EUROGRAPHICS Symposium on Sketch-Based Interfaces and Modeling (2011)
T. Hammond and A. Nealen (Editors)

Reconstructing surfaces from sketched 3D irregular curve networks

S. Morigi †[1] and M. Rucci[1]

[1] Depatment of Mathematics, University of Bologna, Bologna, Italy

Abstract

This paper presents a system for designing free-form surfaces starting from a sketched 3D irregular curve network. By simply dragging a smart-pen device in space, the user draws and refines arbitrary 3D style-curves that define an outline of the desired shape. Unlike previous touch-based sketching systems, the user-drawn strokes can both stay on the model surface to reconstruct parts of an existing object, or freely sketch 3D style-lines of non-existing parts to design new geometry. The wireless smart-pen device is supported by an active stereo acquisition system which makes use of two infrared cameras. For a given set of 3D curves, the system automatically constructs a low-resolution mesh that is naturally refined to produce a smooth surface which preserves curvature features defined by the user on the curve network. The interpolating surface is obtained by applying a high-order diffusion flow. We present an efficient two step approach that first diffuses curvature values preserving the curvature constraints, and then corrects the surface to fit the resulting curvature vector field and interpolating the 3D curve network. This leads to fast implementation of a feature preserving fourth order geometric flow. We show several examples to demonstrate the ability of the proposed advanced design methodology to create sophisticated models possibly having sharp creases and corners.

1. Introduction

The design and development of 3D spatial tracking devices together with digital sketch-based interfaces represents a powerful way to combine the natural and intuitive human expression with the power of computation. However, the potential of these sketch-based systems strongly depends on the effectiveness of the input devices, on the facilities provided by the user interface as well as on the underlying algorithms to create digital 3D models from the input.

The goal of this work is to enable real-time interactive designing/editing of a 3D shape, by sketching a 3D network of curves that approximate the desired shape. To achieve this goal we first developed an acquisition framework for an arbitrary network of 3D curves based on a smart-pen device and an active-stereo system. Second, we provided a shape recovery algorithm to construct the topology of the surface control mesh, and to produce surfaces that interpolate the given curve network preserving curvature features defined on the curve network.

Our work is motivated by the idea that many handly shapes can be described by a sparse collection of characteristic curves [SF98] and that the more natural way the designer has to define control curves is by drawing them onto the shape at any stage of the design process, using a wireless pen device. It is also natural to expect that these curves not only indicate positional constraints, but also curvature features of the surface (smooth shapes/creases/corners). The curve network should captures both the information to produce a more aesthetically pleasing surface.

The notion of a curve network that induces an interpolating surface has been long used in traditional CAD/CAM to model shapes from scratch using free-form surface patches. The idea has been recently combined in [BFL*10] with a curve sketching system providing a Fast Interactive Reverse Engineering System (FIRES) for the acquisition and reconstruction of a virtual 3D model representing an existing physical object which exploits a pen-based active stereo

† Department of Mathematics, University of Bologna, Bologna, Italy

SBIM 2011, Vancouver, British Columbia, Canada, August 5 – 7, 2011.
© 2011 ACM 978-1-4503-0906-6/11/0008 $10.00

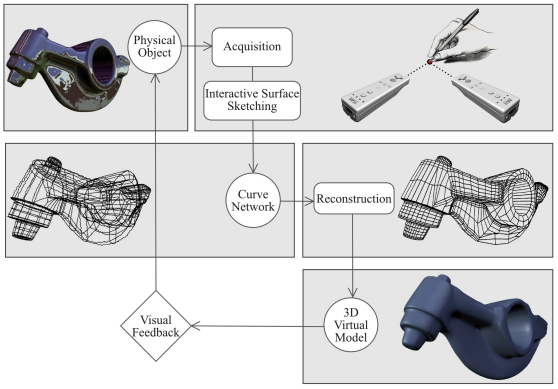


Figure 1: *Fast and Interactive Reverse Engineering pipeline*

acquisition system supported by subdivision surfaces reconstruction. The present work is based on the FIRES framework and introduces the following novel contributions

- A curve sketching interface supported by an innovative, low-cost smart-pen able to draw and select 3D control curves in real-time by simply dragging the pen in space. This introduces a natural way to draw and impose curvature information on the style-lines of the object.
- A simple surface construction procedure based on functional optimization, which, for a given 3D curve network, automatically constructs a smooth surface preserving sharp features defined by the user.

For the free-form surface construction we evolve an initial mesh, which provides a topological structure, by solving a suitable Partial Differential Equation (PDE) flow. Unlike most of the previous free-form modelling techniques, our approach solves fourth-order flow with sharpness constraints for feature preserving. An efficient implementation is provided using two coupled second-order PDEs which solution approaches to the solution of the fourth-order surface diffusion flow, thus avoiding the severe mesh distortions which eventually arise from the direct solution of a high-order PDE. The used PDEs are nonlinear and the geometry is intrinsic, i.e. their solution does not require any explicit parametrization of the surface.

