Pixel Clustering

Juan Salamanca

23 de marzo de 2016

This example presents the use of color clustering for the detection of preassure contact of several handpalms. The source image is the overlay of ten black and white hand palm prints. Each pixel of the composited resulting image is the aggregation of the grey value of each of the corresponding pixels (i.e. the pixels at the same coordinates). Before the aggregation all the images oppacity reduced to 10% in order to achieve 100% oppacity at the resulting image.

Step 1. Loading libraries

```
# Required libraries
# Adapted from http://www.r-bloggers.com/r-k-means-clustering-on-an-image/
library(ggplot2)
library(png)
library(RColorBrewer)
```

Step 2. Loading source image

```
# a variable to store the source image
fileName <- "hand_linear_grey_small.png"
# convert the image to a ratser array
importedImage <- readPNG(fileName)
# Obtain te number of dimensions of the dataframe
dimensions <- dim(importedImage)
str(dimensions)
```

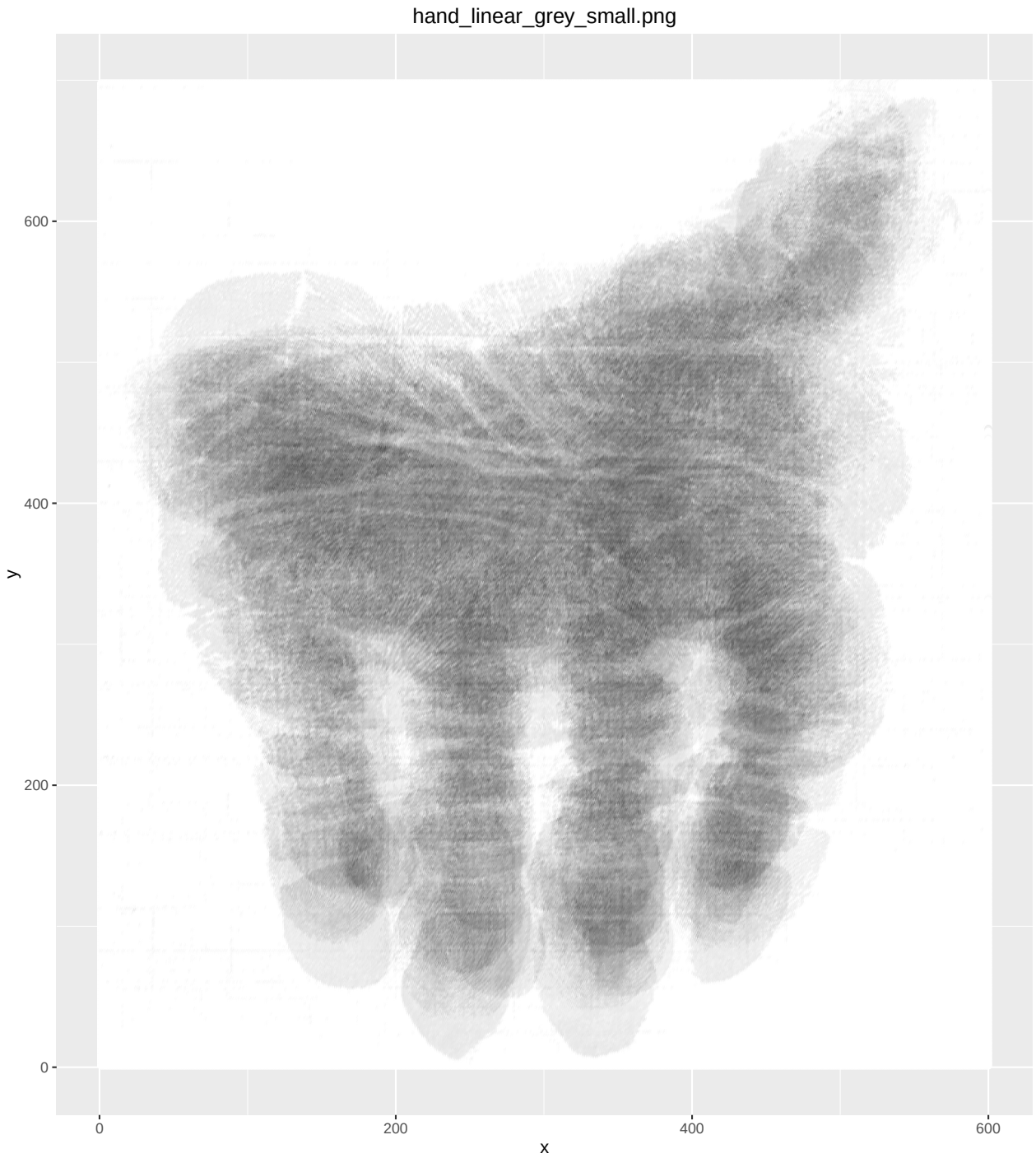
```
##  int [1:2] 698 600
```

Step 3. Convert matrix with color values into data frame

```
# Assign coordinates and RGB channels as data frame
originalImage <- data.frame(
  x = rep(1:dimensions[2], each = dimensions[1]),
  y = rep(dimensions[1]:1, dimensions[2]),
  value = as.vector(importedImage)
)
```

