EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling (2004)
John F. Hughes and Joaquim A. Jorge (Editors)

Constraint Stroke-Based Oversketching for 3D Curves

Timo Fleisch Florian Rechel Pedro Santos André Stork
Fraunhofer IGD
Fraunhoferstr. 5, Darmstadt
{timo.fleisch}{florian.rechel}{pedro.santos}{andre.stork}@igd.fhg.de

ABSTRACT

In this paper we present an extended approach for stroke oversketching. The stroke oversketching technique changes a curve by redrawing parts of it. Finding the part to replace and smoothing the transition is done automatically. Our extension to oversketching constraints the change of the curve to one single side profile leaving the other side profiles of the curve unchanged. Additionally we present the advantage of using this approach in car design following the classic four-side-view drawing approach.

Splines, Line and Curve Generation, Interaction techniques, Virtual reality

1. Introduction

Most designers nowadays still prefer to work with pen and paper. This is due to the fact that on one hand these are the tools they were educated with and on the other hand they are the most intuitive ones. To persuade a designer to use a computer system instead, the hard- and software interface should be similarly intuitive and make use of their trained skills.

Unfortunately Computer Aided Styling (CAS) software is in general not simple to use for designers but requires long adaptation training and an understanding of the math behind the geometries to modify curves and surfaces. Because strokes are usually represented as Bezier Splines, the user has to manipulate the control points to change the curve.

When using a pen as an input device it is much more appropriate to simply redraw parts of curves to refine them. The technique, which allows doing this, is called oversketching (figure 1). This interaction is very close to the sketching behaviour of designers on paper. In a sketch a curve is usually defined by a bundle of oversketched strokes. In our immersive design and modelling system SketchAR, developed within a European research project called SmartSketches, it is possible to define drawing planes cutting a model in any position and project them on the tabletop of the virtual table, the model is displayed on. In this way, designers are able to draw on the tabletop such as on a sketching board and use the technique of oversketching.

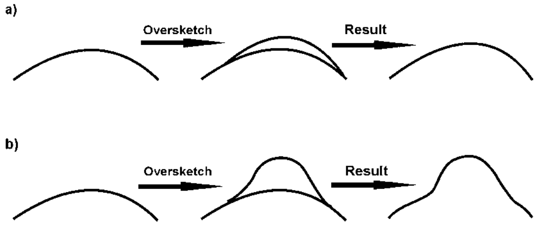

Figure 1: Example of oversketching.
a) scaling factor 0.5 b) scaling factor 1

In this paper an approach to do oversketching in 3d space is presented. As input device the optically tracked *Cyberstilo* [GKS*04] is used in a semi-immersive setup. The *Cyberstilo* is wireless and is used as easily as a pencil. Strokes are drawn directly in 3d space and the oversketching refines all 3 dimensions.

Because precise visual perception and hand interaction of the dimension faced away from the viewers eyes [FMR*03] is difficult to achieve, we extended our oversketching to constrain the changes to the two better controllable and visible axis from the user's viewpoint, without changing the profile of the third one. Please note that in this way the dimensions (two of the three base virtual coordinate system axis) chosen to constraint oversketching depend on the user's actual viewpoint. This form of constraint oversketching can also be used with 2d input devices like tablet PCs to define 3d curves from different views by oversketching.

2. Related work

Creating geometric Spline curves in real-time from direct user input was first described in [BC1990].

Since then there were several systems with stroke input in 2d and 3d. A list of systems for direct 3d input can be found in [FBS*04].

[FB93] were the first to describe constraint modification of mathematic curves. The properties to modify the curve at a designated point include position, tangency and curvature while constraining one ore more of the curve's derivatives at one or more selected parametric points.

Oversketching itself was first presented in [Bau94]. Baudel used a Wacom graphics tablet to sketch and oversketch curves. In the described approach first the replacement interval is located and then the new curve is cut inside with a linear transition interval to smoothen cutting points.

In [GA98] a framework for modification of curves with different representations is presented. Within it, there is a short discussion about modification tools including oversketching, which is mainly a reference to [Bau94].

[ZCG98] describe an approach to modify a NURBS-curve by a sculpturing tool. The tool profile is defined by a NURBS-curve and manipulation is done by replacing the control points of the original curve with the sculpture profile inside a defined modification interval.

