PlantSeg is a Matlab-based tool that simplifies and speeds up the **segmentation** and **analysis** of roots and shoots from raw plant images. The tool's user interface is shown in Fig. 1. For automatic segmentation, the tool uses a combination of image transformations (histogram equalization), thresholding on different color spaces (RGB, YCbCr, Lab, HSV), and segmentation mask post-processing (boundary removal, islands removal) to automatically segment the roots and shoots. For analysis, the tool uses connected components analysis to extract the relevant plant masks, allowing the tool to notify the user if there are fewer or more plants than expected in the segmented mask. In these cases, the user can view the detected connected components and has the option to manually correct the automatic segmentation's results via simple brush strokes directly on the segmentation mask and/or tweak the segmentation parameters and re-run the segmentation. Once the user is satisfied with the segmentation, the number of pixels corresponding to roots and shoots are counted and recorded automatically.

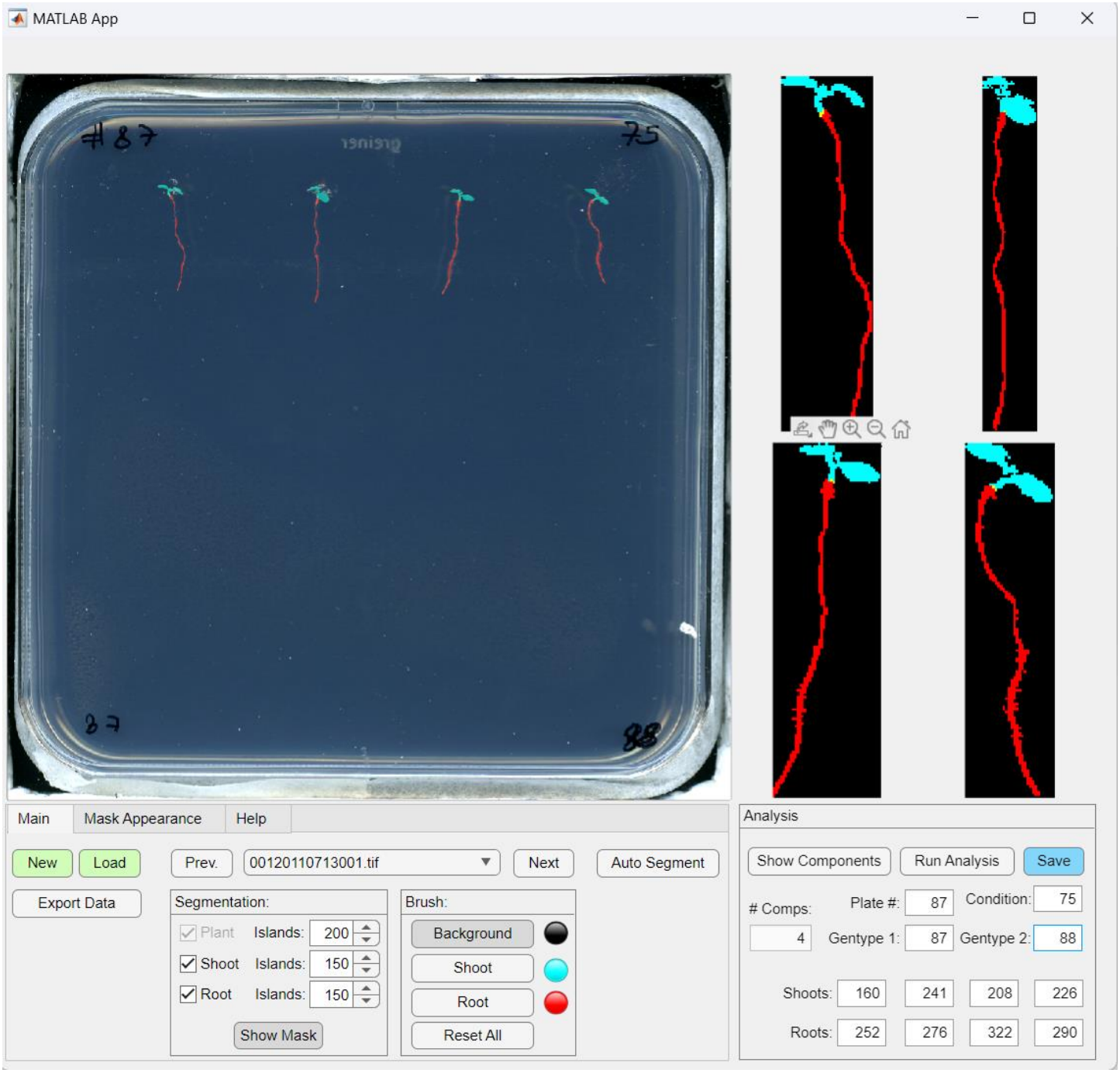

Fig. 1. Graphical user interface for our root and shoot segmentation and analysis tool based on Matlab. The input image with an overlay of the automatic segmentation result is shown on the upper left. The four largest detected

connected components are shown on the upper right with the roots (red) and shoots (cyan) segmentation. On the lower left, the user has several options, including parameters and controls for segmentation. On the lower right, the user can see the analysis results as well as entry points for the sample details (plate, condition, gentypes).

We describe below the automatic segmentation process for a simple case where the user does not need to make manual corrections or parameter tweaks.

We start with the input image (Fig. 2).

Fig. 2. Raw input image.

