Available online at www.sciencedirect.com

ScienceDirect

Advances in Engineering Software 39 (2008) 233–241

ADVANCES IN ENGINEERING SOFTWARE

www.elsevier.com/locate/advengsoft

Logical product models for automated scripting of process-level construction animations

Vineet R. Kamat [*]

Department of Civil and Environmental Engineering, University of Michigan, 2340 G.G. Brown, 2350 Hayward, Ann Arbor, MI 48109-2125, USA

Received 5 May 2006; accepted 22 January 2007
Available online 10 April 2007

Abstract

Animation can add significant value to Discrete-Event Simulation by helping verify, validate, and accredit simulation analyses. This is particularly true in construction where typical decision makers are experts in their domain but are not generally proficient in simulation itself. Existing methods of animating simulated construction processes in 3D have a significant limitation. They require that spatial information about construction workspaces and geometric details of facility components being erected be both captured into simulation models, even though such information is otherwise irrelevant from the process modeling perspective. This cumbersome, time-consuming, and often impossible activity precludes the widespread use of animation in simulation analyses. This research addressed this limitation and studied the applicability of the CIMsteel Integration Standards (CIS/2) in supporting the automated scripting of process-level construction animations. Member geometry, position, and orientation of steel beams and columns were extracted from structural frames described in the CIS/2 format, and were used to automatically generate animation scripting commands. The generated script was input to a kinematically smart crane inside a 3D virtual world to demonstrate the efficacy of logical product models in supporting the automated visualization of simulated construction processes. The designed algorithms and methods were implemented in a software tool called AutoCIS2 that simulation modelers can use to automatically generate animation scripts describing modeled steel erection operations in smooth, continuous, 3D virtual worlds.
© 2007 Elsevier Ltd. All rights reserved.

Keywords: CIS/2; Construction; Computer graphics; Simulation; Structural steelwork; Visualization

1. Introduction

Visualizing construction at the operations level is a complex proposition that involves being able to view the interaction of the various resources as they build the product or perform a support service. In order to visualize an operation it is necessary to see pieces of equipment and craftsmen transforming materials and other physical components into a completed facility. The depicted movements and transformations must be spatially and temporally accurate. In order to depict smooth motion, visual elements must be shown at the right position and orientation several times per second. Issues such as 3D paths, speed, and acceleration need to be considered.

Most of these dynamics are continuous in nature. However, to visualize operations modeled using Discrete-Event Simulation (DES), these dynamics must be recreated from very limited, discrete, and unevenly-spaced pieces of operational information that is captured in and can be communicated by DES models. Conceptually, simulated construction operations can be visualized by linking DES models with 3D CAD models of the infrastructure and the equipment, craftsmen, temporary structures, and other resources. The results are 3D animations of simulated construction operations [3,8].

The current state of knowledge in achieving this link however, is relatively primitive and impractical for widespread adoption in animating simulated processes. In

[*] Tel.: +1 734 764 4325; fax: +1 734 764 4292.
E-mail address: vkamat@umich.edu

particular, the interface between animation methods and DES models that use those methods has a critical limitation. Existing animation methods require that spatial information about construction workspaces and geometric details of the facility components being built be captured into simulation models. Such spatial and geometric data is otherwise irrelevant in DES from the modeling perspective.

1.1. Description of the problem

The existing problem can be illustrated by using the modeled steel erection operation graphically presented in Fig. 1. A DES model created using construction tasks as building blocks can communicate instances of those tasks to animation methods using the vocabulary of the pertinent piece of equipment involved in the modeled tasks (e.g. crawler crane in Fig. 1).

For instance, each time a steel member is erected, i.e. the activity "Erect Steel Member" takes place, the simulation model can communicate the following instruction to an animation script requesting the instantiated virtual crawler crane to perform the assigned task.

```
Crane1.PutThatThere LA2550 (8,9,2) 45 90
60;
```

When the animation script processing algorithm interprets such a statement, it first computes the elemental crane motions necessary to accomplish the task and apportions the total task time communicated by the simulation model (60 in the above statement) to the individual elemental motions. The algorithm then generates elementary motion-describing geometric transformations (e.g. translations and rotations) to manipulate individual CAD elements and graphically depict the performance of the communicated task inside a 3D virtual world.

The animation methods are based on inverse kinematics and are parametric in nature. Thus, in the above example, given the position (8,9,2) and the horizontal and vertical orientations (45 and 90, respectively) where column LA2550 must be installed, the algorithm can compute the elementary geometric motions required to first move the virtual crane's hook to the location where the column is

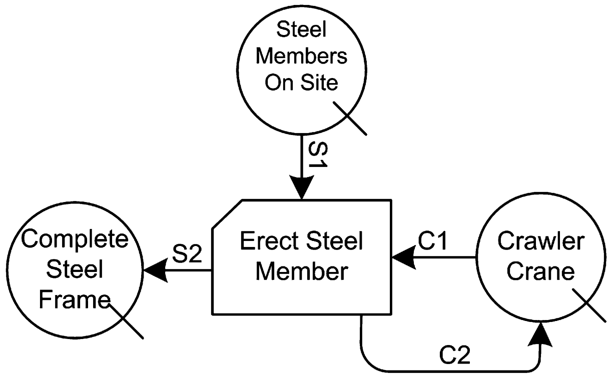


Fig. 1. Modeling a steel erection operation at basic task-level detail.

staged, and then transport it to its final installed location in the structure [7]. The methods are parametric in nature because while the same instruction (e.g. PutThatThere) may appear in an animation script many times, the component to manipulate (e.g. steel member) and the value of the instructions' arguments will differ each time.

