Towards the next generation of 3D content creation

Gerhard H. Bendels
Inst. of Computer Science II
Computer Graphics
University of Bonn
bendels@cs.uni-bonn.de

Ferenc Kahlesz
Inst. of Computer Science II
Computer Graphics
University of Bonn
fecu@cs.uni-bonn.de

Reinhard Klein
Inst. of Computer Science II
Computer Graphics
University of Bonn
rk@cs.uni-bonn.de

ABSTRACT
In this paper we present a novel integrated 3D editing environment that combines recent advantages in various fields of computer graphics, such as shape modelling, video-based Human Computer Interaction, force feedback and VR fine-manipulation techniques. This integration allows us to create a new compelling form of 3D object creation and manipulation preserving the metaphors designers, artists and painters have accustomed to during their day to day practice. Our system comprises a novel augmented reality workbench and enables users to simultaneously perform natural fine pose determination of the edited object with one hand and model or paint the object with the other hand. The hardware setup features a non-intrusive, video-based hand tracking subsystem, see-through glasses and a 3D 6-degree of freedom input device. The possibilities delivered by our AR workbench enable us to implement traditional and recent editing metaphors in an immersive and fully three-dimensional environment, as well as to develop novel approaches to 3D object interaction.

Categories and Subject Descriptors
I.3.5 [**Computational Geometry and Object Modeling**]: Modeling packages; I.3.6 [**Methodology and Techniques**]: Interaction techniques; I.4.8 [**Scene Analysis**]: Tracking

Keywords
mesh-editing, human computer interaction, HCI, augmented reality, AR

1. INTRODUCTION
An ever growing part of the creative design process in artistic as well as in industrial applications involves digital media aiding the designer. Recent advances in 3D data acquisition and 3D shape interaction allow for efficient generation and manipulation of detailed virtual representations of real-life objects.

In various fields of the entertainment, advertisement and gaming industries, in artistic and leisure applications, the user has to be able to modify pre-existing models or to create new ones. In the automotive and manufacturing industries, manipulation methods must guarantee certain surface properties in order to satisfy criteria posed by later stages in the production process.

In contrast to this, in the fields mentioned above the most important deciding factors are intuitivity, flexibility, efficiency, and ease of use. These eventually determine the acceptance by the user community, the productivity of the design process and thus, the success or failure of a method. This means that interaction metaphors closely resembling traditional real-world 3D human-object interaction has inherently a higher chance of adoption.

This observation motivated us to build an augmented reality workbench that makes it possible to experiment with novel methods for *two-handed, truly three-dimensional* object modelling and manipulation. With our setup we aim to support interactions that use the following metaphor: The user holds the virtual manipulandum in one hand, determining its pose, while the other hand is used to synchronously manipulate the object. We have chosen this type of interaction because it is a well understood process from day to day experience. Moreover, using one's hand is the natural way for pose determination, rather than using a stick, a rod or a space-mouse. The presence of the manipulation tool enables haptic rendering, which adds one more level of realism to the interaction.

Our workbench is based on the following main components:

- **vision based hand tracking**, to enable the pose determination of the manipulated object. Using a visual interface rather than a dataglove based method make the setup cost-effective as well as much more immersive.
- **mesh-editing algorithm**, that allows for simultaneous pose determination and mesh manipulation, overcoming limitations of previous approaches.
- **shutter glasses**, to provide 3D visualization.
- **PHANToM device**[32], to simulate the work-tool.

2. PREVIOUS WORK

In this section we review already existing results in areas which form the basis of our integration: workbenches, vision-based hand tracking and mesh editing.

AR/VR workbenches

Various workbenches have been developed since the beginning of the last decade ([30] [14] [27] [28] [8] [16] [24]). In contrast to other virtual/augmented environments, like CAVEs or Display Walls, workbenches are table-like 3D output devices. The use of shutter or polarized glasses combined with stereo projection provides the user with the illusion of seeing virtual 3D objects on the table. Generally, workbenches have a head-tracking subsystem and the user is allowed to walk around the table and examine the virtual objects from different viewpoints. The objects can be moved by the user with the help of various interaction tools like gloves, spacemice or sticks [14] [16].

