-417/\$25.00.

engineering. However, relative to VR, AR-based visualization applications are still in their early stages of development. Although some disciplines (e.g., medical, automotive industry, and military) have been using AR as a frontline technology to overcome visualization challenges in their domain (Feiner et al. 1997; Thomas et al. 2000; Barfield and Caudell 2001; Gleue and Dähne 2001; Livingston et al. 2002; Suthau et al. 2002; Brown et al. 2003; Piekarski and Thomas 2003), work still needs to be done to develop functional and reliable AR-based visualization tools that can be effectively used in areas, such as construction and other engineering disciplines.

The presented research investigates and introduces the applicability of AR-based visualization for animating dynamic operations modeled and simulated using discrete-event simulation (DES). This paper presents the methodology and design of AR-based algorithms, and a powerful animation authoring language together with their practical implementation inside a mobile AR visualization tool. The designed animation language provides a convenient method to automatically author animations of any length and complexity at the operations level of detail using an external software process such as a running DES model. The results of the research have been validated by animating several simulated construction operations in outdoor AR.

Current State of Knowledge

Schematic visualization techniques such as those provided in *EZStroke* (Martinez 2001) and *ABC* (Shi and Zhang 1999) can produce useful results in terms of highlighting important events that occur within a simulation. However, they provide no sense of realistic resource motions, nor can they adequately represent the processes occurring within activities. There is also no notion of physical space and therefore cases dealing with spatial conflicts or congestion can not be studied in such visualizations. This has been a major incentive to introduce the potential of 3D visualization in validating and verifying the simulation models. Smooth and continuous animation can provide more accurate representation of a simulated operation (Kamat and Martinez 2001). It can also help overcome the inadequacies of traditional schematic visualization techniques. Most of such studies, however, have been focused on the application of VR and virtual interactive environments. Limited research has been conducted to investigate the applicability of AR for such purpose.

The application of visualization techniques for planning, analysis, and design in construction and civil engineering is relatively new compared to the sizeable amount of AR-related research conducted for diverse applications in fields such as manufacturing, medical operations, military, and gaming. For example, Webster et al. (1996) presented a system that shows locations of columns behind finished walls, and rebar inside columns. They also presented an AR system to guide workers through the assembly of a space frame (Webster et al. 1996). Roberts et al. (2002) used AR to overlay locations of subsurface utility lines onto real world views. These applications have been designed to demonstrate the potential of AR in helping maintenance workers avoid buried infrastructure and structural elements. Hammad et al. (2004) augmented contextual information on real views of bridges to help inspectors conduct inspections more effectively. Kamat and El-Tawil (2007) used AR to study the extent of horizontal displacements sustained by structural elements due to extreme loading conditions. Behzadan and Kamat (2007) studied the applicability of four-dimensional CAD construction (McKinney

et al. 1996) in AR by performing several AR visualization tests using time tagged CAD models of a building structure. A common characteristic of the existing AR-related research is that most pieces of work have led to a final product designed for a specific purpose (i.e., application). Several of these products are only capable of handling static views or snapshots of an augmented scene manually created to serve as the basis for further visual analysis. A key limitation in existing knowledge has been an automated process to generate dynamic visualizations of simulated operations of arbitrary length and complexity. To the writers' best knowledge, none of the conducted AR-related research has hitherto focused on the use of an external software process such as a DES model to automatically create and animate simulated operations in 3D outdoor AR.

Main Contribution

Construction operations usually include a large number of tasks involving different resources (i.e., equipment, labor, material) based on complex interactions. As a result, illustrating the complex dynamics of a typical construction site solely based on the discrete information obtained from a DES model is a very challenging task. This led the writers to the idea of designing an animation language that would enable a running simulation model to automatically generate a syntactically accurate trace file representing the operations being simulated. The generated trace file consists of chronologically ordered tasks performed during the simulation and is used as the underlying script to create the AR-based animation of the operation. The primary contribution of the presented research is an AR animation authoring language that enables modelers to create augmented reality animations of construction operations simulated in a DES tool.

In order to visualize a construction operation, the required components include, but are not limited to, the facility under construction, equipment, personnel, materials, and temporary structures, as well as their possible movements, transformations, and interactions. All these components have to be accurately depicted within the augmented scene. In order to depict smooth motion, visual elements must be shown at the right position and orientation several times per second. Due to the amount of detail and precision involved, accurate visualization of construction activities at this level has always been a challenging prospect (Kamat and Martinez 2003). Further, features enabling user interactivity within the augmented scene are key factors that have to be integrated into an AR application. In a mobile visualization application, the user should be given the ability to walk through an animation in real time, adjust, modify, or completely change the underlying operational logic of a set of related tasks, properties of a certain resource, or even the overall simulation scenario in order to study the effects of different decisions and outcomes in one experiment. What distinguishes the developed methods in this research from previous work is that in the presented methodology, the user's position inside the 3D augmented world determines what parts of the animation should be displayed.

The constantly updating augmented space only displays what is visible to a user at a particular location. This means that there are no previously defined viewpoints that restrict the location from where an animation can be observed. Instead, the user can freely change position, and view direction (i.e., head orientation), as the position and/or orientation of the virtual contents inside the augmented space are continually updated. The procedure of constantly adjusting the position and orientation of the virtual objects

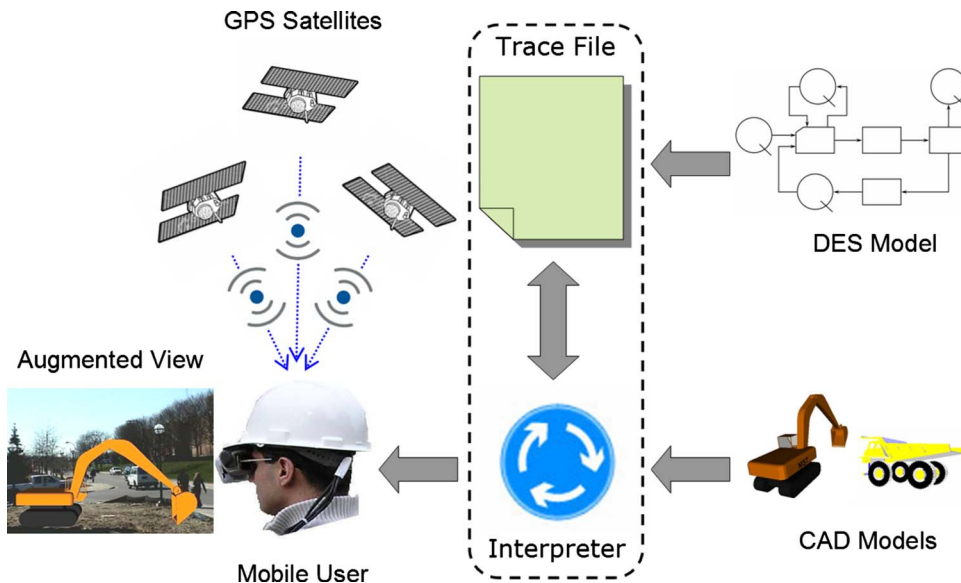


Fig. 1. Data flow from DES, CAD models, and GPS to the application and display

overlaid on top of the real world is often referred to as registration. To achieve precise registration of virtual objects and create augmented scenes of simulated operations, a fully functional global positioning system-based (GPS-based) tracking mechanism has been developed by the writers. The details of this mechanism have been described in Behzadan et al. (2008).