The FIRES updated framework (discussed in Section 2) combines an interactive 3D curve acquisition system (Section 3) with geometry processing techniques (Section 4), to provide intuitive design and editing of surface meshes by means of 3D curve network sketching. The system provides fast editing capabilities to support both the acquisition of physical objects and the editing of non-existing parts of them, which can be easily integrated in a CAD working session in real-time (see examples in Section 6).

1.1. Related work

Research in 3D UIs has addressed both the design of novel 3D input or display devices, and the development of design and/or evaluation approaches specific to 3D UIs. An in-depth survey on new directions in 3D user interfaces is given in [BCF*08]. Much of the early work on 3D user interfaces focused on systems for inferring plausible 3D free-form shapes from visible-contour sketches, which involves the difficulty of interpreting 3D information from 2D input [KH06]. The objective of this research is to set up a low-cost optical tracking system adequate for 3D interaction in an interactive computer aided conceptual design system.

Recently, a system for designing free form surfaces from a collection of projected 3D curves inserted through a 2D line drawing sketching system has been presented in [NISA07]. A real 3D sketching system is instead presented in [WS01] using a projection-based virtual environment. The resulting surface is created stitching together pieces of spline patches with C^0 or G^1 continuity.

The underlying problem is the fitting of a surface mesh which interpolates a given curve network. In general, these methods can be roughly classified into two categories: methods which use smoothly stitching parametric patches, like Bézier, spline patches, or subdivision surfaces [SWZ04], [LHC*07], and, more recent approaches, which construct a smooth surface embedding by applying functional optimization. In this work we followed the latter approach to construct a surface using a surface diffusion flow as in [NISA07] and in [SK01], while considering also curvature information associated at the curves and providing a fast implementation of the geometry update.

2. System overview

The system integrates the 3D curve sketching and the surface construction step into an iterative and incremental process that allows the user to have a real-time visual feedback on the ongoing work. The 3D curve sketching process (see Acquisition in Fig. 1) is supported by an active stereo vision system made of two infrared cameras and (at least) one infrared light emitter mounted on a smart-pen device. The smart-pen 3D position is tracked by the stereo rig and the user can intuitively draw and refine the style lines of the object. The user sketches arbitrary 3D curves, and interactively, at each traced curve, the system adapts the shape so that the sketch becomes a feature line on the model. The process of interactively and incrementally drawing the irregular curve network is called Interactive Surface Sketching (ISS). When a designer defines a shape with 3D curves, it is often the case that these curves indicate the curvature features of the surface. Intuitive and aesthetically pleasing surfaces are obtained by the system using a curve not only as a series of positional constraints but also taking into account the sharpness information given by the user on the characteristic lines.

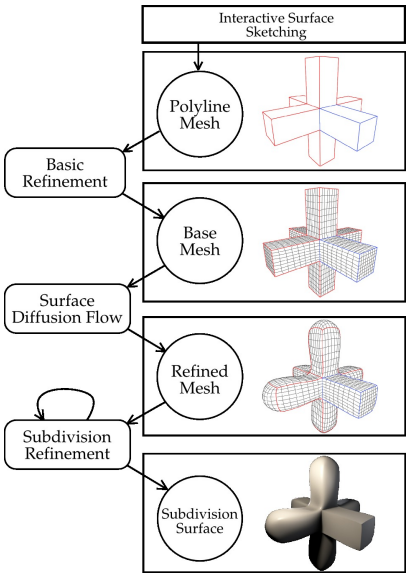


Figure 2: *Multi-step reconstruction process*

Our current implementation allows the user to associate a sharpness value to each sketched curve by simply switching a button on the smart-pen device.

The ISS produces a curve network and, thanks to a set of editing rules, also an associated polyline mesh that is a mesh with faces, vertices and edges augmented with polylines associated to each edge. Irregular curve networks can lead to associated polyline meshes with n-sided, non-planar, and non-convex faces. The polyline mesh is the geometrical representation of the model underlying the curve network created by the user during the process of ISS. However, while the curve network is only a visual representation of the object in terms of spline curves in the 3D space, the polyline mesh contains also topological information. As surfaces of different topology may be compatible with a given curve network the ISS process naturally induces a unique topological structure on the polyline mesh according to the user actions.

The surface construction step (see Reconstruction in Fig. 1) is a multi-step process illustrated in Fig. 2 for a synthetic curve network with sharp (blue colored) curves and non-sharp (red colored) curves. The system first generates a *base mesh* from the polyline mesh, then a surface mesh constructor transforms the base mesh into a *refined mesh* taking into account also the user sharpness constraints, which can eventually be represented as *subdivision surface mesh*. Any of the different model representation forms (curve network/coarse mesh/surface mesh) can be integrated in a CAD system for further processing.