We first apply histogram equalization to the input image to improve the image contrast, making it easier to separate the roots, shoots, and background. The result of histogram equalization is shown in Fig. 3.

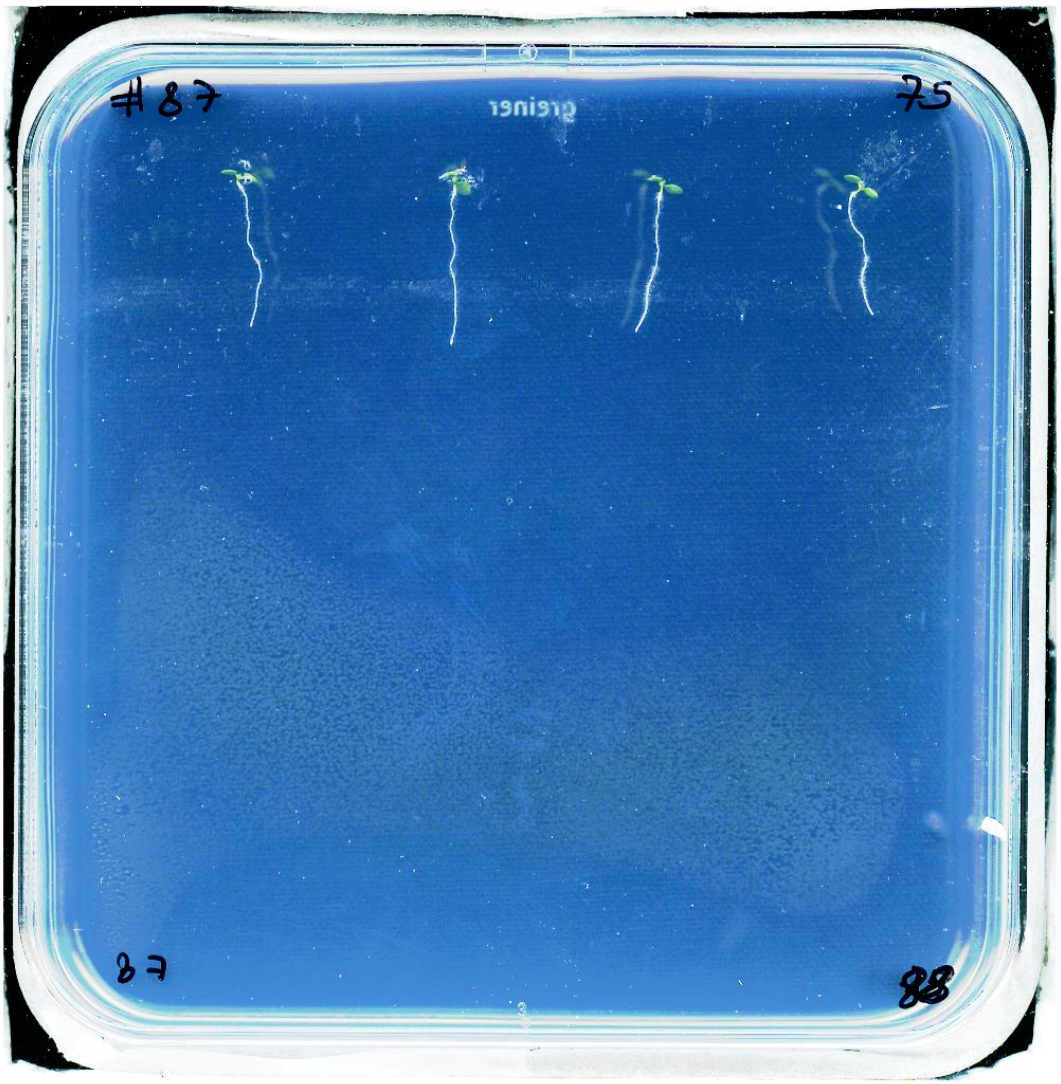

Fig 3. Histogram equalized input image.

The histogram equalized input image is then used for the automatic segmentation. The automatic segmentation consists of three steps: 1) segmentation of the plant, i.e., including both shoots and roots, 2) segmentation of roots, and 3) segmentation of shoots. The initial segmentation of the plant is important to first separate the plant from the background, which can have some challenges, including some unwanted artifacts such as reflections, boundary issues, and noisy white spots around the image. To segment the plant, i.e., roots and shoots together, we apply 1) color transformations (RGB, YCbCr, Lab) on the histogram equalized image followed by 2) thresholding to separate the plants from the background, and 3) boundary and islands removal. The thresholding is initially guided by the color thresholder application in Matlab shown in Figs. 4-6 but the chosen thresholds are later exported to a script that can be reapplied automatically to any image. In other words, the manual threshold setting shown in Fig. 4-6 is only done once for the whole dataset. These threshold values can be easily updated if the image capture settings are changed, e.g., when using a new dataset with different camera settings or background color.

