HHI VR Interface User Test - Data Analysis - R Markdown

Alexander Masurovsky

7/14/2019

Contents


```
knitr::opts_chunk$set(tidy = TRUE, message = FALSE, warning = FALSE, echo=TRUE)
```

Initialize

Initial setup code: directories, libraries, etc.

```
rm(list = ls())  # clear variables
cat("\f")  # clear console
```

```r
# load libraries
library(readr)
library(readxl)
library(tidyverse)
library(dplyr)
library(psych)
library(ggplot2)
library(see)  #geomviolindot
library(RColorBrewer)
library(cowplot)
library(lsr)
library(ggpubr)
library(car)
library(rstatix)
library(tinytex)
library(xtable)
library(stargazer)
library("TOSTER")
source("R_rainclouds.R")
# source('smart_t_test.R')

# set analysis directory (current working directory)
analysis_dir <- "~/R/Projects/HHI_Leap_User_Test/"

# set unity data directory
unity_data_dir <- "~/R/Projects/HHI_Leap_User_Test/Unity_Data"

# set coagulated survey data file name
survey_data_filename <- "survey_results.xlsx"

# Note: drop count data files stored in Drop_Counting directory

# select subjects to remove for analysis (99 means remove none)
remove_these_subjects <- 99  # c(12, 30)
```

Global aesthetics

```r
# theme
theme_set(theme_cowplot())

# plot order
plot_order <- c("B_Leap", "HHI_Leap", "Oculus")

# colors
mycolors = c("darkgrey", "blue", "darkorange")  # colorblind safe (I think)
# mycolors=c('darkorange','forestgreen','blue')
# mycolors=c('grey','forestgreen','black') mycolors=c('mediumseagreen', 'pink3',
# 'black') mycolors=c('mediumseagreen', 'thistle3', 'black')
mysmoothing = 1.5
myalpha = 0.3
```

```
# text sizes
title_size <- 30
axis_text_size <- 30
legend_title_size <- 10
legend_text_size <- 10
legend_pos <- "none"

# significance symbols
mysymbols = c("*** ", "***", "**", "*", "")
mysymbols2 = c("*** ", "***", "**", "*", "p > .05")
```

Pre-processing

Survey data

Load in and organize

1. Save as .xlsx (may need to open in Google Docs first)
2. Clean manually **before running this script** using find-and-replace – numbers initially contain additional text that needs to be eliminated
3. Name file this: survey_results.xlsx The following code will load in the survey data after the above steps have been completed manually. The data set will be called **survey_data**.

```
# Get names: read in old questionnaire for variable names
library(readxl)
survey_data <- read_excel(survey_data_filename, na="")

# clean headers and set up variables
# rename sentence for User Id column to just UserID
survey_data <- survey_data %>% rename(UserID = contains("user"))
# fix variable names by removing everything after the "."
names(survey_data) <- gsub("\\..*","",names(survey_data))

#survey_data$UserID <- factor(survey_data$UserID) # make UserID column into factors
survey_data <- survey_data %>% arrange(UserID) #order by user id

# rename columns
survey_data <- survey_data %>%
  rename(UserID=contains("UserID"),
         Hand=contains("starkeHand"),
         Gender=Sex,
         Arm=armlength,
         Disabilities=contains("Disabilities"),
         SkillController=`Skills[SkillController]`,
         SkillVR=`Skills[SkillVR]`,
         SkillGames=`Skills[SkillGames]`) %>%
  mutate(PrefCondition=factor(PrefCondition),
         Gender=factor(Gender, labels=c("Male","N/A", "Female")),
         UserID=factor(UserID),
         Hand=factor(Hand, labels=c("Left","Right")),
         `OculusQuestion[q04]`=6-`OculusQuestion[q04]`, # reverse score these questions
```

```
        `OculusQuestion[q05]`=6-`OculusQuestion[q06]`,
        `OculusQuestion[q06]`=6-`OculusQuestion[q06]`,
        `StandardLeapQuestion[qL4]`=6-`StandardLeapQuestion[qL4]`,
        `StandardLeapQuestion[qL5]`=6-`StandardLeapQuestion[qL5]`,
        `StandardLeapQuestion[qL6]`=6-`StandardLeapQuestion[qL6]`,
        `HHILeapQuestion[qH4]`=6-`HHILeapQuestion[qH4]`,
        `HHILeapQuestion[qH5]`=6-`HHILeapQuestion[qH5]`,
        `HHILeapQuestion[qH6]`=6-`HHILeapQuestion[qH6]`
         ) %>%
  mutate(PrefCondition=recode_factor(PrefCondition,
   "ohne Controller, Variante 1 (Standard Leap Motion)"="B_Leap",
            "mit Controller"="Oculus",
            "ohne Controller, Variante 2 (Leap Motion mit HHI-Anpassungen)"="HHI_Leap")) %>%
  arrange(UserID)



# create sample size variable for future use
sample_size = length(levels(survey_data$UserID)) # establish sample size
```

Score SUS

```
# Score the SUS (source: https://measuringu.com/sus/) For odd items: subtract one
# from the user response.  For even-numbered items: subtract the user responses
# from 5 This scales all values from 0 to 4 (with four being the most positive
# response).  Add up the converted responses for each user and multiply that
# total by 2.5. This converts the range of possible values from 0 to 100 instead
# of from 0 to 40.

# subset SUS questions by group; only take UserID and SUS data, recombine or
# compare/correlate with other data later
SUS_data <- select(survey_data, UserID, contains("SUS"))
SUS_data <- SUS_data[order(SUS_data$UserID), ]  # order by user id, in case there are missing user id n

# generate SUS score for 1) each user and 2) each Interface, then 3) put into df
# of user ID x (SUS score x Interface) update the following df w/ scores as the
# loop progresses

SUS_scores <- data.frame(UserID = SUS_data$UserID, Standard = 1, HHI = 1, Oculus = 1)
# DO NOT CHANGE GROUP NAMES!

groups <- c("Standard", "HHI", "Oculus")  # quick group names; correspond to Unity data labels

for (subj in 1:length(survey_data$UserID)) {
    # loop through all subjects loop through each condition/Interface
    for (grp in 1:3) {
        tmp_score <- 0  # set temp score, add to it
        tmp_subset <- SUS_data %>% filter(UserID == subj) %>% select(contains(groups[grp]))

        cell_val <- 0  # the subject's response which we will perform calculations upon (reset after ea
        # add up score in this next for loop, captured in tmp_score length of SUS
```

```r
        for (q_num in 1:length(tmp_subset)) {
            cell_val <- tmp_subset[[q_num]]
            if ((q_num%%2) == 0) {
                # check if even
                tmp_score <- tmp_score + (5 - cell_val)
            } else {
                # if odd (not even)
                tmp_score <- tmp_score + (cell_val - 1)
            }
        }
        SUS_scores[subj, groups[grp]] <- tmp_score * 2.5
    }
}

rm(SUS_data)  # for cleanliness
SUS_scores <- rename(SUS_scores, B_Leap = Standard, HHI_Leap = HHI)
```

Unity data

Setup

Combine individual Unity datasets into one

This will loop through every file in a directory and combine it into one excel file. This section of code is a dumb robot, so make sure that ONLY data files to be combined are in the directory specified below.

```r
# Data file directory (.csv files with Unity data from our user test only!):
setwd(unity_data_dir)

file_list <- list.files()

for (file in file_list) {
    # if the data frame variable does not exist, create it and read in first file
    if (!exists("dataset")) {
        dataset <- read_csv2(file, col_names = TRUE, locale = locale(decimal_mark = ","))
    }
    # if the merged dataset does exist, append to it
    if (exists("dataset")) {
        temp_dataset <- read_csv2(file, col_names = TRUE, locale = locale(decimal_mark = ","))
        dataset <- rbind(dataset, temp_dataset)
        rm(temp_dataset)
    }
}

# write to a new file
setwd("~/R/Projects/HHI_Leap_User_Test/")

# The code below prevents an already existing data set from being overwritten.
# Rename file_name to save new file, or delete old file manually.
file_name <- "collected_unity_data.csv"
if (file_name %in% list.files()) {
    print("file already exists!")
} else {
```

```
    write.csv(dataset, file_name)
}
```

```
## [1] "file already exists!"
```

```
rm(dataset)  # remove variable for cleanliness
```

Load Unity data set

The following code * loads the data set (**make sure to check that the filename and working directory are correct!**) * establishes that the id numbers are factors, not numbers (important for later code) * creates a vector of group names

```
# set working directory
setwd(analysis_dir)

# Load Unity dataset (confirm filename!) -- assumes American-style .csv file,
# which the collected data file should now be
my_file = "collected_unity_data.csv"  # confirm filename
data_set <- read.csv(my_file, numerals = "warn.loss")  #, 'no.loss'))

data_set <- data_set %>% select(everything(), -X) %>% rename(Cube_Size = ObjectName,
    Interface = Device, InterfaceOrder = DeviceOrder) %>% mutate(id = factor(id)) %>%
    mutate(Cube_Size = recode_factor(Cube_Size, `Cube10x10x10(Clone)` = "Large",
        `Cube6x6x6(Clone)` = "Medium", `Cube3x3x3(Clone)` = "Small"), Interface = recode_factor(Interfa
        Leap = "B_Leap", Leap_HHI = "HHI_Leap"))

group_names <- c("B_Leap", "HHI_Leap", "Oculus")  # will use this later

# remove duplicates (not sure why these exist)
trials_original_n <- nrow(data_set)
data_set <- data_set %>% distinct(.)
trials_duplicates_removed_n <- nrow(data_set)
```

Load in accidental drop data

An accidental drop is where the subject accidentally dropped the cube on the way to the target. These were initially noted manually by experimenter during the test. These are to be removed and counted as the Errors metric.

```
# load in all files from folder (make sure only data files in folder)
# add error (1 or 0) to trial number
# use SpawnOrder variable to match it up

# read in each data set and append to unity data set

# put name of directory w/ error files here
my_folder <- "Drop_Counting"
my_dir <- paste0(getwd(),"/",my_folder,"/")
setwd(my_dir)

file_list <- list.files()
```

```r
#file<-file_list[30]

for (file in file_list){
  temp_dataset <- read_excel(paste0(my_dir,file), skip=1, na="")
  temp_subject <- sub("\\.xlsx", "", file)
    temp_subject <- as.numeric(sub(".*S", "", temp_subject))

  # mutate temp dataset to match format of dataset (Interface="",id="", Drop=1 or 0)
  temp_dataset_long <- temp_dataset %>%
    slice(1:30) %>%
  select(Trial, Oculus, B_Leap='Leap', HHI_Leap='HHI Leap') %>%
    gather(key="Interface", value="Drop", 'Oculus':'HHI_Leap') %>%
    rename(SpawnOrder=Trial) %>%
    mutate(id=temp_subject,#id=as.character(temp_subject),
           SpawnOrder=as.numeric(SpawnOrder))
  temp_dataset_long$Drop[is.na(temp_dataset_long$Drop)] <- 0 #turn NA's into 0's
  temp_dataset_long$Drop <- as.numeric(temp_dataset_long$Drop) # drops are numbers

  # add temp_dataset_long to big error data set
  if(!exists("error_dataset")){
    error_dataset <- temp_dataset_long
  } else {
    error_dataset <- bind_rows(error_dataset, temp_dataset_long)
  }
}


error_dataset <- error_dataset %>%
  mutate(id=factor(id))

# join error data to unity data set (data_set)
data_set <- left_join(data_set, error_dataset, by=c("id","Interface","SpawnOrder"))


setwd("..") # move back up to original folder

# clean up
rm(error_dataset, temp_dataset, temp_dataset_long, file_list, my_folder, my_dir)
```

InterfaceOrder data

The following code takes Interface order from the Unity data set and adds it to the survey data set.

```r
# compile a data frame of user id and Interface order
Interface_order <- data.frame(id = c(1:sample_size))  #create df to fill w/ Interface order by id
for (x in 1:length(Interface_order$id)) {
    temp_subset <- subset(data_set, id == x)
    Interface_order[x, "InterfaceOrder"] <- temp_subset$InterfaceOrder[[1]]
}
rm(temp_subset)  # remove temp subset variable from list

# re-name Interface order to first letters only L=Standard Leap, H=HHILeap,
# O=Oculus
```

```r
levels(Interface_order$InterfaceOrder) <- gsub("LeapHHI", "H", levels(Interface_order$InterfaceOrder))
levels(Interface_order$InterfaceOrder) <- gsub("Leap", "L", levels(Interface_order$InterfaceOrder))
levels(Interface_order$InterfaceOrder) <- gsub("Oculus", "O", levels(Interface_order$InterfaceOrder))
levels(Interface_order$InterfaceOrder) <- gsub("-", "", levels(Interface_order$InterfaceOrder))

# add Interface order to survey_data TRY WITH DPLYR LATER
survey_data <- cbind(survey_data, InterfaceOrder = Interface_order$InterfaceOrder)
# rm(Interface_order) # remove Interface order dataset to keep tings clean

group_counts <- data.frame(table(Interface_order$InterfaceOrder))
# rename columns to 'Order' and 'Count':
group_counts <- group_counts %>% rename(InterfaceOrder = Var1, Count = Freq)
```

Explore Unity data

The goal is to clean the data so that the mean metrics calculated (accuracy and time measures) reflect normal use of the interfaces, not errors such as accidental drops or a system glitch.

The new strategy is to use only one cleaning data set, using various filters. Determine the filters **first**, then apply all at once.

This will improve our ability to keep track of what we did and prevent multiple data sets from floating around.

Visualize accidental drops

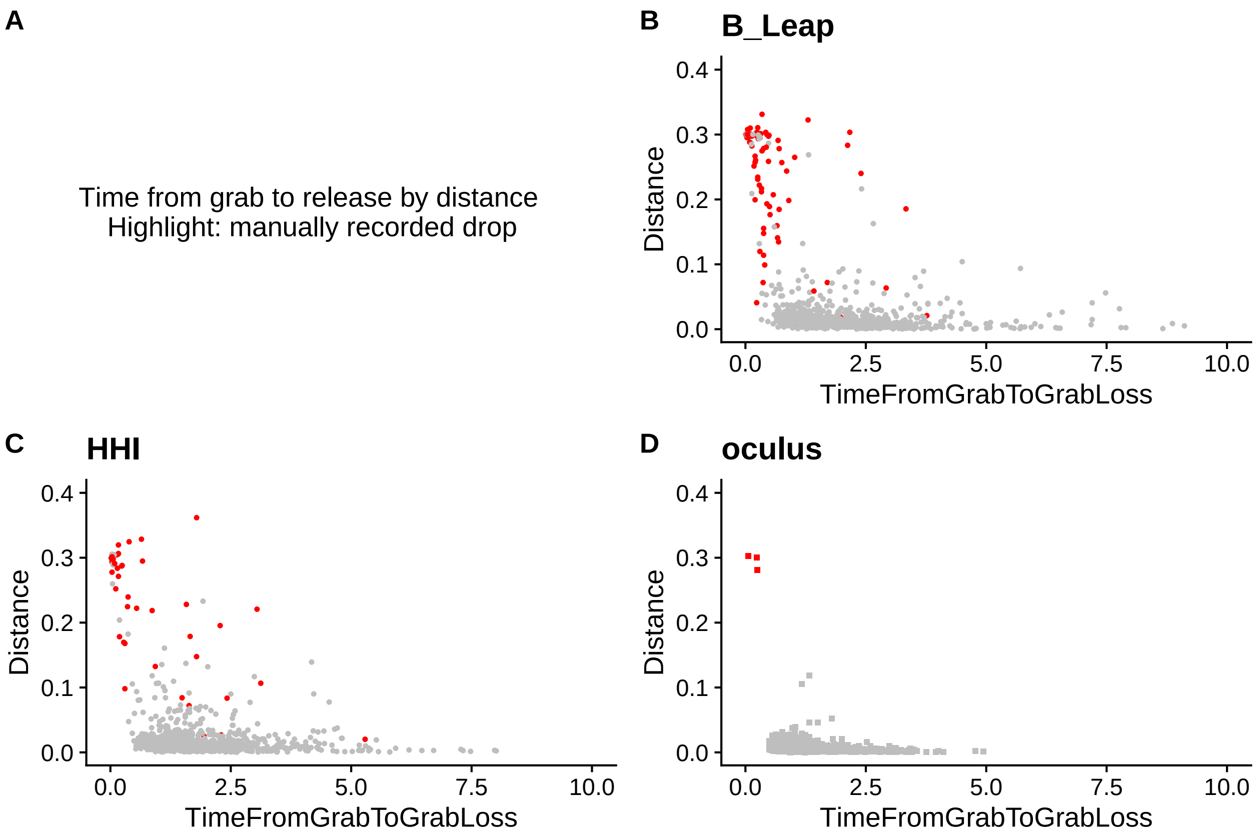
First *plot time from grab to release* by *distance.*

Note: starting distance is 0.3 m.

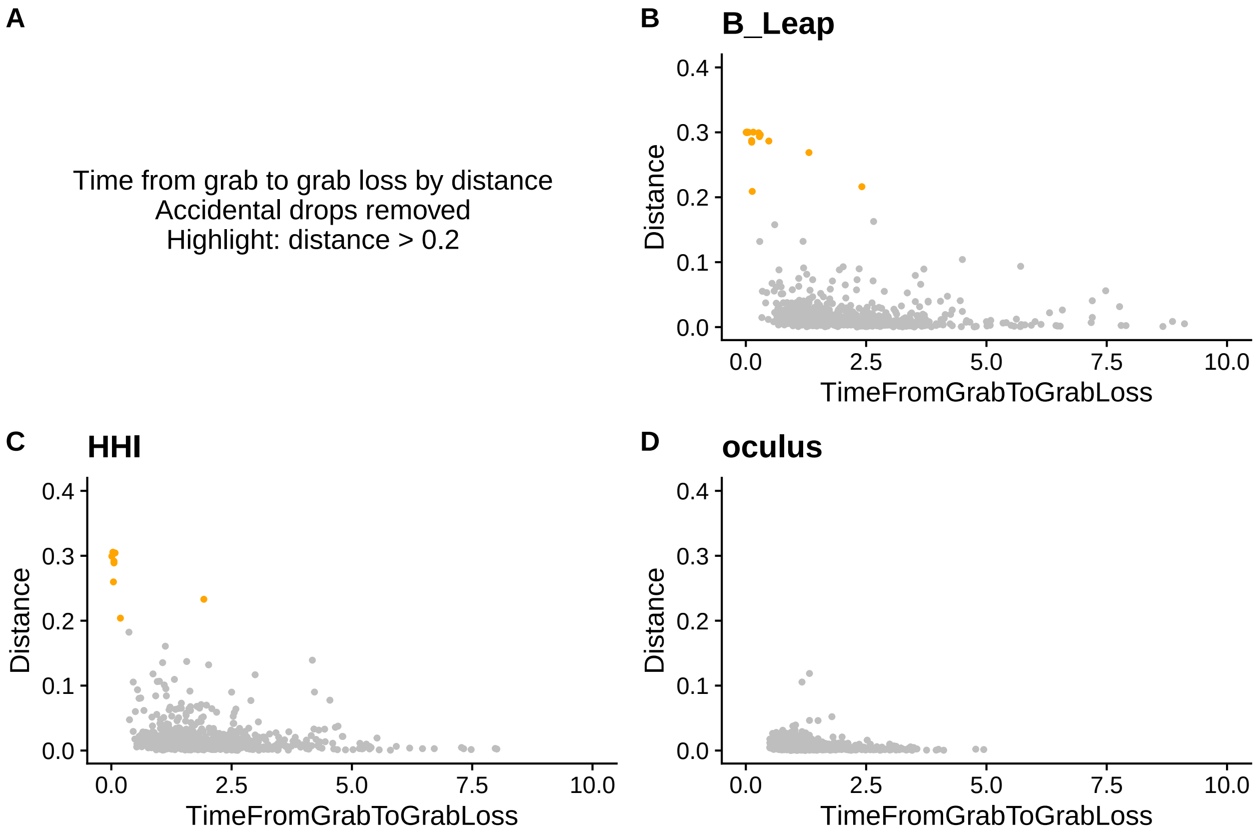
Plots: Distance by release time (time from grab to grab loss). Manually detected accidental drops are highlighted in red; distance > .2 m highlighted in orange; distance >.1 and < .2 highlighted in blue.

```r
# create data_set.clean for inspection
data_set.clean <- data_set %>% filter(Drop == 0, LandOnTable == TRUE)

# set limits
limits_y <- c(0, 0.4)
limits_x <- c(0, 10)

p_title <- ggdraw() + draw_label("Time from grab to release by distance \nHighlight: manually recorded d
    fontface = "plain")

leap_drops <- ggplot(data_set %>% filter(Interface == "B_Leap"), aes(TimeFromGrabToGrabLoss,
    Distance, color = Drop == 1, shape = LandOnTable)) + geom_point(size = 1) + theme(legend.position =
    scale_color_manual(values = c("Grey", "Red")) + scale_shape_manual(values = c(15,
    16)) + ggtitle(label = "B_Leap") + coord_cartesian(xlim = limits_x, ylim = limits_y)

HHI_drops <- ggplot(data_set %>% filter(Interface == "HHI_Leap"), aes(TimeFromGrabToGrabLoss,
    Distance, color = Drop == 1, shape = LandOnTable)) + geom_point(size = 1) + theme(legend.position =
    ggtitle(label = "HHI") + scale_color_manual(values = c("Grey", "Red")) + scale_shape_manual(values =
    16)) + coord_cartesian(xlim = limits_x, ylim = limits_y)

oculus_drops <- ggplot(data_set %>% filter(Interface == "Oculus"), aes(TimeFromGrabToGrabLoss,
```

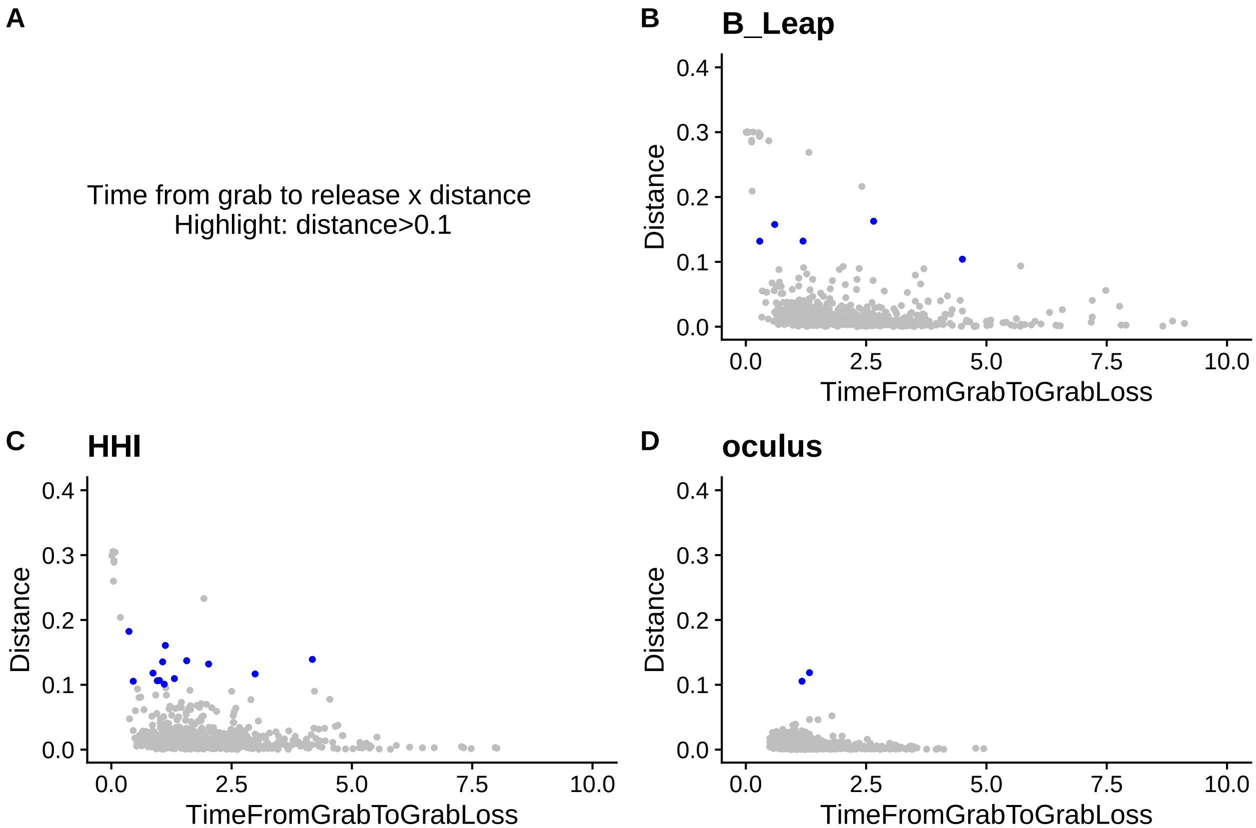
```
    Distance, color = Drop == 1, shape = LandOnTable)) + geom_point(size = 1) + theme(legend.position = 
    ggtitle(label = "oculus") + scale_color_manual(values = c("Grey", "Red")) + scale_shape_manual(value
    16)) + coord_cartesian(xlim = limits_x, ylim = limits_y)

plot_grid(p_title, leap_drops, HHI_drops, oculus_drops, labels = "AUTO")
```




```
# potentially missed accidental drops
p_title <- ggdraw() + draw_label("Time from grab to grab loss by distance\nAccidental drops removed\nHig
    fontface = "plain")

leap_drops <- ggplot(data_set.clean %>% filter(Interface == "B_Leap"), aes(TimeFromGrabToGrabLoss,
    Distance, color = Distance > 0.2)) + geom_point(size = 1) + scale_color_manual(values = c("Grey",
    "Orange")) + theme(legend.position = "none") + ggtitle(label = "B_Leap") + coord_cartesian(xlim = li
    ylim = limits_y)

HHI_drops <- ggplot(data_set.clean %>% filter(Interface == "HHI_Leap"), aes(TimeFromGrabToGrabLoss,
    Distance, color = Distance > 0.2)) + geom_point(size = 1) + ggtitle(label = "HHI") +
    theme(legend.position = "none") + scale_color_manual(values = c("Grey", "Orange")) +
    coord_cartesian(xlim = limits_x, ylim = limits_y)

oculus_drops <- ggplot(data_set.clean %>% filter(Interface == "Oculus"), aes(TimeFromGrabToGrabLoss,
    Distance, color = Distance > 0.2)) + geom_point(size = 1) + theme(legend.position = "none") +
    ggtitle(label = "oculus") + scale_color_manual(values = c("Grey", "Orange")) +
    coord_cartesian(xlim = limits_x, ylim = limits_y)

plot_grid(p_title, leap_drops, HHI_drops, oculus_drops, labels = "AUTO")
```




```
# release times by distance
p_title <- ggdraw() + draw_label("Time from grab to release x distance \nHighlight: distance>0.1",
    fontface = "plain")

leap_drops <- ggplot(data_set.clean %>% filter(Interface == "B_Leap", Drop == 0),
    aes(TimeFromGrabToGrabLoss, Distance, color = Distance >= 0.1 & Distance < 0.2)) +
    geom_point(size = 1) + scale_color_manual(values = c("Grey", "Blue")) + theme(legend.position = "no
    ggtitle(label = "B_Leap") + coord_cartesian(xlim = limits_x, ylim = limits_y)

HHI_drops <- ggplot(data_set.clean %>% filter(Interface == "HHI_Leap", Drop == 0),
    aes(TimeFromGrabToGrabLoss, Distance, color = Distance >= 0.1 & Distance < 0.2)) +
    geom_point(size = 1) + theme(legend.position = "none") + ggtitle(label = "HHI") +
    scale_color_manual(values = c("Grey", "Blue")) + coord_cartesian(xlim = limits_x,
    ylim = limits_y)

oculus_drops <- ggplot(data_set.clean %>% filter(Interface == "Oculus", Drop == 0),
    aes(TimeFromGrabToGrabLoss, Distance, color = Distance >= 0.1 & Distance < 0.2)) +
    geom_point(size = 1) + ggtitle(label = "oculus") + theme(legend.position = "none") +
    scale_color_manual(values = c("Grey", "Blue")) + coord_cartesian(xlim = limits_x,
    ylim = limits_y)

plot_grid(p_title, leap_drops, HHI_drops, oculus_drops, labels = "AUTO")
```



Looking at the pattern of manually recorded accidental drops (red), we can see that though most tend to be short times with distances around 0.3, some have longer times and smaller distances. There is no definitive rule for automatic detection.

Clearly some accidental drops were missed (**orange**), as there is a small cluster of data points in the top left for both Leap variants, with a distance around 0.3 and a time close to 0. The data points with longer times and distances over 0.2 are more mysterious; however, these could very well be accidental drops as well that were not observed by the experimenter.

With a distance cut-off of 0.2, these are removed, leaving a series of data points between the distances of 0.1 and 0.2 which need to be explained – they may or may not also be accidental drops. Visually, the HHI Leap seems to have more of these data points. This is especially confusing because several of these points for both Leap conditions have times (grab and release) that are greater than 2 seconds.

Simply removing all data points with a distance greater than 0.1 would improve the accuracy metric for the HHI Leap. These data points should contain some sort of penalty, either as errors or in the accuracy average.

There are data points with low times that also have high accuracy (low distance). Therefore, simply using a low time from grab to release is not enough to automatically detect accidental drops.

Distance cut-offs

All data points over 0.2 can reasonably be considered accidental drops. More investigation needs to be done to determine what to do with **data points with a distance between 0.1 and 0.2**. For **Longer times**, it is difficult to explain these data points between distances of 0.1 and 0.2, *especially those with longer times from grab to release.*

Shorter release times in the 0.1-0.2 distance range could indicate accidental drops that were not caught by the experimenter.

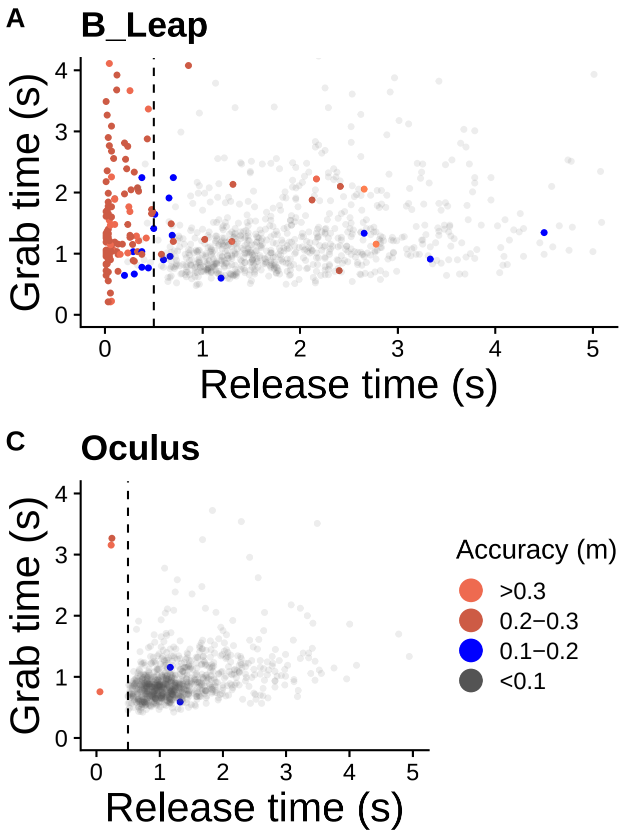
We can try to visualize grab and release errors by plotting time to grab and time from grab to release on one plot. I've done so below, using *color* to indicate long distances. Red: closer to 0.3; blue: between 1 and 2; grey: less than 1. Square data points indicate cubes that did not land on the table.

These plots *include* manually-detected accidental drops (they have not yet been removed).

Metric-based detection: Accidental Drops

```
temp_plot_data <- data_set
dist_group <- "string"

# put each trial into a bin for plotting purposes
for (x in 1:length(temp_plot_data$Distance)) {
    if (temp_plot_data$LandOnTable[x] == FALSE) {
        dist_group[x] <- "off table"
    } else {
        if (temp_plot_data$Distance[x] < 0.1) {
            dist_group[x] <- "<0.1"
        }
        if (temp_plot_data$Distance[x] >= 0.1 & temp_plot_data$Distance[x] < 0.2) {
            dist_group[x] <- "0.1-0.2"
        }
        if (temp_plot_data$Distance[x] >= 0.2 & temp_plot_data$Distance[x] <= 0.3) {
            dist_group[x] <- "0.2-0.3"
        }
        if (temp_plot_data$Distance[x] > 0.3) {
            dist_group[x] <- ">0.3"
        }
    }
}
dist_group = factor(dist_group, levels = c("<0.1", "0.1-0.2", "0.2-0.3", ">0.3",
    "off table"))
if ("dist_group" %in% names(temp_plot_data)) {
} else {
    temp_plot_data <- cbind(temp_plot_data, dist_group)
}
rm(dist_group)

# grab times by release times w/ color gradient for distance


# set parameters
grab_lims = c(0, 4)
release_lims = c(0, 5)
alpharange = c(0.1, 0.4)
val_colors <- c("grey33", "blue", "coral3", "coral2", "coral")


# title
p_title <- ggdraw() + draw_label("Dashed line: Release time = 0.5 s\n\nBlue dots left of dotted line ar
    fontface = "plain", hjust = 0.6)

p_leap <- ggplot(temp_plot_data %>% filter(Interface == "B_Leap"), aes(TimeFromGrabToGrabLoss,
    TimeFromSpawnToGrab, color = dist_group, alpha = Distance > 0.1)) + geom_point(size = 1) +
    # scale_color_gradientn(colors=c('Grey','Blue','Red'), limits=c(0,0.3))+
```

```r
# scale_color_gradient(low='grey33', high='Red', guide='colourbar',
# limits=c(0,0.3))+ scale_alpha_continuous(limits=c(0,0.1), range=alpharange,
# guide=guide_legend(reverse=TRUE, title='< 0.1'))+
theme(plot.title = element_text(size = title_size * 0.6), axis.title = element_text(size = axis_text_si
    0.7)) + scale_shape_manual(values = c(15, 16), guide = "none") + scale_alpha_manual(values = c(0.1,
    1), guide = "none") + scale_color_manual(values = val_colors, guide = "none") +
    geom_vline(aes(xintercept = 0.5), linetype = "dashed") + labs(color = "Accuracy",
    title = "B_Leap", x = "Release time (s)", y = "Grab time (s)") + # theme(legend.position = 'none')+
coord_cartesian(ylim = grab_lims, xlim = release_lims)

p_HHI <- ggplot(temp_plot_data %>% filter(Interface == "HHI_Leap"), aes(TimeFromGrabToGrabLoss,
    TimeFromSpawnToGrab, color = dist_group, alpha = Distance > 0.1)) + geom_point(size = 1) +
    scale_color_manual(values = val_colors) + scale_shape_manual(values = c(15, 16),
    guide = "none") + geom_vline(aes(xintercept = 0.5), linetype = "dashed") + # scale_alpha_continuous
# scale_color_gradientn(colors=c('Grey','Blue','Red'), limits=c(0,0.3))+
scale_alpha_manual(values = c(0.1, 1)) + theme(plot.title = element_text(size = title_size *
    0.6), axis.title = element_text(size = axis_text_size * 0.7)) + # scale_color_gradient(low='Grey',
# limits=c(0,0.3))+
theme(legend.position = "none") + labs(color = "Accuracy", title = "HHI_Leap", x = "Release time (s)",
    y = "Grab time (s)") + coord_cartesian(ylim = grab_lims, xlim = release_lims)

p_oculus <- ggplot(temp_plot_data %>% filter(Interface == "Oculus"), aes(TimeFromGrabToGrabLoss,
    TimeFromSpawnToGrab, color = dist_group, alpha = Distance > 0.1)) + geom_point(size = 1) +
    # scale_color_gradient(low='Grey', high='Red', guide='colourbar',
# limits=c(0,0.3))+ scale_color_gradientn(colors=c('Grey','Blue','Red'),
# limits=c(0,0.3))+
scale_alpha_manual(values = c(0.1, 1), guide = "none") + geom_vline(aes(xintercept = 0.5),
    linetype = "dashed") + theme(plot.title = element_text(size = title_size * 0.6),
    axis.title = element_text(size = axis_text_size * 0.7)) + scale_color_manual(values = val_colors,
    guide = guide_legend(reverse = TRUE, override.aes = list(size = 5)), breaks = c("<0.1",
        "0.1-0.2", "0.2-0.3", ">0.3", "off table")) + scale_shape_manual(values = c(16,
    15), guide = "none") + # scale_alpha_continuous(limits=c(0,0.1), range=alpharange, guide='none')+
# theme(legend.position = 'none')+
labs(color = "Accuracy (m)", title = "Oculus", x = "Release time (s)", y = "Grab time (s)") +
    coord_cartesian(ylim = grab_lims, xlim = release_lims)

plot_grid(p_leap, p_HHI, p_oculus, labels = c("A", "B", "C", ""))  #p_title)
```




```
ggsave("accidental_drop_detection.jpg", width = 10, height = 8)
```

For B_Leap, there are many red dots on the left side of the plot, where time from grab to release is close to 0. This indicates a quick drop right after pick-up and a long distance from the target – almost definitely an accidental drop.

For HHI Leap, there are not as many red dots, but there appear to be more blue dots (between 0.1 and 0.2), e.g., there is a blue dot with a grab time of ~1 second and a release time of ~4 seconds.

The last plot is to determine if the same subjects are responsible for the weird values (distance: 0.1-0.2). Subjects 16 and 25 each have 3 values on this plot. The question we are trying to answer is: are values under a certain time indicative of an accidental drop?

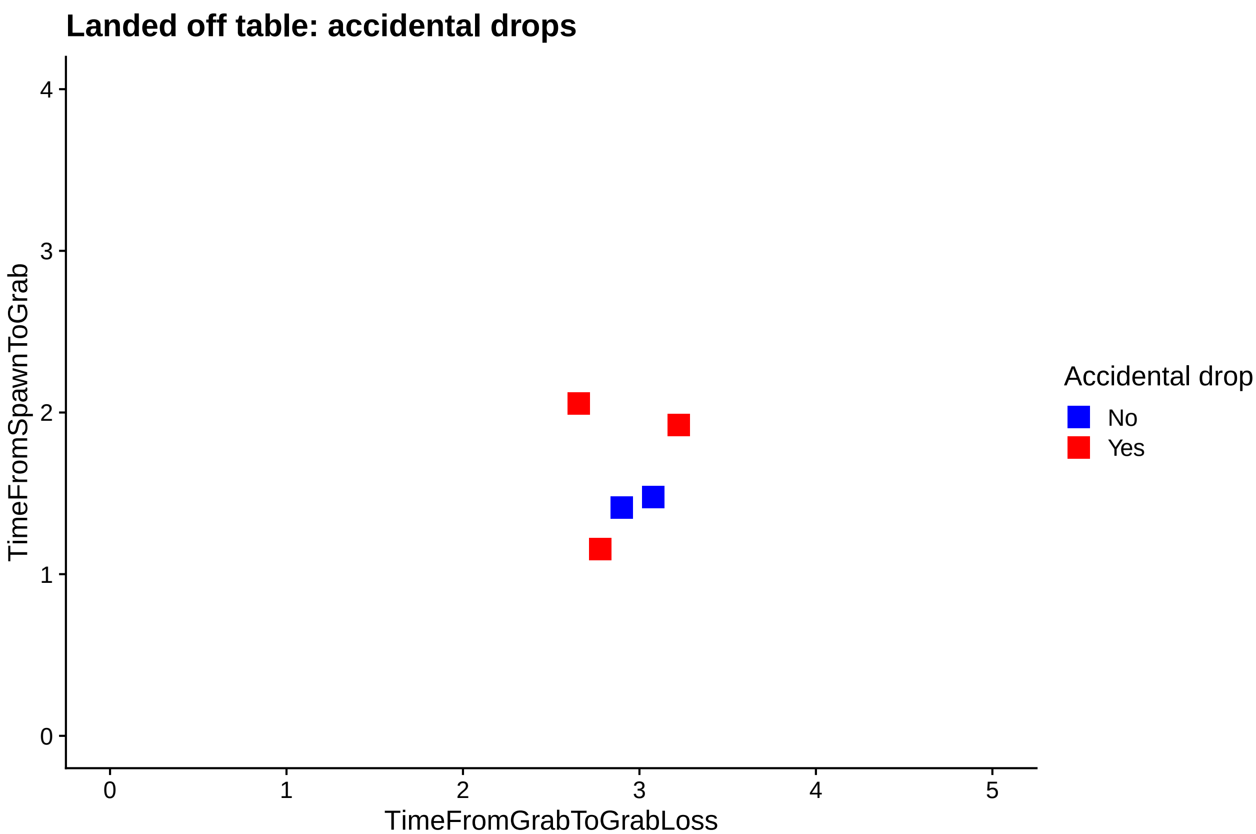
The red dots in the bottom-left corner, where x and y are both close to 0, *may represent an accidental spawn into the hand*. There appears to be only one, for the B_Leap, that really fits this description (however, this is following removal of manually-detected accidental drops). This method could potentially determine which of the manually-detected accidental drops were spawn-into-the-hand errors, **which could potentially demonstrate the success of the HHI Leap's modified grab algorithm**.