The important point to note is that the final installed position and orientation of each component (e.g. steel member) erected by tasks simulated in a DES model must be communicated to the animation methods by the simulation model itself. A DES model, even if created with high sub-task level detail, does not capture and cannot automatically access or generate such geometric or spatial information. DES models are schematic process models that do not have and, in most cases, do not need spatial knowledge about the geometry of modeled workspaces or the erected facility components. In order to animate modeled operations however, all the required geometric and spatial data must be captured inside a simulation model so that it can be later communicated to the animation methods via scripting commands.

The data that is so captured in a DES model must in turn be manually deciphered from the design (e.g. CAD model) of the structure whose erection is being simulated and animated. For instance, in this example, Crane1 can be instructed to erect column LA2550 only after the final erected position (8,9,2) and orientations (45 and 90) are known and captured into the DES model.

Even though such geometric and spatial data is available from the CAD design of a structure (e.g. steel frame), DES models have no straightforward method to access the information. The details must thus be explicitly (and manually) entered into created DES models by the simulation modeler. This cumbersome, time-consuming, and often impossible activity is necessary to allow simulation models to communicate scripted instructions to the animation methods that will portray the modeled processes inside a 3D virtual world.

1.2. Main research contributions

The presented research addressed this limitation and studied the applicability of standard logical product models in supporting the automated scripting of process-level construction animations. Existing animation methods analyze articulated pieces of construction equipment and human craftsmen as systems of linkages whose implement (e.g. excavator's bucket, crane's hook, etc.) can be considered as the end-effector of a kinematic chain. The goals of the end-effectors, i.e. the positions where the equipments' implement should be at key instances during an animated construction task depends on the initial staging location and the final in-place installed configuration (i.e. position and orientation) of the component being erected.

While the initial staging location for a component is project and site dependent, the final geometric configuration can be conceptually extracted from the product model of the facility under consideration. This study evaluated the above hypothesis and investigated the means to allow DES models to automatically extract geometric and spatial component data from a product model of the structure whose construction is being simulated and animated.

In particular, the research studied the extent to which the CIMsteel Integration Standards (CIS/2) can specify product descriptions capable of supporting automated scripting of process-level animations of steel erection operations. Algorithms to automatically extract the geometry, position, and orientation of steel beams and columns from a structural frame described in the CIS/2 format were designed and implemented. The extracted steel member information was used to generate animation scripting commands for a kinematically smart crane inside a 3D virtual world to study the efficacy of logical product models in supporting the automated animation of construction operations modeled using DES.

2. CIMsteel Integration Standards Release 2

The CIMsteel Integration Standards Release 2 (CIS/2) is a product data model for structural steel. It is the result of the pan-European Eureka EU130 CIMsteel Project and has been endorsed by the American Institute of Steel Construction (AISC) as the technical basis for their Electronic Data Interchange initiative. The goal of CIS/2 is to create a seamless and integrated flow of information among all project participants involved in the construction of steel framed structures. These participants include steel designers, analysts, detailers, fabricators, and erectors.

The CIS/2 product data model deals with information about the steel structure throughout its analysis, design, detailing, and fabrication life cycle. The geometry of the structure is only one property of the product data model. Other attributes of the structure include how parts are combined into assemblies and structures, analysis loads and reactions, material types, connection details, associations between members and drawings, and modification history.

The product data model is defined by a schema that details how all the information in the CIS/2 standard is represented and how each entity relates to each other. It also defines rules for using various entities and the information those entities contain. STEP [4], the Standard for Exchange of Product Model Data, is the technical basis used to represent CIS/2 information. STEP is a worldwide effort to develop mechanisms for the exchange and sharing of engineering data. The format of a CIS/2 file that is imported or exported by steel related software packages is specified in a standard way to facilitate the exchange of information among various applications. The STEP Part 21 implementation method defines the physical structure of a file.

2.1. Structure of the CIS/2 product data model

A CIS/2 file can support three primary models of structural steel information – the analysis model, the design model, and the manufacturing model. Information on any of these models can coexist in the same CIS/2 file. An analysis model of a steel structure consists of nodes and elements and supports several static and dynamic analysis methods. A design model represents a steel structure as a design assembly to allow member and connection design. A design assembly can be partitioned into other simpler design assemblies and eventually into design parts and design joint systems. The design parts and joint systems, respectively form the conceptual representations of a basic piece of steel and a basic joint system.

A manufacturing model in CIS/2 represents a steel structure as manufacturing assemblies for the purpose of detailing, production planning, and manufacturing. In a manufacturing model, located assemblies are comprised located parts and located joint systems that, respectively, represent a basic physical piece of steel and a basic physical joint system. All located items can be combined into larger located assemblies that eventually define a complete structure.

