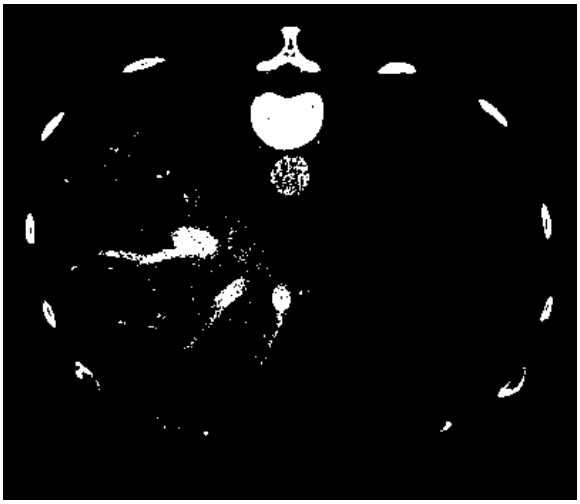
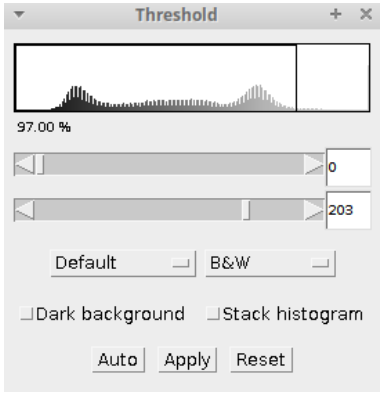
TP2

Segmentation semi-automatique du réseau vasculaire du foie dans une image CT

1. En observant l'image du foie, on peut choisir un seuil de 97%, ce qui donne l'image ci dessous.



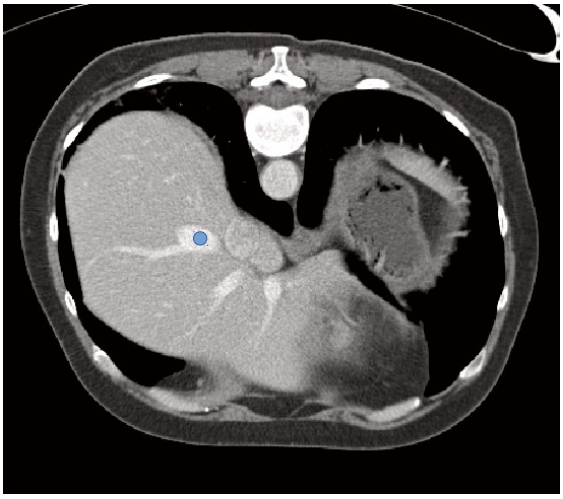


liver.hdr : seuil à 97%

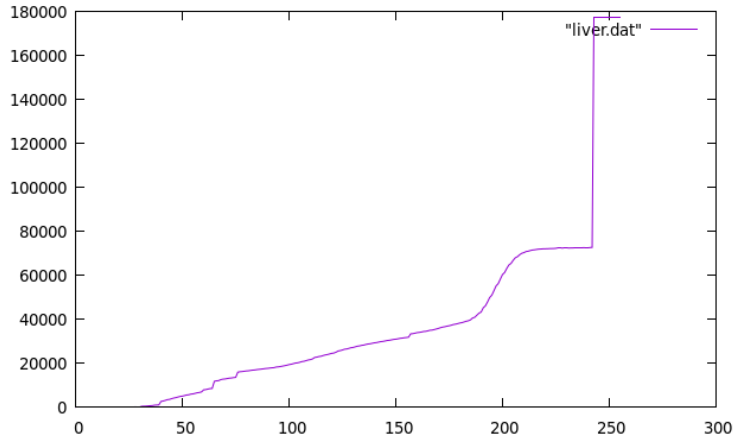
Si on choisit un seuil supérieur, on va perdre trop d'information sur le réseau vasculaire, qui ne sera plus affiché. Avec un seuil inférieur, le foie va s'afficher, et il y aura trop de bruit pour pouvoir isoler le système vasculaire.