Closer look

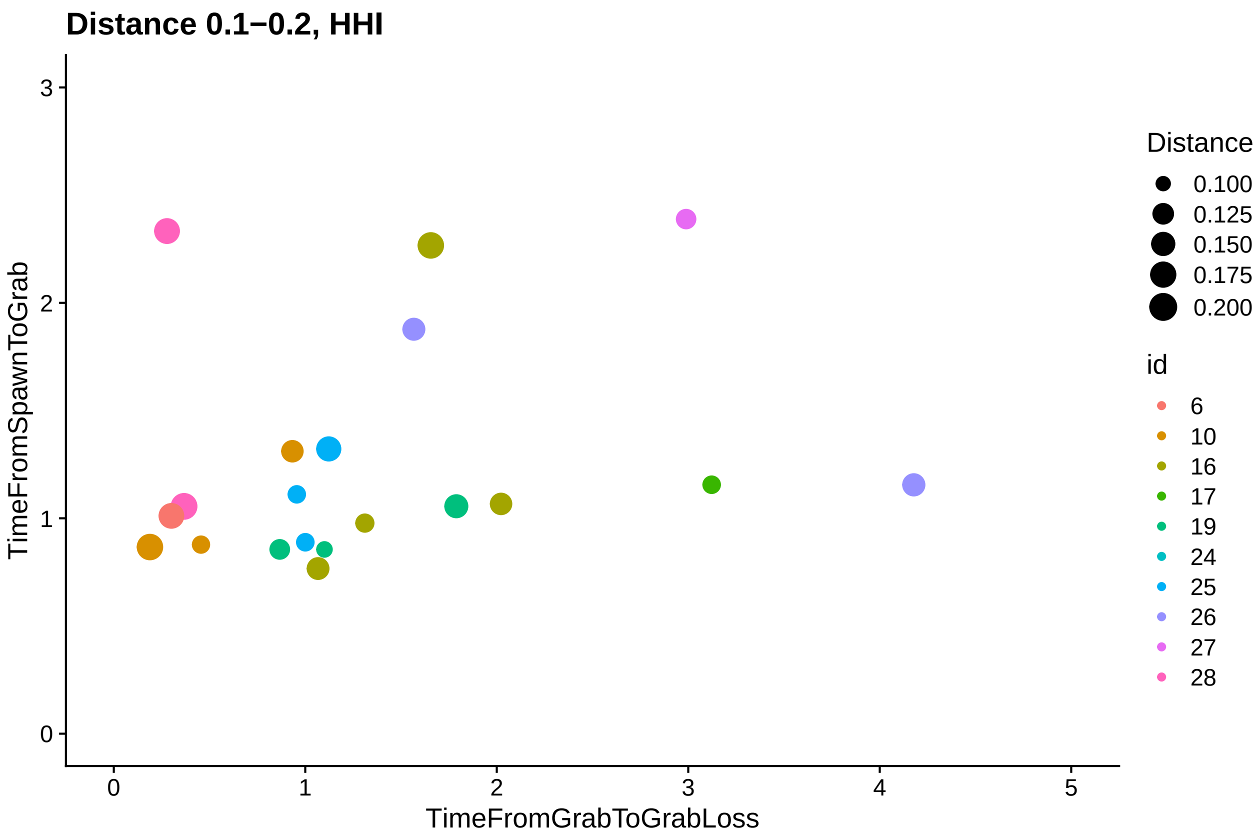
Below are plots of * trials that did not land on the table * distances between 0.1 and 0.2, color-coded by subject * very low grab times with high accuracy, color-coded by subject

```
ggplot(temp_plot_data %>% filter(LandOnTable == FALSE), aes(TimeFromGrabToGrabLoss,
    TimeFromSpawnToGrab, color = as.factor(Drop))) + geom_point(size = 5, shape = 15) +
    scale_color_manual(values = c("blue", "red"), labels = c("No", "Yes"), guide = guide_legend(title =
    scale_shape_manual(values = c(16, 15)) + # theme(legend.position = 'none')+
```

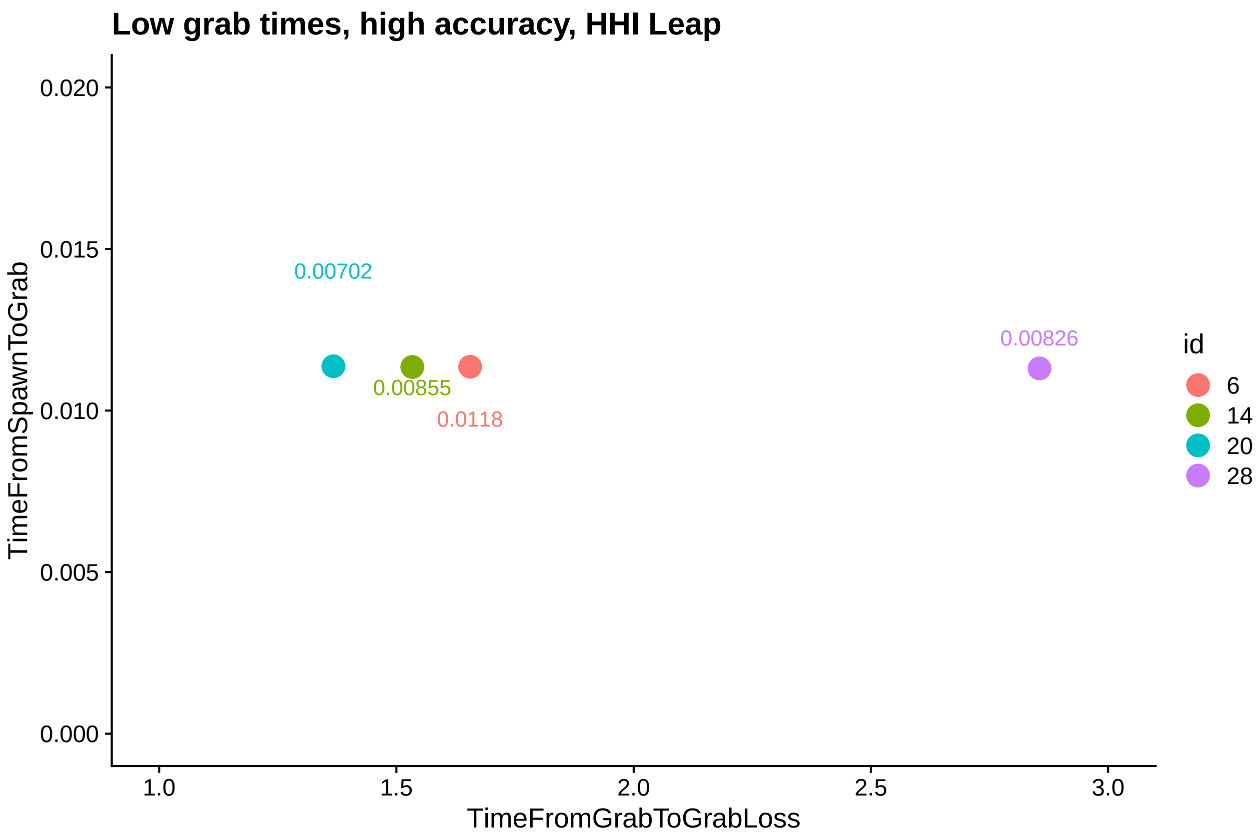
```r
ggtitle(label = "Landed off table: accidental drops") + coord_cartesian(ylim = grab_lims,
    xlim = release_lims)
```




```r
ggplot(temp_plot_data %>% filter(Interface == "HHI_Leap", Distance >= 0.1 & Distance <
    0.2), aes(TimeFromGrabToGrabLoss, TimeFromSpawnToGrab, color = id, size = Distance)) +
    geom_point() + scale_size_continuous(limits = c(0.1, 0.2), range = c(3, 6)) +
    # theme(legend.position = 'none')+
ggtitle(label = "Distance 0.1-0.2, HHI") + coord_cartesian(ylim = c(0, 3), xlim = release_lims)
```




```
ggplot(temp_plot_data %>% filter(Interface == "HHI_Leap", Distance < 0.02, TimeFromSpawnToGrab <
    0.3), aes(TimeFromGrabToGrabLoss, TimeFromSpawnToGrab, color = id)) + geom_point(size = 5) +
    geom_text(aes(label = Distance), vjust = -0.1, position = position_jitter(width = 0,
        height = 0.003)) + # scale_size_continuous(limits=c(0.1,0.2), range=c(3,6))+ theme(legend.posit
# 'none')+
ggtitle(label = "Low grab times, high accuracy, HHI Leap") + coord_cartesian(ylim = c(0,
    0.02), xlim = c(1, 3))
```



There are four points for the HHI Leap that have a grab time between 0.01 and 0.015 seconds, yet very high accuracy (shown in text). This may indicate an error in how the system registered grab time, or perhaps a data entry error. These will be eliminated by setting a grab time cut-off at 0.1 seconds and will *not* be added to the accidental drop counts.

5 trials were logged by the system as having not landed on the table. 3 of them were recorded manually as accidental drops. I think it is safe to say that the other two were also accidental drops, simply missed by the researcher. The additional two will be included as accidental drops.

Visualize all Unity data

Distance or **Accuracy** is defined as the length of the 2D vector from the center point of the cube's bottom surface to the center point of the target. Unit is meters. Distance of the cube at spawn: 30 cm (0.3 m). Spawn = regeneration of cube at the beginning of a new trial, taken from video gaming lingo.

Time from spawn to grab is the time from when the cube spawned to the time that the user successfully grabbed the cube. This will be renamed to **grab time**.

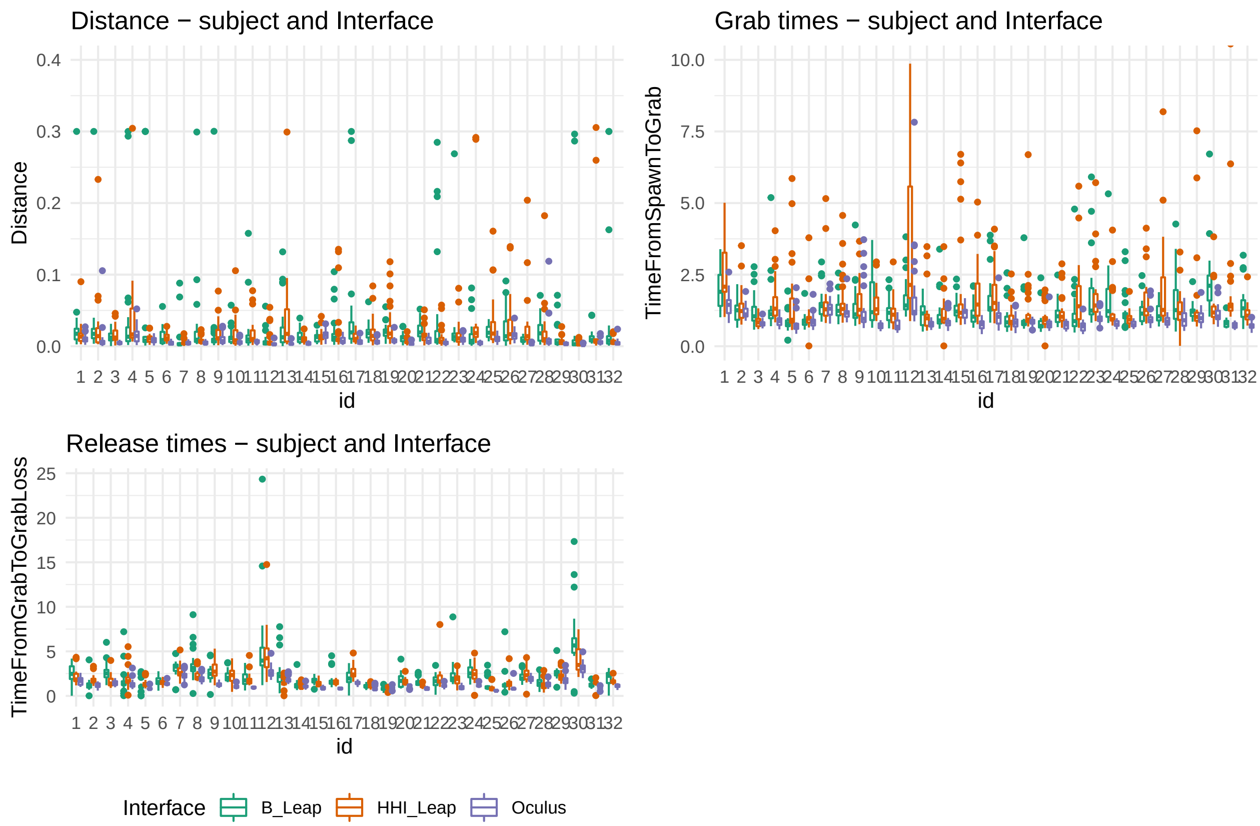
Time from grab to grab loss is the measurement of time from grab to release. This will be renamed to **release time**.

Time from spawn to grab loss is the total time for each trial. This will be renamed to **total time**.

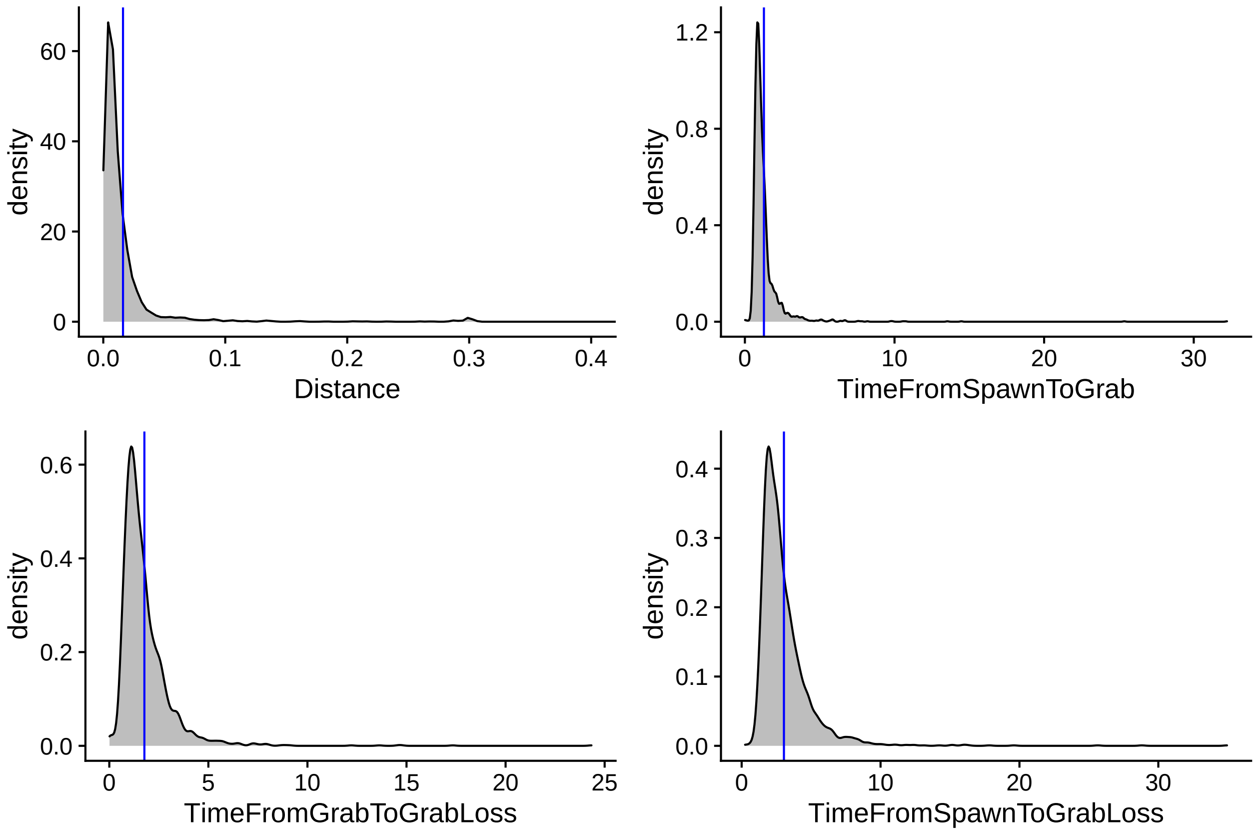
The following plots explore these metrics; manually-recorded drops and trials where the cube did not land on the table are *not inculded*.

```
#note: using data_set.clean (drops and offtable removed)

# box plot, distance by subject, pre clean
```

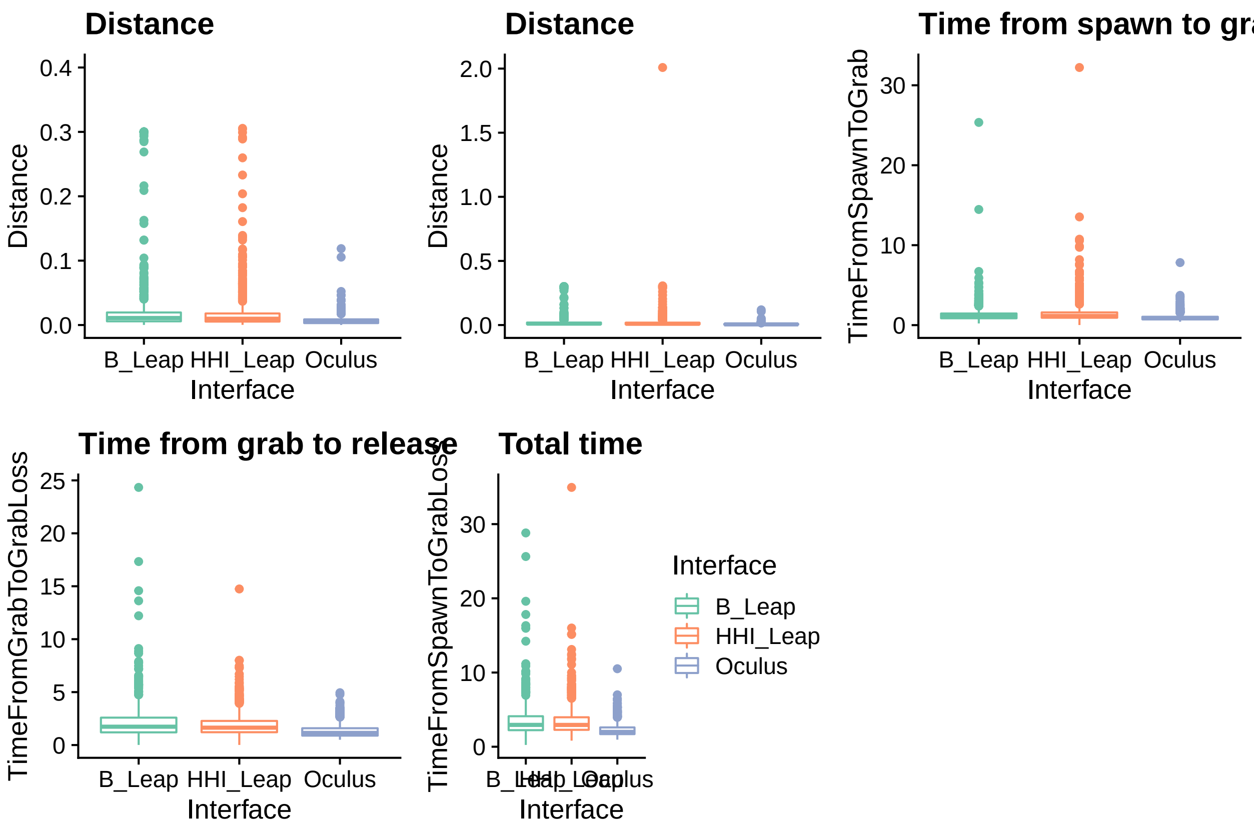
```r
distance_subjects_box<-  ggplot(data_set.clean, aes(id, Distance, color=Interface)) +
  theme_minimal() +
  theme(legend.position = "none") +
  geom_boxplot(#outlier.colour="black",
           outlier.size=1, notch=FALSE) +
  scale_color_brewer(palette="Dark2") +
    labs(title="Distance - subject and Interface") +
    coord_cartesian(ylim=c(0, 0.4))

# grab times box plot of subjects and Interfaces pre clean
grabtime_subjects_box<-  ggplot(data_set.clean, aes(id, TimeFromSpawnToGrab, color=Interface)) +
  theme_minimal() +
  theme(legend.position = "none") +
  geom_boxplot(outlier.size=1, notch=FALSE) +
  # geom_point() +
#  geom_violin(scale="area") +
  scale_color_brewer(palette="Dark2") +
    labs(title="Grab times - subject and Interface") +
    coord_cartesian(ylim=c(0, 10))

# release times - by subject - box plot
releasetime_subjects_box<- ggplot(data_set.clean, aes(id, TimeFromGrabToGrabLoss, color=Interface)) +
theme_minimal() +
theme(legend.position = "bottom") +
geom_boxplot(outlier.size=1, notch=FALSE) +
scale_color_brewer(palette="Dark2") +
  labs(title="Release times - subject and Interface") #+coord_cartesian(ylim=c(0, 0.1))

plot_grid(distance_subjects_box, grabtime_subjects_box, releasetime_subjects_box)
```




```
# density plot: distance
distance_density<- ggplot(data_set.clean, aes(Distance))+
  #geom_histogram(binwidth = 0.1)+
  geom_density(fill="Grey")+labs(paste0("Distance, mean=",round(mean(data_set.clean$Distance),2)))+
  geom_vline(aes(xintercept = mean(Distance)), color="Blue")+theme(legend.position = "none")+coord_carte

# density plot: spawn to grab
grabtime_density<- ggplot(data_set.clean, aes(TimeFromSpawnToGrab))+
  #geom_histogram(binwidth = 0.1)+
  geom_density(fill="Grey")+labs(paste0("Grab time, mean=",round(mean(data_set.clean$TimeFromSpawnToGrab

# density plot: grab to release time
releasetime_density<- ggplot(data_set.clean, aes(TimeFromGrabToGrabLoss))+
  #geom_histogram(binwidth = 0.1)+
  geom_density(fill="Grey")+labs(paste0("Release time, mean=",round(mean(data_set.clean$TimeFromGrabToG
  geom_vline(aes(xintercept = mean(TimeFromGrabToGrabLoss)), color="Blue")+theme(legend.position = "none

# density plot: total time
totaltime_density<- ggplot(data_set.clean, aes(TimeFromSpawnToGrabLoss))+
  #geom_histogram(binwidth = 0.1)+
  geom_density(fill="Grey")+labs(paste0("Total time, mean=",round(mean(data_set.clean$TimeFromSpawnToGr
  geom_vline(aes(xintercept = mean(TimeFromSpawnToGrabLoss)), color="Blue")#+coord_cartesian(xlim=c(0,1

plot_grid(distance_density, grabtime_density, releasetime_density, totaltime_density)
```




```r
# bot plots
distance_box<- ggplot(data_set.clean, aes(Interface, Distance, color=Interface))+geom_boxplot()+scale_c
  #geom_text(data=data_set.clean %>% filter(Distance > 0.5), aes(Interface, Distance, label=paste0(Dist

distance_box2<- ggplot(data_set.clean, aes(Interface, Distance, color=Interface))+
  geom_boxplot(guide="none")+scale_color_brewer(palette="Set2")+ggtitle(label="Distance")+theme(legend.

grabtime_box<- ggplot(data_set.clean, aes(Interface, TimeFromSpawnToGrab, color=Interface))+
  geom_boxplot()+scale_color_brewer(palette="Set2")+ggtitle(label="Time from spawn to grab")+theme(leger

releasetime_box<- ggplot(data_set.clean, aes(Interface, TimeFromGrabToGrabLoss, color=Interface))+
  geom_boxplot()+scale_color_brewer(palette="Set2")+ggtitle(label="Time from grab to release")+theme(leg

totaltime_box<- ggplot(data_set.clean, aes(Interface, TimeFromSpawnToGrabLoss, color=Interface))+
  geom_boxplot()+scale_color_brewer(palette="Set2")+ggtitle(label="Total time")#+coord_cartesian(ylim=c

plot_grid(distance_box, distance_box2, grabtime_box, releasetime_box, totaltime_box)
```



Extreme oultliers

There are some extremely long times in both the grab and release time metrics.

For **grab times**, a time of 30 seconds may be due to a problem that we do not want to include in our metric, such as if the subject stopped and asked the experimenter for help. These outliers may have an impact on the mean and will certainly affect the variance.

Subject 12 seems to have absurdly long grab times for the HHI Leap Motion.

Subjects 12 and 30 both seem to absurdly have long release times. They also have very low distance times. Clearly they prioritized accuracy over time.

Subjects 31 and 32 were run as potential replacements. The grab times for subject 31 on the HHI Leap seem much higher than the other two, but the difference from the other two Interfaces, and variance, are not so large as with subject 12. Release times for subjects 31 and 32 seems normal.

There is a case for considering these two to be outliers.

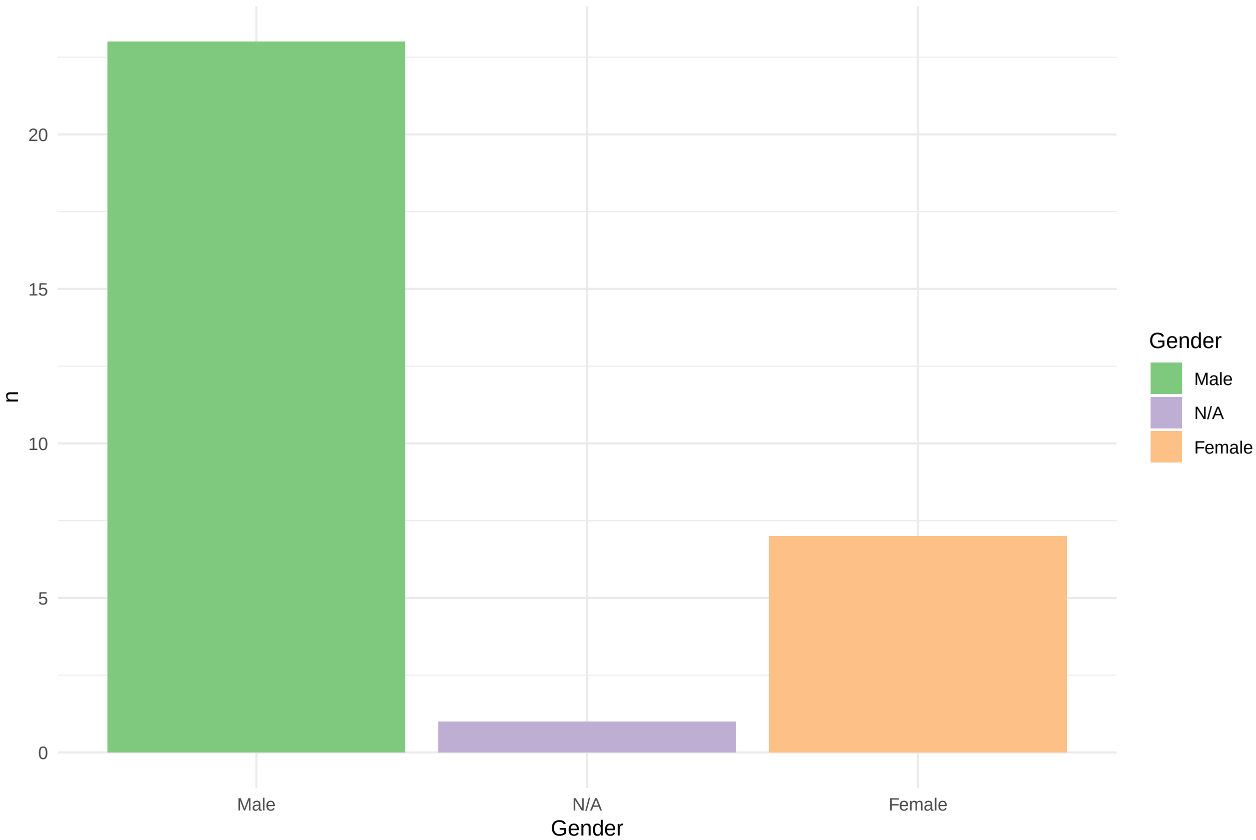
Let's take a quick look at subject 12 and 30's demographics and other stats compared to the rest of the over-all group.

Then we will look at overall times for subjects 12 and 30, as well as overall times for subjects 31 and 32.

Replace subjects?

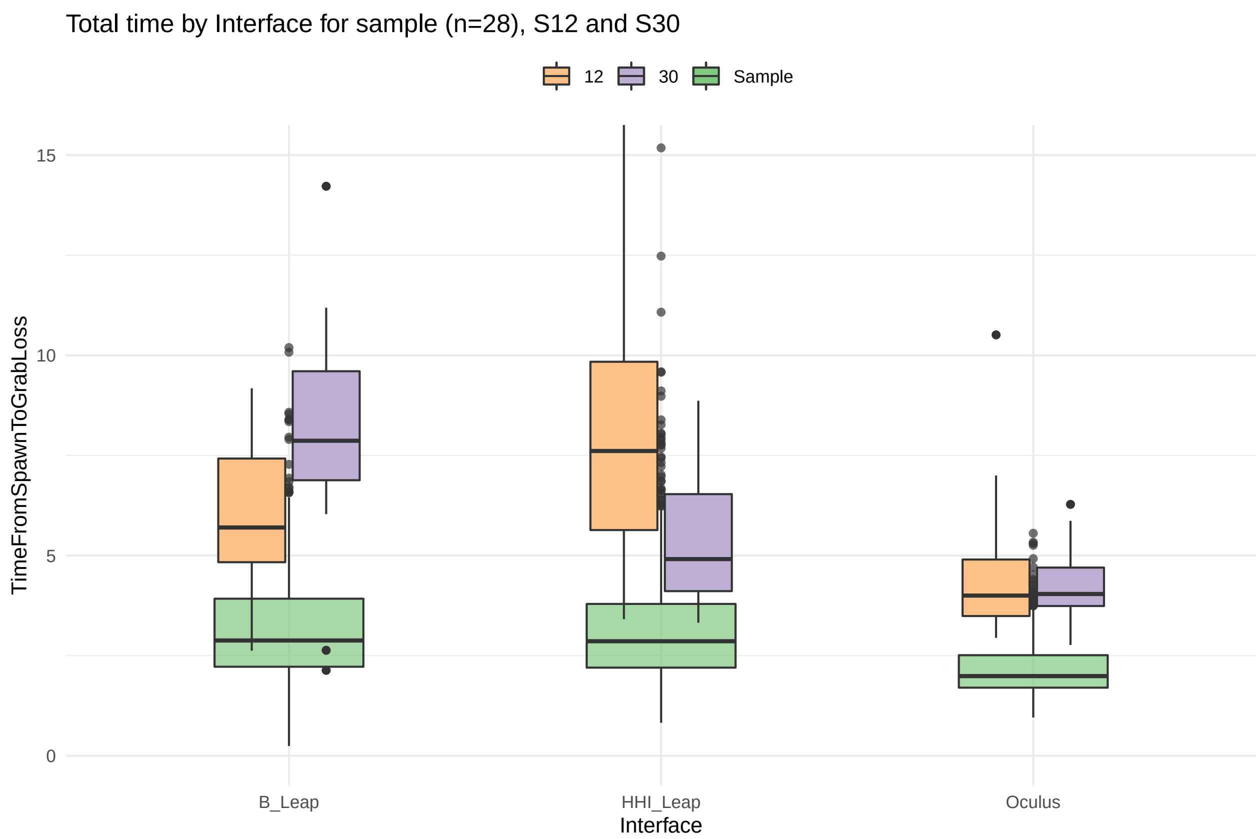
```
# subject 12 and 30's demo's
cat("\nSubject 12 and 30:\n")
survey_data %>%
  filter(UserID==12 | UserID==30) %>%
```

```r
  select(UserID, Age:SkillGames, Gender, Hand)

# the sample
cat("\nThe sample:\n")
survey_data %>%
  filter(UserID!=12 & UserID!=30) %>%
  select(Age:SkillGames) %>%
  summarise_each(median)
cat("\n")

# sample gender distribution
ggplot(survey_data %>%
         filter(UserID!=12) %>%
         select(UserID, Gender) %>%
         count(Gender) %>%
         mutate(percent=round(100*(n/sum(n)),1)),
       aes(x=Gender, y=n, fill=Gender)) +
  geom_bar(stat="identity") + theme_minimal() + scale_fill_brewer(palette=1, type="qual")
```

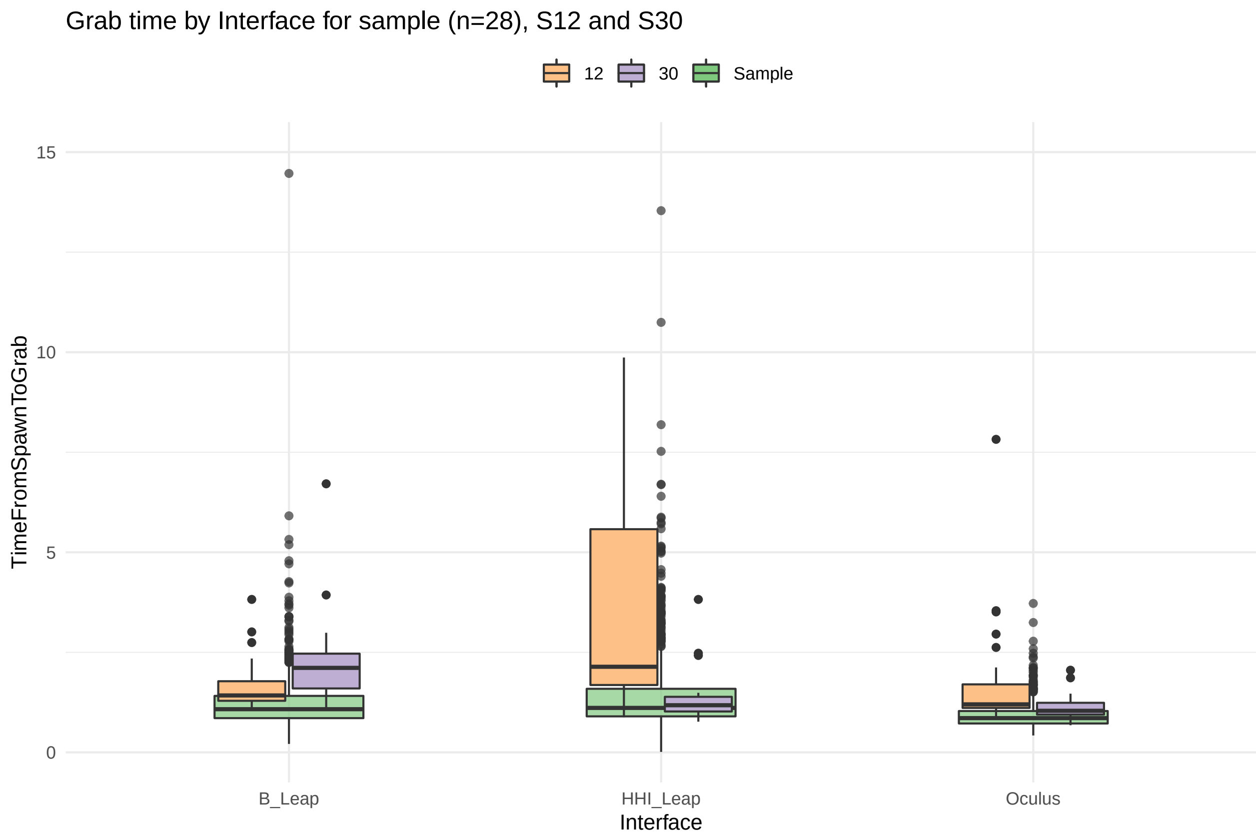



```r
# sample hand
survey_data %>%
  filter(UserID!=12) %>%
  select(UserID, Hand) %>%
  count(Hand) %>%
  mutate(percent=round(100*(n/sum(n)),1))
```

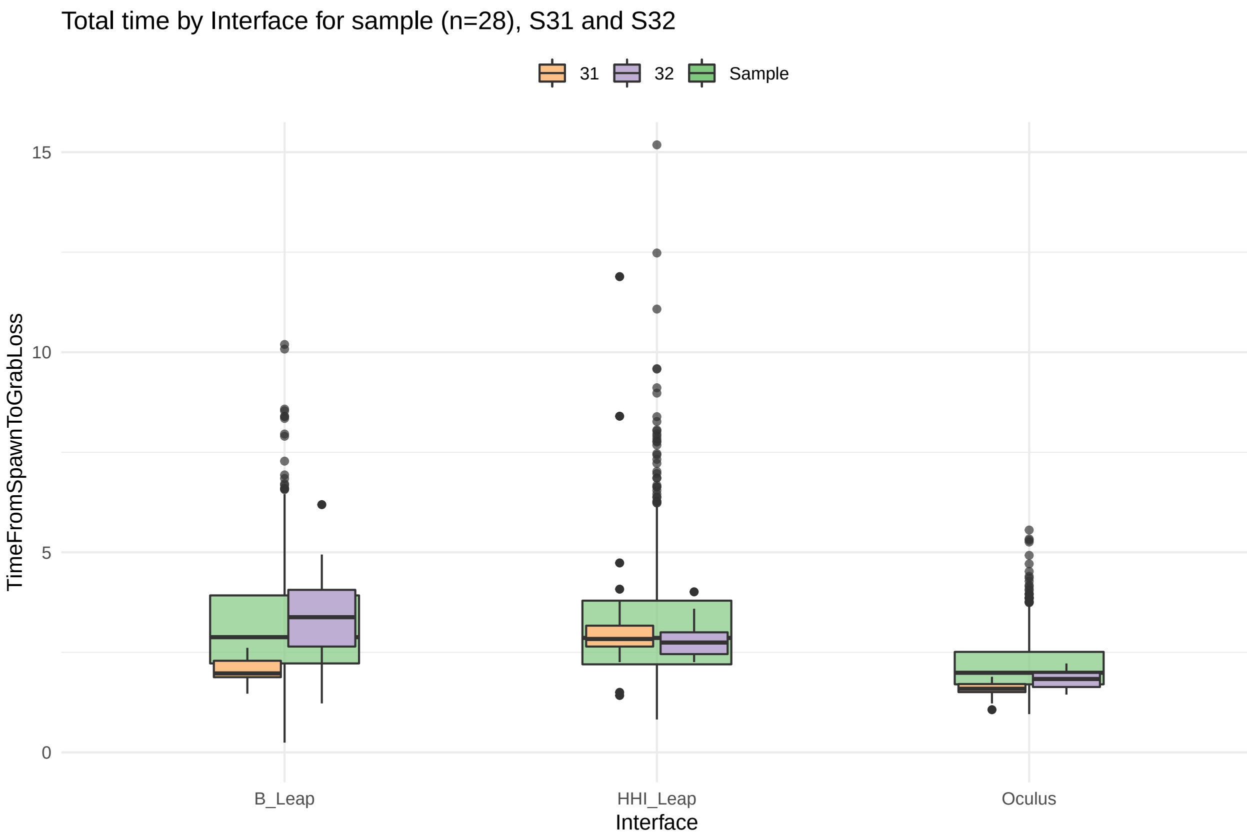
```r
# compare total times for subjects 12 and 30 to sample
ggplot(data_set.clean %>% filter(id!=12, id!=30, id!=31, id!=32),
       aes(Interface, TimeFromSpawnToGrabLoss, fill="Sample")) +
  theme_minimal() +
  scale_fill_brewer(palette=1, type="qual", direction=-1)+#1, type="qual")+
  # scale_fill_manual(values=brewer.pal(n = 8, name = "Set2"),
  #   labels=c("Subject 12", "Subject 13", "B_Leap", "HHI", "Oculus"))+
  theme(legend.position = "top", legend.title=element_blank()) +
  geom_boxplot(width=0.4, alpha=0.7) +
  geom_boxplot(data=data_set.clean %>% filter(id==12 | id==30), aes(Interface, TimeFromSpawnToGrabLoss,
  labs(title="Total time by Interface for sample (n=28), S12 and S30") +
  coord_cartesian(ylim=c(0, 15))
```




```r
# compare total times for subjects 12 and 30 to sample
ggplot(data_set.clean %>% filter(id!=12, id!=30, id!=31, id!=32),
       aes(Interface, TimeFromSpawnToGrab, fill="Sample")) +
  theme_minimal() +
  scale_fill_brewer(palette=1, type="qual", direction=-1)+#1, type="qual")+
  # scale_fill_manual(values=brewer.pal(n = 8, name = "Set2"),
  #   labels=c("Subject 12", "Subject 13", "B_Leap", "HHI", "Oculus"))+
  theme(legend.position = "top", legend.title=element_blank()) +
  geom_boxplot(width=0.4, alpha=0.7) +
  geom_boxplot(data=data_set.clean %>% filter(id==12 | id==30), aes(Interface, TimeFromSpawnToGrab, fil
  labs(title="Grab time by Interface for sample (n=28), S12 and S30") +
  coord_cartesian(ylim=c(0, 15))
```




```r
# compare total times for subjects 31 and 32 to sample
ggplot(data_set.clean %>% filter(id!=12, id!=30, id!=31, id!=32),
       aes(Interface, TimeFromSpawnToGrabLoss, fill="Sample")) +
theme_minimal() +
scale_fill_brewer(palette=1, type="qual", direction=-1)+#1, type="qual")+
# scale_fill_manual(values=brewer.pal(n = 8, name = "Set2"),
#   labels=c("Subject 12", "Subject 13", "B_Leap", "HHI", "Oculus"))+
theme(legend.position = "top", legend.title=element_blank()) +
geom_boxplot(width=0.4, alpha=0.7) +
geom_boxplot(data=data_set.clean %>% filter(id==31 | id==32), aes(Interface, TimeFromSpawnToGrabLoss,
labs(title="Total time by Interface for sample (n=28), S31 and S32") +
coord_cartesian(ylim=c(0, 15))
```




```
##
## Subject 12 and 30:
##   UserID Age Height Arm Disabilities SkillController SkillVR SkillGames Gender
## 1     12  27    175  69            1               2       2          5 Female
## 2     30  25    174  75            1               3       3          3 Female
##    Hand
## 1 Right
## 2 Right
##
## The sample:
##   Age Height Arm Disabilities SkillController SkillVR SkillGames
## 1  28    180  NA            1               3       2          4
##
## # A tibble: 2 x 3
##   Hand      n percent
##   <fct> <int>   <dbl>
## 1 Left      1     3.2
## 2 Right    30    96.8
```

We would be losing two females from an already small group.

Looking at release times of all involved, those of subjects 12 and 30 are clearly far above the normal range compared to those of 31 and 32. However, this may not be enough justification to remove them: All we know is that these are users who participated in the study. They may represent a part of the true population variance.

For now, we will not remove any subjects.

Clean Unity data

All cleaning will occur in this section.

NOTE: the variable below, *remove_subjects*, takes those subjects out of the data set. The variable *include* is used to filter subjects that will be included in the analysis. For now, all subjects are kept in the *include* variable.

