Semantic Information and Local Constraints for Parametric Parts in Interactive Virtual Construction

Peter Biermann, Christian Fröhlich, Marc Latoschik, Ipke Wachsmuth

Laboratory for Artificial Intelligence and Virtual Reality, Faculty of Technology,
University of Bielefeld

Abstract. This paper introduces a semantic representation for virtual prototyping in interactive virtual construction applications. The representation reflects semantic information about dynamic constraints to define objects' modification and construction behavior as well as knowledge structures supporting multimodal interaction utilizing speech and gesture. It is conveniently defined using XML-based markup for virtual building parts. The semantic information is processed during runtime in two ways: Constraint graphs are mapped to a generalized data-flow network and scene-graph. Interaction knowledge is accessed and matched during multimodal analysis.

1. Introduction

One important task when implementing a virtual construction platform like [3] is the definition of new building parts. These part-definitions must cover more than the geometry of the new part. Since the parts are to be employed in virtual construction their definition must include mating geometries, which can be used to connect two parts. If the parts are to be parametrically modifiable, their definition needs to specify such parameters as well as their effect exerted on the geometry of the parts. Such modifications may, e.g., involve scaling, rotation or translation of subparts. More complex modifications should enable to couple different transformations, which can also be used for the simulation of kinematic chains [4].

The interaction with the parts in the virtual environment with speech and gesture requires further semantic information. This information should also be generated from the description of the part. The enrichment of virtual environments with semantic information can be useful in many ways, especially in virtual construction and object manipulation, as we pointed out in [11]. Several research groups have recently made use of such an idea. Lugrin and Cavazza [13] for example use semantic representations to specify certain object behaviors. They have integrated a knowledge layer in an interactive virtual environment, which contains possible processes and actions that can be carried out inside the environment, as well as common sense knowledge for supporting inferences on virtual objects.

Semantic networks are a powerful means to represent data for intelligent virtual reality applications. They allow virtual objects to be enriched by task-specific information, such as physical properties, by way of introducing meaningful semantic relations. Intelligent simulative virtual environments supported by semantic networks are for example presented by Latoschik et al. [12] or Kalogerakis et al. [9]. Semantic information has also been used to design virtual reality applications, as is shown in [10].

One focus in our current research is the interaction via iconic gestures, which is accomplished using a shape description via the IDT model developed by Timo Sowa [15]. The testbed for this interaction is the generation of new parts and interaction for the modification and assembly of building parts with parametric bendable sections.

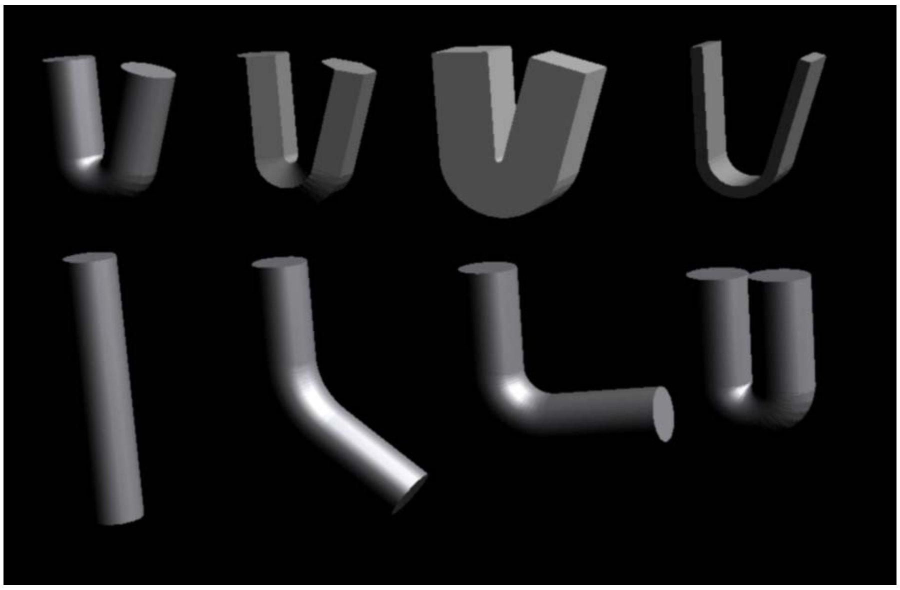
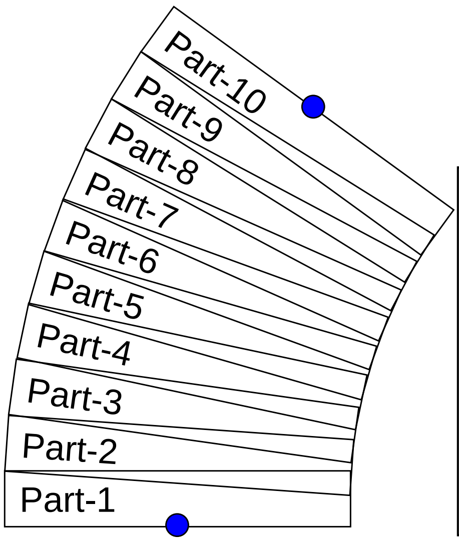


