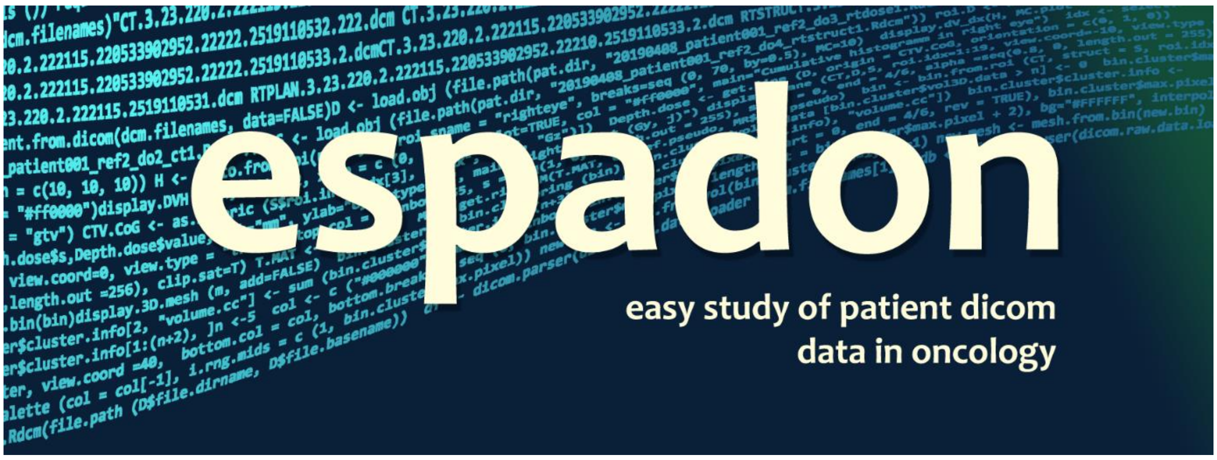


Creation of volumes

C. Fontbonne and J.-M. Fontbonne | LPC Caen
Normandie Univ, ENSICAEN, UNICAEN, CNRS/IN2P3, LPC Caen, 14000 Caen, France
Contact.espadon@lpccaen.in2p3.fr
https://espadon.cnrs.fr/

Table of content

1 Introduction. What we will learn?

Research activities often require reworking of the volumes contoured by the radiotherapists from the rt-struct or directly from CT or MR selections. To this end, espadon has multiple functionalities to simplify the life of developers, by gathering RoIs, filtering volumes using mathematical morphology operators expressed in 3D, or clustering complex and noisy volumes. To illustrate this document, we will need a CT and an rt-struct, loaded for example as follows:

```
rm (list=ls ())
require (espadon)

pat.dir <- "E:/Rdicom data/SBRT_Poumon_Test"

pat <- load.patient.from.dicom (pat.dir, data = TRUE)

S <- pat$rtstruct[[1]]
CT <- pat$ct[[1]]
```

The rt-struct contains the following ROIs:

```
S$roi.info$roi.pseudo

 [1] "ptvlidexpi"           "greatves"             "brachialplexl"        "lungr-itvexpi"        "gtv10%"
 [6] "aorta"                "supvenacava"          "artefactspoumon-836"  "hu-poumon"            "lungs-itvexpi"
[11] "itvexpi"              "i0%"                  "trachea"              "pulmarteries"         "infvenacava"
[16] "brachialplexr"        "chestwallr"           "spinalcanal"          "i100%"                "heart"
[21] "pulmveins"            "spinalcord"           "bronchusprox"         "gtv0%"                "gtvapnea"
[26] "lungr"                "bronchusprox+2cm"     "lungl"                "external"             "itv"
[31] "esophagus"            "implant10%"           "varianigrtthin(0.04)" "varianigrtthin(0.30)"
```

In this document, we will not discuss the processing of binary images in a defined order, as there is none. Instead, we will illustrate the extraction of the alveoli and the blood circuit in the lungs. This could, for example, be the beginning of a detailed study of the effects of dose distribution in these two compartments. This exercise will allow us to exploit several techniques, which we will emphasise when necessary.