Summary of cut-offs

Below includes all cut-offs (including initial cleaning):

Cut and add to accidental drop count: * Distance >=0.2 * Distance >= 0.1 & <= 0.2 & TimeFromGrabToGrabLoss < 0.5 * LandOnTable==FALSE

```
sink("accidental_drop_counts.txt")


remove_subjects <- 99 #remove_these_subjects #c(12, 30) # insert subject numbers seperated by commas he
include <- c(1:sample_size)
include <- include[-remove_subjects] # this line removes subjects specified above

# remove accidental drops and add to accidental drop count
# data_set will contain all original drop counts; unity_data_clean will contain updated drops

cat("Total original number of trials:", length(data_set[[1]]))
```

```
## Total original number of trials: 2880
```

```
temp_plot_data <- data_set

accidental_drops_manual <- length((data_set %>% filter(Drop==1))[[1]])
#accidental_drops_total

accidental_drops_auto_detect <- 0 # to count accidental drops detected by the metric-based method

cat("\n# accidental drops counted manually: ", accidental_drops_manual, "*")
```

```
##
## # accidental drops counted manually:  199 *
```

```
# tag above criteria as accidental drops
for (x in 1:length(temp_plot_data$id)){
  #if (temp_plot_data$Drop[x]==0){
    if (temp_plot_data$LandOnTable[x]==FALSE){
      temp_plot_data$Drop[x]<-1
      accidental_drops_auto_detect <- accidental_drops_auto_detect+1
      }
    else if (temp_plot_data$Distance[x] >= 0.2){
      temp_plot_data$Drop[x]<-1
      accidental_drops_auto_detect <- accidental_drops_auto_detect+1}
    else if (temp_plot_data$Distance[x] >= 0.1 & temp_plot_data$Distance[x]<0.2 & temp_plot_data$TimeFr
      temp_plot_data$Drop[x]<-1
```

```r
      accidental_drops_auto_detect <- accidental_drops_auto_detect+1}
  #}
}

cat("\n# accidental drops counted via auto-detect: ", accidental_drops_auto_detect, "*")
```

```
##
## # accidental drops counted via auto-detect:  202 *
```

```r
accidental_drops_total <- length((temp_plot_data %>% filter(Drop==1))[[1]])

cat("\n# accidental drops counted via auto-detect, not counted manually", accidental_drops_total - acci
```

```
##
## # accidental drops counted via auto-detect, not counted manually 34 *
```

```r
cat("\n# accidental drops counted via by both auto-detect and manual methods", accidental_drops_manual
```

```
##
## # accidental drops counted via by both auto-detect and manual methods 168 *
```

```r
cat("\n   * Note: these numbers don't mean much, except to compare with each other")
```

```
##
##    * Note: these numbers don't mean much, except to compare with each other
```

```r
cat("\n# total accidental drops (flagged by either method): ", accidental_drops_total)
```

```
##
## # total accidental drops (flagged by either method):  233
```

```r
# unity_data_drops will serve as the data set from which drop counts will be derived.
unity_data_drops <- temp_plot_data

# now, create clean data set (unity_data_clean)
# remove accidental drops
unity_data_clean <- unity_data_drops %>%
  filter(Drop==0,
         #TimeFromSpawnToGrab > 0.1,
         id %in% include)

# remove subjects, if any
survey_data <- survey_data %>%
  filter(UserID %in% include)

# recalculate sample size, if necessary
sample_size <- length(survey_data$UserID)

# total number of accidental drops
accidental_drops_total <- length((unity_data_drops %>% filter(Drop==1))[[1]])
cat("\n# total accidental drops according to unity_data_drops: ", accidental_drops_total)
```

```
##
## # total accidental drops according to unity_data_drops:  233
```

```
cat("\n# total number of trials remaining after cutting drops: ", length(unity_data_clean[[1]]))
```

```
##
## # total number of trials remaining after cutting drops:  2647
```

```
cat("\n% of original trials flagged and removed as accidental drops: ", round(100*(accidental_drops_tota
```

```
##
## % of original trials flagged and removed as accidental drops:  8.09 %
```

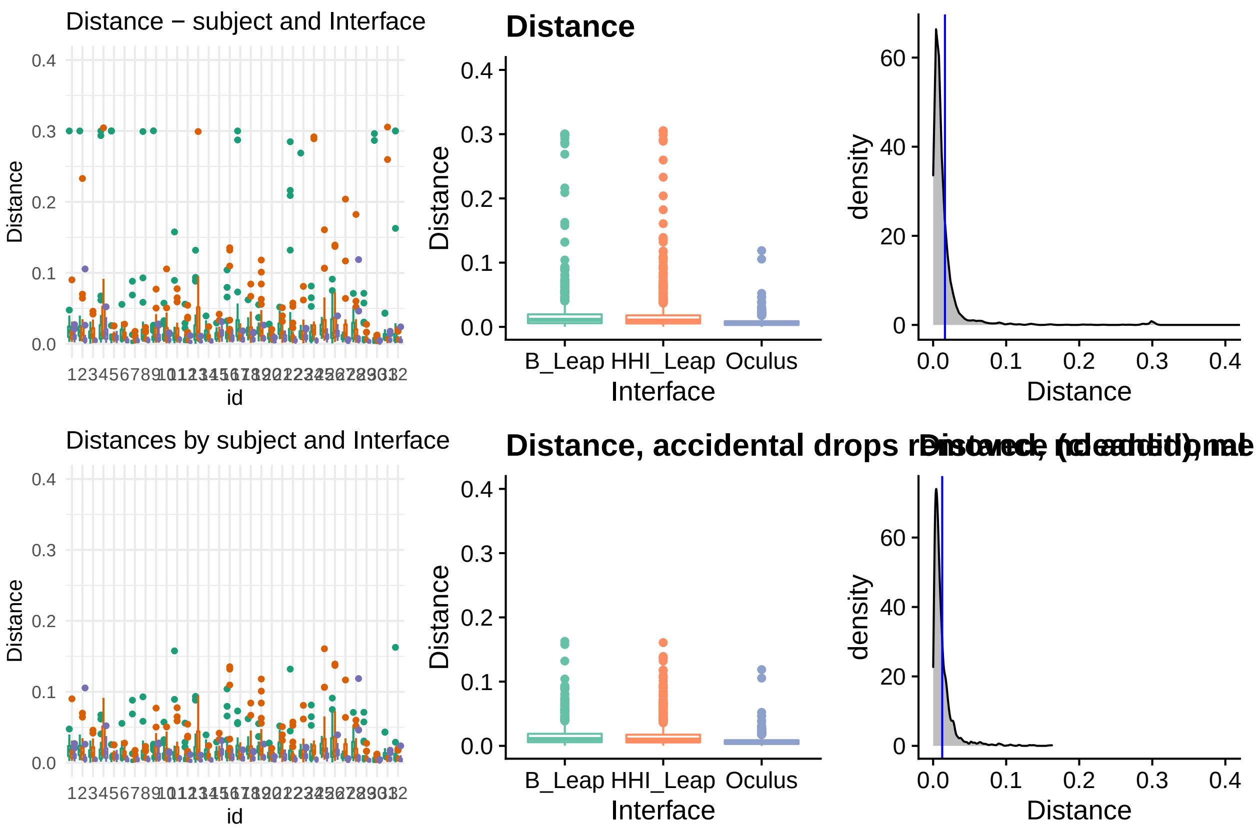
```
# count number of manually detected accidental drops
accidental_drops_manual <- length((data_set %>% filter(Drop==1))[[1]])
accidental_drops_manual.percent <- 100*(accidental_drops_manual/length(data_set[[1]]))

# auto detect drops percent
accidental_drops_auto_detect.percent <- 100*(accidental_drops_auto_detect/length(data_set[[1]]))

# manual, not auto-detected
accidental_drops_manual_only <- accidental_drops_total - accidental_drops_auto_detect

# auto, not manual-detected
accidental_drops_auto_only <- accidental_drops_total - accidental_drops_manual

sink()
```

Visualize cleaning results

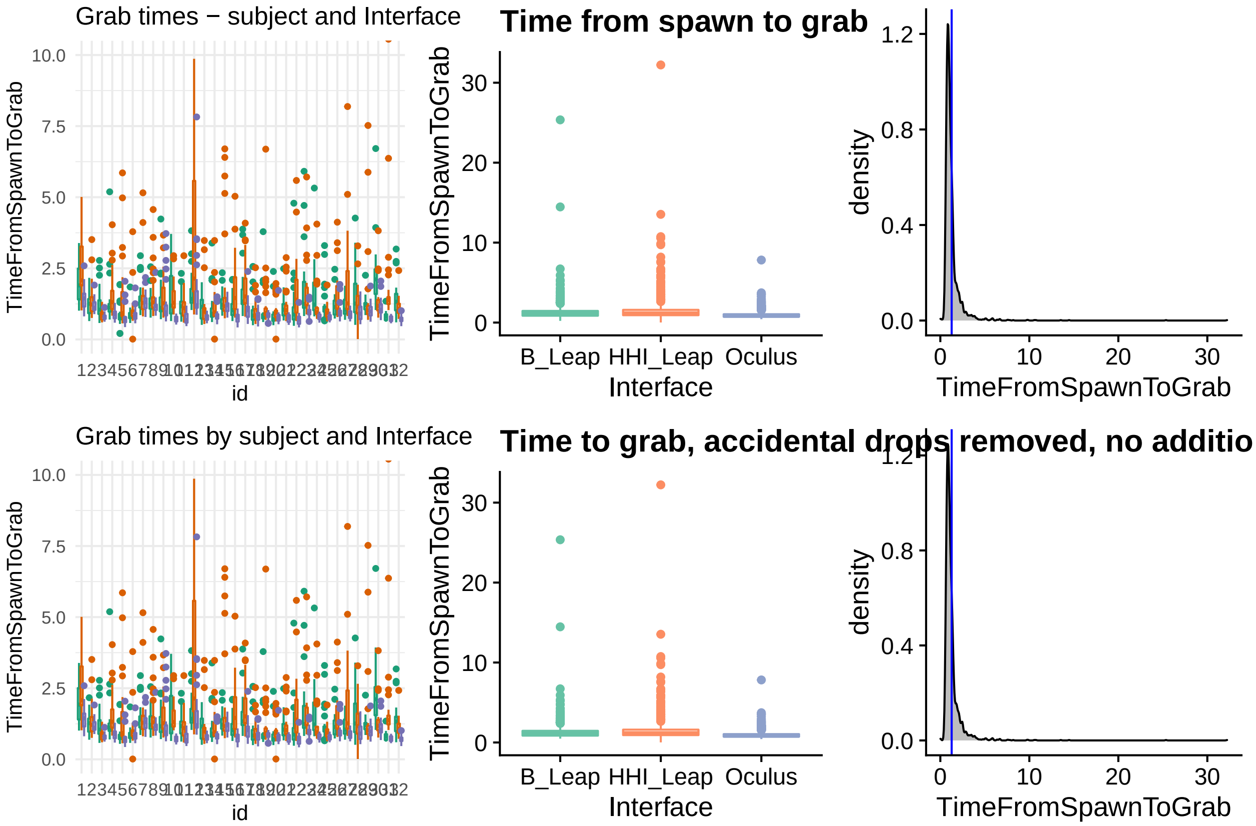
```
# box plot, distance by subject
distance_subjects_clean_box<-  ggplot(unity_data_clean, aes(id, Distance, color=Interface)) +
  theme_minimal() +
  theme(legend.position = "none") +
  geom_boxplot(#outlier.colour="black",
           outlier.size=1, notch=FALSE) +
  scale_color_brewer(palette="Dark2") +
    labs(title="Distances by subject and Interface") +
    coord_cartesian(ylim=c(0, 0.4))

# grab times box plot of subjects
grabtime_subjects_clean_box<-  ggplot(unity_data_clean, aes(id, TimeFromSpawnToGrab, color=Interface)) +
  theme_minimal() +
  theme(legend.position = "none") +
  geom_boxplot(outlier.size=1, notch=FALSE) +
  # geom_point() +
#  geom_violin(scale="area") +
  scale_color_brewer(palette="Dark2") +
    labs(title="Grab times by subject and Interface") +
    coord_cartesian(ylim=c(0, 10))
```

```r
# release times - by subject - box plot
releasetime_subjects_clean_box<- ggplot(unity_data_clean, aes(id, TimeFromGrabToGrabLoss, color=Interfac
theme_minimal() +
theme(legend.position = "bottom") +
geom_boxplot(outlier.size=1, notch=FALSE) +
scale_color_brewer(palette="Dark2") +
  labs(title="Release times by subject and Interface") #+coord_cartesian(ylim=c(0, 0.1))

# total times - by subject - box plot
totaltime_subjects_clean_box<- ggplot(unity_data_clean, aes(id, TimeFromSpawnToGrabLoss, color=Interface
theme_minimal() +
theme(legend.position = "bottom") +
geom_boxplot(outlier.size=1, notch=FALSE) +
scale_color_brewer(palette="Dark2") +
  labs(title="Total times by subject and Interface") #+coord_cartesian(ylim=c(0, 0.1))


# density plot: distance
distance_clean_density<- ggplot(unity_data_clean, aes(Distance))+
  #geom_histogram(binwidth = 0.1)+
  geom_density(fill="Grey")+labs(title=paste0("Distance (cleaned), mean=",round(mean(unity_data_clean$Di
  geom_vline(aes(xintercept = mean(Distance)), color="Blue")+theme(legend.position = "none")+coord_carte

# density plot: spawn to grab
grabtime_clean_density<- ggplot(unity_data_clean, aes(TimeFromSpawnToGrab))+
  #geom_histogram(binwidth = 0.1)+
  geom_density(fill="Grey")+labs(paste0("Grab time (cleaned), mean=",round(mean(unity_data_clean$TimeFr

# density plot: grab to release time
releasetime_clean_density<- ggplot(unity_data_clean, aes(TimeFromGrabToGrabLoss))+
  #geom_histogram(binwidth = 0.1)+
  geom_density(fill="Grey")+labs(paste0("Release time (cleaned), mean=",round(mean(unity_data_clean$Time
  geom_vline(aes(xintercept = mean(TimeFromGrabToGrabLoss)), color="Blue")+theme(legend.position = "none

# density plot: total time
totaltime_clean_density<- ggplot(unity_data_clean, aes(TimeFromSpawnToGrabLoss))+
  #geom_histogram(binwidth = 0.1)+
  geom_density(fill="Grey")+labs(paste0("Total time (cleaned), mean=",round(mean(unity_data_clean$TimeF
  geom_vline(aes(xintercept = mean(TimeFromSpawnToGrabLoss)), color="Blue")#+coord_cartesian(xlim=c(0,1

# box plots
distance_clean_box<- ggplot(unity_data_clean, aes(Interface, Distance, color=Interface))+geom_boxplot()+
  #geom_text(data=data_set.clean %>% filter(Distance > 0.5), aes(Interface, Distance, label=paste0(Dist

grabtime_clean_box<- ggplot(unity_data_clean, aes(Interface, TimeFromSpawnToGrab, color=Interface))+
  geom_boxplot()+scale_color_brewer(palette="Set2")+ggtitle(label="Time to grab, accidental drops remove

releasetime_clean_box<- ggplot(unity_data_clean, aes(Interface, TimeFromGrabToGrabLoss, color=Interface)
  geom_boxplot()+scale_color_brewer(palette="Set2")+ggtitle(label="Time from grab to release, accidental

totaltime_clean_box<- ggplot(unity_data_clean, aes(Interface, TimeFromSpawnToGrabLoss, color=Interface)
  geom_boxplot()+scale_color_brewer(palette="Set2")+ggtitle(label="Total time, accidental drops removed
```

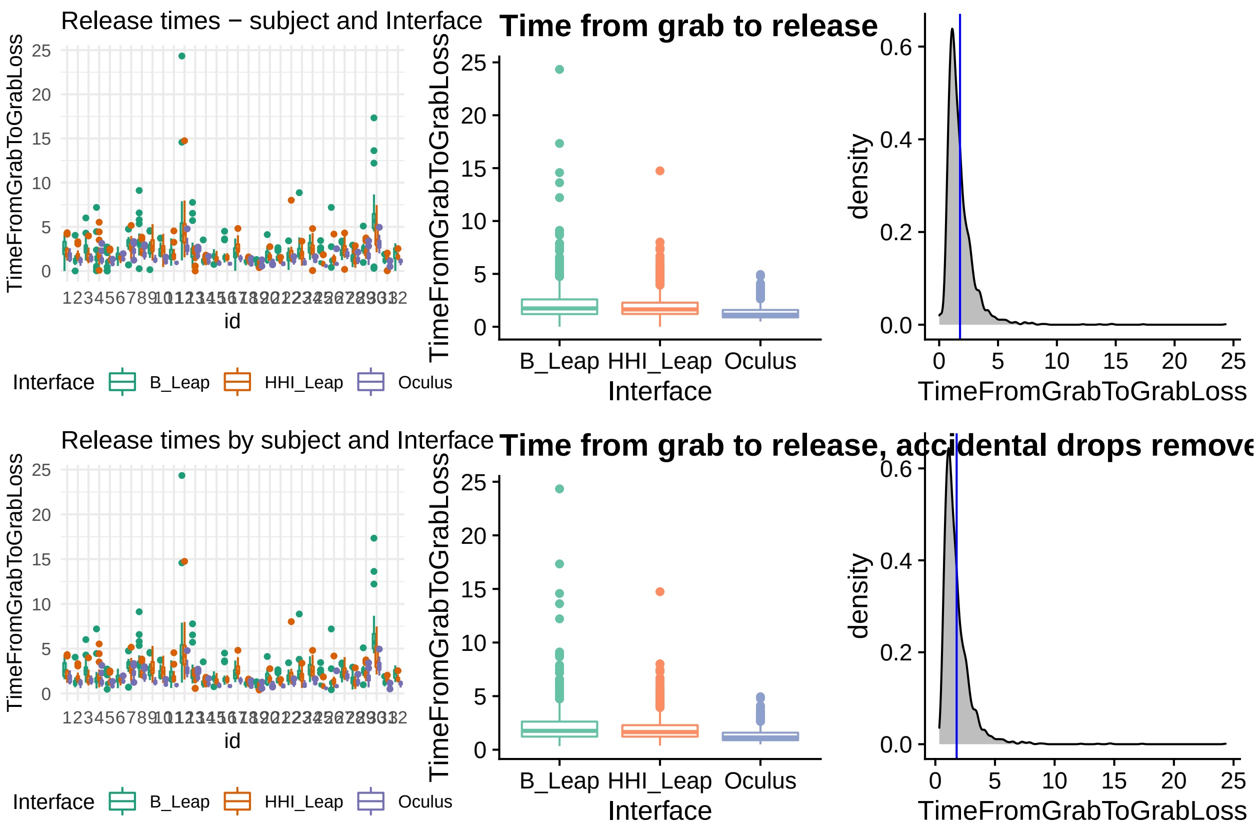
```r
# distance
plot_grid(distance_subjects_box, distance_box, distance_density, distance_subjects_clean_box, distance_c
```



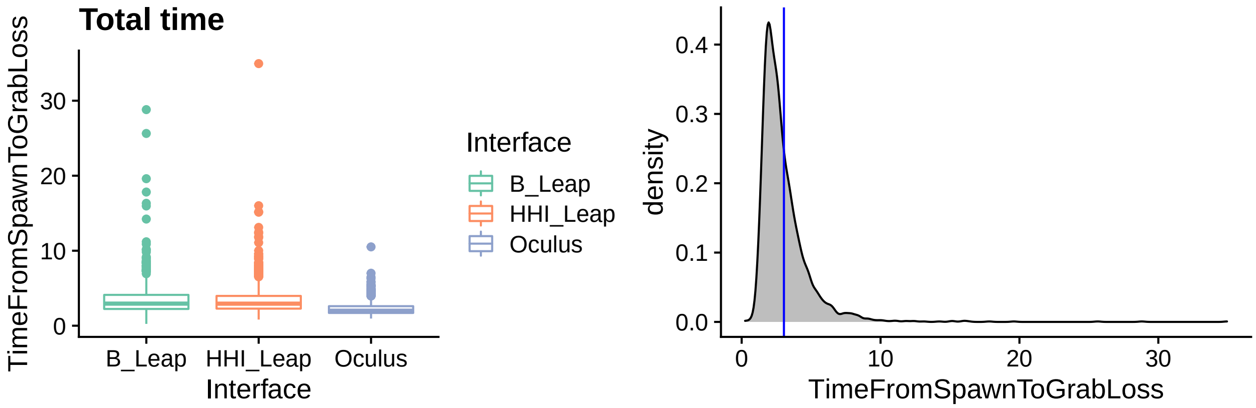

```r
# grab time
plot_grid(grabtime_subjects_box, grabtime_box, grabtime_density, grabtime_subjects_clean_box, grabtime_c
```

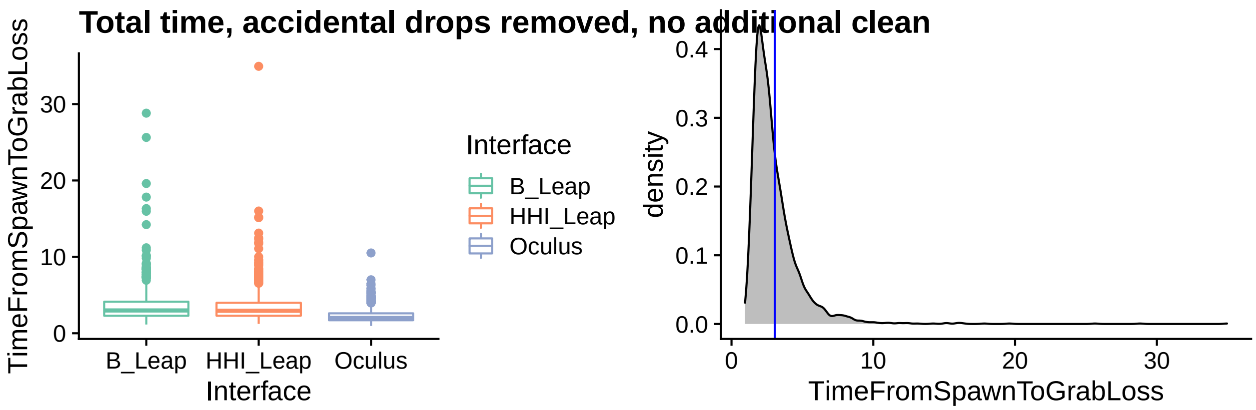



```
# release time
plot_grid(releasetime_subjects_box, releasetime_box, releasetime_density, releasetime_subjects_clean_bo
```




```
# total time
plot_grid(totaltime_box, totaltime_density, totaltime_clean_box, totaltime_clean_density)
```





Visual inspection of the density plots suggests that the density (and related measures such as the median) converge between the 3 Interfaces. The grand mean (grey line) gets noticably lower after the clean.

Data sets for further analysis

These comipilations will contain the *subject means*, not the individual trials.

Cube size data

Long data format divided into cube size.

```
#cube size
# calculate means and sd's for cube size by Interface & id
# so it's now 6 means for each subject -- s3 dist leap small... etc...
# add cube size to data set

# means for cube size by Interface
  subject_data_cube_size <- unity_data_clean %>% #subject_data_cube_size %>%
#    left_join(unity_data_clean %>%
    group_by(id, Interface, Cube_Size) %>%
    summarise(Distance=mean(Distance),
              grabtime=mean(TimeFromSpawnToGrab),
              releasetime=mean(TimeFromGrabToGrabLoss),
              totaltime=mean(TimeFromSpawnToGrabLoss))#,by=c("id", "Interface", "Cube_Size"))
```

```
  # add accidental drop counts at cube level to data set
  subject_data_cube_size <- subject_data_cube_size %>%
    left_join(unity_data_drops %>%
                group_by(id,Interface,Cube_Size) %>%
                summarise(Drop_Count=sum(Drop)))

  # accidental drop rate
    subject_data_cube_size <- subject_data_cube_size %>%
    mutate(Drop_Rate=(Drop_Count/10)*100)

# set factor levels
subject_data_cube_size <- subject_data_cube_size %>% ungroup() %>%
  mutate(Cube_Size=factor(Cube_Size, levels=c("Small","Medium","Large")),
                        Interface=factor(Interface, levels=plot_order))
```

Wide format

Wide format; one line per subject.

```
# compile wide data set and calculate means where necessary starting w/ wide
# because some metrics (i.e. subjective questions and demographics) are already
# in wide format

# start by taking from survey_data (already contains demos and subj. q's)
subject_data_all_wide <- survey_data %>% filter(UserID %in% include) %>% select(id = UserID,
    Gender:Disabilities, Hand, InterfaceOrder, OculusExpComment:PrefCondition, -contains("SUS"),
    -contains("comment"))

# leap groups (for Interface order analysis)
leap_first <- which(subject_data_all_wide$InterfaceOrder == "LOH" | subject_data_all_wide$InterfaceOrde
    "LHO" | subject_data_all_wide$InterfaceOrder == "OLH")
HHI_Leap_first <- which(subject_data_all_wide$InterfaceOrder == "OHL" | subject_data_all_wide$Interface
    "HOL" | subject_data_all_wide$InterfaceOrder == "HLO")
# add group designation to subject_data_all_wide
subject_data_all_wide[leap_first, "Leap_Group"] <- "B_Leap_first"
subject_data_all_wide[HHI_Leap_first, "Leap_Group"] <- "HHI_Leap_first"
subject_data_all_wide$Leap_Group <- factor(subject_data_all_wide$Leap_Group)

# oculus groups (for Interface order analysis)
Oculus_first <- which(subject_data_all_wide$InterfaceOrder == "OLH" | subject_data_all_wide$InterfaceOr
    "OHL")
Oculus_last <- which(subject_data_all_wide$InterfaceOrder == "LHO" | subject_data_all_wide$InterfaceOrd
    "HLO")
subject_data_all_wide[Oculus_first, "Oculus_Group"] <- "Oculus_first"
subject_data_all_wide[Oculus_last, "Oculus_Group"] <- "Oculus_last"
subject_data_all_wide$Oculus_Group <- factor(subject_data_all_wide$Oculus_Group)

# add error totals (errors previously calculated -- error_totals)
subject_data_all_wide <- subject_data_all_wide %>% left_join(data_set %>% filter(id %in%
    include, Interface == "B_Leap") %>% group_by(id) %>% summarise(errors_Leap = sum(Drop)),
    by = "id") %>% left_join(data_set %>% filter(id %in% include, Interface == "HHI_Leap") %>%
    group_by(id) %>% summarise(errors_Leap_HHI = sum(Drop)), by = "id") %>% left_join(data_set %>%
    filter(id %in% include, Interface == "Oculus") %>% group_by(id) %>% summarise(errors_Oculus = sum(D
```

```r
    by = "id")

# add sus scores
subject_data_all_wide <- subject_data_all_wide %>% left_join(SUS_scores %>% rename(id = UserID,
    SUS_Leap = B_Leap, SUS_Leap_HHI = HHI_Leap, SUS_Oculus = Oculus), by = "id")

# distance
subject_data_all_wide <- subject_data_all_wide %>% left_join(unity_data_clean %>%
    group_by(id, Interface) %>% summarise(Mean = mean(Distance)) %>% ungroup(id) %>%
    spread(Interface, Mean) %>% rename(distance_Leap_mean = B_Leap, distance_Leap_HHI_mean = HHI_Leap,
    distance_Oculus_mean = Oculus), by = "id")

# grab time
subject_data_all_wide <- subject_data_all_wide %>% left_join(unity_data_clean %>%
    group_by(id, Interface) %>% summarise(Mean = mean(TimeFromSpawnToGrab)) %>% ungroup(id) %>%
    spread(Interface, Mean) %>% rename(grabtime_Leap_mean = B_Leap, grabtime_Leap_HHI_mean = HHI_Leap,
    grabtime_Oculus_mean = Oculus), by = "id")

# release time add to data set
subject_data_all_wide <- subject_data_all_wide %>% left_join(unity_data_clean %>%
    group_by(id, Interface) %>% summarise(Mean = mean(TimeFromGrabToGrabLoss)) %>%
    ungroup(id) %>% spread(Interface, Mean) %>% rename(releasetime_Leap_mean = B_Leap,
    releasetime_Leap_HHI_mean = HHI_Leap, releasetime_Oculus_mean = Oculus), by = "id")

# total time means
subject_data_all_wide <- subject_data_all_wide %>% left_join(unity_data_clean %>%
    group_by(id, Interface) %>% summarise(Mean = mean(TimeFromSpawnToGrabLoss)) %>%
    ungroup(id) %>% spread(Interface, Mean) %>% rename(totaltime_Leap_mean = B_Leap,
    totaltime_Leap_HHI_mean = HHI_Leap, totaltime_Oculus_mean = Oculus), by = "id")

# practice time calculate practice time (later correlate w/ performance) =
# total_time recorded - sum of time from spawn to grab
subject_data_all_wide <- subject_data_all_wide %>% left_join(data_set %>% select(id,
    Interface, TimeForTrainingPhase) %>% group_by(id, Interface) %>% distinct(.) %>%
    spread(Interface, TimeForTrainingPhase) %>% rename(TrainingTime_Leap = B_Leap,
    TrainingTime_HHI = HHI_Leap, TrainingTime_Oculus = Oculus), by = "id")
```

Long format

This one is good for ggplots and ANOVA.

```r
# start a long version for statistical testing-ready data
subject_data_all_long <- subject_data_all_wide %>%
  select(id, InterfaceOrder, Leap_Group, Oculus_Group)

# accidental drops
subject_data_all_long <- subject_data_all_long %>%
  left_join(unity_data_drops %>%
              filter(id %in% include) %>%
              group_by(id,Interface) %>%
              summarise(Drop_Count=sum(Drop)))

# percentage
```

```r
subject_data_all_long <- subject_data_all_long %>%
  mutate(Drop_Rate=(Drop_Count/30)*100)

# SUS
subject_data_all_long <- subject_data_all_long %>%
  left_join(SUS_scores %>%
              gather(key="Interface", value="SUS", "B_Leap","HHI_Leap","Oculus") %>%
              rename(id=UserID) %>%
              filter(id %in% include),
            by=c("id","Interface"))

# Distance means
subject_data_all_long <- subject_data_all_long %>%
  left_join(unity_data_clean %>%
    group_by(id, Interface) %>%
    summarise(Distance=mean(Distance)) %>%
    ungroup(id), by=c("id","Interface"))

# Grab Time means
subject_data_all_long <- subject_data_all_long %>%
  left_join(unity_data_clean %>%
    group_by(id, Interface) %>%
    summarise(grabtime=mean(TimeFromSpawnToGrab)) %>%
    ungroup(id), by=c("id","Interface"))

# Release time means
subject_data_all_long <- subject_data_all_long %>%
  left_join(unity_data_clean %>%
    group_by(id, Interface) %>%
    summarise(releasetime=mean(TimeFromGrabToGrabLoss)) %>%
    ungroup(id), by=c("id","Interface"))

# Total time means
subject_data_all_long <- subject_data_all_long %>%
  left_join(unity_data_clean %>%
    group_by(id, Interface) %>%
    summarise(totaltime=mean(TimeFromSpawnToGrabLoss)) %>%
    ungroup(id), by=c("id","Interface"))

# training time
subject_data_all_long <- subject_data_all_long %>%
  left_join(unity_data_clean %>%
              select(id, Interface, practice_time=TimeForTrainingPhase) %>%
              group_by(id, Interface) %>%
              distinct(.), by=c("id","Interface"))

# find SD from grand mean to detect outliers, per Interface
find_z <- function(group_scores, score) {
  group_sd <- sd(group_scores)
  group_mean <- mean(group_scores)
  z <-
  z
}
```

```r
# Subjective questions
for (x in 1:8){
  # make a temp data set of question x totals
  temp_question_totals <- subject_data_all_wide %>%
    filter(id %in% include)%>%
    select(id, contains(as.character(x))) %>%
    rename(B_Leap=contains("Standard"),HHI_Leap=contains("HHI"),
           Oculus=contains("Oculus")) %>%
    gather(key=Interface,value=Score,c(2:4))
  # rename vars
  names(temp_question_totals)[3] <- paste0("Q_",x,"_Score")
  # names(temp_question_totals)[4] <- paste0("Q_",x,"_Rank")
  # add to big data set
  subject_data_all_long <- subject_data_all_long %>%
  left_join(temp_question_totals, by=c("id","Interface"))
}

# agency
subject_data_all_long <- subject_data_all_long %>%
  left_join(subject_data_all_wide %>%
              select(id, contains("Agency")) %>%
              rename(B_Leap=contains("Stan"), HHI_Leap=contains("HHI"), Oculus=contains("Oculus")) %>%
              gather(key="Interface", value="agency", c(2:4)) %>%
              group_by(id),# %>% mutate(agency_Rank=dense_rank(desc(agency))),
            by=c("id","Interface"))

# overall satisfaction
subject_data_all_long <- subject_data_all_long %>%
  left_join(subject_data_all_wide %>%
              select(id, contains("KunKey")) %>%
              rename(B_Leap=contains("Stan"), HHI_Leap=contains("HHI"), Oculus=contains("Oculus")) %>%
              gather(key="Interface", value="satisfaction", c(2:4))%>%
              group_by(id),#%>% mutate(satisfaction_Rank=dense_rank(desc(satisfaction))),
            by=c("id","Interface"))

# overall preferred condition
  subject_data_all_long <- subject_data_all_long %>%
    left_join(subject_data_all_wide %>%
                select(id, contains("PrefCondition")),
              by="id") %>%
    mutate(PrefCondition=factor(PrefCondition)) %>%
    mutate(PrefCondition=recode_factor(PrefCondition,
     "ohne Controller, Variante 1 (Standard Leap Motion)"="B_Leap",
              "mit Controller"="Oculus",
              "ohne Controller, Variante 2 (Leap Motion mit HHI-Anpassungen)"="HHI_Leap"))

# demographics
subject_data_all_long <- subject_data_all_long %>%
left_join(subject_data_all_wide %>%
            select(id, Age, Height, Arm),
          by="id")

# experience
```

```r
  # w/ video game controllers
  subject_data_all_long <- subject_data_all_long %>%
    left_join(subject_data_all_wide %>%
                select(id, contains("SkillController")),
              by="id")

  # w/ VR
  subject_data_all_long <- subject_data_all_long %>%
  left_join(subject_data_all_wide %>%
              select(id, contains("SkillVR")),
            by="id")

  # w/ any game
  subject_data_all_long <- subject_data_all_long %>%
  left_join(subject_data_all_wide %>%
              select(id, contains("SkillGames")),
            by="id")


# reorder factors
  subject_data_all_long <- subject_data_all_long %>%
    mutate(Interface=factor(Interface, levels=plot_order))
```

Outlier classification: Z-scores

Note: SD's are calculated *per interface (Interface)*, not overall.

```r
z_flag <- 3

sink("outlier_report.csv")

cat("Flagging number of data points over this many z-scores: ", z_flag, "\n\n")
```

```
## Flagging number of data points over this many z-scores:  3
```

```r
z_accuracy <- subject_data_all_long %>% select(id, Interface, Distance) %>% group_by(Interface) %>%
    mutate(z = (Distance - mean(Distance))/sd(Distance), flag = z > z_flag | z <
        -1 * z_flag, flag_level = z_flag)
cat("Accuracy")
```

```
## Accuracy
```

```r
table(z_accuracy$flag)
```

```
##
## FALSE
##    96
```

```r
z_grabtime <- subject_data_all_long %>% select(id, Interface, grabtime) %>% group_by(Interface) %>%
    mutate(z = (grabtime - mean(grabtime))/sd(grabtime), flag = z > z_flag | z <
        -1 * z_flag, flag_level = z_flag)
cat("Grab time")
```

```
## Grab time
```

```r
table(z_grabtime$flag)
```

```
##
## FALSE  TRUE
##    94     2
```

```r
z_releasetime <- subject_data_all_long %>% select(id, Interface, releasetime) %>%
    group_by(Interface) %>% mutate(z = (releasetime - mean(releasetime))/sd(releasetime),
    flag = z > z_flag | z < -1 * z_flag, flag_level = z_flag)
cat("Release time")
```

```
## Release time
```

```r
table(z_releasetime$flag)
```

```
##
## FALSE  TRUE
##    93     3
```

```r
z_totaltime <- subject_data_all_long %>% select(id, Interface, totaltime) %>% group_by(Interface) %>%
    mutate(z = (totaltime - mean(totaltime))/sd(totaltime), flag = z > z_flag | z <
        -1 * z_flag, flag_level = z_flag)
cat("Total time")
```

```
## Total time
```

```r
table(z_totaltime$flag)
```

```
##
## FALSE  TRUE
##    94     2
```

```r
z_drops <- subject_data_all_long %>% select(id, Interface, Drop_Count) %>% group_by(Interface) %>%
    mutate(z = (Drop_Count - mean(Drop_Count))/sd(Drop_Count), flag = z > z_flag |
        z < -1 * z_flag, flag_level = z_flag)
cat("Accidental drops")
```

```
## Accidental drops
```

```r
table(z_drops$flag)
```

```
##
## FALSE
##    96
```

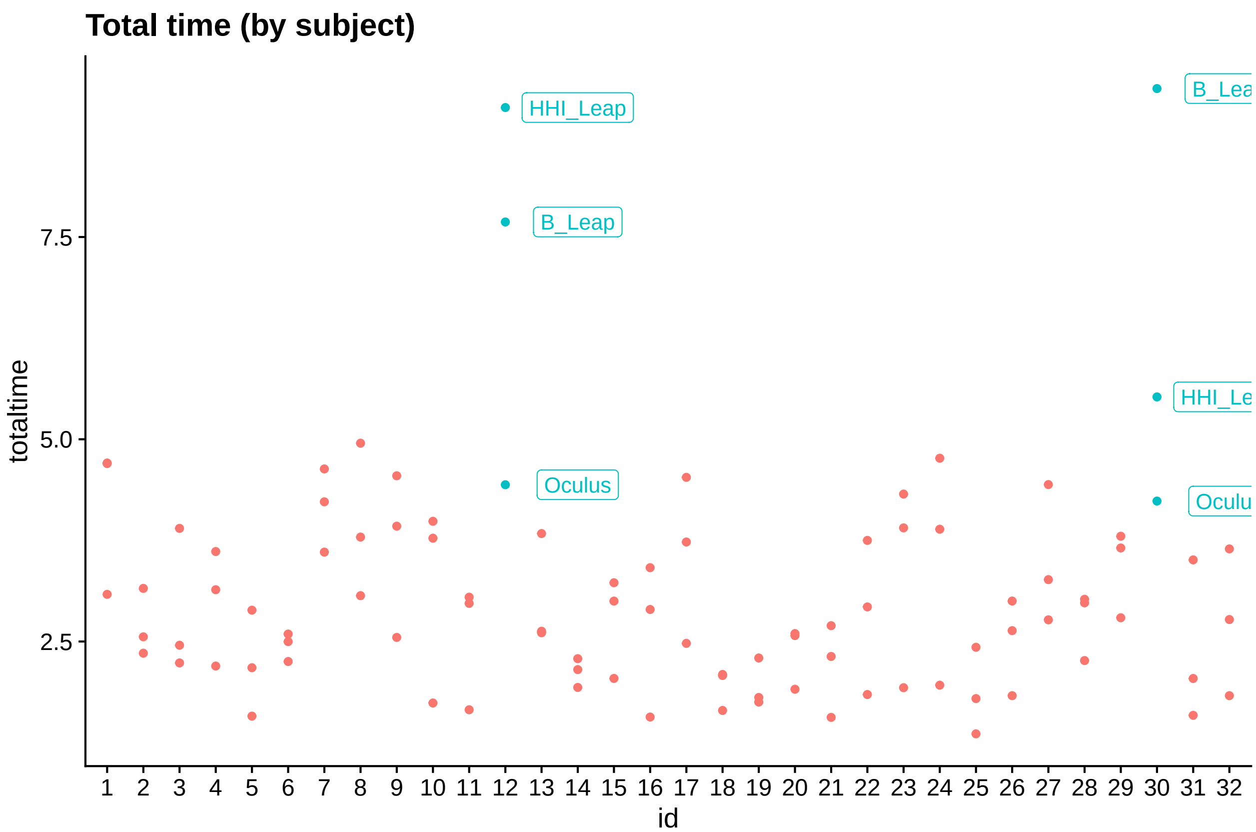
```
z_practice_time <- subject_data_all_long %>% select(id, Interface, practice_time) %>%
    group_by(Interface) %>% mutate(z = (practice_time - mean(practice_time))/sd(practice_time),
    flag = z > z_flag | z < -1 * z_flag, flag_level = z_flag)
cat("Practice time")
```

```
## Practice time
```

```
table(z_practice_time$flag)
```

```
##
## FALSE
##    96
```

```
sink()

ggplot(subject_data_all_long, aes(id, totaltime, color = id %in% c(12, 30))) + geom_point() +
    theme(legend.position = "none") + geom_label(data = subject_data_all_long %>%
    filter(id %in% c(12, 30)), aes(label = Interface), nudge_x = 2) + labs(title = "Total time (by subje
    subtite = "Subjects 12 and 30 highlighted")
```



Analysis

Demographics

```
# gender
gender <- table(subject_data_all_wide$Gender)

# handedness
handedness <- table(subject_data_all_wide$Hand)

demographics <- list(descriptives = subject_data_all_wide %>% get_summary_stats(Age,
    Height, Arm, SkillController, SkillVR, SkillGames) %>% select(variable, n, mean,
    sd, max, min, median, iqr))
cat("\nGender")
```

```
##
## Gender
```

```
gender
```

```
##
##   Male    N/A Female
##     23      1      8
```

```
cat("\nHandedness")
```

```
##
## Handedness
```

```
handedness
```

```
##
##  Left Right
##     1    31
```

```
demographics$gender <- gender
demographics$handedness <- handedness
demographics
```

```
## $descriptives
## # A tibble: 6 x 8
##   variable            n  mean    sd   max   min median   iqr
##   <chr>           <dbl> <dbl> <dbl> <dbl> <dbl>  <dbl> <dbl>
## 1 Age                32  27.8  3.37    36    22   27.5  5.25
## 2 Arm                31  78.2  5.74    89    63   79    6
## 3 Height             32 178.   8.88   194   159  179   12
## 4 SkillController    32   3    1.05     5     2    3    2
## 5 SkillGames         32   3.72 1.35     5     1    4    2.25
## 6 SkillVR            32   2.25 1.14     5     1    2    2
```

```
##
## $gender
##
##   Male    N/A Female
##     23      1      8
##
## $handedness
##
##  Left Right
##     1    31
```

```r
cat("Demographics\n\n")
```

```
## Demographics
```

```r
print(as.data.frame(demographics))
```

```
##   descriptives.variable descriptives.n descriptives.mean descriptives.sd
## 1                   Age             32            27.812           3.374
## 2                   Arm             31            78.194           5.735
## 3                Height             32           177.750           8.875
## 4       SkillController             32             3.000           1.047
## 5            SkillGames             32             3.719           1.350
## 6               SkillVR             32             2.250           1.136
##   descriptives.max descriptives.min descriptives.median descriptives.iqr
## 1               36               22                27.5             5.25
## 2               89               63                79.0             6.00
## 3              194              159               179.0            12.00
## 4                5                2                 3.0             2.00
## 5                5                1                 4.0             2.25
## 6                5                1                 2.0             2.00
##   gender.Var1 gender.Freq handedness.Var1 handedness.Freq
## 1        Male          23            Left               1
## 2         N/A           1           Right              31
## 3      Female           8            Left               1
## 4        Male          23           Right              31
## 5         N/A           1            Left               1
## 6      Female           8           Right              31
```

```r
cat("\nGender")
```

```
##
## Gender
```

```r
print(gender)
```

```
##
##   Male    N/A Female
##     23      1      8
```

```r
cat("\nHandedness\n")
```

```
##
## Handedness
```

```r
print(table(subject_data_all_wide$Hand))
```

```
##
##  Left Right
##     1    31
```

```r
cat("\nInterface Order")
```

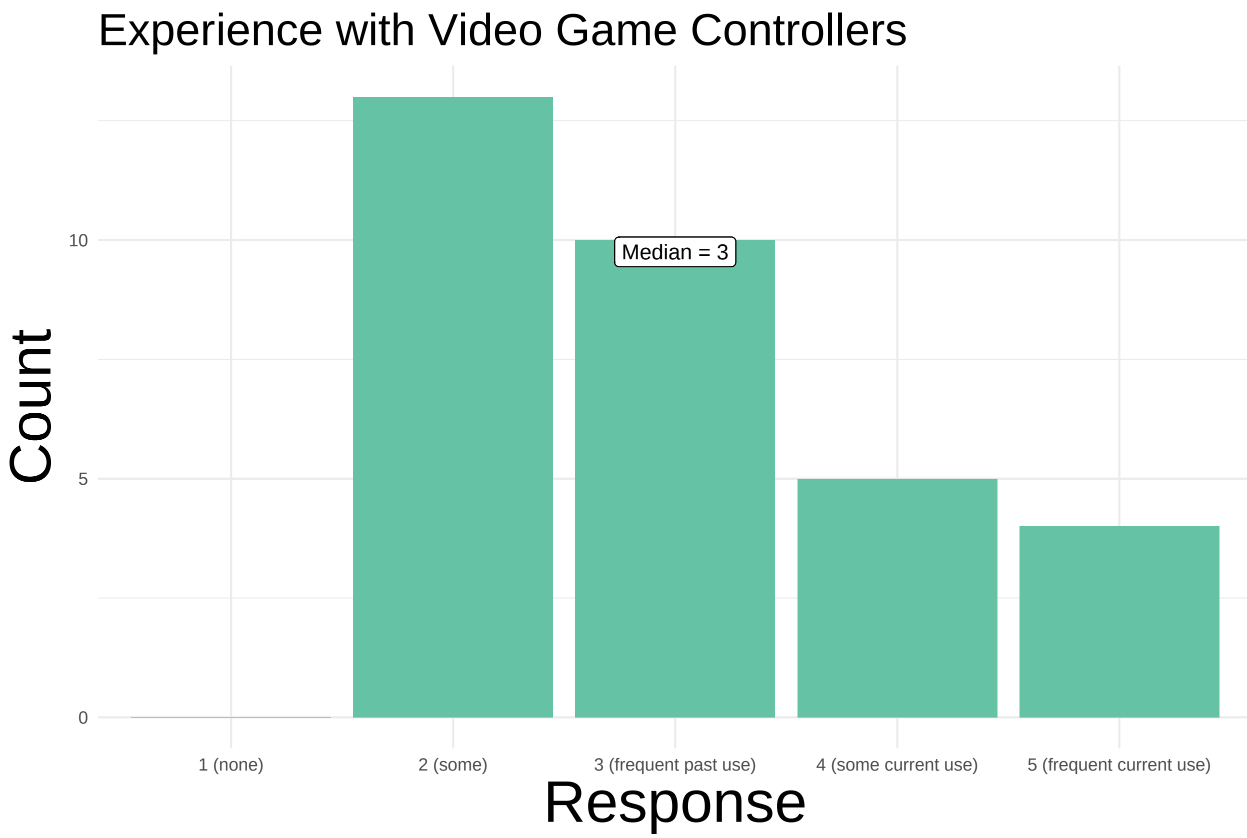
```
##
## Interface Order
```

```r
print(table(subject_data_all_wide$InterfaceOrder))
```

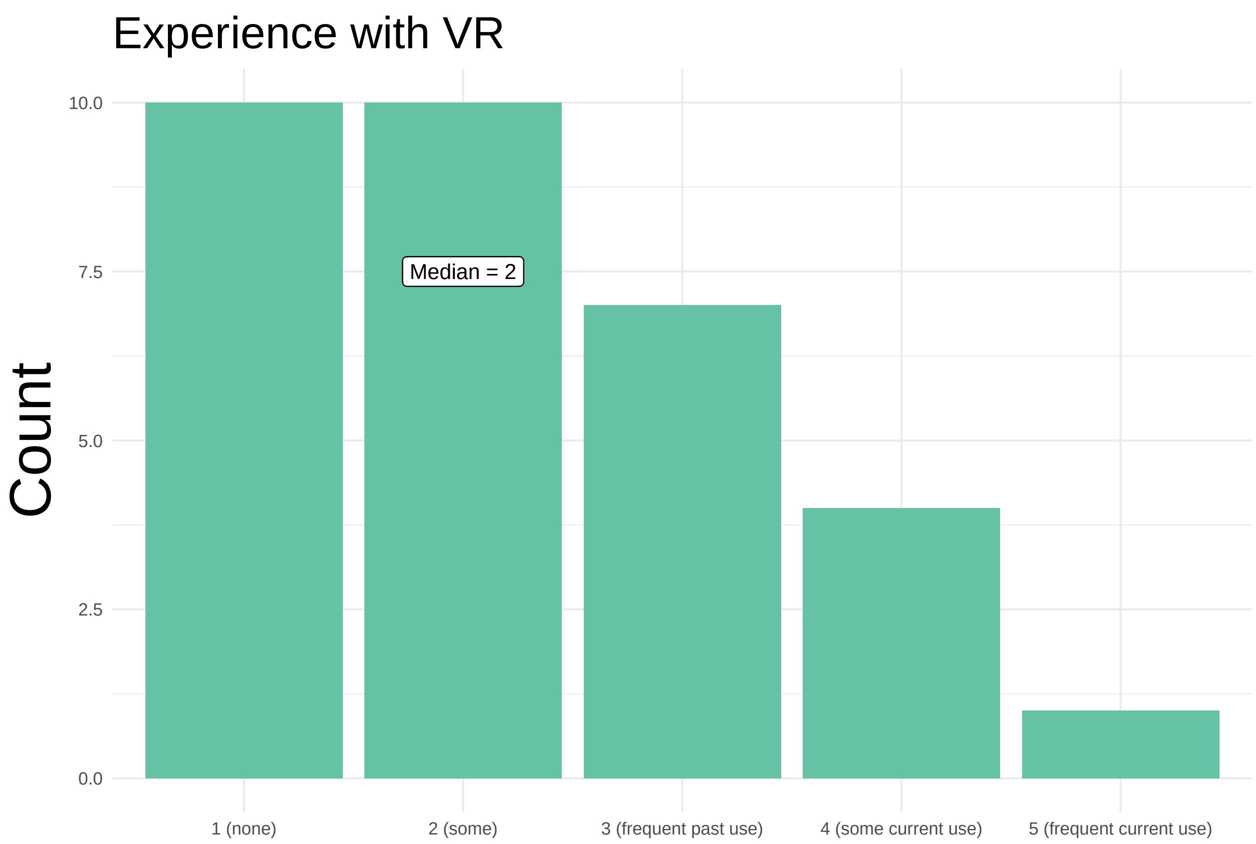
```
##
## LHO LOH HLO HOL OLH OHL
##   5   6   7   3   6   5
```