Fig. 2 shows a sample of a CIS/2 file for a part with a location. The file is represented in the standard STEP Part 21 format. Each CIS/2 entity instance is assigned a number indicated by a pound (#) sign. The name of the CIS/2 entity then appears in upper case letters. Every entity has a number of fields that contain text strings, numeric values, boolean values, references to other entities, or null values indicated by a dollar sign. The indentation has been added to show the hierarchy and relationships between the various entities.

The top-level entity is a LOCATED_PART (#43) that associates a PART (#33) with a coordinate system (#42). The LOCATED_PART also refers to a LOCATED_ASSEMBLY (#20). The PART refers to a SECTION_PROFILE (#21), a LENGTH (#26), and a MATERIAL (#27). The LOCATED_ASSEMBLY also refers to a COORD_SYSTEM (#18), an ASSEMBLY (#19), and a STRUCTURE (#10). These statements describe that the part, which is a W14X158 wide-flange section with a given length, has a location in an assembly that in turn is located in the structure.

According to the CIS/2 schema, all located parts must be unique. However, there can be multiple references to a single part. For a simple framed structure, each beam or column is a located part. However, there need be only one referenced part for members that have the same section profile and length. Multiple located parts can refer to the same located assembly. For example a beam and the clip angles and gusset plates at its ends can all be part of the same located assembly. Features such as hole layouts, copes, and miters can be applied to a part with the LOCATED_FEATURE_FOR_LOCATED_PART entity. This entity refers to the specifications and dimensions of the feature, the coordinate system to locate the feature on the part, and the located part on which they are applied. A joint system can be either a bolted or welded connection and is located on an assembly in a similar manner.

```
#43= LOCATED_PART(92,'92','brace',#42,#33,#20);
  #42= (COORD_SYSTEM('','Part CS’,$,3)
      COORD_SYSTEM_CARTESIAN_3D(#40)COORD_SYSTEM_CHILD(#18));
    #40= AXIS2_PLACEMENT_3D('Part axes',#34,#38,#36);
      #34= CARTESIAN_POINT('Part origin',(0.,0.,0.));
      #38= DIRECTION('Part z-axis',(0.,0.,1.));
      #36= DIRECTION('Part x-axis',(1.,0.,0.));
    #18= COORD_SYSTEM_CARTESIAN_3D('','Assembly CS',$,3,#17);
      #17= AXIS2_PLACEMENT_3D('Assembly axes ',#11,#15,#13);
        #11= CARTESIAN_POINT('Assembly origin ',(720.,540.,120.));
        #15= DIRECTION('Assembly z-axis ',(-0.37139068,0.,0.92847669));
        #13= DIRECTION('Assembly x-axis ',(0.92847669,0.,0.37139068));
  #33= (PART(.UNDEFINED.,$)PART_PRISMATIC()PART_PRISMATIC_SIMPLE(#21,#26,$,$)
      STRUCTURAL_FRAME_ITEM(92,'92','brace')STRUCTURAL_FRAME_PRODUCT($)
      STRUCTURAL_FRAME_PRODUCT_WITH_MATERIAL(#27,$,$));
    #21= SECTION_PROFILE(1,'W14X158',$,$,5,.T.);
    #26= POSITIVE_LENGTH_MEASURE_WITH_UNIT
       (POSITIVE_LENGTH_MEASURE(258.48791),#3);
      #3= (CONTEXT_DEPENDENT_UNIT('INCH')LENGTH_UNIT()NAMED_UNIT(#1));
        #1= DIMENSIONAL_EXPONENTS(1.,0.,0.,0.,0.,0.,0.);
    #27= MATERIAL(1,'GRADE50',$);
  #20= LOCATED_ASSEMBLY(92,'92','brace',#18,$,#19,#10);
    #18= COORD_SYSTEM_CARTESIAN_3D('','Assembly Coordinate System',$,3,#17);
      #17= AXIS2_PLACEMENT_3D('Assembly axes ',#11,#15,#13);
        #11= CARTESIAN_POINT('Assembly origin ',(720.,540.,120.));
        #15= DIRECTION('Assembly z-axis ',(-0.37139068,0.,0.92847669));
        #13= DIRECTION('Assembly x-axis ',(0.92847669,0.,0.37139068));
    #19= ASSEMBLY_MANUFACTURING(92,'92','brace',$,$,$,$,$,$,$);
    #10= STRUCTURE(1,'cis_2','Unknown');
```

Fig. 2. Located part in a CIS/2 file.

3. Research initiative

The idea of embedding additional information into product models to help describe the installation process has been explored in previous work [10]. In addition, the possibility of capitalizing on information contained in product models for purposes other than design has also been suggested many years ago [9]. For example, Reed [11] described how CIS/2 models can allow the automatic extraction of spatial steel member information that can be conveniently used to format instructions to automation equipment.