The basic refinement step tasselates each n-sided face in four-sided polygons by following the same splitting rules

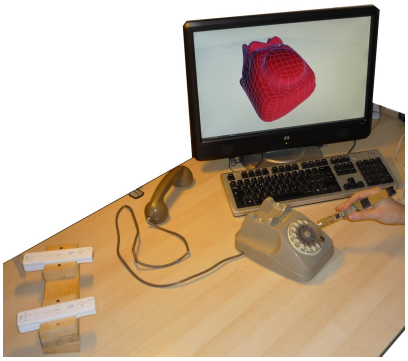


Figure 3: *System set-up for a work session aimed to the reconstruction of an old-fashion telephone*

as of a generalized Catmull-Clark subdivision, i.e., for each face a new vertex is placed at the face centroid and is connected with the midpoint of each edge. The basic refinement can be repeated iteratively as necessary. The base mesh is however very inaccurate because it does not take into account the global shape of the model outlined by the curve network. Therefore, we apply a functional optimization to transform the base mesh into a resulting refined quadrilateral mesh which interpolates points on the polylines and well represents the shape defined by the curve network. Eventually, the resulting mesh can be refined by a few steps of a subdivision scheme that produces a smooth surface interpolating or approximating the given curve network.

3. 3D curve sketching system

The acquisition framework, illustrated in Fig. 3, consists in a set of techniques that, combining a custom designed smart-pen and a stereo optical tracking system, allow the user to draw 3D curves in space, to navigate the scene and to easily interact with the system.

The smart-pen is provided with four collinear IR led emitters, but, unlike the active-pen device introduced in [BFL*10], is further equipped wih a 3-axis accelerometer, a 3-axis gyroscope, four mode buttons, a vibrating motor and a speaker, and it has been designed and realized for satisfying the main requirements in terms of 3D tracking and Human Computer Interaction. Its wireless capability, small dimensions and low-weight allow to move the pen naturally in the workspace. The inertial sensors gather information about the acceleration and orientation of the device and, together with the 4 IR led emitters, provide the hardware support for the tracking system. The 4-mode buttons equipped on the device allow to interact with the system without the need of using the keyboard and mouse. These characteristics allow to maintain the attention and effort onto the object to be reconstructed.

The stereo optical tracking system exploits commercial

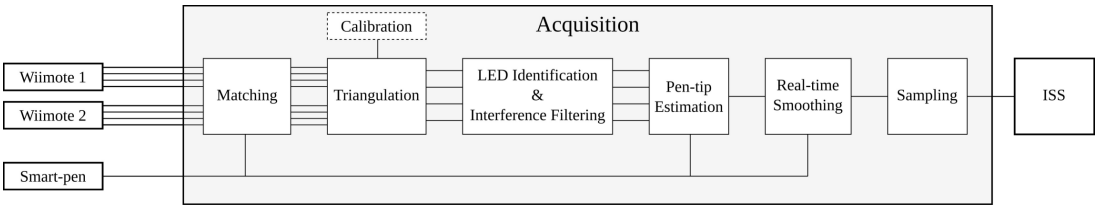


Figure 4: *3D curve acquisition pipeline.*

infrared cameras (available in the Nintendo Wii Remote), which are equipped with a 128×96 IR monochrome camera that includes a built-in processor capable of tracking up to 4 moving IR sources, and, with further processing, provides an image plane (u,v) with a virtual resolution of 1024×768 pixels. The wiimotes were chosen because of their wide availability, their accessible price, and their technical specifications that allow us to perform a real-time 100Hz tracking of four IR led emitters. Both the wiimotes and the smart-pen communicate with the software layers via a bluetooth interface.

The raw data gathered both from the smart-pen and from the wiimote cameras is processed according to the acquisition pipeline described in Fig. 4. The projected led image points of both cameras are first matched and then triangulated in order to reconstruct at most four 3D point locations representing the position in space of the leds mounted on the smart-pen. After identification the 3D points are used to estimated the position of the pen-tip. The sequence of 3D pen-tip positions traces a curve that is shown to the user after a real-time smoothing and sampling step.

Using the sensor inertial data, the pen orientation is estimated and used by the matching process to correctly order the image points on the camera plane. The triangulation step then outputs the coordinates of the 3D points computed from the ordered pairs of corresponding image points. Calibration must be performed before each acquisition session.

The gyroscopic data from the smart-pen is combined with the vision system data in a Kalman filter to detect the correct pen orientation, used to compute the pen-tip position. The smart-pen sensor's accelerometric data is also useful in the estimation of the pen-tip location when more than two leds are occluded and the pen-tip is hidden to the stereo vision system. In these cases, the pen-tip position is obtained by integrating the pen acceleration in time starting from the last known position computed by the vision system.

Due to the limited resolution of the IR cameras, the pen-tip position is in general perturbed by noise. We exploit sensor-fusion techniques by combining the pen-tip position estimated using the stereo vision data with the accelerometric data from the smart-pen sensors to build a simple and effective Kalman filter.