A technique to modify a 3d curve without changing the shape from the current camera position is presented in [CMZ*99]. Their approach is to modify a 3d curve with a 2d interface by sketching an additional shadow to the curve. The curve and the shadow can be further refined using oversketching similar to [Bau94].

Oversketching in an immersive environment was presented in [BL03]. After finding the oversketch influence interval the replacement is done based on control points. For smoother transition on the cut points they calculate the transition interval length based on the curves' parametric values but the transition function itself is not further explained.

3. Overview

Our oversketching implementation is integrated in the immersive modelling and design system SketchAR, which is being developed at our department. Details about the setup can be found in [FAS*02] [FBS*04]. The 3d data input for the curve creation comes from an optical tracking system [ART] using the Cyberstilo [GKS*04] a 3d input device used similar to a pen. When the user moves the Cyberstilo the tracker delivers continuous data on the pen position. For computational geometry the software depends on the CAD library ACIS [ACIS]. Although our software works with mathematic geometries, our 3d oversketching approach uses discrete curve points. Converting from Splines to discrete curves and vice versa is done by ACIS, the mathematical background can be found in [PT95].

A brief description of our oversketching method has already been given in [FBS*04]. There two different approaches of oversketching are discussed: merging the curves by replacing the control points and merging by replacing the curves' sample points. Here we discuss the method of replacing the sample points in more detail. Later we extend the idea to constraint oversketching.

3.1 3D-Oversketching

In the first step of oversketching the user has to select a destination curve C_d for the modification. Because oversketching is done in 3d, the user has to use a 3d output device, which allows him to see the curve floating in 3d space. Using the Cyberstilo the user sketches a new curve C_o near the part of the destination curve he wants to change. The algorithm then creates the resulting curve C_r by substituting the oversketched part of C_d with C_o.

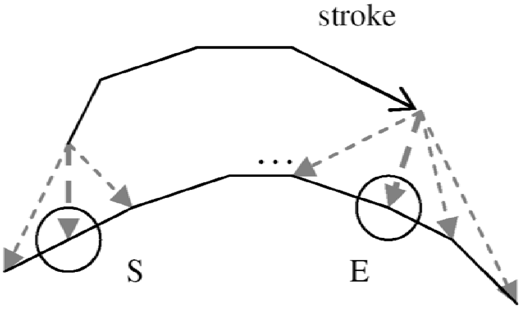

Figure 2: Finding the replacement interval.

The first step of the oversketching algorithm is to find the segment that is substituted with C_o. Because of the curve is represented as a sequence of points the curves are defined as:

$$C_d = (x_0, x_1, \ldots, x_n)$$

$$C_o = (y_0, y_1, \ldots, y_m)$$

$$C_r = (r_0, r_1, \ldots, r_o)$$

For the correctness of the algorithm it is required that the points are equidistant. As seen in figure 2 the segment start x_s and segment end x_e is found by finding the minimum distance between x_j and y_0 and y_m (j=0,..,n):

$$x_s = \min(|x_j - y_0|), j = 0,..,n$$

$$x_e = \min(|x_j - y_m|), j = 0,..,n$$

The resulting modified curve then contains the points:

$$C_r = (x_0,..,x_{s-1}, y_0,.., y_m, x_{e+1},.., x_n)$$

If the segment is simply replaced, there might appear hard breaks in the curve at the edges x_{s-1}, y_0 and y_m, x_{e+1}. To avoid that, the curve is smoothened in a transition interval as described in the next chapter.

3.2 User Customizable Parameters

The oversketching algorithm can be influenced by two parameters to give the user more control over the results:

- scaling factor
- transition interval size

The scaling factor is defined from 0 to 1. It weighs the influence of C_o on C_d. That means: If the scaling factor is 0 the destination curve will not be modified and if the scaling factor is 1 the resulting curve will follow exactly the oversketch curve in the replacement interval.