Step 4. Recreate original image with ggplot

```
# Build the original color image
image_ggplot <- ggplot(data = originalImage, aes(x = x, y = y))
image_ggplot <- image_ggplot + geom_point(colour = grey(originalImage$value))
image_ggplot <- image_ggplot +  labs(title = as.character(fileName))
image_ggplot <- image_ggplot + xlab("x") + scale_size("identity")
image_ggplot <- image_ggplot +ylab("y")

# Plot image
image_ggplot
```



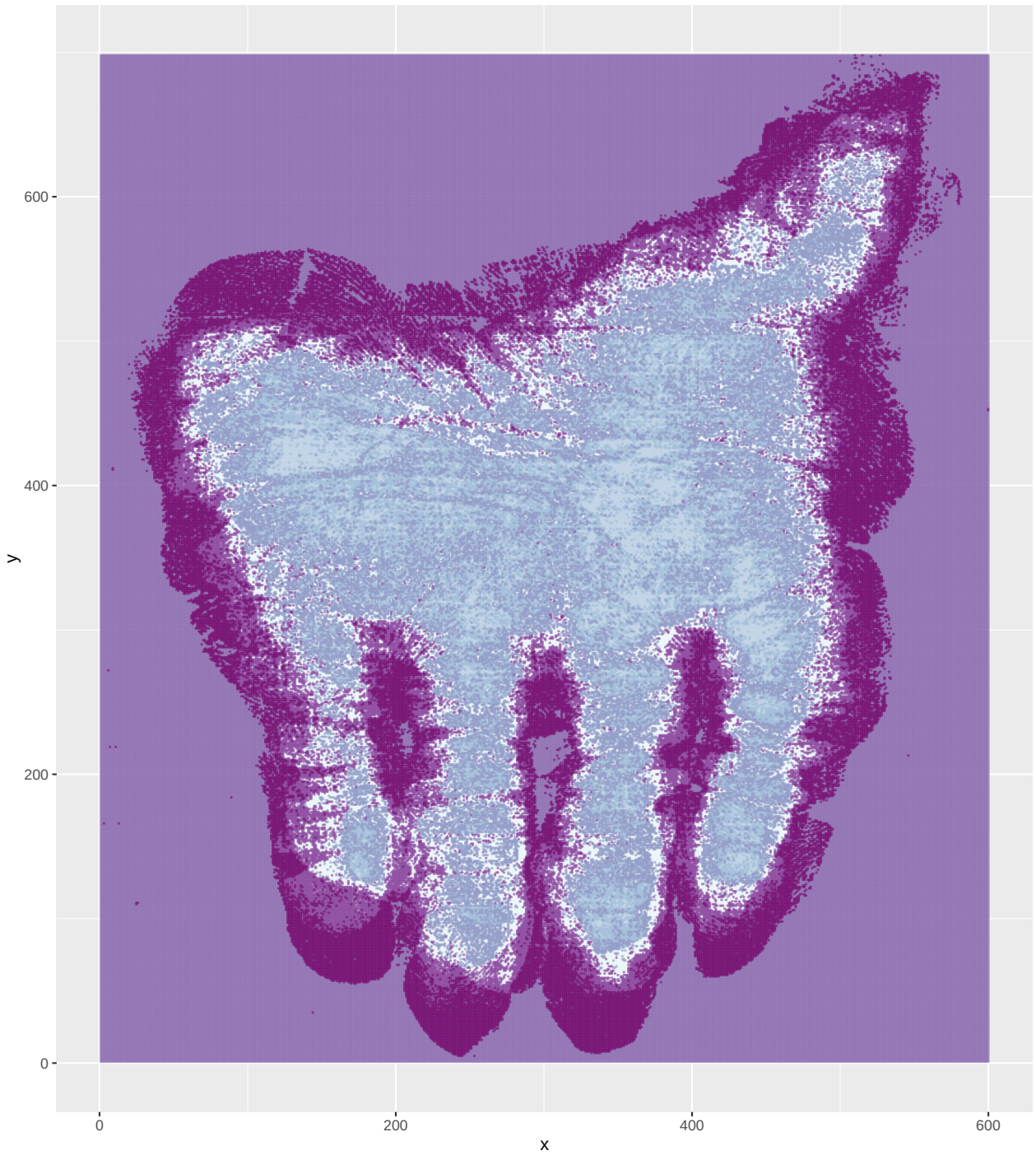

```
# originalImage
#png("originalImage.png")
#print(image_ggplot)
```

