embodied Package

Tyler W. Rinker

January 30, 2014

The **embodied** package (Rinker, 2014) is designed work with video data. **embodied** is named for the movement within education that understands human learning to be the intersection of body and brain. This package interfaces with the FFmpeg and ImageMagick to convert videos to still images for manual coding.

embodied was inspired by Roy's (2011) Ted Talk in which he used video data to demonstrate the intersection of dialogue and movement within space. Later, Xie's (2013) package began to show that animated visualizations were within the grasp of the R user (see this blog post). The following documentation will demonstrate some of the uses of the **embodied** package.

1 Major Functions

In essence the user is likely to want to:

1. Convert a video to multiple still images
2. Add grid line to the images
3. Record the location(s) of participants over time
4. Read the data back into R
5. Analyze and plot the results

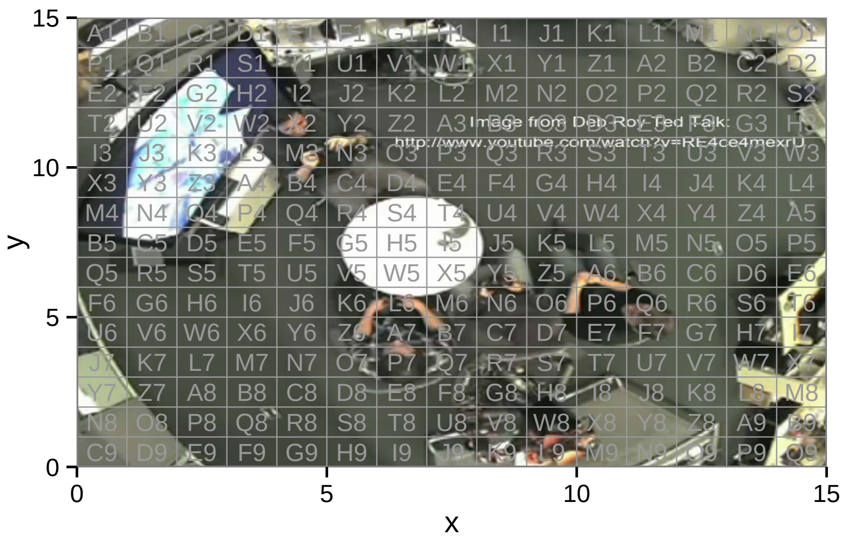
1.1 gridfy

Use of the **gridify** function can accomplish The first 2 steps. Ideally, the suer will supply a .mp4 file to the function, though **gridify** also takes a directory of .png files as well. The function utilizes **ggplot2** (Wickham, 2009) to output a single, scrollable .pdf or a directory of .png image files with grids and labels that the user can then use to record the movement (step 3) of participants across space and time. These motions are recorded in the code sheet .csv that is printed out.

The following code chunk demonstrates the usage:

```r
library(embodied)
deb <- system.file("extdata", package = "embodied")
gridify(deb, "out")
```

Which in turn produces images with grid lines as follows:



1.2 Code Sheet

A code sheet is also printed out in which he user can input the label locations of the participants across time.

id	time	person_1	person_2	person_3
img-1	00:01.00	***h5***	***p7***	***h6***
img-2	00:02.00	***i5***		
img-3	00:03.00	***i5***		
img-4	00:04.00			
img-5	00:05.00			
.	.			
.	.			
.	.			
img-n	01:15.75			

1.3 Reading in Code Sheet

Step 4 can be addressed with the **read_embodied** function. The data will be read into R and converted to long format. The user should specify the number of `columns` and `rows` if a value other than `30` was used in `gridify`. The user can utilize the following commands to read in the code sheet:

```r
dat <- read_embodied("foo.csv", columns = 30, rows = columns)
```

2 Motion Paths

The following short script allows us to generate a motion path animated .mp4 and .gif. I have chosen to access the tools of FFmpeg and ImageMagick directly via the `command` function; however, one could easily use the functions from Yiui Xie's (2013) **animation** package.