Fig. 1: A bendable elbow with different base geometries and bend-values

For the definition of new parts and their properties – geometric as well as semantic information – we use the XML-definition for parametric parts – the Variant Part Markup Language (VPML)[2]. While other XML-based markup languages for parametric CAD-models – for example the representation presented by Yang et al. [7] or commercial products like CadXML[1] – concentrate on the exchange between different CAD-Tools, our markup language allows a fast and convenient definition of new parts, their parametric transformations and further semantic information. In the following sections, we describe how VPML is extended by template definitions to create meta-descriptions of parametric building parts. These meta-definitions allow, in addition to the specification of parameters, the usage of predefined properties, such as, e.g., base geometries or common attributes. Figure 1 shows various instances of a bendable part-template with different base geometries and bend-values from 0 to 180 degrees.

2. Definition of New Parts

New building parts are defined using the XML-based markup language VPML. For more complex parts, which are designed as parametrical meta-descriptions, a new extension for VPML was developed to describe part templates. These part templates can be instantiated by using predefined XML-attributes to change e.g. the appearance of the part and including further XML-tags which can be inserted in the resulting description. For this meta-definition of such part templates we use an XSLT script [16], which translates the template to the more general VPML-description. This VPML description is used to build the resulting scene-graph structure for the new building part and its subparts including special constraint engines (see Section 3), which realize the constraint movement of the defined subparts. The example of a part template in Section 2.1 is used to generate bendable elbows for different sweep geometries.



```
 <PartTemplate:Elbow name="TubeElbow1“
         segments="10" length="1.5"
         minangle="-180" maxangle="180">
 <SemanticInformation>
  <HasLex value="Tube"/>
 </SemanticInformation>
 <ShapeGeom>
   <Cylinder length="1" radius="0.5"/>
  </ShapeGeom>
 <LowerPort name="Port1"
       type="Extrusionport" … />
 <UpperPort name="Port2"
       type="Extrusionport" … />
</PartTemplate:Elbow>
```

Fig. 2: Illustration of a bendable tube with connection ports (black dots) and its VPML-meta-definition, shown right.

Fig. 2 shows an instantiation of this template for a bendable, cylindrical tube with 10 segments and a connection spot at each end. The meta-definition is translated to a description of a part with 10 subparts whose rotation is coupled to a bend-parameter of the main part. The resulting subparts have a cylinder-geometry and are rotated by a fraction of the bend-parameter depending on the amount of segments. Each segment is connected at the center of the upper end as child of it predecessor in a linear scene-graph tree. To realize a continuous surface at the outer edge of the bended structure the subparts are scaled along the sweep direction commensurate to the bend-parameter. This simple scaling causes an internal overlap of the segments at the inner edge, which is acceptable for a real-time visualization, whereas a correct bending simulation requires an inhomogeneous scaling of the segments.

This is planned by means of Vertex-Shaders (see [8]) for real time processing. A real time feedback process is needed for an interactive modification of the parameters to show the user the effect on the geometry of the parts. After adjusting a parameter, a new polygonal representation of the part is computed using CSG-Operations of the ACIS [6] CAD kernel. The problem of the overlapping inner surfaces can also be handled by the CSG-Operations of this process.