Demographics Plots

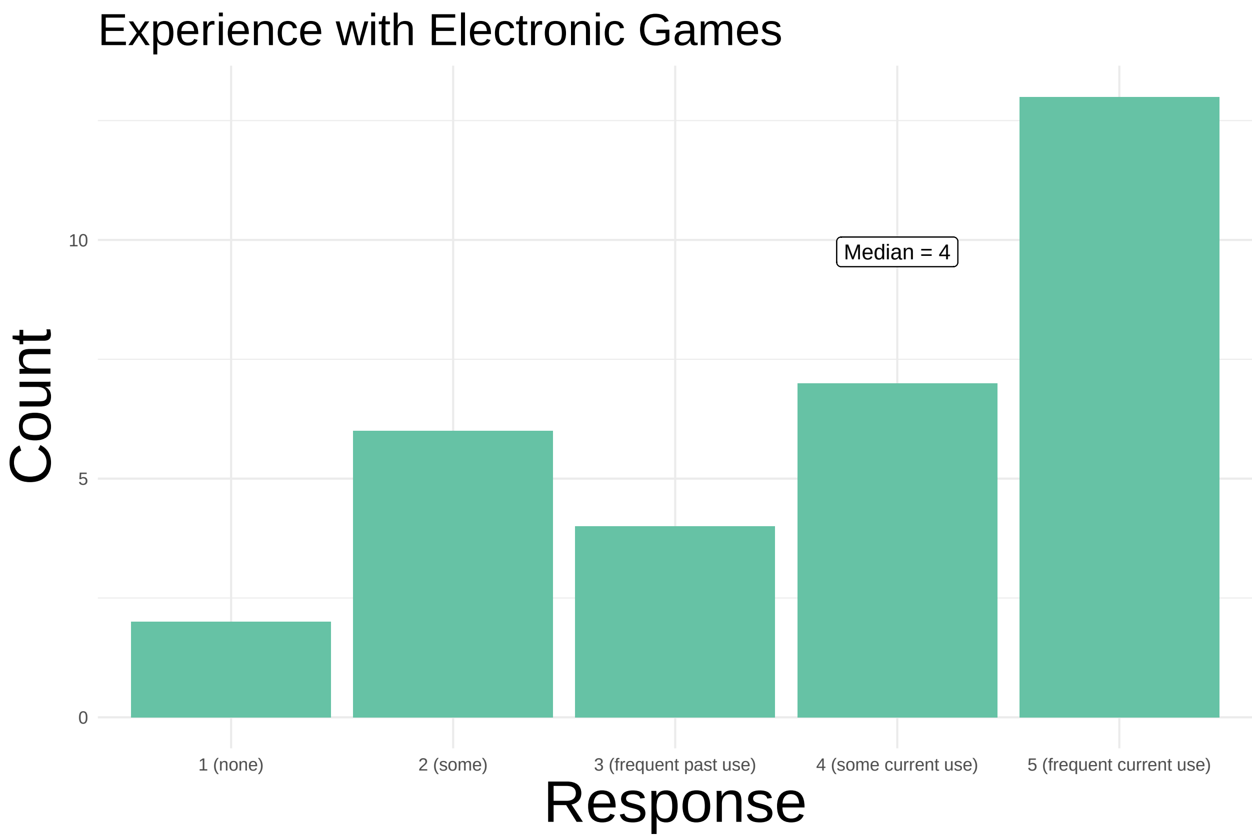
```r
# experience controller make temp plot dataset
temp_plot_data <- data.frame(Response = c(1:5)) %>% mutate(Response = factor(Response)) %>%
    left_join(data.frame(table(subject_data_all_wide$SkillController)) %>% rename(Response = Var1,
        Count = Freq), by = c("Response")) %>% mutate(Response = factor(Response),
    Count = replace_na(Count, 0))
exp_counts <- temp_plot_data %>% rename(controller_count = Count)  #build a data set of counts for late
# bar chart
p <- ggplot(temp_plot_data, aes(x = Response, y = Count)) + geom_bar(stat = "identity",
    fill = brewer.pal(n = 8, name = "Set2")[1]) + theme_minimal() + theme(plot.title = element_text(size
    0.75), axis.title = element_text(size = axis_text_size)) + geom_label(aes(x = median(subject_data_al
    label = paste0("Median = ", median(subject_data_all_long$SkillController)), y = 0.75 *
        max(Count))) + labs(title = "Experience with Video Game Controllers") + scale_x_discrete(labels
    "2 (some)", "3 (frequent past use)", "4 (some current use)", "5 (frequent current use)"))
p
```



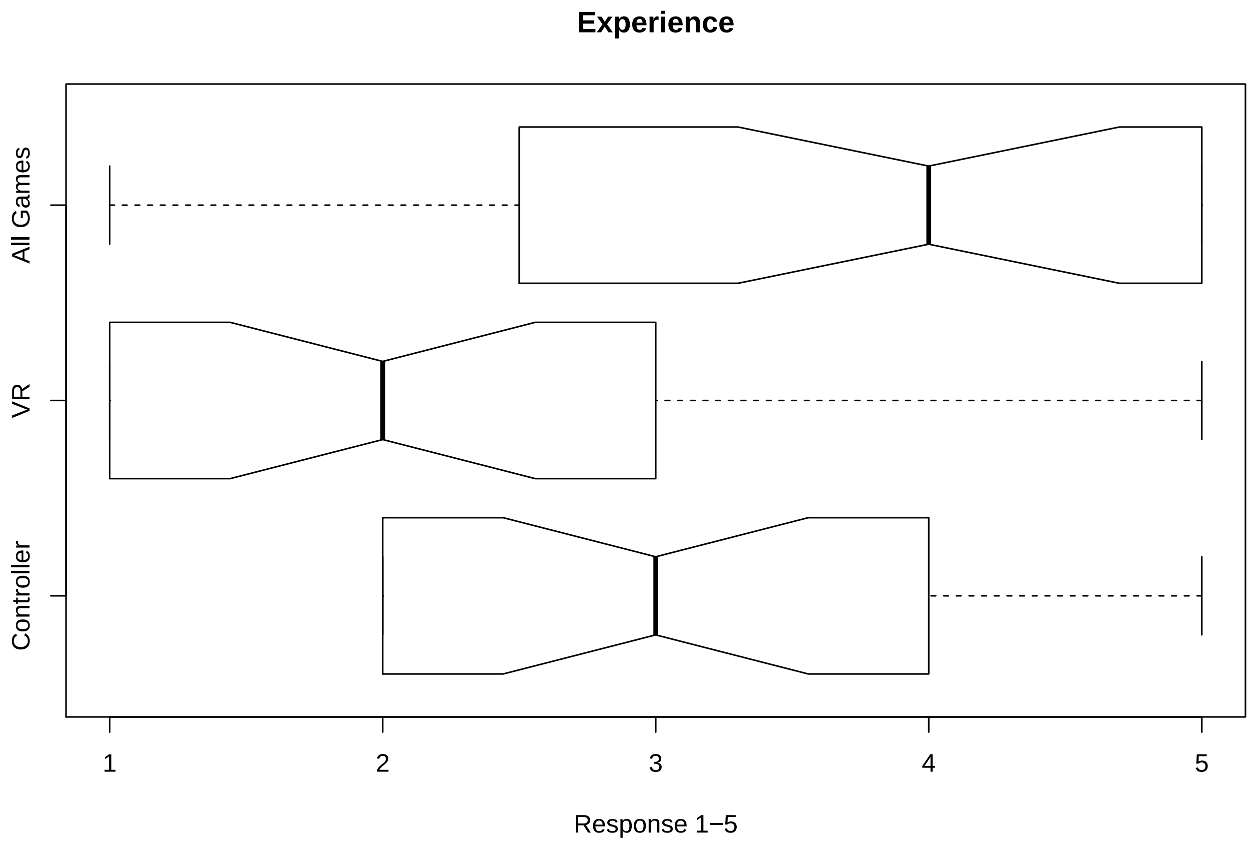

```r
ggsave(p, file = "experience_controller.jpg")


# VR
temp_plot_data <- data.frame(Response = c(1:5)) %>% mutate(Response = factor(Response)) %>%
    left_join(data.frame(table(subject_data_all_wide$SkillVR)) %>% rename(Response = Var1,
        Count = Freq), by = c("Response")) %>% mutate(Response = factor(Response),
    Count = replace_na(Count, 0))
exp_counts <- exp_counts %>% left_join(temp_plot_data %>% rename(vr_count = Count),
    by = "Response")
# bar chart
p <- ggplot(temp_plot_data, aes(x = Response, y = Count)) + geom_bar(stat = "identity",
    fill = brewer.pal(n = 8, name = "Set2")[1]) + geom_label(aes(x = median(subject_data_all_long$SkillV
    label = paste0("Median = ", median(subject_data_all_long$SkillVR)), y = 0.75 *
        max(Count))) + # scale_color_brewer(palette='Dark2')+ scale_fill_brewer(palette='Set3') +
theme_minimal() + xlab(NULL) + theme(plot.title = element_text(size = title_size *
    0.75), axis.title = element_text(size = axis_text_size)) + labs(title = "Experience with VR") +
    scale_x_discrete(labels = c("1 (none)", "2 (some)", "3 (frequent past use)",
        "4 (some current use)", "5 (frequent current use)"))
p
```



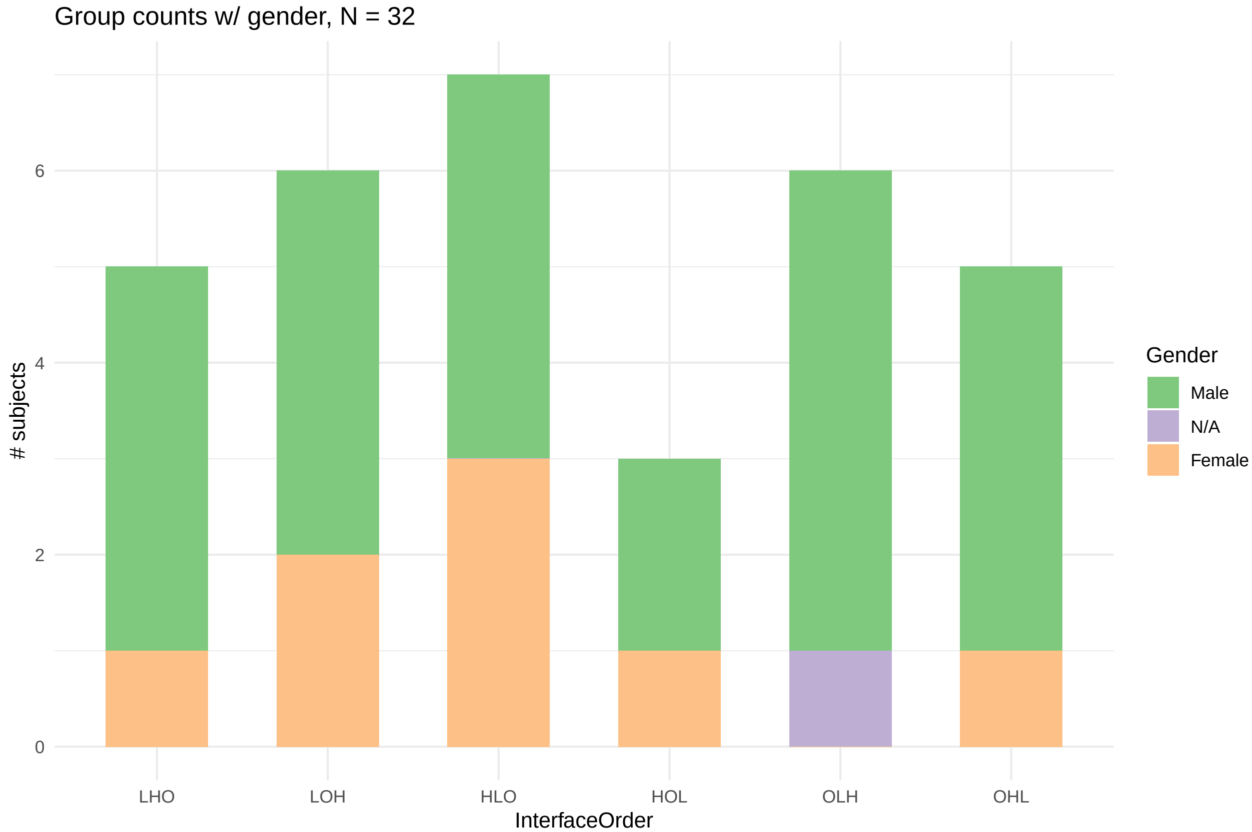

```
ggsave(p, file = "experience_vr.jpg")


# all games
temp_plot_data <- data.frame(Response = c(1:5)) %>% mutate(Response = factor(Response)) %>%
    left_join(data.frame(table(subject_data_all_wide$SkillGames)) %>% rename(Response = Var1,
        Count = Freq), by = c("Response")) %>% mutate(Response = factor(Response),
    Count = replace_na(Count, 0))
exp_counts <- exp_counts %>% left_join(temp_plot_data %>% rename(allgames_count = Count),
    by = "Response")
# bar chart
p <- ggplot(temp_plot_data, aes(x = Response, y = Count)) + geom_bar(stat = "identity",
    fill = brewer.pal(n = 8, name = "Set2")[1]) + theme_minimal() + theme(plot.title = element_text(size
    0.75), axis.title = element_text(size = axis_text_size)) + geom_label(aes(x = median(subject_data_al
    label = paste0("Median = ", median(subject_data_all_long$SkillGames)), y = 0.75 *
        max(Count))) + labs(title = "Experience with Electronic Games") + scale_x_discrete(labels = c("
    "2 (some)", "3 (frequent past use)", "4 (some current use)", "5 (frequent current use)"))
p
```




```
ggsave(p, file = "experience_allgames.jpg")

# box plot of all responses
boxplot(subject_data_all_wide$SkillController, subject_data_all_wide$SkillVR, subject_data_all_wide$Skil
    main = "Experience", names = c("Controller", "VR", "All Games"), xlab = "Response 1-5",
    notch = TRUE, horizontal = TRUE)
```




```
# count gender for each group
gender_count_groups <- data.frame(InterfaceOrder = NULL, Gender = NULL, Count = NULL)
for (x in levels(survey_data$InterfaceOrder)) {
    for (y in levels(survey_data$Gender)) {
        temp.frame <- data.frame(InterfaceOrder = x, Gender = y, Count = length(which(survey_data$Inter
            x & survey_data$Gender == y)))
        gender_count_groups <- rbind(gender_count_groups, temp.frame)
    }
}

# plot groups with gender make-up
gender.group.plot <- ggplot() + theme_minimal() + # theme(legend.position = 'none') +
geom_bar(stat = "identity", aes(y = Count, x = InterfaceOrder, fill = Gender), data = gender_count_grou
    width = 0.6) + scale_fill_brewer(type = "qual") + # geom_errorbar(aes(ymin=Mean-(SE*1.96), ymax=Mea
# colour=InterfaceOrder), width=.2) +
labs(title = paste0("Group counts w/ gender, N = ", sample_size), y = "# subjects")
# geom_text(data=gender_count_groups, aes(y = Count, x = InterfaceOrder, label =
# Count),size=4) coord_cartesian(ylim=c(0.0035, 0.006))
gender.group.plot
```




```r
# ggsave(gender.group.plot, file='gender_groups.jpg')
```

Performance Metrics

- These are the primary results of this study
- Two IV's: Cube Size and Interface
- DV's: accuracy, 3 time measures, errors (accidental drops)
- Main effect of cube size: shows cube sizes mattered
- Interaction effect: indicates Interfaces may be better or worse at cubes of different sizes

Interface

All Performance t-tests

Here I re-did the t-tests that follow this section, but all in one place. Then I apply the Holm correction to all 5 tests per each interface, where as before I was applying it to 2 tests (2 interfaces per each of the 5 measures). The previous application resulted in the Holm correction essentially not being applied to the comparisons between the HHI_Leap and the B_Leap.

```r
# HHI vs. Oculus

Measure=c("Accuracy (m)","Total Time (s)","Grab Time (s)","Release Time (s)","Accidental Drops (#)")

temp_df <- subject_data_all_long %>% ungroup() %>% filter(Interface=="HHI_Leap")

hhi_leap_mean_sd <- #left_join(get_summary_stats(temp_df, Distance,))
```

```r
  data.frame(Measure=Measure,
      HHI_Leap_Mean=c(mean(temp_df$Distance), mean(temp_df$totaltime), mean(temp_df$grabtime), mean(temp
      HHI_Leap_SD=c(sd(temp_df$Distance), sd(temp_df$totaltime), sd(temp_df$grabtime), sd(temp_df$releas
             )

temp_df <- subject_data_all_long %>% ungroup() %>% filter(Interface=="Oculus")

oculus_mean_sd <-
  data.frame(Measure=Measure,
      Oculus_Mean=c(mean(temp_df$Distance), mean(temp_df$totaltime), mean(temp_df$grabtime), mean(temp_d
      Oculus_SD=c(sd(temp_df$Distance), sd(temp_df$totaltime), sd(temp_df$grabtime), sd(temp_df$release
             )

hhi_leap_oculus_mean_sd <- left_join(hhi_leap_mean_sd, oculus_mean_sd)

# Construct a data frame with all t-tests HHI_Leap vs. Oculus.
# Then apply the Holm adjustment at the end with the adjust_pvalue() function from rstatix
# (Holm is the default type of correction)
all_performance_ttests_oculus <- bind_rows(
  subject_data_all_long %>% t_test(Distance ~ Interface, comparisons = list(c("HHI_Leap","Oculus")), pa
  subject_data_all_long %>% t_test(totaltime ~ Interface, comparisons = list(c("HHI_Leap","Oculus")), p
  subject_data_all_long %>% t_test(grabtime ~ Interface, comparisons = list(c("HHI_Leap","Oculus")), pa
  subject_data_all_long %>% t_test(releasetime ~ Interface, comparisons = list(c("HHI_Leap","Oculus")),
  subject_data_all_long %>% t_test(Drop_Count ~ Interface, comparisons = list(c("HHI_Leap","Oculus")), 
) %>% adjust_pvalue(p.col="p", output.col="p.adj") %>% cbind(Measure) %>% select(Measure, t=statistic, 

performance_ttests_hhi_leap_oculus <- left_join(hhi_leap_oculus_mean_sd, all_performance_ttests_oculus)

# then set p values to be "p < .001" (if true) and remove the *
performance_ttests_hhi_leap_oculus[which(performance_ttests_hhi_leap_oculus$p<.0001),"p(Holm)"] <- "p <


rm(temp_df, oculus_mean_sd, hhi_leap_oculus_mean_sd, all_performance_ttests_oculus)

stargazer(performance_ttests_hhi_leap_oculus, summary=FALSE, rownames = FALSE, title="Paired Samples t-
```

```
##
## % Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harva
## % Date and time: Sun, Oct 18, 2020 - 22:55:26
## \begin{table}[!htbp] \centering
##   \caption{Paired Samples t-tests: HHI_Leap and Oculus}
##   \label{}
## \begin{tabular}{@{\extracolsep{5pt}} cccccccc}
## \\[-1.8ex]\hline
## \hline \\[-1.8ex]
## Measure & HHI\_Leap\_Mean & HHI\_Leap\_SD & Oculus\_Mean & Oculus\_SD & t & df & p(Holm) \\
## \hline \\[-1.8ex]
## Accuracy (m) & $0.016$ & $0.008$ & $0.007$ & $0.004$ & $7.127$ & $31$ & p \textless  .0001 \\
## Total Time (s) & $3.515$ & $1.347$ & $2.271$ & $0.753$ & $7.884$ & $31$ & p \textless  .0001 \\
## Grab Time (s) & $1.573$ & $0.652$ & $0.946$ & $0.248$ & $6.939$ & $31$ & p \textless  .0001 \\
## Release Time (s) & $1.942$ & $0.856$ & $1.324$ & $0.575$ & $6.701$ & $31$ & p \textless  .0001 \\
## Accidental Drops (\#) & $2.406$ & $2.092$ & $0.125$ & $0.336$ & $6.243$ & $31$ & p \textless  .0001 \
## \hline \\[-1.8ex]
```

```
## \end{tabular}
## \end{table}
```

```r
# HHI vs. B_Leap

temp_df <- subject_data_all_long %>% ungroup() %>% filter(Interface=="B_Leap")

b_leap_mean_sd <-
  data.frame(Measure=Measure,
      B_Leap_Mean=c(mean(temp_df$Distance), mean(temp_df$totaltime), mean(temp_df$grabtime), mean(temp_d
      B_Leap_SD=c(sd(temp_df$Distance), sd(temp_df$totaltime), sd(temp_df$grabtime), sd(temp_df$releaset
             )

hhi_leap_b_leap_mean_sd <- left_join(hhi_leap_mean_sd, b_leap_mean_sd)

all_performance_ttests_b_leap <- bind_rows(
  subject_data_all_long %>% t_test(Distance ~ Interface, comparisons = list(c("HHI_Leap","B_Leap")), pai
  subject_data_all_long %>% t_test(totaltime ~ Interface, comparisons = list(c("HHI_Leap","B_Leap")), pa
  subject_data_all_long %>% t_test(grabtime ~ Interface, comparisons = list(c("HHI_Leap","B_Leap")), pai
  subject_data_all_long %>% t_test(releasetime ~ Interface, comparisons = list(c("HHI_Leap","B_Leap")),
  subject_data_all_long %>% t_test(Drop_Count ~ Interface, comparisons = list(c("HHI_Leap","B_Leap")), p
) %>% adjust_pvalue(p.col="p", output.col="p.adj") %>% cbind(Measure) %>% select(Measure, t=statistic, d

performance_ttests_hhi_leap_b_leap <- left_join(hhi_leap_b_leap_mean_sd, all_performance_ttests_b_leap)

rm(temp_df, hhi_leap_b_leap_mean_sd, b_leap_mean_sd, hhi_leap_mean_sd, all_performance_ttests_b_leap)

stargazer(performance_ttests_hhi_leap_b_leap, summary=FALSE, rownames = FALSE)
```

```
##
## % Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harva
## % Date and time: Sun, Oct 18, 2020 - 22:55:26
## \begin{table}[!htbp] \centering
##   \caption{}
##   \label{}
## \begin{tabular}{@{\extracolsep{5pt}} cccccccc}
## \\[-1.8ex]\hline
## \hline \\[-1.8ex]
## Measure & HHI\_Leap\_Mean & HHI\_Leap\_SD & B\_Leap\_Mean & B\_Leap\_SD & t & df & p(Holm) \\
## \hline \\[-1.8ex]
## Accuracy (m) & $0.016$ & $0.008$ & $0.015$ & $0.005$ & $0.293$ & $31$ & $1$ \\
## Total Time (s) & $3.515$ & $1.347$ & $3.566$ & $1.569$ & $$-$0.293$ & $31$ & $1$ \\
## Grab Time (s) & $1.573$ & $0.652$ & $1.354$ & $0.410$ & $2.253$ & $31$ & $0.108$ \\
## Release Time (s) & $1.942$ & $0.856$ & $2.212$ & $1.236$ & $$-$2.317$ & $31$ & $0.108$ \\
## Accidental Drops (\#) & $2.406$ & $2.092$ & $4.750$ & $2.676$ & $$-$3.863$ & $31$ & $0.003$ \\
## \hline \\[-1.8ex]
## \end{tabular}
## \end{table}
```

Accuracy

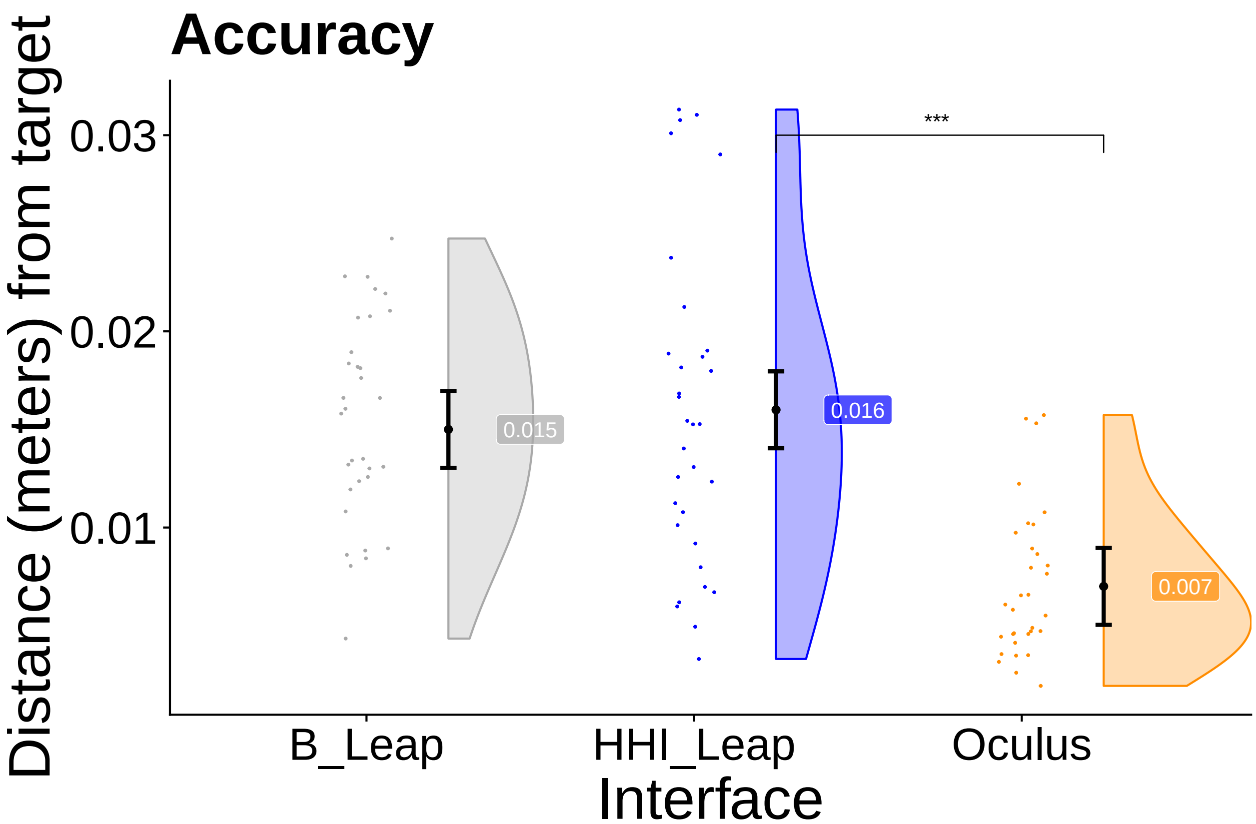
```r
# Distance/accuracy
  temp_plot_data <- subject_data_all_long %>%
    group_by(Interface) %>% get_summary_stats(Distance)

# run t test
stat.test <- subject_data_all_long %>%
  ungroup(.) %>%
  pairwise_t_test(Distance ~ Interface, paired=TRUE, comparisons=list(c("B_Leap","HHI_Leap"),c("HHI_Lea
  left_join(subject_data_all_long %>% ungroup(.) %>% cohens_d(Distance ~ Interface, paired=TRUE) %>% se
  mutate(Interface=group1)
stat.test
```

```
## # A tibble: 2 x 13
##   .y.   group1 group2    n1    n2 statistic    df        p   p.adj p.adj.signif
##   <chr> <chr>  <chr>  <int> <int>     <dbl> <dbl>    <dbl>   <dbl> <chr>
## 1 Dist~ B_Leap HHI_L~    32    32    -0.293    31 7.71e-1 7.71e-1 ""
## 2 Dist~ HHI_L~ Oculus    32    32     7.13     31 5.22e-8 1.04e-7 "*** "
## # ... with 3 more variables: effsize <dbl>, magnitude <ord>, Interface <chr>
```

```r
stat.test.anova <-
  anova_summary(effect.size="pes",aov(Distance ~ Interface + Error(id/Interface), data=subject_data_all_
stat.test.anova
```

```
##      Effect DFn DFd      F        p p<.05   pes
## 1 Interface   2  62 41.711 3.33e-12     * 0.574
```

```r
  distance_Interface_raincloud <- ggplot(subject_data_all_long, aes(x=Interface,y=Distance, fill = Inte
    geom_flat_violin(position = position_nudge(x = .25, y = 0), alpha=myalpha, adjust = mysmoothing)+
    geom_point(position = position_jitter(width = .08), size = .25)+
    geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(.25), colo
    stat_pvalue_manual(data=stat.test %>% filter(p.adj < 0.05), xmin="group1", xmax="group2", label = "p
    geom_label(data=temp_plot_data, aes(Interface, mean, label=round(mean,3)), alpha=.7, position=positi
    geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(se
    ylab('Distance (meters) from target')+
    xlab("Interface")+
    #theme(title = element_text(size=30))+
    guides(fill = FALSE, colour = FALSE) + #coord_flip()+
    scale_color_manual(values=mycolors)+#scale_colour_brewer(palette = "Set2")+
    scale_fill_manual(values=mycolors)+#scale_fill_brewer(palette = "Set2", direction=1)+
    labs(title="Accuracy")+
    theme(plot.title = element_text(size=title_size), axis.title = element_text(size=axis_text_size), a
  distance_Interface_raincloud
```




```r
  ggsave("accuracy_plot_main.jpg", width=10, height=7)

# ALMOST fails Levene's test for homogeneity of variance
leveneTest(Distance ~ Interface, data=subject_data_all_long %>% filter(Interface=="B_Leap" | Interface==
```

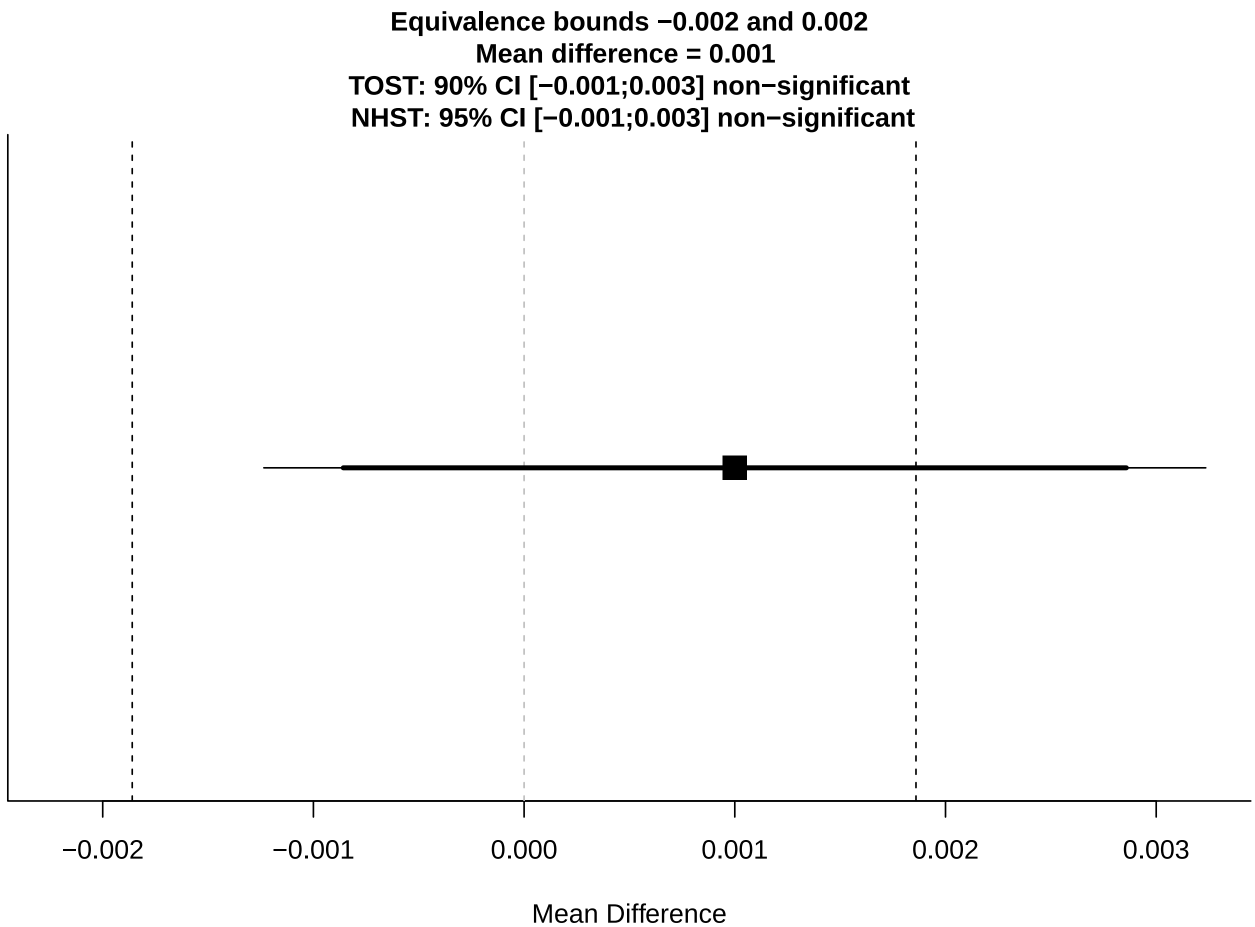
```
## Levene's Test for Homogeneity of Variance (center = median)
##       Df F value  Pr(>F)
## group  1  3.8323 0.05478 .
##       62
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# transform for data output
stat.test <- stat.test %>% select(-.y., -n1, -n2, -Interface)%>% mutate(p=round(p, 4), p.adj=round(p.ad

#stat.test.anova <- stat.test.anova %>% mutate(p=round(p, 4))

# save for output later
anova.Interface.distance <- stat.test.anova


ttest.Interface.distance <- stat.test
descriptives.Interface.distance <- subject_data_all_long %>% group_by(Interface) %>% get_summary_stats(
descriptives.Interface.distance
```

```
## # A tibble: 3 x 7
```

```
##   Interface variable  mean    sd   min   max   iqr
##   <fct>     <chr>    <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 B_Leap    Distance 0.015 0.005 0.004 0.025 0.007
## 2 HHI_Leap  Distance 0.016 0.008 0.003 0.031 0.009
## 3 Oculus    Distance 0.007 0.004 0.002 0.016 0.005
```

```
oculus_diffs <- data.frame(Difference=round(temp_plot_data$mean[1]/temp_plot_data$mean[3], 3),
                           Type = "Ratio", row.names="Accuracy", stringsAsFactors = FALSE)
oculus_diffs["Accuracy","cohen's d"] <- round(stat.test[3,"effsize"], 3)

# equivalence: HHI Leap and B_Leap

# find correlation of dependent variable between HHI Leap and B_Leap
r_val <- cor.test(
  x = subject_data_all_long[which(subject_data_all_long$Interface=="HHI_Leap"),"Distance"],
  y = subject_data_all_long[which(subject_data_all_long$Interface=="B_Leap"),"Distance"],
  method = "pearson",
  alternative = "two.sided"
  )
r_val <- r_val$estimate[[1]]

TOSTaccuracy <- TOSTpaired(n=32,
  m1 = descriptives.Interface.distance[which(descriptives.Interface.distance$Interface=="HHI_Leap"),"mea
  m2 = descriptives.Interface.distance[which(descriptives.Interface.distance$Interface=="B_Leap"),"mean
  sd1 = descriptives.Interface.distance[which(descriptives.Interface.distance$Interface=="HHI_Leap"),"sd
  sd2 = descriptives.Interface.distance[which(descriptives.Interface.distance$Interface=="B_Leap"),"sd"]
  r12 = r_val,
  low_eqbound_dz = -.3,
  high_eqbound_dz = .3,
  alpha = .05,
  plot= TRUE,
  verbose = TRUE
)
```




```
## TOST results:
## t-value lower bound: 2.61    p-value lower bound: 0.007
## t-value upper bound: -0.785  p-value upper bound: 0.219
## degrees of freedom : 31
##
## Equivalence bounds (Cohen's dz):
## low eqbound: -0.3
## high eqbound: 0.3
##
## Equivalence bounds (raw scores):
## low eqbound: -0.0019
## high eqbound: 0.0019
##
## TOST confidence interval:
## lower bound 90% CI: -0.001
## upper bound 90% CI:  0.003
##
## NHST confidence interval:
## lower bound 95% CI: -0.001
## upper bound 95% CI:  0.003
##
## Equivalence Test Result:
## The equivalence test was non-significant, t(31) = -0.785, p = 0.219, given equivalence bounds of -0.0
## Null Hypothesis Test Result:
```

```
## The null hypothesis test was non-significant, t(31) = 0.912, p = 0.369, given an alpha of 0.05.
## Based on the equivalence test and the null-hypothesis test combined, we can conclude that the observe
```

```
TOSTaccuracy
```

```
## $diff
## [1] 0.001
##
## $TOST_t1
## [1] 2.609462
##
## $TOST_p1
## [1] 0.006918714
##
## $TOST_t2
## [1] -0.784651
##
## $TOST_p2
## [1] 0.2193068
##
## $TOST_df
## [1] 31
##
## $alpha
## [1] 0.05
##
## $low_eqbound
## [1] -0.001859981
##
## $high_eqbound
## [1] 0.001859981
##
## $low_eqbound_dz
## [1] -0.3
##
## $high_eqbound_dz
## [1] 0.3
##
## $LL_CI_TOST
## [1] -0.0008582957
##
## $UL_CI_TOST
## [1] 0.002858296
##
## $LL_CI_TTEST
## [1] -0.001235315
##
## $UL_CI_TTEST
## [1] 0.003235315
```

Grab time

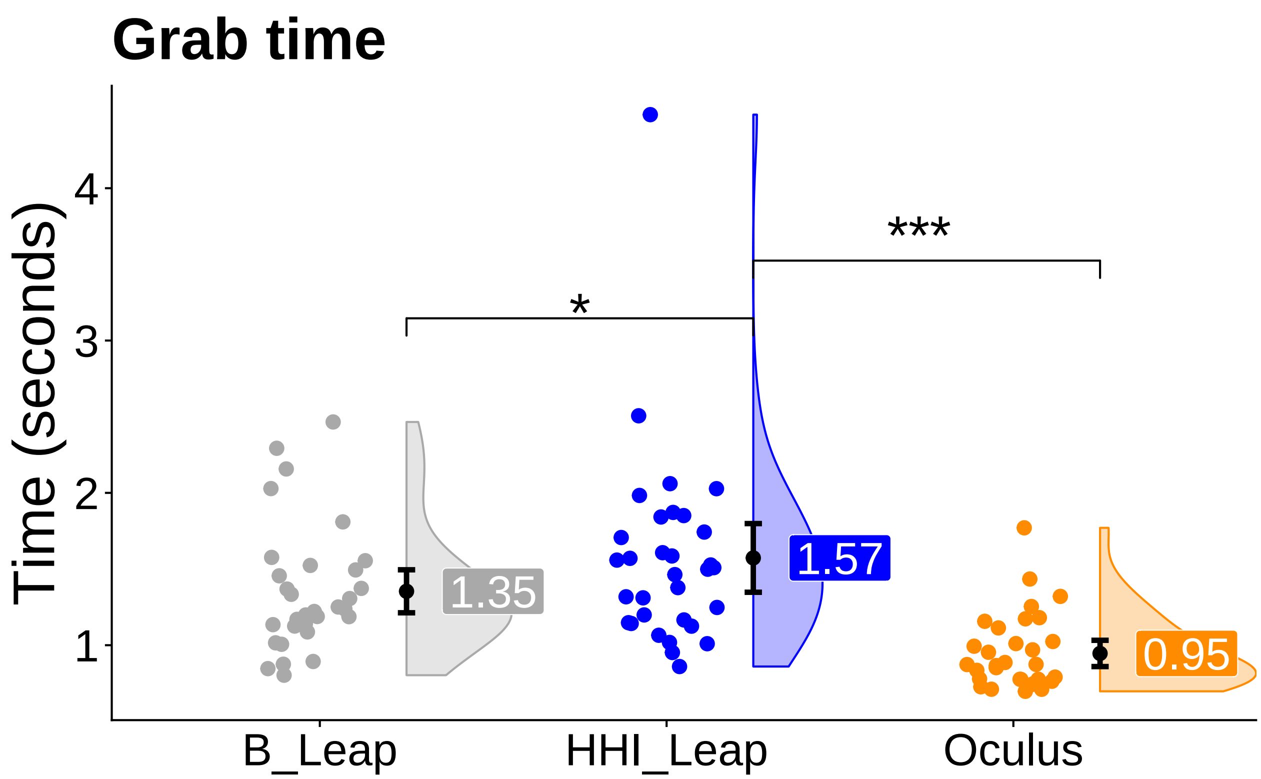
```r
# grab time
  # dot plot w/ error bars
  temp_plot_data <- subject_data_all_long %>%
      group_by(Interface) %>% get_summary_stats(grabtime)

stat.test <- subject_data_all_long %>%
  ungroup(.) %>%
  pairwise_t_test(grabtime ~ Interface, paired=TRUE, comparisons=list(c("B_Leap","HHI_Leap"),c("HHI_Lea
  left_join(subject_data_all_long %>% ungroup(.) %>% cohens_d(grabtime ~ Interface, paired=TRUE) %>% se
  mutate(Interface=group1)
stat.test
```

```
## # A tibble: 2 x 13
##   .y.   group1 group2    n1    n2 statistic    df       p   p.adj p.adj.signif
##   <chr> <chr>  <chr>  <int> <int>     <dbl> <dbl>   <dbl>   <dbl> <chr>
## 1 grab~ B_Leap HHI_L~    32    32     -2.25    31 3.20e-2 3.20e-2 *
## 2 grab~ HHI_L~ Oculus    32    32      6.94    31 8.76e-8 1.75e-7 "*** "
## # ... with 3 more variables: effsize <dbl>, magnitude <ord>, Interface <chr>
```

```r
stat.test.anova <-
  anova_summary(effect.size="pes",aov(grabtime ~ Interface + Error(id/Interface), data=subject_data_all
stat.test.anova
```

```
##      Effect DFn DFd      F        p p<.05   pes
## 1 Interface   2  62 29.057 1.25e-09     * 0.484
```

```r
  #raincloud grabtime
  grabtime_Interface_raincloud <- ggplot(subject_data_all_long,aes(x=Interface, y=grabtime, fill = Inte
    geom_flat_violin(position = position_nudge(x = .25, y = 0),  alpha=myalpha, adjust = mysmoothing)+
    geom_point(position = position_jitter(width = .15), size = 3)+
    geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(.25), col
    geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(s
geom_label(data=temp_plot_data, aes(Interface, mean, label=round(mean,2)), alpha=1, position=position_n
    ylab('Time (seconds)')+xlab('')+theme_cowplot()+guides(fill = FALSE, colour = FALSE)+#coord_flip()+
    scale_color_manual(values=mycolors)+#scale_colour_brewer(palette = "Set2")+
    scale_fill_manual(values=mycolors)+#scale_fill_brewer(palette = "Set2", direction=1)+
    labs(title="Grab time")+
    stat_pvalue_manual(data=stat.test %>% filter(p.adj < 0.05), xmin="group1", xmax="group2", label = "
    theme(plot.title = element_text(size=title_size), axis.title = element_text(size=axis_text_size), a
grabtime_Interface_raincloud
```




```
  ggsave("grabtime_plot_main.jpg", width=10, height=7)


# transform for data output
stat.test <- stat.test %>% select(-.y., -n1, -n2, -Interface)%>% mutate(p=round(p, 4), p.adj=round(p.ad

#stat.test.anova <- stat.test.anova %>% mutate(p=round(p, 4))

# save for later
anova.Interface.grabtime <- stat.test.anova
ttest.Interface.grabtime <- stat.test
descriptives.Interface.grabtime <- subject_data_all_long %>% group_by(Interface) %>% get_summary_stats(
descriptives.Interface.grabtime
```

```
## # A tibble: 3 x 7
##   Interface variable  mean    sd   min   max   iqr
##   <fct>     <chr>    <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 B_Leap    grabtime 1.35  0.41  0.803  2.46 0.369
## 2 HHI_Leap  grabtime 1.57  0.652 0.86   4.48 0.606
## 3 Oculus    grabtime 0.946 0.248 0.697  1.77 0.274
```

```
oculus_diffs["Grab time","Difference"] <- round(temp_plot_data$mean[1]/temp_plot_data$mean[3], 3)
oculus_diffs["Grab time","Type"] <- "Ratio"
oculus_diffs["Grab time","cohen's d"] <- round(stat.test[3,"effsize"], 3)

r_val <- cor.test(
  x = subject_data_all_long[which(subject_data_all_long$Interface=="HHI_Leap"),"grabtime"],
  y = subject_data_all_long[which(subject_data_all_long$Interface=="B_Leap"),"grabtime"],
```

```
  method = "pearson",
  alternative = "two.sided"
  )
r_val <- r_val$estimate[[1]]

# TOSTpaired(n=32,
#   m1 = descriptives.Interface.grabtime[which(descriptives.Interface.grabtime$Interface=="HHI_Leap"),"
#   m2 = descriptives.Interface.grabtime[which(descriptives.Interface.grabtime$Interface=="B_Leap"),"me
#   sd1 = descriptives.Interface.grabtime[which(descriptives.Interface.grabtime$Interface=="HHI_Leap"),
#   sd2 = descriptives.Interface.grabtime[which(descriptives.Interface.grabtime$Interface=="B_Leap"),"s
#   r12 = r_val,
#   low_eqbound_dz = -.3,
#   high_eqbound_dz = .3,
#   alpha = .05,
#   plot= TRUE,
#   verbose = TRUE
# )
```

