3. Übungen 3

Donnerstag, 5. Mai 2022 17:39

Aufg. 3.2. Erstellen Sie ein ImageJ-Plugin, das von einem 8-Bit-Grauwertbild das kumulative Histogramm berechnet und als neues Bild darstellt.
Hinweis: Verwenden Sie zunächst die Methode `int[] getHistogram()` der Klasse `ImageProcessor`, um das Histogramm des aktuellen Bilds zu ermitteln, und berechnen Sie im resultierenden Array das kumulative Histogram „in-place“ mit der Formel

$$H(i) = \begin{cases} h(0) & \text{für } i = 0 \\ H(i-1) + h(i) & \text{für } 0 < i < K \end{cases}$$

Erzeugen Sie ein neues (leeres) Bild von passender Größe (e. g., 256×150) und tragen Sie darin das skalierte Histogramm als schwarze, vertikale Balken ein, so dass der Maximaleintrag exakt die verfügbare Bildhöhe einnimmt. Das Beispiel-Plugin in Prog. 3.4 (`Create_New_Image.java`) zeigt, wie ein neues Bild erzeugt und dargestellt werden kann.

Jan Andre Freirich *

```java
@Override
public void run(String s) {

    ImageTools tools = new ImageTools().withLoadedImage( path: "/Users/ftwr/WORKSPACES/W_UNI/W_DIGIBILD/" +
            "HelloWorld_DigiBild/images/lennaGrey.png");
    tools.showImage();

    IJTools.makeSimpleHistogram1k8(tools, width: 256, height: 150);
    IJTools.makeHistogram1k8(tools, width: 256, height: 150);

}
```

1 usage Jan Andre Freirich *

```java
public static void makeSimpleHistogram1k8(ImageTools tools, int width, int height) {
    ImageProcessor ip = tools.getImageProcessor();
    int width_ip = ip.getWidth();
    int height_ip = ip.getHeight();

    double sq = width_ip*height_ip;

    int[] histogram;
    if (width < 256) {
        histogram = ip.getHistogram(width);
    } else {
        histogram = ip.getHistogram( nBins: 256);
    }

    ImageTools hist_it = new ImageTools().withNewImage(width, height, title: "Histogram", ImageTools.WHITE);
    ImageProcessor hist_ip = hist_it.getImageProcessor();

    double sum = 0;
```

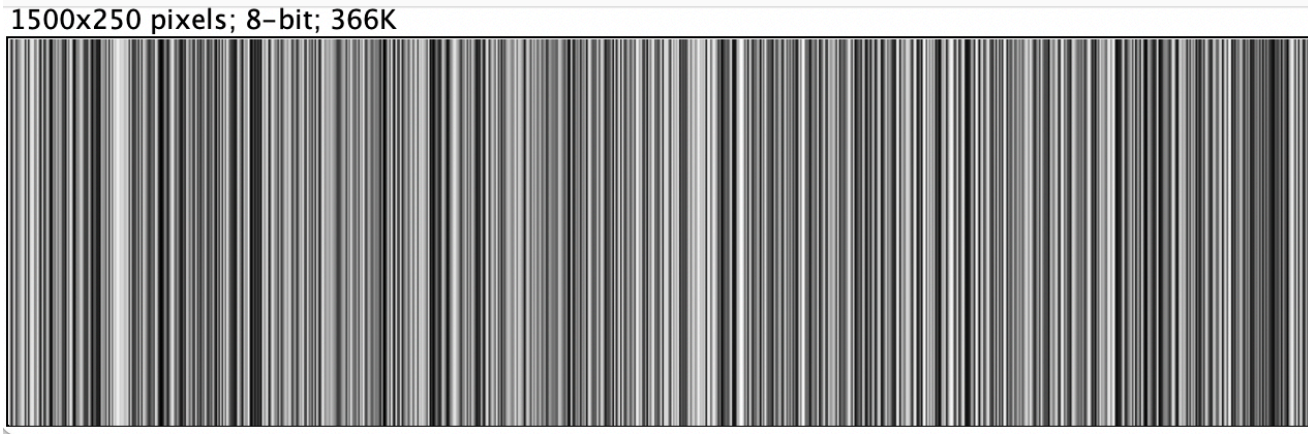
```
    for (int i = 0; i < histogram.length; i++) {

        double h = (histogram[i]/sq)*height;
        sum += h;
        for (int j = 0; j < sum && j < height; j++) {
            hist_ip.set(i, i1: (height-1)-j, ImageTools.getPixelOfRGB(new int[]{0, 255, 0}));
        }
    }

    hist_it.showImage();
}
```

1 usage Jan Andre Freirich *

```
public static void makeHistogram1k8(ImageTools tools, int width, int height) {

    ImageProcessor ip = tools.getImageProcessor();
    int width_ip = ip.getWidth();
    int height_ip = ip.getHeight();

    double max = Integer.MIN_VALUE, min = Integer.MAX_VALUE;
    int[] hist = new int[256];
    for (int i = 0; i < height; i++) {
        for (int j = 0; j < width; j++) {
            int val = ip.get(j, i);
            hist[val]++;
            if (max < hist[val]) {max = hist[val];}
            if (min > hist[val]) {min = hist[val];}
        }
    }

    ImageTools itHist = new ImageTools().withNewImage(width, height, title: "Histogram", ImageTools.WHITE);
    ImageProcessor ipHist = itHist.getImageProcessor();

    for (int i = 0; i < hist.length; i++) {
        int h = (int)(height*(hist[i]/max));
        for (int j = 0; j < h; j++) {
            ipHist.set(i, i1: (height-1)-j, ImageTools.getPixelOfRGB(new int[]{0, 255, 0}));
        }
    }
    itHist.showImage();
}
```