2.1 Check ffmpeg/convert Access

First you may want to test if [ffmpeg](http://www.ffmpeg.org/ffmpeg.html) and [ImageMagick](http://www.imagemagick.org/script/index.php) are available and on your path:

```r
library(embodied)
command("ffmpeg --version")
command("convert --version")
```

If you have command line access you will see something like:

```
> command("ffmpeg --version")
ffmpeg version N-59815-gb79bccb Copyright (c) 2000-2014 the FFmpeg developers
  built on Jan 13 2014 22:01:47 with gcc 4.8.2 (GCC)
  configuration: --enable-gpl --enable-version3 --disable-w32threads --enable-avisynth --enable-bzli
  libavutil      52. 62.100 / 52. 62.100
  libavcodec     55. 48.101 / 55. 48.101
  libavformat    55. 23.103 / 55. 23.103
  libavdevice    55.  5.102 / 55.  5.102
  libavfilter     4.  1.100 /  4.  1.100
  libswscale      2.  5.101 /  2.  5.101
  libswresample   0. 17.104 /  0. 17.104
  libpostproc    52.  3.100 / 52.  3.100

> command("convert --version")
Version: ImageMagick 6.8.8-2 Q16 x64 2014-01-09 http://www.imagemagick.org
Copyright: Copyright (C) 1999-2014 ImageMagick Studio LLC
Features: DPC Modules OpenMP
Delegates: bzlib cairo freetype jbig jng jp2 jpeg lcms lqr pangocairo png ps rsvg tiff webp xml zlib
```

2.2 Steps to Plot Animated Paths

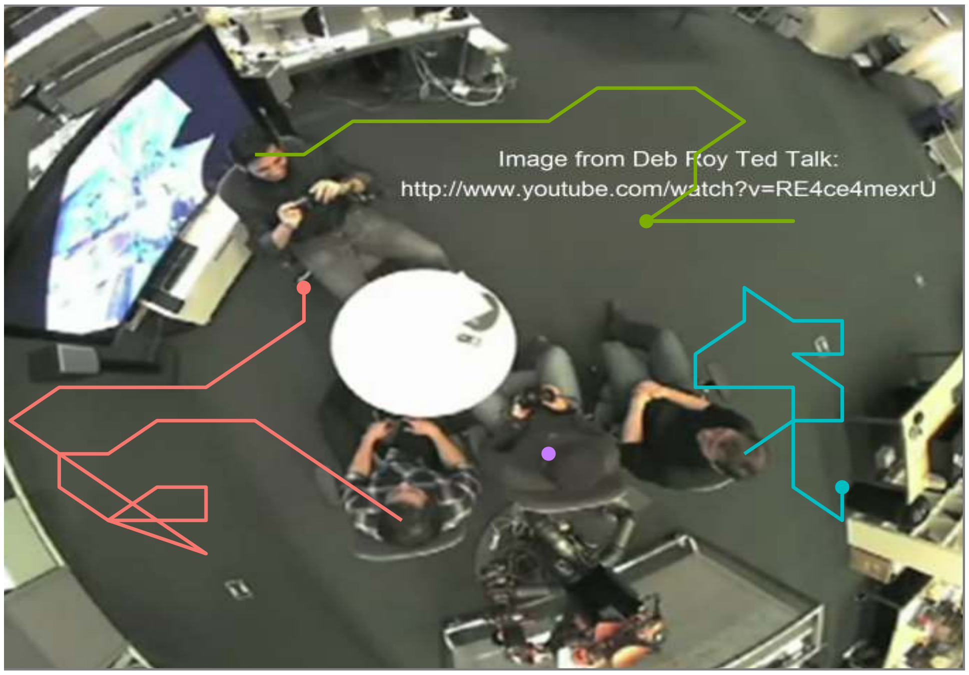
We can now:

1. Read in the image(s) (set row and column numbers)
2. Create a function (pp) that plots the poth (animates the sequence)
3. Plot multiple .pngs files showing the paths
4. Stitch the multiple .pngs into a .mp4 or animated .gif.