Technical Approach

The application of DES tools to simulate construction processes has been widely investigated (Tucker et al. 1998; Martinez and Ioannou 1999). DES is a powerful objective function evaluator that is well suited for the design of construction operations. It runs on the concept of separate activities linked together based on their resource needs and precedence logic. DES as applied to construction operations planning and analysis entails the creation of models that represent how construction operations will be performed. These models consider the different resources that are required to carry out the construction operations, the rules under which the different tasks that compose the operations are performed, the managerial decisions made during the operations and the stochastic nature of events. Once the models are created, the modeled operations can be simulated in the computer and the statistical measures of performance for the operations can be studied (Kamat and Martinez 2002).

In order to create smooth animations from the results of a DES model, events that mark the beginning and end of discrete activities have to be interpreted and communicated to the visualization tool in a continuous chronological order. To achieve this objective, using an authoring language that automatically creates time stamped events in the form of sequential statements written to an animation trace file is a very viable approach (Kamat 2003). The generated animation trace file can be then fed into the visualization application to link each simulation entity to a CAD object, and dynamically manipulate CAD objects based on the contents of the trace file. In addition, to cause objects under the control of such simulations to be aware of, and react to the user motions in the augmented scene, the augmented environment implementation must communicate bidirectionally, on a continuous basis, and at

high speed with location tracking devices. This communication is very critical in obtaining real time position and orientation as the user walks inside the animation to observe the scene from different perspectives. The data acquired from tracking devices are vital parts of the application in order to display the final animated output. Fig. 1 shows the approach taken in the presented research to use the result of a running DES model to generate the animation trace file of an AR augmentation, upload the required CAD objects inside the user's augmented view, and update the contents of the AR scene based on the user's latest position obtained from the tracking devices.

As shown in the schema presented in Fig. 1, the resulting AR animation trace file is sequentially interpreted line by line and appropriate CAD objects are uploaded and/or transformed as each line of the trace file is executed. In addition, the positional data coming through the tracking devices connected to the user are simultaneously extracted, integrated, and used to generate an augmented viewing frustum with the user's eye at the center of the projection. Inside this frustum, the contents of the virtual world (i.e., CAD objects of construction entities) are superimposed on top of live video streams of the surrounding world as captured by a video camera installed in front of the user's eye. The result is a dynamic augmented view of the ongoing construction operation, which is fully responsive to position and orientation changes of the user in 3D space (Behzadan and Kamat 2007). The animation is shown to the user through a head mounted display (HMD). The video camera, head orientation tracking device, and HMD are all installed on the user's hard hat. The user can walk freely on the site with minimum physical constraints and observe the animated scenes from different positions. The heart of the system is a laptop computer which is installed and secured inside a backpack. Other devices included in the backpack are a GPS receiver unit, and an external battery pack. A miniature keyboard and a touch pad are also connected to the laptop and carried by the user to provide full interaction capability during the course of the animation when there is no physical access to the laptop computer (Behzadan et al. 2008). Fig. 2 shows a profile of the user wearing the mobile backpack equipped with all necessary components.



Fig. 2. Profile of the user with the mobile backpack and registration devices

ARVISCOPE Animation Authoring Language

As noted earlier, an authoring language is a critical component of the AR application to extract simulation results from a running DES model. The writers designed a powerful, self-contained animation authoring language called ARVISCOPE (acronym for augmented reality visualization of simulated construction operations). ARVISCOPE is a high-level 3D animation authoring language that can allow an external software process (e.g., a running DES model) to author a dynamic visualization in outdoor AR. Sequential statements written in this language can describe a smooth and continuous operation of arbitrary length and complexity. The communicated statements (i.e., events) are interpreted by the visualization engine of the AR application. Appropriate data structures, algorithms, and routines are then invoked to manipulate CAD models and other 3D geometric primitives to present a smooth, accurate representation of the operations. Despite the fact that the ARVISCOPE authoring language is powerful enough to describe the complexities involved in a typical construction operation, the syntax of the language is not very complex. According to their functionality, ARVISCOPE language statements can be grouped into scene construction, dynamic, and control statements. These statements can be sequentially recorded into and interpreted from a text file referred to as the trace file in this paper. The trace file begins to be parsed as soon as the application starts, the individual statements are processed, and the graphical representation corresponding to the event in each line of the trace file is created and depicted inside

the augmented view simultaneously until the end of file is reached. During this process, the user can freely move in the animated augmented space.

Scene Construction Statements

Scene construction statements are designed to set up the animation environment and manage the initial and dynamic creation and destruction of simulation entities. This is done by referencing CAD models of relevant resources (e.g., equipment) in different graphical formats, creating instances of specific CAD objects, creating complex CAD metaobjects by assembling simple CAD objects into logical geometric hierarchies, and specifying the initial position and orientation of objects in the desired state on the construction site. This group also contains statements that are used to define routes (i.e., 3D trajectories) that entities may travel on when performing operations. Table 1 lists the scene construction statements of ARVISCOPE together with a brief explanation of their functionality.

Dynamic Statements

Dynamic statements constitute the core of the ARVISCOPE language. The group consists of several statements that can be used to dynamically manipulate instantiated scene objects to depict the performance of a smooth and continuous operation. Statements in this group describe dynamic geometric transformations of scene objects. These transformations change the position, orientation, and scale (i.e., size) of objects in the 3D augmented space to depict the accurate motion objects undergo when performing operations. The most important statements of this group are those describing single elemental motions that a construction resource undergoes during a specific operation. Examples of such statements are TRAVEL, ORIENT, and SIZE. A time-stamped sequence of an arbitrary number of such elemental motions can effectively describe a smooth, continuous 3D rendition of the pertinent construction operation. Table 2 lists the scene construction statements of ARVISCOPE together with a brief explanation of their functionality.

Control Statements

The primary control statement in ARVISCOPE is SIMTIME. It keeps track of the simulation clock as the animation is running. Every discrete event that is represented by a statement inside the

Table 1. List of ARVISCOPE Scene Construction Statements and Their Functionality

Statement syntax	Functionality
LOADMODEL <GroupName> <3DFileName>;	Assign a CAD file to a class of objects
ORIENTMODEL <GroupName> <Axis> <Degree>;	Change a CAD file orientation
CHANGEMODEL <ObjectName> <NewGroupName>;	Change the CAD file of an object
OBJECT <ObjectName> <GroupName>;	Create an instance of an object class
REMOVE <ObjectName>;	Remove an object from the scene
CONNECT <ChildName> <ParentName> <AtPoint>;	Create a child node to a parent node
STICK <ChildName> <ParentName> <AtPoint>;	Connect an object without change in size
DISCONNECT <ChildName> <ParentName>;	Detach a child from its parent node
ROUTE <RouteName> <Points_Array_XYZ>;	Define a 3D route for mobile objects
POSITION <ObjectName> AT <PointsXYZ>;	Place an object in the augmented view
ADJUST GROUP MODEL <GroupName ModelName> RGP FORCLR AFTCLR <Value>;	Set properties of a group or an object

Table 2. List of ARVISCOPE Dynamic Statements and Their Functionality

Statement syntax	Functionality
HRZORIENT <ObjectName> <YawValue>;	Change horizontal orientation
VRTORIENT <ObjectName> <PitchValue>;	Change vertical orientation
SIDEORIENT <ObjectName> <RollValue>;	Change side orientation
ORIENT <ObjectName> X Y Z <Value> <Duration>;	Change orientation by a certain amount
ORIENTTO <ObjectName> X Y Z <Value> <Duration>;	Change orientation to a target value
TRAVEL <ObjectName> <RouteName> <Duration>;	Move an object on a route for a certain duration
TRANSFER <ObjectName> <RouteName> <Speed>;	Move an object on a route at a certain speed
SHIFT <ObjectName> <Vector> <Duration>;	Move an object by a certain amount
SHIFTTO <ObjectName> <PointXYZ> <Duration>;	Move an object to a certain location
SIZE <ObjectName> <ScaleChange> <Duration>;	Change the scale of an object
SIZETO <ObjectName> <ScaleTarget> <Duration>;	Change the scale to a target value

trace file has a preceding SIMTIME statement that indicates the simulation time at which the event begins to take place.

