Virtual Reality Procedure Training Simulators in X3D

Olavo Belloc, Rodrigo Ferraz, Marcio Cabral, Roseli Lopes, Marcelo Zuffo
Agenda

- Introduction
- Related Work
- Proposed Nodes
- Results and Live Demo
- Conclusions
Introduction

- Virtual Reality training applications have become increasingly common – [Blümel et al. 2005][Lin et al. 2002]
- Emphasis:
  - on training individuals for complex or risky procedures
  - on training individuals’ dexterity
- Procedures – Involves a complex set of actions that must be performed in a specific order or under very specific conditions
Introduction

- **Procedure or Decision** training simulator [Ponder et al. 2003]:
  - Emphasizes on training procedure execution;
  - Disregards the evaluation of the user’s dexterity;
  - Low user interaction requirements.

- Most common applications:
  - Maintenance and operation of complex machinery;
  - Life-threatening tasks;
  - Training takes place on virtual prototypes (physical device is unavailable).
Introduction

• **Goals:**
  • Allow virtual training activities to be deployed and executed everywhere – desktop, mobile phones and tablets
  • Maintainability and creation of new training scenarios
    • Allow re-usability

• **Our approach:**
  • Uses X3D and X3DOM as the foundation for a standard web-based solution
  • Augment X3D by presenting additional nodes which facilitates the creation and maintenance of procedure training activities
Related work

- VDT Platform [Blümel et al. 2005]
  - Fraunhofer IFF
  - Generic frameworks to facilitate the creation of procedure training scenarios
  - GUI oriented designer
  - Developed in C++, not targeting web applications.
  - Our approach adopt Blümel’s taxonomy
Related work

- Project GVT [Gerbaud et al. 2008]
  - INRIA
  - Another generic framework
  - Applications on military equipment
    - More than 50 procedures
  - Also developed in C++

- Similar commercial tools:
  - EON Reality and Virtools.
Related work

- Medical simulator in X3D by [Jung et al. 2008]
  - Targeted at training motor and dexterity skills;
  - There is no emphasis on procedure or decision training.
- iVT training tool by [Corvaglia 2004]
  - Allows the creation of procedure training scenarios by custom VRML nodes;
  - Very similar to our approach;
  - However, details are not clearly unveiled;
  - Does not support web-based applications.
Procedure Training Simulators

- Our approach follows the taxonomy proposed by [Blümel et al. 2005]
  - The standard X3D nodes are used to define the Structure layer;
  - New nodes are implemented to meet the requirements of the topmost layers;
Procedure Training Simulators

- **Middle layer** - Defines the interactive objects in the scene and their behaviors

- **Most common approaches used to define behavioral objects** [Mollet et al. 2007]:
  - Sensor actuator-networks;
  - Rule-based systems;
  - Finite state machine;

- **Our approach does allows different techniques to be used**
  - Implementation was focused on FSMs
Procedure Training Simulators

- Topmost layer – Defines the procedure to be executed by the user
- Common alternatives to define dependency relationships:
  - Goals and Constraints [Wasfy et al. 2004];
  - Graphs [Ponder et al. 2003];
  - LORA Diagram [Gerbaud et al. 2008];
  - Petrinets [Lin et al. 2002];
- Our approach allows different techniques to be used
  - Implementation was focused on Petrinets
Proposed Nodes
Nodes for Procedure Simulators

- Abstract `TrainingObjectNode`
  - Represents the second layer on Blümel taxonomy;

  ```
  TrainingObjectNode : X3DGroupingNode {
    MFNode [out] actions : TrainingObjectActionNode
  }
  ```

- Group scene elements in it’s `children` attribute
- `actions`: Attribute with the object’s available actions
- Specializations should determine behavior
Nodes for Procedure Simulators

- Abstract `TrainingObjectActionNode`:
  - The only way to interact with objects in the scene
  - User must choose which action to execute from the list of available actions

```
TrainingObjectActionNode : X3DNode {
    SFBool [in] execute
    SFBool [out] executed
    SFTime [out] executedTime
}
```

- Output attributes allow events to be routed
  - Trigger animations to reproduce the action’s effects
Nodes for Procedure Simulators

- **StateMachineObject**
  - Specialization of **TrainingObjectNode**
  - Object behavior is defined by a Finite State Machine

```plaintext
StateMachineObject : TrainingObjectNode {
    MFString [] url
    MFNode [in] transitionActions :
    StateMachineTransitionAction
}
```

- **url**: Attribute pointing to file with FSM description
- **transitionActions**: Input attribute with actions which are represented by transitions of the state machine.

