A Spline-Driven Image Slicer

Release 1.00

Jérôme Velut

September 19, 2011

jerome.velut@gmail.com

Abstract

In this article, a spline driven image slicer algorithm is presented. It is concretized through a vtkAlgorithm-inherited class that takes two inputs and gives two outputs. The first input is the volume from which the slice should reconstructed while the second one is the spline which tangent vector is used as slicing plane normal. The first output is a 2D image resliced from the volume. The second input gives the geometric information of the slice plane location in the volume space. An example of use is given through a simulation of a dental panoramic view from a MDCT volume.

Contents

Introduction

Slicing through volume data is a common task in scientific visualization. Lots of existing image processing tools provide this necessary functionality as soon as the visualization of 3D (or more) data has to be performed on a 2D screen.

In VTK, different ways are possible, depending on the particular slicing needed by the end-user. The simplest is `vtkExtractVOI` if the desired slice is parallel to the input image axes. This method is used in ParaView in the Slice Representation. The most sophisticated is the `vtkImageReslice` filter in which the cutting plane may be oriented arbitraly in the volume space. It enables the well-known Multi-Planar Reconstruction(MPR) visualisation.

The algorithm presented here aims at computing the slicing orientation of a `vtkImageReslice` filter directly from an input spline curve. This orientation is defined as the Frenet-Serret frame at each point of the spline. In the following, a Frenet-Serret variant is proposed in order for the spline tangents to be used as slice plane normals. An application on mandibule panoramic visualisation is presented, along with an extension to a Straightened Reformatted Volume rendering.

1 Frenet-Serret frame along a curve

Frenet [1] and Serret [5] defined independently several properties on non degenerate curves from the euclidean space. Though the Frenet–Serret formulas aim to describe moving particles, they may be of interest for defining local charts in a 3D space.

1.1 Initial definition

Let $\mathcal{C}$ be a continuous differentiable curve in the euclidean space $\mathbb{R}^3$ parameterized by t. The Frenet–Serret frame of $\mathcal{C}$ at t is made of the tangent T, normal N and binormal B unit vectors with:

$$\begin{aligned} T(t) &= \frac{dC(t)}{dt} && (1)\\ N(t) &= \frac{dT(t)}{dt} && (2)\\ B(t) &= T(t) \times N(t) && (3)\end{aligned}$$

Figure 1 shows such frames along a closed curve in $3D$-space. It is important to note that equations (1)–(3) define a set of charts without taking into account the curvature and torsion of the curve. In the following, only discrete curves are concerned. Central differences are used to compute derivatives, constant padding is used at boundaries and curvilinear abscissae t becomes a discrete point index k.

1.2 Coherent consecutive normals computation

The normals of a curve have the property to be directed according to the center of curvature. A corollary is that when the curvature changes its sign, the normal will flip. Figure 2(a) illustrates this behaviour. As the

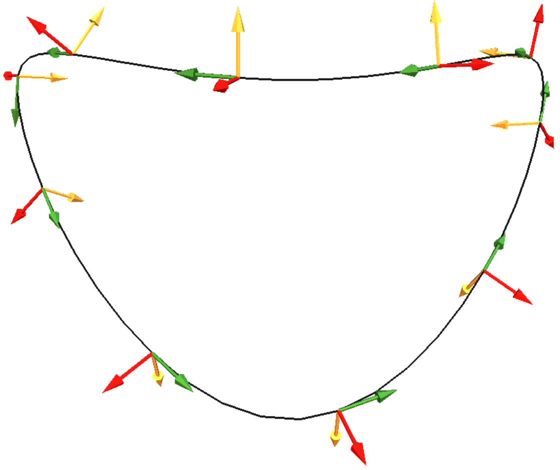
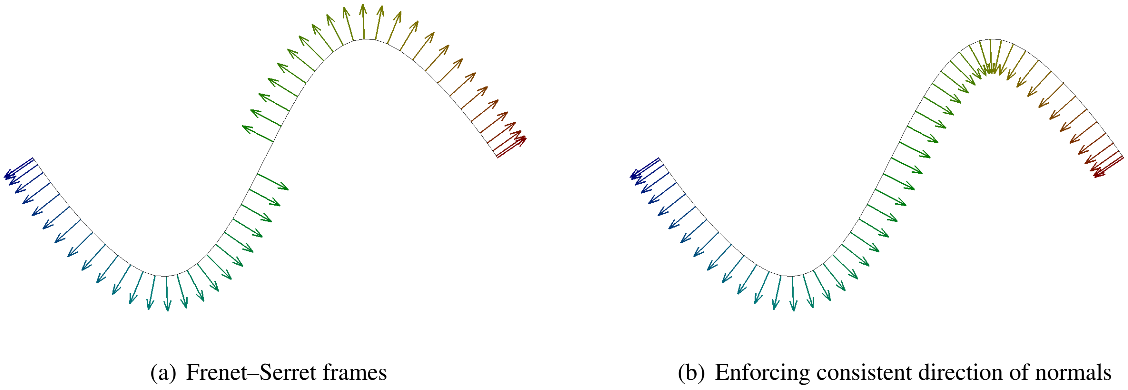