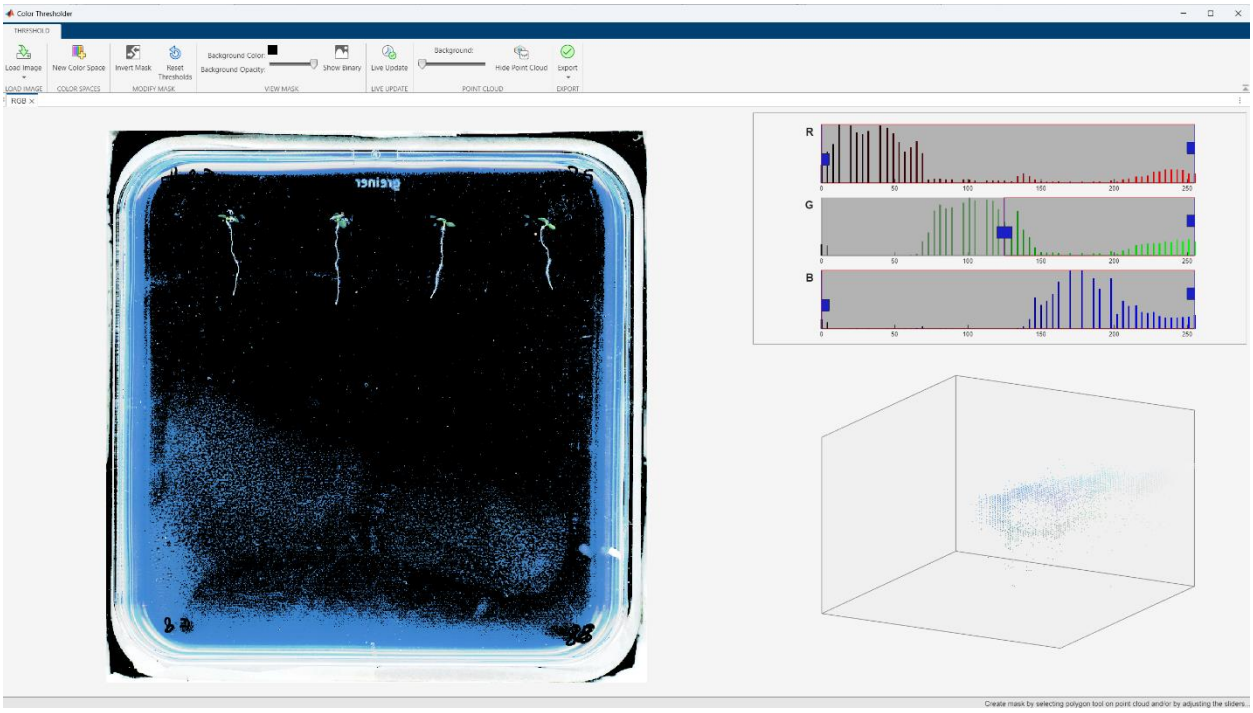

Fig. 4. Thresholding the histogram equalized image in RGB space. We accept pixels with G (green) channel >= 128.