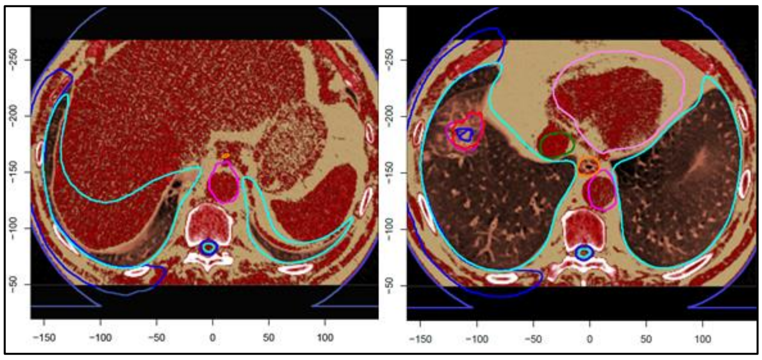
2 The example from scratch

2.1 Let's start by trying some simple things.

As we have loaded the data, we can already see what the lung contours look like by producing, for example, a pdf file that will contain each section:

```
display.3D.contour(S, roi.name = c("lungr", "lungl"))

lungs <- nesting.roi (CT, S, roi.name = c("lungl", "lungr"),
                      xyz.margin = c (10, 10, 10))

pdf ("lungs.pdf")
display.plane(lungs, struct = S,
              view.type = "trans", view.coord = lungs$patient.xyz0[, 3],
              bottom.col = pal.RVV (255), bottom.breaks = seq (-1000, 1000, length.out = 256),
              sat.trans = TRUE)
dev.off ()
```

Lungl and lungr RoIs


Two slices of the CT where we see that automated contouring is sometimes approximative

The image on the right shows that the contours generated on the basis of atlases are sometimes approximate. This is the reason for this example.

Note that the first operation we performed was to reduce the volume of the CT to the part encompassing only the lungs using `nesting.roi`. This saves us a lot of memory space since `object.size (CT)` tells us that it takes up 864 MB while lungs only takes up 152 MB. We could of course get rid of the CT by running `rm (CT)`. The `xyz.margin` option adds 10 mm to the surrounding box, for convenience.

If the lung contours had been of good quality, our problem would almost have been solved in three lines, by selecting the left lung, the right lung using `bin.from.roi`, and adding the two volumes using `bin.sum`:

```
lr.bin <- bin.from.roi (lungs, S, roi.name = "lungr")
ll.bin <- bin.from.roi (lungs, S, roi.name = "lungl")
l.bin <- bin.sum (lr.bin, ll.bin)
```

Unfortunately, we have just seen that this would undermine our objectives. We will therefore have to create contours by means of the script that we will develop below. Before we do this, we need to be clear about what we want to do and what might happen.

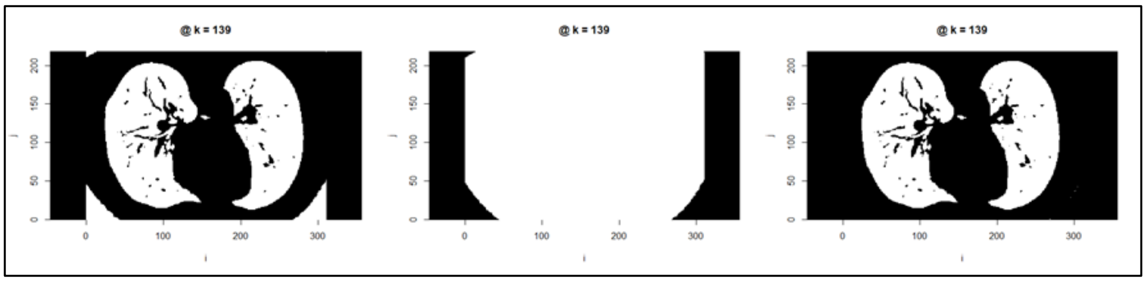
You can already see that the lungs are filled with air... In the Hounsfield units, they are below -500 HU. However, the lungs are not the only cavities visible on the CT (let alone the air surrounding the patient). Regarding the air surrounding the patient, it can probably be removed by selecting the patient contour and its interior.

If we work from the CT and a selection on Hounsfield units, we will certainly have to clean up this selection... Then we will see what happens, but let's start there!