Figure 1: Frenet-Serret frames along a curve: Tangents T are green arrows, Normals N are red and Binormals B are yellow.



(a) Frenet–Serret frames

(b) Enforcing consistent direction of normals

Figure 2: Normals of a curve. (a) The mathematical definition of normal induces a change of orientation of the normal. The point where curvature is 0 has null normal. (b) Arbitrary enforcing normals orientation brings consistency along the curve.

purpose of this paper is to propose a volume slicer along a spline, the consequence will be an annoying flip of two consecutive image slices at inflection points. In order to avoid that, the previous normal N_{k-1} is used to determine the current one N_k. The main idea is to find a projection of N_{k-1} in the plane orthogonal to $\mathcal{C}$ at k. This projection is obtained thanks to the cascaded cross product:

$$\begin{aligned} B_k &= N_{k-1} \times T_k \quad (4) \\ N_k &= T_k \times B_k \quad (5) \end{aligned}$$

where T_k is the tangent at point k. In equation (4), B_k represents a consistent binormal regarding the previous one. It is not the binormal *stricto sensus* as it is computed from the previous consistent normal N_{k-1}, which is in turn not the derivative of T but defined in equation (5).

Figure 2(b) shows that the consistent normals are not changing their orientation at inflection points. Moreover, at the exact inflection point, the consistent normal is not null as opposed to the conventional computation (figure 2(a)). A slightly different approach was presented in [4].

1.3 VTK implementation

The Frenet-Serret frame computation along a spline is implemented in a class `vtkFrenetSerretFrame` that inherits from `vtkPolyDataAlgorithm`. The input has to be a `vtkPolyData` containing at least one polyline. The output is a deep copy of the input plus two (optionnaly three) vector arrays, `FSTangents` and `FSNormals` (optionnally `FSBinormals`). The consistent computation is an option `ConsistentNormalsOn/Off`.

According to the formulation of consistent normal vectors, they are not related to curvature anymore. It also implies that the initial direction will be used as a reference for the next normals. This initial direction can be arbitrary set through the `ViewUp` option.

This filter is available in ParaView by setting `BUILD_PARAVIEW_PLUGIN` to `ON` during the CMake process.

2 Image slicing

The algorithm that extracts a 2D slice from a volume harnesses the previous `vtkFrenetSerretFrame` filter and the native VTK filter `vtkImageReslice`. Both are connected and embedded in `vtkSplineDrivenImageSlicer`, the latter offering an interface for the formers' parameters.

2.1 Inputs and parameters

Input port 0 of `vtkSplineDrivenImageSlicer` is a `vtkImageData` port. It will be processed by the internal `vtkImageReslice` filter.

Input port 1 is a `vtkPolyData` port. The polydata must contain one or more polylines and will serve the input of the internal `vtkFrenetSerretFrame` filter.

The parameters of `vtkSplineDrivenImageSlicer` are related to the two main algorithms. Table 1 summarizes these parameters.