The first workbenches have been enhanced in various ways. As the use of different tracking methods can be relatively tethering to the user and also requires tedious calibration methods, research has been conducted to provide visual, non-intrusive means instead of the electromechanical and ultrasonic methods. Several replacements have been developed which were able to realize point-and-select interfaces in real-time with or without the need for active or structured lighting [30] [8] [14] [24].

Considerable research has been done to examine the efficiency of the different interaction methods [22] [29] and develop new ones (e.g. virtual buttons) [8] [27]. Also, the quality of multi-user collaboration has been addressed [1].

Visual hand tracking

The first serious attempt to recognize hand postures was the system of Regh et al. [19]. They registered a 27-DOF 3D kinematic hand model with images from multiple cameras based on point- and edge-like features. Unfortunately, their system was not able to correctly handle self occlusions.

In an "analysis by synthesis" approach, Ouhaddi and Horain [18] tried to maximize the overlapping of a projected hand model area with the area of the detected hand using the downhill simplex algorithm. Their hand tracker was not real-time and was able to track only limited DOFs.

Rosales et al. [20] developed a new stochastic learning scheme (SMA) to learn the mapping function from Hu moments of hand images to hand states. The learning data was captured with a dataglove. The results of the method vary from correct to bad posture reconstruction. An advantage of the method is its ability to compute the uncertainty of the joint angle estimations.

Wu et al. [31] also used a dataglove to capture hand movement and used PCA to reduce the hand state space from 20 to 7 dimensions. In this 7 dimensional space 28 basis configurations were selected. The authors claimed that human hand movements are on linear manifolds spanned between these basis configurations. The results they present seem accurate, however, no timing is given and the tracked movement is rather constrained. Moreover, in [26] it was disputed that the manifolds connecting selected basis configurations are not sufficient to represent all possible hand articulations.

Shimada et al. [17] has built a database of more than 16000 possible hand postures (silhouette contours). The authors developed a contour matching algorithm to search the database in every frame. To achieve real-time performance, the system has been implemented on a 6-PC cluster. Nevertheless, the performance of their tracker is not satisfactorily.

Stenger et al. [25] built a 3D hand model from quadrics. Using projective geometry, they were able to project the contours of the hand model into the images from calibrated cameras. A candidate hand posture was accepted when enough edge pixels were detected along the projected contours. The tracker also utilized an Unscented Kalman Filter. Unfortunately, the system, aiming at tracking a full 27 degrees of freedom, is not apt for real-time applications.

Mesh editing

The classical free-form deformation techniques, based on Barr's approach on regular deformation of solids [4], that were later extended in [23, 9, 10, 11] and others, already lead to powerful modelling methods. They share the big advantage that the complexity of the editing operation is independent of the complexity of the deformed object. Nevertheless, by editing the object through control point manipulation, even conceptually simple operations might require much expertise and be time-intensive.

Addressing the limited intuitivity and flexibility of the above methods, various approaches have implemented *click-and-drag*-like editing operations.

Focussing on meshes with subdivision connectivity, Zorin et al. [33] introduced multiresolution mesh editing, later extended to arbitrary meshes in [13]. The idea of multiresolution editing is to use different levels of detail of the object to perform edits on different scales: Detail edits are performed on finer meshes and large scale edits on coarser meshes representing the same object. The one-ring of the edited vertex defines the region of influence of the edit. Saving the finer meshes as details with respect to the coarser meshes provides for detail preservation during large scale edits.

In the same line of thought, in [7], the multiresolution representation of the object is used to separate details from the underlying base surface. Thereby, details can be cut from one surface and pasted to another.

