Расчёт объёма дыхательных пазух: выбор алгоритма/подхода

Способ через отдельную 3D-программу

В качестве 3D программы для пробы того, что будет ли способ вообще рабочим, был выбран Blender, так как он поддерживает множество форматов, а также является open source приложением.

Преобразование nii в stl

Было попробовано преобразование **nii** в **stl** через данный метод:
http://wiki.osll.ru/doku.php/projects:otolaryngologist:nii_to_mesh

Вывод:

данная идея была сразу отброшена по данным причинам:

- Сложно настраивать параметры, потому что нужно каждый раз запускать скрипты, чтобы увидеть результат, так ещё и через двойной прогон;

  Без настройки параметров можно увидеть множество шумов и дефектов.
  
- Также способ сам по себе не удобен и долгий, ведь предполагает использование множества шагов почти несвязанных друг с другом.

Анализ библиотек плагинов 3D Slicer'а для экспорта моделей

- Были найдены библиотеки для Python для импорта/экспорта разных 3D форматов из сохранённой сегментации, но это всё не равно не удобно, ведь присутствует дополнительный шаг, хоть и эта библиотека работает с уже сегментированном форматом файлов, поэтому позволяет подготовить сегментацию в **3D Slicer**;

  Вот пример:
  https://pypi.org/project/slicerio/
- Также было найдено, что есть такой набор библиотек, как **VTK**. Он встроен в **3D Slicer**, что позволяет вообще нативно проводить сегментацию снимков (с множеством настроек), а также имеется так нужный нам экспорт.

Вывод:

мы нашли способ позволяющий несколькими кликами экспортировать медицинские снимки в популярные 3D форматы файлов.

Инструкция по способам:

Давайте тут подробнее разберём шаги:

1. Через **3D Slicer** открываем наши снимки (пусть для примера это будут **nii** файл);
2. Далее нам нужно сегментировать нашу область, поэтому мы открываем **Segment Editor**, который находится в списке всех модулей по такому пути: **Segmentation/Segment Editor**;
3. Создаём области с помощью инструмента **Threshold**: крутим нижний порог до того, чтобы на взгляд почти везде хорошо выглядело;
4. А на тех местах, где не дорисовывается стенка — мы используем инструмент **Paint** и дорисовываем;
5. Затем мы переименовываем сегмент, как нам удобно и нажимаем на выдвигающиеся меню на кнопке с зелёной стрелкой. И выбираем пункт **Export to Files**. Выбираем разве что только путь, куда сохранится 3D объект и его формат (будем использовать **stl**);
6. Уже в **Blender** удаляем внешние стенки (например через выделение через лассо с включенным **x-ray** с видом сверху);
7. Затем удаляем аккуратно всё лишнее посередине через тот же способ, что и выше;
8. Пользуемся функцией выделения сетки по соседям (Select Linked) на каждой из пазухи и делаем **Separate** в отдельные объекты;
9. Подчищаем геометрию, если нужно. Можно даже подчистить с дырками;
10. Ставим плагин под названием **Mesh: 3D-print Toolbox**;
11. Открываем в свойствах объекта вкладку **3D-print** и пользуемся функцией **Clean Up/Make Manifold**, которая заделывает дырки;
12. И нажимаем **Statistics/Volume** - и видим объём.

Проблемы и попытки их решить

3D объект получается большим

Из-за чего **Blender**'у немного тяжело работать в **Edit Mode** (может в среднем думать по 2-3 секунды на приблизительно среднем железе), когда мы пытаемся сделать какие-нибудь преобразования над мешем.

- Можно уменьшить кол-во сетки с помощью модификатор **Decimate** в **Blender**'е, но тогда мы можем потерять в точности вычисления объёма, поэтому мы отказались от такой идеи;
- Можно в **3D Slicer**'е воспользоваться инструментом **Scissors**, и он нам помогает, но уже в **Blender** придётся аккуратнее тогда удалять стенки, которые появляются ещё в **3D Slicer**'е из-за логики работы сегментации. Ну и ножницами тоже надо аккуратнее пользоваться, чтобы не отрезать лишнего.

Невозможно подкрутить идеально Threshold в 3D Slicer'e.