2.1 Processing of Part Templates

```
<xsl::template match="PartTemplate:Elbow">
<Part name="@name">                                   <!-- Begin main part with a bend parameter-->
<Parameter name="BendAngle" min="0" max="180" init="0">
<SemanticInformation>
  <Attribute name="is-bendable" value="true">
  <xsl:value-of select="SemanticInformation">
</SemanticInformation>
<xsl:value-of select=„LowerPort“ >                     <!--insert lower port -->
<xsl:repeat* count=„{@segments}“>                     <!-- start the segments -->
<Subpart name=„ElbowPart-{$count-number*}“ translation=„{@length div @segments}  0.0  0.0“>
 <ParentScaling fixpoint="-{1 div @segments} 0.0 0.0">
  <ScaleMode axes="XYZ" mode="none">
 </ParentScaling>
<ParametricTrf>
  <Rotation name=„ElbowPart-{$count-number*}-Rot“ axis=„0 1 0“
     connect=„BendAngle“ transmission=„{1 div @segments}“/>
  <Scaling name=„ElbowPart-{$count-number*}-Scl“ axis=„1 0 0“
     connect=„BendAngle“ transmission=„{(1 div @segments) + 1.0}“/>
 </ParametricTrf>
 <Geometry scale=„{@length div @segments }  1  1“>
   <xsl:value-of select=„ShapeGeom“ >
 </Geometry>
</xsl:repeat*>
<xsl:value-of select=„UpperPort“ >                     <!--insert upper port at last segment-->
<xsl:repeat* count=„@segments“>                       <!-- finish the segments -->
 </Subpart>
</xsl:repeat>
</Part>
</xsl::template>
```

Fig. 3: VPML-template for a bendable elbow

The part templates for the meta-description of VPML-parts are defined using an XSLT-description. Fig. 3 shows the template description for a bendable elbow which is used in the example before (see Figure 2). The Template defines an additional Parameter (the bend-angle), a number of subparts together with their rotational and scaling constraints and the placement of the two ports at each end of the Tube.

The template is defined using common XSLT-commands[1], which allows the direct translation of the template instances to the complete VPML-code. At compile time the VPML-Code is generated and translated into a scene-graph structure together with an underlying semantic representation.

3. Generating Scene-Graph Structures with Local Constraints

The virtual parts are automatically generated by translating the VPML-definition to a) a scene-graph structure, b) additional constraint engines and c) semantic information, as will be explained below.

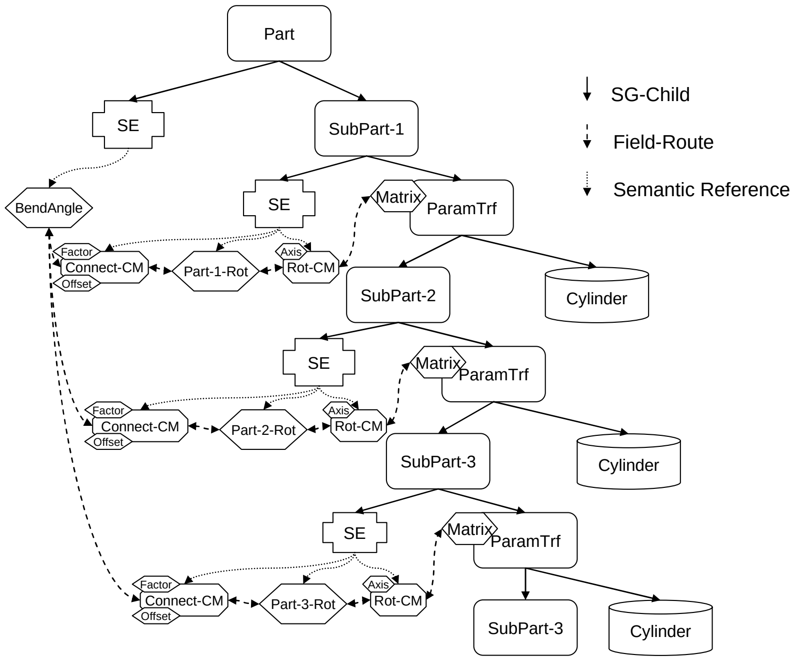


Fig. 4: Scene-Graph with Constraint Mediators

[1] An exception is the xsl:repeat* command which is not part of the regular XSLT 1.0 reference. It is assumed that this command loops for “count”-times, while ‘$count-number*’ is increased in each loop. Simple repeat-loops like this have to be realised, for example by using recursive structures in XSLT. It is simplified here for better readability of the code.