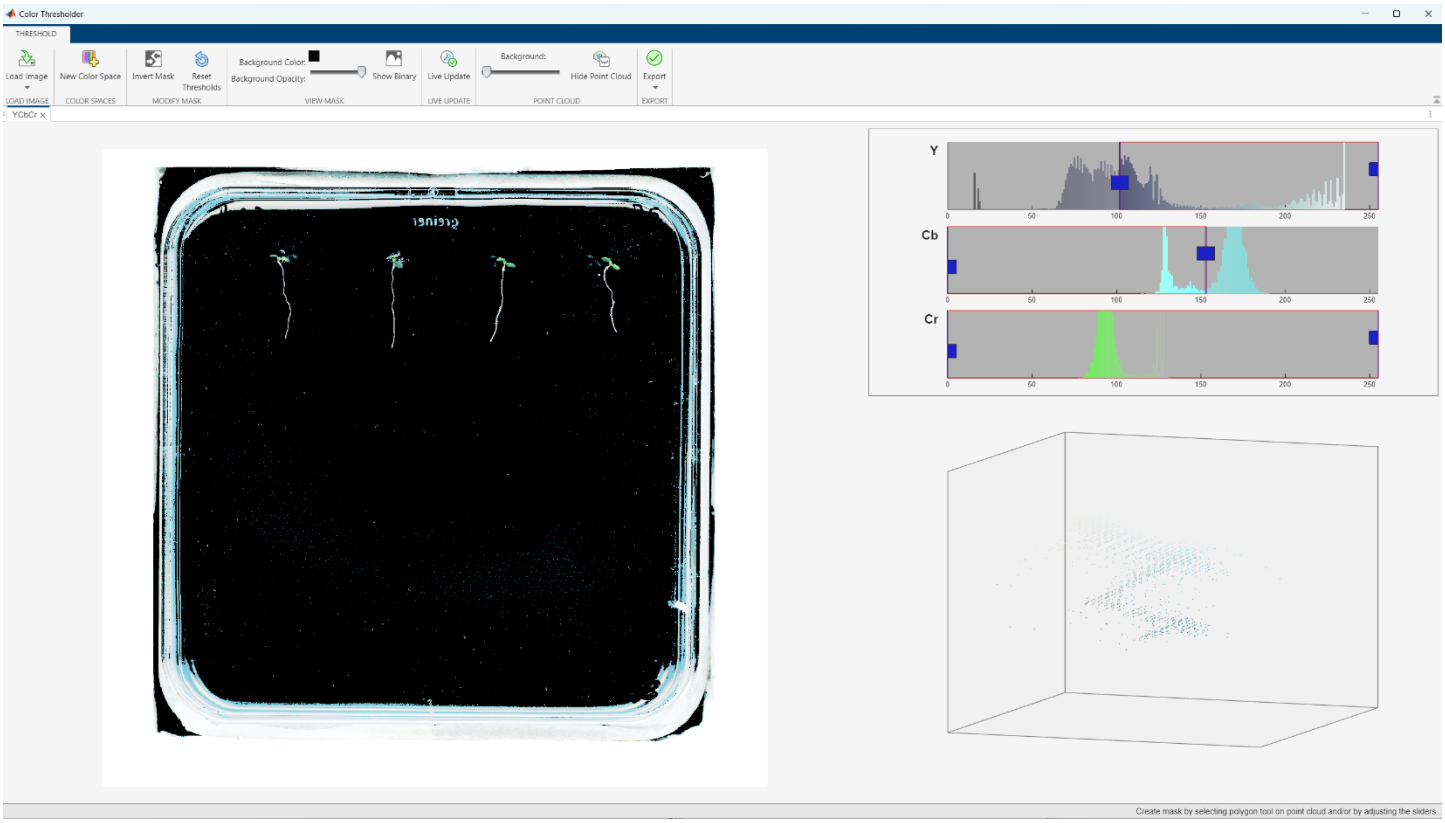

Fig. 5. Thresholding the histogram equalized image in YCbCr space. We accept pixels with Y >= 107 and Cb <= 156.

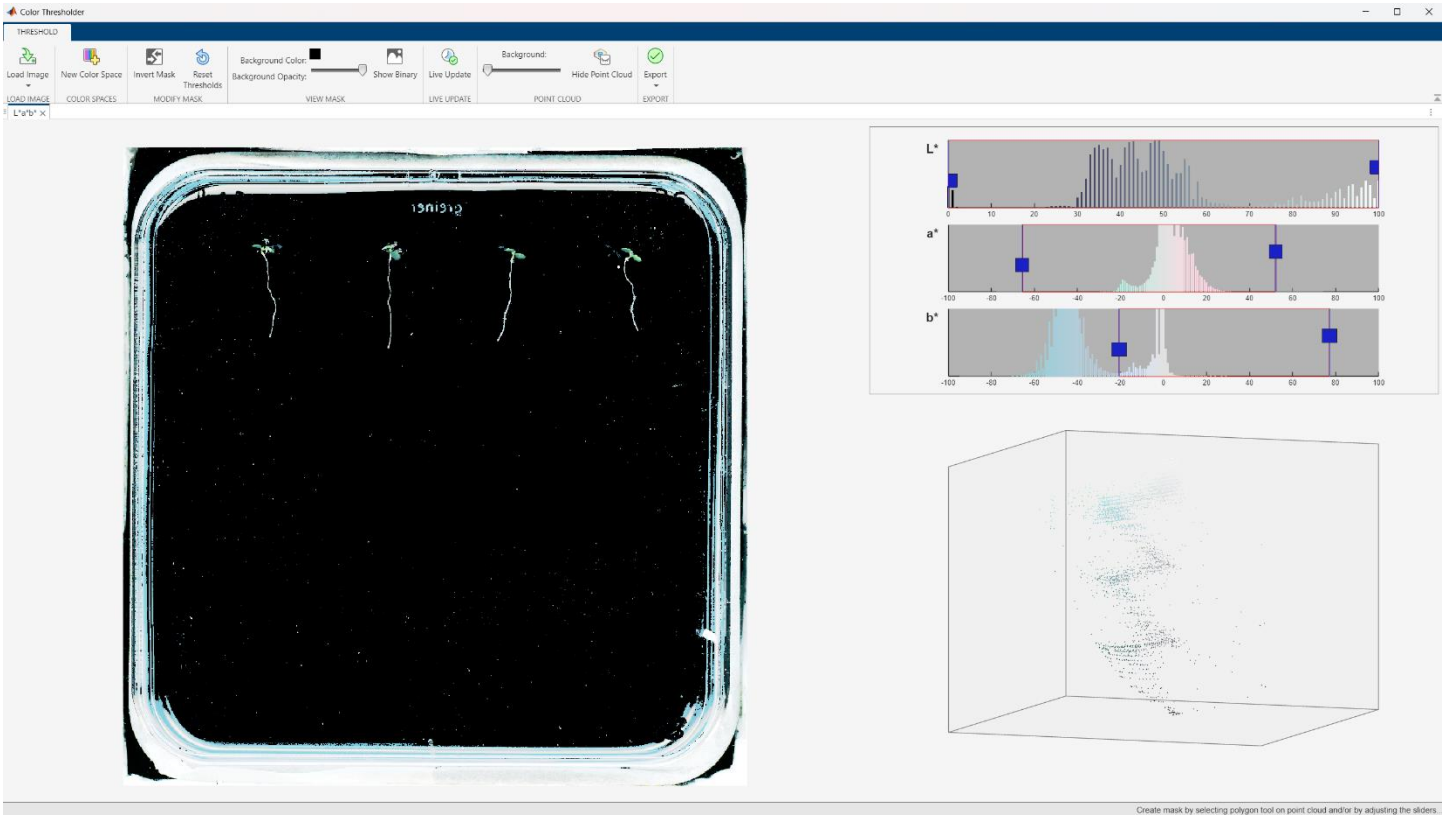

Fig. 6. Thresholding the histogram equalized image in Lab space. We accept pixels with b >= -20.7.

After applying the thresholding in three different color spaces, we combine the results via logical AND operation to only accept pixels that are accepted in all color spaces. For our example above, the result is shown in Fig. 7. As we can see in Fig. 7, the mask still includes the boundary of the slide as well as small dots that come from noise in the original image. We then remove components that are connected to the border (we use Matlab's imclearborder function) and remove the small islands, i.e., connected pixel neighborhoods smaller than a user specified parameter – we set this to 200 by default, but the user can tweak it as needed via the user interface (we use Matlab's bwareaopen function). The result after removing the border and then removing the islands are shown in Figs. 8 and 9 respectively.

