Digital Geometry (UGE M2 Informatique “Imagery Science”)

Laboratory Sessions 1-2:

Counting and measuring grains[*]

Yukiko Kenmochi

October–November 2024

Practical information

Submission: Upload a compressed file containing your code and report with the file name of "your last name" via eLearning.

Deadline: December 20th, 2024.

Evaluation environment: Linux ubuntu (This means that your program will be compiled and ran in a Linux environment for the evaluation, without any special setting.)

Note: Please include with your code a CMakeLists.txt file, which facilitates its compilation, and a Readme file, which gives examples of command lines. Please do not include files which are not necessary e.g., binary files obtained from a compilation of the code.

Preparation

Install DGtal

You need Digital Geometry Tools and Algorithms Library, DGtal for the laboratory sessions. On the PCs in the laboratory room, the library is not installed by default. Thus you need to settle the working environment where DGtal is pre-installed.

[*]The original subject was prepared with my former PhD student, Kacper Pluta, in 2016.

In order to install DGtal in your directory of a laboratory PC (or on your own PC), for which either Linux or Mac OS is recommended:

1. download the package here, and
2. install DGtal following the instruction.

Please also see here for quick install. As only 2D images are treated for this laboratory session, no particular install option is required except for CAIRO.

If you would like to install DGtal in your own directory, after doing "make" without "sudo make install", you simply add the following line in the beginning of CMakeLists.txt of your project:

```
SET(CMAKE_INSTALL_PREFIX ${CMAKE_INSTALL_PREFIX} "yourDirectory/DGtal/build")
```

Make your project using cmake tools

It is recommended to use cmake tools for your project as your program needs to be associated to DGtal with all its dependencies. In your project directory, please make a CMakeLists.txt file, following the instruction here. You can also download an example here.

If you can compile a simple program such as the example *helloworld.cpp* (see here) with cmake, for example:

```
mkdir build
cd build
cmake ..
make
```

then, you must be ready to start the following experiments.

Experiments

Step 1: Download and observe segmented images

Download the compressed file, RiceGrains.zip, containing the original colored images[1] and their segmented images of the following three different types of rice grains:

(A) Basmati rice (Original: Rice_basmati.pgm; Segmented: Rice_basmati_seg_bin.pgm),
(B) Camargue rice (Original: Rice_camargue.pgm; Segmented: Rice_camargue_seg_bin.pgm),
(C) Japanese rice (Original: Rice_japanese.pgm; Segmented: Rice_japanese_seg_bin.pgm).

From simple visual observation of those images, characterize the general grain shape of each rice type in the report.

Hereafter, we treat the pre-segmented images, namely binary images whose pixel value is 255 if the pixel is in a rice grain and 0 otherwise.

[1] I thank my former colleague, Michel Couprie, for his help with taking very nice pictures of rice grains!

Step 2: Count grains

Read each binary image (the file name must be "xxx_seg_bin.pgm") and count how many connected components are in each image. You can download the program, main.cpp, to start your project.

1. Use PGMReader or GenericReader to read each input image.
2. Convert a binary image to a "digital set" following the instruction of "DigitalSet from threholded image".
3. Construct a "digital object" from a "digital set" (see here) with a choice of a good adjacency pair (in other words, set a digital topology over a digital space). Here, you should choose $(4,8)$ (4-adjacency for the foreground and 8-adjacency for the background) as the input binary images were made with this setting.
4. Compute connected components of a "digital object" using WriteComponents.

Verify if the images are well-composed or not. If not, you can optionally show which parts in the images cause the ill-composedness.

Try to eliminate the grains whose entire bodies do not appear in the image frame.

Step 3: Extract digital object boundary

For each connected component, we extract the inter-pixel boundary. Visualize such inter-pixel boundaries in your report (create pdf, svg, or png files and show them in your report).

1. Make a cubical complex from a "digital object" corresponding to each connected component. For that, we first create a topological space made from grid cells, which is called a Khalimsky space in DGtal, and initialize the space from the "digital object" (see here for the more information).
2. Extract the boundary of each connected component as a set (sequence) of 1-cells (see here for the practical information). If you have a problem when you use findABell for finding an initial bell randomly, please set the parameter "nbtries" to be larger (such as 10000).
3. Use Board2D to visualize the results.

Step 4: Polygonize digital object boundary

First of all, make a “grid curve” object from a boundary (see here). For polygonization of each boundary, there are various methods. Among them, use the method for recognizing digital straight segments, and make a polygonization, called GreedySegmentation in DGtal, without any parameter setting (see also here for an example).

Visualize the results using drawLine or drawPolyline, for example. You can also see here for another example.

Step 5: Calculate area

For each connected component, calculate its area in the following two ways:

1. as the number of the 2-cells, which is equal to the number of grid points in a "digiatl object";
2. as the area of the polygon of a "digital object".

In order to calculate the area of a polygonal shape, you can use Shoelace formula, which requires the coordinates of all the polygonal vertices. The coefficients and associated information of each polygonal segment can be obtained; this page may help you.

Mention in the report whether those area measurements maintain the multigrid convergence or not, citing the theoretical and experimental results seen during the courses.

Once you calculate the areas for all grains in an image, observe the distribution of estimated areas for each rice type, and analyze it. Can the area be a principal component for classifying grains into different types? Answer those questions in the report.

STEP 6: CALCULATE PERIMETER

Similarly to the area calculation, for each connected component, calculate its perimeter in the following two ways:

1. as the number of the 1-cells of the boundary of a "digital object";
2. as the perimeter of the polygon of a "digital object".

Mention in the report whether those perimeter measurements maintain the multigrid convergence or not, citing the theoretical and experimental results seen during the previous courses.

Once you calculate the perimeters for all grains in an image, observe the distribution of estimated perimeters for each rice type, and analyze it. Can the perimeter be a principal component for classifying grains into different types? Answer those questions in the report.

STEP 7: PROPOSE AND CALCULATE CIRCULARITY

For each connected component (segmented grain), we calculate a shape circularity based on the previously calculated area and perimeter, whose methods guarantee the multigrid convergence property.

1. Propose a circularity definition (write a mathematical formulation in the report).
2. For grains of each rice type, calculate their circularities following your definition.
3. Compare the results between different grains, and make a discussion on characterization of the grain shapes using your circularity measure.

STEP 8 (OPTIONAL): FIND USEFUL MEASURES FOR THE GRAIN CLASSIFICATION

For each rice type, calculate the above geometric measures (area, perimeter, and circularity) of all the connected components, and make a statistical study of them (summarize your data using indexes, such as mean, standard deviation, etc. and/or visualize them using a boxplot, for example). If you find some outliers, you can ignore them from your analysis. In this case, please explain how to detect the outliers.

Verify if the geometric measures are useful for the shape characterization of rice grains. Are they useful enough to classify grains into the three different types (Basmati, Camargue, and Japonais)? If not, what other measures should be added?

STEP 9 (OPTIONAL): CLASSIFICATION OF GRAINS

You can test your classification idea using images in which grains of different types are mixed. Two images are available here (Original color images: Rice_mixed2.pgm and Rice_mixed3.pgm; Segmented images: Rice_mixed2_seg_bin.pgm and Rice_mixed3_seg_bin.pgm).

STEP 10 (OPTIONAL): IMPROVE GRAIN SEGMENTATION

As you probably notice, the segmentation results of the given binary images are not perfect. One way to improve such segmentation results is using knowledge of object shapes. Propose your ideas if any.