2.3 Animated Path Script

```
library(embodied); library(grid); library(ggplot2)
dat <- grid_coord(deb_long, 20)
file <- system.file("extdata/deb_roy.png", package = "embodied")
base <- read_png(file, 20)
dir.create("out")
```

```r
ids <- as.character(unique(dat[, "id"]))
pp <- function() {
    for (i in 1:length(ids)) {
       png(sprintf("out/img-%s.png", pad(1:length(ids))[i]), width=450, height = 400)
        if (i == 1) {
            print(base +
                geom_point(
                    data = dat[dat[, "id"] %in% ids[1:i], ],
                    aes(group = person,
                        colour=person),
                    size = 2) +
                theme(legend.position="bottom",
                    plot.margin=unit(c(0,0,1,-1), "cm"),
                    legend.title=element_blank(),
                    axis.title.x = element_blank()))
        } else {
            print(base +
                geom_point(data = dat[dat[, "id"] %in% ids[i], ],
                    aes(group = person,
                        colour=person),
                    size = 2) +
                geom_path(data = dat[dat[, "id"] %in% ids[1:i], ],
                    aes(group = person, colour=person)) +
                theme(legend.position="bottom",
                    plot.margin=unit(c(0,0,1,-1), "cm"),
                    legend.title=element_blank(),
                    axis.title.x = element_blank()))
        }
       dev.off()
    }
}
pp()
command("ffmpeg -r 5 -i out/img-%02d.png -c:v libx264 -r 30 -pix_fmt yuv420p out/out.mp4")
command("convert -delay 10 -loop 0 out/*.png out/out.gif")
```



Brandon
Deb
Philip
Rony

3 Dialogue Density Plots

The following short script allows us to generate a series dialogue density plot using the **ggplot2** (Wickham, 2009) and the **embodied** (Rinker, 2014) packages.

3.1 Reading in Image and Preparing Data

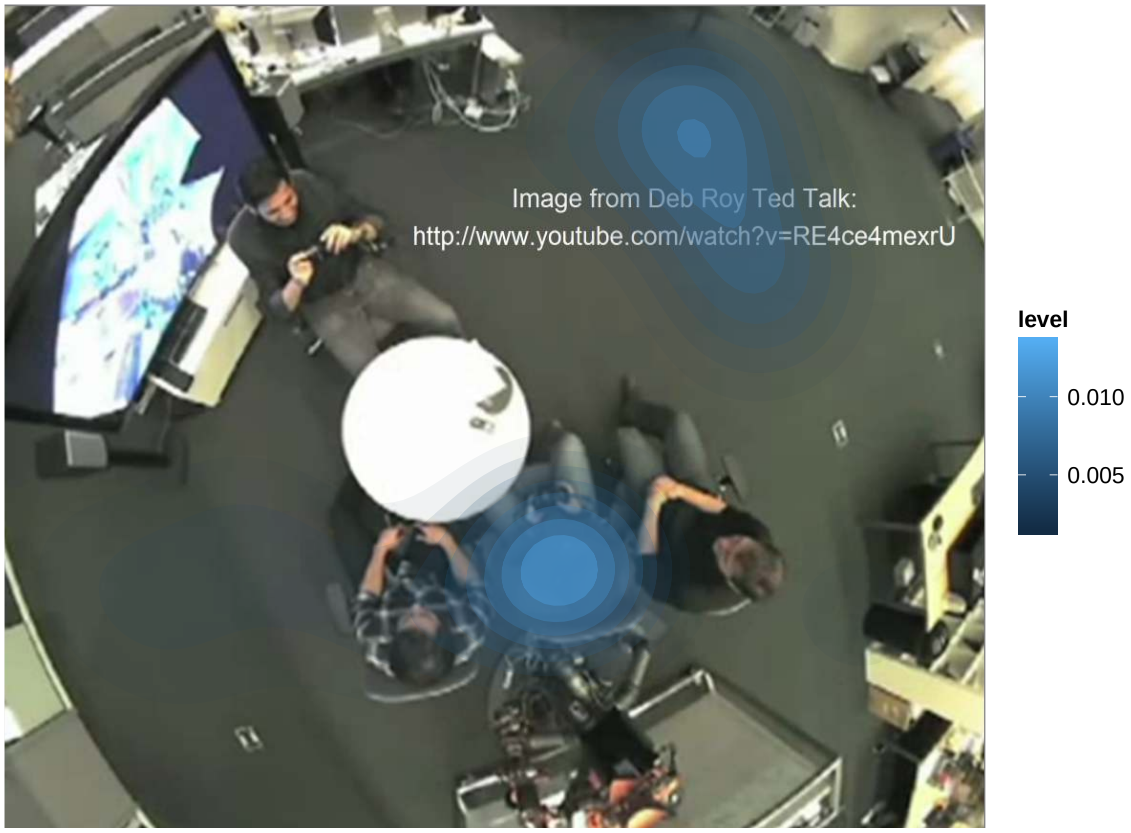
```
library(embodied); library(ggplot2); library(grid)
file <- system.file("extdata/deb_roy.png", package = "embodied")
base <- read_png(file, columns=20)
```

```r
dat <- deb_complete[rep(1:nrow(deb_complete), deb_complete$wc), ]
head(dat)
```

```
    id       time person coord                dialogue wc   x    y
4   04 00:11:22.0    Deb    i4     should be colliers.  3 6.5 15.5
4.1 04 00:11:22.0    Deb    i4     should be colliers.  3 6.5 15.5
4.2 04 00:11:22.0    Deb    i4     should be colliers.  3 6.5 15.5
9   09 00:11:24.5    Deb    r3 I strike quickly, being  4 9.5 16.5
9.1 09 00:11:24.5    Deb    r3 I strike quickly, being  4 9.5 16.5
9.2 09 00:11:24.5    Deb    r3 I strike quickly, being  4 9.5 16.5
```