Release time

```
# release time
  temp_plot_data <- subject_data_all_long %>%
    group_by(Interface) %>% summarise(mean=mean(releasetime), sd=sd(releasetime), se=(sd/sqrt(sample_si

stat.test <- subject_data_all_long %>%
  ungroup(.) %>%
  pairwise_t_test(releasetime ~ Interface, paired=TRUE, comparisons=list(c("B_Leap","HHI_Leap"),c("HHI_
  add_significance(p.col="p.adj", output.col="p.adj.signif", symbols=mysymbols) %>%
  left_join(subject_data_all_long %>% ungroup(.) %>% cohens_d(releasetime ~ Interface, paired=TRUE) %>%
  mutate(Interface=group1)
stat.test
```

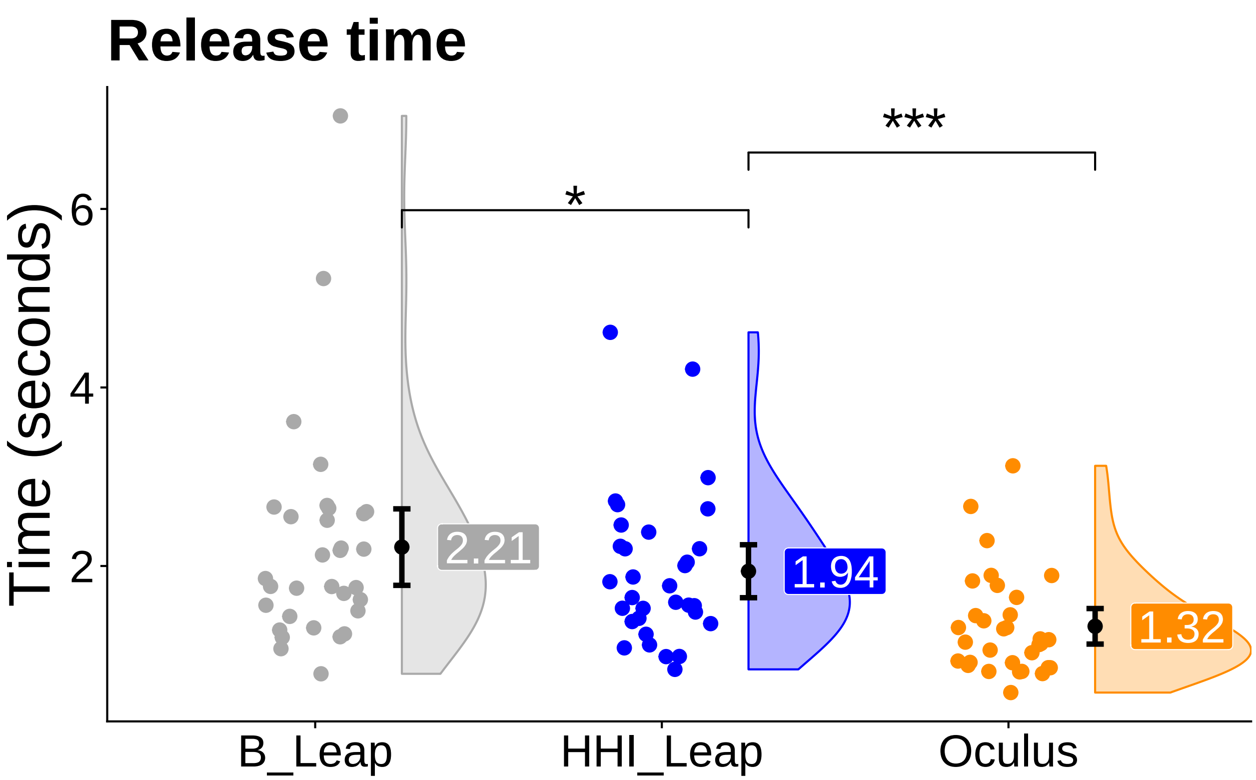
```
## # A tibble: 2 x 13
##   .y.   group1 group2    n1    n2 statistic    df       p   p.adj p.adj.signif
##   <chr> <chr>  <chr>  <int> <int>     <dbl> <dbl>   <dbl>   <dbl> <chr>
## 1 rele~ B_Leap HHI_L~    32    32      2.32    31 2.70e-2 2.70e-2 *
## 2 rele~ HHI_L~ Oculus    32    32      6.70    31 1.70e-7 3.40e-7 "*** "
## # ... with 3 more variables: effsize <dbl>, magnitude <ord>, Interface <chr>
```

```
stat.test.anova <-
  anova_summary(effect.size="pes",aov(releasetime ~ Interface + Error(id/Interface), data=subject_data_a
stat.test.anova
```

```
##      Effect DFn DFd      F        p p<.05   pes
## 1 Interface   2  62 30.342 6.48e-10     * 0.495
```

```
#raincloud releasetime
  releasetime_Interface_raincloud <- ggplot(subject_data_all_long,aes(x=Interface,y=releasetime, fill =
    geom_flat_violin(position = position_nudge(x = .25, y = 0), alpha=myalpha, adjust = mysmoothing)+
    geom_point(position = position_jitter(width = .15), size = 3)+
    geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(.25), colo
    #stat_compare_means(comparisons=list(c("B_Leap", "HHI_Leap"), c("HHI_Leap", "Oculus"), c("B_Leap",
```

```r
    geom_label(data=temp_plot_data, aes(Interface, mean, label=round(mean,2)), alpha=1, position=positi
    geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(se
    ylab('Time (seconds)')+xlab('')+theme_cowplot()+guides(fill = FALSE, colour = FALSE) +
    scale_color_manual(values=mycolors)+#scale_colour_brewer(palette = "Set2")+
    scale_fill_manual(values=mycolors)+#scale_fill_brewer(palette = "Set2", direction=1)+
    labs(title="Release time")+
    stat_pvalue_manual(data=stat.test %>% filter(p.adj < 0.05), xmin="group1", xmax="group2", label = "p
    theme(plot.title = element_text(size=title_size), axis.title = element_text(size=axis_text_size), ax
  releasetime_Interface_raincloud
```




```r
    ggsave("releasetime_plot_main.jpg", width=10, height=7)


# transform for data output
stat.test <- stat.test %>% select(-.y., -n1, -n2, -Interface)%>% mutate(p=round(p, 4), p.adj=round(p.ad

#stat.test.anova <- stat.test.anova %>% mutate(p=round(p, 4))

anova.Interface.releasetime <- stat.test.anova
ttest.Interface.releasetime <- stat.test
descriptives.Interface.releasetime <- subject_data_all_long %>% group_by(Interface) %>% get_summary_sta
descriptives.Interface.releasetime
```

```
## # A tibble: 3 x 7
##   Interface variable     mean    sd   min   max   iqr
##   <fct>     <chr>       <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 B_Leap    releasetime  2.21 1.24  0.792  7.04 1.11
## 2 HHI_Leap  releasetime  1.94 0.856 0.842  4.62 0.855
```

```
## 3 Oculus    releasetime  1.32 0.575 0.582  3.12 0.593
```

```
oculus_diffs["Release time","Difference"] <- round(temp_plot_data$mean[1]/temp_plot_data$mean[3], 3)
oculus_diffs["Release time","Type"] <- "Ratio"
oculus_diffs["Release time","cohen's d"] <- round(stat.test[3,"effsize"], 3)
```

Total time

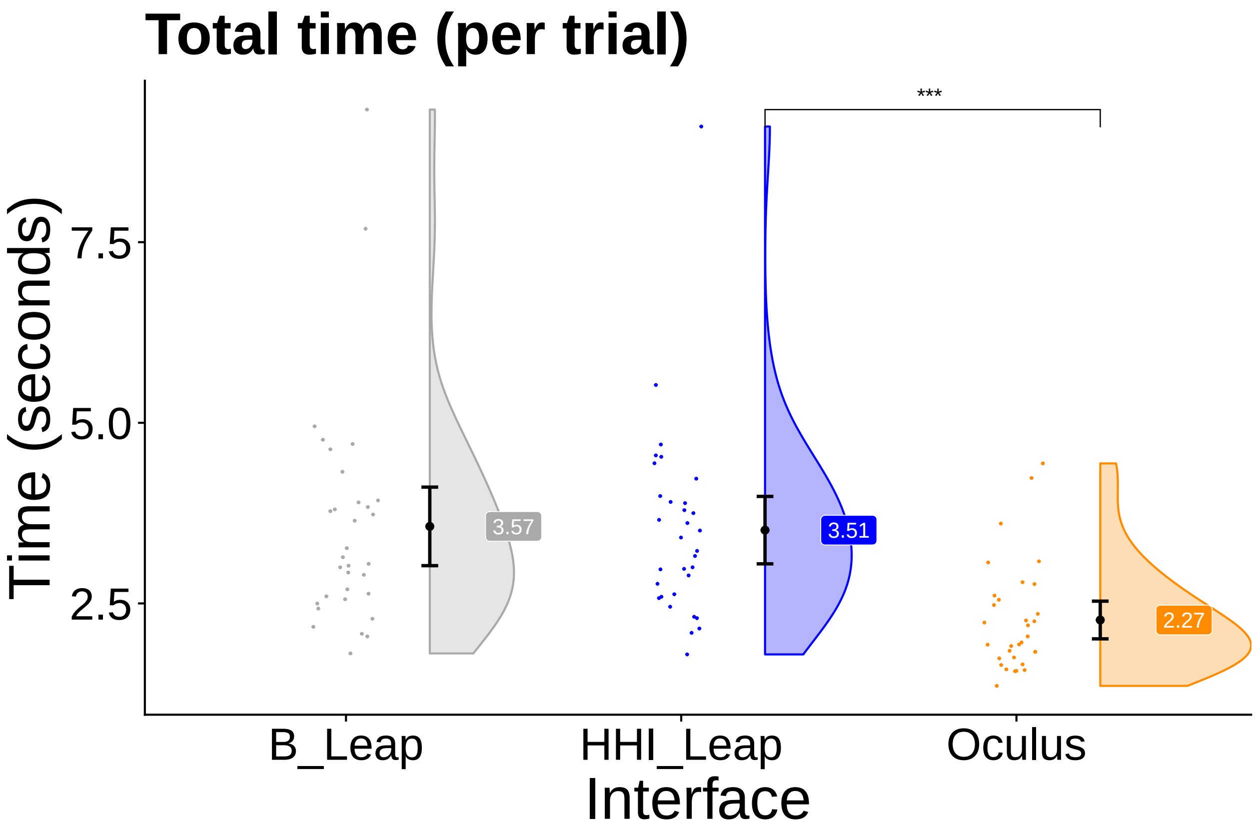
```
#total time
temp_plot_data <- subject_data_all_long %>%
    group_by(Interface) %>% summarise(mean=mean(totaltime), sd=sd(totaltime), se=(sd/sqrt(sample_size))

stat.test.anova <-
  anova_summary(effect.size="pes",aov(totaltime ~ Interface + Error(id/Interface), data=subject_data_al
stat.test.anova
```

```
##      Effect DFn DFd      F        p p<.05   pes
## 1 Interface   2  62 36.619 3.16e-11     * 0.542
```

```
stat.test <- subject_data_all_long %>%
  ungroup(.) %>%
  pairwise_t_test(totaltime ~ Interface, paired=TRUE, comparisons=list(c("B_Leap","HHI_Leap"),c("HHI_Le
  add_significance(p.col="p.adj", output.col="p.adj.signif", symbols=mysymbols) %>%
  left_join(subject_data_all_long %>% ungroup(.) %>% cohens_d(totaltime ~ Interface, paired=TRUE) %>% s
  mutate(Interface=group1)
stat.test
```

```
## # A tibble: 2 x 13
##   .y.   group1 group2    n1    n2 statistic    df       p   p.adj p.adj.signif
##   <chr> <chr>  <chr>  <int> <int>     <dbl> <dbl>   <dbl>   <dbl> <chr>
## 1 tota~ B_Leap HHI_L~    32    32     0.293    31 7.72e-1 7.72e-1 ""
## 2 tota~ HHI_L~ Oculus    32    32     7.88     31 6.72e-9 1.34e-8 "*** "
## # ... with 3 more variables: effsize <dbl>, magnitude <ord>, Interface <chr>
```

```
  #raincloud totaltime
  totaltime_Interface_raincloud <- ggplot(subject_data_all_long,aes(x=Interface, y=totaltime, fill = In
    geom_flat_violin(position = position_nudge(x = .25, y = 0), alpha=myalpha, adjust = mysmoothing)+
    geom_point(position = position_jitter(width = .1), size = .25)+
    geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(.25), col
    #stat_compare_means(comparisons=list(c("B_Leap", "HHI_Leap"), c("HHI_Leap", "Oculus"), c("B_Leap",
    geom_label(data=temp_plot_data, aes(Interface, mean, label=round(mean,2)), alpha=1, position=positi
    geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(s
    ylab('Time (seconds)')+xlab('Interface')+theme_cowplot()+guides(fill = FALSE, colour = FALSE) +
    scale_color_manual(values=mycolors)+#scale_colour_brewer(palette = "Set2")+
    scale_fill_manual(values=mycolors)+#scale_fill_brewer(palette = "Set2", direction=1)+
    labs(title="Total time (per trial)")+
    stat_pvalue_manual(data=stat.test %>% filter(p.adj < 0.05), xmin="group1", xmax="group2", label = "
    theme(plot.title = element_text(size=title_size), axis.title = element_text(size=axis_text_size), a
totaltime_Interface_raincloud
```




```
  ggsave("totaltime_plot_main.jpg", width=10, height=7)


# transform for data output
stat.test <- stat.test %>% select(-.y., -n1, -n2, -Interface)%>% mutate(p=round(p, 4), p.adj=round(p.ad

#stat.test.anova <- stat.test.anova %>% mutate(p=round(p, 4))

anova.Interface.totaltime <- stat.test.anova
ttest.Interface.totaltime <- stat.test
descriptives.Interface.totaltime <- subject_data_all_long %>% group_by(Interface) %>% get_summary_stats
descriptives.Interface.totaltime
```

```
## # A tibble: 3 x 7
##   Interface variable   mean    sd   min   max   iqr
##   <fct>     <chr>     <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 B_Leap    totaltime  3.57 1.57   1.81  9.34 1.32
## 2 HHI_Leap  totaltime  3.52 1.35   1.79  9.10 1.31
## 3 Oculus    totaltime  2.27 0.753  1.36  4.44 0.817
```

```
oculus_diffs["Total time","Difference"] <- round(temp_plot_data$mean[1]/temp_plot_data$mean[3], 3)
oculus_diffs["Total time","Type"] <- "Ratio"
oculus_diffs["Total time","cohen's d"] <- round(stat.test[3,"effsize"], 3)

# Interface order
Interface.order.anova <-
```

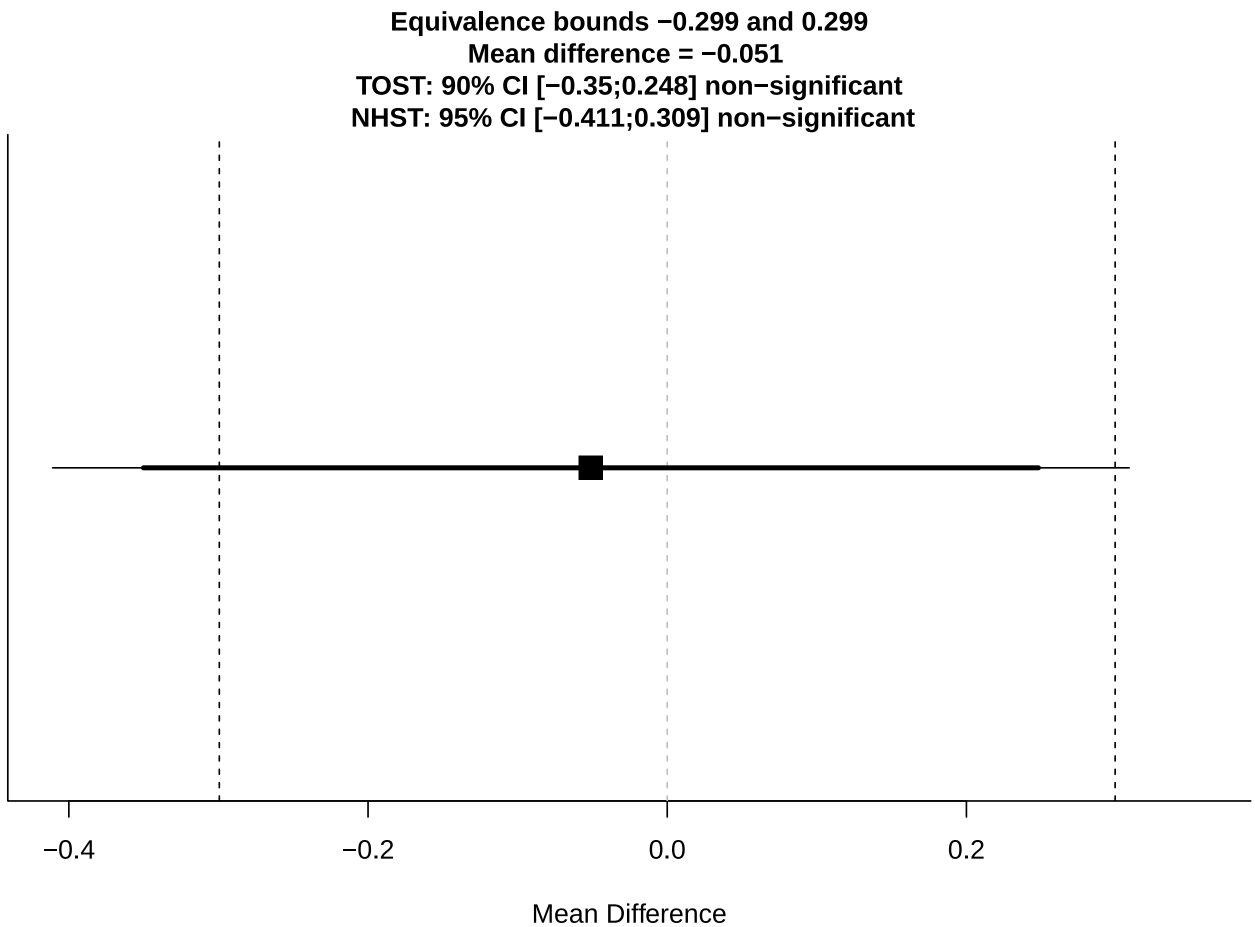
```r
  anova_summary(effect.size="pes",aov(totaltime ~ Interface*InterfaceOrder + Error(id/Interface), data=s
Interface.order.anova
```

```
##                      Effect DFn DFd      F        p p<.05   pes
## 1            InterfaceOrder   5  26  0.929 4.78e-01       0.152
## 2                 Interface   2  52 48.200 1.44e-12     * 0.650
## 3 Interface:InterfaceOrder  10  52  2.961 5.00e-03     * 0.363
```

```r
#Leap group
Interface.order.anova2 <-
  anova_summary(effect.size="pes",aov(totaltime ~ Interface*Leap_Group + Error(id/Interface), data=subje
Interface.order.anova2
```

```
##                  Effect DFn DFd      F     p p<.05   pes
## 1            Leap_Group   1  30  0.443 0.511       0.015
## 2             Interface   1  30  0.114 0.737       0.004
## 3 Interface:Leap_Group   1  30 11.434 0.002     * 0.276
```

```r
r_val <- cor.test(
  x = subject_data_all_long[which(subject_data_all_long$Interface=="HHI_Leap"),"totaltime"],
  y = subject_data_all_long[which(subject_data_all_long$Interface=="B_Leap"),"totaltime"],
  method = "pearson",
  alternative = "two.sided"
  )
r_val <- r_val$estimate[[1]]

TOSTpaired(n=32,
  m1 = descriptives.Interface.totaltime[which(descriptives.Interface.totaltime$Interface=="HHI_Leap"),"m
  m2 = descriptives.Interface.totaltime[which(descriptives.Interface.totaltime$Interface=="B_Leap"),"mea
  sd1 = descriptives.Interface.totaltime[which(descriptives.Interface.totaltime$Interface=="HHI_Leap"),
  sd2 = descriptives.Interface.totaltime[which(descriptives.Interface.totaltime$Interface=="B_Leap"),"sd
  r12 = r_val,
  low_eqbound_dz = -.3,
  high_eqbound_dz = .3,
  alpha = .05,
  plot= TRUE,
  verbose = TRUE
)
```




```
## TOST results:
## t-value lower bound: 1.41    p-value lower bound: 0.085
## t-value upper bound: -1.99   p-value upper bound: 0.028
## degrees of freedom : 31
##
## Equivalence bounds (Cohen's dz):
## low eqbound: -0.3
## high eqbound: 0.3
##
## Equivalence bounds (raw scores):
## low eqbound: -0.2994
## high eqbound: 0.2994
##
## TOST confidence interval:
## lower bound 90% CI: -0.35
## upper bound 90% CI:  0.248
##
## NHST confidence interval:
## lower bound 95% CI: -0.411
## upper bound 95% CI:  0.309
##
## Equivalence Test Result:
## The equivalence test was non-significant, t(31) = 1.408, p = 0.0845, given equivalence bounds of -0.
## Null Hypothesis Test Result:
```

```
## The null hypothesis test was non-significant, t(31) = -0.289, p = 0.774, given an alpha of 0.05.
## Based on the equivalence test and the null-hypothesis test combined, we can conclude that the observe
```

Accidental drops

```r
#temp_plot_data <- subject_data_all_long %>% group_by(Interface) %>% summarise(mean=mean(Drop_Rate), sd

temp_plot_data <- subject_data_all_long %>% group_by(Interface) %>% get_summary_stats(Drop_Count)

stat.test.levene <- subject_data_all_long %>% levene_test(Drop_Count ~ Interface) %>% mutate(p=round(p,
stat.test.levene
```

```
## # A tibble: 1 x 4
##     df1   df2 statistic     p
##   <int> <int>     <dbl> <dbl>
## 1     2    93      32.5     0
```

```r
stat.test.shapiro <- subject_data_all_long %>% group_by(Interface) %>% shapiro_test(Drop_Count) %>% muta
stat.test.shapiro
```

```
## # A tibble: 3 x 5
##   Interface variable   statistic      p normal
##   <fct>     <chr>          <dbl>  <dbl> <lgl>
## 1 B_Leap    Drop_Count     0.946 0.113  TRUE
## 2 HHI_Leap  Drop_Count     0.884 0.0025 FALSE
## 3 Oculus    Drop_Count     0.391 0      FALSE
```

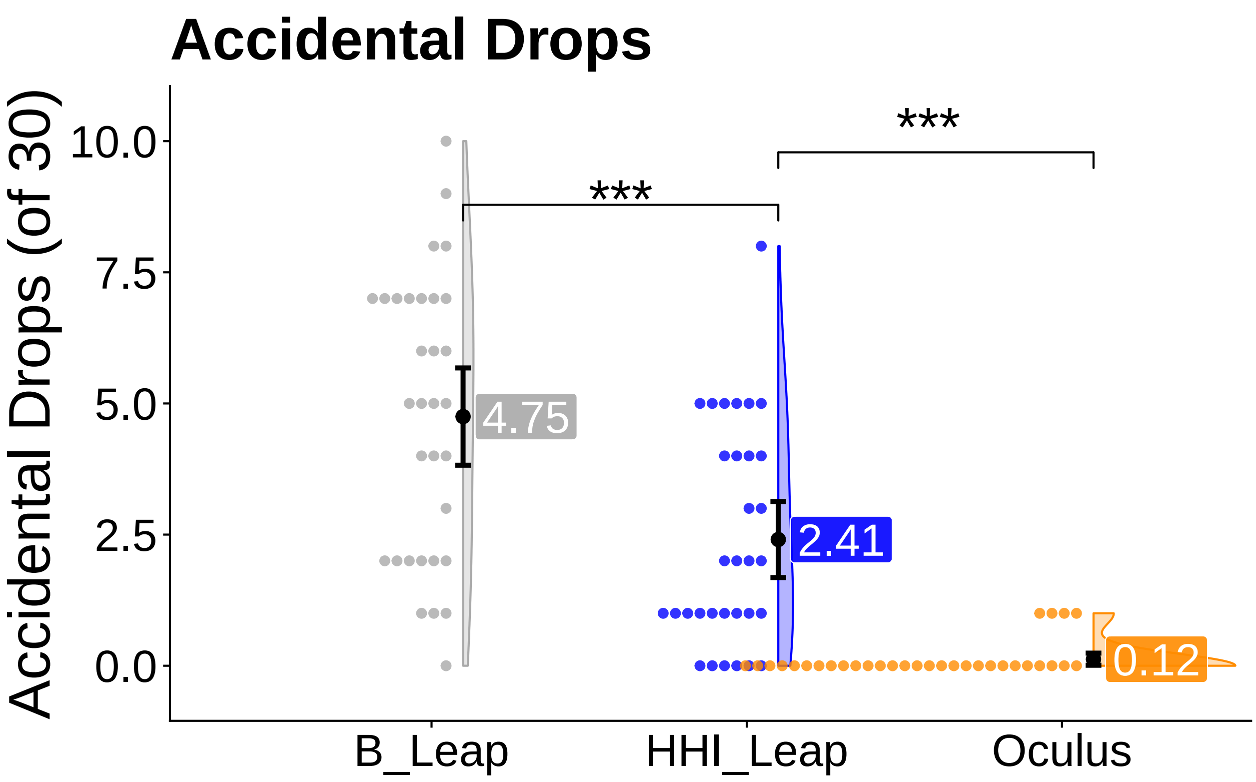
```r
stat.test.anova <-
anova_summary(effect.size="pes",aov(Drop_Count ~ Interface + Error(id/Interface), data=subject_data_all_
stat.test.anova
```

```
##      Effect DFn DFd      F        p p<.05   pes
## 1 Interface   2  62 44.383 1.09e-12     * 0.589
```

```r
#write.csv(stat.test.anova, file="dropcount_anova.csv")

stat.test <- subject_data_all_long %>%
  ungroup(.) %>%
  pairwise_t_test(Drop_Count ~ Interface, paired=TRUE, comparisons=list(c("B_Leap","HHI_Leap"),c("HHI_Le
  add_significance(p.col="p.adj", output.col="p.adj.signif", symbols=mysymbols) %>%
  left_join(subject_data_all_long %>% ungroup(.) %>% cohens_d(Drop_Count ~ Interface, paired=TRUE) %>% s
  mutate(Interface=group1)
stat.test
```

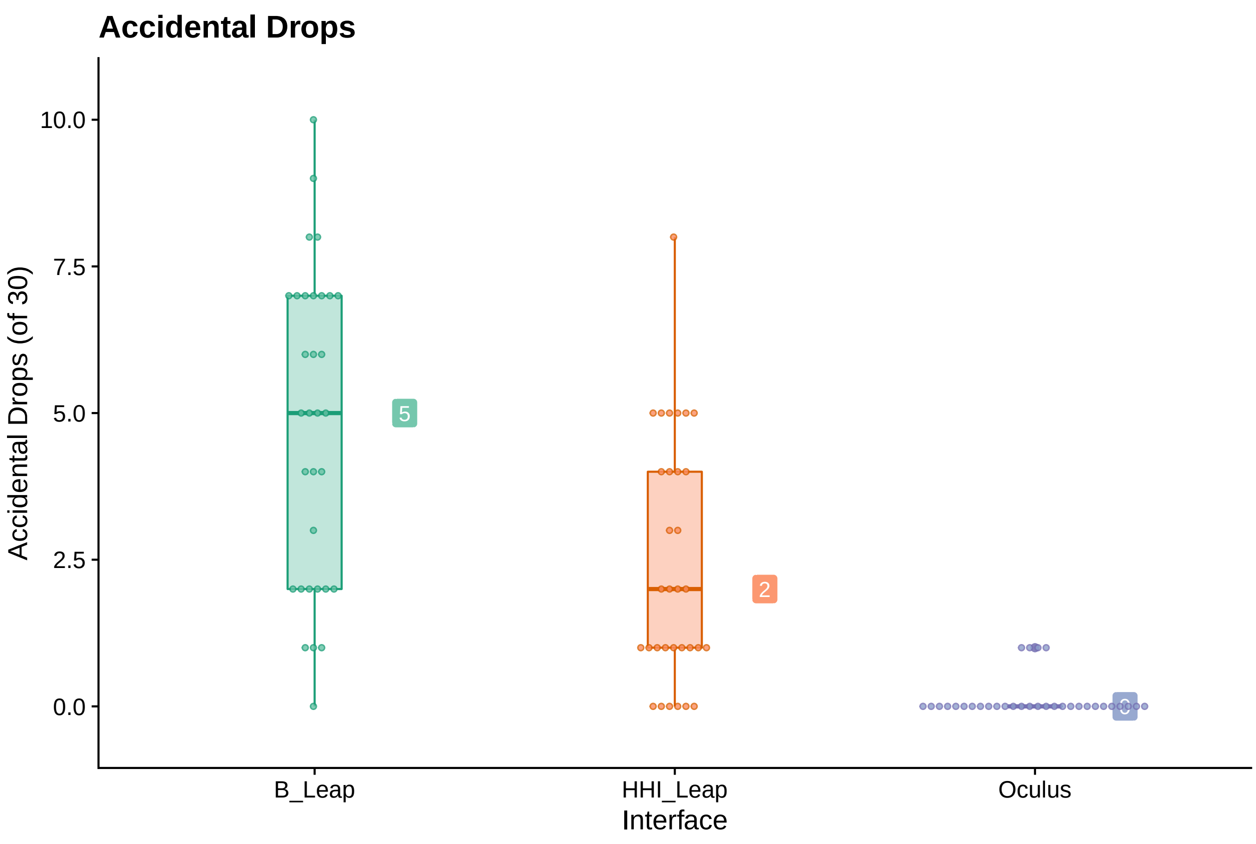
```
## # A tibble: 2 x 13
##   .y.   group1 group2    n1    n2 statistic    df       p   p.adj p.adj.signif
##   <chr> <chr>  <chr>  <int> <int>     <dbl> <dbl>   <dbl>   <dbl> <chr>
## 1 Drop~ B_Leap HHI_L~    32    32      3.86    31 5.34e-4 5.34e-4 ***
## 2 Drop~ HHI_L~ Oculus    32    32      6.24    31 6.16e-7 1.23e-6 "*** "
## # ... with 3 more variables: effsize <dbl>, magnitude <ord>, Interface <chr>
```

```
  # plot
drops_raincloud <-  ggplot(subject_data_all_long, aes(Interface, Drop_Count, fill = Interface, color = I
    geom_flat_violin(position = position_nudge(x = .1, y = 0), alpha=myalpha, adjust = mysmoothing)+
    #geom_pointrange(data=temp_plot_data, aes(Interface, median, ymin=q1, ymax=q3), width=.1, position=
    #geom_boxplot(width=.1, alpha=.4)+
    #geom_bar(data=temp_plot_data, aes(y=mean), stat="identity", width=.2)+
    #geom_point(position = position_jitter(width = .1, height=.01), size = .25)+
    #geom_linerange(data=temp_plot_data, aes(x=Interface, y=NULL, ymin=q1, ymax=q3), color="black", siz
    #geom_label(data=temp_plot_data, aes(Interface, median, label=paste0(ceiling(median),"%")), color="
   geom_label(data=temp_plot_data, aes(Interface, y=mean, label=round(mean, 2)), color="white", positio
    geom_dotplot(binaxis = "y", stackratio=1.3, binwidth = 1, stackdir="down", dotsize=.18, alpha=.8, p
    geom_point(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(se*1
    geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(s
    # geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+
    ylab('Accidental Drops (of 30)')+xlab('')+theme_cowplot()+guides(fill = FALSE, colour = FALSE, shap
    scale_color_manual(values=mycolors)+#scale_colour_brewer(palette = "Set2")+
    scale_fill_manual(values=mycolors)+#scale_fill_brewer(palette = "Set2", direction=1)+
    #geom_text(data=temp_plot_data, aes(label=mean),hjust=-.2, vjust=.2)+
    labs(title="Accidental Drops")+
    stat_pvalue_manual(data=stat.test %>% filter(p.adj < 0.05), xmin="group1", xmax="group2", label = "p
  theme(plot.title = element_text(size=title_size), axis.title = element_text(size=axis_text_size), axis
drops_raincloud
```

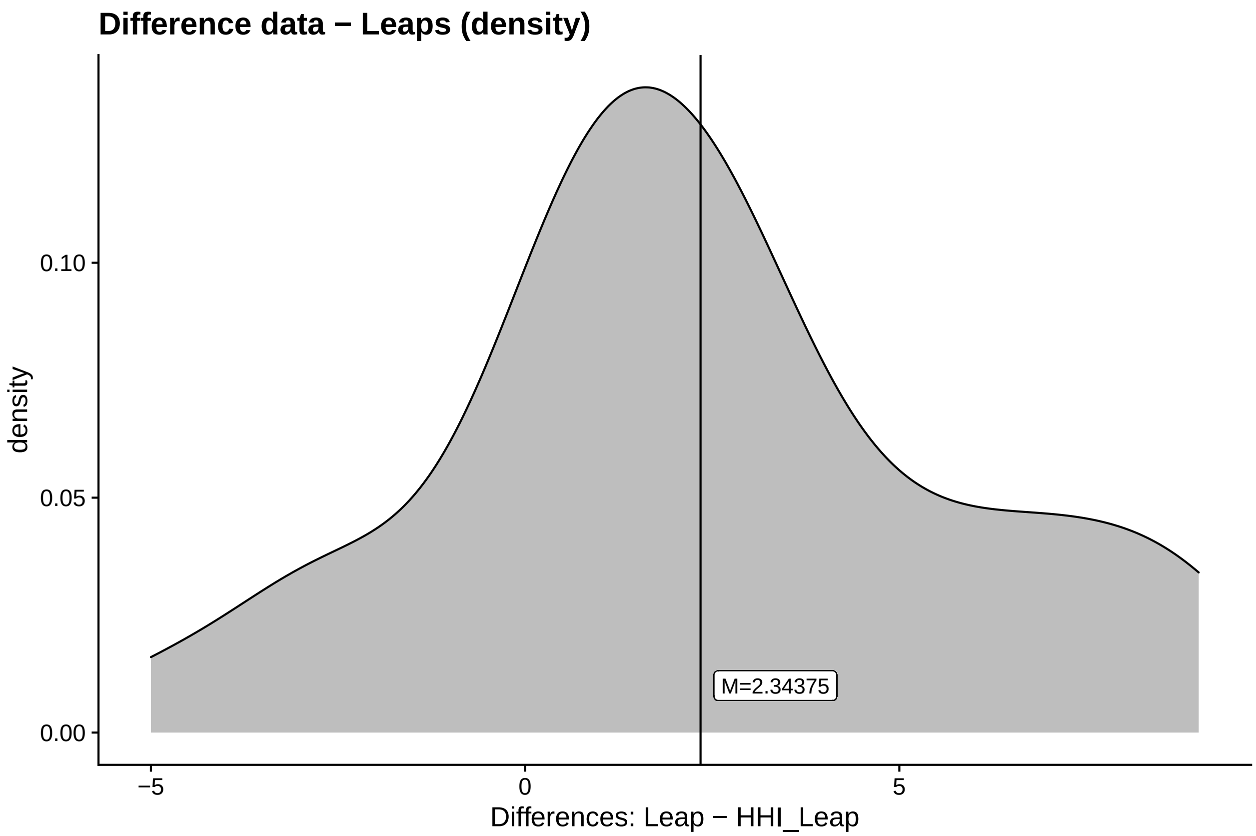



```
  ggsave("accidental_drops_main.jpg", width=10, height=7)


  # plot 2 (nonparametric)
drops_plot <-  ggplot(subject_data_all_long, aes(Interface, Drop_Count, fill = Interface, color = Inter
```

```
    #geom_flat_violin(position = position_nudge(x = .1, y = 0), alpha=.7)+#)+#adjust=2)+
    #geom_pointrange(data=temp_plot_data, aes(Interface, median, ymin=q1, ymax=q3), width=.1, position=
    geom_boxplot(width=.15, alpha=.4)+
    #geom_bar(data=temp_plot_data, aes(y=mean), stat="identity", width=.2)+
    #geom_point(position = position_jitter(width = .1, height=.01), size = .25)+
    #geom_linerange(data=temp_plot_data, aes(x=Interface, y=NULL, ymin=q1, ymax=q3), color="black", siz
    geom_label(data=temp_plot_data, aes(Interface, median, label=round(median, 2)), color="white", posi
   #geom_label(data=temp_plot_data, aes(Interface, y=mean, label=round(mean, 2)), color="white", positi
    geom_dotplot(binaxis = "y", stackratio=1.4, binwidth = 1, stackdir="center", dotsize=.1, alpha=.8, 
    #geom_point(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(se*
    #geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(
    # geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+
    ylab('Accidental Drops (of 30)')+xlab('Interface')+theme_cowplot()+guides(fill = FALSE, colour = FA
    scale_colour_brewer(palette = "Dark2")+
    scale_fill_brewer(palette = "Set2")+
    #geom_text(data=temp_plot_data, aes(label=mean),hjust=-.2, vjust=.2)+
    labs(title="Accidental Drops")#, caption = "Means, 95% CI; Within-subjects T-test, adj.: Holm")+
    #stat_pvalue_manual(data=stat.test, xmin="group1", xmax="group2", label = "p.adj.signif", step.incr
drops_plot
```




```
# plot the difference data
diff_set <- subject_data_all_long %>% select(Interface, id, Drop_Count) %>% spread(Interface, Drop_Coun

ggplot(diff_set, aes(diff)) +
  geom_density(fill="grey")+labs(title="Difference data - Leaps (density)", x="Differences: Leap - HHI_
  geom_vline(aes(xintercept=mean(diff)))
```




```
cat("Leap diff data passes Shapiro test for normal distribution? ", (diff_set %>% shapiro_test(diff))$p
```

```
## Leap diff data passes Shapiro test for normal distribution?  TRUE
```

```
describe(diff_set$diff)
```

```
##    vars  n mean   sd median trimmed  mad min max range skew kurtosis   se
## X1    1 32 2.34 3.43      2    2.31 2.97  -5   9    14 0.15    -0.45 0.61
```

```
# transform for data output
stat.test <- stat.test %>% select(-.y., -n1, -n2, -Interface)%>% mutate(p=round(p, 4), p.adj=round(p.ad

#stat.test.anova <- stat.test.anova %>% mutate(p=round(p, 4))

# non-parametric
# stat.test.friedman <- subject_data_all_long %>% friedman_test(Drop_Count ~ Interface | id) %>% mutate
# stat.test.wilcox <- subject_data_all_long %>% wilcox_test(Drop_Count ~ Interface, paired=TRUE)
# stat.test.friedman
# stat.test.wilcox

anova.Interface.drops <- stat.test.anova
ttest.Interface.drops <- stat.test
descriptives.Interface.drops <- subject_data_all_long %>% group_by(Interface) %>% get_summary_stats(Drop
descriptives.Interface.drops
```

```
## # A tibble: 3 x 9
##   Interface variable    mean    sd median   mad   min   max   iqr
##   <fct>     <chr>      <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 B_Leap    Drop_Count 4.75  2.68       5  2.96     0    10     5
## 2 HHI_Leap  Drop_Count 2.41  2.09       2  2.96     0     8     3
## 3 Oculus    Drop_Count 0.125 0.336      0  0        0     1     0
```

```
oculus_diffs["Accidental drops","Difference"] <- round(temp_plot_data$mean[1]-temp_plot_data$mean[3], 3
oculus_diffs["Accidental drops","Type"] <- "Mean difference"
oculus_diffs["Accidental drops","cohen's d"] <- round(stat.test[3,"effsize"], 3)

# #print to file
# sink("results_all.txt", append = TRUE)
# cat("\n/Accidental drops/\n\nT-Test\n")
# print(as.data.frame(stat.test))
# cat("\nANOVA\n")
# print(stat.test.anova)
# cat("\n---\n")
# sink()
```

Cube size

Accuracy

```
temp_plot_data <- subject_data_cube_size %>%
  group_by(Interface, Cube_Size) %>%
  get_summary_stats(Distance)

stat.test <- subject_data_cube_size %>%
  group_by(Cube_Size) %>%
  pairwise_t_test(Distance ~ Interface, paired=TRUE, comparisons=list(c("B_Leap","HHI_Leap"),c("HHI_Lea
  left_join(subject_data_cube_size %>% ungroup(.) %>% cohens_d(Distance ~ Interface, paired=TRUE) %>% s
  mutate(Interface=group1) %>%
 left_join(subject_data_cube_size %>% group_by(Cube_Size) %>% summarise(y.position=max(Distance))) %>% 
  adjust_pvalue() %>% add_significance(p.col="p.adj", output.col="p.adj.signif", symbols=mysymbols)

stat.test
```

```
## # A tibble: 6 x 16
##   Cube_Size .y.   group1 group2    n1    n2 statistic    df       p    p.adj
##   <fct>     <chr> <chr>  <chr>  <int> <int>     <dbl> <dbl>   <dbl>    <dbl>
## 1 Small     Dist~ B_Leap HHI_L~    32    32    -0.638    31 5.28e-1 1.00e+0
## 2 Small     Dist~ HHI_L~ Oculus    32    32     5.54     31 4.57e-6 2.74e-5
## 3 Medium    Dist~ B_Leap HHI_L~    32    32     0.355    31 7.25e-1 1.00e+0
## 4 Medium    Dist~ HHI_L~ Oculus    32    32     4.75     31 4.33e-5 2.16e-4
## 5 Large     Dist~ B_Leap HHI_L~    32    32    -0.283    31 7.79e-1 1.00e+0
## 6 Large     Dist~ HHI_L~ Oculus    32    32     4.66     31 5.70e-5 2.28e-4
## # ... with 6 more variables: p.adj.signif <chr>, effsize <dbl>,
## #   magnitude <ord>, Interface <chr>, y.position <dbl>, rounded_p <dbl>
```

```
# anova - all interfaces
stat.test.anova <-
```

