Reconstruction of surfaces from planar contours through contour interpolation

Kyle Sunderland, Boyeong Woo, Csaba Pinter, and Gabor Fichtinger

Laboratory for Percutaneous Surgery, School of Computing, Queen's University, Kingston, ON

Introduction

- Segmented structures representing targets and organs at risk are stored as 2D contours contained on evenly spaced cross-sectional slices.
- The **conversion of planar contours into a 3D surface** is a process that is required by many radiation oncology treatment planning software packages. Different contour conversion algorithms can create surfaces that are not identical and can cause differences in dosimetry and dose volume histograms for the same structure.
- Our goal was to implement an **open source contour interpolation algorithm** in the **SlicerRT** extension for the **3D Slicer** platform.

Methods

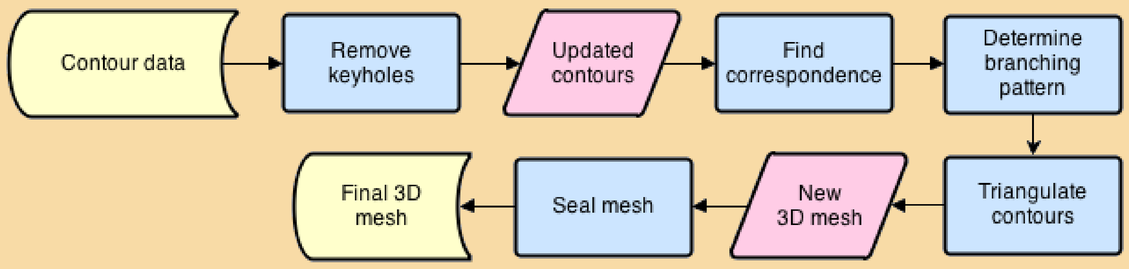

Figure 1: Data flow diagram representing the process of converting a set of 2D contours into a corresponding surface mesh.

- The DICOM standard defines the storage of hollow structures as **keyholes** that are represented as an inner and outer contour connected by and arbitrarily small channel in order to be considered a single contour (Figure 2/Left). These are **separated into individual contours** before triangulation, as the keyhole channels can potentially cause issues with the triangulation process.
- **Correspondence** between contours is determined through the use of **bounding-box overlapping** to determine which contours should be triangulated together.
- **Branching** contours occur when a contour corresponds to multiple contours on an adjacent slice (Figure 2/Right). In this case a pattern for constructing the branching mesh needs to be calculated. This is done by **separating contours into multiple sections** based on which adjacent contour they are closest. The sections are then triangulated independent of each other.

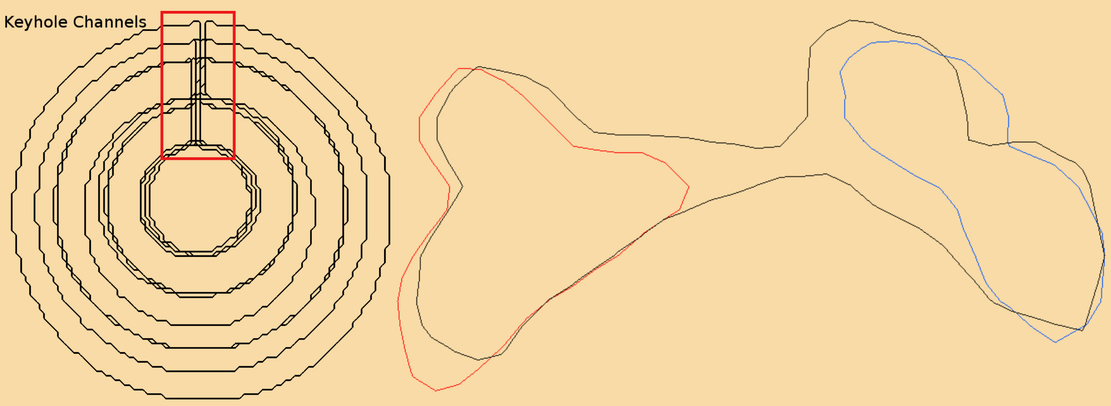

Figure 2. A fabricated set of contours containing keyholes (left) and a set of contours with multiple correspondence, requiring branching (right).