In automated steel erection, by knowing what the final installed position and orientation of a steel member in a frame is going to be, a piece of robotic equipment can use inverse kinematics algorithms to first move its grippers to the current location of the member, and then transport it to the location in the frame where it is to be installed. This concept of autonomous pick and place of a steel beam has been demonstrated using the NIST RoboCrane and a simple two-column frame with ATLSS connectors [12]. Typically, additional spatial information such as the geometry of the member and the partially completed structure, and other existing obstructions is also required to be input to the equipment so that interference detection algorithms can work with the inverse kinematics algorithms to compute a collision-free path for the equipment and the steel member [1].

A piece of robotic equipment can conceptually decipher the final position and orientation of each steel member in the completed structure from a rich geometric product model of the structural frame. CIS/2 models, for instance, contain the definition of all steel parts, prefabricated assemblies, and connecting joint systems that are to be erected in a steel structure. The nested coordinate systems can define the location of steel parts in assemblies that in turn can be located in larger assemblies, ultimately culminating in defining the location of the entire frame in a local coordinate system. The spatial configuration (position and orientation) of each steel member in its final installed location can then be computed by transforming the model's coordinate frame into a global geo-referenced coordinate system where the structure is to be erected.

In creating a 3D animation of a construction operation with process-level detail, the encountered problem is very similar to that faced by robotics engineers. Conceptually, each piece of equipment on a virtual construction site can be considered as a robot performing a certain construction task. Just as real robots perform specified tasks using real resources, virtual pieces of equipment are required to perform construction tasks on the computer using virtual resources. The striking analogy between the two arises from the fact that both robots and virtual equipment pieces do not have any intrinsic knowledge about performing assigned tasks and need to be programmed in order to perform particular motion sequences.

In terms of information needs, therefore, virtual pieces of construction equipment are very similar to industrial robots. The two may differ in context (real vs. virtual) and shape, but in both cases, engineers are basically interested in trying to manipulate a multiply articulated structure (i.e. a kinematic chain) and move a particular component (e.g. steel member) from one location to another. While inverse kinematics can determine a virtual piece of equipment's articulation to allows its implement to reach a particular position, the identification of those positions itself is a whole separate problem. In the context of this research, those positions are the final in-place configurations where structural steel members will be installed inside a 3D virtual world.

By extracting such target positions from product models, a kinematically smart piece of virtual equipment can automatically compute the articulation that allows its implement (and the attached component) to reach the desired positions. A smooth process-level animation can then be achieved by simply interpolating between the equipment's initial articulation and the computed target articulation. In such a visualization scheme, the process authoring the animation script (e.g. a DES model) only needs to indicate which component to install, and can relegate the entire responsibility of describing the visualization to the animation methods. Such a capability has significant advantages in making process-level animation of simulated operations more practical.

4. Automated scripting of construction animations

The authors' implementation of algorithms that extract member geometry, position, and orientation information from CIS/2 files is a software tool that allows DES models to automatically communicate a smooth, continuous 3D animation of simulated steel erection processes without having to describe the spatial characteristics of the structure being erected. This tool, called AutoCIS2, is implemented as an extension (add-on) to the VITASCOPE visualization system [5,7].

4.1. Implemented animation statements

The AutoCIS2 add-on extends the VITASCOPE animation language by implementing the statements presented in Table 1. In particular, the add-on defines animation statements that automatically parse and interpret the geometry of a CIS/2 structure, and then allow an existing virtual crane to be instructed to automatically erect specific steel members from that structure in the communicated simulation time units.

4.2. Parsing and interpreting member geometry from a CIS2 structure

AutoCIS2 automatically parses and interprets the geometry of steel members that comprise a CIS/2 structure in response to the LOADCIS2VRML statement and its arguments. In particular, the LOADCIS2VRML statement takes three arguments as input. The first argument is a string that specifies the name to be used in referring to this structure in subsequent animation statements. The second argument is a reference to a CAD file that contains the geometry of a CIS/2 structure converted to the VRML format. The final argument represents the units of measurement (e.g. inches, centimeters, etc.) that are applicable to the numerical values inside the CAD file that is being loaded. When AutoCIS2 processes such a statement, it parses the entire CAD file referenced and creates an internal representation of the geometry, position, and orientation of each steel member that comprises the structural frame.

The parsing and interpretation of CIS/2 information contained in a converted VRML file, and its subsequent processing yields the following information for each steel member contained in the represented structural steel frame:

- The name of the member.
- The geometry of the member in local space.
- The position of the member in global space.
- The orientation of the member relative to each coordinate axis in global space.

Together, this automatically extracted data comprises the geometric and spatial information required by a virtual piece of equipment to automatically erect a steel member in 3D by transporting it from a staging area to its installed location along a computed collision-free path.

The intermediate conversion of CIS/2 files to VRML was adopted for the following reasons:

(1) Although CIS/2 supports explicit description of geometry, most CIS/2 parts are expressed using implicit descriptions of geometry based on a transverse section profile and a longitudinal length through which the profile is swept [11]. The VRML translator converts all implicit geometry to regular geometry facilitating member shape extraction.
(2) Implementation of a VRML file parser that can traverse the described scene graph to extract member geometry (shape) and pose (transformation) information was relatively straightforward compared to the implementation of a full-fledged CIS/2 file parser.