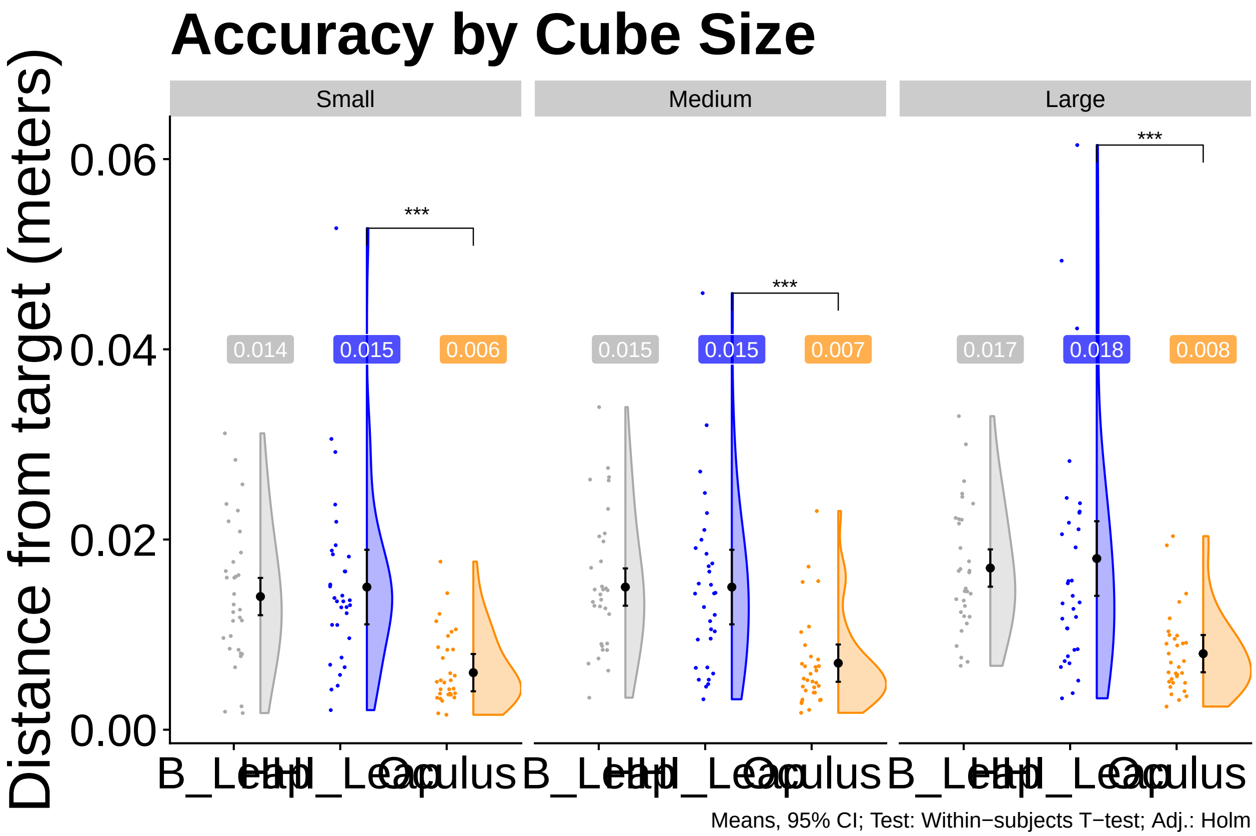
```r
anova_summary(effect.size="pes",aov(Distance ~ Interface*Cube_Size + Error(id/(Interface*Cube_Size)), da
stat.test.anova
```

```
##                Effect DFn DFd      F        p p<.05   pes
## 1           Interface   2  62 41.437 3.75e-12     * 0.572
## 2           Cube_Size   2  62  4.055 2.20e-02     * 0.116
## 3 Interface:Cube_Size   4 124  0.237 9.17e-01       0.008
```

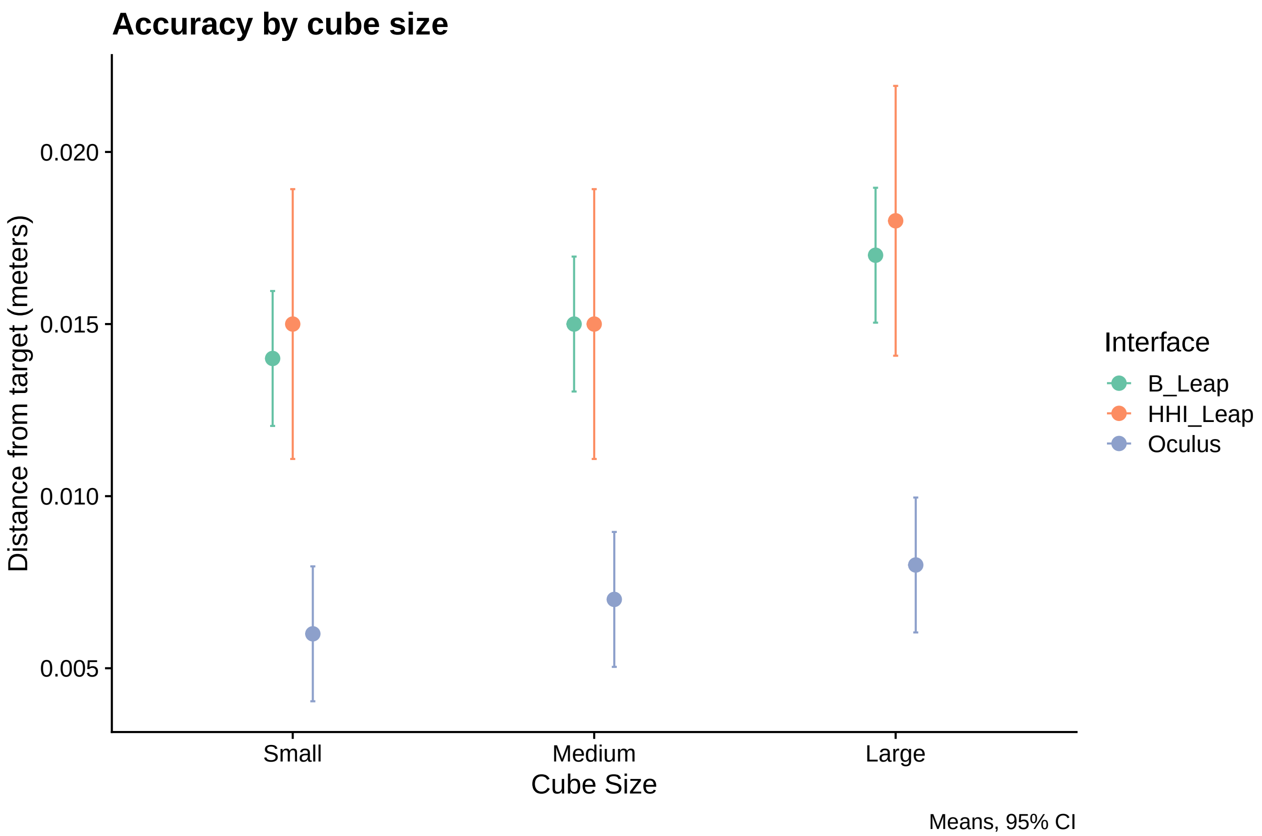
```r
#write.csv(stat.test.anova, file="accuracy_cubesize_anova.csv")

# anova (Leaps only)
stat.test.anova2 <-
anova_summary(effect.size="pes",aov(Distance ~ Interface*Cube_Size + Error(id/(Interface*Cube_Size)), da
stat.test.anova2
```

```
##                Effect DFn DFd     F     p p<.05   pes
## 1           Interface   1  31 0.106 0.747       0.003
## 2           Cube_Size   2  62 2.317 0.107       0.070
## 3 Interface:Cube_Size   2  62 0.221 0.803       0.007
```

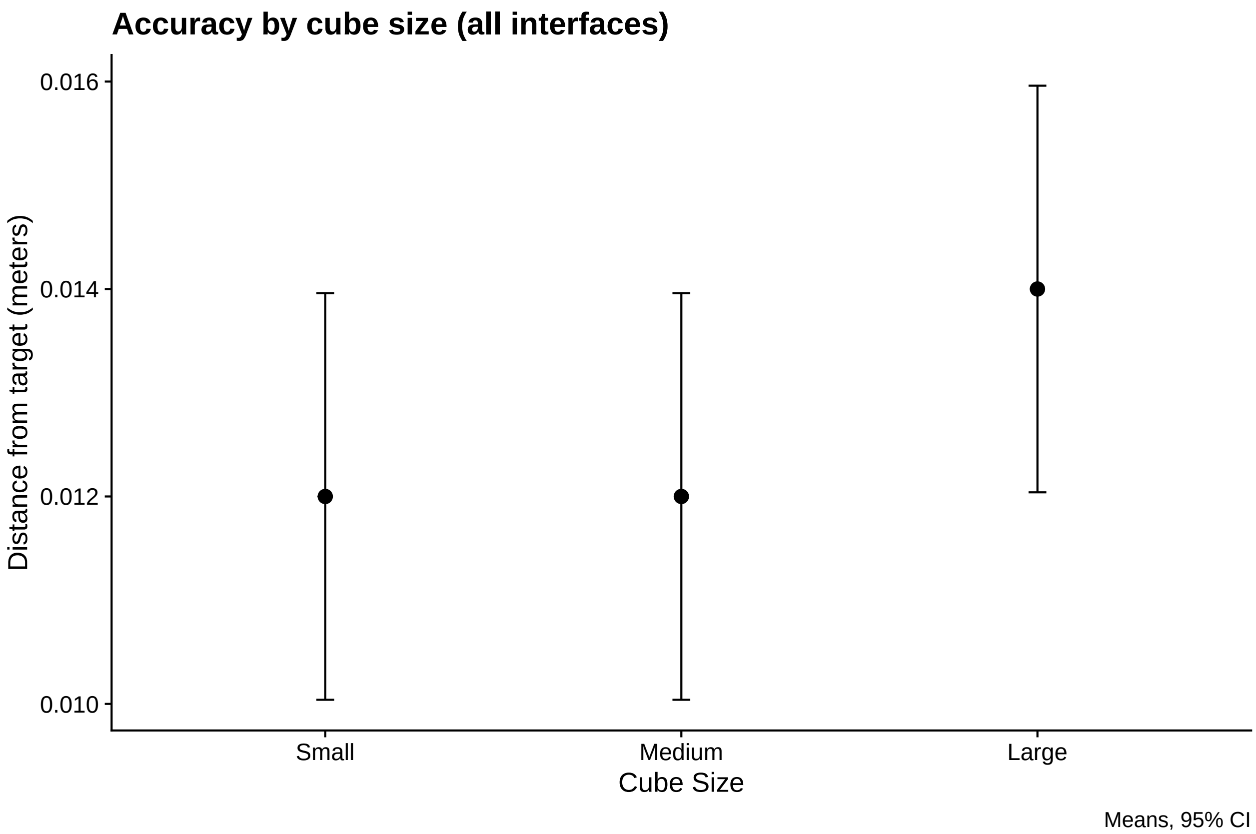
```r
#write.csv(stat.test.anova, file="accuracy_cubesize_anova.csv")

  # create plot
distance_cubesize_plot <-
  #ggplot(temp_plot_data, aes(x=Cube_Size, y=mean, color=Interface, group=Interface)) +
  ggplot(subject_data_cube_size, aes(Interface, Distance, color=Interface, fill=Interface))+
    # theme(legend.position = legend_pos, legend.title=element_text(size=legend_title_size),
    #       legend.text=element_text(size=legend_text_size),
    #       axis.text=element_text(size=axis_text_size),
    #       title = element_text(size=title_size, hjust=.5))+
    geom_point(position = position_jitter(width = .1, height=0), size = .25)+
    geom_violinhalf(position = position_nudge(x = .25, y = 0), alpha=myalpha, adjust = mysmoothing)+
    geom_point(data=temp_plot_data, aes(x=Interface, y=mean), position = position_nudge(.25), color="bla
    geom_errorbar(data=temp_plot_data, aes(x=Interface, y=mean, ymin=mean-(se*1.96), ymax=mean+(se*1.96)
    geom_label(data=temp_plot_data, aes(x=Interface, y=0.04, label=round(mean, 3)), color="white", posi
    theme_cowplot()+guides(fill=FALSE, color=FALSE)+#scale_x_discrete(labels=NULL)+
    #geom_point(aes(label=id), position=position_jitterdodge(jitter.width=.08, jitter.height=0.005, dod
    #scale_size_manual(values=c(10,6,3))+
    #geom_path(data=temp_plot_data, aes(x=Cube_Size, y=mean, group=Interface, color=Interface),size=.5,
    scale_color_manual(values=mycolors)+#scale_colour_brewer(palette = "Set2")+
    scale_fill_manual(values=mycolors)+#scale_fill_brewer(palette = "Set2", direction=1)+
    labs(title="Accuracy by Cube Size", caption="Means, 95% CI; Test: Within-subjects T-test; Adj.: Holm
    stat_pvalue_manual(data=stat.test %>% filter(p.adj < 0.05), label = "p.adj.signif", position=positio
    facet_grid(. ~ Cube_Size)+
  theme(plot.title = element_text(size=title_size), axis.title = element_text(size=axis_text_size), axis
  #  coord_cartesian(ylim=c(0.0035, 0.006))
distance_cubesize_plot
```




```r
  ggsave("accuracy_plot_cubesize.jpg", width=10, height=7)


ggplot(temp_plot_data, aes(Cube_Size, mean, color=Interface))+
  geom_point(size=3, position=position_dodge(.2))+theme_cowplot()+
  scale_color_brewer(palette="Set2")+geom_errorbar(aes(ymin=mean-(se*1.96), ymax=mean+(se*1.96)), width
  labs(title="Accuracy by cube size", caption="Means, 95% CI", y="Distance from target (meters)", x="Cu
```




```
ggsave("distance_cubesize_interface_plot.jpg")

ggplot(subject_data_cube_size %>% group_by(Cube_Size) %>% get_summary_stats(Distance), aes(Cube_Size, me
  geom_point(size=3, position=position_dodge(.2))+theme_cowplot()+
  scale_color_brewer(palette="Set2")+geom_errorbar(aes(ymin=mean-(se*1.96), ymax=mean+(se*1.96)), width=
  labs(title="Accuracy by cube size (all interfaces)", caption="Means, 95% CI", y="Distance from target
```




```
ggsave("distance_cubesize_main_effect_plot.jpg")


# transform for data output
stat.test <- stat.test %>% mutate(p=round(p, 4), p.adj=round(p.adj, 4), statistic=round(statistic, 3), 

stat.test.anova <- stat.test.anova %>% mutate(p=round(p, 4))

anova.Interface.cubesize.distance <- stat.test.anova
ttest.Interface.cubesize.distance <- stat.test
descriptives.Interface.cubesize.distance <- subject_data_cube_size %>% group_by(Interface, Cube_Size) %
descriptives.Interface.cubesize.distance
```

```
## # A tibble: 9 x 8
##   Interface Cube_Size variable  mean    sd   min   max   iqr
##   <fct>     <fct>     <chr>    <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 B_Leap    Small     Distance 0.014 0.007 0.002 0.031 0.009
## 2 B_Leap    Medium    Distance 0.015 0.007 0.003 0.034 0.011
## 3 B_Leap    Large     Distance 0.017 0.007 0.007 0.033 0.01
## 4 HHI_Leap  Small     Distance 0.015 0.01  0.002 0.053 0.008
## 5 HHI_Leap  Medium    Distance 0.015 0.009 0.003 0.046 0.01
## 6 HHI_Leap  Large     Distance 0.018 0.013 0.003 0.061 0.014
## 7 Oculus    Small     Distance 0.006 0.004 0.002 0.018 0.005
## 8 Oculus    Medium    Distance 0.007 0.005 0.002 0.023 0.004
## 9 Oculus    Large     Distance 0.008 0.004 0.002 0.02  0.005
```

####Grab time

```r
# Time from spawn to grab: dot plot of Interface * cube size
  # create descriptives

stat.test.anova <-
anova_summary(effect.size="pes",aov(grabtime ~ Interface*Cube_Size + Error(id/(Interface*Cube_Size)), da
stat.test.anova
```

```
##                Effect DFn DFd      F        p p<.05   pes
## 1           Interface   2  62 23.828 2.10e-08     * 0.435
## 2           Cube_Size   2  62 17.216 1.13e-06     * 0.357
## 3 Interface:Cube_Size   4 124  5.270 5.93e-04     * 0.145
```

```r
stat.test.anova2 <-
anova_summary(effect.size="pes",aov(grabtime ~ Interface*Cube_Size + Error(id/(Interface*Cube_Size)), da
stat.test.anova2
```

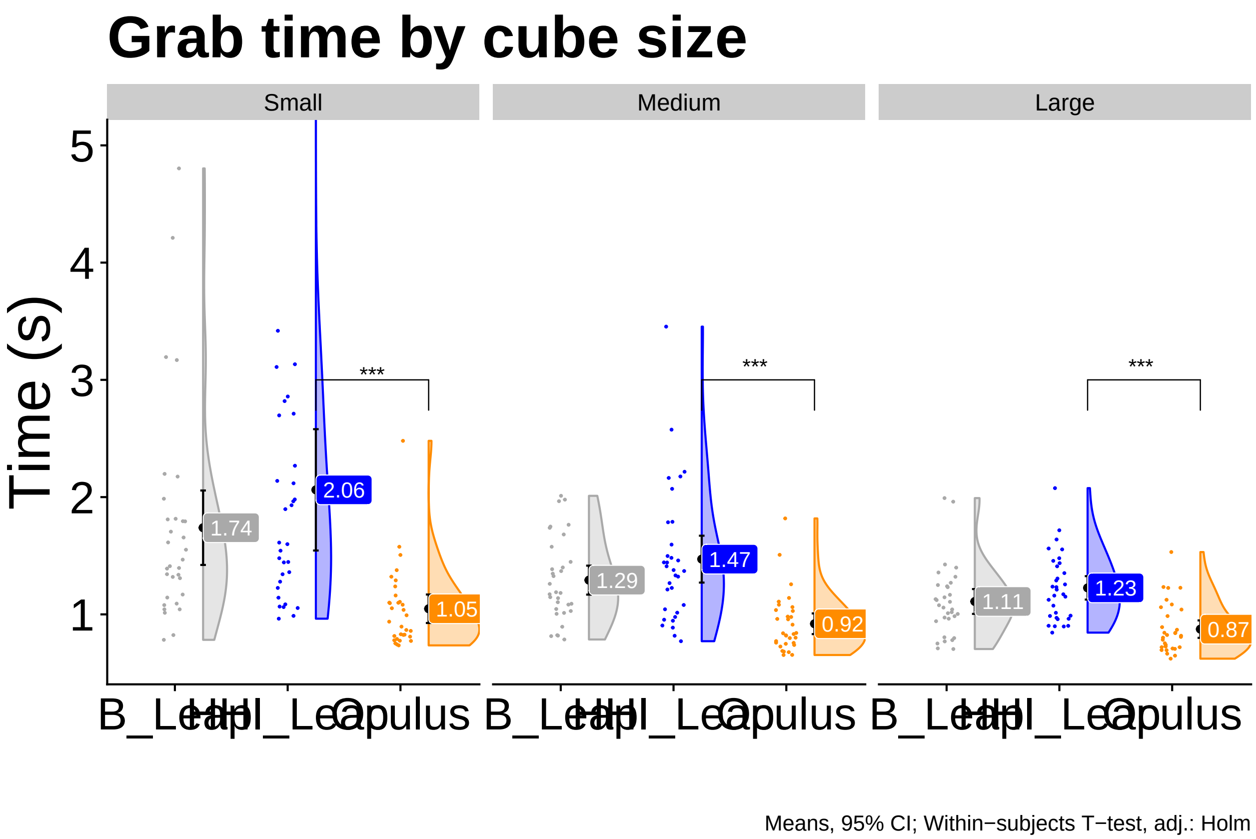
```
##                Effect DFn DFd      F        p p<.05   pes
## 1           Interface   1  31  3.434 7.30e-02       0.100
## 2           Cube_Size   2  62 16.085 2.36e-06     * 0.342
## 3 Interface:Cube_Size   2  62  0.814 4.48e-01       0.026
```

```r
#write.csv(stat.test.anova, file="grabtime_cubesize_anova.csv")

stat.test <- subject_data_cube_size %>%
  group_by(Cube_Size) %>%
  pairwise_t_test(grabtime ~ Interface, paired=TRUE, comparisons=list(c("B_Leap","HHI_Leap"),c("HHI_Leap
   left_join(subject_data_cube_size %>% ungroup(.) %>% cohens_d(grabtime ~ Interface, paired=TRUE) %>% s
  mutate(Interface=group1, y.position=3, rounded_p=round(p.adj, 4)) %>%
  adjust_pvalue() %>% add_significance(p.col="p.adj", output.col="p.adj.signif", symbols=mysymbols) #%>%
stat.test
```

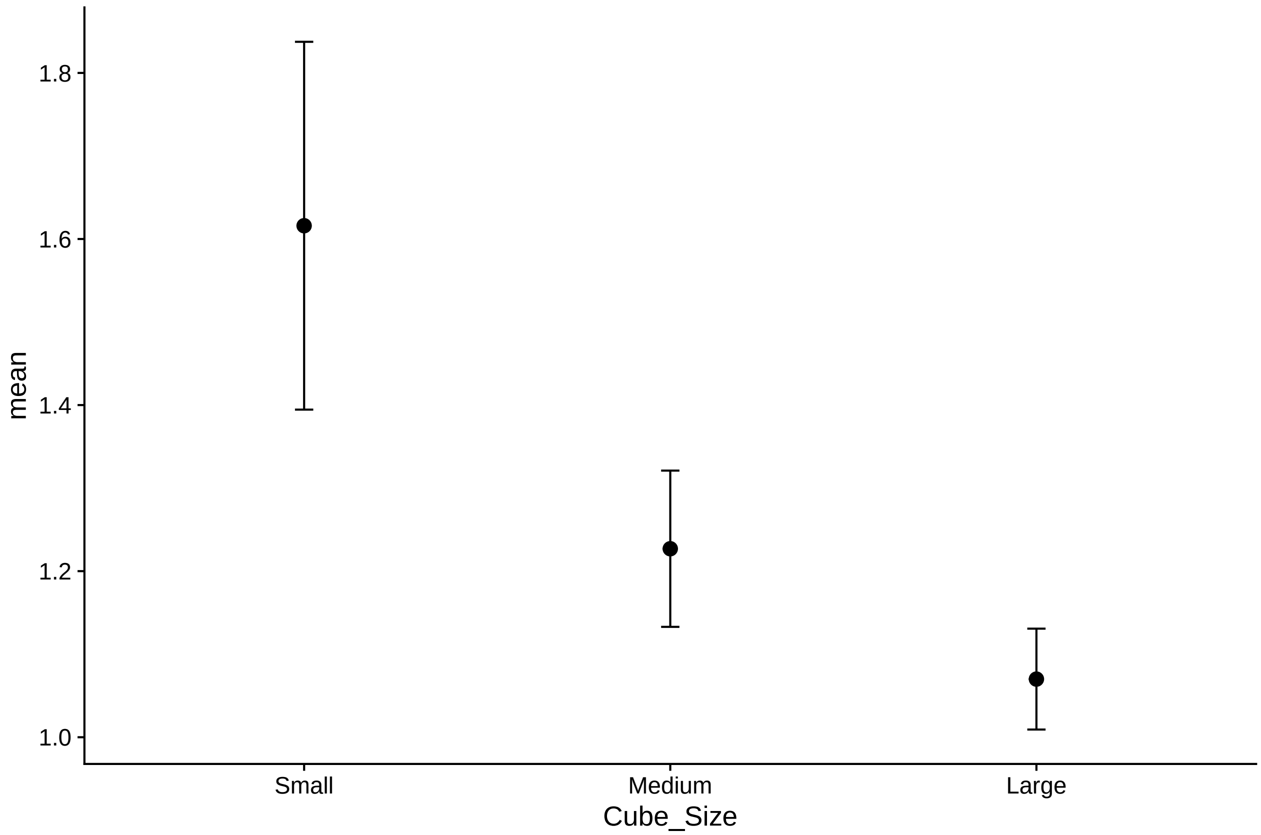
```
## # A tibble: 6 x 16
##   Cube_Size .y.   group1 group2    n1    n2 statistic    df       p   p.adj
##   <fct>     <chr> <chr>  <chr>  <int> <int>     <dbl> <dbl>   <dbl>   <dbl>
## 1 Small     grab~ B_Leap HHI_L~    32    32     -1.43    31 1.62e-1 1.84e-1
## 2 Small     grab~ HHI_L~ Oculus    32    32      4.58    31 7.21e-5 2.88e-4
## 3 Medium    grab~ B_Leap HHI_L~    32    32     -1.74    31 9.20e-2 1.84e-1
## 4 Medium    grab~ HHI_L~ Oculus    32    32      7.09    31 5.72e-8 3.43e-7
## 5 Large     grab~ B_Leap HHI_L~    32    32     -2.01    31 5.40e-2 1.62e-1
## 6 Large     grab~ HHI_L~ Oculus    32    32      6.73    31 1.57e-7 7.85e-7
## # ... with 6 more variables: p.adj.signif <chr>, effsize <dbl>,
## #   magnitude <ord>, Interface <chr>, y.position <dbl>, rounded_p <dbl>
```

```r
temp_plot_data <- subject_data_cube_size %>%
  group_by(Interface, Cube_Size) %>%
  get_summary_stats(grabtime)

  # create plot
  grabtime_cubesize_plot <-
    ggplot(temp_plot_data, aes(x=Interface, y=mean, color=Interface, fill=Interface)) +
    theme_cowplot() +
    # theme(legend.position = legend_pos, legend.title=element_text(size=legend_title_size),
```

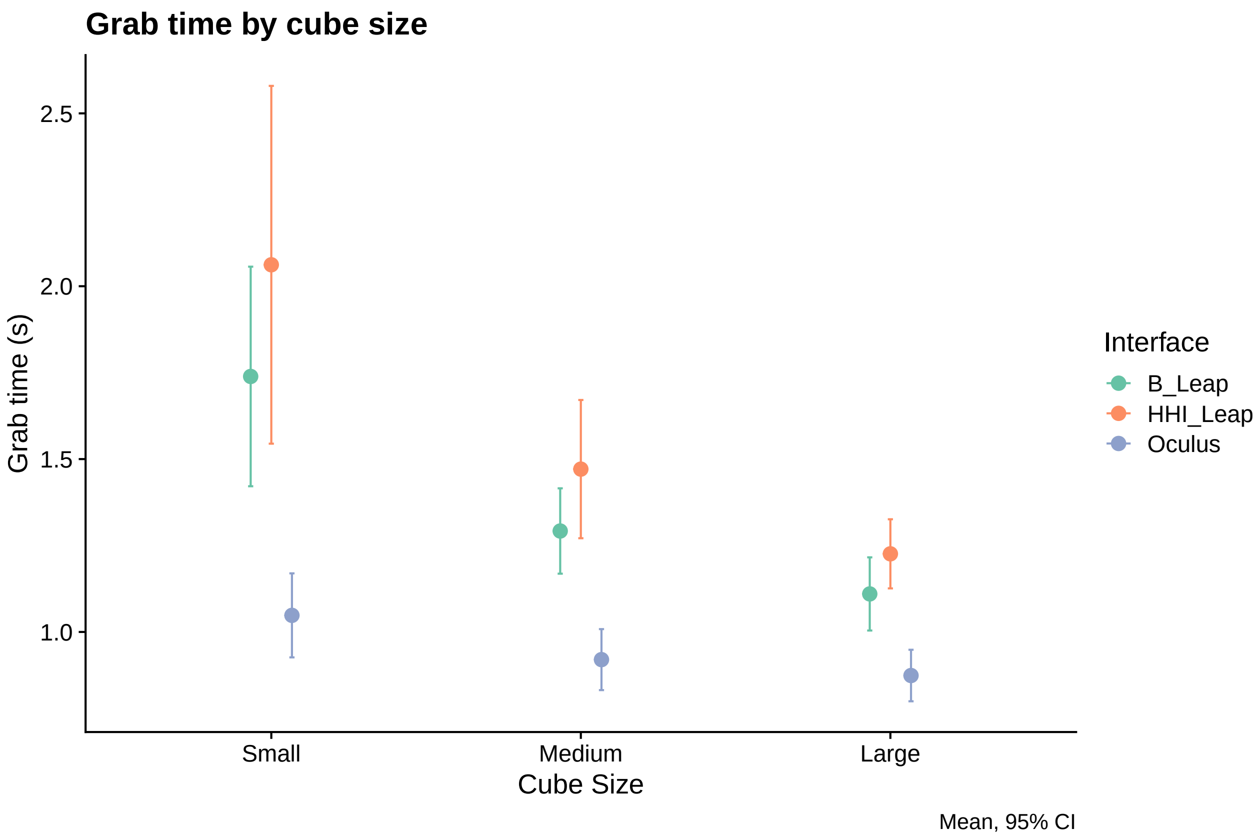
```
#       legend.text=element_text(size=legend_text_size),
#       axis.text=element_text(size=axis_text_size),
#       title = element_text(size=title_size, hjust=.5)) +
  geom_point(data=subject_data_cube_size, aes(Interface, grabtime), position = position_jitter(width =
  geom_violinhalf(data=subject_data_cube_size, aes(Interface, grabtime), position = position_nudge(x =
  geom_point(position = position_nudge(.25), color="black")+#shape=15)+
  geom_errorbar(aes(ymin=mean-(se*1.96), ymax=mean+(se*1.96)), color="black", width=.05, size=.5, posi
  geom_label(aes(label=round(mean, 2), y=mean), color="white", position=position_nudge(.5))+
  scale_color_manual(values=mycolors)+#scale_colour_brewer(palette = "Set2")+
  scale_fill_manual(values=mycolors)+#scale_fill_brewer(palette = "Set2", direction=1)+
  labs(title="Grab time by cube size", caption="Means, 95% CI; Within-subjects T-test, adj.: Holm",
       y="Time (s)",
       x="")+
  guides(fill=FALSE, color=FALSE)+
  stat_pvalue_manual(data=stat.test %>% filter(p.adj < 0.05), label = "p.adj.signif", position=positi
  coord_cartesian(ylim=c(min(subject_data_cube_size$grabtime), 5))+
  facet_grid(. ~ Cube_Size)+#, scales="free", space="free", shrink=TRUE)+
  theme(plot.title = element_text(size=title_size), axis.title = element_text(size=axis_text_size), ax
grabtime_cubesize_plot
```




```
ggsave("grabtime_plot_cubesize.jpg", width=10, height=7)



ggplot(subject_data_cube_size%>%group_by(Cube_Size)%>%get_summary_stats(grabtime), aes(Cube_Size, mean)
  geom_point(size=3)+theme_cowplot()+scale_color_brewer(palette="Set2")+geom_errorbar(aes(ymin=mean-(se
```




```
ggplot(subject_data_cube_size%>%group_by(Cube_Size,Interface)%>%get_summary_stats(grabtime), aes(Cube_Si
  geom_point(size=3, position=position_dodge(.2))+theme_cowplot()+scale_color_brewer(palette="Set2")+ge
```




```
ggsave("grabtime_cubesize_plot.jpg")

# transform for data output
stat.test <- stat.test %>% mutate(p=round(p, 4), p.adj=round(p.adj, 4), statistic=round(statistic, 3), 

stat.test.anova <- stat.test.anova %>% mutate(p=round(p, 4))

anova.Interface.cubesize.grabtime <- stat.test.anova
ttest.Interface.cubesize.grabtime <- stat.test
descriptives.Interface.cubesize.grabtime <- subject_data_cube_size %>% group_by(Interface, Cube_Size) %
descriptives.Interface.cubesize.grabtime
```

```
## # A tibble: 9 x 8
##   Interface Cube_Size variable  mean    sd   min   max   iqr
##   <fct>     <fct>     <chr>    <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 B_Leap    Small     grabtime 1.74  0.917 0.783  4.80 0.649
## 2 B_Leap    Medium    grabtime 1.29  0.357 0.786  2.01 0.439
## 3 B_Leap    Large     grabtime 1.11  0.304 0.704  1.99 0.284
## 4 HHI_Leap  Small     grabtime 2.06  1.49  0.963  9.24 1.11
## 5 HHI_Leap  Medium    grabtime 1.47  0.579 0.771  3.45 0.607
## 6 HHI_Leap  Large     grabtime 1.23  0.287 0.844  2.08 0.434
## 7 Oculus    Small     grabtime 1.05  0.349 0.735  2.48 0.306
## 8 Oculus    Medium    grabtime 0.92  0.253 0.653  1.82 0.276
## 9 Oculus    Large     grabtime 0.874 0.214 0.622  1.53 0.279
```

Release time

```r
# Time from grab to release: dot plot of Interface * cube size
  # create descriptives
stat.test.anova <-
anova_summary(effect.size="pes", aov(releasetime ~ Interface*Cube_Size + Error(id/(Interface*Cube_Size)
stat.test.anova
```

```
##                Effect DFn DFd      F        p p<.05   pes
## 1           Interface   2  62 32.023 2.80e-10     * 0.508
## 2           Cube_Size   2  62  0.884 4.18e-01       0.028
## 3 Interface:Cube_Size   4 124  9.721 7.15e-07     * 0.239
```

```r
#write.csv(stat.test.anova, file="releasetime_cubesize_anova.csv")

stat.test.anova2 <-
anova_summary(effect.size="pes",aov(releasetime ~ Interface*Cube_Size + Error(id/(Interface*Cube_Size))
stat.test.anova2
```

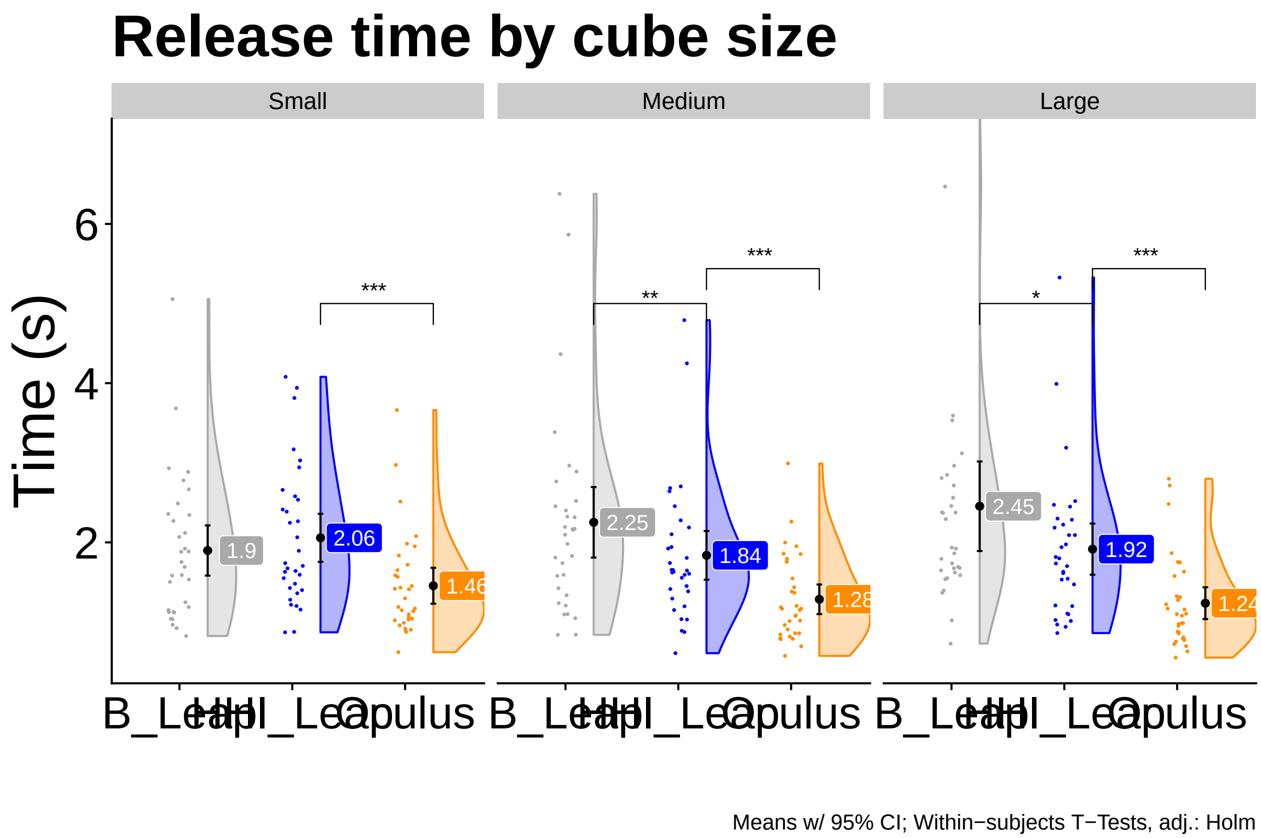
```
##                Effect DFn DFd      F        p p<.05   pes
## 1           Interface   1  31  5.727 2.30e-02     * 0.156
## 2           Cube_Size   2  62  2.465 9.30e-02       0.074
## 3 Interface:Cube_Size   2  62 11.194 7.07e-05     * 0.265
```

```r
stat.test <- subject_data_cube_size %>%
  group_by(Cube_Size) %>%
  pairwise_t_test(releasetime ~ Interface, paired=TRUE, comparisons=list(c("B_Leap","HHI_Leap"),c("HHI_
   left_join(subject_data_cube_size %>% ungroup(.) %>% cohens_d(releasetime ~ Interface, paired=TRUE) %
  mutate(Interface=group1, y.position=5) %>%
  adjust_pvalue() %>% add_significance(p.col="p.adj", output.col="p.adj.signif", symbols=mysymbols)#%>%

stat.test
```

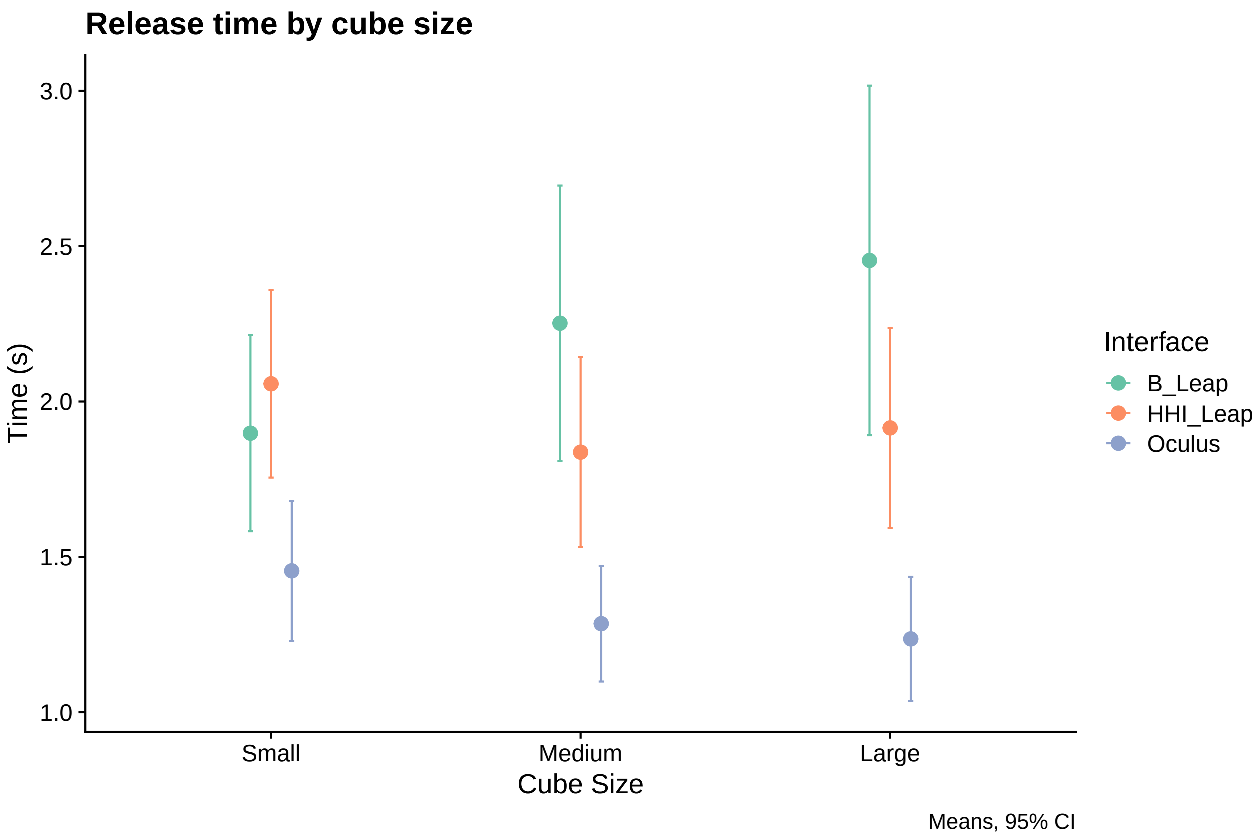
```
## # A tibble: 6 x 15
##   Cube_Size .y.   group1 group2    n1    n2 statistic    df       p   p.adj
##   <fct>     <chr> <chr>  <chr>  <int> <int>     <dbl> <dbl>   <dbl>   <dbl>
## 1 Small     rele~ B_Leap HHI_L~    32    32     -1.51    31 1.42e-1 1.42e-1
## 2 Small     rele~ HHI_L~ Oculus    32    32      5.50    31 5.16e-6 2.10e-5
## 3 Medium    rele~ B_Leap HHI_L~    32    32      3.42    31 2.00e-3 6.00e-3
## 4 Medium    rele~ HHI_L~ Oculus    32    32      5.57    31 4.21e-6 2.10e-5
## 5 Large     rele~ B_Leap HHI_L~    32    32      2.85    31 8.00e-3 1.60e-2
## 6 Large     rele~ HHI_L~ Oculus    32    32      6.26    31 5.95e-7 3.57e-6
## # ... with 5 more variables: p.adj.signif <chr>, effsize <dbl>,
## #   magnitude <ord>, Interface <chr>, y.position <dbl>
```

```r
  temp_plot_data <- subject_data_cube_size %>%
    group_by(Interface, Cube_Size) %>%
    get_summary_stats(releasetime)

  releasetime_cubesize_plot <-
    ggplot(temp_plot_data, aes(x=Interface, y=mean, color=Interface, fill=Interface)) +
    theme_cowplot() +
    geom_point(data=subject_data_cube_size, aes(Interface, releasetime), position = position_jitter(wid
    geom_violinhalf(data=subject_data_cube_size, aes(Interface, releasetime), position = position_nudge
```