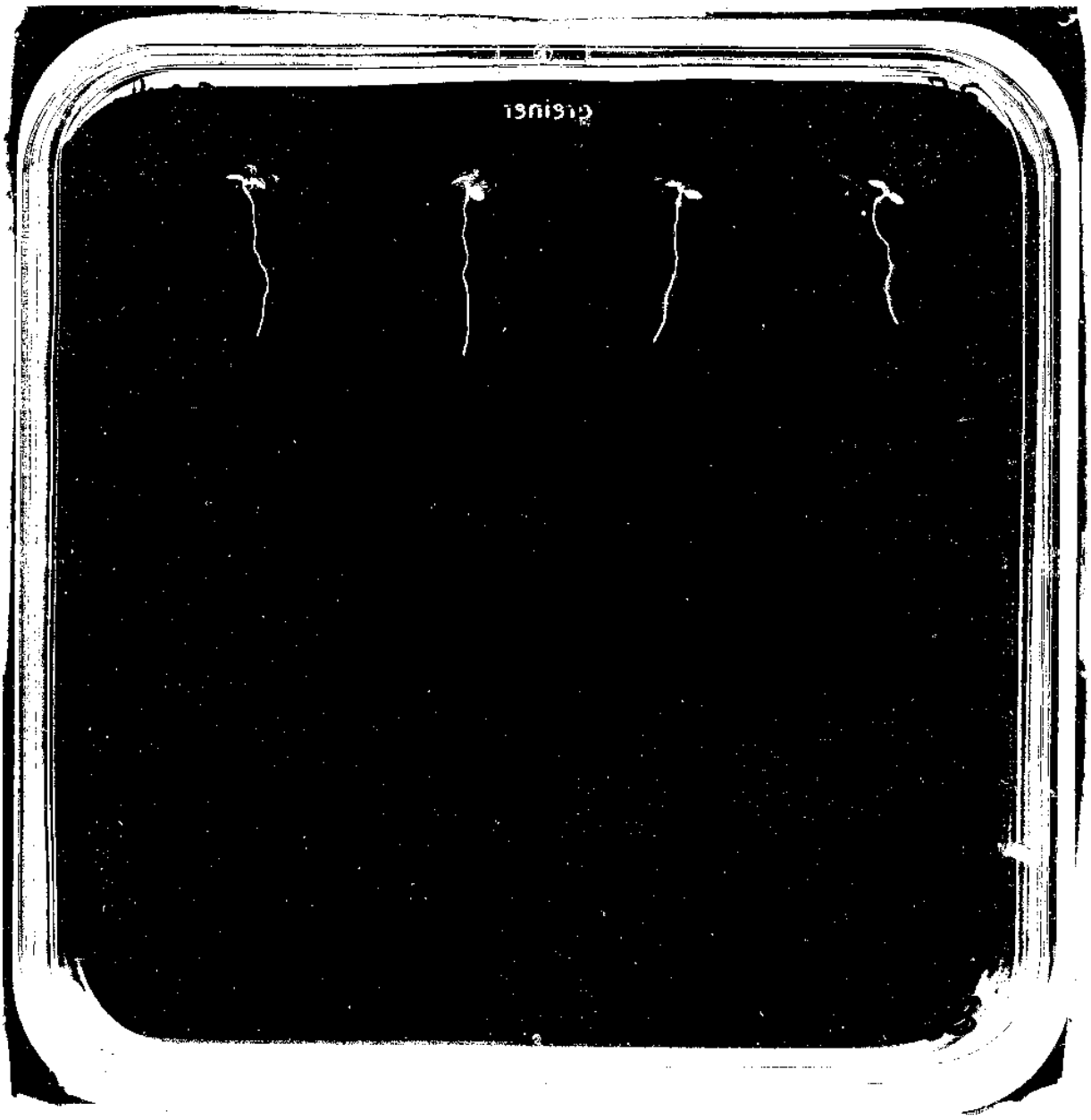

Fig. 7. Result of logical AND operation on three segmentations masks generated from thresholding in three different color spaces.