Although triangle meshes are still the key 3D object representation in Computer Graphics due to their extensive hardware support, the manipulation of unstructured point data gained more and more research attention in recent years. Approaches by Zwicker et al. and Pauly et al. generalize standard 2D image editing techniques to 3D, reconstructing well-known image editing tools transferring multiresolution results to the case of point-sampled geometry. In extension of ideas presented at [12], later enhanced in [6], shapes are modified by defining a so-called zero-region and a one-region. The one-region undergoes the full user-defined translation or rotation rigidly, whereas the zero-region remains fixed and a predefined blending function is used to create a smooth

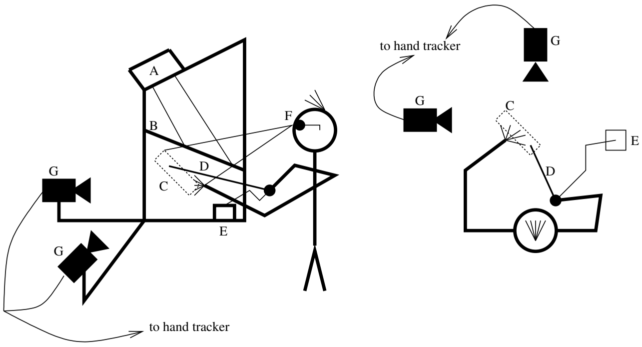

Figure 1: The figure shows the side and top view of our Mesh Editing Workbench. 'A' - monitor, 'B' - semipermeable mirror, 'C' - virtual object, 'D' - virtual tool, 'E' - PHANToM, 'F' - shutter glasses, 'G' - cameras

transition between the two regions.

The above editing metaphors all expect the user first to determine the pose of the object; the transformation is performed *afterwards* with a fixed object. This is partly due to the fact that, at common PC workstations, the mouse is a primary input device, providing only part of the degrees of freedom that would be desirable in a 3D-editing context. Even in [15] where a two-handed editing interface is proposed using two 6-DOF-Trackers, pose determination and transformation are performed consecutively.

This poses a severe limitation to the flexibility of the editing metaphor and considerably increases training times for artists used to work with real life objects and modelling or painting tools. As an example consider the painting of a china (porcelain) vase. This sophisticated operation requires the synchronous pose determination and painting, an option that is also vital for editing operations that require the specification of arbitrary rotations or bending part of an object around another object.

3. SYSTEM SETUP

Figs. 1 and 2 show the assembly of the Mesh Editing Workbench (MEW). The system consists of a monitor ('A' in Fig. 1) mounted above the user's head. The user sees the screen of the monitor in a semipermeable mirror ('B') fastened above her workspace. Stereo rendering in combination with shutter glasses ('F') creates the illusion that the virtual object the user wants to interact with ('C') is placed below the mirror along with the virtual manipulation tool ('D'). However, as the mirror is semipermeable she also sees her own hands.

The object (the edited mesh) is modified by the virtual tool, which is operated by the user's right hand. The virtual tool is implemented with the help of a PHANToM device. This has two advantages: on the one hand, the 6 DOF of the tool are precisely measured by the hardware, on the other hand this adds force-feedback to the MEW, which substantially contributes to the immersiveness of the interaction.

During the editing process the user is able to grab or release the object and actively determine its pose with her left hand. This functionality will be implemented with the help of the visual hand tracking system (see Section 4). The input data for the tracker is provided by calibrated cameras ('G') mounted around the workspace (two cameras in the current setup).

Figure 2: The MEW. Top: Total. Middle: Edit session from behind. Bottom: A possible augmented workspace with a virtual object (simulation).

The MEW makes it possible to implement and experiment with realistic and intuitive two-handed interaction schemes. The separation of the user's hands into a manipulating and a poser hand has also the gratis advantage that a simpler visual tracker should suffice, as there is no self occlusion between the two hands.

4. HAND TRACKING

The performance and the DOF of the hand tracking subsystem decide between success or failure of the MEW. The provided DOF (number and positions) should be adequate to drive the mesh-editing algorithm. Our current manipulation method requires 7 inputs: position, orientation and a grab/no-grab information (see Section 5 for more details). On one hand, this poses a greater challenge than a simple point-and-grab interface. On the other hand it is also substantially simpler than building a full 27 DOF dataglove replacement.

Our idea is to implement a database-based algorithm. These methods have been several times proposed to solve the problem of full DOF hand tracking [20], [17] or [3]. The basic idea behind these methods is to fill a database with a huge number of hand-postures by rendering a hand model from a large amount of possible viewpoints.[1] In each database record, the hand-state vector (joint angles) along with the relative rotation to the camera (view-position) is stored. Then, to retrieve the state and relative rotation to the camera of a real hand in an image, one can use appropriate visual features to index the database. The result can generally be refined locally using analysis-by-synthesis methods, however this can be time-consuming.