Table 1
Animation statements implemented in AutoCIS2

Statement (1)	Usage (2)
`LoadCIS2VRML [StructureName] [DataFileName] [MeasurementUnits];`	Load the product model data defined in the specified file using the indicated measurement units
`[StructureName].PlaceShapeAt [MemberType] [MemberName] [Position];`	Place a specific steel member from the loaded structure at the indicated location on the virtual jobsite
`[StructureName].ErectShapeUsing [MemberType] [MemberName] [CraneName] [OperTime];`	Erect a specific steel member using the specified crane in the indicated amount of time units
`[StructureName].DisplayFullFrame;`	Display the full loaded steel structure for test purposes only

(3) The VRML translator provides a visual interface to the underlying CIS/2 data thereby providing a means to visualize the represented steel structure in a 3D virtual world.

The reader is referred to Kamat and Lipman [6] for a complete description of the algorithms designed to interpret member geometry and spatial configuration from CIS/2 VRML files.

4.3. Animating erection of steel columns and beams

AutoCIS2's intention is to allow an animation authoring process such as a DES model to communicate high-level construction tasks via scripting commands, and then automatically interpret the geometric characteristics of the communicated task and compute the elemental motions required to smoothly portray that task in 3D. The merit behind AutoCIS2's design is that a detailed process-level animation of a construction task can be achieved with minimal operational details being communicated by the animation authoring processes themselves.

Fig. 3 presents an animation script with statements that parse and interpret the CIS/2 structure in the VRML file Bldg1Frame.wrl, and then instruct a virtual crane to install specific steel beams and columns from that structure at specific times. In particular, at animation time zero, a column LP2550 and a beam LP2506 are placed at their staging locations. At time 105, AutoCIS2 is instructed to erect column LP2550 with the help of a pre-instantiated virtual crane named Crane1 in 25 simulation time units. Similarly at time 952, AutoCIS2 is instructed to erect beam LP2506 using the same crane in 30 simulation time units.

AutoCIS2 thus essentially describes an additional layer of abstraction in the interface between DES models and existing methods of animating operations in 3D. This layer of abstraction automatically interprets the geometry and spatial configuration of the components that are to be installed and then subsequently generates the elemental, operation-level details that are required to describe the smooth continuous animation of the construction process.

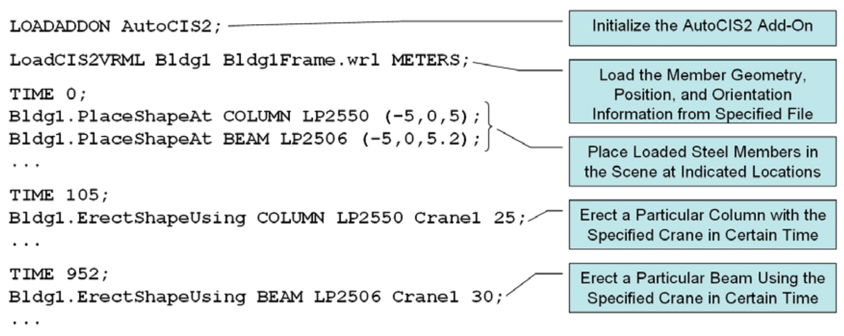
5. Technical approach for automated animation scripting

Fig. 4 graphically presents the relationship between DES models, CIS/2 product models, existing methods of animating simulated processes, and AutoCIS2. Using existing animation methods, a running simulation model would have to communicate a high-level installation task, and the geometric details of the component whose installation is to be animated. The geometric details that the simulation model would have to explicitly communicate to the existing animation methods include the name of the component (e.g. steel member), its final installed position and 3D orientation, the height of the component, and the point on the component where it would be attached to a piece of equipment (e.g. a crane's hook). The last two pieces of information are specifically required to allow the inverse kinematics based animation methods to compute a collision-free path for the virtual equipment erecting the component [5].

With AutoCIS2, the same simulation model can now communicate only the identity of the component to be erected and the name of the virtual piece of equipment to be used to accomplish the task, and can relegate the responsibility of interpreting the geometric and spatial characteristics of the structure to AutoCIS2. The algorithms in AutoCIS2 can then subsequently communicate with the existing animation methods in a construction task-level vocabulary. The authors' implementation of existing inverse kinematics based methods to animate simulated construction processes is a tool called KineMach that itself is implemented as an add-on to the VITASCOPE visualization system [7]. AutoCIS2 thus adds a layer of abstraction between DES models and KineMach, and performs the automated geometry and spatial configuration extraction tasks that would otherwise have to be manually performed by the simulation modeler and included in DES models.

5.1. Hierarchy of animation statements

The relationship between the various components in the AutoCIS2 animation scheme presented in Fig. 4 can be fur-



```
LOADADDON AutoCIS2;
LoadCIS2VRML Bldg1 Bldg1Frame.wrl METERS;
TIME 0;
Bldg1.PlaceShapeAt COLUMN LP2550 (-5,0,5);
Bldg1.PlaceShapeAt BEAM LP2506 (-5,0,5.2);
...
TIME 105;
Bldg1.ErectShapeUsing COLUMN LP2550 Crane1 25;
...
TIME 952;
Bldg1.ErectShapeUsing BEAM LP2506 Crane1 30;
...
```

Initialize the AutoCIS2 Add-On
Load the Member Geometry, Position, and Orientation Information from Specified File
Place Loaded Steel Members in the Scene at Indicated Locations
Erect a Particular Column with the Specified Crane in Certain Time
Erect a Particular Beam Using the Specified Crane in Certain Time

Fig. 3. Communication of steel member erection tasks using AutoCIS2 statements.