Fig. 4 shows the resulting scene-graph for the part defined in section 2. The Parameters of the part template can define the number of the subparts and their geometry. The parameters of the Connect-Constraint-Mediators, which implement the coupling of the BendAngle-parameter and the Rotation-parameter of the subparts, are derived from the number of instantiated subparts for the elbow. Fig. 4 shows three different levels of graph structures for the building part:

- a common Scene-Graph structure, which encodes geometric shapes and their hierarchy and relative positions. In this example the SG simply consists of group- or transform-nodes, basic geometry-nodes and parent-child relations
- an application graph, which consists of the parameter-fields of the part and subparts respectively, the Constraint-Mediators and the field routes between them
- a semantic structure, which binds the semantic properties of a part/ subpart to its Semantic Entity

A Semantic Entity (SE) is a special scene-graph node, that can reference objects which belong to the definition of the part but are not located in the scene-graph structure. In the example above these objects are the fields, which hold the parameter of each subpart and the constraint engines, which implement the local constraints for these parameters. Instead of adding these objects directly to the scene-graph, as e.g. done in VRML/X3D[5], the Semantic Entities can store different types of objects as key-value pairs for easy referencing. The Semantic Entity also acts as the link between the scene-graph and the underlying semantic structure. This semantic structure reflects all aspects of the previously presented graphs and structures. It is described in the next section.

4. The Knowledge Representation Layer

The virtual as well as the interaction-relevant parts of the real world (in general, simplified user representations of, e.g. the hands) are defined by a knowledge representation layer (KRL). The KRL provides a net of interconnected entity definitions on top of a semantic net base formalism which ties possibly distributed world representations together. The KRL's primary purpose is to provide a common application layer for *semantic reflection*. By this we mean, that the relevant aspects of a given simulation as a whole are mapped on the KRL and made accessible to participating software modules on a unified basis. This approach enables software designers to develop portable and extendable tools since the KRL effectively decouples application design from utilized software modules.

For example, Figure 5 illustrates a snippet of the knowledge base which represents a virtual part that can be bent by user interaction. Both, the actual bending functionality of the part itself as well as the user's interaction can be implemented in a variety of ways, e.g., the part's morphological changes can be implemented using tailored scene graph metaphors, hardware accelerated shading routines, or volumetric rendering approaches. Each of these methods requires its own software module to gain control over the manipulated part during the interaction. An example for a scene graph based approach is illustrated in Figures 5 and 6.

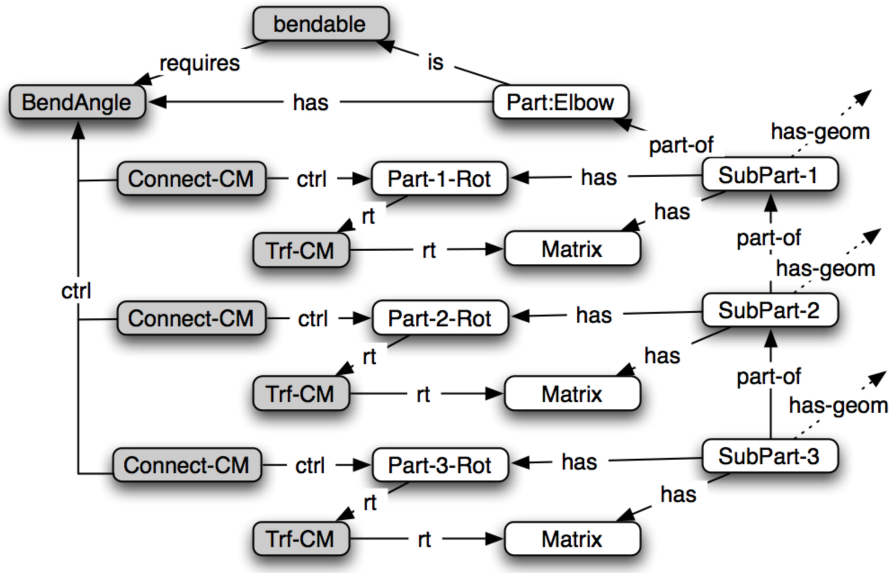


Fig. 5: A snippet of the KRL which describes a bendable virtual part. The overall knowledge structure defines parts, subparts, their attributes and relations which are a generalization of a possible scene graph oriented representation. The grey boxes depict the required concepts which may implement bending as a function of the main part using a specific field-route based simulation module.