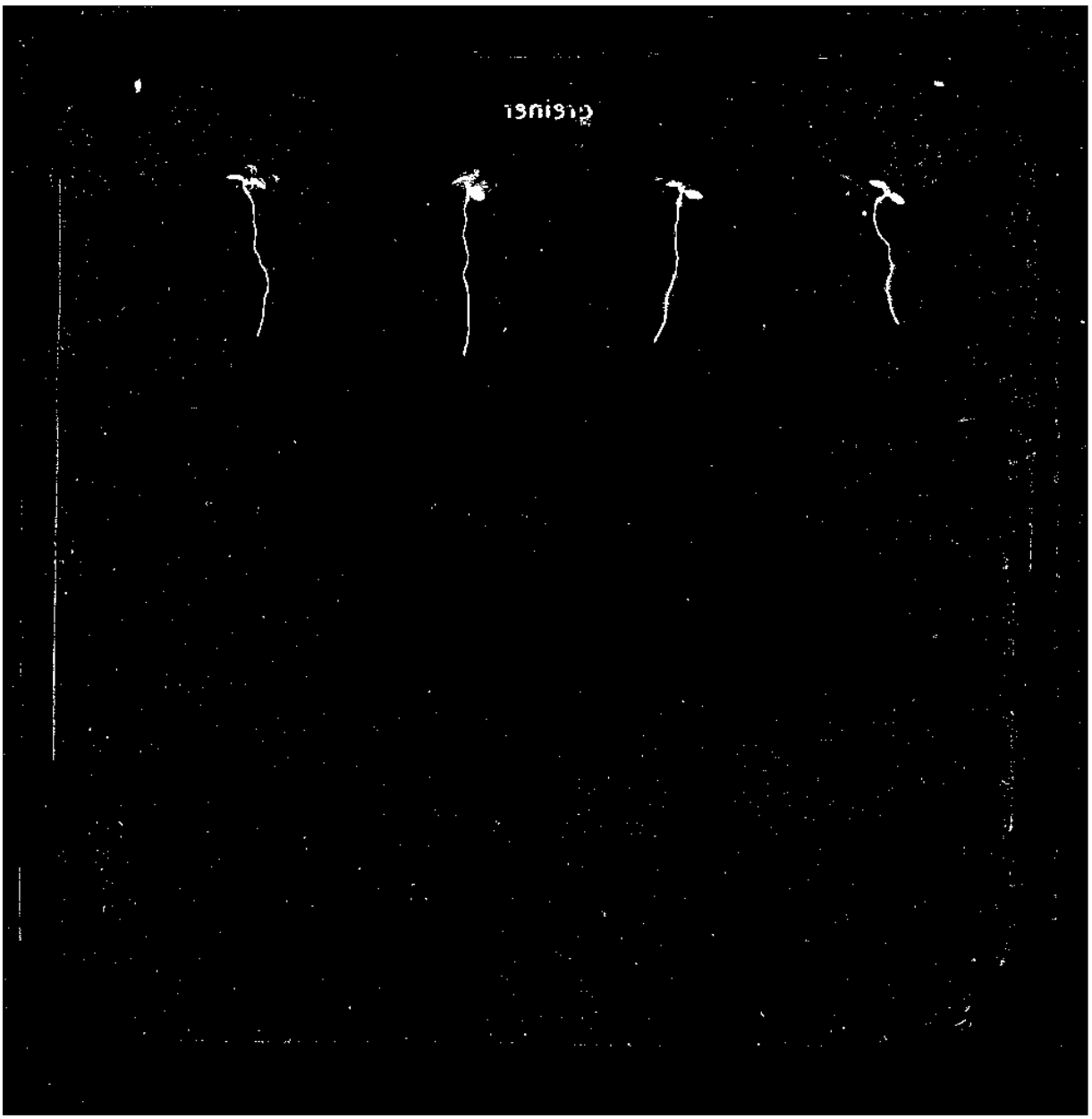

Fig. 8. Result after removing boundary artifacts.