Although computationally very expensive, we can still exploit these methods because in our context (with the the limited number of states - grab, no grab, see fig. 3) we don't need the full spectrum delivered by the data base. This drastic reduction of complexity results in a considerable speed-up of the retrieval and a greater robustness of the whole process.

A drawback, however, is that these methods usually assume that the hand is in front of the camera. This means that only rotations relative to the camera are computed. Obviously, this is not enough for us, as we need the full pose including the relative translation of the hand from the camera. A straightforward solution is to sample also the translations of the hand in the workspace.

5. MESH EDITING

The integrated AR workbench described above enables users to simultaneously determine pose and specify object manipulations. As an application, we describe an editing approach that exploits this.

Overview over the editing algorithm

In our environment an editing operation will be structured as follows:
The first step of the algorithm is to separate the object into three disjoint regions: The *handle*, the *deformed area*, and the *fixed region*. These regions can either be defined explicitly by drawing its borders onto the object or by selecting the handle and adjusting a scalar influence parameter, defined as a function of the distance to the handle. If not defined explicitly, the fixed area consists of all vertices whose influence parameter is zero.

[1]Usually, a sphere around the hand-model is sampled to generate these viewpoints.

Then a transformation for the handle is specified by grabbing and dragging the handle with an optional twist around the surface normal. This is performed using the PHANToM[32] device, that allows to simultaneously define these 6DOF. The algorithm then computes the corresponding affine transformation matrix.

The specific layout of the editing operation can then be manipulated using a so-called shape function (see fig. 4). This 1d function determines the influence of the transformation on vertices depending on their distance to the handle / handle border. Depending on the shape function, the transformation is then propagated to the vertices in the deformed region.

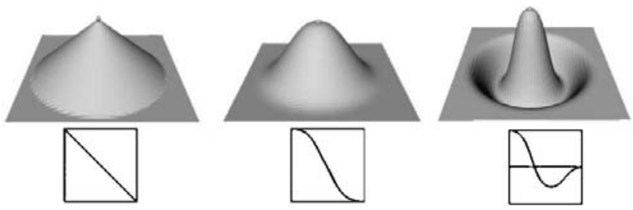

Figure 4: Different shape function settings applied to the same editing operation.

For clarity, we first show how our editing paradigm works on a fixed object and then describe how the synchronous pose determination can be incorporated into the editing method and how it can be used to overcome limitations of previous approaches.

Manipulating a Fixed Object

An atomic editing operation consists of a user defined selection and dragging of a handle (consisting of one vertex or a more complex, rigid part of the object), i.e. a mapping

$$(\mathbf{p}_0, \mathbf{n}_0) \mapsto (\mathbf{p}_1, \mathbf{n}_1),$$

where $\mathbf{n}_0$ and $\mathbf{n}_1$ denote the surface normals at $\mathbf{p}_0$ and $\mathbf{p}_1$ resp. (see fig. 5), whose (signed) length encodes the angle of the optional twist around the surface normal. Following [21], this mapping defines a unique minimal screw motion T, that is composed of a translation by a vector $\mathbf{d}$ and a rotation R around the axis defined by $\mathbf{d}$.[2] All vertices in the handle region undergo the same transformation T. For the vertices in the deformed region the transformation has to be determined. As in [6], we define this transformation for an arbitrary point $\mathbf{p}$ as

$$\mathbf{p} \mapsto (\alpha \odot T)\mathbf{p}. \quad (1)$$

Here, α is the influence parameter depending on the shape function and on the distance of $\mathbf{p}$ to the handle. $\odot$ denotes the scalar multiple of affine transformations, that for rotations is easily understood as rotating around α times the

[2]Please note that for this combination of transformations where the rotation axis is parallel to the translation, the corresponding transformation matrices D and R commute.