From a user's point of view the actual simulation method utilized for a required object or part related functionality is of minor importance. The same is true for the simulation system itself. It just has to provide (e.g., link to) and activate the software module responsible for the actual function implementation. The KRL semantically reflects the required object-related function specification, the specific implementation module's capabilities and features as well as the interaction specification. An interaction is established by automatically matching a user action with a possible object function. This requires both sides to be linked to a common concept which represents the object feature to be changed as illustrated in Figure 5 by the bendable concept.

For the sake of clarity, Figure 5 only depicts the functional decomposition for the virtual part which we will further illustrate here. With respect to the interaction representation, the bendable concept in Figure 5 has additional links to an interaction representation which defines interactions and their required parameters for multimodal interactions which is out of scope for this paper.

4.1 Transformation to Scene-Graph Structures

The chosen semantic decomposition of the virtual part in Figure 5 already depicts a hierarchical object structure which can be easily mapped and transformed to a concrete scene graph for a running simulation. The grey-colored nodes define a more specific application design which requires additional functionality by the interpreting module. Figure 6 illustrates a partial transformation of the knowledge structure in Figure 5. The part-of relations naturally map to a scene graph's child-of relation. The has relation for matrices transform to a new node (ParamTrf) with a matrix attribute which now is inserted in-between the Sub-Part nodes and their geometry as depicted in Figure 6 for SubPart-1.

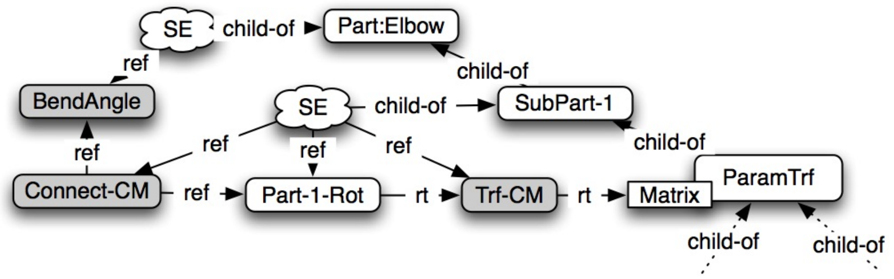


Fig. 6: Transformation of the KRL structure from Figure 1 into an enhanced scene graph structure (see text). Additional Semantic Entity (SE) nodes allow KRL access during scene graph traversal.

The specific grey structures map their control (ctrl) and routing (rt) to object references and field routing in Figure 6. Semantic Entity nodes grant KRL access during scene graph traversal. They enable the development of portable methods since information of the required object-structures can automatically be accessed via the SE. This target structure requires an enhanced scene graph library which supports field-routing as well as referential access between nodes. Even if this example in Figure 5 still shows some module specific structures in the grey nodes, it illustrates the basic idea: While the structures in Figure 5 somehow depend on a specific scene graph based tool, the concept shown in Figure 6 is an important step toward a complete decoupling of the parts' functional, graphical and other simulation related aspects using the KRL. Linked by a common ontology, multiple substructures can be provided which represent a specific implementation for a required function and interaction for specific application provided modules, e.g., for certain graphic, physics or interaction libraries whose basic capabilities will additionally be semantically reflected on the KRL. The KRL's base formalism is already tailored for this approach and it is the central structure of a new simulation core currently under development [12].

4.2 Semantic Information for Multimodal Interaction with Parametric Parts

Multimodal interaction is an integral feature provided by the developed object representation and simulation system. Successful analysis of user input requires knowledge structures that reflect conceptual, lexical, and gesture relevant information about the virtual parts. Possible interactions range from direct interactions to simultaneous speech and gesture utterances. The latter are used to

a) *reference* parts, objects, and places (deictic gestures),
b) *modify* parts' and objects' attributes (mimetic gestures), and
c) describe parts and objects' shapes (iconic gestures), either to *reference* them or to *modify* their appearance.