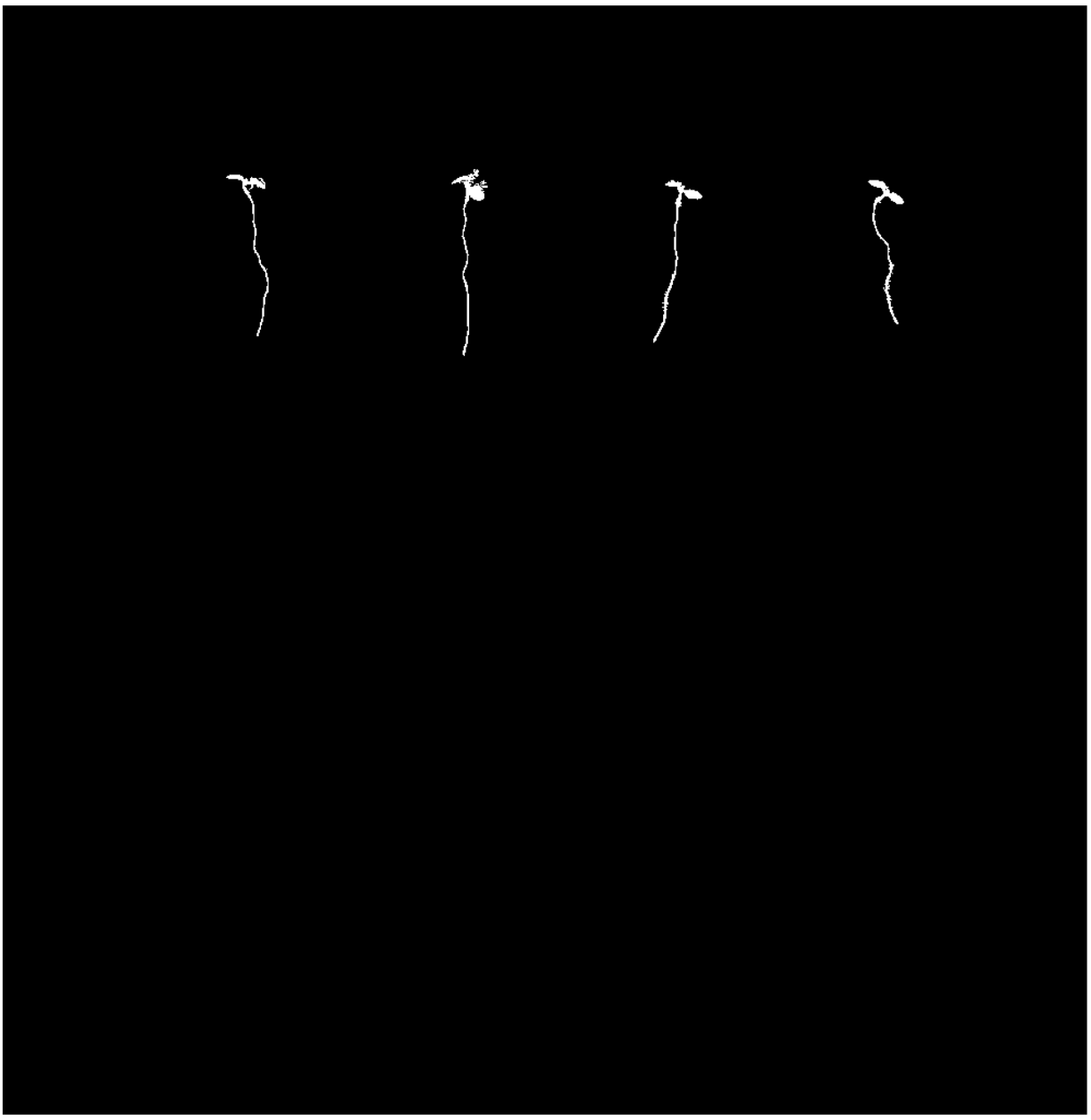

Fig. 9. Result after removing islands. This is the final segmentation mask of the plant (includes both roots and shoots).

We then use the segmentation mask of the plant to isolate the plant pixels from the rest of the image. On the isolated pixels, we then proceed to apply different color space transformations and thresholding to extract the roots and shoots. For roots, we use Lab color space of the masked input image and masked histogram equalized image for thresholding followed by border cleaning and islands removal. Segmented roots results are shown in Fig. 10. For shoots, we use HSV color space of the masked input image for thresholding followed by border cleaning and islands removal. The segmented shoots for our example are shown in Fig. 11. The threshold values can be inspected in our provided Matlab code.

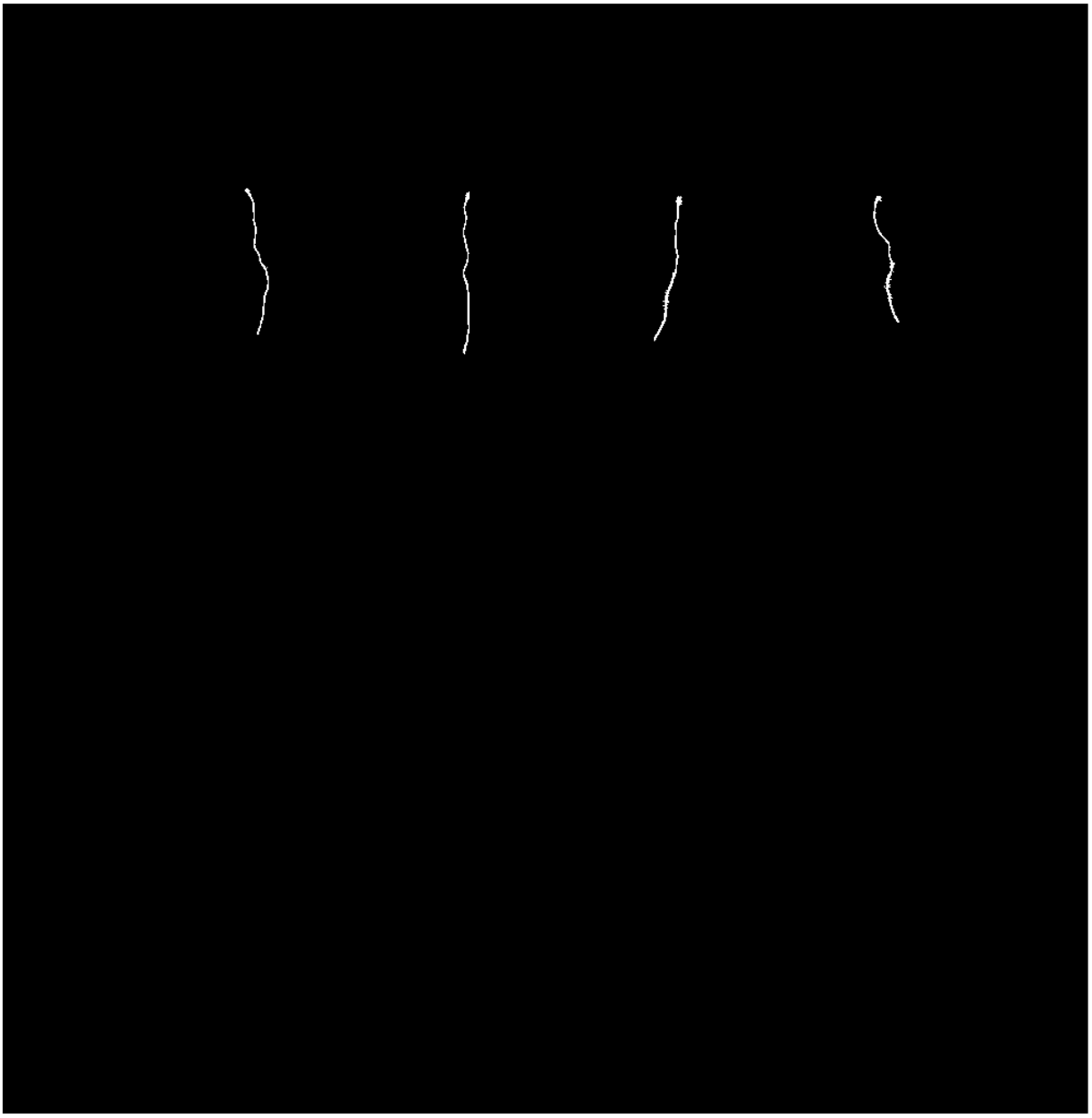

Fig. 10. Final segmented roots from our sample image.