- The surface mesh **triangulation** is designed to minimize the length of the edges spanning the two contours. The **minimum length triangulation** is calculated using a **dynamic programming algorithm** on a directed graph in which each of the nodes in the graph represents a line connecting a point on each contour.
- The final step in the mesh construction is to **seal** the contours that reside on the top and bottom of the mesh. The triangulation is completed by using a **Delaunay triangulation** on the contour.

Results

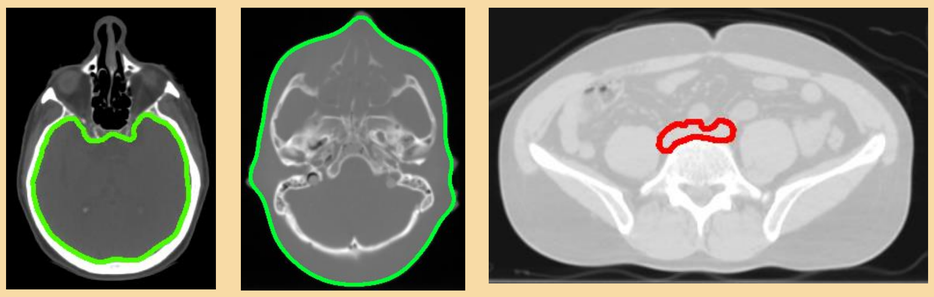

Figure 3. Example CT slices used to produce the meshes in Figure 4, with brain (left), head (center), and vessel (right) contours.

- The algorithm was **implemented as a Python scripted module** in SlicerRT.
- The implemented algorithm was **tested on several sets of contours** that were segmented from **real CT images** (Figure 3).
- The implemented algorithm was found to be able to produce **qualitatively acceptable 3D meshes** for most sets of planar contours that were tested (Figure 4).

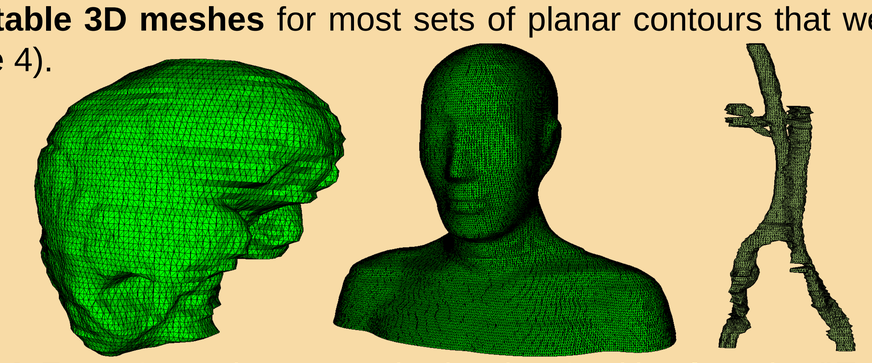

Figure 4. Example mesh generated using the algorithm from the CT scan images shown in Figure 3, from brain (left), head and neck (center), and vessel (right) contours.

- While the algorithm was able to **handle individual instances** of rapid change, branching and internal contours, the occurrence of **several of these issues simultaneously** was found to **cause problems**.
- Problems were found to occur in **regions of rapid change** in which **contours were not similar in shape or size**. This caused the triangulation to create triangles which all converged to a single point in the smaller contour.
- The simultaneous occurrence of **branching contours and internal contours** was found to **cause problems** with the branch calculation step.
- **Rapid change in regions containing internal contours** was sometimes found to **cause problems** as the external contours could be incorrectly triangulated to the internal ones.

Conclusion

- **Successfully implemented an algorithm in SlicerRT** as a Python scripted module that **converted sets of 2D planar contours into 3D surface mesh** and was able to handle a variety of issues.
- The implemented algorithm was found to be able to produce 3D surface mesh that were considered **qualitatively good** for most of the sets of contours that were tested.

Acknowledgement

Kyle Sunderland was supported by the Natural Sciences and Engineering Research Council of Canada through an Undergraduate Student Research Award. Gabor Fichtinger is supported as a Cancer Care Ontario Research Chair in Cancer Imaging.