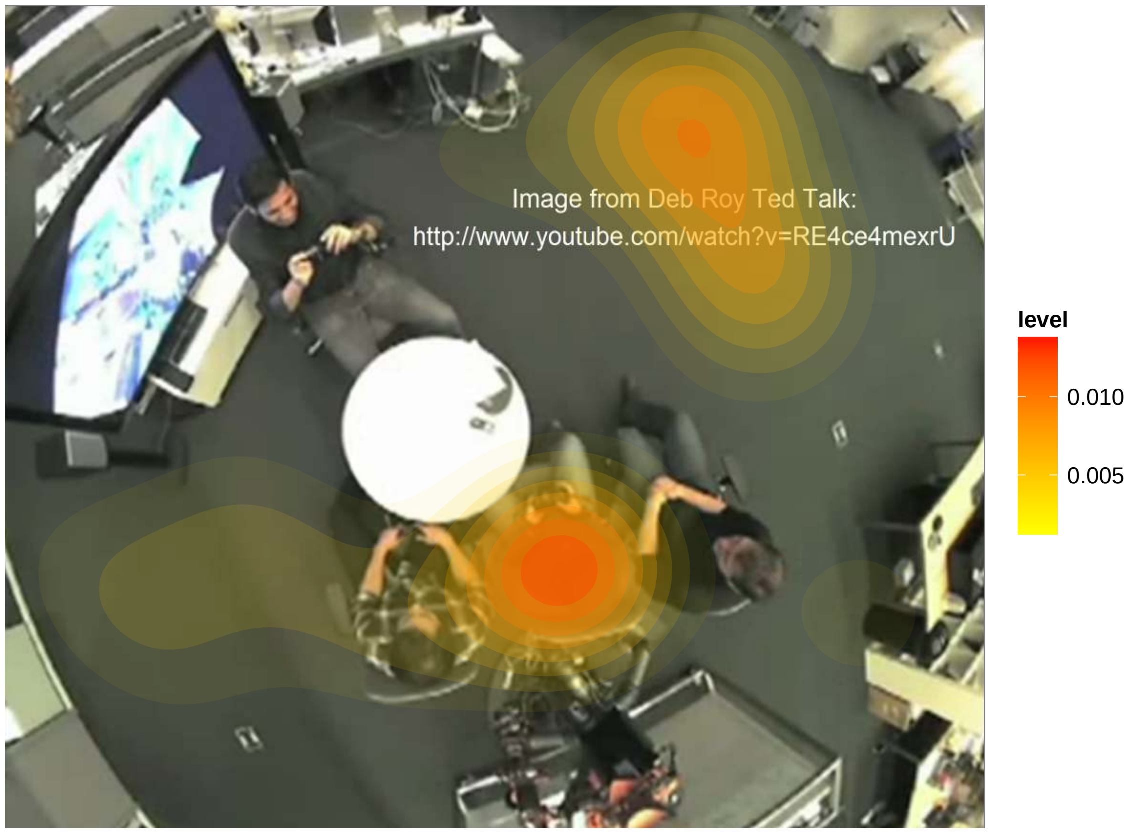
3.2 Basic Design

```
base +
    stat_density2d(data = dat,
        aes(x=x, y=y, alpha=..level.., fill=..level..), size=2, bins=10,
        geom="polygon") +
    scale_alpha(range = c(0.00, 0.5), guide = FALSE)
```