2.2 Airway selection

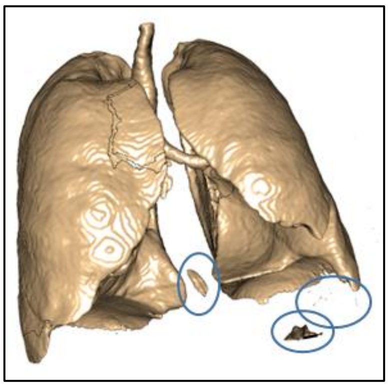
To select the airways, it is sufficient to apply the scheme we have outlined above:

```
air <- bin.from.vol (lungs, min =-Inf, max = -500)
p <- bin.from.roi (lungs, S, roi.name = "external")
air_ <- bin.intersection (air, p)
```



Binary selection of CT < -500 HU on the left, of the patient RoI in the middle, and intersection of both on the right

We selected the air directly on the CT using `bin.from.vol`. For the patient contour, we used bin.from.roi, specifying the RoI name (in this case, not `patient`, nor `body`, but `external`). The last operation, `bin.intersection`, gives the air inside the patient alone. If we zoom in on the last image, we can see white pixels on the sides, which will inevitably produce points that do not belong to the lung:



Selection based on images (CT, MR…) always produces isolated voxels or volumes

To clean up this result, we can attempt a clustering:

```
air.clust <- bin.clustering (air_)
```

which produces for the first clusters:

```
head (air.clust$cluster.info)

  label value volume.cc
1    NA    NA     0.000
2    bg     0 14139.322
3     1     1  4925.478
4     2     2     2.156
5     3     3     1.046
6     4     4     0.936
```

Clusters have a label (“NA” when we have NAs in the image, “bg” (i.e. background) for the volume that was set to `FALSE`, and here from “1” to a lot for the different clusters). They are sorted by volumes in descending order (except “NA” and “bg” that are always first). The associated value is that stored in `$vol3D.data`. Here, we are probably interested in the second cluster of size 4,9 liters. We need to convert the “cluster” modality into “binary” modality. As there are several ways to do this, there is no specific instruction under espadon, but it only takes 6 lines:

```
lungs.air <- air.clust
lungs.air$vol3D.data  <- air.clust$vol3D.data == 1
lungs.air$max.pixel <- TRUE
lungs.air$min.pixel <- FALSE
lungs.air$modality <- "binary"
lungs.air$cluster.info <- NULL

m.lungs.air <- mesh.from.bin (lungs.air, smooth.iteration = 1)
display.3D.mesh (m.lungs.air, col ="burlywood2")
```

Isolated volumes are removed using clustering

As we can see, we were able to perform a binary selection on the alveoli of the lungs without too much difficulty. The residual volumes have disappeared. The trachea would still have to be eliminated, but we will come back to this later. Now we will tackle the other part of the problem, namely, making a selection on the blood circuit within the lungs.

2.3 Blood circuit selection

The selection of the blood circuit is not as immediate as the pulmonary alveoli. Indeed, there is no external contour of the lung and if a selection is made directly on the CT, the entire blood circuit of the patient will emerge. To continue the exercise, we will have to look at so-called mathematical morphology techniques that are implemented in 3D in espadon.

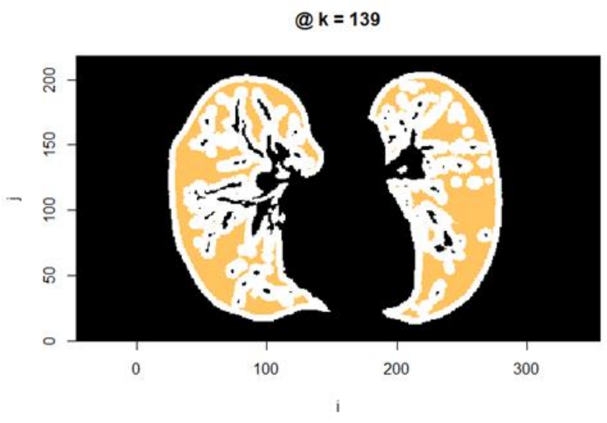
There are 4 basic operations in mathematical morphology: erosion, dilatation, opening and closing. They all use a structuring element, in espadon, a sphere of given radius. Before continuing our exercise, we will look in detail at what these operators do. To do this, we have applied them to the lung section shown above. However, as these are 3D operations, it is important to understand that the planes above and below, in an area that includes the radius of the structuring element, have consequences for the plane shown. In the following graphs. The background image (white) corresponds to the lung section, on which the operator's result (orange with transparency) is superimposed.