Step 5. Clustering pixels

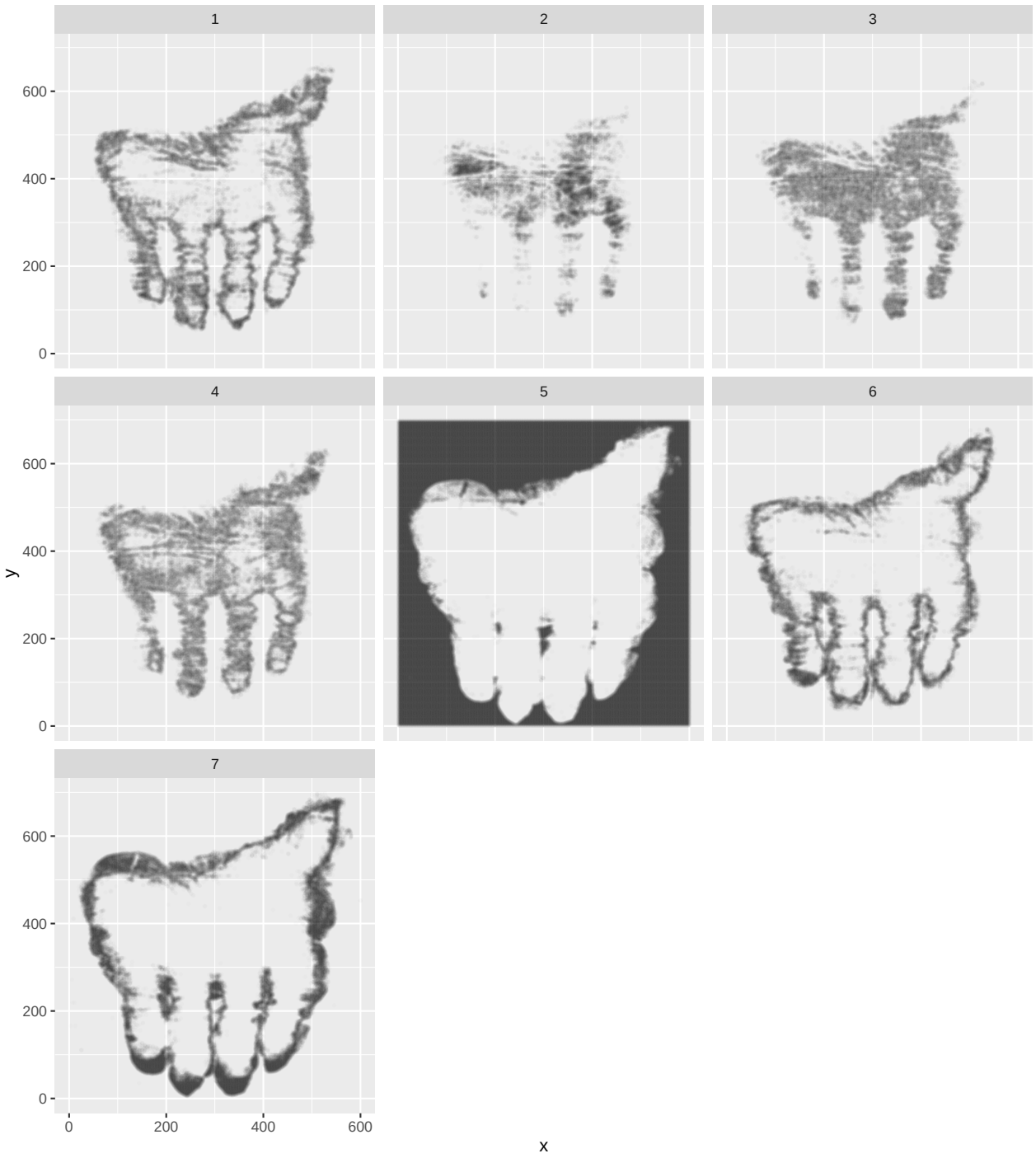
```
# Number of clusters
kClusters <- 7

# Clustering
kMeans <- kmeans(originalImage[,c("value")], centers = kClusters)

# bind cluster data
originalImage$clusters <-  kMeans$cluster

# Remove Backgorund data
finalImg <- originalImage

# Plot the resulting image
pal <- brewer.pal(kClusters,"BuPu")
clusteredImage <- ggplot (finalImg, aes(x,y, colour= as.factor(finalImg$clusters)))
clusteredImage <- clusteredImage + geom_point(data = subset(finalImg,finalImg$clusters==1), color = pal
clusteredImage <- clusteredImage + geom_point(data = subset(finalImg,finalImg$clusters==2), color = pal
clusteredImage <- clusteredImage + geom_point(data = subset(finalImg,finalImg$clusters==3), color = pal
clusteredImage <- clusteredImage + geom_point(data = subset(finalImg,finalImg$clusters==4), color = pal
clusteredImage <- clusteredImage + geom_point(data = subset(finalImg,finalImg$clusters==5), color = pal
clusteredImage <- clusteredImage + geom_point(data = subset(finalImg,finalImg$clusters==6), color = pal
clusteredImage <- clusteredImage + geom_point(data = subset(finalImg,finalImg$clusters==7), color = pal


# facets
facetedImage <- ggplot (finalImg, aes(x,y), colour = cluster) + geom_point(size=0.5, alpha = 1/100)
facetedImage <- facetedImage + facet_wrap( ~ clusters)
```