4. Surface construction from 3D curve network

The approach we follow for surface construction is based on the surface diffusion flow

$$\triangle_{\mathcal{M}} H = 0, \tag{1}$$

where $\triangle_{\mathcal{M}}$ is the Laplace-Beltrami operator and H the mean curvature, with magnitude equal to the sum of the principal curvatures k_1 and k_2. Equation (1) can be derived as a simplification of the Euler-Lagrange equation resulting from minimizing the total curvature functional

$$\int_A k_1^2 + k_2^2 dA, \tag{2}$$

which leads to a minimal energy surface. The idea in fact is to produce the simplest shape which interpolates a given curve net. Moreover, since constant mean curvature surfaces satisfy equation (1), important basic shapes as spheres, cylinders and minimal surfaces with $H = 0$ can be reconstructed.

The resulting surface has to satisfy both geometric constraints, given by a set $\overline{X_0}$ of points on the 3D curve network, and sharpness constraints associated at each given curve, while preserving the topology defined by the polyline-mesh. The process of sharpness tagging is performed by simply selecting the curves on the object and a sharpness value.

In this section we present the construction process for a mesh M which represents a piecewise-linear approximation of a smooth two-dimensional manifold $\mathcal{M}$ embedded in $\mathbb{R}^3$, and we denote by (Ω, X) a chart of $\mathcal{M}$, where $\Omega \subset \mathbb{R}^2$ is an open reference domain and X is the corresponding coordinate map, that is the parametrization of $\mathcal{M}$ at a given point.

Let X_0 be an initial surface which interpolates the set $\overline{X_0}$ of points on the 3D given curves and preserves the topology defined on the base-mesh, and let $\overrightarrow{H}(X) = H(X)\overrightarrow{N}(X)$ be the mean curvature normal vector field. Then we solve a global optimization problem by applying directly to the coordinate maps X the following fourth order flow

$$\frac{\partial X}{\partial t} = \triangle_{\mathcal{M}} \overrightarrow{H} + \lambda(X - \overline{X_0}), \quad X(0) = X_0, \tag{3}$$

where λ is a positive parameter which controls the effect of the data fidelity term that places positional constraints on all vertices of the 3D curves.

The construction method is based on a preliminary step (named basic refinement in Fig. 2) where we construct a sufficiently refined mesh X_0, which includes $\overline{X_0}$, by tessellating each polygon in the base mesh following the same splitting rules of the generalized Catmull-Clark subdivision and iterating so that each n-sided face is subdivided into $16n$ quads. The newly introduced vertices move according to (3) towards a minimal energy surface which satisfies the given geometry and feature constraints.

We propose a splitting strategy for solving the fourth order flow (3), which involves two coupled second order PDEs. The first PDE diffuses the mean curvature normals on $\mathcal{M}$ preserving the sharp features defined on the curve network, and the second PDE refits the parameterization X according to the computed mean curvature distribution. In the first step we solve:

$$\frac{\partial \overrightarrow{H}}{\partial t} = \triangle^w_{\mathcal{M}} \overrightarrow{H}(X), \quad H(0) = H_0. \tag{4}$$

where $\triangle^w_{\mathcal{M}}$ is the weighted Laplace-Beltrami, whose weights are related to the sharpness of the curve network.

In the second step the obtained mean curvature vector field $\overrightarrow{H}$ is used to solve:

$$\triangle_{\mathcal{M}} X = -H\overrightarrow{N}(X) + \lambda(X - X_0), \tag{5}$$

where the data fidelity term forces the fitting of the $\overline{X_0}$ points on the 3D curve network.

To create sharp features along the curve network, a weighted Laplace-Beltrami operator is introduced in (4). The weights depend on a similarity measure between surface patches derived by the sharpness constraints along the curve network. The mean curvature diffusion is then penalized where creases and corners should be reproduced.

To this end, the interactive sketching process allows the user to specify a sharpness measure for each 3D curve of the curve network. Tipically, it could be 0 or 1, where 1 indicates a sharp feature. The mesh edges constructed along sharp curves are labeled as sharp edges. Thus curves with vanishing sharpness values lead only to geometric constraints on the vertex positions, while curves labelled as sharp, lead to both vertex and curvature constraints. Let $S(X_i)$ be the sharpness of the vertex X_i defined as the sum of the number of adjacent sharp edges. The vertex X_i will represent a corner if $S(X_i) > 2$, a crease if $S(X_i) = 2$, or a dart if $S(X_i) = 1$.

Our proposal of similarity weight functions, based on sharpness values, defines the weights as follows

$$\begin{array}{ll} W_{ij} = & \frac{1}{Z(i)} e^{-D(X_i,X_j)/\sigma^2}, \\ D(X_i,X_j) = & |S(X_i) - S(X_j)|, \quad j \in N(i), \end{array} \tag{6}$$

where $N(i)$ is the set of 1-ring neighbor vertices of vertex X_i, and $Z(i)$ is the normalizing constant $Z(i) = \sum_j e^{-D(X_i,X_j)/\sigma^2}$. The parameter σ controls how much the similarities of two patches are penalized. Larger σ gives results with sharper features. By using (6) the contribution of the vertices with different curvature features in $\triangle^w_{\mathcal{M}}$ is strongly penalized. Moreover, this is a rotationally invariant measure.