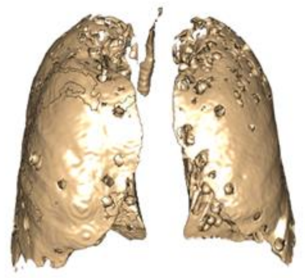
The code for obtaining the illustrations is something like that:

```
b.e <- bin.erosion (lungs.air, radius = 5)
dev.new (width = 7, height = 5, noRStudioGD = TRUE)
display.kplane (lungs.air)
display.kplane (b.e, col = c ("#FFFFFF00", "#00FF0080"), add = TRUE) # as orange overlay
display.3D.mesh (mesh.from.bin (b. e), col = "burlywood2")
par3d (windowRect = c (100, 100, 600, 600))
```

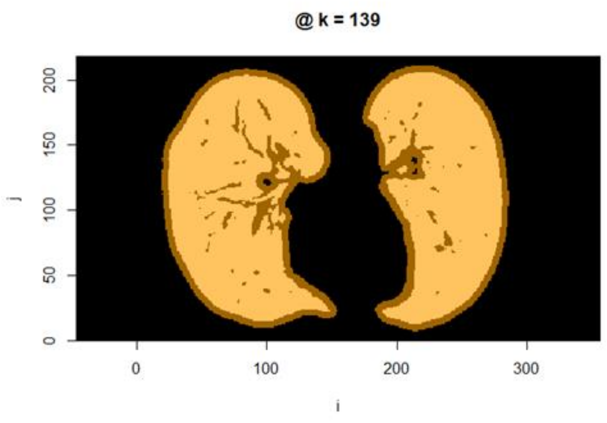
Let us examine the operator’s results:

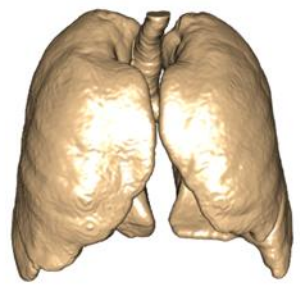
Erosion, `bin.erosion`: reduces the outer volume, digs inner hollows and disconnects objects.



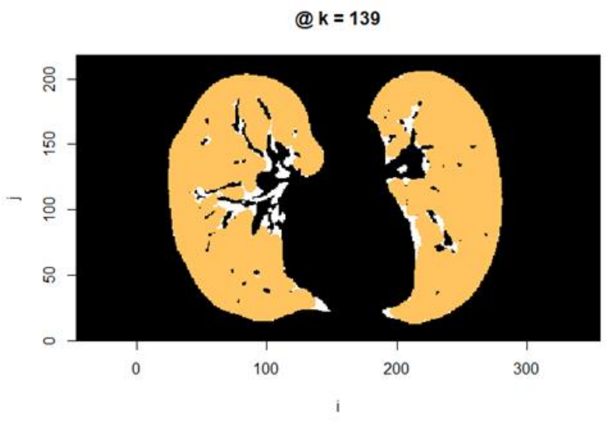


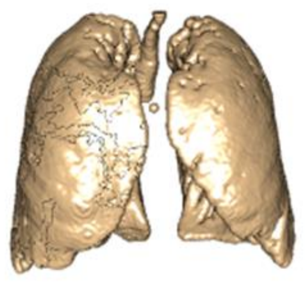
Dilation, `bin.dilation`: increases the outer volume, fills inner hollows and connects objects.



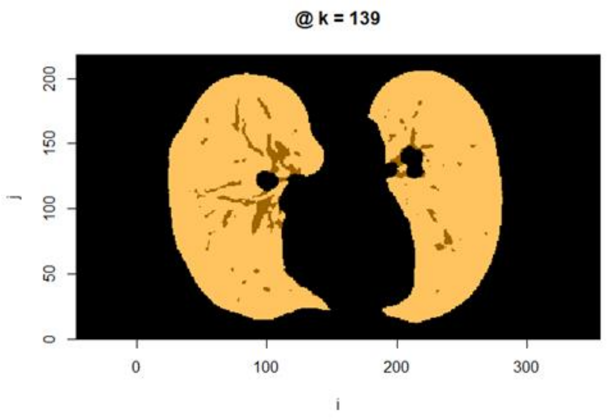


Opening, `bin.opening`: it is an erosion followed by a dilation, both having the same structuring element. It smoothies outer volume, leaving inner hollows intact.