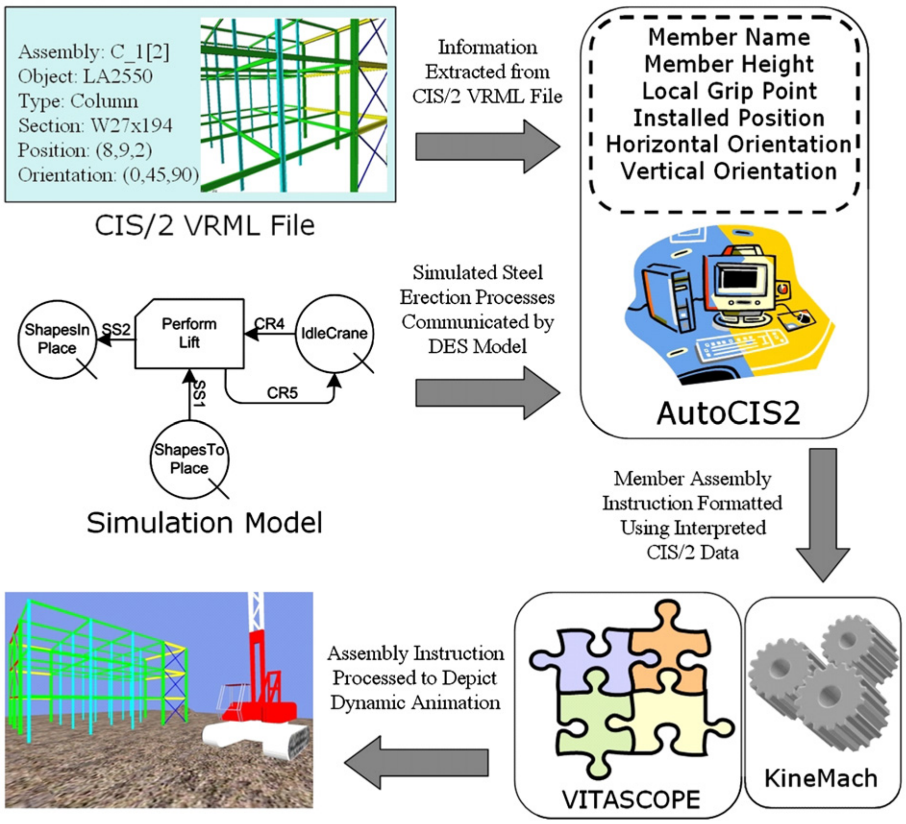


Fig. 4. AutoCIS2 animation schema.

ther elucidated by returning to the example presented earlier in Section 1.1. A DES model created using construction tasks as building blocks (Fig. 1) can communicate instances of those tasks to KineMach's inverse kinematics based methods using the pertinent piece of equipment's vocabulary (e.g. PutThatThere). When KineMach processes each such statement, it first computes the elemental crane motions necessary to accomplish the task and apportions the total task time to the individual elemental motions. KineMach then generates elementary motion-describing animation statements in the VITASCOPE language and forwards them to the visualization engine to graphically depict the operation in 3D.

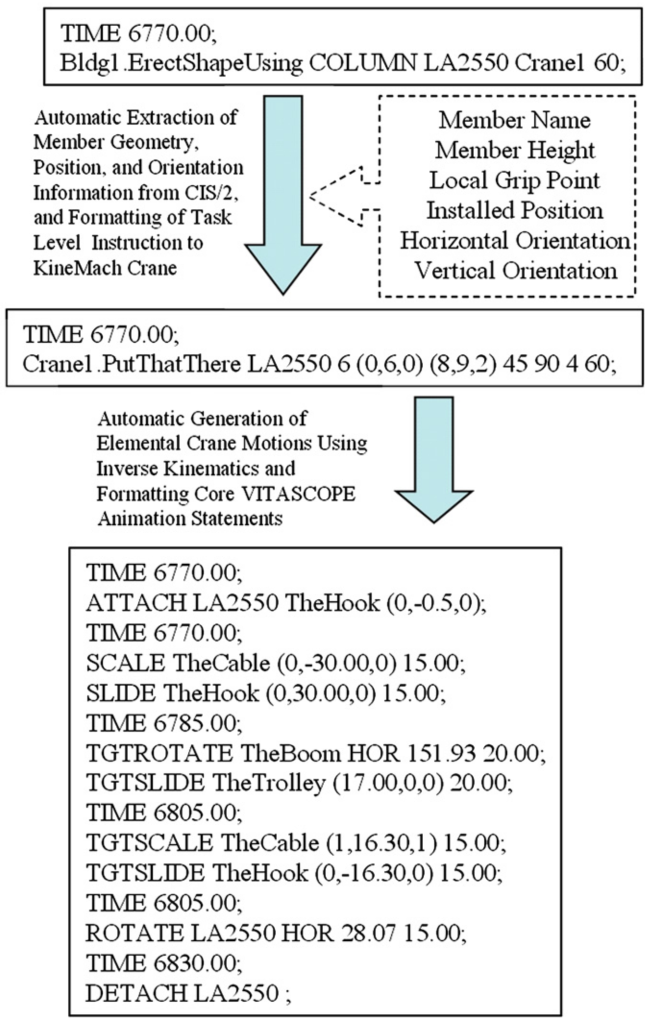
This hierarchy of animation statements among the existing methods is graphically portrayed in the middle and lower half of Fig. 5. By interpreting a communicated ''PutThatThere'' statement that contains the name and the geometric properties of a component to be erected, KineMach automatically generates the geometric transformation level core VITASCOPE animation scripting statements required to animate that task. With the introduction of AutoCIS2, the numeric values of the arguments for KineMach's ''PutThatThere'' statement, i.e. the position (8, 9, 2), horizontal orientation (45), vertical orientation (90), member height (6), local grip point (0, 6, 0), and desired vertical clearance (4) of the component need not be explicitly communicated in an animation statement.