На выбор у нас есть множество пресетов: **Otsu**, **Huang**, **IsoData**, **Kittler-Illingworth**, **Maximum Entropy**, **Moments**, **Renyi entropy**, **Shanbhag**, **Triangle**, **Yen**, но они все не работали хорошо

на примере, который мы рассматривали и приходилось вручную двигать ползунки **threshold**'а.

Но если даже двигать их вручную, то появлялись такие артефакты, как не бралась область, где на глаз должна быть стенка. Но когда мы увеличивали значение **threshold**'а, то брались области, где должна быть пустота.

Учитывая пункты выше можно пойти двумя путями:

1. **threshold** не надо завышать и нужно будет воспользоваться инструментом **Paint** и провести стенки самим, там где они должны были быть. В основном это не надо делать больше, чем на нескольких снимках;
2. Либо же чуток завышаем **threshold**, чтобы появлялись артефакты, но мы ими просто пренебрегаем, потому что нам нужна приблизительная оценка.

Только встроенными инструментами 3D Slicer'a

С использованием только лишь встроенного функционала

Был проанализирован функционал **3D Slicer**'а и были найдены такие интересные инструменты, как:

1. Инструмент под названием **Islands** в модуле **Segment Editor**;
2. Модуль **Segment Statistics**, который находится в категории **Quantification**.

Сочетание этих двух инструментов открыл ещё более быстрый и удобный путь для получения объёма.

Инструкция по нему

1. Первые шаги по открытии и выбора модуля **Segmentation/Segment Editor** аналогичны;
2. Выбираем один из способов выделения областей описанных после данной инструкции и применяем его;
3. Далее выбираем инструмент **Islands** и далее выбираем **Split Islands to segments**. В **Minumum Size** ставим, например, **70k** (данное значение позволило выделить только пазухи и оставляет всё ещё запас, потому что по итогу пазухи имели в районе **130k** вокселей, поэтому можно даже взять значение больше и не бояться, что пазухи не выделяться);
4. Возможно перед следующим шагом надо будет применить ещё какие-то инструкции из способов ниже;
5. Тут у нас должны остаться лишь 3 сегмента: две пазухи и внешняя часть. Если это не так, то можем повысить кол-во вокселей на прошлом шаге или вручную удалить, ведь внешнюю часть в любом случае надо будет удалить;
6. Затем заходим в **Quantification/Segment Statistics** и выбираем в поле **Segmentation** нашу сегментацию с двумя сегментами (то есть нашими пазухами);
7. В `Advanced` по умолчанию настроено уже, как нам нужно, но можно на всякий случай убрать **Scalar Volume**, потому что там будут идентичные результаты с **Label Map** (хотя он и так работает только тогда, когда выбираешь **Scalar Volume** вместо **None**, поэтому и говорю, что и по умолчанию всё хорошо);

> Для работы **Closed Surface** нужно, чтобы в **Segment Editor** было включено **Show 3D**.

8. Нажимаем **Apply** и видим результаты. Их разбор будет пунктом ниже.

Алгоритмы выбора сегментов

1. Вручную дополнять:

   Это случай с завышением **Threshold**'а и ручным дорисовыванием с помощью инструмента **Paint** или убирать лишнее с помощью **Erase**.

   Результаты:

Segment	Voxel count	Surface mm2	Volume cm3
Left	152976	4232.45	19.179
Right	136020	4071.59	17.0564

   Вывод: Это не удобно и очень муторно.

2. Завысить Threshold:

   Главное, чтобы пазуха была отдельной частью.

   Результаты:

Segment	Voxel count	Surface mm2	Volume cm3
Left	121637	4759.2	15.2877
Right	118842	4451.78	14.9167

   Вывод: имеет смысл, но результаты получаются не такими точными и красивыми, как хотелось бы.

3. Завысить Threshold с Smoothing

   Инструкция:

   1. после применения инструмента **Islands** мы для **каждой** из пазух применяем инструмент **Smoothing** в режиме **Closing (fill holes)** (я выбрал **10.00mm**);

   Результаты:

Segment	Voxel count	Surface mm2	Volume cm3
Left	135528	3913.61	16.994
Right	132024	3928.04	16.5569

   Вывод: уже в разы лучше.