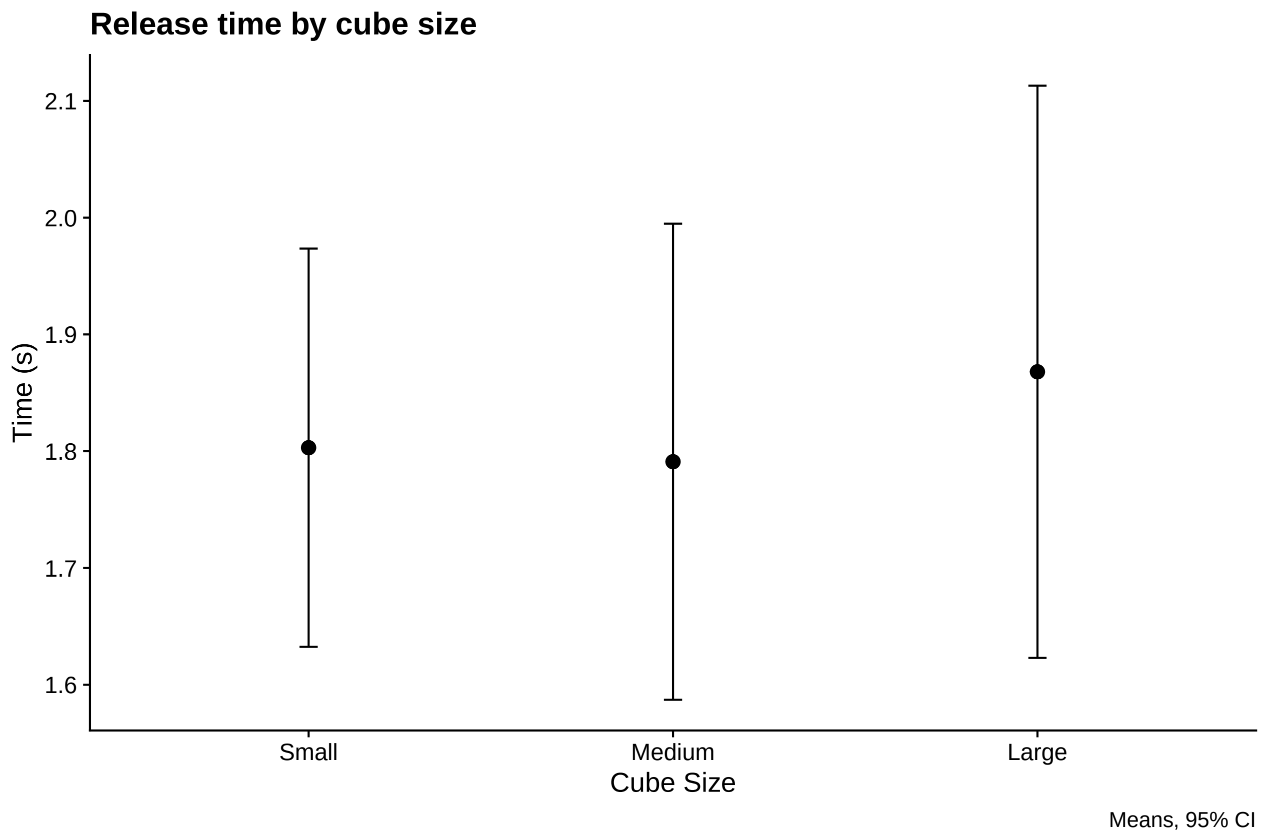
```r
    geom_point(position = position_nudge(.25), color="black")+#shape=15)+
    geom_errorbar(aes(ymin=mean-(se*1.96), ymax=mean+(se*1.96)), color="black", width=.05, size=.5, posi
    geom_label(aes(label=round(mean, 2), y=mean), color="white", position=position_nudge(.55))+
    scale_color_manual(values=mycolors)+#scale_colour_brewer(palette = "Set2")+
    scale_fill_manual(values=mycolors)+#scale_fill_brewer(palette = "Set2", direction=1)+
    labs(title="Release time by cube size", y="Time (s)", x="", caption="Means w/ 95% CI; Within-subject
    stat_pvalue_manual(data=stat.test %>% filter(p.adj < 0.05), label = "p.adj.signif", position=positi
    facet_grid(. ~ Cube_Size)+#, scales="free", space="free", shrink=TRUE)
    guides(fill=FALSE, color=FALSE)+
    theme(plot.title = element_text(size=title_size), axis.title = element_text(size=axis_text_size), a
    releasetime_cubesize_plot
```




```r
    ggsave("releasetime_plot_cubesize.jpg", width=10, height=7)

# simplified cube size plots
ggplot(temp_plot_data, aes(Cube_Size, mean, color=Interface))+
  geom_point(size=3, position=position_dodge(.2))+theme_cowplot()+
  scale_color_brewer(palette="Set2")+geom_errorbar(aes(ymin=mean-(se*1.96), ymax=mean+(se*1.96)), width
  labs(title="Release time by cube size", caption="Means, 95% CI", y="Time (s)", x="Cube Size")
```




```
ggsave("releasetime_cubesize_interface_plot.jpg")

ggplot(subject_data_cube_size %>% group_by(Cube_Size) %>% get_summary_stats(releasetime), aes(Cube_Size
  geom_point(size=3, position=position_dodge(.2))+theme_cowplot()+
  scale_color_brewer(palette="Set2")+geom_errorbar(aes(ymin=mean-(se*1.96), ymax=mean+(se*1.96)), width
  labs(title="Release time by cube size", caption="Means, 95% CI", y="Time (s)", x="Cube Size", x="Cube
```




```
ggsave("releasetime_cubesize_main_effect_plot.jpg")

# transform for data output
stat.test <- stat.test %>% mutate(p=round(p, 4), p.adj=round(p.adj, 4), statistic=round(statistic, 3), 

stat.test.anova <- stat.test.anova %>% mutate(p=round(p, 4))

anova.Interface.cubesize.releasetime <- stat.test.anova
ttest.Interface.cubesize.releasetime <- stat.test
descriptives.Interface.cubesize.releasetime <- subject_data_cube_size %>% group_by(Interface, Cube_Size
descriptives.Interface.cubesize.releasetime
```

```
## # A tibble: 9 x 8
##   Interface Cube_Size variable     mean    sd   min   max   iqr
##   <fct>     <fct>     <chr>       <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 B_Leap    Small     releasetime  1.90 0.913 0.824  5.06 1.20
## 2 B_Leap    Medium    releasetime  2.25 1.28  0.839  6.38 1.07
## 3 B_Leap    Large     releasetime  2.45 1.62  0.728  9.31 1.10
## 4 HHI_Leap  Small     releasetime  2.06 0.869 0.87   4.08 1.13
## 5 HHI_Leap  Medium    releasetime  1.84 0.883 0.608  4.79 0.761
## 6 HHI_Leap  Large     releasetime  1.92 0.928 0.861  5.33 1.03
## 7 Oculus    Small     releasetime  1.46 0.65  0.62   3.66 0.639
## 8 Oculus    Medium    releasetime  1.28 0.538 0.575  2.99 0.741
## 9 Oculus    Large     releasetime  1.24 0.578 0.552  2.80 0.544
```

Total time

```r
# Time: total: dot plot of Interface * cube size

stat.test.anova <-
anova_summary(effect.size="pes",aov(totaltime ~ Interface*Cube_Size + Error(id/(Interface*Cube_Size)), d
stat.test.anova
```

```
##                Effect DFn DFd      F        p p<.05   pes
## 1           Interface   2  62 35.794 4.63e-11     * 0.536
## 2           Cube_Size   2  62 14.115 8.88e-06     * 0.313
## 3 Interface:Cube_Size   4 124  5.232 6.29e-04     * 0.144
```

```r
stat.test.anova2 <-
anova_summary(effect.size="pes",aov(totaltime ~ Interface*Cube_Size + Error(id/(Interface*Cube_Size)), d
stat.test.anova2
```

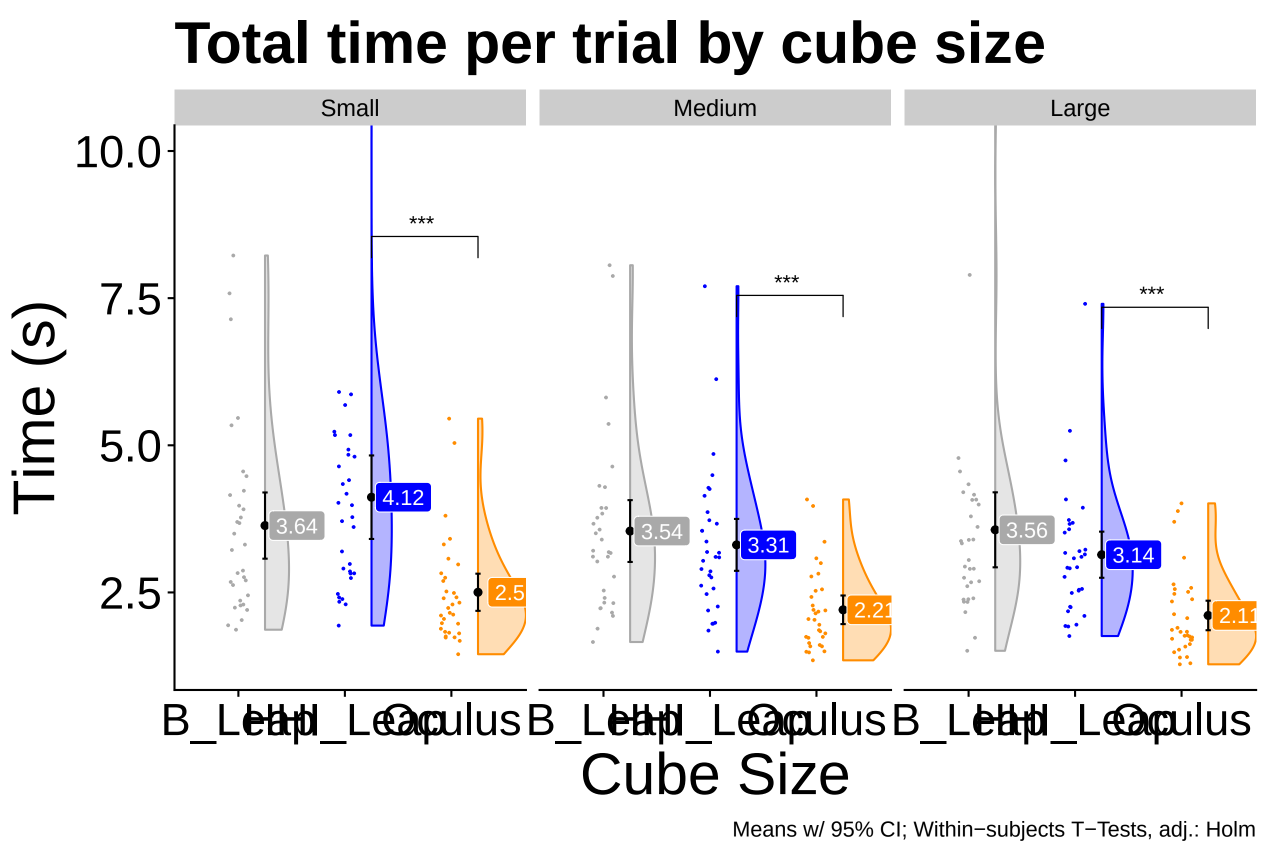
```
##                Effect DFn DFd     F        p p<.05   pes
## 1           Interface   1  31 0.109 0.743000       0.004
## 2           Cube_Size   2  62 8.524 0.000536     * 0.216
## 3 Interface:Cube_Size   2  62 7.212 0.002000     * 0.189
```

```r
#write.csv(stat.test.anova, file="totaltime_cubesize_anova.csv")

stat.test <- subject_data_cube_size %>%
  group_by(Cube_Size) %>%
  pairwise_t_test(totaltime ~ Interface, paired=TRUE, comparisons=list(c("B_Leap","HHI_Leap"),c("HHI_Lea
   left_join(subject_data_cube_size %>% ungroup(.) %>% cohens_d(totaltime ~ Interface, paired=TRUE) %>%
  mutate(Interface=group1) %>% left_join(subject_data_cube_size %>% group_by(Cube_Size) %>% summarise(y
  adjust_pvalue() %>% add_significance(p.col="p.adj", output.col="p.adj.signif", symbols=mysymbols)
stat.test
```

```
## # A tibble: 6 x 15
##   Cube_Size .y.   group1 group2    n1    n2 statistic    df       p   p.adj
##   <fct>     <chr> <chr>  <chr>  <int> <int>     <dbl> <dbl>   <dbl>   <dbl>
## 1 Small     tota~ B_Leap HHI_L~    32    32     -1.67    31 1.06e-1 2.12e-1
## 2 Small     tota~ HHI_L~ Oculus    32    32      5.89    31 1.68e-6 6.72e-6
## 3 Medium    tota~ B_Leap HHI_L~    32    32      1.50    31 1.43e-1 2.12e-1
## 4 Medium    tota~ HHI_L~ Oculus    32    32      7.68    31 1.16e-8 6.96e-8
## 5 Large     tota~ B_Leap HHI_L~    32    32      1.88    31 6.90e-2 2.07e-1
## 6 Large     tota~ HHI_L~ Oculus    32    32      7.21    31 4.20e-8 2.10e-7
## # ... with 5 more variables: p.adj.signif <chr>, effsize <dbl>,
## #   magnitude <ord>, Interface <chr>, y.position <dbl>
```

```r
grand_stats <- subject_data_cube_size %>%
  group_by(id, Cube_Size) %>% get_summary_stats(totaltime, type="common") %>% group_by(Cube_Size) %>% ge

grand_stats2 <- unity_data_clean %>% group_by(id, Cube_Size) %>% #summarise(mean=mean(TimeFromSpawnToGr
  get_summary_stats(TimeFromSpawnToGrabLoss, type="common") %>% group_by(Cube_Size) %>%
  get_summary_stats(mean, type="common")

  temp_plot_data <- subject_data_cube_size %>%
    group_by(Interface, Cube_Size) %>%
    get_summary_stats(totaltime)
```

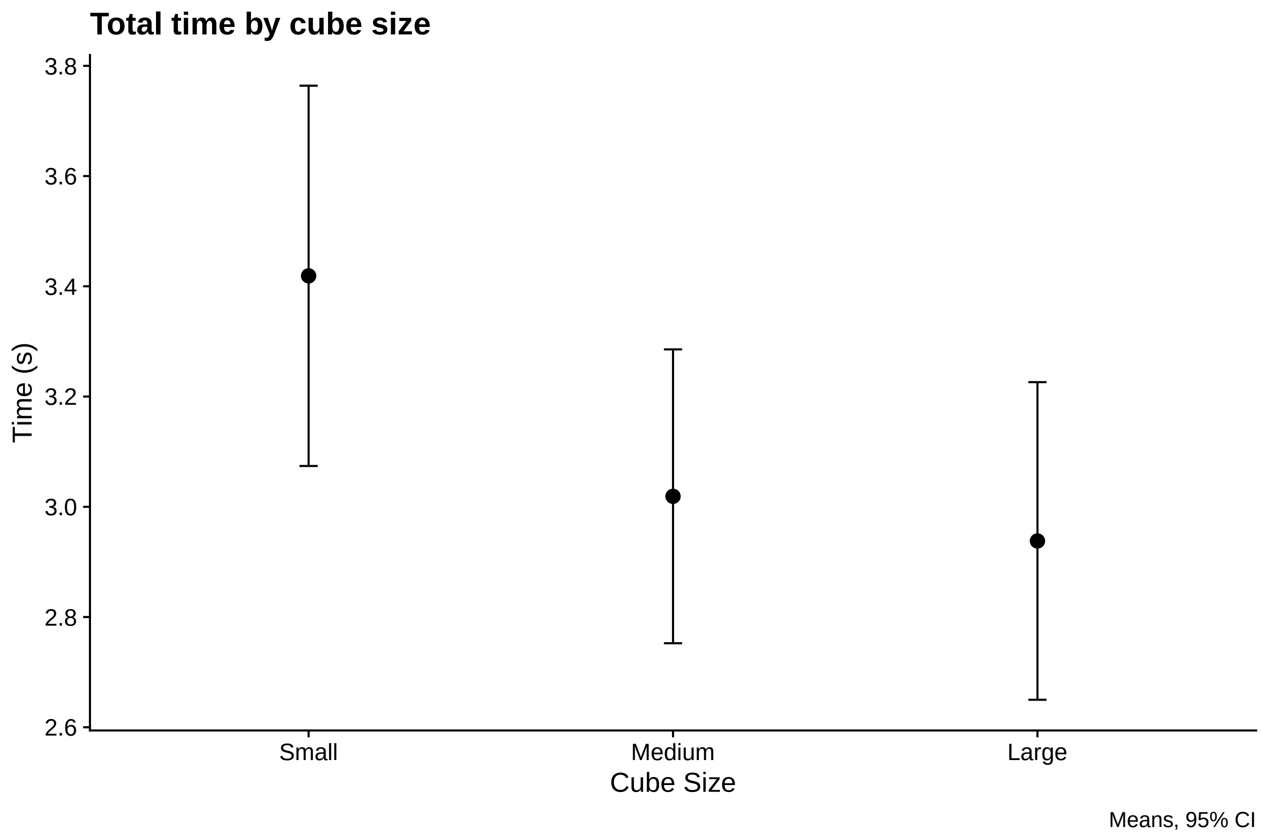
```r
totaltime_cubesize_plot <-
  ggplot(temp_plot_data, aes(x=Interface, y=mean, color=Interface, fill=Interface)) +
  theme_cowplot() +
  # theme(legend.position = legend_pos, legend.title=element_text(size=legend_title_size),
  #       legend.text=element_text(size=legend_text_size),
  #       axis.text=element_text(size=axis_text_size),
  #       title = element_text(size=title_size, hjust=.5)) +
  geom_point(data=subject_data_cube_size, aes(Interface, totaltime), position = position_jitter(width
  geom_violinhalf(data=subject_data_cube_size, aes(Interface, totaltime), position = position_nudge(x
  geom_point(position = position_nudge(.25), color="black")+#shape=15)+
  geom_errorbar(aes(ymin=mean-(se*1.96), ymax=mean+(se*1.96)), color="black", width=.05, size=.5, posi
  geom_label(aes(label=round(mean, 2), y=mean), color="white", position=position_nudge(.55))+
  scale_color_manual(values=mycolors)+#scale_colour_brewer(palette = "Set2")+
  scale_fill_manual(values=mycolors)+#scale_fill_brewer(palette = "Set2", direction=1)+
  labs(title="Total time per trial by cube size", y="Time (s)", x="Cube Size", caption="Means w/ 95% C
  guides(fill=FALSE, color=FALSE)+
  stat_pvalue_manual(data=stat.test %>% filter(p.adj < 0.05), label = "p.adj.signif", position=positio
  facet_grid(. ~ Cube_Size)+#, scales="free", space="free", shrink=TRUE)
  theme(plot.title = element_text(size=title_size), axis.title = element_text(size=axis_text_size), ax
totaltime_cubesize_plot
```




```r
ggsave("totaltime_plot_cubesize.jpg", width=10, height=7)


# simplified cube size plots
  #ggplot(temp_plot_data, aes(Interface, mean, color=Interface))+
```

```
ggplot(temp_plot_data, aes(Cube_Size, mean, color=Interface))+
  #facet_grid(. ~ Cube_Size)+
  geom_point(size=3, position=position_dodge(.2))+theme_cowplot()+
  scale_color_brewer(palette="Set2")+geom_errorbar(aes(ymin=mean-(se*1.96), ymax=mean+(se*1.96)), width=
  labs(title="Total time per trial by cube size", caption="Means, 95% CI", y="Time (s)", x="Cube Size")+
  #stat_pvalue_manual(data=stat.test, label = "p.adj.signif", step.increase = .05, step.group.by = "Cub
  ggsave("totaltime_cubesize_interface_plot.jpg")
```


```
ggplot(subject_data_cube_size %>% group_by(Cube_Size) %>% get_summary_stats(totaltime), aes(Cube_Size, m
  geom_point(size=3, position=position_dodge(.2))+theme_cowplot()+
  scale_color_brewer(palette="Set2")+geom_errorbar(aes(ymin=mean-(se*1.96), ymax=mean+(se*1.96)), width=
  labs(title="Total time by cube size", caption="Means, 95% CI", y="Time (s)", x="Cube Size", x="Cube Si
ggsave("totaltime_cubesize_main_effect_plot.jpg")
```




```
# transform for data output
stat.test <- stat.test %>% mutate(p=round(p, 4), p.adj=round(p.adj, 4), statistic=round(statistic, 3),

stat.test.anova <- stat.test.anova %>% mutate(p=round(p, 4))

anova.Interface.cubesize.totaltime <- stat.test.anova
ttest.Interface.cubesize.totaltime <- stat.test
descriptives.Interface.cubesize.totaltime <-
  subject_data_cube_size %>% group_by(Interface, Cube_Size) %>% get_summary_stats(totaltime) %>%
  select(Interface, Cube_Size, variable, mean, sd, min, max, iqr)
descriptives.Interface.cubesize.totaltime
```

```
## # A tibble: 9 x 8
##   Interface Cube_Size variable   mean    sd   min   max   iqr
##   <fct>     <fct>     <chr>     <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 B_Leap    Small     totaltime  3.64 1.62   1.87  8.22 1.74
## 2 B_Leap    Medium    totaltime  3.54 1.52   1.66  8.06 1.55
## 3 B_Leap    Large     totaltime  3.56 1.84   1.51 11.3  1.52
## 4 HHI_Leap  Small     totaltime  4.12 2.05   1.94 13.3  2.04
## 5 HHI_Leap  Medium    totaltime  3.31 1.27   1.50  7.70 1.22
## 6 HHI_Leap  Large     totaltime  3.14 1.13   1.76  7.40 1.16
## 7 Oculus    Small     totaltime  2.50 0.911  1.45  5.45 0.899
## 8 Oculus    Medium    totaltime  2.20 0.701  1.35  4.08 0.827
## 9 Oculus    Large     totaltime  2.11 0.724  1.28  4.01 0.808
```

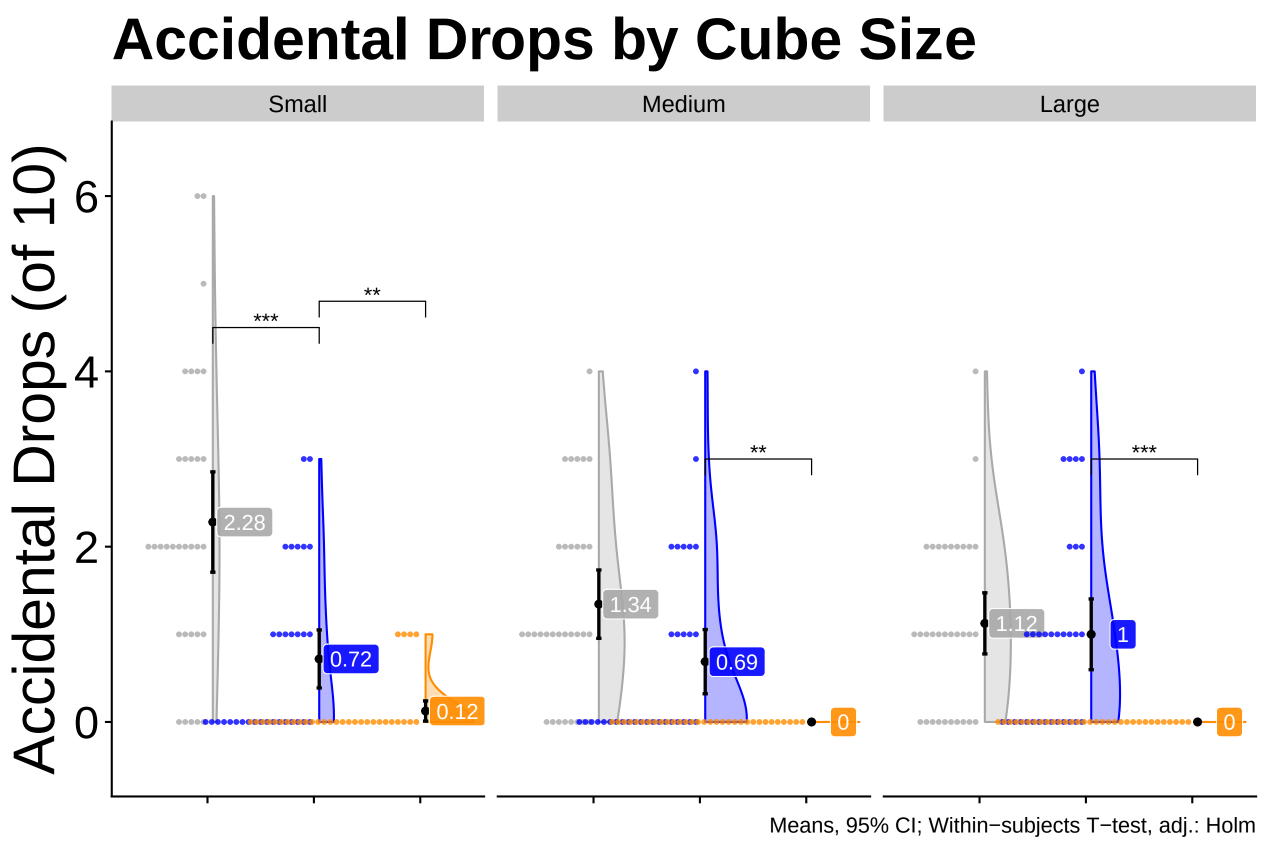
Accidental drops

```r
# Accidental drops: total: dot plot of Interface * cube size
  temp_plot_data <- subject_data_cube_size %>%
    group_by(Interface, Cube_Size) %>%
    get_summary_stats(Drop_Count)

  #print("ANOVA: accidental drops - Interface*cube size")
  # summary(aov(Drop_Count ~ Interface*Cube_Size + Error(id/(Interface*Cube_Size)), data=subject_data_c
  stat.test.anova <- anova_summary(effect.size="pes",aov(Drop_Count ~ Interface*Cube_Size + Error(id/(I

    stat.test.anova2<-anova_summary(effect.size="pes",aov(Drop_Count ~ Interface*Cube_Size + Error(id/(

 # write.csv(stat.test.anova, "accidentaldrops_cubesize_anova.csv")

stat.test <- subject_data_cube_size %>%
  group_by(Cube_Size) %>%
pairwise_t_test(Drop_Count ~ Interface, paired=TRUE, comparisons=list(c("B_Leap","HHI_Leap"),c("HHI_Lea
   left_join(subject_data_cube_size %>% ungroup(.) %>% cohens_d(Drop_Count ~ Interface, paired=TRUE) %>
  mutate(Interface=group1) %>%
  left_join(subject_data_cube_size %>% group_by(Cube_Size) %>% summarise(y.position=.75*max(Drop_Count)
  adjust_pvalue() %>% add_significance(p.col="p.adj", output.col="p.adj.signif", symbols=mysymbols)
stat.test
```

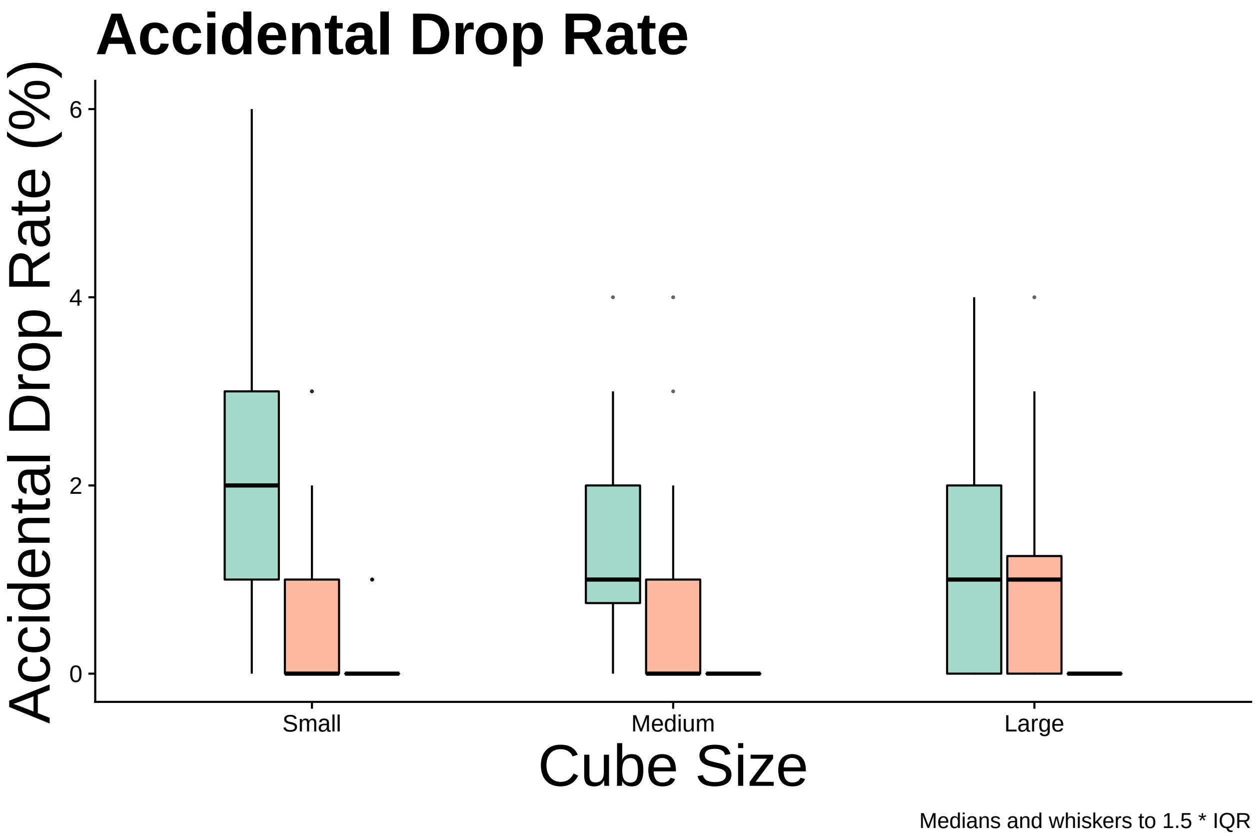
```
## # A tibble: 6 x 15
##   Cube_Size .y.   group1 group2    n1    n2 statistic    df       p   p.adj
##   <fct>     <chr> <chr>  <chr>  <int> <int>     <dbl> <dbl>   <dbl>   <dbl>
## 1 Small     Drop~ B_Leap HHI_L~    32    32     4.32     31 1.50e-4 7.50e-4
## 2 Small     Drop~ HHI_L~ Oculus    32    32     3.69     31 8.63e-4 3.45e-3
## 3 Medium    Drop~ B_Leap HHI_L~    32    32     2.21     31 3.40e-2 6.80e-2
## 4 Medium    Drop~ HHI_L~ Oculus    32    32     3.67     31 9.14e-4 3.45e-3
## 5 Large     Drop~ B_Leap HHI_L~    32    32     0.502    31 6.19e-1 6.19e-1
## 6 Large     Drop~ HHI_L~ Oculus    32    32     4.86     31 3.21e-5 1.93e-4
## # ... with 5 more variables: p.adj.signif <chr>, effsize <dbl>,
## #   magnitude <ord>, Interface <chr>, y.position <dbl>
```

```r
  #summarise(mean=mean(Drop_Count), sd=sd(Drop_Count), se=(sd/sqrt(sample_size)), median=median(Drop_Co

  # plot
  drop_count_cubesize_plot <-
    ggplot(subject_data_cube_size, aes(Interface, Drop_Count, fill = Interface, color = Interface))+
    facet_grid(. ~ Cube_Size)+
    #geom_violinhalf(position = position_nudge(x = .05, y = 0), alpha=.7)+#)+#adjust=2)+
    #geom_crossbar(data=temp_plot_data, aes(Interface, median, ymin=q1, ymax=q3), width=.05, position=p
    geom_point(position = position_jitter(width = .1, height=.05), size = .25)+
    #geom_label(data=temp_plot_data, aes(Interface, median, label=paste0(ceiling(median),"%")), color="
    #geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(.05), co
    geom_bar(stat="identity", data=temp_plot_data, aes(x=Interface, y=mean, fill=Interface), alpha=.5, 
    #geom_point(data = temp_plot_data, aes(x = Interface, y = median), shape=10, size= 5, position = po
    geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(s
    geom_label(data=temp_plot_data, aes(Interface, y=mean, label=round(mean, 2)), color="white", positi
    #geom_dotplot(binaxis = "y", stackratio=1.4, binwidth = 1, stackdir="down", dotsize=.05, alpha=.8, 
    ylab('Accidental Drops (of 10)')+xlab(NULL)+theme_cowplot()+guides(fill = FALSE, colour = FALSE)+
    scale_x_discrete(labels=NULL)+
    scale_color_manual(values=mycolors)+#scale_colour_brewer(palette = "Set2")+
```

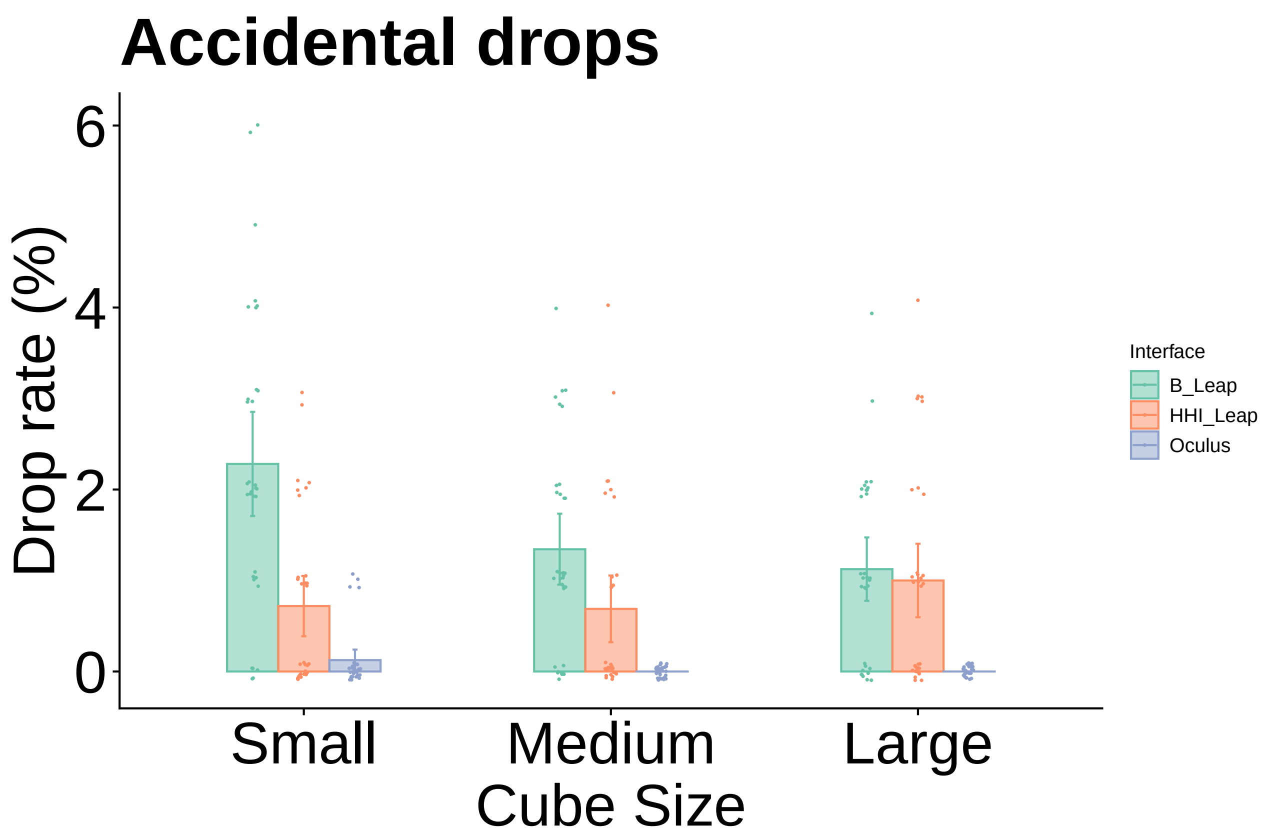
```
  scale_fill_manual(values=mycolors)+#scale_fill_brewer(palette = "Set2", direction=1)+
  stat_pvalue_manual(data=stat.test %>% filter(p.adj < 0.05), label = "p.adj.signif", position=positi
  labs(title="Accidental Drops by cube size", caption="Means, 95% CI; Within-subjects T-test, adj.: Ho
  theme(plot.title = element_text(size=title_size), axis.title = element_text(size=axis_text_size), a
#drop_count_cubesize_plot


drop_count_cubesize_raincloud <-
  ggplot(subject_data_cube_size, aes(Interface, Drop_Count, fill = Interface, color = Interface))+
  facet_grid(. ~ Cube_Size)+
  geom_violinhalf(position = position_nudge(x = .05, y = 0), alpha=myalpha, adjust = mysmoothing)+
  #geom_crossbar(data=temp_plot_data, aes(Interface, median, ymin=q1, ymax=q3), width=.05, position=p
  #geom_point(position = position_jitter(width = .1, height=.05), size = .25)+
  #geom_label(data=temp_plot_data, aes(Interface, median, label=paste0(ceiling(median),"%")), color="
  geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(.05), col
  #geom_bar(stat="identity", data=temp_plot_data, aes(x=Interface, y=mean, fill=Interface), alpha=.5,
  #geom_point(data = temp_plot_data, aes(x = Interface, y = median), shape=10, size= 5, position = po
  geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(s
  geom_label(data=temp_plot_data, aes(Interface, y=mean, label=round(mean, 2)), color="white", positi
  geom_dotplot(binaxis = "y", stackratio=1.4, binwidth = 1, stackdir="down", dotsize=.05, alpha=.8, p
  ylab('Accidental Drops (of 10)')+xlab(NULL)+theme_cowplot()+guides(fill = FALSE, colour = FALSE)+
  scale_x_discrete(labels=NULL)+
  scale_color_manual(values=mycolors)+#scale_colour_brewer(palette = "Set2")+
  scale_fill_manual(values=mycolors)+#scale_fill_brewer(palette = "Set2", direction=1)+
  stat_pvalue_manual(data=stat.test%>% filter(p.adj < 0.05), label = "p.adj.signif", position=position
  labs(title="Accidental Drops by Cube Size", caption="Means, 95% CI; Within-subjects T-test, adj.: Ho
  theme(plot.title = element_text(size=title_size), axis.title = element_text(size=axis_text_size), a
drop_count_cubesize_raincloud
```




```
  ggsave("accidental_drop_plot_cubesize.jpg", width=10, height=7)


  # box and whisker
ggplot(subject_data_cube_size, aes(Cube_Size, Drop_Count, color = Interface, fill=Interface))+
    #facet_grid(. ~ Cube_Size)+
    geom_boxplot(width=.5, alpha=.6, color="black", outlier.size=.25)+
    #geom_flat_violin(position = position_nudge(x = .25, y = 0), draw_quantiles=.5, alpha=.7)+#)+#adjus
    #geom_violin(draw_quantiles = .5)+#)+#adjust=2)+
    #geom_point(position = position_jitter(width = .15, height=.1), size = .25)+
    ylab('Accidental Drop Rate (%)')+xlab("Cube Size")+theme_cowplot()+guides(fill = FALSE, colour = FAI
    #geom_point(data=temp_plot_data, aes(y=median), color="black", size=2)+
    scale_colour_brewer(palette = "Set2")+
    scale_fill_brewer(palette = "Set2")+
    labs(title="Accidental Drop Rate", caption="Medians and whiskers to 1.5 * IQR")+
    theme(plot.title = element_text(size=title_size), axis.title = element_text(size=axis_text_size))
```




```
  #stat_compare_means(comparisons=list(c("B_Leap", "HHI_Leap"), c("HHI_Leap", "Oculus")), method="wil
 # stat_compare_means(comparisons=list(c("B_Leap", "HHI_Leap"), c("HHI_Leap", "Oculus")), method="t.t

  ggplot(temp_plot_data, aes(x=Cube_Size, y=mean, color=Interface, group=Interface), position=positio
  theme_cowplot() +
  theme(legend.position = "right", legend.title=element_text(size=legend_title_size),
        legend.text=element_text(size=legend_text_size),
        axis.text=element_text(size=axis_text_size),
        title = element_text(size=title_size, hjust=.5)) +
  #geom_pointrange(aes(ymin=mean-(se*1.96), ymax=mean+(se*1.96)), position=position_dodge2(.5))+
  geom_point(data=subject_data_cube_size, aes(Cube_Size, Drop_Count), size=.2, position=position_jitte
  geom_bar(stat="identity", aes(y=mean, fill=Interface), alpha=.5, position="dodge", width=.5)+
  geom_errorbar(aes(ymin=mean-(se*1.96), ymax=mean+(se*1.96)), width=.05, size=.5, position=position_
  scale_fill_brewer(palette="Set2")+
  scale_color_brewer(palette="Set2")+
  #geom_text(aes(y=mean, label=paste0(round(mean,2))), position=position_dodge2(7))+
  labs(title="Accidental drops",
       y="Drop rate (%)",
       x="Cube Size")
```




```
      #facet_grid(. ~ Cube_Size)+
  #   coord_cartesian(ylim=c(0.0035, 0.006))

# does data deviate from normal?
shapiro.test((subject_data_cube_size %>% filter(Interface=="HHI_Leap" & Cube_Size=="Medium"))$Drop_Coun
```

```
## [1] FALSE
```

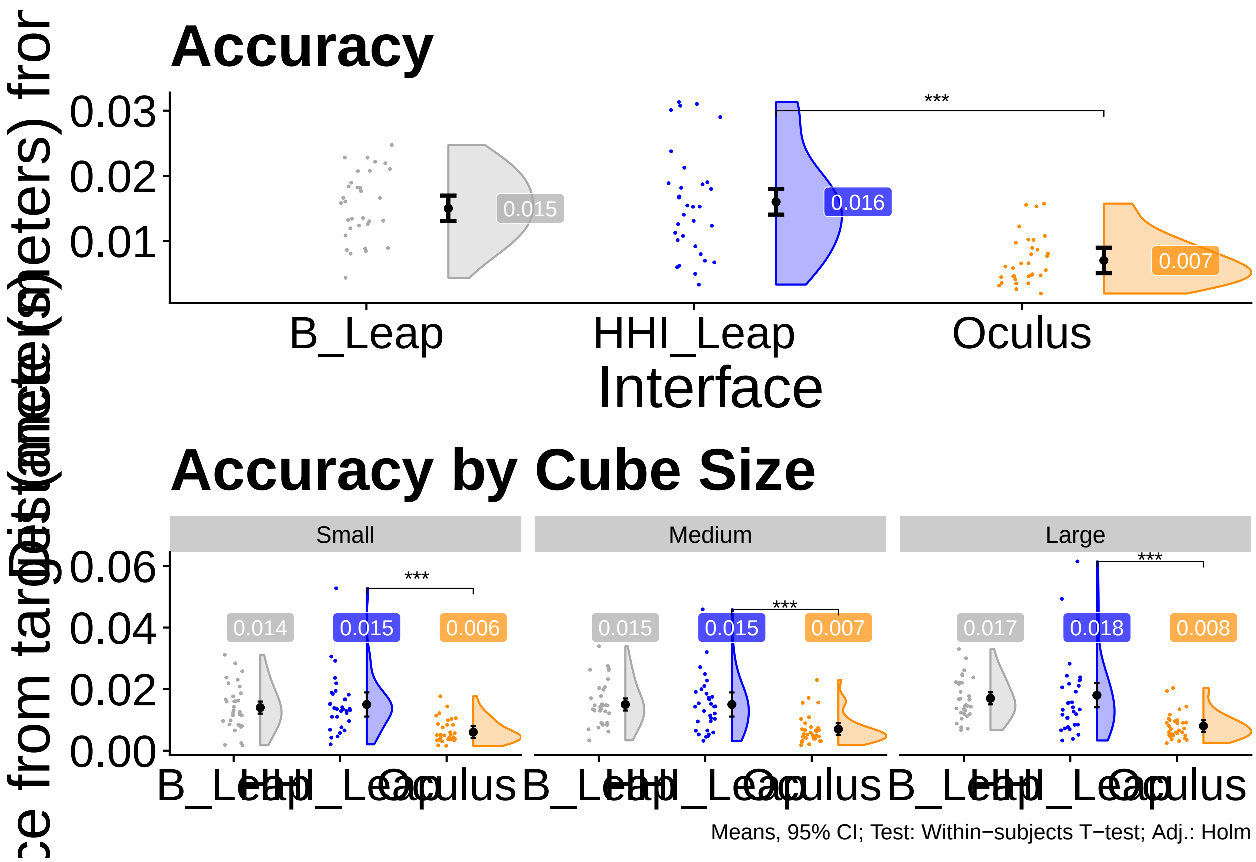
```
# save for output later
anova.Interface.cubesize.drops <- stat.test.anova
ttest.Interface.cubesize.drops <- stat.test
descriptives.Interface.cubesize.drops <-
  subject_data_cube_size %>% group_by(Interface, Cube_Size) %>% get_summary_stats(Drop_Count) %>%
  select(Interface, Cube_Size, variable, mean, sd, min, max, iqr)
descriptives.Interface.cubesize.drops
```

```
## # A tibble: 9 x 8
##   Interface Cube_Size variable    mean    sd   min   max   iqr
##   <fct>     <fct>     <chr>      <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 B_Leap    Small     Drop_Count 2.28  1.65      0     6  2
## 2 B_Leap    Medium    Drop_Count 1.34  1.12      0     4  1.25
## 3 B_Leap    Large     Drop_Count 1.12  1.01      0     4  2
## 4 HHI_Leap  Small     Drop_Count 0.719 0.958     0     3  1
## 5 HHI_Leap  Medium    Drop_Count 0.688 1.06      0     4  1
## 6 HHI_Leap  Large     Drop_Count 1     1.16      0     4  1.25
## 7 Oculus    Small     Drop_Count 0.125 0.336     0     1  0
```