Automated extraction of that information can be relegated to the AutoCIS2 algorithms. Thus, DES models scripting a process-level animation only need to communicate a high-level animation instruction using AutoCIS2's ''ErectShapeUsing'' statement, whose arguments contain just the identity and structural function of the member that is to be erected. As portrayed at the top of Fig. 5, AutoCIS2 thus adds a higher level of abstraction in the hierarchy of animation scripting statements.

AutoCIS2 was evaluated by animating the erection of several CIS/2 VRML files available at the NIST CIS/2 website at http://cic.nist.gov/vrml/cis2.html. Video clips of selected animations are available for download from the author's website at http://pathfinder.engin.umich.edu/.

6. Future work

This research can be extended in several interesting directions. The CIS/2 product model encapsulates the final installed position and orientation of steel members in a structural frame. However, the CIS/2 model captures no information on where steel members are temporarily staged on the jobsite. Such information is required to provide animation methods with automated access to source information. A possible way to address this issue is by extending the CIS/2 model to describe project specific construction planning and site layout information.



```
TIME 6770.00;
Bldg1.ErectShapeUsing COLUMN LA2550 Crane1 60;
```

Automatic Extraction of Member Geometry, Position, and Orientation Information from CIS/2, and Formatting of Task Level Instruction to KineMach Crane
Member Name
Member Height
Local Grip Point
Installed Position
Horizontal Orientation
Vertical Orientation

```
TIME 6770.00;
Crane1.PutThatThere LA2550 6 (0,6,0) (8,9,2) 45 90 4 60;
```

Automatic Generation of Elemental Crane Motions Using Inverse Kinematics and Formatting Core VITASCOPE Animation Statements

```
TIME 6770.00;
ATTACH LA2550 TheHook (0,-0.5,0);
TIME 6770.00;
SCALE TheCable (0,-30.00,0) 15.00;
SLIDE TheHook (0,30.00,0) 15.00;
TIME 6785.00;
TGTROTATE TheBoom HOR 151.93 20.00;
TGTSLIDE TheTrolley (17.00,0,0) 20.00;
TIME 6805.00;
TGTSCALE TheCable (1,16.30,1) 15.00;
TGTSLIDE TheHook (0,-16.30,0) 15.00;
TIME 6805.00;
ROTATE LA2550 HOR 28.07 15.00;
TIME 6830.00;
DETACH LA2550 ;
```

Fig. 5. Hierarchy of animation statements.

The CIS/2 product model contains many weakly specified uses of character strings as identifiers, labels, and descriptions [11]. For example, in the definition of steel parts and assemblies, there are several placeholders in definitions that could be used, for instance, to indicate the structural function (e.g. beam, column, brace, etc.) of a defined part. Such information could be very useful for animation methods in computing the elemental motions of virtual equipment required to graphically depict the simulated processes, and preclude the need to indicate the member function in scripting commands as is the case in AutoCIS2 methods.

However, because the use of such CIS/2 strings is not enforced (i.e. weakly specified), no assumptions can be made by the file parsing algorithms. Two possible ways to address this issue would be to strongly specify the use of information rich string tags in future versions of the product model, or to develop a set of recommended tagging practices at the user-level so that misinterpretation of expected information is minimized [11]. Recommended user-group level practices could also be developed to capture erection sequences accurately during the authoring of CIS/2 files. Such information could be extracted and used by DES models to interpret the correct sequence of erecting members in a structure, thereby precluding the need to manually indicate the order of erection as is done in the AutoCIS2 methods.

Sequence information can be expressed in CIS/2 with the ZONE_OF_STRUCTURE entity. Information from fabrication software such as what material has been shipped to the job site could also be used to determine the erection sequence. In addition, CIS/2 allows the indirect determination of the parts that a LOCATED_JOINT_SYSTEM entity connects through the LOCATED_PART_JOINT entity. This information could be used to determine which assemblies are connected together in the structure and could help eliminate problems such as not having columns in place before a beam connected to them is erected.