Methodology for Creating an Animation Trace File

Fig. 3 shows the activity cycle diagram of an earth-moving operation in STROBOSCOPE format. STROBOSCOPE (Martinez 1996) is a programmable and extensible simulation system designed for modeling complex construction operations in detail and for the development of special-purpose simulation tools (Kamat and Martinez 2001). The operation shown in Fig. 3 consists of a load-haul-dump-return cycle within which different resources are either used (i.e., loader and hauler) or transferred (i.e., soil). As the focus of this paper is to describe the process of visualizing modeled operations in AR and not the process of modeling earth-moving operations, issues such as volume of the work, productivity, and necessary equipment and steps to perform the operation itself are excluded from the discussion. As noted earlier, an animation trace file is required to create augmented animations of simulated operations in ARVISCOPE. This file can be either created manually (for short animations) or automatically during a simulation run. Manual generation of an animation trace file is typically not practical except in the case of simple demonstrative examples of short animated duration. Automatic generation of a trace file is more recommended as it requires less time and produces more accurate results. Automatic generation of an ARVISCOPE animation trace file requires instrumenting a simulation

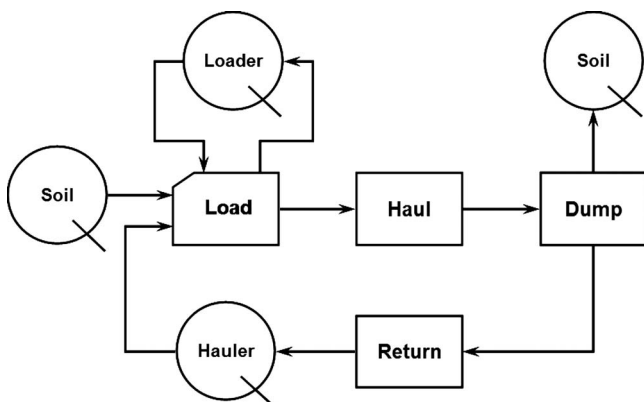


Fig. 3. Activity cycle diagram for earth-moving operation

model (i.e., including additional code in a simulation model). For example, Fig. 4 shows how two new lines are created inside the ARVISCOPE animation trace file of the earth-moving operation (Fig. 3) as a result of a statement added to the STROBOSCOPE model of the same operation. These two lines will be written to the trace file numerous times with different arguments (e.g., time tag, duration, object name, route name) depending on the specific instance of the activity taking place. The completed trace file will contain other lines of text that will be written out when other parts of the modeled operation take place. Thus, the time-ordered sequence of animation statements written out by all the activities in the model during a simulation run constitutes the trace file required to visualize the modeled operations in AR.

Fig. 5 shows the algorithm used in the ARVISCOPE application to read, extract, and interpret the contents of an animation trace file. As shown in Fig. 5, once the animation trace file is opened and the animation clock is started, the first available SIMTIME statement is read and the specified time argument is extracted. This argument is then compared to the continuously progressing animation time. If the time argument is less than or equal to the current animation time, the entire block of statements between the current and next SIMTIME statements are read and stored in an empty list. Otherwise, the application suspends the processing of the trace file until the above-mentioned condition

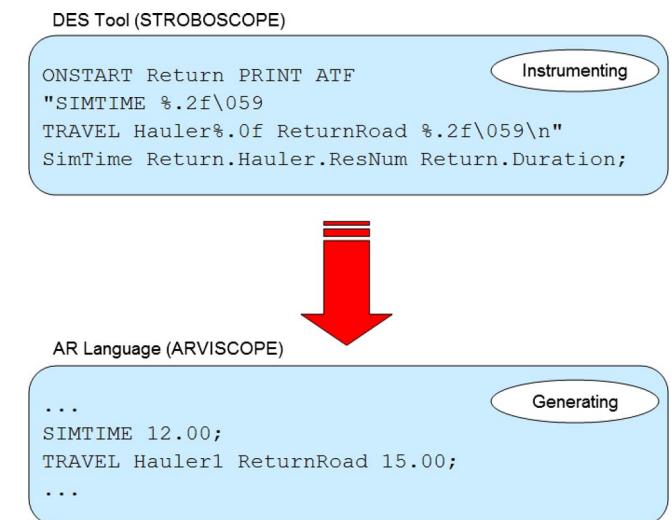


Fig. 4. Automatic generation of ARVISCOPE animation trace file by instrumenting a STROBOSCOPE simulation model

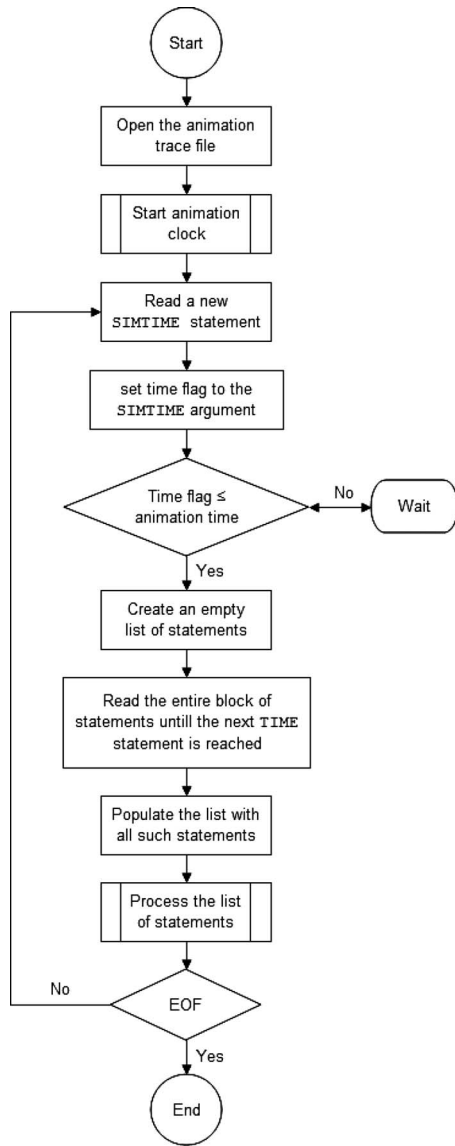


Fig. 5. Main processing loop of an animation trace file in ARVISCOPE

holds. At this point, each statement inside the list is separately processed and the contents of the augmented scene are accordingly updated. This process continues until the end of animation trace file is reached. Fig. 6 presents a small portion of the animation trace file of the earth-moving operation shown in Fig. 3 in the ARVISCOPE language. By referring to Tables 1 and 2, the result of processing each statement in Fig. 6 can be explained. First, a route called ReturnRoad is defined by specifying the beginning, ending, and two intermediate points in terms of global values of longitude, latitude, and altitude. The 3D models of a hauler and its bucket are then loaded. An instance of each of these models is then created and the bucket is attached to the hauler. This new metaobject, now called Hauler1, is then placed on the predefined route. Further processing of the trace file is suspended until simulation time 12 is reached. At this time, Hauler1 will start moving on the route ReturnRoad and this trip requires 15 units of simulation time to complete. During the animation, the ratio of simulated time to viewing time (also known as viewing ratio) is maintained at a constant value specified by the user at the beginning of the animation. For example, if this ratio is 3, the anima-

```

SIMTIME 0;
ROUTE ReturnRoad (-83.728180,42.287913,260.00)
(-83.727064,42.287469,260.00)
(-83.726099,42.288374,260.00)
(-83.726442,42.289152,260.00);

LOADMODEL Hauler Truck.lwo;
LOADMODEL Bucket Bucket.lwo;
OBJECT Hauler1 Hauler;
OBJECT Bucket1 Bucket;
CONNECT Bucket1 Hauler1 (-7,2.2,0);
POSITION Hauler1 ON ReturnRoad;

SIMTIME 12;
TRAVEL Hauler1 ReturnRoad 15;
  
```

Fig. 6. Portion of a sample ARVISCOPE animation trace file

tion will show Hauler1 at the beginning of route ReturnRoad for 4 seconds and then the truck starts traveling on the route for 5 seconds.