Closing, `bin.closing`: a dilation followed by an erosion of same structuring element. Closing, leaves outer volume intact and fills inner hollows whose size is lower than structuring element.

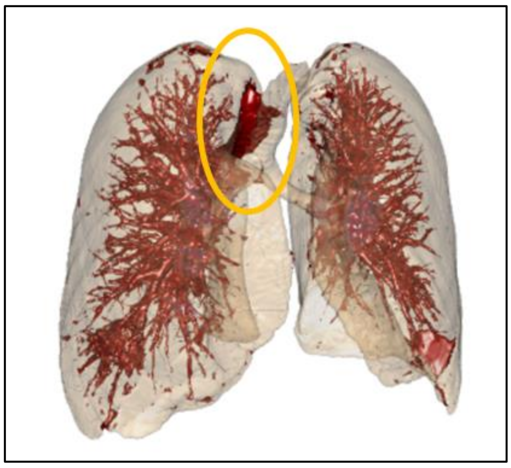


Some additional elements:

i) mathematical morphology operators are non-linear in the sense that they are not reversible, for example, an erosion followed by a dilation does not give the original image, but constitutes a closure,
ii) these operators work in 3D, so the result observed on a plane depends on what is above and below,
iii) they are actually based on convolutions (in reality we go through 3D FFTs) which are time consuming, iv) lastly, their result is not always intuitive and always requires exploration to define the size of the structuring elements.

Knowing all this, let us return to our example. To extract the blood circuit in the lungs, we can start by performing a closure. This will preserve the external volume, but block all the vessels in the lungs. We can then make a soft tissue selection on the CT, and calculate its intersection with the new lung volume. The soft tissue will then be limited to the lungs:

```
lungs.c <- bin.closing (lungs.air, radius = 8)
tissue <- bin.from.vol (lungs, min=-499, max=+Inf)
blood <- bin.intersection (lungs.c, tissue)

display.3D.mesh (mesh.from.bin (blood), col = "darkred")
display.3D.mesh (mesh.from.bin (lungs.air), col = "burlywood2", alpha = 0.3)
```

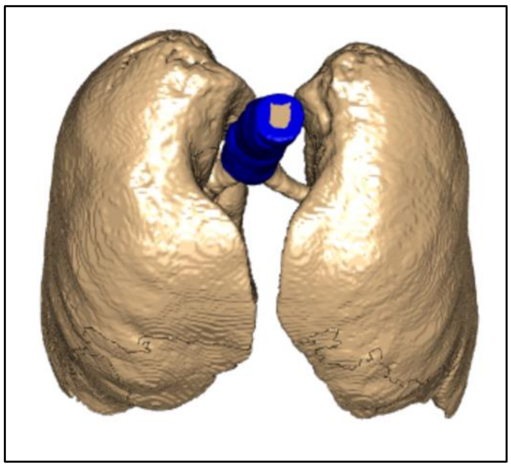

Result of the operations mentioned above. We managed to select the blood circuit inside the lungs, but unfortunately, the trachea being close to the lungs, soft tissue between trachea and right lung appear in the selection.

The closure connected the lung contour to the trachea. As a result, all soft tissue between the trachea and the right lung appeared in the selection. To finalize this example, we can try to remedy this.

The idea could be to perform an erosion, so as to detach the trachea from the right and left lungs. This would be followed by a clusterisation, allowing the trachea alone to be selected. If we understand what we are doing, the trachea should be the third cluster, the two largest being the right and left lung.