Name	Description
SliceExtent	Number of pixels in x and y of the output slice
SliceSpacing	Size of pixels in x and y of the output slice
SliceThickness	z-Spacing of the output slice, useful if appending the output to other slices
OffsetLine	Index of the cell in the `Lines` input along which to extract the slice
OffsetPoint	Index of the point in the selected spline input that will represent the center of the output slice
Incidence	Correspond to the `ViewUp` parameter of the `vtkFrenetSerretFrame` filter
ProbeInput	Specify if the filter should or not probe the input data on the output plane

Table 1: List of parameters of `vtkSplineDrivenImageSlicer`

2.2 Outputs

This algorithm provides two output ports. Output port 0 is a `vtkImageData` representing a slice of the input volume, orthogonal to the input spline at the specified location p (`OffsetLine` and `OffsetPoint`). Image extent is $[0, \texttt{SliceExtent}[0], 0, \texttt{SliceExtent}[1], 0, 1]$. Image spacing is $[\texttt{SliceSpacing}[0], \texttt{SliceSpacing}[1], \texttt{SliceThickness}]$. The image is centered on p. An additional output is provided on output port 1 with the geometry of the slice plane. It is a `vtkPolyData` representing a grid with same resolution and size as the output slice but oriented and centered in $\mathbb{R}^3$ according to the parameters, the spline and the Frenet-Serret frame. The option `ProbeInput` allows for probing the input volume with this grid.

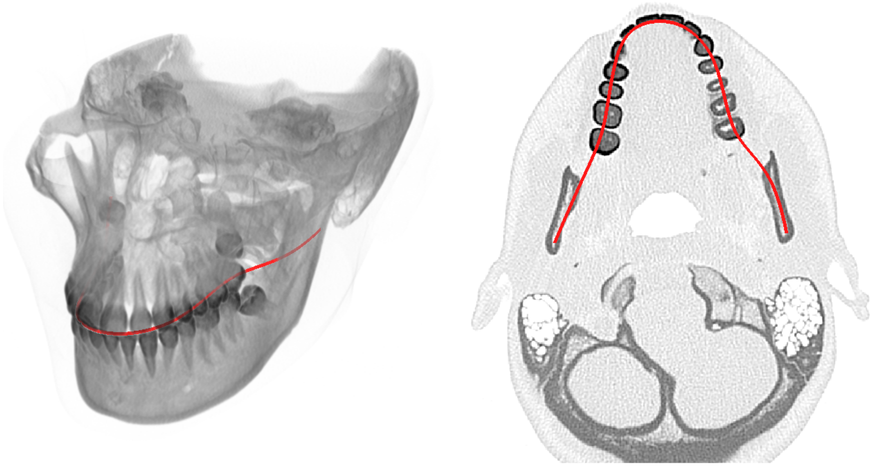
2.3 ParaView plugin

A ParaView plugin is possibly built to enable the `vtkSplineDrivenImageSlicer`. The `OffsetPoint` parameter is animateable as well as the `Incidence` parameter. For illustration, the DICOM dataset INCISIX from Osirix website [2] is sliced along a manually traced spline. As the data represent a mandibule, the spline is chosen to go through the teeth (figure 3). Figure 4 have been obtained by connecting two spline-driven slicers to the volume and the spline (`spline2.vtp`).

3 Straightened Reformatted Volume

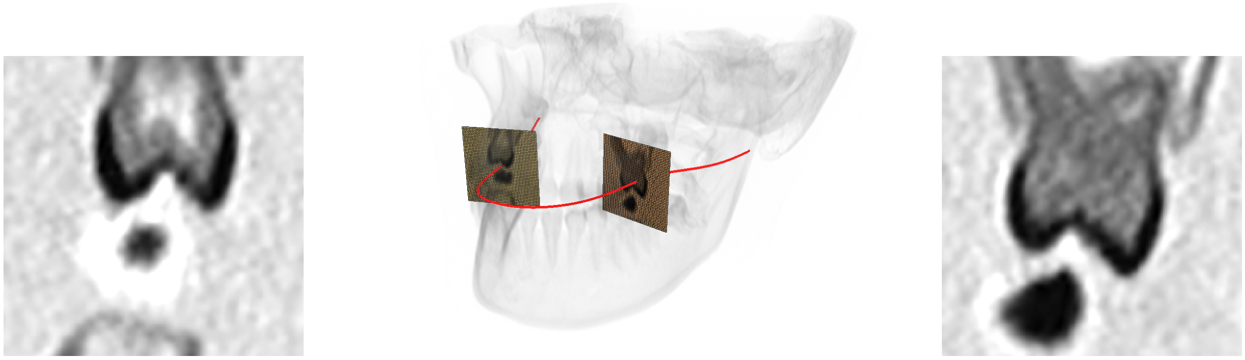
The previous spline-driven volume slicer outputs $2D$ images that share the same geometric properties (dimensions and pixel size) despite the position of the slice planes along the curve. It allows a concatenation of successive images into a single, reformatted volume. This new volume will have X and Y dimensions equal to the slice extent, whereas the Z dimension is the number of points of the curve. Though the voxel x and y spacings are automatically deducted from the slicer parameters, the z-spacing has to be specified. It makes sense to give a z-spacing equal to the sampling size of the spline.