Language Design Issues

The writers had to address several basic object handling issues, simultaneously designing the syntax of the ARVISCOPE language to make it capable of describing complex construction scenes. A major challenge was the constant need inside the application to switch from global to local coordinate systems when updating the virtual contents of the augmented viewing frustum at each animation frame. In this research, the global coordinate system is defined by three major axes: X , which is parallel to the equator with its positive direction pointing toward the geographical east, Y , which is perpendicular to the earth surface with its positive location pointing upwards, and Z , which is parallel to the prime meridian with its positive direction pointing toward the geographical south. X , Y , and Z readings in the global coordinate system are used to determine the longitude, altitude, and latitude of the user as well as CAD objects in the global space. At the same time, each CAD object has its own unique local coordinate system. The definition of the three major axes in a local coordinate system varies between CAD objects and mainly is specified when the CAD model is created in a graphical software such as *AutoCAD* or *3DS*. However, to make sure that CAD objects are placed in the global coordinate system with guaranteed consistency, their local coordinate frame should be oriented in a way that it is exactly aligned with the global coordinate frame (e.g., local X axis is parallel to the global X axis).

Although objects can be placed in the scene by defining their global coordinates (i.e., longitude, latitude, and altitude), all transformations (i.e., translations, and rotations) have to be done relative to the user. As a result, the application must be able to keep track of the latest relative position and orientation of each CAD object inside the user's coordinate frame. As shown in Fig. 7, although the user and all CAD objects have their global position determined in the global coordinate system, CAD objects still have to be placed relative to the user's eyes. So, for each CAD object, an additional step has to be taken in order to calculate the relative distance between the user and that object and use this distance to construct or update the transformation matrix corresponding to that CAD object. Only after this matrix is calculated, the CAD object can be correctly placed inside the user's viewing frustum. As the user is assumed to be moving, all transformation calculations must be done in real time with the latest global position of CAD objects and the one for the user being important

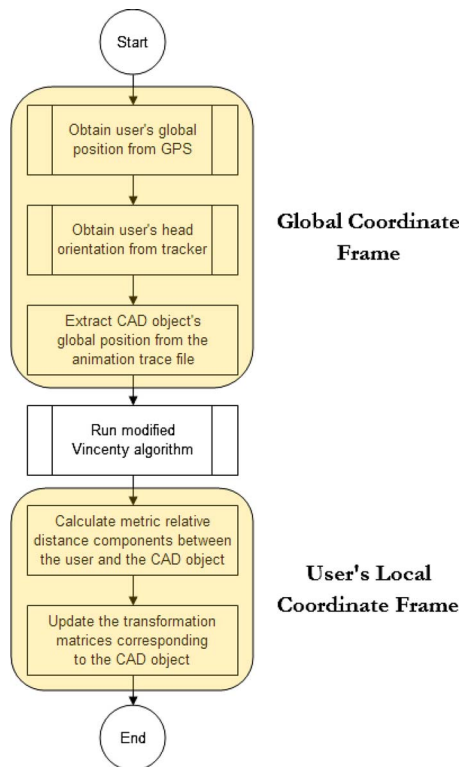


Fig. 7. Switching from global coordinate frame to user's local coordinate frame

input parameters to such calculations. As a result, switching from the global coordinate frame to the user's local coordinate frame and adapting appropriate conversion methods to achieve the expected result have been key challenges in designing the ARVISCOPE language statements.

The language statements are designed in a way that they can accept and handle positional values in both global and local coordinate frames as arguments when necessary. Two such statements are POSITION and ROUTE. As described earlier, the POSITION statement is used to place a static or a moving object in the augmented scene by defining its global coordinates (e.g., longitude, latitude, and altitude values). The ROUTE statement is used to define a 3D trajectory for moving objects. As shown in Table 1, both statements take global point coordinates as arguments. To improve the initial design of these two statements and make them more powerful and practical, they were further modified to also accept local point coordinates relative to a reference point with known global coordinates. This was important for constructing the animated scene especially due to the fact that when the number of global points increase, the computational time required to measure the longitude, latitude, and altitude of each point grows dramatically. For an animation to be a precise representation of a simulated system, it is necessary to have contingency in measuring longitude, latitude, and altitude values of different points in the scene. Sometimes, it is practically impossible to measure these values on the site. A good example is a point far beyond physical access (e.g., a point on a body of water) or a number of points with limited distance to be accurately measured with a GPS device (e.g., two CAD objects located a few centimeters from each other, whereas the GPS accuracy is in the order of meters). Another situation is the case in which a large number of CAD objects are to be placed in a scene and the only known pieces of data are the relative distances between them as

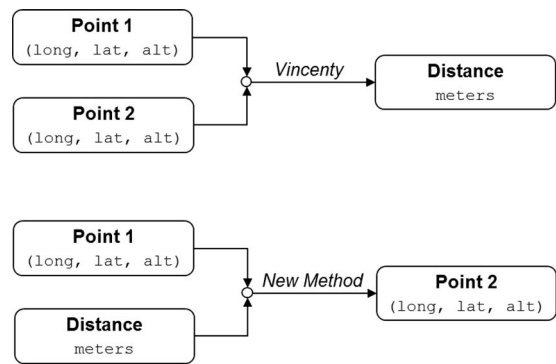
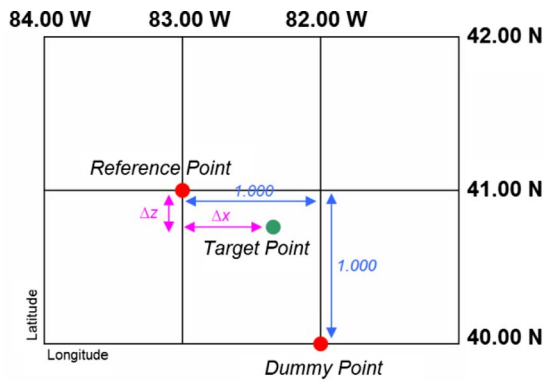


Fig. 8. Calculation of a global point coordinates in ARVISCOPE

opposed to their absolute longitude, latitude, and altitude values. As a common practice in drawing the site layout plans, position of important points are marked relative to the positions of a number of known surveying benchmarks. Sometimes the only points on the site with known longitude, latitude, and altitude values are benchmarks and the positional values of everything else is measured and calculated relative to these points. Although obtaining the global position values for every item on a construction site is a very time consuming and often impossible process, finding these values for only a few points and placing others relative to them appears to be a more reasonable approach.

Hence, a numerical algorithm was developed and implemented by the writers to calculate the global position of a point based on the same values of a reference point and the relative distance between the two points. This required establishing methods to convert local to global coordinate values. The initial methods to convert global coordinates to local values in the presented work were adapted from an algorithm originally introduced by Vincenty (1975) and presented in more details in Behzadan and Kamat (2007). Following this algorithm, the distance between any two given points on Earth is entirely a function of where they are located on the Earth's surface. This is due to the fact that the parallels (the rings around the Earth parallel to the Equator) become smaller as they approach the poles and the meridians (the rings around the Earth parallel to the Prime Meridian) converge at the poles. As a direct result, for every 1° change in longitude and latitude, the equivalent displacement in meters will be larger close to the Equator and smaller close to the two poles. These values range between 1.6 and 111 km depending on where the location of the point is on the surface of the Earth. Although an average of 96.6 km is recommended for approximation, a more precise way to convert changes in degrees into displacements in meters had to be used in ARVISCOPE to be able to create exact graphical representation of a construction site.