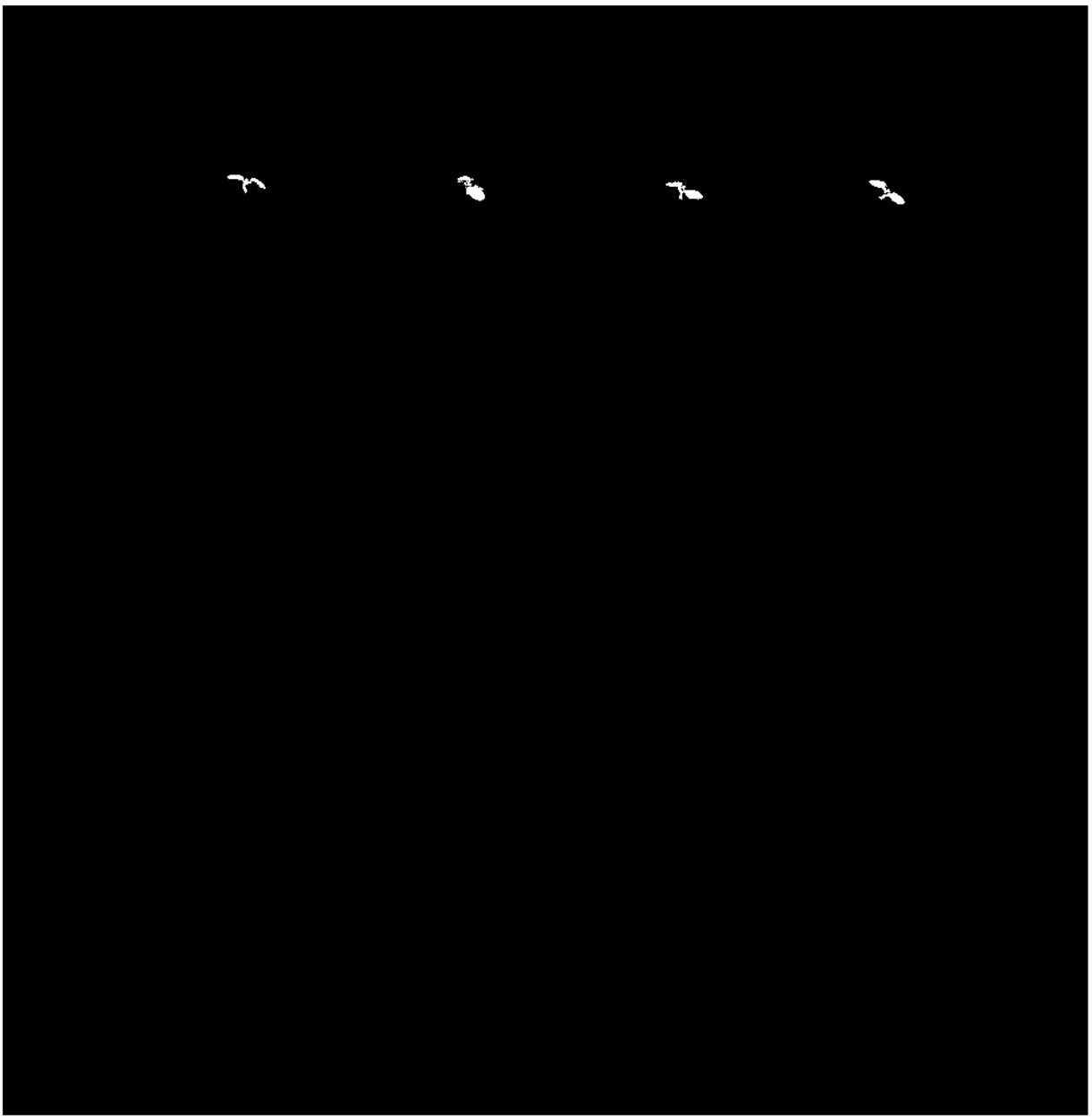

Fig. 11. Final segmented shoots from our sample image.

After segmentation, the user can run the analysis where the number of segmented plants is detected and the number of pixels for the roots and shoots of each plant are recorded and stored for later analysis. The number of plants is determined via counting of connected components in the plant segmentation mask. For most images, the number of connected components is expected to be four. If the number of detected components is fewer or greater than four, the user is notified (# Comps box will turn red) and the user can then inspect the detected components visually. The components image can guide the user to where the manual editing can be done to fix the segmentation results. Once the user is satisfied with the segmentation results, the analysis can be executed again to recount the plants and the pixels for roots and shoots.

In some difficult cases, e.g., the images can be noisy, the roots can be attached to the boundary, or the shoots color are too dark, the automatic segmentation will not give perfect results. In such situations, the user can directly fix the segmentation results by brushing directly on the image and manually assign labels to the image pixels (background, root, or shoot). Overall, the automatic segmentation and analysis gives very good results that require very minimal manual corrections, thus speeding up the image analysis pipeline.

The code and executables are publicly available in https://github.com/ronellsicat/PlantSeg.