Sortie de l’algorithme :

Sur liver.hdr :



En bleu la zone cliquée



Histogramme associé

On obtient un seuil situé entre 40 et 45, ce qui se rapproche du seuillage manuel effectué.

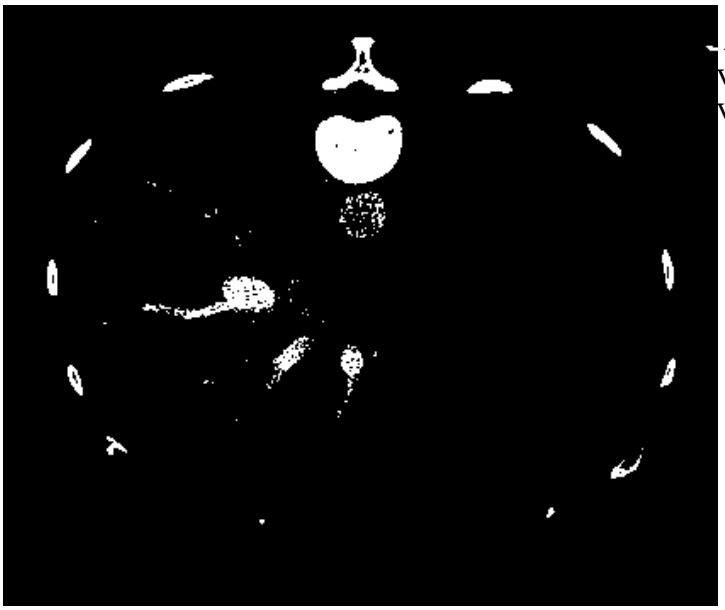
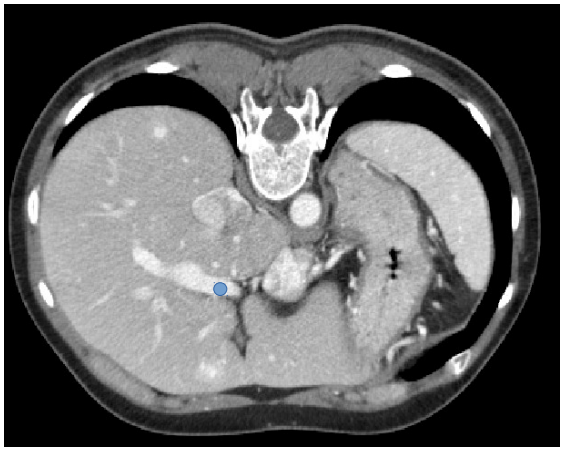


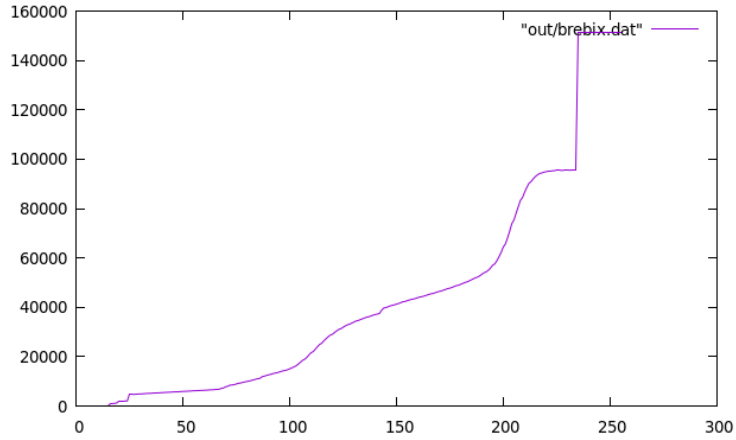
Image avec un seuil de 45

Avec ce seuil on retrouve les différentes veines du foie, ainsi qu’un partie de la vésicule billiaire.

Sur brebix.hdr :



En bleu la zone cliquée



Histogramme associé

Ici, on obtient un seuil entre 25 et 30.