As the initial Vincenty algorithm uses numerical methods to minimize the distance error margin by trial and error, there is no inverse Vincenty method available to convert local values to global coordinates. As a result, the writers designed their own algorithm which internally uses parts of the original Vincenty algorithm in its calculations. Fig. 8 shows the difference in terms of input and output parameters between the original Vincenty algorithm and the developed method in this research. As shown in Fig. 8, the developed method calculates how many meters of displacement in $x(z)$ direction correspond to a 1° change in longitude (latitude) value. As discussed earlier, the answer totally depends on where the point is located on Earth. The points as defined and used in the calculations are shown in the planar view



```
DummyPoint.long = ReferencePoint.longitude - 1.000000
DummyPoint.lat = ReferencePoint.latitude - 1.000000
DummyPoint.alt = ReferencePoint.altitude

TargetPoint.long = ReferencePoint.long + [Δx ÷ (MIn1DegLong)]
TargetPoint.lat = ReferencePoint.lat - [Δz ÷ (MIn1DegLat)]
TargetPoint.alt = ReferencePoint.alt + Δy
```

Fig. 9. Relation between reference, dummy, and target points in planar view

of Fig. 9. In Fig. 9, the global position of the *Reference Point* and the relative distance between the *Target Point* and the reference point along the three major axes are known. The goal is to calculate the global position of the target point which can be a point on a route or a point on which a CAD object is to be placed. Knowing the global coordinates of the reference point and as shown in Fig. 9, a *Dummy Point* can be imagined which is exactly one unit (in longitudinal and latitudinal directions) away from the reference point.

Based on the coordinates of the reference and dummy points, the developed method can calculate how many meters of displacement correspond to a 1° change in longitude (i.e., *MIn1DegLong*), as well as a 1° change in latitude (i.e., *MIn1DegLat*) by making a number of internal calls to the modified Vincenty algorithm (Behzadan and Kamat 2007). With a good approximation (and assuming the reference point is close enough to the point of interest), the surface containing the reference and dummy points can be assumed planar and linear interpolation (or extrapolation) is used to calculate longitude, latitude, and altitude values of the target point. These steps are presented as pseudocode in Fig. 9. For sign compatibility, x and y values are added to the coordinates of the reference point, whereas the z value is subtracted to obtain the coordinates of the object. Adding this new functionality to the original syntax of the *POSITION* statement, there are three distinctive ways to place an object in the augmented scene in ARVISCOPE: (1) to place it on a point with known global position values; (2) to place it on a route with known point coordinates; and (3) to place it relative to a reference point with known global position values. Fig. 10 shows different syntax of *POSITION* and *ROUTE* statements in ARVISCOPE. As shown in Fig. 10, in order to distinguish between the three types of *POSITION* statement, three different keywords are used: *AT*, *ON*, *REL*. In addition, the two different types of *ROUTE* statements can be distinguished using the *REL* keyword. According to the definition of major axes in ARVISCOPE, Δx shows the relative distance between the object and the reference point along

```
POSITION objectname AT (longitude, latitude, altitude);
POSITION objectname ON RouteName;
POSITION objectname REL ReferencePoint (Δx, Δy, Δz);

ROUTE RouteName (Long1, Lat1, Alt1)... (Longn, Latn, Altn);
ROUTE RouteName REL ReferencePoint (Δx1, Δy1, Δz1)... (Δxn, Δyn, Δzn);
```

Fig. 10. Different syntax of *POSITION* and *ROUTE* statements

the longitudinal axis, Δy shows the relative distance between the object and the reference point along the altitudinal axis, and Δz shows the relative distance between the object and the reference point along the negative latitudinal axis. Recalling the trace file shown in Fig. 6, the route *HaulRoad* can now be expressed in terms of its end and mid points' coordinates defined relative to the coordinates of a known point (i.e., *BM*). The new definition is shown in Fig. 11.

Manipulating CAD Objects inside the Augmented View

A key factor in designing a mobile AR application is that the motion of objects as well as those of the user can change the transformation parameters of the scene elements. In order to keep real time track of the position, orientation, and scale at the CAD object level and create a dynamic animated scene, each CAD object is handled separately. The augmented view is an assembled scene containing all the individual CAD objects in a hierarchical structure. Objects are either defined at the root level (with no dependency on each other), or at child level of other objects (to form a metaobject). At each animation frame, the relative distance between the user and each of the CAD objects at the root level is calculated. The result is used inside a translation matrix to position each CAD object inside the augmented viewing frustum. Further, and as a direct result of the fact that the CAD objects are animating a simulated operation, even if the user's position is fixed, the position, orientation, and scale of each CAD object inside the augmented view can change if corresponding statements such as *TRAVEL*, *ORIENT*, and *SIZE* are invoked on that object. Fig. 12 shows how the contents of the user's viewing frustum can change based on either the user's or CAD objects' movements during the course of the animation.

As noted earlier, GPS measurements in form of longitude, latitude, and altitude values are used to measure the 3D position of the user as well as the CAD objects in this frame. As longitudinal and latitudinal values are expressed in degrees, it is necessary to adapt a method to convert these values to their metric equivalents before using them in transformation calculations. In order to do so, the writers have modified the original Vincenty method

```
SIMTIME 0;
LOADMODEL Reference BM.ac;
OBJECT BM Reference;
POSITION BM AT (-83.707000, 42.29560, 265.00);

ROUTE HaulRoad REL BM (20.000, 0, -5.500)
(112.125, 0, 44.100)
(191.675, 0, -56.450)
(163.400, 0, -142.900);
```

Fig. 11. Defining a route using relative coordinate values

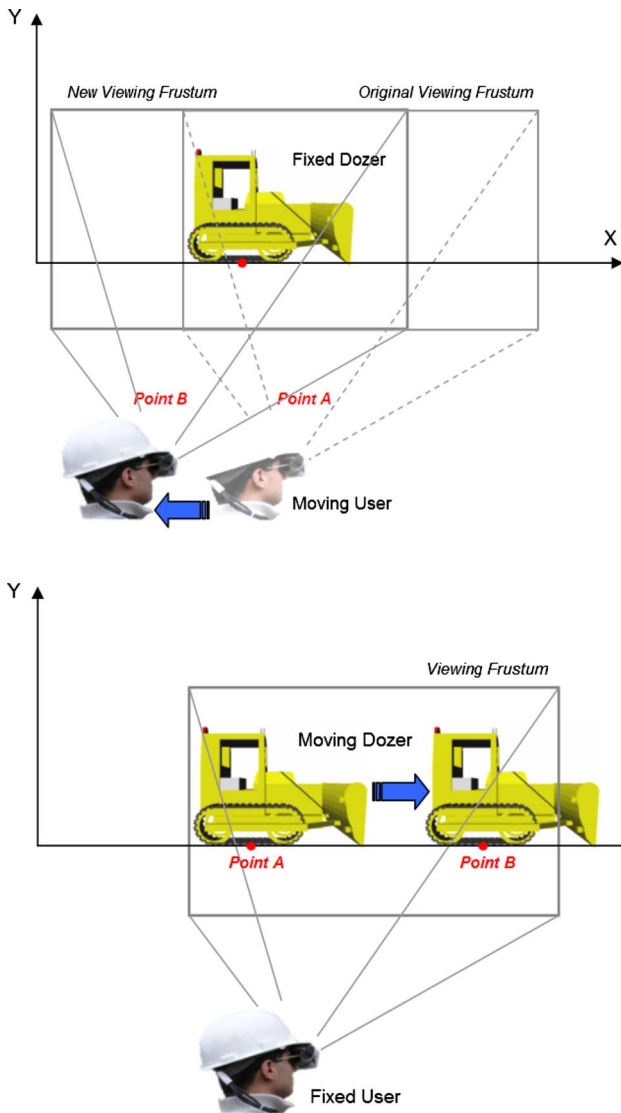


Fig. 12. Change in the contents of user's viewing frustum during the animation

(Vincenty 1975). The new method, when applied on two points with known global positions, calculates their relative distance components along the three major axes in meters (Behzadan and Kamat 2007). These values are later used inside the AR application for various transformational computations. Fig. 13 shows the different outputs of the original Vincenty method and the method developed by the writers.