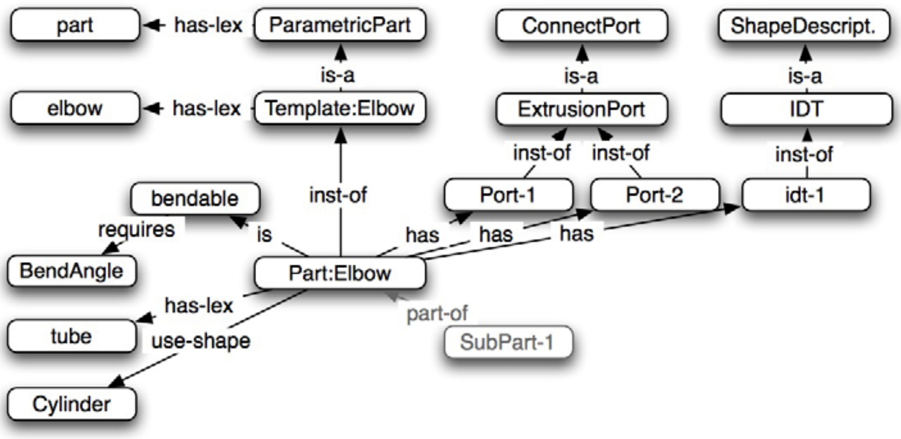


Fig. 7: A different view on the KRL-structure for the Part:Elbow part. The part's concept is augmented with lexical information required for speech-based disambiguation of the part itself using has-lex relations. The bendable concept allows a functional mapping of the interaction. The IDT concept provides the necessary links to a graph-based representation of the part's shape that is matched with a similar representation as generated by gesture analysis.

The VPML-description of the building parts is the source to build up the underlying KRL, which is accessible from the scene-graph via the Semantic Entities. This semantic information is used while interpreting the user's instructions. Figure 7 illustrates a part of the KRL-structure generated from the VPML-meta-description. The semantic information for the new building parts allows to reference the parts in natural language descriptions such as: ”the tube”, “the elbow” or “that part”, etc. by following the has-lex relations during the multimodal analysis phase.

This information is integrated with deictic information generated by the gesture analysis process directly from the user-centric spatial arrangement of the graphical scene. For example, a close temporal relation between articles or nouns of nominal phrases and the climax of an accompanying pointing gesture will sort the set of all objects matching the noun phrase w.r.t. the main pointing direction. Interaction mapping is provided by attribute assignments as depicted for the bendable concept of part Part:Elbow. That is, required user actions are matched to the part's features to identify possible interactions, e.g., adjustment of a given bending parameter is matched to the semantic analysis of the sentence “Change the bend angle of the elbow to 90 degrees”.

In addition, the KRL also includes links to Imagistic Description Trees (IDTs) as shape descriptions specially designed for interaction with iconic gestures. They contain abstract information about the described movement of the user's body. For example, IDTs are used to represent linear path movements, their possibly varying relative length, angles between these linear segments, or symmetry aspects. For detailed information about the IDT-model see [14].

The same base formalism is used to describe a given object's form and extent. During reference resolution, the IDTs generated from the user's gestures are compared with the IDTs of existing parts. Since both are represented in the same graph based formalism, this is done using a graph match approach. For artificially generated man-made objects like the building parts, the IDTs complexity can be assumed to be rather low, hence graph matching causes no performance impacts.

The IDTs of the user's expression is also enhanced with information from the speech channel. Where the vocal instruction contains words, describing the shape of an object – such as “round”, “long” or “cylindrical” – this information is translated and also coded in the resulting IDT.

A visual programming system provides an intuitive rapid prototyping approach for the development of dataflow programs. This system is the core module of the gesture analysis process. It includes various gesture analysis and detection networks required by the application. These dataflow networks receive their input from sensor data recorded from 6DOF markers and the user's hands and detect significant movement features.

5. Conclusion

We have presented a convenient way of defining new building parts and their properties for interactions in virtual construction. The XML-based description format allows the translation into different structures which are needed for multimodal interaction and virtual construction simulation. The direct translation to scene-graph structures enhanced with field routes and local constraint engines enables the simulation of restricted or coupled movements and connections.