Because the number of points of the replacement interval and the oversketch curve usually are not equal, we do the scaling for the points $y_0,..,y_m$ scaled from linear interpolated points $i_0,..,i_k$ from C_d inside the replacement interval, where k=m, $i_0=x_s$ and $i_k=x_e$. Be l=e-s the number of points in the replacement interval of C_d. Usually $l \neq m$, therefore for each x_t we have to find an interpolated point i_t on C_d as the scaling origin. i_t lies between two points $[x_w,x_{w+1}]$ on the replacement interval, where $s \leq w \leq e$. Now:

$$w_t = trunc(t\frac{m}{n})$$

and the interpolation factor is inside $[x_w,x_{w+1}]$

$$v_t = t\frac{m}{n} - w_t$$

All i_t are then calculated the following way:

$$i_t = x_{s+w_t} + (x_{s+w_t+1} - x_{s+w_t}) * v_t$$

Now every y_t has a corresponding i_t and the scaling factor can be applied:

$$r_{s+t} = i_t + scale * (y_t - i_t).$$

The schema of how to calculate one scaled point is shown in figure 3.

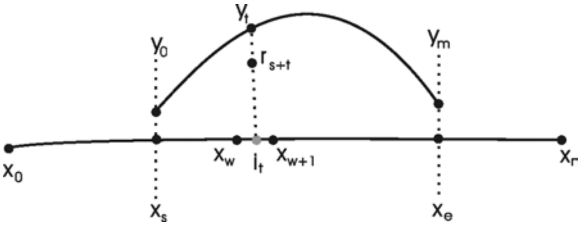

Figure 3: Scaling of the oversketch

For smoother transitions on the edges of the replacement interval we define a transition function and interval. The smoothing only affects parts of the destination curve C_d before and after the replacement interval, C_o is not influenced. The interval size is defined by number of points affected before and after the replacement and is influenced by a user parameter value $V_u = [0..1]$.

The number of influenced points then is:

$$I = n * \frac{V_u}{2}$$

and the transition interval:

$$TI_{start} = [i_{s1}..i_{e1}] \text{ with } i_{s1} = x_{s-1-I},\ i_{e1} = x_{s-1}$$

$$TI_{end} = [i_{s2}..i_{e2}] \text{ with } i_{s2} = x_{e+1},\ i_{e2} = x_{e+1+I}$$

whereas

$$i_{s1} = 0 \text{ if } s-1-I<0$$

$$i_{e2} = n \text{ if } e+1+I>n$$

The transition interval is depicted in figure 4.

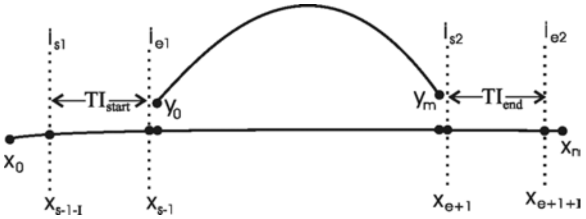

Figure 4: Transition interval calculation

The smoothening in the transition interval is done using the sinus-function. Using a sinus instead of a linear transition produces a more natural looking transition. Using other transition function might be even more suitable, but this has not been the scope of our research yet. The used transition function is as follows:

$$trans(t) = 0.5 * \sin(t + 0.5 * \pi) + 0.5$$

The function range of trans(x) is {0,1} which has to be applied to the destination interval points. The smooth vector for each point i_t in the transition interval is then:

$$v_{smooth}(t) = (y_0 - x_s) * trans(t)$$

Applied to the interval points results in:

$$x(t) = x(t) + v_{smooth}(t)$$

4. Constraint Oversketching

The oversketching method described in the previous chapter allows oversketching the curve in all three dimensions directly, when applied in an immersive environment using direct 3d input devices. The here presented extension to the idea of oversketching adds accuracy by constraining the changes to two coordinates. To perform a constraint oversketch the user has to choose a 2d orthographic view of the profile he intends to change. For example in figure 5 the oversketch is done on the xy-plane. The modification of the curve is now constraint in a way, that the original profile from the xz-plane is not changed.