This process tends to unfold the input volume along the spline. It is strongly related to the Straightened Curved Planar Reformation presented in [3]. Indeed, the latter is a subset of the reformatted volume that we



(a) Volume rendering (b) Axial slice

Figure 3: Visualization from ParaView of a mandibule CT scan and a spline



(a) Slice at 150th point (b) Volume rendering with slice planes (c) Slice at 350th point

Figure 4: Two slices of dimension 200 × 200, spacing 0.1 × 0.1 and incidence $2\pi/3$

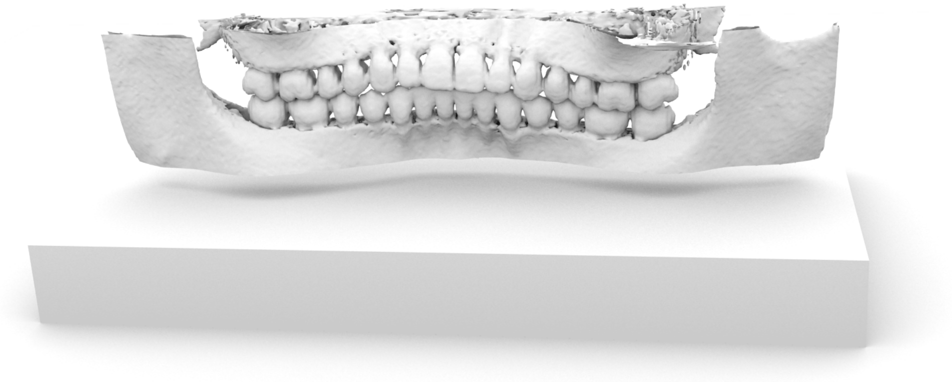


Figure 5: Surface rendering of an SRV extracted from the dataset INCISIX.

will called Straightened Reformatted Volume (SRV) in the following.

3.1 Dental panoramic visualisation

A common practice in dental care is the dental panoramic X-ray exploration. It is a 2D projection of the mooth tissue and teeth modulated by their radio-opacity. Another practice takes advantage of the computed tomography from which a volume of the head can be obtained. However, the naturally carved aspect of the mandible makes it difficult to explore the dataset in 3D. The proposed SRV is able to keep the 3D information while virtually deploying the teeth in a unique plane.

In figure 5, the INCISIX dataset is sliced orthogonally to the spline showed in figure 3 at each point. The slices are appended on the Z-axis, thus building a Straightened Reformatted Volume. For visualisation purposes, the SRV is thresholded and a contour filter is applied. Finally, this surface is rendered in Blender. The provided code includes an example "StraightenedReformattedVolume.cxx" that computes an SRV from an input volume (format `vti`) and an input spline (format `vtp`).