4. С использованием Margin и опционально Smoothing

   Инструкция:

   1. Тут мы можем взять такой **Threshold**, который будет удовлетворять визуально и забить на то, что могут быть небольшие соединения пазух с другими элементами носа. И применяем его;
   2. Затем мы выбираем инструмент **Margin** и выбираем в **Operation**: **Shrink**, а в качестве размера можно выбрать: **1.00mm**;
   3. Далее используем инструмент **Islands** по инструкции выше;
   4. И теперь применяем обратно **Margin** с таким же размером, но в режиме: **Grow**.
   5. Опционально ещё можем применить инструмент **Smoothing** в режиме **Closing (fill holes)**, например, на **5.00mm**.

   Результаты только с Margin:

Segment	Voxel count	Surface mm2	Volume cm3
Left	145906	4250.16	18.2842
Right	141524	4202.23	17.743

   Результаты также ещё и с Closing (fill holes) (5.00mm):

Segment	Voxel count	Surface mm2	Volume cm3
Left	148392	4114.5	18.6082
Right	143420	4130.2	17.9869

   Выводы: По результатам можно увидеть, что **Smoothing** (остальные алгоритмы я тоже пробовал, но они визуально работали хуже) не особо что-то меняет, потому что и без него получается очень приближённо к настоящим результатам, но при этом он бывает сглаживает не в тех местах, из-за чего, например, **Right** и повысился. Поэтому данный способ лучше юзать без сглаживания (либо с небольшим сглаживанием, но результаты тогда будут на уровне +1-2%).

1. Окончательный алгоритм:

   Этапы:

   1. Создаём сегмент для маски, которую мы будем в дальнейшем использовать и делаем её активной;
   2. Применяем инструмент **Threshold** (параметры для него задаёт пользователь и подробнее про них будет ниже) в качестве **MinimumThreshold**;
   3. Затем применяем **Smoothing** с параметрами: **Closing (fill holes)** и 1.5mm;
   4. Затем инвертируем сегмент маски;
   5. Переключаемся на первоначальный сегмент, а также выставляем в качестве маски, тот сегмент маски, с которым мы работали до этого;
   6. Применяем **Local Threshold** (инструмент из данного пакета: ссылка) и затем задаём такие параметры: в качестве уже **MaximumThreshold** ставим параметр от пользователя, в качестве **SegmentationAlgorithm** выбираем **GrowCut**, а в качестве **MinimumDiameterMm** 1.0mm;
   7. Теперь мы выполнили всю работу с маской, тем самым можем удалить эту маску;
   8. Далее применяем **Islands** с режимом удаления мелких островков и с параметром

MinimumSize=3000, чтобы отбросить лишнее, если оно будет;

9. Под конец я применяем **Smoothing** с параметрами: **Closing (fill holes)** и 1.5mm, чтобы соединить те части, которые могли потенциально разъединиться от шагов выше.

Segment	Voxel count	Surface mm2	Volume cm3
Left	156162	4492.03	19.5664
Right	141382	4258.5	17.7276

Выводы

На основе всех вычислений выше (и на основе вычислений значений на других наборах слайсов), и так как это был наихудший пример — было принято взять последний алгоритм, так как он работал более стабильнее.

Инструкция по использованию



1. Раскрывает/закрывает работу с пазухами;
2. Можно выбрать/создать/переименовать ноду сегментации;
3. Можно выбрать тот объём слайсов с которым будет производится работа;
4. Можно настроить отклонение порога, где базовым значение является значение выставленное алгоритмом в следующем пункте;
5. Позволяет выбрать алгоритм, который будет выставлять базовое значение порога;
6. Переключает отображение 3D;
7. Является переключателем, показывающим примерную область, которая будет выделена при нажатии на слайс;
8. Список имён, которое переключает выбранный сегмент сегментации;
9. Можно выбрать таблицу, куда будут экспортированы данные;
10. Подсчитывает результаты и отображает их внизу экрана.

From:
http://wiki.osll.ru/ - **Open Source & Linux Lab**

Permanent link:
http://wiki.osll.ru/doku.php/projects:otolaryngologist:task_calculate_volume

Last update: **2024/05/06 20:48**