The underlying Knowledge Representation Layer provides semantic information for the multimodal interaction, whereas the Imagistic Description Trees allow the comparison of the shapes of different parts or iconic gestures. Altogether this forms a powerful model for various interactions within virtual construction environments.

Acknowledgement

This work is partially supported by the Deutsche Forschungsgemeinschaft (DFG) under grant Wa 815/2.

References

[1] AfterCAD Software Inc. *CadXML: vendor Neutral File Format for CAD Data Exchange.* from http://www.aftercad.com/cadxml.html

[2] Biermann, P. & Jung, B. (2004). *Variant Design in Immersive Virtual Reality: A Markup Language for Scalable CSG Parts*. In: Articulated Motion and Deformable Objects, AMDO-2004, Proceedings. Palma de Mallorca, Spain (LNCS 3179, pp. 123-133). Berlin Heidelberg: Springer.

[3] Biermann, P., Jung, B., Latoschik M. & Wachsmuth I. (2002). Virtuelle *Werkstatt: A Platform for Multimodal Assembly in VR.* In: Proceedings Fourth Virtual Reality International Conference (VRIC 2002), Laval, France, 19-21 June 2002, 53-62.

[4] Biermann, P., & Wachsmuth, I. (2004). *Non-Physical Simulation of Gears and Modifiable Connections in Virtual Reality.* In: Proceedings of Sixth Virtual Reality International Conference (VRIC 2004). Laval, France.

[5] Carey, R. and Bell G. (1997). *The Annotated VRML 2.0 Reference Manual.* Addison-Wesley

[6] Corney, J. (2002). Theodore Lin. *3D Modeling with ACIS.* Paul & Co Pub Consortium

[7] Yang, J., Han, S., Cho, J., Kim, B. and Lee, H. (2004). *An XML-Based Macro Data Representation for a Parametric CAD Model Exchange,* Computer-Aided Design and Applications (CADA), Vol. 1, pp. 153-162

[8] NVIDIA Corp. (2002). *Cg Toolkit, a Developer's Guide to Programmable Graphics*. Santa Clara, CA

[9] Kalogerakis, E., Moumoutzis, N., & Christodoulakis S. (2006). *Coupling ontologies with graphics content for Knowledge Driven Visualization*, In: Proceedings of the IEEE Virtual Reality Conference 2006 (IEEE VR '06), pp.43-50, Virginia, USA

[10] Kleinermann, F., De Troyer, O., Mansouri, H., Romero, R., Pellens, B., Bille, W. (2005). *Designing Semantic Virtual Reality Applications*, In Proceedings of the 2nd INTUITION International Workshop, Senlis, France

[11] Latoschik, M. E., Biermann, P. & Wachsmuth, I. (2005). *Knowledge in the loop: Semantics representation for multimodal simulative environments.* In: Proceedings of the 5th International Symposium on Smart Graphics 2005 (pp. 25-39). Berlin: Springer (LNCS 3638)

[12] Latoschik, M. E., Fröhlich, C. & Wendler, A. (2006). *Scene synchronization in close coupled world representations using SCIVE*. In: International Journal of Virtual Reality, IJVR volume 5, number 3, (pp. 47-52).

[13] Lugrin, J. and Cavazza, M. (2007). *Making sense of virtual environments: action representation, grounding and common sense.* In: Proceedings of the 12th international Conference on intelligent User interfaces (Honolulu, Hawaii, USA, January 28 - 31, 2007). IUI '07. ACM Press, New York, NY, 225-234.

[14] Sowa, T., (2006). *Towards the integration of shape-related information in 3-D gestures and speech*, In: Proceedings of the Eighth International Conference on Multimodal Interfaces (pp. 92-99), ACM Press, New York

[15] Sowa, T. and Wachsmuth, I. (2003). *Coverbal Iconic Gestures for Object Descriptions in Virtual Environments: An Empirical Study*. In: M. Rector, I. Poggi & N. Trigo (eds.): Proceedings of the Conference "Gestures. Meaning and Use.", 365-376, Porto: Edições Fernando Pessoa

[16] W3 Consortium. (1999). *XSL transformations (XSLT).* W3C Recommendation. http://www.w3c.org/TR/xslt.