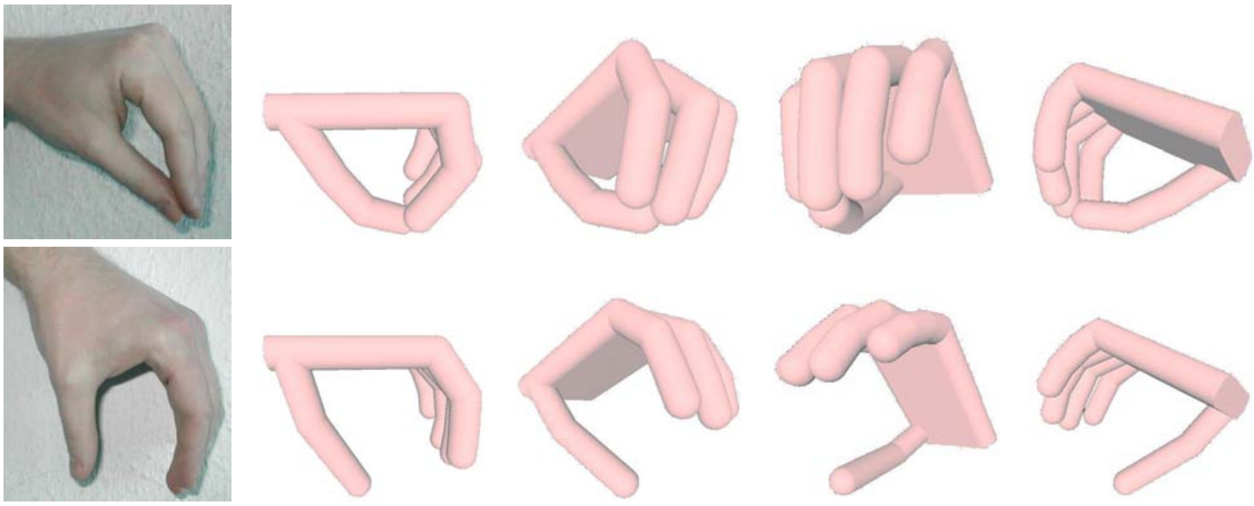

Figure 3: The two gestures we want to use for 'grab' (top row) and 'release' (bottom row). The last four images of the rows show sample images from the generated gesture database.

original angle and for translations is simply a translation by α times the original translation. This way, the mapping in eqn. (1) leads to a smooth, detail preserving transformation of the deformed region, but can also be used to introduce sharp corners, if desired, as was shown in [5]. See fig. 6 for an exemplary editing operation.

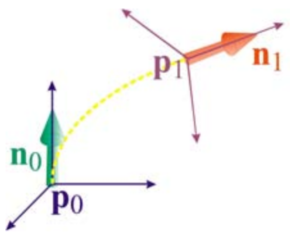

Figure 5: The six degrees of freedom of a screw motion.

Synchronous Pose Determination

The editing algorithm so far describes a useful tool if the model to be manipulated is fixed; in this case, the transformation is dependent on starting and ending pose only. In case of a synchronous pose determination the path of the rigid transformation of the object to be manipulated has also to be accounted for. As an example, one can think of painting a circle on a sphere. Here, starting and ending pose may coincide, yet the transformation is not the identity. Therefore, we track the path of both, the pose determination performed with the video tracked hand and the manipulation performed with the 6 DOF input device, over time.

Let $T_{\text{pose}}(i)$ and $T_{\text{manip}}(i)$ be the transformations defined by the two inputs for a time step i, $i = 1, \ldots, k$. At each time step, these form a combined transformation

$$T_i = [\alpha \odot T_{\text{manip}}(i)] \oplus T_{\text{pose}}(i),$$

where $\oplus$ denotes the *commutative addition* of transformations as defined in [2].[3] Then we have an overall path dependent transformation of

$$\mathbf{p} \mapsto T_k \cdots T_1 \mathbf{p}.$$

Please note, that whereas T_i depends on α, and therefore on the vertex $\mathbf{p}$ itself, the transformations $T_{\text{manip}}(i)$ and $T_{\text{pose}}(i)$ are each constant over the mesh, thereby allowing efficient evaluation.

6. CONCLUSIONS

In this paper we have presented the concept of a novel augmented reality workbench, the MEW. The MEW combines an intuitive two-handed editing paradigm with a three-dimensional interaction environment. The system features force-feedback object manipulation as well as hand-tracked pose determination.