Validation

The effectiveness of ARVISCOPE in facilitating automatic, rapid generation of operations level animations of simulation scenarios in AR was validated at different locations within the vicinity of the city of Ann Arbor, Mich. The results of each test were validated from two points of view: animation correctness, and positional tracking. DES models of several modeled construction operations were used to automatically generate ARVISCOPE animation trace files as they ran. Although the execution of a DES model typically takes a few seconds, the generated animation trace file can contain several thousand lines depending on the

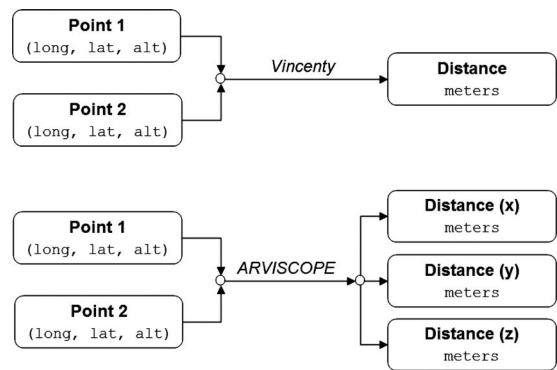


Fig. 13. Calculating the relative distance between two global points

duration of the operation, the number of activities to be performed, and the level of detail desired in animating each activity in ARVISCOPE.

The resulting trace file is a sequence of ARVISCOPE language statements ordered by their time of execution. The trace file was then input to the AR application to investigate the extent to which the simulated (and communicated) operations can be recreated in the 3D augmented environment of the construction site. The degree of accuracy was a function of the amount of detail communicated by the driving process (i.e., the DES models in this case), and the level of accuracy of the tracking devices. As the entire content of an animation trace file is created by an external process (i.e., a running DES model), alternate operational scenarios can also be generated for each case simply by manipulating the quantitative and/or logical simulation parameters of the driving process model. These parameters include the number and type of resources (e.g., number of trucks in an earth-moving operation, type of crane to use in steel erection, space for temporary storage of materials etc.), the rules under which the different tasks that compose the operations are performed and different random variables to describe the durations of individual tasks. The modified models can be promptly rerun to produce an alternate animated scenario that can then be processed and visualized in AR by ARVISCOPE for evaluation.

Validation of Animation Correctness

The ARVISCOPE animation language was able to accurately describe and depict graphical 3D representations of the communicated operations, and superimpose them on top of live video streams of the surrounding environment. Each of the conducted validation scenarios was designed very carefully with regards to several factors such as operational logic, test location, and CAD object selection and manipulation. First, a sketch of each test was created showing different resources used in the operation together with their maneuvering range and movement directions. STROBOSCOPE was then used to create a DES input file of the operation. The equivalent ARVISCOPE animation trace file was generated automatically by instrumenting the original DES model using ARVISCOPE scene construction, dynamic, and control statements. Finally, the user was equipped with the AR mobile backpack (as shown in Fig. 2) and the animation was run in real time as the user was walking freely in the scene looking at the dynamic augmented animation from different positions and orientations. The animation was created in a 640 by 480 resolution screen using a refresh rate of 30 frames per second. To assure animation correctness for the modeled operations, the output of

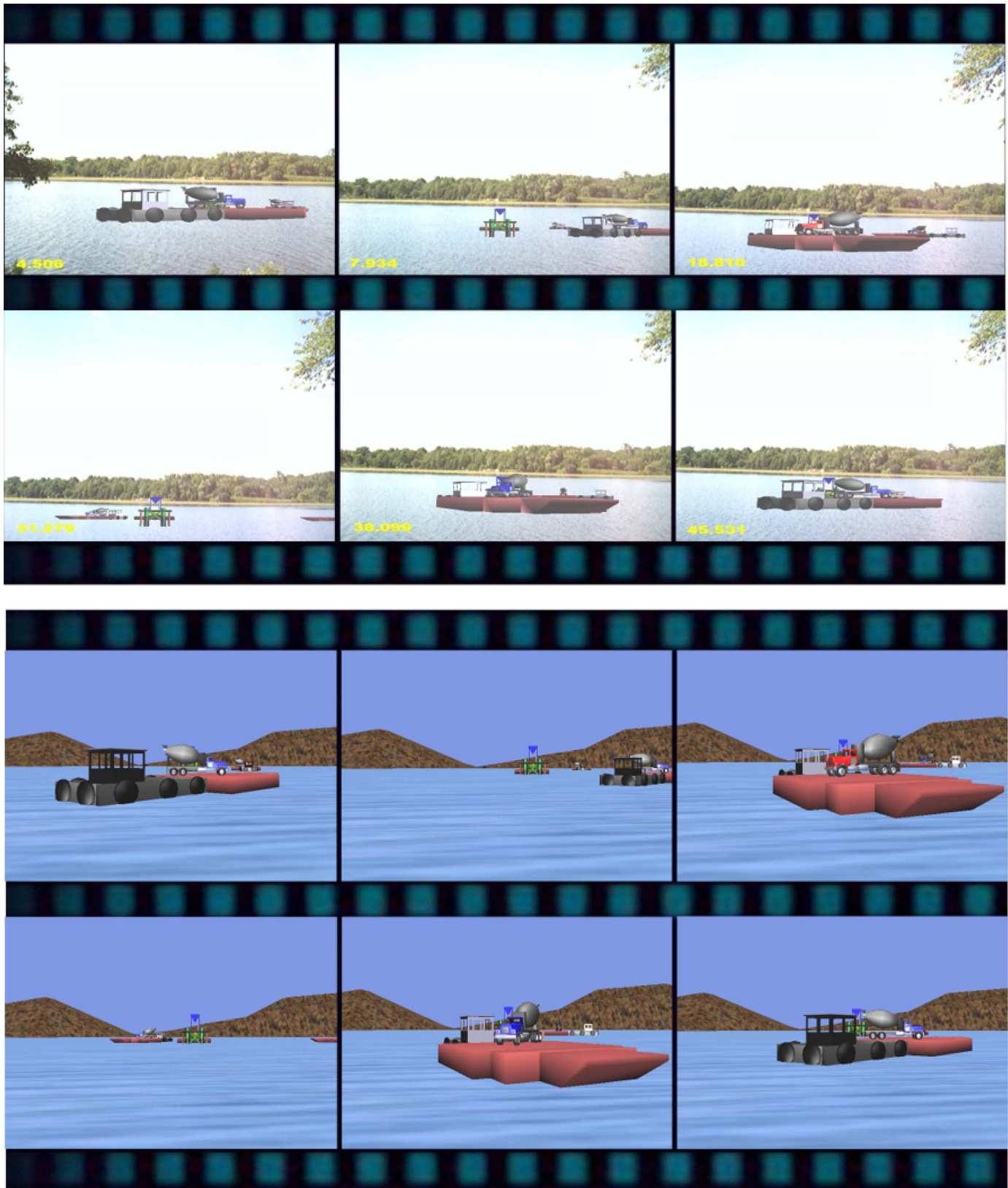


Fig. 14. Concrete delivery to a bridge pier in ARVISCOPE and VITASCOPE

the ARVISCOPE screen at each stage was compared side by side to that of VITASCOPE (Kamat 2003). VITASCOPE is a user extensible 3D animation language designed specifically for visualizing simulated construction operations in smooth, continuous, animated 3D virtual worlds (Kamat and Martinez 2003). The

same instrumentation procedure was used to create VITASCOPE animation trace files using its own language statements. Each resulting trace file was then loaded and run inside the VITASCOPE environment and several snapshots were taken at the exact same time stamps as those of ARVISCOPE snapshots. This facilitated



Fig. 15. Steel erection operations (ARVISCOPE) with user's global position change

side by side comparison of the two sets of animations (e.g., VR based and AR based) for each validation scenario. Fig. 14 shows the results of an outdoor test conducted to validate the animation correctness of ARVISCOPE.