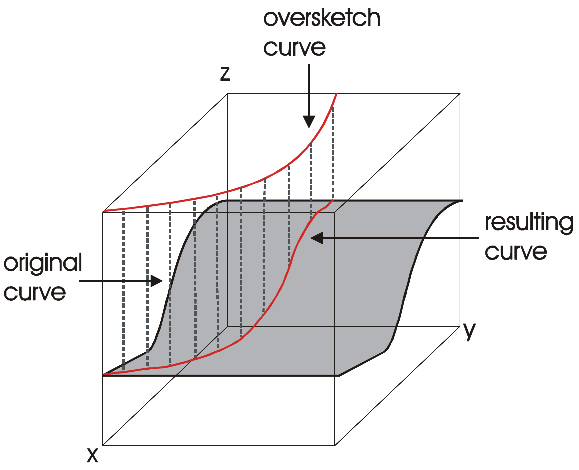

Figure 5: Constraint oversketching principle

The implementation approach of the constraint oversketching is very straightforward and makes use of the CAD library ACIS. In contrast to the technique presented before, the curves need to be represented as B-Splines. Conversion from point sets to B-Splines is also achieved using the CAD library. To combine the oversketch curve with the destination curve the destination curve is extruded to a surface. The extrude path has to be parallel to the constraint plane and can either be vertical or horizontal, as seen in figure 6.

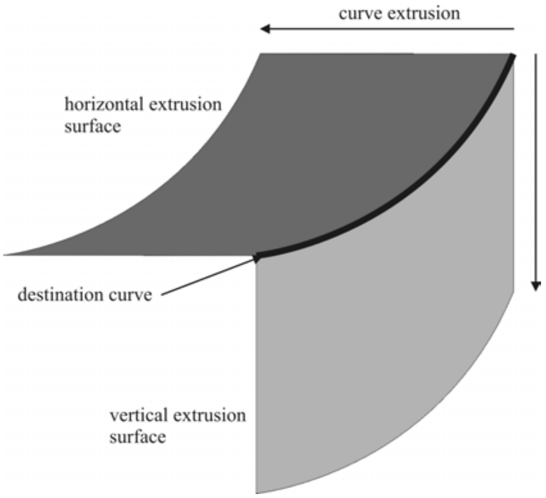

Figure 6: Two possible curve extrusions

The selection between vertical and horizontal is done by the user, by this he selects which side profile stays unchanged.

After the extrusion the oversketch curve is projected on the surface. This projection is then the constraint oversketch result.

5. Application Example

The development of our immersive modelling application is guided by designers from the automobile industry. The concept of constraint oversketched evolved from the analysis of the work process in automobile design. In the common process the design studies start with perspective sketches on paper. After choosing a favourite design from the sketches the car the designer creates technical four side drawings. These drawings give a detailed 2d view of the side, top, front and rear view of the car model.

These drawings then serve as blue print for a CAD engineer inside his CAD/CAS software, where he has to create the 3d curves that match these drawings from all sides.

Using the proposed constraint oversketching technique this task can be performed much more easily. An example sequence is shown in figure 7 and shows the following steps:

- The user starts with sketching a curve of the hood in side view.
- He switches to top view.
- He makes an oversketch defining the side part of the hood.
- The resulting oversketched curve is created.
- The user switches back to 3D view to see the result.

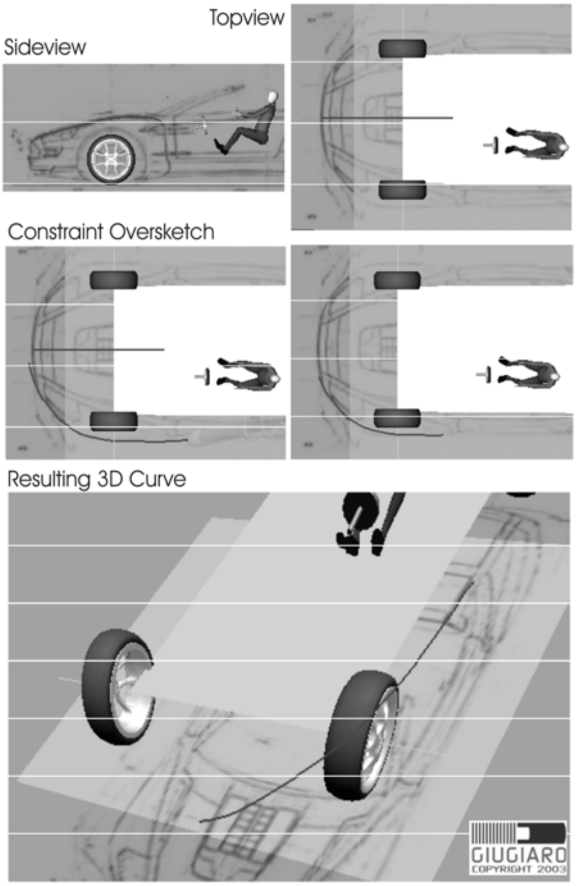

Figure 7: Example of constraint oversketching

6. Future work

In this paper we presented an extension to oversketching to do modifications on the curve without changing another side profile.