In the rest of this section we theoretically justify the two-steps construction method, which approaches to the solution of the fourth-order PDE (3) representing a surface diffusion flow on $\mathcal{M}$, when $\lambda = 0$.

Theorem 1 *Let $\overrightarrow{V}(X) := \Delta_{\mathcal{M}} H(X)\overrightarrow{N}(X)$. Discretizing (3) in time, with time-step dt, and $\lambda = 0$, we get*

$$X^{n+1} = X^n + dt\,\overrightarrow{V}^{n+1}(X). \tag{7}$$

Solving the system of two second order PDEs

$$\begin{array}{l} \frac{\partial \overrightarrow{H}}{\partial t} = \triangle_{\mathcal{M}} \overrightarrow{H}(X), \\ \triangle_{\mathcal{M}} X = \overrightarrow{H}, \end{array} \tag{8}$$

that is, evaluating the corresponding time discretizations

$$\begin{array}{l} \overrightarrow{H}^{n+1} - \overrightarrow{H}^n = dt \triangle_{\mathcal{M}} \overrightarrow{H}^{n+1} \\ \triangle_{\mathcal{M}} X^{n+1} = \overrightarrow{H}^{n+1}, \end{array} \tag{9}$$

produces a sequence of iterates $\{X^{n+1}\}$ which converges to the solution of (7), that is to the solution X^* of the fourth order surface diffusion flow (SDF) (1) on $\mathcal{M}$, where M is the piecewise linear representation of $\mathcal{M}$.

Proof We recall from basic differential geometry the relation

$$\overrightarrow{H} = H\overrightarrow{N} = \triangle_{\mathcal{M}} X. \tag{10}$$

At time step $n+1$, by replacing X given by (7) in (10), we have

$$\overrightarrow{H}^{n+1} = \triangle_{\mathcal{M}}(X^n + dt\,\overrightarrow{V}^{n+1}(X)),$$

that is

$$\overrightarrow{H}^{n+1} - dt \triangle_{\mathcal{M}} \overrightarrow{V}^{n+1}(X) = \triangle_{\mathcal{M}} X^n$$

which leads to the first equation in (9). The second equation in (9) which relates position X with mean curvature vectors $\overrightarrow{H}$ follows from (10). □

5. Discretization

The surface reconstruction algorithm iterates on the two steps implementing the PDEs (4) and (5), named in the sequel STEP 1 and STEP 2, converging to a mesh approximating the given 3D curve network and preserving the sharpness features defined by the user. From our experimental work we tuned up the maximum number of iterations to be between 5 and 10.

The space discretization on M of the Laplace-Beltrami $\Delta_{\mathcal{M}}$ used in (5) applied to the vertex coordinate components X, is the connectivity matrix $L \in \mathbb{R}^{(|X|\times|X|)}$ which is more conveniently decomposed as $L = DL^s$, where D is the diagonal matrix with $D_{ii} = 1/(2A_i \sum_{j\in N(i)}(\cot\alpha_{ij} + \cot\beta_{ij}))$, L^s

is the symmetric matrix with elements

$$L^s_{ij} = \begin{cases} -\sum_{j\in N(i)} w_{ij} & i = j \\ +w_{ij} & i \neq j, j \in N(i) \\ 0 & otherwise \end{cases} \qquad (11)$$

where $N(i)$ is the set of 1-ring neighbor vertices of vertex X_i, and the weights w_{ij} are positive numbers and satisfy the normalization condition $\sum_{j\in N(i)} w_{ij} = 1$. The weights w_{ij} are "local", thus the summation in (11) is "local", and they are chosen as in [MDS*02], as

$$w_{ij} = (\cot\alpha_{ij} + \cot\beta_{ij}), \qquad (12)$$

where α_{ij} and β_{ij} are the two angles opposite to the edge in the two triangles sharing the edge (X_j, X_i).

We discretize $\triangle^w_{\mathcal{M}}$ on M the *weighted Laplace-Beltrami operator* used in (4), analogously to (11) with weights w_{ij} replaced by

$$w_{ij}W_{ij}, \qquad (13)$$

where w_{ij} is defined as in (12), while W_{ij} depends on a similarity measure between ith and jth vertex and are given by (6).

Since the mesh consists of quadrilateral faces, relation (10) is used to compute curvature normals $\overrightarrow{H}$, where the computation at each vertex involves a local triangulation suitably built around the vertex itself.