The main feature of our work is that by allowing *synchronous* pose determination and object manipulation, we can reproduce interaction with virtual objects using metaphors artists and designers are accustomed to.

In our current system, hand tracking is still realized by a dataglove subsystem. Therefore, our most important current field of research is the full integration of the video-based hand tracking and the further enhancement of the robustness of the data base retrieval.

In future applications we will include additional functionality for the posing hands, such as virtual button palettes, etc. Further research is also required to robustly detect more than seven degrees of freedom of the posing hand using the

[3] This commutative addition can be thought of as applying both transformations at the same time rather than one after the other. This is achieved, in principle, by applying small parts of the respective transformation interchanged. See [2] for details.

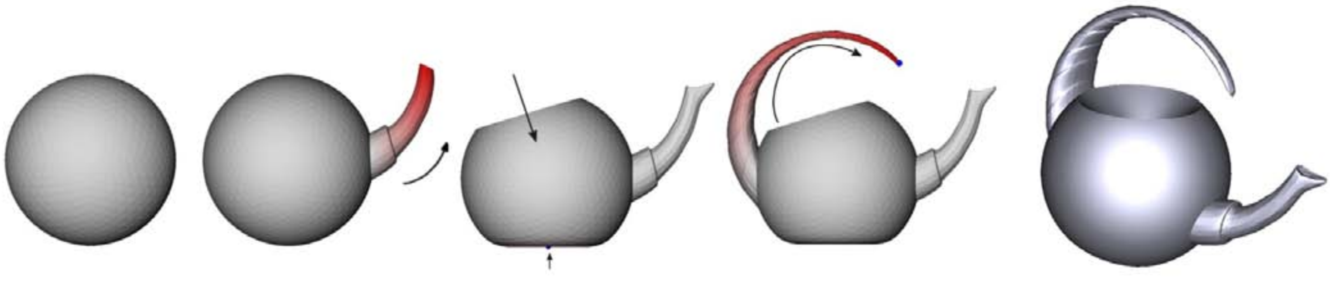

Figure 6: Creating a teapot from a primitive (left) with just a few editing operations. The arrows indicate the modification applied to the handles.

video data from the attached cameras without the need of additional markers.

So far, our implementation is purely vertex-based, the user can only choose vertices as handles and anchors. It would be desirable to allow for arbitrary points on the object surface to be picked.

Acknowledgements

The authors would like to thank Ralf Sarlette for all the help with the build-up of the workbench hardware setup.

7. REFERENCES

[1] M. Agrawala, A. C. Beers, I. McDowall, B. Fröhlich, M. Bolas, and P. Hanrahan. The two-user responsive workbench: Support for collaboration through independent views of a shared space. *Computer Graphics*, 31(Annual Conference Series):327–332, 1997.

[2] M. Alexa. Linear combination of transformations. In *Computer Graphics (SIGGRAPH 2002 Proceedings)*, pages 380–387, July 2002.

[3] V. Athitsos and S. Sclaroff. An appearance-based framework for 3D hand shape classification and camera viewpoint estimation. Technical Report 2001-022, 22 2001.

[4] A. H. Barr. Global and local deformations of solid primitives. In *Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, pages 21–30, 1984.

[5] G. H. Bendels and R. Klein. Mesh forging: Editing of 3d-meshes using implicitly defined occluders. In *Proceedings of the Eurographics Symposium on Geometry Processing 2003*, June 2003.

[6] G. H. Bendels, R. Klein, and A. Schilling. Image and 3d-object editing with precisely specified editing regions. In T. Ertl, B. Girod, G. Greiner, H. Niemann, H.-P. Seidel, E. Steinbach, and R. Westermann, editors, *Vision, Modeling and Visualisation 2003*, pages 451–460. Akademische Verlagsgesellschaft Aka GmbH, Berlin, November 2003.

[7] H. Biermann, I. Martin, F. Bernardini, and D. Zorin. Cut-and-paste editing of multiresolution surfaces. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 312–321. ACM Press, 2002.

[8] O. Bimber. Gesture controlled object interaction: A virtual table case-study. In V. Skala, editor, *WSCG'99 Conference Proceedings*, 1999.

[9] S. Coquillart. Extended free-form deformation: a sculpturing tool for 3d geometric modeling. In *Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, pages 187–196. ACM Press, 1990.