- Behavior: Select the available actions from **transitionActions** and place them on the output attribute **actions**.
Nodes for Procedure Simulators

- Abstract **TrainingProcedureNode**
  - Represents the top most layer on Blümel taxonomy;

  ```
  TrainingProcedureNode : X3DBindableNode {
    MFNode [out] forwardActions   : TrainingObjectActionNode
    MFNode [out] backwardActions : TrainingObjectActionNode
    SFBool  [in] executeForward
    SFBool  [in] executeBackward
    SFNode  [out] executedAction : TrainingObjectActionNode
    SFBool  [out] isFinished
  }
  ```

- Provides the next **correct** actions for the current procedure
  - It **knows** all the actions from all the existing objects in the scene
- Specializations should determine the procedure definition
Nodes for Procedure Simulators

- **PetrinetProcedure**
  - Specialization of *TrainingProcedureNode*
  - A *transition* in a Petrinet represents an object action

  ```
  PetrinetProcedure : TrainingProcedureNode {
      MFString [] url
  }
  ```

  - *url*: Attribute pointing to file with Petrinet description
  - Behavior: Check the available transitions in the Petrinet and places the associated actions in its output attributes.
Implementation

- Nodes implemented directly into X3DOM
  - No PROTOs available yet
- Motivations:
  - Support on most modern browsers;
    - Requires WebGL
  - Create applications easily:
    - X3DOM + HTML5 + Javascript
Workflow

- **Overview of a Training Scenario Workflow:**
  - Create an X3D scene: Geometric models and animations
  - Create interactive objects by grouping their geometric elements into `StateMachineObject` node’s
    - Create a FSM for each object
    - Insert their actions (transitions of their FSM)
    - ROUTE those actions to appropriate animations
  - Create a training procedure (e.g. Petrinet) and place a `PetrinetProcedure` node in the scene.
    - Create Petrinet with PNML file format
    - Associate the Petrinet transitions with the objects actions
  - Insert the required meta-data to hold additional information
    - Part names, functional description, icons, attached files and others
      - Might be used for instruction or create the application GUIs.
Validation

Finite State Machine

- Currently supports a subset of SCXML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<scxml xmlns="http://www.w3.org/2005/07/scxml" version="1.0" initial="inserted">
  <state id="inserted">
    <transition event="remove" target="removed"/>
  </state>
  <state id="removed">
    <transition event="insert" target="inserted"/>
  </state>
</scxml>
```

Petrinet

- Supports PNML file format
Results
Results

- Training Application
  - User selects an object
  - Available actions are displayed
  - User selects the desired action
  - The application checks if the action is correct:
    - Ok, the action is executed
    - Otherwise, an error message is displayed
  - The user can also request the application to automatically advance or retrocede the procedure
Use Cases

- Real assembly-disassembly procedure
  - Hydroelectric Generating Unit;
  - Large geometric model;
  - Simple training procedure - Sequential;
Use Cases

- Fictitious procedure
  - Car engine disassembly
  - Simple geometric model
    - Mobile devices
  - More complex procedure
Conclusions

- Architecture for creating procedure training simulators in X3D
  - Declarative syntax
  - Easier path for end users
  - Step closer to easily create/edit training applications
    - WebGL + HTML5 + Javascript
  - From desktop to mobile
  - Requires only a web browser
Limitations

- Manually editing the X3D scene files can be labor intensive
  - Embed much of the required workflow into plugins to 3D modeling tools;
- We believe our approach can be used as a guideline for authoring tools
References

- WEB 3D CONSORTIUM. 2012. X3d script node specification iso- 19775-x3dabstractspecification, April.
Thank you!

- Questions?

- Contact:
  mcabral@lsi.usp.br
  www.lsi.usp.br/~mcabral