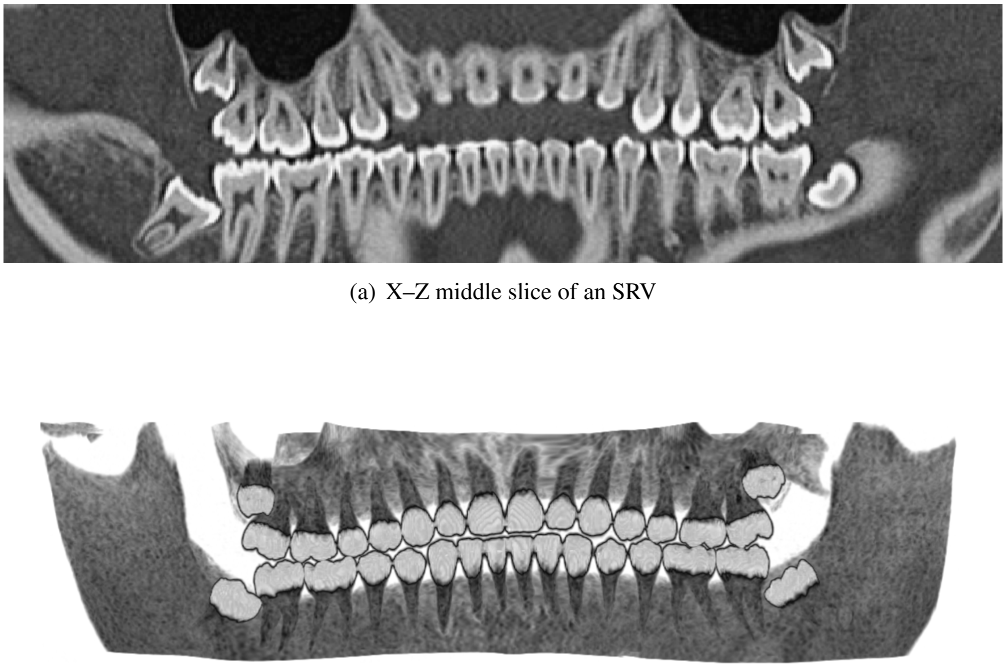
3.2 Straightened Curved Planar Reformation

In the proposed implementation, the slices are centered on the spline points. A direct consequence is that the X–Z slice taken at $Y = dimension_Y/2$ from a Straightened Reformatted Volume is strictly equivalent to the Straightened Curved Planar Reformation used in [3] for vessels visualisation. Moreover, slicing in the SRV on Y adds the possibility to explore boundary structures (figure 6(a)).

Finally, the SRV can be represented as MIP (Maximum Intensity Projection), giving a visualisation close to the usual X-ray dental panoramic (figure 6(b)).

4 Conclusion

In this paper, a new VTK class that computes Frenet-Serret frames along curves was proposed. A deviation from the strict mathematical definition was performed in order to avoid a curvature sign-dependent



(a) X–Z middle slice of an SRV

(b) MIP rendering of an SRV

Figure 6: Different visualisations of a Straightened Reformatted Volume in a Straightened Curved Planar Reformation manner

orientation of the normals. This constraint was defined as “consistent normals computation”.

These consistent Frenet-Serret frames were then used to slice an input volume along a given spline, yielding a spline-driven image reslice algorithm. It was applied on a CT of a mandibule in the ParaView GUI. It was shown that appending successive 2D slices along a spline produces a new volume called “Straightened Reformatted Volume” (SRV). This SRV includes the straightened curved planar reformation found in the literature.

Despite the dental-exclusive application shown here, the vascular visualisation softwares should benefit from this algorithm. Another improvement could be to memorized the transformation of each slice, so that a user could explore the SRV while having the localisation information in the original volume.

A Software Requirements

The provided code has been successfully built with ParaView git master branch and the corresponding version of VTK at the time of submission. System was Linux, Fedora-14-x86_64.

This paper and codes are hosted on github at `http://github.com/jeromevelut/vtkKinship`, on the branch `SplineDrivenImageSlicer-VTKJournal`.

References

[1] Jean-Frédéric Frenet. Sur les courbes à double courbure. *Journal de mathématiques pures et appliquées*, 17:437–447, 1852. 1

[2] INCISIX. Dataset from the osirix website. In *http://pubimage.hcuge.ch:8080/DATA/INCISIX.zip*. 2.3

[3] A. Kanitsar, D. Fleischmann, R. Wegenkittl, P. Felkel, and E. Groller. Cpr - curved planar reformation. In *Proc. IEEE Visualization VIS 2002*, pages 37–44, 2002. 3, 3.2

[4] Fopke Klok. Two moving coordinate frames for sweeping along a 3d trajectory. *Computer Aided Geometric Design*, 3(3):217 – 229, 1986. 1.2

[5] Joseph Serret. Sur quelques formules relatives à la théorie des courbes à double courbure. *Journal de mathématiques pures et appliquées*, 16:193–207, 1851. 1