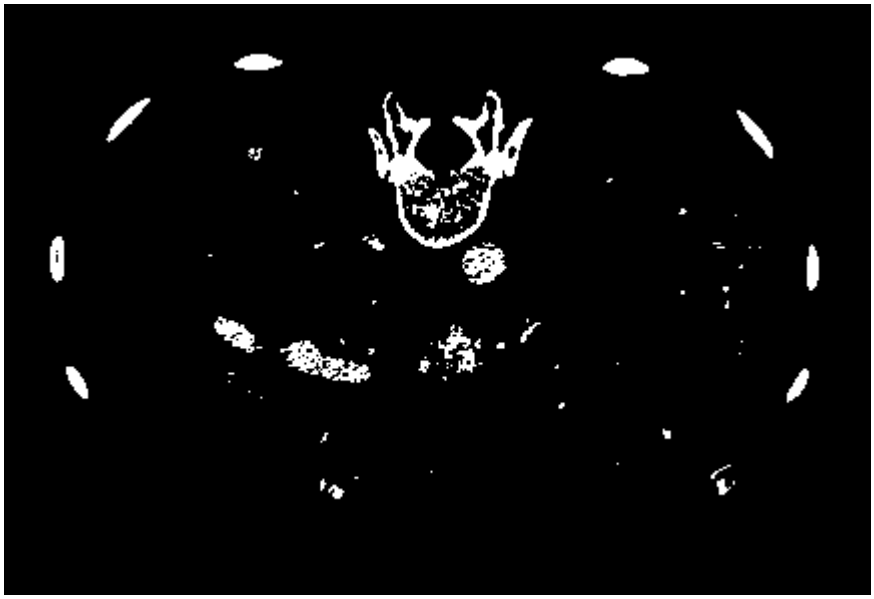


Image seuillée à 30

Pour remplacer la fonction draw_fill, on pourrait utiliser une fonction dont le principe serait le suivant. Plutôt que de garder en mémoire toute une image, qui n'est pas affichée, on garderait simplement un tableau contenant la position des voxels (sous forme d'une structure par exemple). Ce tableau serait ensuite étendu en ajoutant les nouveaux voxels en parcourant l'image originale en fonction du seuil de tolérance défini par l'utilisateur.

Annexes : code du programme

```
int main(int argc, char *argv[]){
    char cNomImgLue[250], nomHisto[250];
    if (argc != 3) {
        std::cerr << "Usage: ImageIn.hdr out/nomhisto.dat" << std::endl;
        exit(1);
    }
    sscanf(argv[1], "%s", cNomImgLue);
    sscanf(argv[2], "%s", nomHisto);
    int cpt = 0;
    CImg<> img(cNomImgLue);
    CImgDisplay display(img, cNomImgLue);
    std::ofstream histo(nomHisto);
    while (!display.is_closed()) {
        if(display.is_keyESC())
            break;
        if(display.button()){
            int x = display.mouse_x() * (float)((float)img.width() / (float)display.width());
            int y = display.mouse_y() * (float)((float)img.height() / (float)display.height());

            std::cout << "Clicked at : x = " << x << ", y = " << y << ". Gray level is : " << img(x, y, cpt) << std::endl;

            for(int i = 0; i <= 255; ++i){
                CImg<> image = img;
                CImg<> region;
                int regionSize = 0;
                unsigned char white[] = {255, 255, 255};
                image.draw_fill(x, y, cpt, white, 1, region, i);
                for(int j = 0; j < region.width(); ++j)
                    for(int k = 0; k < region.height(); ++k)
                        if(region(j, k) > 0)
                            ++regionSize;
                histo << i << " " << regionSize << std::endl;
            }
            std::cout << "End procesing" << std::endl;
            histo.close();
        }
        if(display.wheel()){
            cpt += display.wheel();
            display.set_wheel();
        }
        display.display(img.get_slice(cpt));
        display.wait();
    }
    return 0;
}
```