Simulation of Standard Control Actuators in Dynamic Virtual Environments

Frank Gommlich* Guido Heumer† Arnd Vitzthum‡ Bernhard Jung§

VR and Multimedia Group, TU Bergakademie Freiberg, Germany

ABSTRACT

Realistic behavior of control actuators is important for virtual prototyping applications. We present a systematic approach for modeling such articulated components as described in the European Standard EN 894-3. Control actuators may have several rotational and translational degrees of freedom (DOFs), possibly with discrete lock states. During user interactions, information about the actuators' manipulation is collected and made available to the higher application layers in the form of interaction events. This allows for recording and playback of demonstrated manipulation sequences for many purposes, such as ergonomics evaluations involving virtual humans. The framework uses XML for declaration and is implemented using a freely available physics engine.

Index Terms: I.3.6 [Computer Graphics]: Methodology and Techniques—Interaction techniques; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—[Virtual Reality]

1 INTRODUCTION

Ergonomic analyses on virtual prototypes are a successful application area of immersive Virtual Reality. However, a challenging task remains in the modeling and simulation of interactions with dynamic control elements such as sliders, switches etc. Addressing this, we have developed a framework that simplifies the realization of such *control actuators* in interactive virtual environments. The presented approach builds on the European Standard EN 894-3 [1] and covers all types of control actuators defined therein. This standard defines systematic guidelines for the choice and configuration of control actuators to ergonomically design machinery.

Only few existing research projects specifically deal with modeling and implementation of control actuators in (immersive) virtual environments. For example, Möhring and Fröhlich present a method to enable VR interaction with control actuators in a car interior [4]. However, only automotive-specific actuators are considered. In contrast, we support a broader spectrum of control actuator types (namely those described in EN 894-3). While Möhring and Fröhlich apply a geometry-based heuristic method, we use a physics engine to ensure the realistic behavior of control actuators and a natural hand-object interaction.

In order to describe and implement control actuators we examined several existing standards and approaches such as Collada [3] and X3D [2]. Both X3D and Collada offer rigid body components which allow the definition of joint constraints and other physical properties. In principle, these facilities could be used to define simple types of control actuators, although modeling would be rather cumbersome. Moreover, neither X3D nor Collada support state machines natively and thus are not well suited to model control actuators with discrete lock states. Furthermore, just like real control actuators serve to control the behavior of machines or other appliances in the real world, our virtual control actuators, by throwing *interaction events* of flexible detail, can be used to effect changes in other objects of the virtual environment.

*e-mail: frank.gommlich@informatik.tu-freiberg.de
†e-mail: guido.heumer@informatik.tu-freiberg.de
‡e-mail: arnd.vitzthum@informatik.tu-freiberg.de
§e-mail: jung@informatik.tu-freiberg.de

IEEE Virtual Reality 2009
14-18 March, Lafayette, Louisiana, USA
978-1-4244-3812-9/09/$25.00 ©2009 IEEE

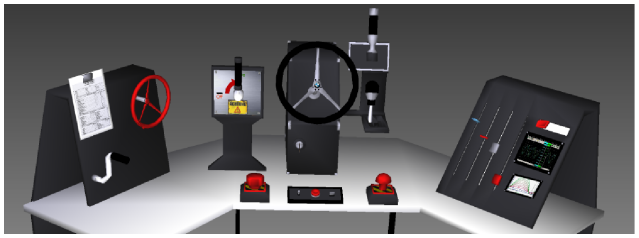

Figure 1: Selection of control actuators defined in standard EN 894-3

2 CONTROL ACTUATORS

A basic control actuator consists of two connected objects, the fitting and the actuator. The latter can be moved by a user or a virtual operator by applying a force. The composition of both is referred here as a control actuator. Multiple actuators with the same fitting are also possible, e.g. keyboards or sound mixer systems.

The framework presented here is based on an extended version of *annotated objects* [5]. Annotated objects and their attributes are represented in a compact and easy to use XML structure. To ensure extensibility of the format, all object attributes are denoted by an all-purpose parameter format consisting of name, type and value, represented by the *param* element. Both, fitting and actuators are declared as separate annotated objects. The fitting part references connected actuators as child objects. The *joint* element of an actuator contains all characteristics of the joint connecting the actuator to the fitting. A joint definition contains at least the type (e.g. DoubleAxisHinge for a joystick, or Slider for a volume control), the specification of the DOF axes and the constraints along these axes.

In case of discrete actuators it is necessary to define at least one discrete state which is represented by an angular or translational offset of the neutral position as well as a direction vector. Also, unsteady states can be specified which are transitional discrete states and automatically lead to a neighboring state. Unsteady states are very useful for realizing complex movement patterns. Figure 2 illustrates this with the gear shift of a car.

All control actuators presented in EN 894-3 can be implemented with at most two DOFs for either translation or rotation. A combination of translation and rotation is not requested by the standard. Therefore it was effectual to divide the standard actuators into four different types according to their DOFs. We called these four joint types *Slider*, *Hinge*, *DoubleAxisSlider* and *DoubleAxisHinge*.