Aufg. 3.4. Erstellen Sie ein Plugin, das ein zufälliges
Bereich [0, 255] erzeugt. Verwenden Sie dazu die Ja
überprüfen Sie mittels des Histogramms, wie weit d

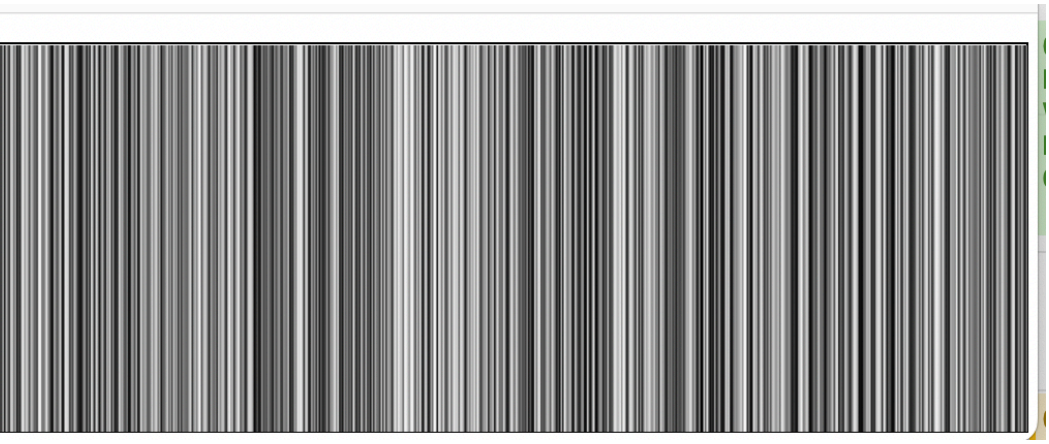
```
    @Override
    public void run(String s) {

        IJTools.createImageGleichverteilt
    }
}
```

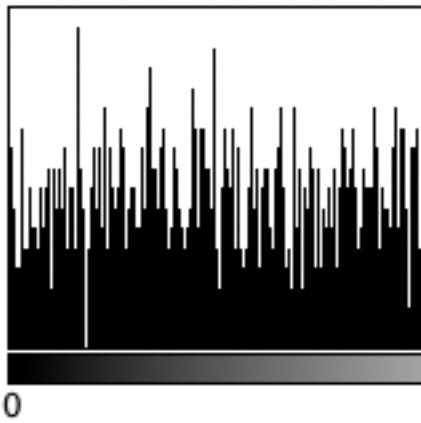


Aufg. 4.2. Ändern Sie das Plugin für den Histogram
(`Equalize_Histogram.java`) in der Form, dass e
verwendet.

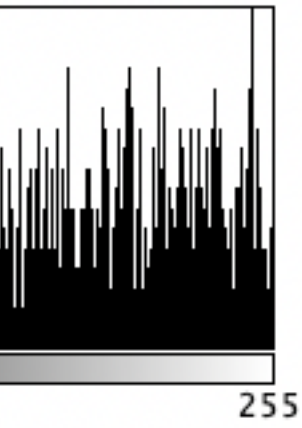
s Bild mit gleichverteilten Pixelwerten im
va-Methode `Math.random()` und
ie Pixelwerte tatsächlich gleichverteilt sind.

```
t( width: 2000, height: 100);
```





nmausgleich in Prog. 4.2
es eine Lookup-Tabelle für die Berechnung


255

```java
public void run(ImageProcessor ip) {
  int M = ip.getWidth();
  int N = ip.getHeight();
  int K = 256; // number of intensity values

  // compute the cumulative histogram:
  int[] H = ip.getHistogram();
  for (int j = 1; j < H.length; j++) {
    H[j] = H[j - 1] + H[j];
  }

  // equalize the image:
  for (int v = 0; v < N; v++) {
    for (int u = 0; u < M; u++) {
      int a = ip.get(u, v);
      int b = H[a] * (K - 1) / (M * N); // s.
      ip.set(u, v, b);
    }
  }
}
```

1 usage Jan Andre Freirich *

```java
public static void makeSimpleHistogram1k8(ImageTools tools, int widt
    ImageProcessor ip = tools.getImageProcessor();
    int width_ip = ip.getWidth();
    int height_ip = ip.getHeight();

    double sq = width_ip*height_ip;

    int[] histogram;
    if (width < 256) {histogram = ip.getHistogram(width);
```

Programm 4.2
Histogrammausgleich (`Equalize_Histogram`). Zunächst wird (in Zeile 7) mit der `ImageProcessor`-Methode `ip.getHistogram()` das Histogramm des Bilds `ip` berechnet. Das kumulative Histogramm wird innerhalb desselben Arrays („in place“) berechnet, basierend auf der rekursiven Definition in Gl. 3.8 (Zeile 9). Die in Gl. 4.11 vorgesehene *floor*-Operation erfolgt implizit durch die `int`-Division in Zeile 16.

Gleichung 4.11

```
th, int height) {
```