Considering a uniform discretization of the time interval $[0,T], T > 0$, with a temporal time step dt, then (4) can be discretized on time using the forward finite difference scheme which yields a first order scheme in time. Explicit time-stepping schemes are easily computable for every time-step dt, but the stability condition determines an upper bound for the time-step dt that guarantees stability of the evolution. The discrete elliptic operator of a fourth order problem is known to be characterized by a condition number $O(h^4)$, where h indicates the grid size. To ensure stability of an explicit discretization we would be led to a severe restriction of the type $dt \leq Ch^4$ for the time step size dt [SK01]. Explicit methods therefore are computationally expensive.

Our two-step method reduces to the solution of two second order PDEs, and we used implicit time-stepping scheme in (4) which allows much larger time steps. Numerical experiments show that time steps of the order of the spatial grid size are still feasible with respect to the stability of the approach.

The PDE model (3) is then fully discretized and solved by iterating the alternate solutions of STEP 1 and STEP 2, as follows

STEP 1 $(I - dtL_w)\overrightarrow{H}^{n+1} = \overrightarrow{H}^n, \quad n = 0, 1, .., n_{MAX}$
STEP 2 $min_X \|LX - \overrightarrow{H}\|_2^2 + \lambda\|X - \overline{X_0}\|_2^2.$

The alternating iterations represent a key aspect of the algorithm, as the initial rough approximation H_0 is improved at each step, providing a better accuracy to the entire process.

Decomposing L_w as $L_w = DL^s_w$, where L^s_w is the symmetric matrix derived from (11) with weights (13), and considering that W_{ij} in (6) are symmetric, then STEP 1 can be rewritten as the following symmetric definite positive system

$$(D^{-1} - dtL^s_w)\overrightarrow{H}^{n+1} = D^{-1}\overrightarrow{H}^n,$$

which is solved at each step by a preconditioned conjugate gradient iterative solver [Saa03]. The resulting mean curvature normal vector field is then plugged into the constrained least square problem in STEP 2. The minimization problem in STEP 2 is rewritten as the following linear least-squares problem,

$$min_u \|Au - b\|_2, \qquad (14)$$

where

$$A = \begin{pmatrix} L \\ C \end{pmatrix}, \quad u = \begin{pmatrix} X \\ P \end{pmatrix} \quad b = \begin{pmatrix} \overrightarrow{H} \\ X_0 \end{pmatrix} \qquad (15)$$

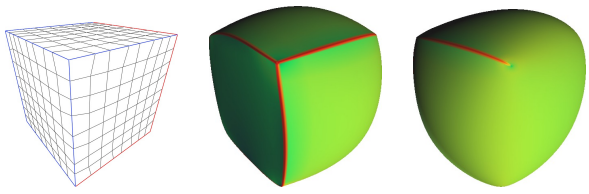
$A \in \mathbb{R}^{(|X|+|\overline{X_0}|)\times|X|}$, $C \in \mathrm{R}^{(|\overline{X_0}|\times|\overline{X_0}|)}$ is the positional constraint matrix, rearranged as an Id matrix, b is the right-hand side, and u is the unknown vector, with the matrix P containing the $\overline{X_0} \subset X$ constraint vertices.

Solution of (14) is efficiently computed by applying the iterative conjugate gradient solver to the system of normal equations $A^TAu = A^Tb$ associated to (14), also known as CGNR method [Saa03]. The CGNR algorithm is not only a residual minimizing method, but also an error-reducing method and is very well suited for solving large linear systems in a number of iterations which is much smaller than the dimension of the matrix. The dominating computational work for large-scale problems is the evaluation of matrix-vector products; each CGNR iterative step requires the evaluation of one matrix-vector product with A and one with A^T. Unlike direct linear solvers, we remark that these matrices are not explicitly formed. Moreover, since A is given by (15), and L is symmetric, the matrix-vector product is split into two parts, and considering also the special form of C we can conclude that the computational effort required is really negligible.

The construction algorithm implemented by alternating STEP 1 and STEP 2 is a global process which involves the solution of sparse but large linear systems whose dimension increases at increasing resolutions of the tesselation. In order to further improve the system performance we applied STEP 1 and STEP 2 to a medium resolution base-mesh X_0. A finer resolution mesh is eventually obtained, after the Surface Diffusion Flow, by applying a few subdivision steps of a Catmull-Clark scheme, which also preserves sharp features by ad hoc subdivision refinement rules.

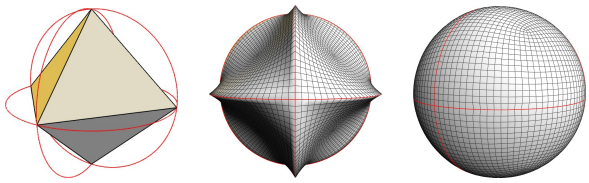
6. Experiment results

In this section we provide some experiments to show the characteristics of our RE system. The system implementation is written in C running on the Linux platform. All the



(a) (b) (c)

Figure 5: *The reconstruction of sharp features on a cube-shaped synthetic curve network.*



(a) (b) (c)

Figure 6: *The reconstruction of a sphere starting from three orthogonal circumferences.*