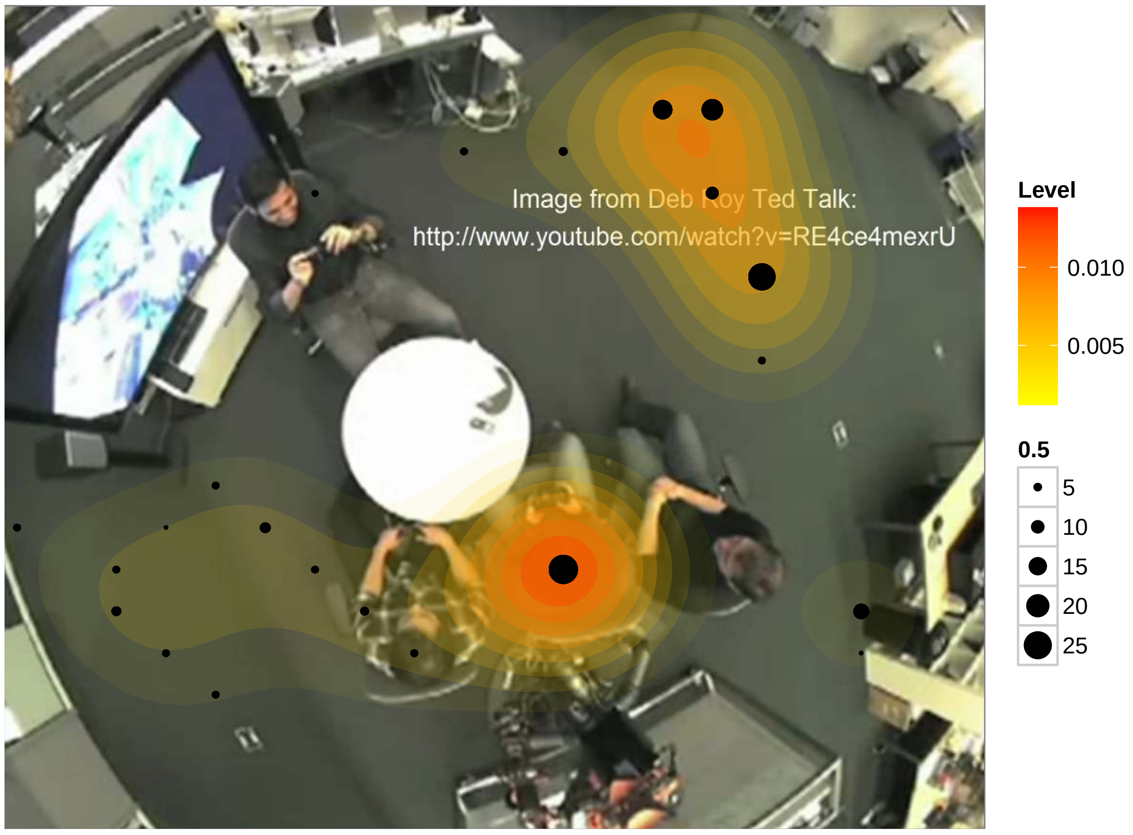
3.3 Change Density Colour

```
base +
    stat_density2d(data = dat,
        aes(x=x, y=y, alpha=..level.., fill=..level..), size=2, bins=10,
        geom="polygon") +
    scale_fill_gradient(low = "yellow", high = "red") +
    scale_alpha(range = c(0.00, 0.5), guide = FALSE)
```