The implementation of control actuators was realized with the freely available Newton dynamics engine. To implement discrete actuators and their individual positions it is necessary for the physics engine to know when the manipulation of an actuator begins and when it ends. During manipulation, the actuator is continuously movable according to its joint constraints. When the actuator

is released, the engine performs a snap to the closest discrete state.If the reached state is unsteady, the snap movement continues to the following state. This action will be repeated until the first steady state is reached or until the actuator is manipulated again. In order to restrict the actuator movements, e.g. to special tracks in a two dimensional layer, we used the automatic collision response mechanism of the physics engine. This mechanism ensures a realistic movement of, e.g., the stick inside the tracks of the fitting seen in Figure 2, without explicitly modeling the permissible pathways.

```
<annotatedobject>
 <param name="name" type="string">buttonFitting</param>
 <param name="child" type="string">buttonActuator</param>
</annotatedobject>
<annotatedobject>
 <param name="name" type="string">buttonActuator</param>
 <joint>
  <param name="jointtype" type="string">slider</param>
  <param name="pinDir" type="array3">0 0 -1</param>
  <param name="minValue" type="double">0</param>
  <param name="maxValue" type="double">0.1</param>
  <discrete-state>
   <param name="name" type="string">deactivated</param>
   <param name="value" type="double">0</param>
  </discrete-state>
  <discrete-state>
   <param name="name" type="string">activated</param>
   <param name="value" type="double">0.05</param>
  </discrete-state>
 </joint>
</annotatedobject>
```

Listing 1: Code example of an emergency button

3 Interaction Events

In addition to the simulation of realistic behavior of control actuators, information about interactions of the user with such objects and the resulting state changes need to be processed. This is addressed in our approach by the concept of *interaction events* which are generated whenever a control actuator changes its state as a result of user interactions. The events are realized as observer pattern, enabling other parts of the system to subscribe as listeners.

The frequency of events and the extent of information in the events is configurable on a per-object basis. This ranges from notification about final states only – sent when user interaction with an actuator ends – to the recording of a complete history during interaction, including the current actuator state in every frame.

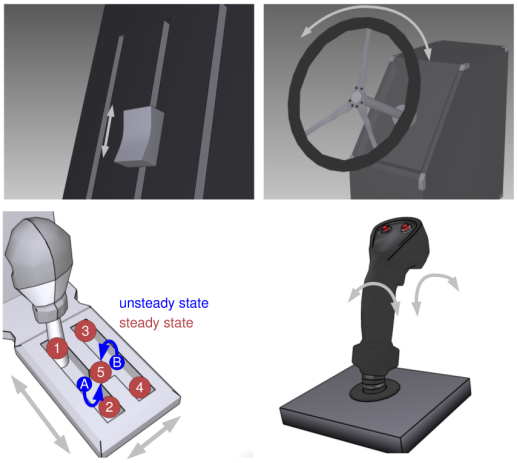

Figure 2: Overview of the different joint types

Interaction events can also be written to files in a custom XML format. Thus, persistence of the interaction is achieved. Listing 2 shows an example of the XML representation of an interaction event. The example contains the reference to a history file (reference-id), timing information (timestamp and duration), old and current state, and information about the discrete states assumed.

```
<event timestamp="0.3" type="actuator" duration="2.3"
       reference-id="CAT-128c534a-...-0011d8e96ab4">
<actuator-states>
<state type="rotational" dof="1" old="0.0">42.23</state>
<state type="rotational" dof="2" old="15.0">45.00</state>
<state type="discrete" old="second">third</state>
</actuator-states>
</event>
```

Listing 2: XML representation of an example interaction event

This enables the reproduction of either the outcome of the interaction (final state), a reproduction of the state change order (intermediate discrete states) or an exact reproduction of the complete interaction (history). The latter allows replay in a purely graphical scene, without the further need for an active dynamics simulation.

4 Conclusion

Due to its extendable nature, the framework for control actuators presented here can be used in many different types of VR applications where movable parts with physically believable behavior are called for. In our particular area of research - which has virtual prototyping as application domain - we use the framework to simulate parts of machinery or other virtual prototypes. Besides, many different use cases are conceivable, such as elaborate VR simulators, ergonomics studies, instructional animations, etc. The only prerequisite is a mechanism to determine forces and torques exerted on the actuators. Further, interactive scene functionality can be realized by subscribing and reacting to interaction events. For example, a car radio sound could be played when the car radio button is pressed. The volume could be adjusted, when the knob is turned, etc.

Control actuators and other movable parts can be easily declared using a custom XML description format. The employment of a physics engine enables realistic response of the movable parts to manipulation by the user hand as well as upon collisions with other scene objects. The full set of control actuators specified in the European standard EN 894-3 is supported by our implementation. Objects in this standard only call for either one to two rotational or one to two translational degrees of freedom. In some situations, e.g. cockpit interior design, it might be necessary to use more complex DOF combinations. Future versions of our framework will be extended in this regard.

References

[1] DIN. *Safety of machinery - Ergonomic requirements for the design of displays and control actuators - Part 3: Control actuators, EN 894-3*. DIN Deutsches Institut für Normung e.V., June 2006.

[2] International Organization for Standardization. *ISO/IEC FDIS 19775-1:2008: Information technology – Computer graphics and image processing – Extensible 3D (X3D) – Part 1: Architecture and base components, 2. Edition*, 2008.

[3] Khronos Group. *Collada - Digital Asset Schema Release 1.4.1 Specification*, 2006.

[4] M. Möhring and B. Fröhlich. Pseudo-physical interaction in a virtual car interior. In *IPT/EGVE Workshop 2005*, 2005.

[5] M. Weber, G. Heumer, H. B. Amor, and B. Jung. An animation system for imitation of object grasping in virtual reality. In *Proc. 16th International Conference on Artificial Reality and Telexistence, ICAT*, pages 65–76. Springer, 2006.