[10] W. M. Hsu, J. F. Hughes, and H. Kaufman. Direct manipulation of free-form deformations. *Computer Graphics*, 26(2):177–184, 1992.

[11] S.-M. Hu, H. Zhang, C.-L. Tai, and J.-G. Sun. Direct manipulation of ffd: efficient explicit solutions and decomposible multiple point constraints. *The Visual Computer*, 17, 2001.

[12] R. Klein. 3d mesh-editing. In *Dagstuhl Seminar: Image Synthesis and Interactive 3D Graphics*, volume 25. Michael Cohen, Heinrich Mueller, Claude Puech, Hans-Peter Seidel, June 2000 – http://cg.cs.uni-bonn.de/docs/publications/2000/dagstuhl-presentation.pdf.

[13] L. Kobbelt, S. Campagna, J. Vorsatz, and H.-P. Seidel. Interactive multi-resolution modeling on arbitrary meshes. *Computer Graphics*, 32(Annual Conference Series):105–114, 1998.

[14] W. Krger, C.-A. Bohn, B. Frhlich, H. Schth, W. Strauss, and G. Wesche. The responsive workbench, 1995.

[15] I. Llamas, B. Kim, J. Gargus, J. Rossignac, and C. D. Shaw. Twister: A space-warp operator for the two-handed editing of 3d shapes. *ACM Transactions on Graphics*, 22(3):663–668, July 2003.

[16] F. P. M. Koutek. The Responsive Workbench Simulator: a Tool for Application Development and Analysis. In V. Skala, editor, *Journal of WSCG 2002*, pages 255–262, February 2002.

[17] Y. S. N. Shimada, K. Kimura. Real-time hand posture estimation based on 2d-appearance retreaval using monocular camera, 2001.

[18] H. Ouhaddi and P. Horain. Hand tracking by 3d model registration, 1999.

[19] J. M. Rehg and T. Kanade. Model-based tracking of self-occluding articulated objects. In *ICCV*, pages 612–617, 1995.

[20] R. Rosales, V. Athitsos, L. Sigal, and S. Sclaroff. 3d hand pose estimation using specialized mappings. Technical report, Los Alamitos, CA, July 9–12 2001.

[21] J. Rossignac and J.J.Kim. Computing and visualizing pose-interpolating 3d motions. *Computer Aided Design*, 33(4):279–291, April 2001.

[22] A. Seay, D. Krum, W. Ribarsky, and L. Hodges. Multimodal interaction techniques for the virtual workbench.

[23] T. W. Sederberg and S. R. Parry. Free-form deformation of solid geometric models. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 151–160. ACM Press, 1986.

[24] T. Starner and et al. The perceptive workbench: Computer-vision-based gesture tracking, object tracking, and 3d reconstruction for augmented desks, 2003.

[25] B. Stenger, P. R. S. Mendonça, and R. Cipolla. Model-based hand tracking using an unscented kalman filter. In *Proc. British Machine Vision Conference*, volume I, pages 63–72, Manchester, UK, September 2001.

[26] A. Thayananthan, B. Stenger, P. H. S. Torr, and R. Cipolla. Learning a kinematic prior for tree-based filtering. In *British Machine Vision Conference*, 2003.

[27] B. Ullmer and H. Ishii. The metadesk: Models and prototypes for tangible user interfaces. In *ACM Symposium on User Interface Software and Technology*, pages 223–232, 1997.

[28] J. Underkoffler and H. Ishii. Illuminating light: An optical design tool with a luminous-tangible interface. In *CHI*, pages 542–549, 1998.

[29] v Pol, R. William, R. Larry, and H. Frits. Interaction in semi-immersive large display environments, 1999.

[30] P. Wellner. Interacting with paper on the DigitalDesk. *Communications of the ACM*, 36(7):86–97, 1993.

[31] Y. Wu, J. Lin, and T. Huang. Capturing natural hand articulation. pages 426–432, 2001.

[32] www.sensable.com.

[33] D. Zorin, P. Schröder, and W. Sweldens. Interactive multiresolution mesh editing. *Computer Graphics*, 31(Annual Conference Series):259–268, 1997.