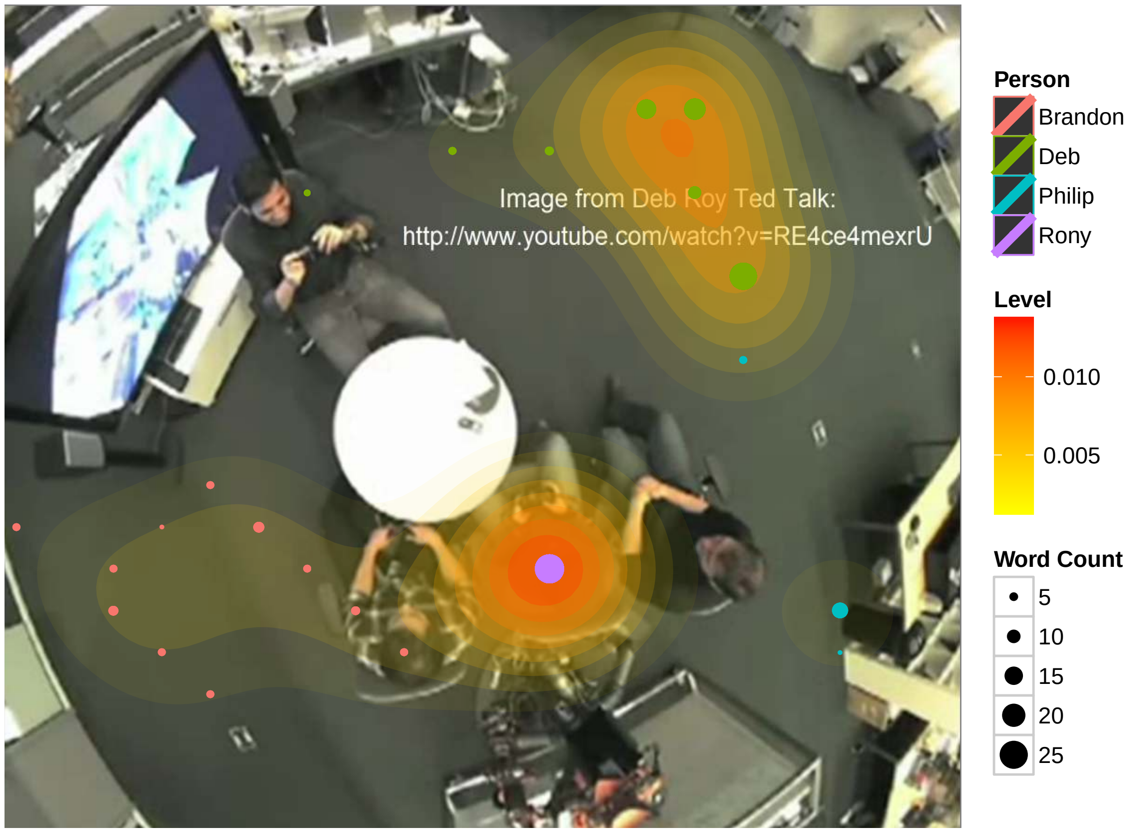
3.4 Adding Black, Sized Word Count Points

```
base +
    stat_density2d(data = dat,
        aes(x=x, y=y, alpha=..level.., fill=..level..), size=2, bins=10,
        geom="polygon") +
    scale_fill_gradient(low = "yellow", high = "red", name = "Level") +
    scale_alpha(range = c(0.00, 0.5), guide = FALSE) +
    geom_point(data = dat, aes(size = wc, x=x, y=y), colour="black")
```