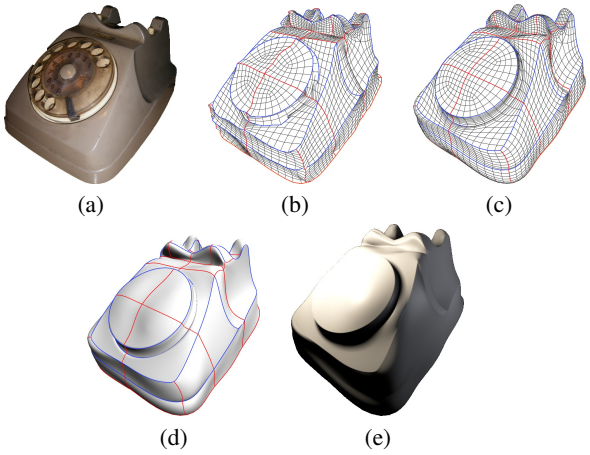
test examples were performed on an Intel Core i7-720QM Quad-Core machine, with 4GB RAM and Nvidia G335M graphics card. The smart-pen device, togheter with the acquisition system described in Section 3, allow the acquisition of 3D points with an error in the range $[-2;2]$ mm at a distance of 1m from the camera.

The outcome of the STEP 1 in the Surface Diffusion Flow depends on four parameters that in our examples revealed to be rather constants, indipendently on the mesh. In particular, for optimal results the data fidelity parameter λ in (3) is set to be 1. The similarity parameter σ in (6) has been tuned to the value 0.2. The time-step dt is in the range [0.1,1.0] and $n_{MAX} = 5$.

Example 1: In the first example the reconstruction capabilities of our RE system are evaluated on three synthetic curve networks: the cross-shaped model in Fig. 2, the sphere in Fig. 6 and the cube in Fig. 5.

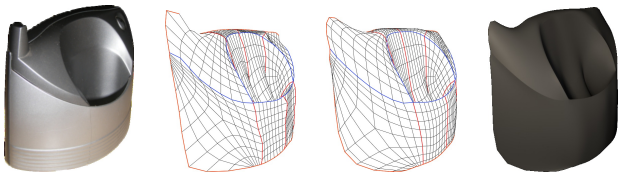
The cross-shaped model in Fig.2 presents different surface features (smooth curves, corners, edges and darts). The proposed reconstruction approach succesfully creates an intuitive piece-wise smooth surface (Fig. 2, bottom) starting from the simple 3D input curve network (Fig. 2, top). The two intermediate steps show the Base Mesh, output of the Basic Refinement process, and the Refined Mesh, output of the Surface Diffusion Flow.

Similarly to the cross-shaped example, the cube in Fig. 5(a) presents different surface features. Due to the penalizations introduced by the weights defined in (6) the face bounded by four sharp edges (marked in blue in Fig. 5(a)), is reconstructed into a planar surface (Fig. 5(b)). In Fig. 5(c)



(a) (b) (c) (d) (e)

Figure 7: *The reconstruction of an old-style telephone.*



(a) (b) (c) (d)

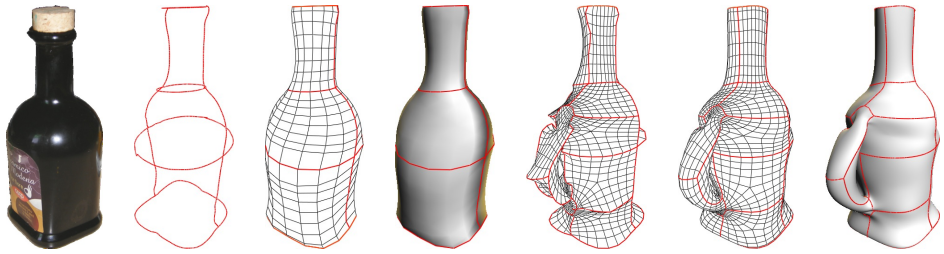
Figure 8: *The reconstruction of an object with cavities.*

the reconstructed piece-wise smooth surface is shown from a different point of view where crease and dart features are well visible. The surface colors represent the surface curvatures, where green color indicates low curvature and red color high curvature values.

The example in Fig. 6 shows the ability of our method to precisely reconstruct a spherical surface starting from the three orthogonal circumferences and the underlying polyline mesh illustrated in Fig. 6(a). In this particular example we applied the Basic Refinement step to obtain a very refined mesh, shown in Fig. 6(b). The surface diffusion process evolves the base mesh into the spherical mesh of Fig. 6(c), perfectly fitting the initial curve network.

Example 2: In the second experiment we show the result of the complete reverse engineering (RE) pipeline for two real objects shown in Fig. 7(a) and 8(a): an old-style telephone and a phone charger for a wireless telephone.