A dilation would then be necessary to regain its original shape. This new trachea would then be subtracted from the full volume to make it disappear. Let us try this:

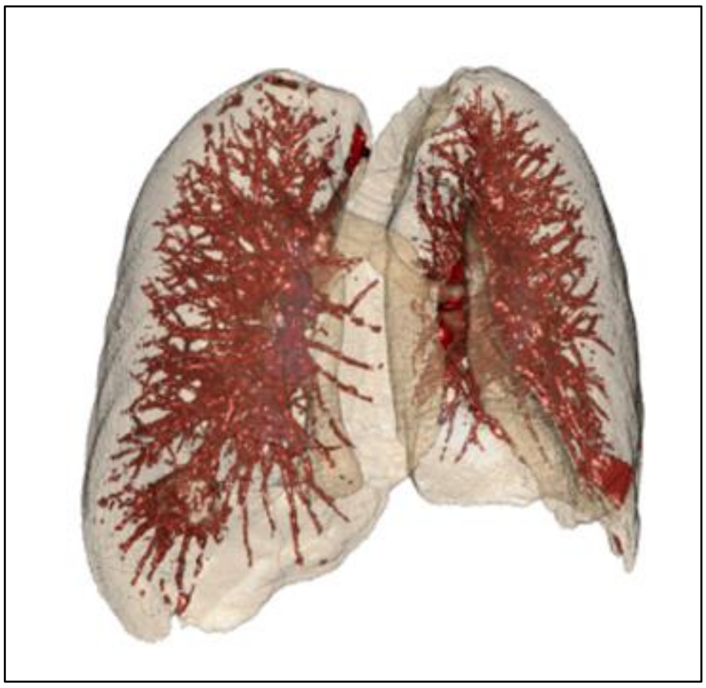
```
lungs.o <- bin.opening (lungs.air, radius = 5)
display.3D.mesh (mesh.from.bin (lungs.o), col = "burlywood2")

trachea.sel <- bin.clustering (lungs.o)

trachea <- trachea.sel
trachea$vol3D.data  <- trachea.sel$vol3D.data == 3
trachea$max.pixel <- TRUE
trachea$min.pixel <- FALSE
trachea$modality <- "binary"
trachea$cluster.info <- NULL

trachea.reconstructed <- bin.dilation (trachea, radius = 8)
display.3D.mesh (mesh.from.bin (trachea.reconstructed), col = "blue")
```


In this figure we have isolated the trachea from the rest of the airway selection.

You will notice that in the codes that have been run, the radius of the structuring element has been changed multiple times. In the previous instructions, for example, we have performed the erosion with a radius of 5 mm, and the dilation (after clustering), with a radius of 6 mm. These figures themselves are dependent on the problem at hand and are really related to its geometry. They therefore require some trial and error adjustments. Once we have found the combination that works on the test patient, we can think of generalising it on other patients, or even finding a relation between these combinations and the patient's morphology...

As we are satisfied with this result, we can continue by removing the trachea from the closing operated on lungs contour:

```
lungs.inner <- bin.erosion (lungs.c, radius = 2)
lungs.minus.trachea <- bin.subtraction (lungs.inner, trachea.reconstructed)
blood <- bin.intersection (lungs.minus.trachea, tissue)
display.3D.mesh (mesh.from.bin (blood, smooth.iteration = 0), col = "darkred")
display.3D.mesh (mesh.from.bin (lungs.air, smooth.iteration = 0), col = "darkred", alpha = 0.3)
```


Result of the full process

The result seems quite correct, and we will stop here for this example.

3 how to finalise this example?

Of course, it took a bit of experimentation to produce the results outlined in the previous paragraph (approximately 1/2 day's work from scratch). Now that we are happy with the result, it would be interesting to encapsulate the instructions in a function that does all the work, cleaning up and reordering all the slag from this experimentation. This would result in code that looks like this:

```
air.and.blood.in.lungs <- function (CT, S, patient, lungs.rois) {

  ## reducing the computation volume
  lungs <- nesting.roi (CT, S, roi.name = lungs.rois,
                        xyz.margin = c(10, 10, 10))

  ## computing of air inside the patient
  air <- bin.from.vol (lungs, min = -Inf, max = -500)
  p <- bin.from.roi (lungs, S, roi.name = patient)
  air_ <- bin.intersection (air, p)

  ## reducing air to lungs (+ trachea)
  air.clust <- bin.clustering (air_)
  lungs.air <- air.clust
  lungs.air$vol3D.data  <- air.clust$vol3D.data == 1
    # we assumed lungs is the first cluster.
  lungs.air$max.pixel <- TRUE
  lungs.air$min.pixel <- FALSE
  lungs.air$modality <- "binary"
  lungs.air$cluster.info <- NULL
  ## done for lungs air.

  ## computing trachea alone
  lungs.o <- bin.opening (lungs.air, radius = 5)
  trachea.sel <- bin.clustering (lungs.o)
  trachea <- trachea.sel
  trachea$vol3D.data  <- trachea.sel$vol3D.data == 3
    # we assumed trachea is the third cluster
  trachea$max.pixel <- TRUE
  trachea$min.pixel <- FALSE
  trachea$modality <- "binary"
  trachea$cluster.info <- NULL
  trachea.reconstructed <- bin.dilation (trachea, radius = 8)
  ## done for trachea

  ## computing blood circuit
  lungs.c <- bin.closing (lungs.air, radius = 8)
  lungs.inner <- bin.erosion (lungs.c, radius = 2)
  lungs.minus.trachea <- bin.subtraction (lungs.inner, trachea.reconstructed)
  tissue <- bin.from.vol (lungs, min = -499, max = +Inf)
  blood <- bin.intersection (lungs.minus.trachea, tissue)
  ## done for blood

  return (list (air = lungs.air, blood = lungs.blood))
}
```