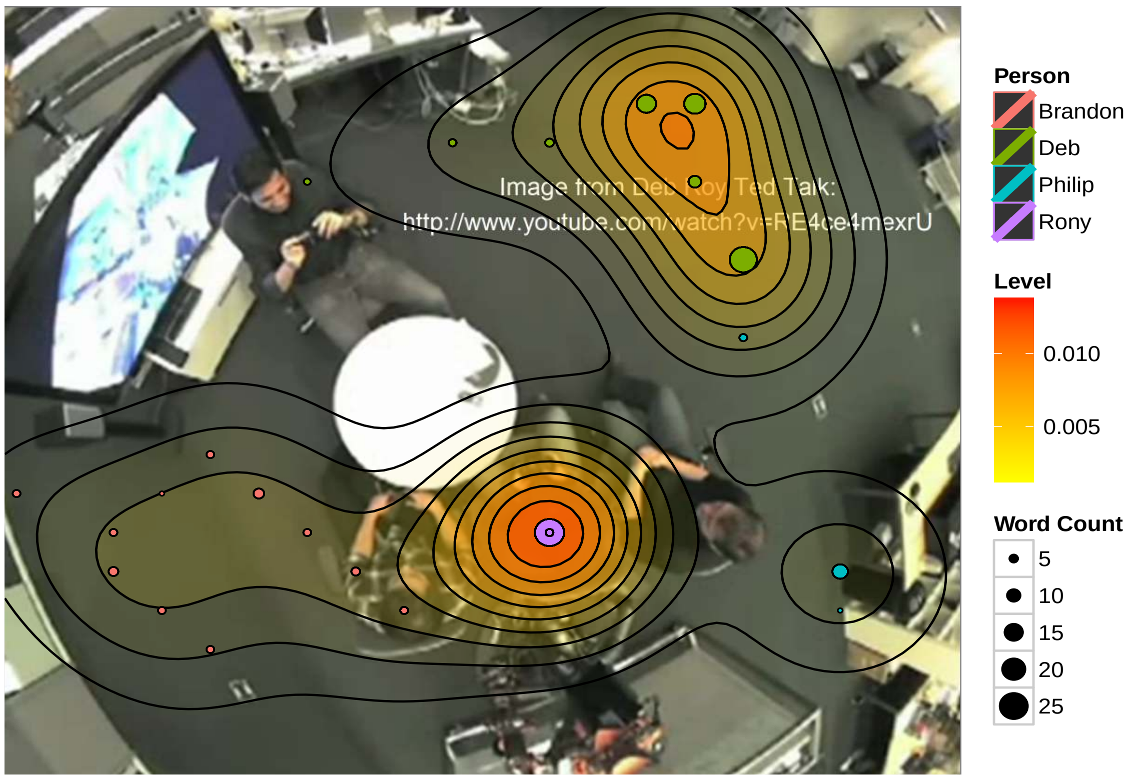
3.5 Color Word Count Points by Person

```
base +
    stat_density2d(data = dat,
        aes(x=x, y=y, alpha=..level.., fill=..level..), size=2, bins=10,
        geom="polygon") +
    scale_fill_gradient(low = "yellow", high = "red", name = "Level") +
    scale_alpha(range = c(0.00, 0.5), guide = FALSE) +
    geom_point(data = dat, aes(size = wc, colour = person, x=x, y=y)) +
    guides(alpha=FALSE,
        colour=guide_legend(title="Person"),
        size=guide_legend(title="Word Count"))
```