Validation of the Positional Tracking

To validate the functionality of the tracking devices inside the application, the user was allowed to walk in the animation, get closer to CAD objects, and change head orientation. The objective of this part was to ascertain that the tracking devices are fully capable of obtaining the real time user's position and head orientation, and that the application is robust enough to update the contents of the user's augmented viewing frustum based on these values. Continuous communication with the GPS and head orientation tracking devices to obtain real time 6 degrees-of-freedom (6DOF) spatial data of the user was very critical as the augmented viewing frustum has to be constantly reconstructed in front of the user's eyes during the course of the animation. Aside from position measurement errors of the wide area augmentation system service provided at no cost to the GPS users, the overall performance of the application to obtain, store, and retrieve 6DOF data from the tracking devices was found to be acceptable. Fig. 15 shows a validation test in which the user moves inside a steel erection animation and gets closer to the CAD objects. In Fig. 16, the contents of the augmented scene are updated based on the user's latest global position. Fig. 16 shows the results of another validation test in which the user is standing in a fixed global position inside an earth-moving animation, simultaneously changing head orientation. In each case, the ARVISCOPE animation

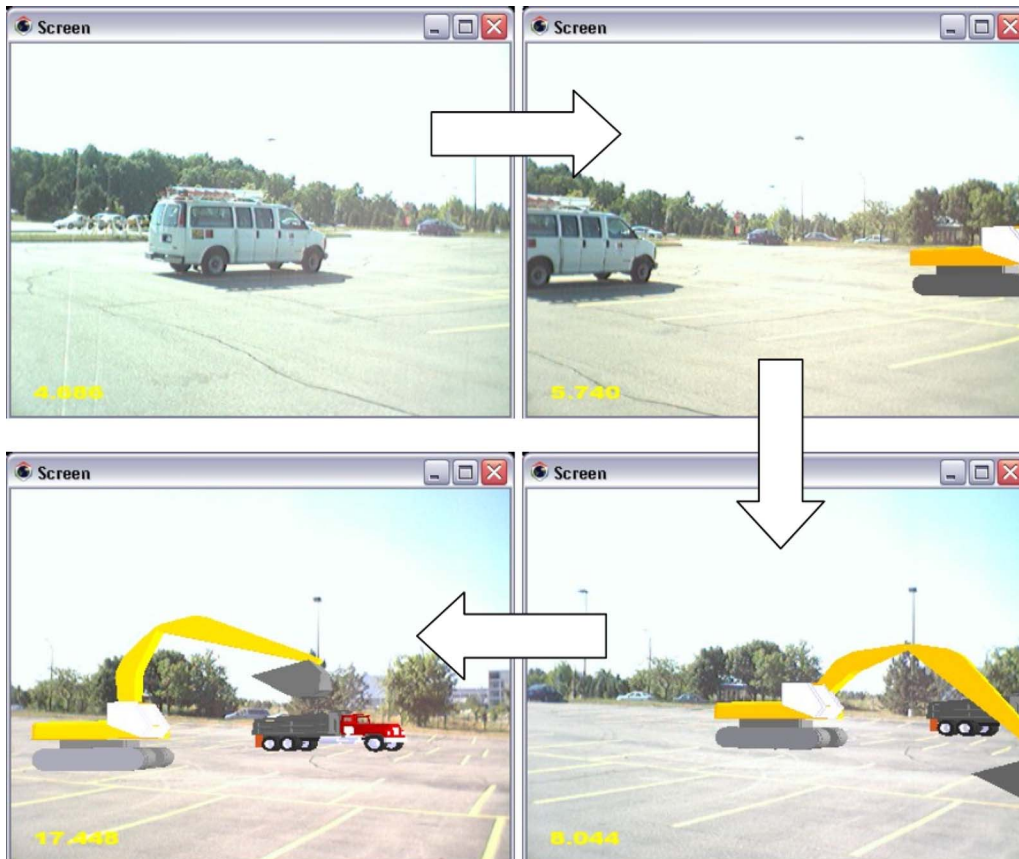


Fig. 16. Earth-moving operations (ARVISCOPE) with user's head orientation change

was completely responsive to changes in user's global position and head orientation by continuously updating the position and/or orientation of the CAD objects inside the augmented viewing frustum. The head tracker device used in these experiments was capable of providing orientation data with a resolution of 0.1° root mean square (RMS). The yaw, pitch, and roll angles were also measured with accuracies typically less than 0.5° RMS which is satisfactory for wide range operations.

Future Work

The AR-based animation authoring language introduced in the presented paper has the potential for improvement to include other complexities encountered in typical construction operations. The writers are working on the syntax design and implementation of a number of new statements in the form of add-ons to visualize certain types of visual effects such as pouring fluid material (e.g., water, concrete, shotcrete, slurry). From the point of view of the tracking devices, work is in progress to minimize the effect of video jittering as a result of GPS and head orientation tracker data inaccuracy. The writers are in the design process of a rigid hard hat and backpack to eliminate inline data inaccuracy. Automatic terrain data retrieval from electronic sources such as geographical information system maps is another aspect of the presented research, which is currently being investigated by the writers. Having the knowledge of terrain, CAD objects can be correctly placed and oriented on the real terrain with minimum elevation error. The writers are also working on remote sensing and fast prototyping of the real world as real time data acquisition of the 3D space enhances the AR application knowledge of the surrounding environment. The obtained information can be effectively used to resolve problems arising from interactions between real and virtual objects such as occlusion, and collision detection. The results of this stage can be eventually used to generate mixed reality environments in which real and CAD objects (i.e., material, resources) are able to interact and communicate with each other.

Conclusions

Creating smooth and chronologically accurate animated scenes of a simulated operation modeled in a DES tool requires careful assignment of time tags to discrete events occurring during a simulation run. These time tags can then be used to generate an animation scenario which can be interpreted by an AR visualization application. Several steps need to be taken to create a logical link between a running DES model and an AR animation. First, an animation trace file has to be generated. The generated trace file has to be interpreted by the AR visualization application and required graphical representations (i.e., CAD objects) have to be loaded and displayed in their appropriate position and orientation inside the user's viewing frustum. As the user of the AR application is given full mobility inside the animation, the contents of this frustum have to be completely sensitive to changes in user's position and head orientation. This requires the AR application to be constantly communicating with registration devices (e.g., GPS and head orientation tracker) and using their output to calculate the coordinates and update the contents of the viewing frustum in real time. The writers have successfully designed methods and algorithms to address the previous challenges in AR. In this paper, an animation authoring language called ARVISCOPE, designed to

validate the functionality and effectiveness of the methods described earlier was presented and validated. Although the results of the presented paper have been validated using a number of simulated construction operations, most of the findings of this research project are generic and widely applicable to other fields of science and engineering where the need to animate and communicate simulated operations is as important as that in construction.