You can store is in a file, for instance “my_lung_processing.R”, and use it anytime you want or need it, for instance:

```
require (espadon)
source ("my_lung_processing.R")

pat.dir <- "E:/Rdicom data/SBRT_Poumon_Test"
pat <- load.patient.from.dicom(pat.dir, data = TRUE)
S <- pat$rtstruct[[1]]
CT <- pat$ct[[1]]
D <- pat$rtdose[[1]]
lungs.rois <- c("lungl", "lungr")       # here, put the name of R & L lungs
```

```
patient <- "external"                   # here, the name of patient’s contour

pat.lungs <- air.and.blood.in.lungs (CT, S, patient, lungs.rois)

## here, you can do what you want using pat.lungs$air or pat.lungs$blood
## which are binary volumes. For instance:
get.volume.from.bin (pat.lungs$air)

[1] 4925.478

get.volume.from.bin (pat.lungs$blood)

[1] 233.349
```

This example requires a large amount of calculation. The production of the two volumes takes approximately 3.5 minutes. Of course, it is recommended to store the result when the computations are finished...

4 What we learned

Espadon has multiple functions that allow new volumes to be purposefully constructed with clear definitions. Of course, there are many ways to achieve a satisfactory result, and only trial and error methods can achieve this. The generalization of processes must also be ensured.

It is interesting to note the following. In general, contouring is done by means of TPS, using anatomical atlases or, manually when precision is needed.

The undeniable advantage of manual contouring by the radiotherapist is that it uses all his knowledge of the problem and his intelligence. However, this solution can quickly generate errors, even if only with the adjustment of the ambient light or the screen.

For instance, it is reasonable to assume that the contouring of a 10 mm radius organ is executed to within +/- 1 mm plane by plane. Take, for example, a uniform distribution of 9.5 to 1.05 mm. What frequently comes into play in radiotherapy is not the radius, but the volume of the organ. In the case of a sphere, an error of just 1 mm (10%) on the radius translates into an error of 30% on the volume.

The indisputable advantage of an automatic method is that it will always make the same error, day after day, which tends to stabilize the uncertainty of models that might be built from measured data.

The idea is then to benefit from both universes by using the intelligence of the human to point out what is important, and the stupidity, but consistency of the machine to perform the systematic measurements.

If you are interested in improving volume definitions with espadon, please consult the user manual. You can also discover some of the possibilities by looking at the following instructions:

```
add.margin                      Adding or removing a margin to a volume
bin.closing                     Binary volume closing
bin.clustering                  Binary volume clustering
bin.dilation                    Binary volume dilation
bin.erosion                     Binary volume erosion
bin.from.roi                    Creation of a binary volume according to RoI
bin.from.vol                    Creation of a binary volume according to the voxel values of a volume
bin.intersection                Intersection of two binaries
bin.inversion                   Inversion of a binary
bin.opening                     Binary volume opening
bin.subtraction                 Subtraction of two binaries
bin.sum                         Sum of two binaries
get.volume.from.bin             Volume selected by binary volume
get.volume.from.roi             Volume of a region of interest (RoI)
nesting.cube                    Restriction of a volume to a rectangular parallelepiped
nesting.roi                     Restrict volume to RoI
```