The reconstructed 3D virtual model (Fig. 7(e)) is the result of the multi-step reconstruction process consisting of the Basic Refinement step (Fig. 7(b)), the Surface Diffusion step (Fig. 7(c)), and the Subdivision Refinement (Fig. 7(d)). The final result of Fig. 7(e) clearly shows how a complex model with a variety of different surface features can be succesfully reconstructed starting from a very simple and rough curve network (overlayed in Fig. 7(d)). The blue curves were



(a) (b) (c) (d) (e) (f) (g)

Figure 9: *The reconstruction of a bottle ((a)-(d)), and the result of adding a virtual handle to the bottle ((e)-(g)).*

marked as sharp during the RE session using our smart-pen sketching device.

The second reconstruction of the physical object shown in Fig. 8(a), followed the multi-step recontruction process on a simpler physical object with a pronounced cavity, a feature that often creates problems with optical 3D scanner devices. Thanks to safe-occlusion characteristics of the smart-pen device, the user is able to sketch curves inside any cavities. The reconstruction system is then able to correctly build the smooth surface illustrated in Fig. 8(d). The intermediate reconstruction steps of Basic Refinement and Surface Diffusion Flow are shown in Fig. 8(b) and Fig.8(c), respectively.

Example 3: In this example we describe an edit session performed using our RE system to interactively sketch new curves on the free space in order to add virtual parts to an object. In particular, in Fig. 9 the users draws the minimal amount of curves (Fig. 9(b)) necessary to reconstruct the basic shape of the bottle object in Fig. 9(a). The system provides a real-time visualization of the reconstructed smooth surface (Fig. 9(d)), created from the subdivision of the refined mesh of Fig. 9(c), output of the Surface Diffusion Flow.

The user then decides to add a handle to the virtual bottle. Using the "hole creation tool" and "border gluing tool" from the smart-pen sketching interface (described in [BFL*10]) the user traces the 3D outlines representing the handle. The system reconstructs the smooth surface of Fig. 9(g) by first applying the Basic Refinement step (Fig. 9(e)) and then performing the Surface Diffusion Flow that produces the mesh in Fig. 9(f).

7. Conclusion

This paper has presented a system for reconstructing free-form surfaces from sketched irregular curve networks acquired with a smart pen. The smart pen device is an extent of the active-pen device introduced in [BFL*10], which has been suitable designed to provide a more accurate and fast support for the tracking system. The surface reconstruction consists in a basic first step which builds a low resolution base-mesh associated with the curve network, and a second step in which the base-mesh is refined to produce a smooth surface which preserves curvature features defined by the user on the curve network. While the system is not perfectly accurate as, for example, other, usually expensive, laser scanning systems, it can be effectively used in 3D curve/silhoulette sketching tools in CAD system environments where slight inaccuracies will not impede user task performance.

References

[BCF*08] BOWMAN D., COQUILLART S., FROEHLICH B., HIROSE M., KITAMURA Y., KIYOKAWA K., STUERZLINGER W.: 3d user interfaces: New directions and perspectives. *IEEE Computer Graphics and Applications 8*, 6 (2008), 20–36. 2

[BFL*10] BECCARI C., FARELLA E., LIVERANI A., MORIGI S., RUCCI M.: A fast interactive reverse engineering system. *Computer Aided Design 42*, 10 (2010), 860–873. 1, 3, 8

[KH06] KARPENKO O. A., HUGHES J. F.: Smoothsketch: 3d free-form shapes from complex sketches. *ACM Trans. Graph. 25* (2006), 589–598. 2

[LHC*07] LIN, HONGWEI, CHEN, WEI, BAO, HUJUN: Adaptive patch-based mesh fitting for reverse engineering. *Comput. Aided Des. 39* (2007), 1134–1142. 2

[MDS*02] MEYER M., DESBRUN M., SCHROEDER P., , BARR A.: Differential geometry operators for triangulated 2-manifolds. In *Proc. VisMath '02, Germany* (2002), pp. 237–242. 6

[NISA07] NEALEN A., IGARASHI T., SORKINE O., ALEXA M.: Fibermesh: designing freeform surfaces with 3d curves. *ACM Trans. Graph. 26* (2007). 2

[Saa03] SAAD Y.: *Iterative Methods for Sparse Linear Systems*, 2nd ed. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2003. 6

[SF98] SINGH K., FIUME E.: Wires: A geometric deformation techniques. In *In Proceedings of SIGGRAPH 98* (1998), pp. 405–Ű414. 1

[SK01] SCHNEIDER R., KOBBELT L.: Geometric fairing of irregular meshes for free-form surface design. *Computer Aided Geometric Design 18*, 4 (2001), 359–379. 2, 6

[SWZ04] SCHAEFER S., WARREN J., ZORIN D.: Lofting curve networks using subdivision surfaces. In *Proc. Eurographics Sym. on Geometry Processing* (2004), pp. 103–113. 2

[WS01] WESCHE G., SEIDEL H.-P.: Freedrawer: a free-form sketching system on the responsive workbench. In *Proceedings of the ACM symposium on Virtual reality software and technology* (New York, NY, USA, 2001), VRST '01, ACM, pp. 167–174. 2