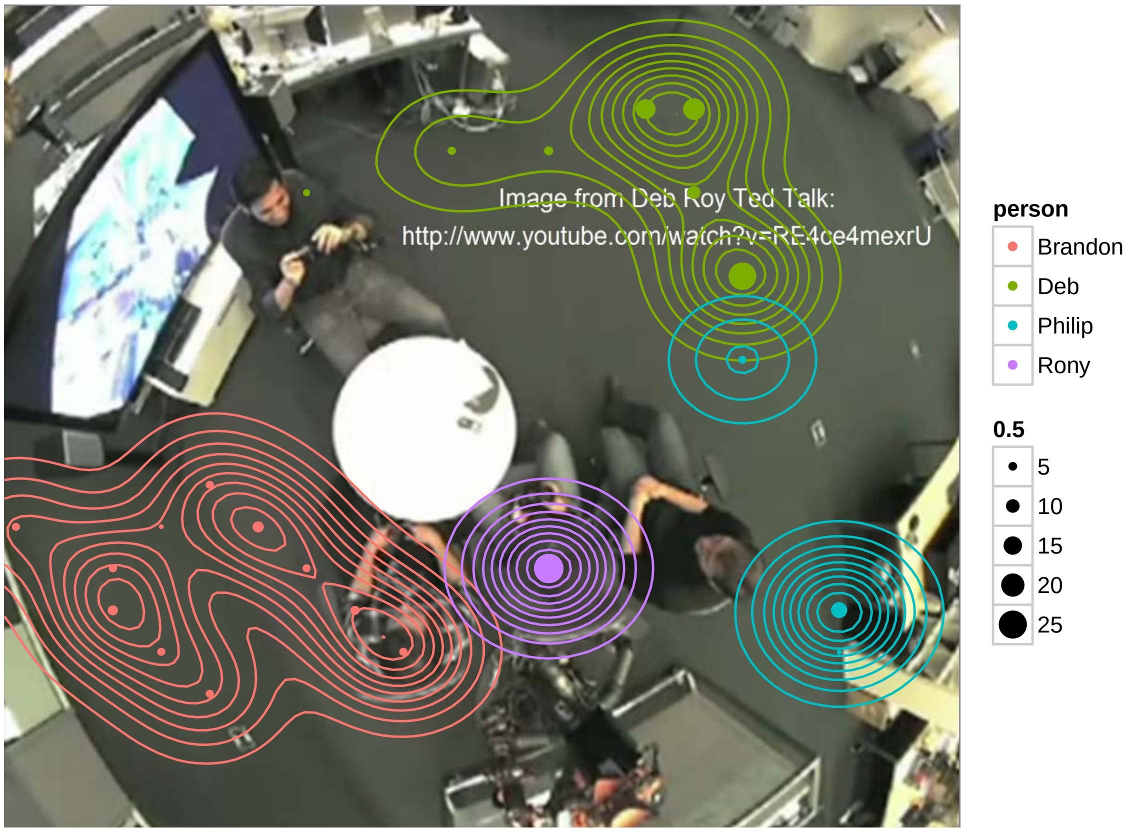
3.6 Add Black Desnity Curves and Point Outlines

```
base +
    stat_density2d(data = dat,
        aes(x=x, y=y, alpha=..level.., fill=..level..), size=2, bins=10,
        geom="polygon") +
    scale_fill_gradient(low = "yellow", high = "red", name = "Level") +
    scale_alpha(range = c(0.00, 0.5), guide = FALSE) +
    geom_density2d(data = dat, aes(x=x, y=y), colour="black", bins=10,
        show_guide=FALSE) +
    geom_point(data = dat, aes(size = wc, colour = person, x=x, y=y)) +
    geom_point(data = dat, aes(size = wc, colour = person, x=x, y=y),
        shape = 1, colour = "black", guide=FALSE) +
    guides(alpha=FALSE,
        colour=guide_legend(title="Person"),
        size=guide_legend(title="Word Count"))
```



3.7 Density Lines Colored by Person

```
base +
    geom_density2d(data = dat, aes(x=x, y=y, color=person),
        bins=10, h=4, show_guide=FALSE) +
    geom_point(data = dat, aes(size = wc, colour = person, x=x, y=y))
```



References

Rinker TW (2014). *embodied: Ebodiement related analysis tools.* University at Buffalo/SUNY, Buffalo, New York. URL `http://github.com/trinker/embodied`.

Roy D (2011). "Deb Roy: The birth of a word." URL `http://www.ted.com/talks/deb_roy_the_birth_of_a_word.html`.

Wickham H (2009). *ggplot2: Elegant graphics for data analysis.* Springer New York. URL `http://had.co.nz/ggplot2/book`.

Xie Y (2013). *animation: A gallery of animations in statistics and utilities to create animations.* R package version 2.2, URL `http://CRAN.R-project.org/package=animation`.