Future research could also explore the applicability of other product models such as the Industry Foundation Classes (IFC) [2] and study their effectiveness in supporting the automated animation of simulated construction processes other than structural steel erection.

7. Summary and conclusions

A piece of virtual equipment such as a crawler crane has no intrinsic knowledge of any construction process it animates inside a virtual world. Thus, in animating operations modeled using DES, geometric and spatial information about components to be erected must be explicitly communicated to the equipment using scripting commands so that motion sequences required to install the components can be computed. The temporary staging position and orientation of components to be erected is project and site dependent, and thus cannot be automatically determined beforehand by a process scripting an animation (e.g. a DES model). However, the final in-place spatial configuration of an installed component can be conceptually extracted automatically from a 3D logical product model of the pertinent structure.

The presented study evaluated this hypothesis and investigated the extent to which the CIS/2 product model can specify product descriptions capable of supporting automated scripting of process-level construction animations. Algorithms to automatically extract the geometry, position, and orientation of steel beams and columns from a structural frame described in the CIS/2 format were designed and implemented. A kinematically smart crane modeled using inverse kinematics was implemented in a 3D virtual world. High-level animation scripting commands that only specify what component to install and in how much time were designed to allow DES process models to communicate assembly instructions to the virtual crane. The information extracted from the CIS/2 file was then used to automatically decipher the final position and orientation of the steel members to be erected. The information was subsequently used to automatically compute the elemental crane motions required to graphically portray the erection process inside the virtual world.

Based on the obtained results, it was found that a logical product model such as CIS/2 does encapsulate the basic geometry and configuration (position and orientation) of steel members in a format that can be readily extracted to support automated animation scripting of process-level animations. However, it was found that while CIS/2 itself defines an information rich product model, the semantics of many statements could be improved to strengthen the standard's applicability for capturing assembly sequences and other construction related information. In addition, it was also found that while a product model can describe the final installed configuration of components to be erected, extension of the model itself or integration with a compatible site layout planning system is required to automatically capture the initial configuration of construction components in temporary staging areas.

Acknowledgments

The presented work has been supported by the National Science Foundation (NSF) through Grant CMS-0408538. The author gratefully acknowledges NSF's support. The author thanks Mr. Robert R. Lipman of the National Institute of Standards and Technology (NIST) for providing the CAD files used in this study, for modifying the NIST CIS/2 to VRML converter to facilitate information extraction using AutoCIS2, and for sharing his expertise on CIS/2. Any opinions, findings, conclusions, and recommendations expressed in this paper are those of the author and do not necessarily reflect the views of the NSF, NIST or the individual mentioned here.

References

[1] Cho YK, Haas CT. Rapid geometric modeling for unstructured construction workspaces. J Comput Aided Civil Inf Eng 2003:242–53.

[2] IAI – International Alliance for Interoperability. Industry Foundation Classes; 2003. <www.iai-na.org>.

[3] Ioannou PG, Kamat VR. Intelligent preemption in construction of a manmade island for an airport. In: Proceedings of the 2005 winter simulation conference. Piscataway (NJ): Institute of Electrical and Electronics Engineers (IEEE); 2005. p. 1515–23.

[4] ISO 10303-42. Industrial automation systems – product data representation and exchange – Part 21: Integrated generic resource: Geometric and topological representation. ISO/IEC Geneva, Switzerland (with Technical Corrigendum 1, 2001); 2000.

[5] Kamat VR. VITASCOPE: extensible and scalable 3D visualization of simulated construction operations. PhD Dissertation, Department of Civil and Environmental Engineering, Virginia Tech, Blacksburg (VA); 2003.

[6] Kamat VR, Lipman RR. Emulation of automated structural steelwork erection using CIMsteel Integration Standards. In: Proceedings of the 2006 joint international conference on computing and decision-making in civil and building engineering. Reston (VA): American Society of Civil Engineers; 2006.

[7] Kamat VR, Martinez JC. Dynamic 3D visualization of articulated construction equipment. J Comput Civil Eng 2005;19(4).

[8] Kamat VR, Martinez JC. Visualizing simulated construction operations in 3D. J Comput Civil Eng 2001;15(4):329–37.

[9] Nevis DP, Zabilski RJ. Graphical database for construction planning and cost control. In: Preparing for construction in the 21st century: proceedings of the construction congress. Reston (VA): ASCE; 1991. p. 266–71.

[10] Op den Bosch A. Design/construction processes simulation in real-time object-oriented environments. PhD Dissertation, Georgia Institute of Technology, Atlanta (GA); 1994.

[11] Reed KA. Role of the CIMsteel Integration Standards in automating the erection and surveying of constructional steelwork. In: 19th International symposium on automation and robotics in construction (ISARC). Gaithersburg (MD): NIST; 2002. p. 15–20.

[12] Saidi KS, Lytle AM, Scott NA, Stone WC. Developments in automated steel construction. NISTIR 7264, National Institute of Standards and Technology, Gaithersburg (MD); 2005.