Step 6. Display images

```
clusteredImage
```



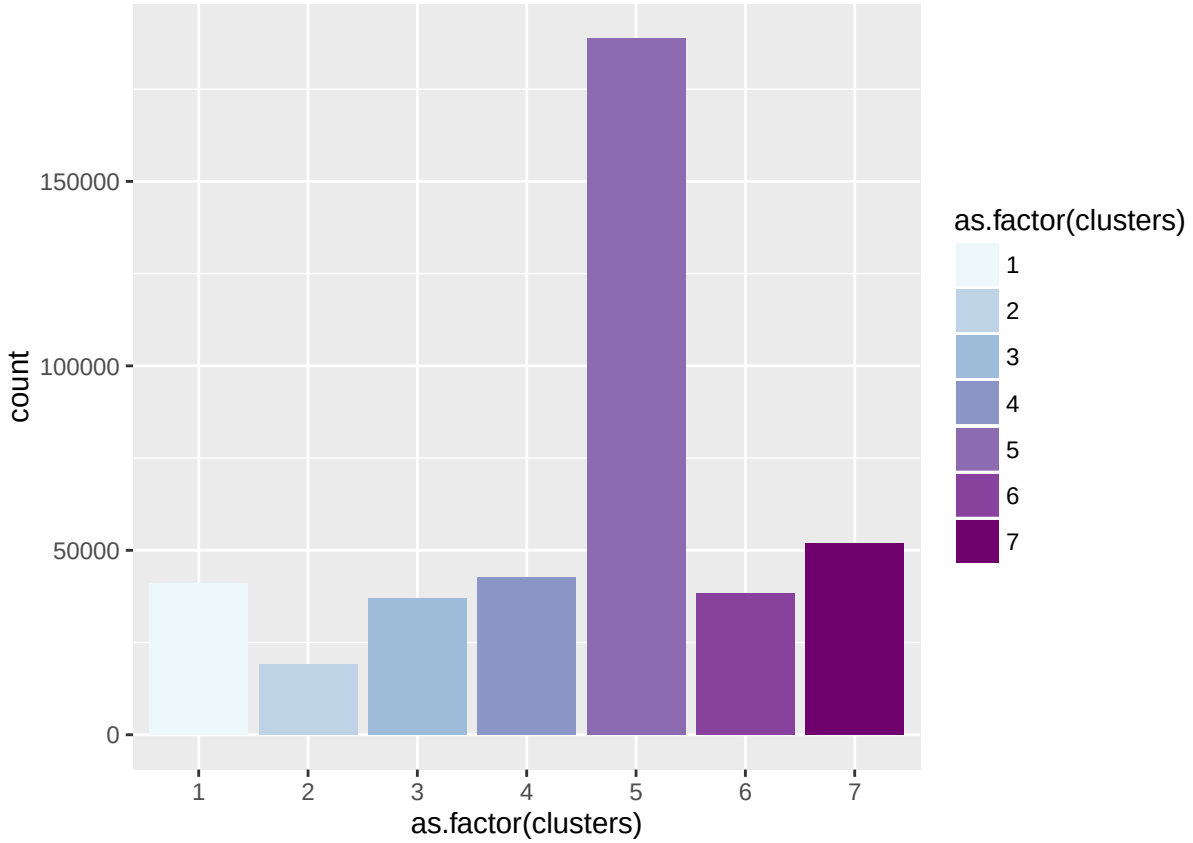

```
#png("clusteredImage.png")
#print(clusteredImage)

 facetedImage
```




```
#png("facetedImage.png")
#print(facetedImage)
```

Step 6. Cluster distribution

```
cDist <- ggplot(finalImg,aes(as.factor(clusters), fill=as.factor(clusters)))
cDist <- cDist + geom_bar()
cDist <- cDist + scale_fill_brewer(palette= "BuPu")
cDist
```



Practice

Take the file **grey_palette.png** and process it experimenting with several numbers of clusters. Determine the minimal number of clusters that better differentiate the gradient levels.