Acknowledgments

The presented work has been supported by the National Science Foundation (NSF) through Grant No. CMS-0448762. The writers gratefully acknowledge NSF's support. Any opinions, findings, conclusions, and recommendations expressed in this paper are those of the writers and do not necessarily reflect the views of the NSF.

References

- Barfield, W., and Caudell, T. (2001). *Fundamentals of wearable computers and augmented reality*, Lawrence Erlbaum Associates, London.
- Barnes, M. R. (1997). "An introduction to QUEST." *Proc., Winter Simulation Conf. (WSC97)*, IEEE, Piscataway, N.J., 619–623.
- Behzadan, A. H., and Kamat, V. R. (2005). "Visualization of construction graphics in outdoor augmented reality." *Proc., Winter Simulation Conf. (WSC05)*, IEEE, Piscataway, N.J., 1914–1920.
- Behzadan, A. H., and Kamat, V. R. (2007). "Georeferenced registration of construction graphics in mobile outdoor augmented reality." *J. Comput. Civ. Eng.*, 21(4), 247–258.
- Behzadan, A. H., Timm, B. W., and Kamat, V. R. (2008). "General purpose modular hardware and software framework for mobile outdoor augmented reality applications in engineering." *Adv. Eng. Inf.*, 22(1), 90–105.
- Bishop, J. L., and Balci, O. (1990). "General purpose visual simulation system: A functional description." *Proc., 1990 Winter Simulation Conf. (WSC90)*, IEEE, Piscataway, N.J., 504–512.
- Brooks, F. P., Jr. (1999). "What's real about virtual reality?" *IEEE Comput. Graphics Appl.*, 19(6), 16–27.
- Brown, D., Julier, S., Baillot, Y., and Livingston, M. (2003). "An event-based data distribution mechanism for collaborative mobile augmented reality and virtual environments." *Proc., IEEE Virtual Reality*, IEEE, Piscataway, N.J., 23–29.
- Feiner, S., MacIntyre, B., Hollerer, T., and Webster, A. (1997). "A touring machine: Prototyping 3D mobile augmented reality systems for exploring the urban environment." *Proc., Int. Symp. on Wearable Computing (ISWC '97)*, IEEE, Cambridge, Mass., 74–81.
- Gleue, T., and Dähne, P. (2001). "Design and implementation of a mobile device for outdoor augmented reality in the archeoguide project." *Proc., 2001 Conf. on Virtual Reality, Archeology, and Cultural Heritage*, ACM Press, Glyfada, Greece, 161–168.
- Halpin, D. W., and Riggs, L. S. (1992). *Planning and analysis of construction operations*, Wiley, New York.
- Hammad, A., Garrett, J. H., and Karimi, H. (2004). "Location-based computing for infrastructure field tasks." *Telegeoinformatics: Location-based computing and services*, CRC, Boca Raton, Fla., 287–314.
- Kamat, V. R. (2003). "Vitascope: Extensible and scalable 3D visualization of simulated construction operations." Ph.D. dissertation, Virginia Polytechnic Institute and State Univ., Blacksburg, Va.
- Kamat, V. R., and El-Tawil, S. (2007). "Evaluation of augmented reality for rapid assessment of earthquake-induced building damage." *J. Comput. Civ. Eng.*, 21(5), 303–310.

- Kamat, V. R., and Martinez, J. C. (2001). "Visualizing simulated construction operations in 3D." *J. Comput. Civ. Eng.*, 15(4), 329–337.
- Kamat, V. R., and Martinez, J. C. (2002). "Scene graph and frame update algorithms for smooth and scalable 3D visualization of simulated construction operations." *Comput. Aided Civ. Infrastruct. Eng.*, 17(4), 228–245.
- Kamat, V. R., and Martinez, J. C. (2003). "Automated generation of dynamic, operations level virtual construction scenarios." *Electron J. Inf. Technol. Constr.*, 8, 65–84.
- Livingston, M., Rosenblum, L., Julier, S., Brown, D., and Baillot, Y. (2002). "An augmented reality system for military operations in urban terrain." *Proc., Interservice/Industry Training, Simulation, and Education Conf. (IITSEC '02)*, National Defense Industrial Assoc., Orlando, Fla., 1–8.
- Martinez, J. C. (1996). "Stroboscope: State and resource based simulation of construction operations." Ph.D. dissertation, Univ. of Michigan, Ann Arbor, Mich.
- Martinez, J. C. (2001). "EZStrobe—General-purpose simulation system based on activity cycle diagrams." *Proc., 1998 Winter Simulation Conf. (WSC98)*, IEEE, Piscataway, N.J., 341–348.
- Martinez, J. C., and Ioannou, P. G. (1999). "General-purpose systems for effective construction simulation." *J. Constr. Eng. Manage.*, 125(4), 265–276.
- McKinney, K., Kim, J., Fischer, M., and Howard, C. (1996). "Interactive 4D-CAD." *Proc., 3rd Congress on Computing in Civil Engineering*, ASCE, Reston, Va., 383–389.
- Op den Bosch, A. (1994). "Design/construction process simulation in real-time object-oriented environments." Ph.D. dissertation, Georgia Institute of Technology, Atlanta.
- Piekarski, W., and Thomas, B. H. (2003). "ARQuake—Modifications and hardware for outdoor augmented reality gaming (LINUX03)." *Proc., 4th Australian Linux Conf.*, Univ. of Western Australia, Perth, Australia.
- Roberts, G. W., Evans, A., Dodson, A., Denby, B., Cooper, S., and Hollands, R. (2002). "The use of augmented reality, GPS, and INS for subsurface data visualization." FIG XXII International Congress, Washington, D.C.
- Rohrer, M. W. (2000). "Seeing is believing: The importance of visualization in manufacturing simulation." *Proc., Winter Simulation Conf. (WSC00)*, IEEE, Piscataway, N.J., 1211–1216.
- Rohrer, M. W., and McGregor, I. W. (2002). "Simulating reality using AUTOMOD." *Proc., Winter Simulation Conf. (WSC02)*, IEEE, Piscataway, N.J., 173–181.
- Rojas, E. M. (2000). "Virtual environments for construction engineering and management education." *Proc., Construction Congress VI*, ASCE, Reston, Va., 263–270.
- Shi, J. J., and Zhang, H. (1999). "Iconic animation of construction simulation." *Proc., 1999 Winter Simulation Conf. (WSC99)*, IEEE, Piscataway, N.J., 992–997.
- Suthau, T., Vetter, M., Hassenpflug, P., Meinzer, H., and Hellwich, O. (2002). "A concept work for augmented reality visualization based on a medical application in liver surgery." Technical Univ., Berlin, Commission V, WG V/3.
- Thomas, B., Close, B., Donoghue, J., Squires, J., Bondi, P., Morris, M., and Piekarski, W. (2000). "ARQuake: An outdoor/indoor first person augmented reality application." *Proc., 4th Int. Symp. on Wearable Computers (ISWC2000)*, IEEE, Piscataway, N.J., 149–146.
- Tucker, S. N., Lawrence, P. J., and Rahilly, M. (1998). "Discrete-event simulation in analysis of construction processes." *CIDAC Simulation Paper*, Melbourne, Australia, 1–14.
- Vincenty, T. (1975). "Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations." *Surv. Rev.*, 23176, 88–93.
- Webster, A., Feiner, S., MacIntyre, B., Massie, W., and Krueger, T. (1996). "Augmented reality in architectural construction, inspection and renovation." *Proc., 3rd Congress on Computing in Civil Engineering*, ASCE, Reston, Va., 913–919.