Although the presented method already produces good results for certain curves it will fail for others. For example curves having loops cannot be processed correctly.

Furthermore the user still has to specify the direction, the curve will be extruded to a surface. This necessary specification is not very intuitive for the original goal to create a tool for designers without a mathematics background.

The original intent of our research is modelling in immersive environments. Although the constraint oversketching runs in our environment, it is still necessary to switch to a 2d view. We therefore intend to extend the constraint oversketching, by finding the 2d side plane for the algorithm automatically by evaluation of the camera view and up vector without actually switching to a 2d view mode.

Another important task for the future is to accomplish a comprehensive user test of the oversketching technique. The system has already been used by automotive designers but no tests where carried out that give reliably statements about usability and practicability or time advantages when compared to other techniques.

7. Acknowledgement

This work was funded in part by the European Commission Grant #IST-2000-28169 (SmartSketches project).

We also want to thank Italdesign – Guigiaro for providing us with the pictures of the technical drawings within figure 7.

8. References

[ACIS] 3D ACIS Modeler, Spatial Corp., http://www.spatial.com, Westminster, Colorado 80021 U.S.A.

[ART] A.R.T., Advanced Realtime Tracking GmbH, http://www.ar-tracking.de, 82211 Herrsching, Germany

[Bau94] Thomas Baudel. A mark-based interaction paradigm for free-hand drawing. In Proceedings of UIST 94, pages 185-192. ACM SIGGRAPH, 1994.

[FBS*04] T. Fleisch, G. Brunetti, P. Santos, A. Stork, Stroke-Input Methods for Immersive Styling Environments, to be published SMI04, Genua, Italy, 2004.

[FB93] Barry M. Fowler and Richard H. Bartels. Constraint-based curve manipulation. IEEE Computer Graphics and Applications, pages 43-49, September 1993.

[FAS*02] M. Fiorentino, R. De Amicis, A. Stork, G. Monno, “Spacedesign: A Mixed Reality Workspace for Aesthetic Industrial Design”, Proceedings of ISMAR 2002 IEEE and ACM International Symposium on Mixed and Augmented Reality, Darmstadt, Germany, Sept. 30 - Oct. 1, 2002.

[FMR*03] M. Fiorentino, G. Monno, P. A. Renzulli, A. E. Uva, "3d pointing in virtual reality: experimental study", XIII ADM - XV INGEGRAF International Conference on TOOLS AND METHODS EVOLUTION IN ENGINEERING DESIGN, Napoli, June 3th and June 6th, 2003.

[GA98] Cindy Grimm and Matthew Ayers. A framework for synchronized editing of multiple curve representations. In EUROGRAPHICS '98, pages C-31 – C-40, 1998.

[GKS*04] H. Graf, M. Koch, A. Stork, O. Barski. Cyberstilo – Towards an Ergonomic and Aesthetic Wireless 3D-Pen. In Proceedings of IEEE VR2004 Workshop “Beyond Wand and Glove Based Interaction”, pages 51-54, 2004.

[PT95] L. Piegl, W. Tiller, „The NURBS Book“, ISBN-3-540-55069-0, Springer, 1995.

[ZCG98] J. M. Zheng, K.W. Chan, and I. Gibson. A new approach for direct manipulation of free-form curves. In EUROGRAPHICS '98, pages C-327 – C-334, 1998.

[CMZ*99] J. M. Cohen, L. Markosian, R. C. Zeleznik, J. F. Hughes, and R. Barzel. An interface for sketching 3D curves. In Proceedings of the Conference on the 1999 Symposium on interactive 3D Graphics, pages 17--22. ACM Press, Apr. 1999.

[BL03] Bruno F., Luchi M. L., Muzzupappa M., Rizzuti S., The over-sketching technique for free-hand shape modelling in Virtual Reality. In Proceedings of Virtual Concept 2003, Biarritz – France, November 5-7 2003.