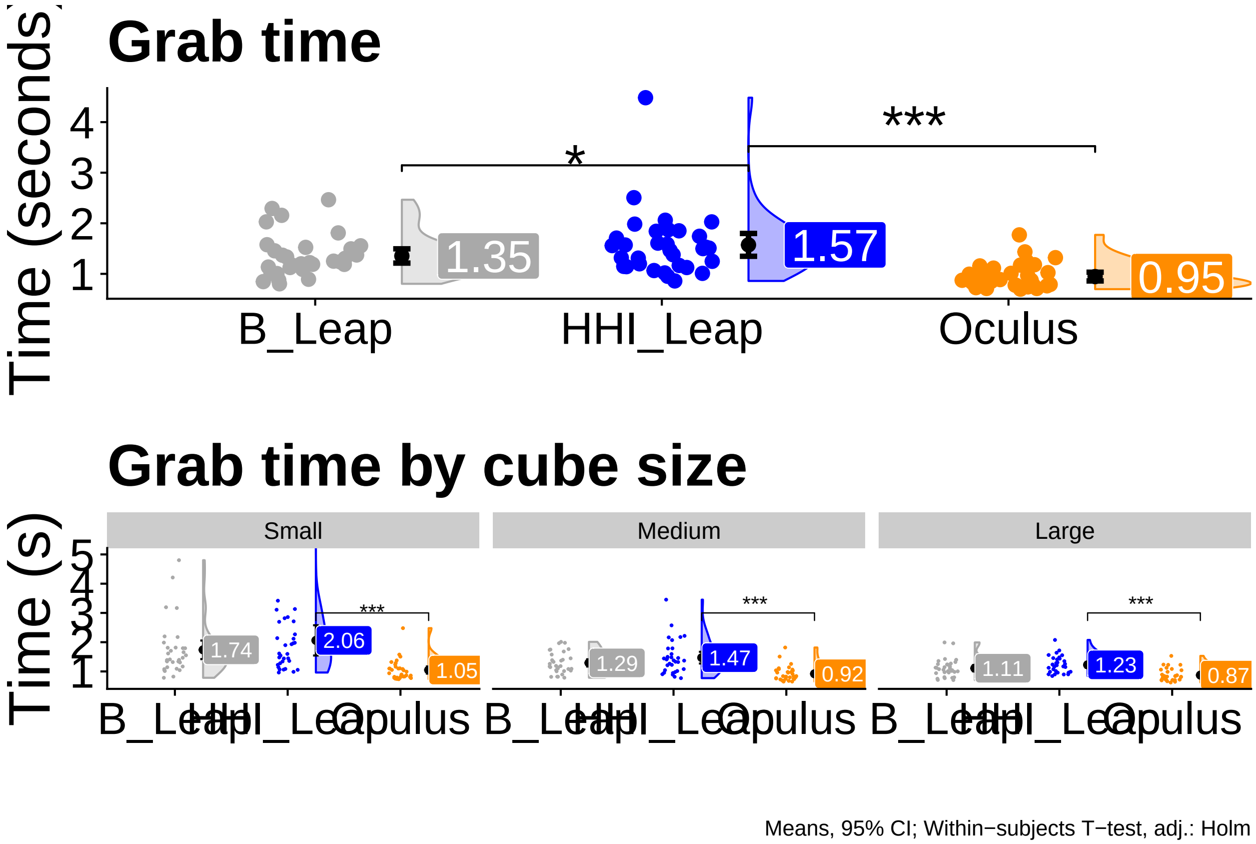
```
## 8 Oculus     Medium     Drop_Count 0      0          0      0  0
## 9 Oculus     Large      Drop_Count 0      0          0      0  0
```

Plot compilations - performance

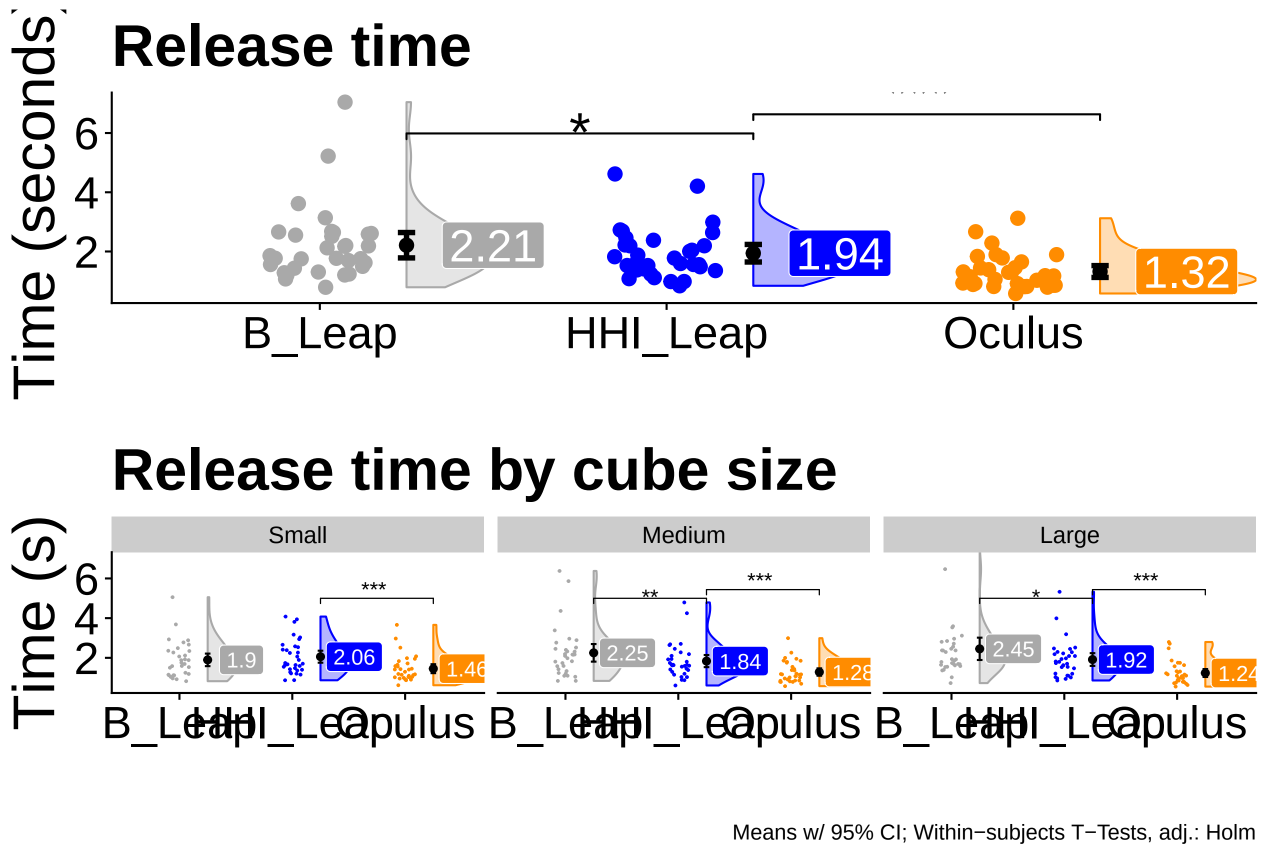
```r
# performance accuracy/distance
plot_grid(distance_Interface_raincloud, distance_cubesize_plot, nrow = 2)
```



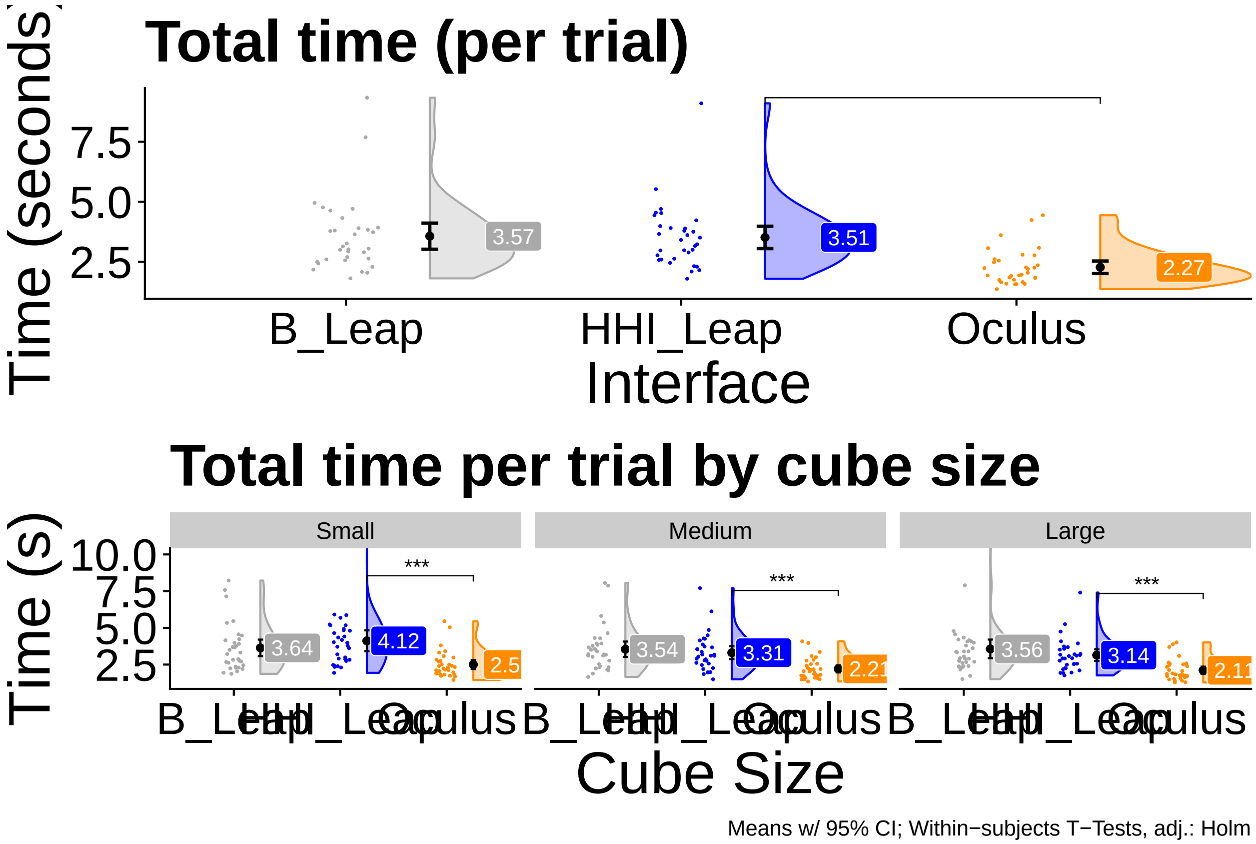

```r
ggsave(filename = "accuracy_plots.jpg", width = 10, height = 12)

# grab time
plot_grid(grabtime_Interface_raincloud, grabtime_cubesize_plot, nrow = 2)
```



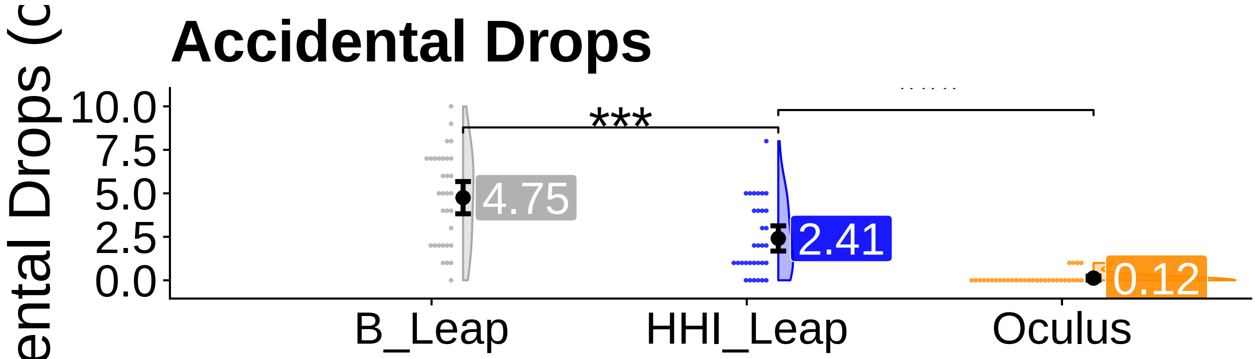

```r
ggsave(filename = "grabtime_plots.jpg", width = 10, height = 12)

# release time
plot_grid(releasetime_Interface_raincloud, releasetime_cubesize_plot, nrow = 2)
```

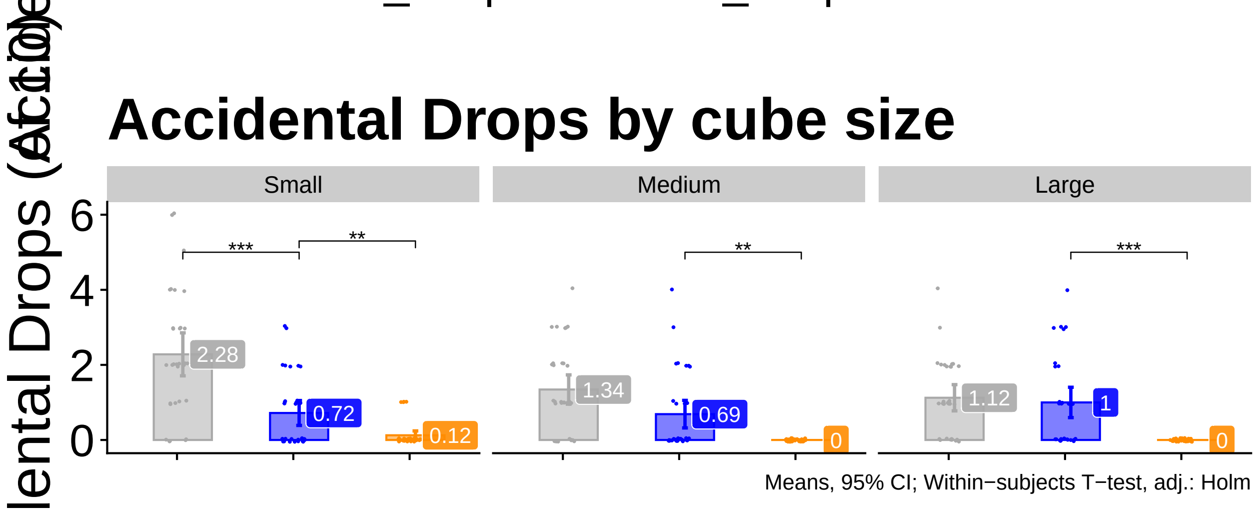



```
ggsave("releasetime_plots.jpg", width = 10, height = 12)

# total time
plot_grid(totaltime_Interface_raincloud, totaltime_cubesize_plot, nrow = 2)  #, totaltime_subjects_clea
```

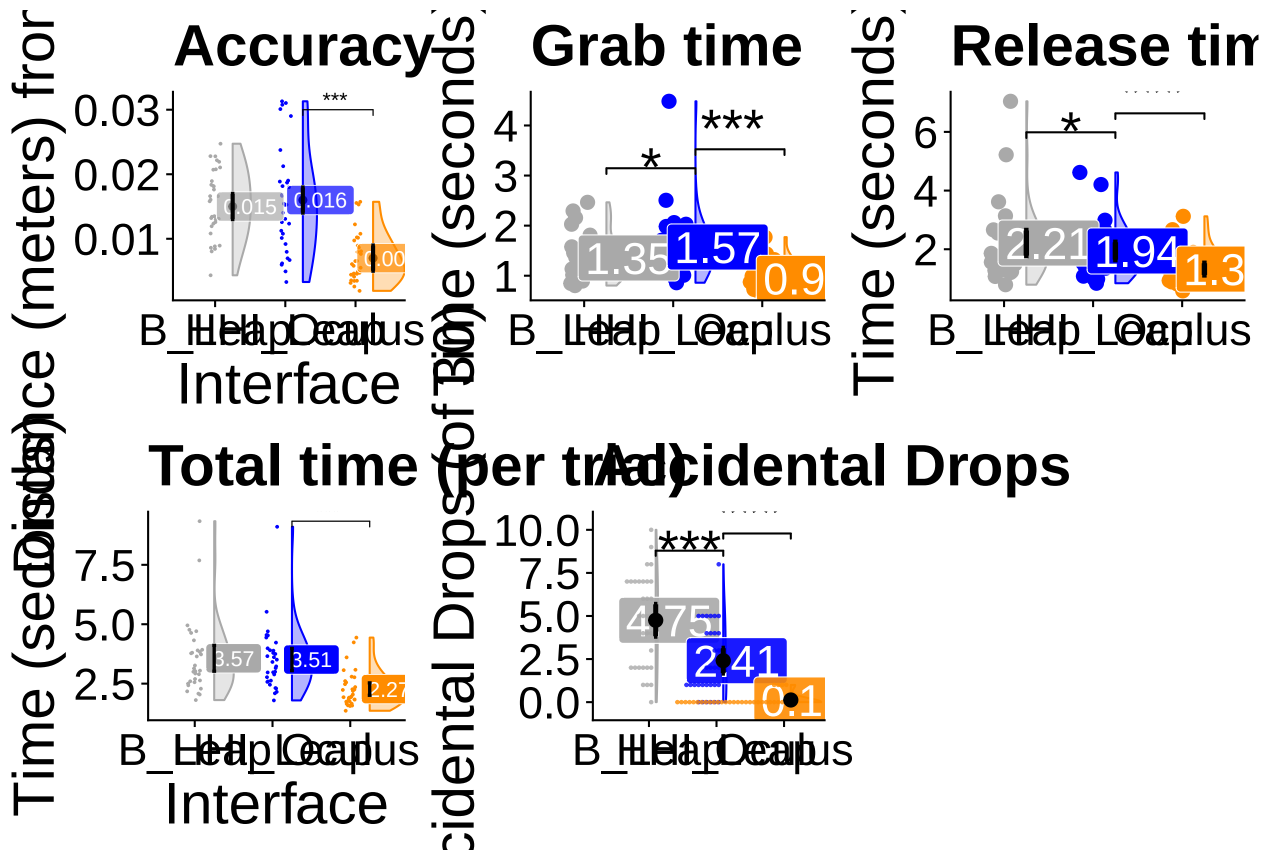



```r
ggsave("totaltime_plots.jpg", width = 10, height = 12)

plot_grid(drops_raincloud, drop_count_cubesize_plot, nrow = 2)  #, drops_subject)
```

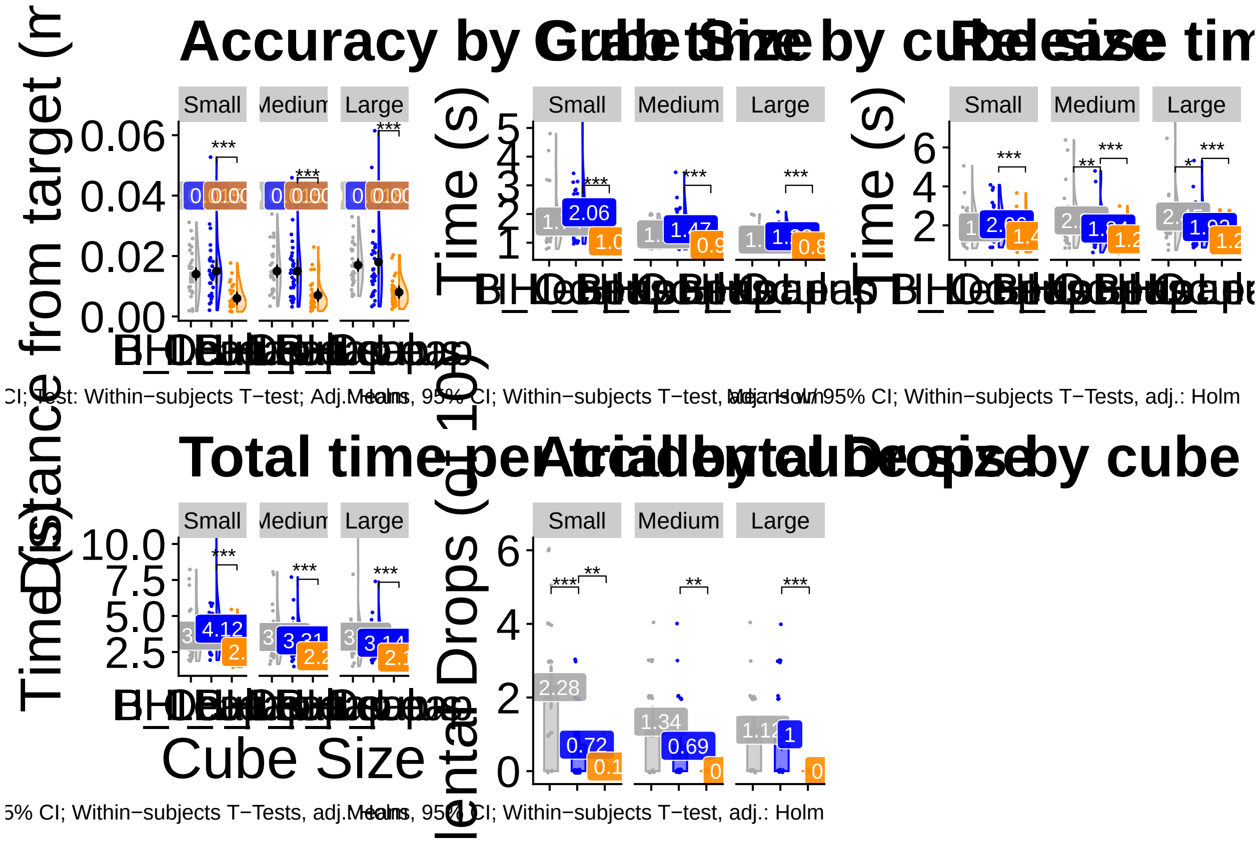





```r
ggsave("accidental_drop_plots.jpg", width = 10, height = 12)

# Interface level
plot_grid(distance_Interface_raincloud, grabtime_Interface_raincloud, releasetime_Interface_raincloud,
    totaltime_Interface_raincloud, drops_raincloud)
```




```r
ggsave("Interface_level_plots.jpg", width = 14, height = 8.5)


# cube size level
plot_grid(distance_cubesize_plot, grabtime_cubesize_plot, releasetime_cubesize_plot,
    totaltime_cubesize_plot, drop_count_cubesize_plot)
```




```
ggsave("cube_size_level_plots.jpg", width = 14, height = 8.5)
```

Subjective Metrics

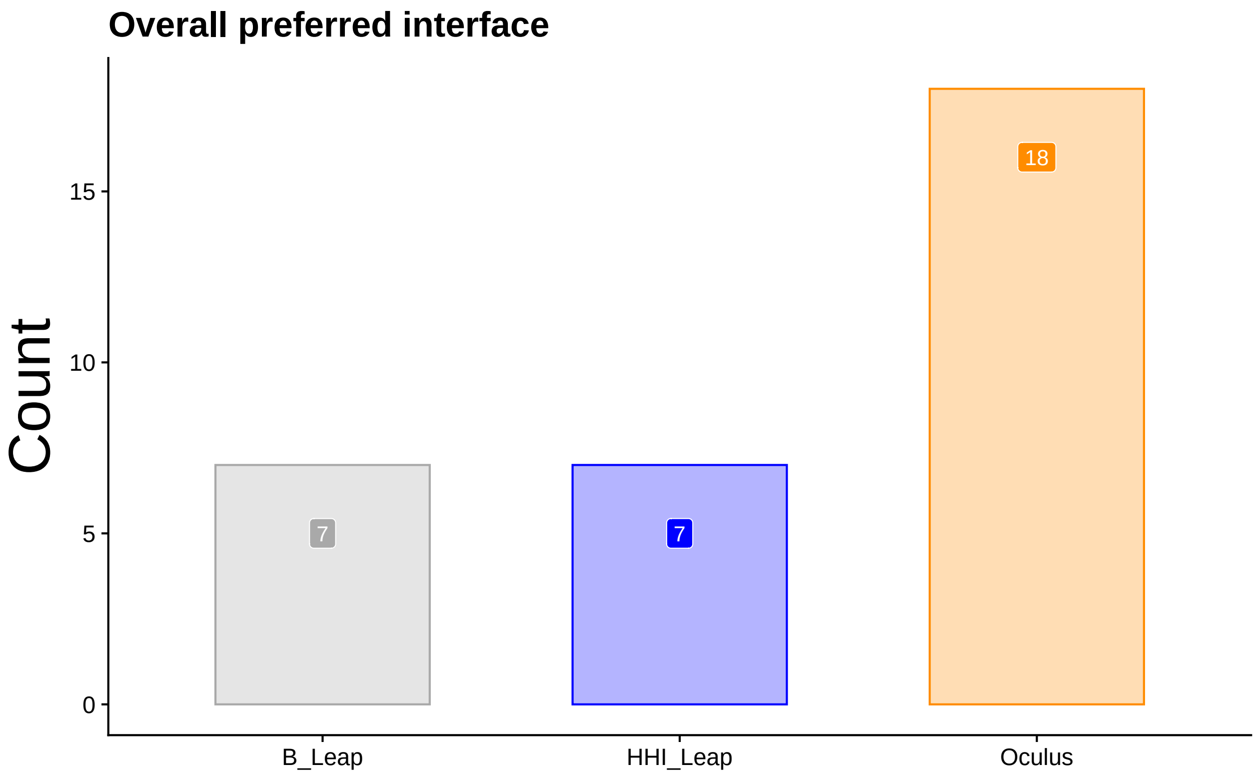
Overall preferred interface

```
stat.test <- table(subject_data_all_wide$PrefCondition)
stat.test
```

```
##
##    B_Leap   Oculus HHI_Leap
##         7       18        7
```

```
preference <- stat.test

# preferred condition
temp_plot_data <- subject_data_all_long %>% select(id, PrefCondition) %>% ungroup(.) %>%
    distinct(.) %>% count(PrefCondition) %>% mutate(PrefCondition = factor(PrefCondition,
    levels = c("B_Leap", "HHI_Leap", "Oculus")))
# make bar plot of counts
preferred_plot <- ggplot(temp_plot_data, aes(PrefCondition, n, fill = PrefCondition,
    color = PrefCondition)) + geom_bar(stat = "identity", alpha = myalpha, width = 0.6) +
    scale_fill_manual(values = mycolors) + scale_color_manual(values = mycolors) +
    geom_label(aes(label = n), position = position_nudge(y = -2), color = "white") +
```

```
    guides(fill = FALSE, color = FALSE) + labs(x = "", y = "Count") + theme_cowplot() +
    ggtitle("Overall preferred interface") + theme(plot.title = element_text(size = title_size *
    0.6), axis.title = element_text(size = axis_text_size))
preferred_plot
```




```
# ggsave('preferred_interface.jpg')
```

Likert data

Scores & Descriptives

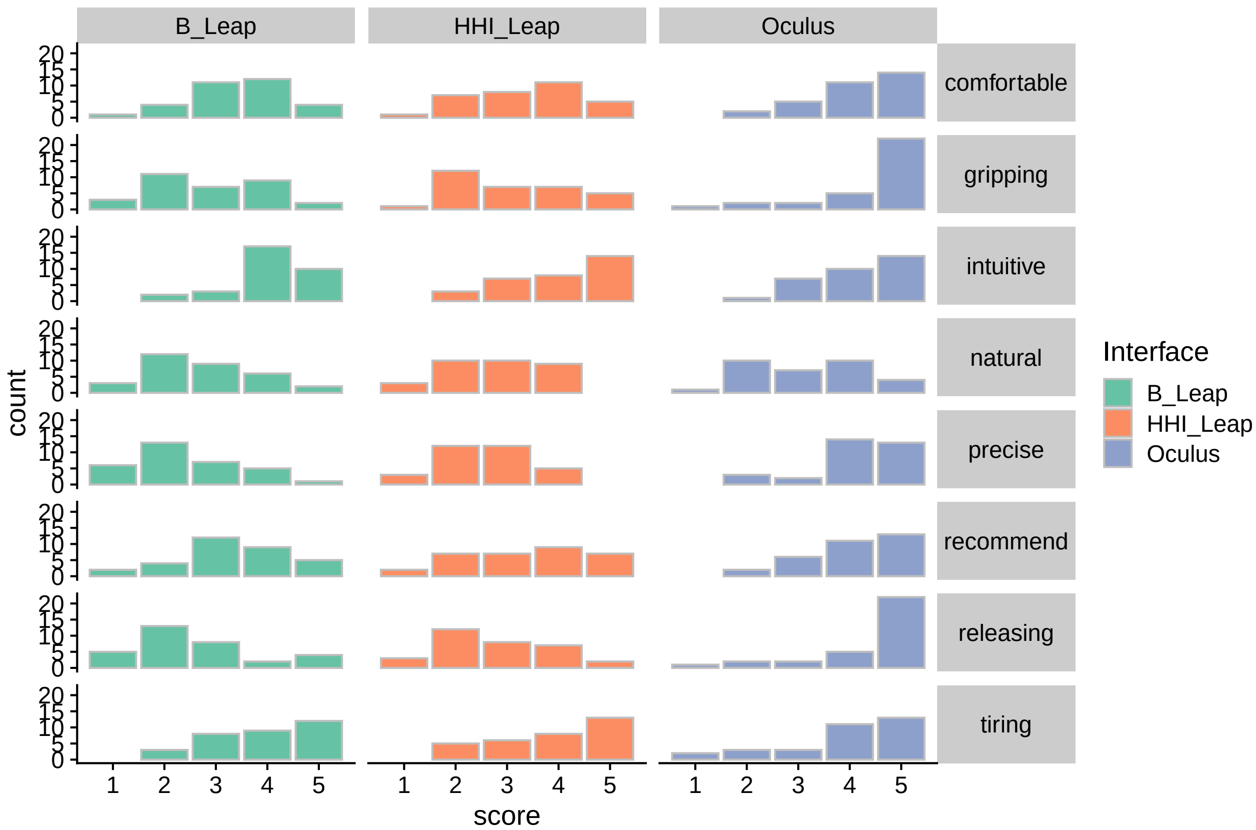
```
#collect scores
likert_scores <- subject_data_all_long %>%
  select(id, Interface, Q_1_Score:satisfaction, InterfaceOrder, Leap_Group, Oculus_Group) %>%
  rename(comfortable=Q_1_Score, precise=Q_2_Score, intuitive=Q_3_Score, tiring=Q_4_Score, gripping=Q_5_S

likert_5pt_grand_scores <- likert_scores %>% filter(question != "agency" & question != " satisfaction")
  group_by(id, Interface) %>% summarise(grand_mean=mean(score)) %>% ungroup(.)

likert_7pt_grand_scores <- likert_scores %>% filter(question == "agency" | question == " satisfaction")
  group_by(id, Interface) %>% summarise(grand_mean=mean(score)) %>% ungroup(.)

#descriptives
descriptives.subjective5pt <- likert_scores %>%
  filter(question != "agency" & question != "satisfaction") %>%
  group_by(question, Interface) %>%
  get_summary_stats(type="common") %>%
```

```
  select(interface=Interface, question, n, mean, sd, median, iqr, min, max) %>%
   bind_rows(likert_scores %>%
              group_by(question) %>%
              filter(question != "agency" & question != "satisfaction") %>%
              get_summary_stats(score, type="common") %>%
              mutate(interface="all") %>%
              select(interface, question, n, mean, sd, median, iqr, min, max)) %>%
  bind_rows(likert_scores %>%
              ungroup(.) %>%
              filter(question != "agency" & question != "satisfaction") %>%
              get_summary_stats(score, type="common") %>%
              mutate(question="all", interface="all") %>%
              select(interface, question, n, mean, sd, median, iqr, min, max)) %>%
  bind_rows(likert_scores %>%
              group_by(Interface) %>%
              filter(question != "agency" & question != "satisfaction") %>%
              get_summary_stats(score, type="common") %>%
              mutate(question="all", interface=Interface) %>%
              select(interface, question, n, mean, sd, median, iqr, min, max)) %>%
  bind_rows(likert_scores %>%
              filter(question != "agency" & question != "satisfaction") %>%
              group_by(question, Interface) %>%
              get_summary_stats(type="common") %>%
              get_summary_stats(mean, type="common") %>%
              mutate(question="means", interface="means") %>%
              select(interface, question, n, mean, sd, median, iqr, min, max)
            ) %>%
  mutate(SDs_from_mid = sd*(mean-3))
#descriptives.subjective5pt

descriptives.subjective7pt <- likert_scores %>%
  filter(question == "agency" | question == "satisfaction") %>%
  group_by(question, Interface) %>%
  get_summary_stats(type="common") %>%
  select(interface=Interface, question, n, mean, sd, median, iqr, min, max) %>%
      bind_rows(likert_scores %>%
              group_by(question) %>%
              filter(question == "agency" | question == "satisfaction") %>%
              get_summary_stats(score, type="common") %>%
              mutate(interface="all") %>%
              select(interface, question, n, mean, sd, median, iqr, min, max)) %>%
    bind_rows(likert_scores %>%
              ungroup(.) %>%
              filter(question == "agency" | question == "satisfaction") %>%
              get_summary_stats(score, type="common") %>%
              mutate(question="all", interface="all") %>%
              select(interface, question, n, mean, sd, median, iqr, min, max)) %>%
    bind_rows(likert_scores %>%
              group_by(Interface) %>%
              filter(question == "agency" | question == "satisfaction") %>%
              get_summary_stats(score, type="common") %>%
              mutate(question="all", interface=Interface) %>%
              select(interface, question, n, mean, sd, median, iqr, min, max)) %>%
```

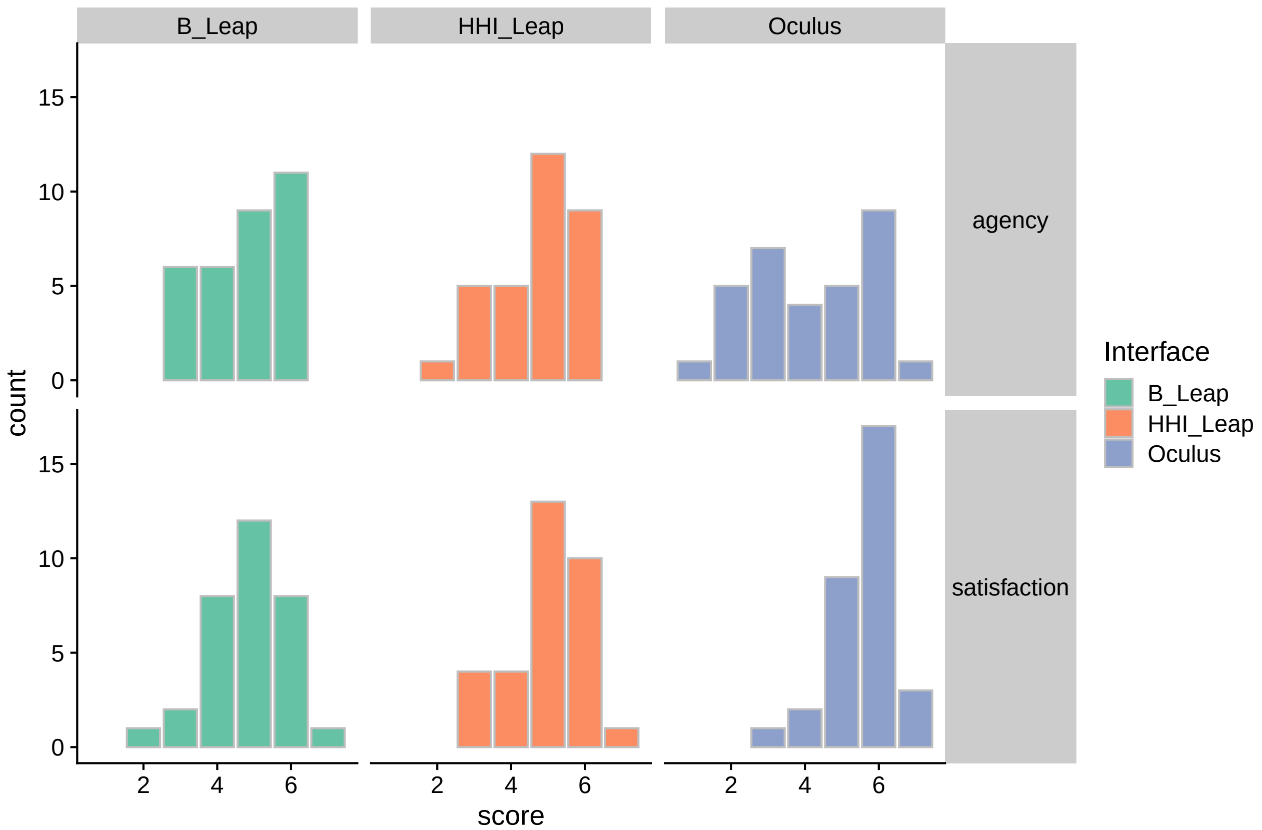
```r
    bind_rows(likert_scores %>%
              filter(question == "agency" | question == "satisfaction") %>%
              group_by(question, Interface) %>%
              get_summary_stats(type="common") %>%
              get_summary_stats(mean, type="common") %>%
              mutate(question="means", interface="means") %>%
              select(interface, question, n, mean, sd, median, iqr, min, max)) %>%
 mutate(SDs_from_mid = sd*(mean-4))
descriptives.subjective7pt
```

```
## # A tibble: 13 x 10
##    interface question          n  mean    sd median   iqr   min   max SDs_from_mid
##    <chr>     <chr>         <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl>        <dbl>
##  1 B_Leap    agency           32  4.78 1.13     5   2       3     6          0.881
##  2 HHI_Leap  agency           32  4.72 1.14     5   2       2     6          0.822
##  3 Oculus    agency           32  4.19 1.66     4   3       1     7          0.311
##  4 B_Leap    satisfacti~      32  4.84 1.08     5   2       2     7          0.912
##  5 HHI_Leap  satisfacti~      32  5    1.05     5   1.25    3     7          1.05
##  6 Oculus    satisfacti~      32  5.59 0.875    6   1       3     7          1.39
##  7 all       agency           96  4.56 1.34     5   3       1     7          0.755
##  8 all       satisfacti~      96  5.15 1.05     5   1       2     7          1.20
##  9 all       all             192  4.85 1.24     5   2       1     7          1.06
## 10 B_Leap    all              64  4.81 1.10     5   2       2     7          0.891
## 11 HHI_Leap  all              64  4.86 1.10     5   2       2     7          0.941
## 12 Oculus    all              64  4.89 1.49     5   2       1     7          1.33
## 13 means     means             6  4.85 0.455    4.81 0.227  4.19  5.59       0.389
```

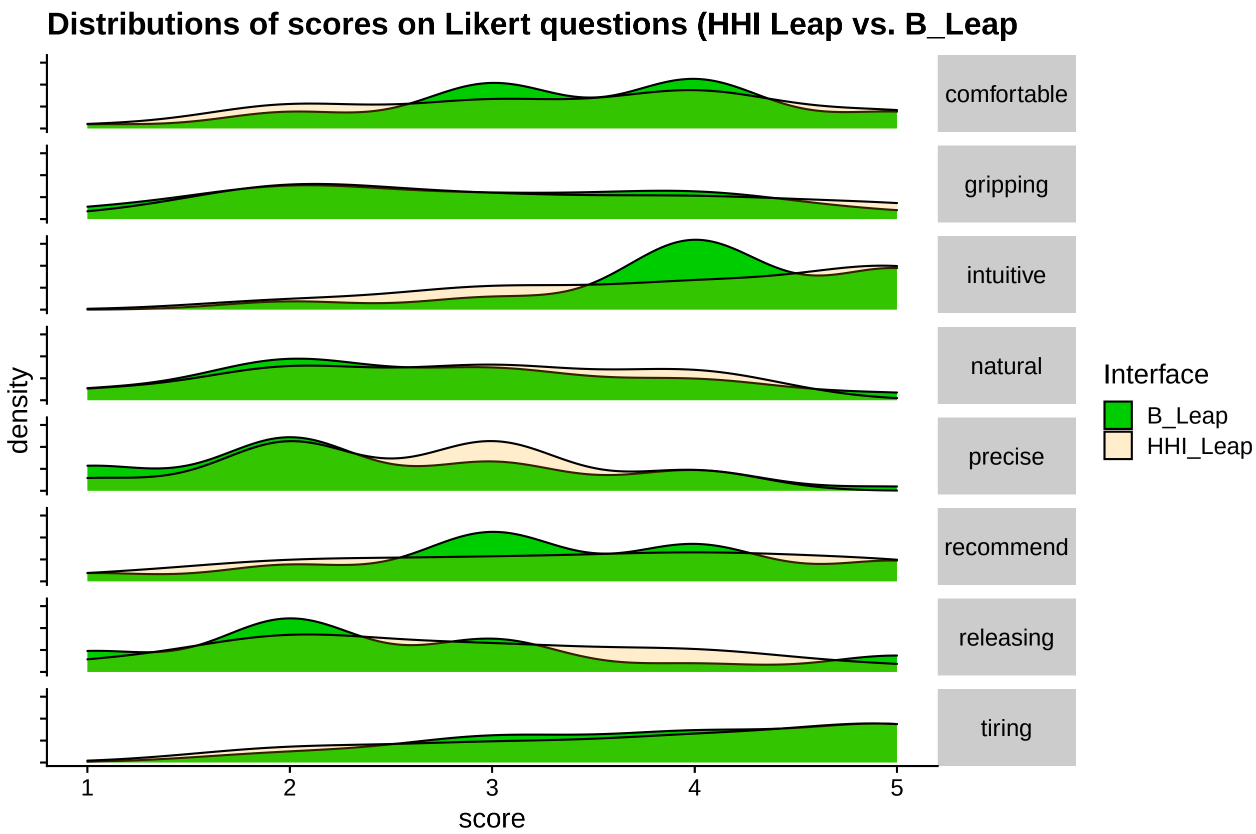
```r
# bar - 5 pt
# need to redo with counts if I want to add the means
ggplot(likert_scores %>% group_by(Interface, question) %>% filter(question!="agency" & question!="satis
  #geom_vline(aes(xintercept=mean(score)))+
  geom_bar(color="grey")+
  facet_grid(question ~ Interface)+
  theme(strip.text.y = element_text(angle = 360))+#geom_histogram(bins=32)+
  scale_fill_brewer(palette="Set2")+scale_color_manual(values = c("grey","black"))
```



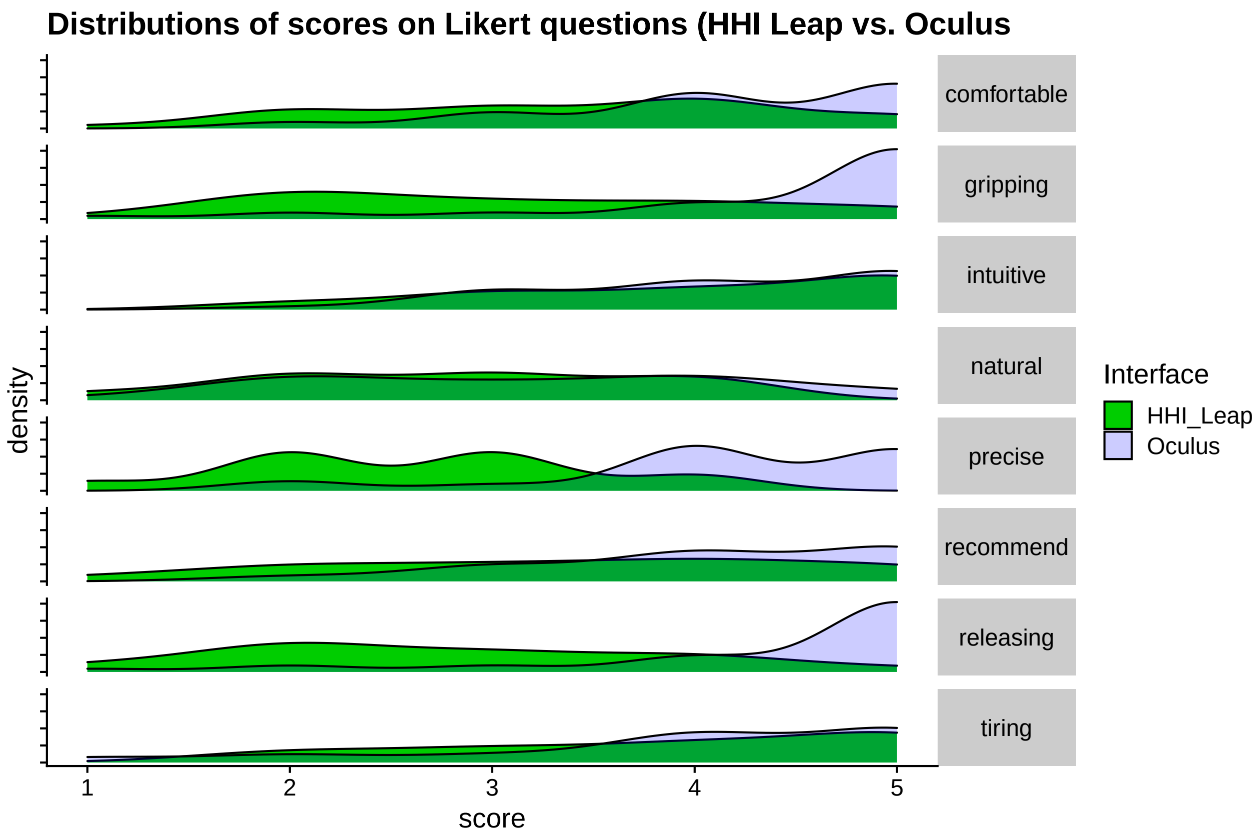

```
# bar - 7 pt
# need to redo with counts if I want to add the means
ggplot(likert_scores %>% group_by(Interface, question) %>% filter(question=="agency" | question=="satis
  #geom_vline(aes(xintercept=mean(score)))+
  geom_bar(color="grey")+
  facet_grid(question ~ Interface)+
  theme(strip.text.y = element_text(angle = 360))+#geom_histogram(bins=32)+
  scale_fill_brewer(palette="Set2")+scale_color_manual(values = c("grey","black"))
```



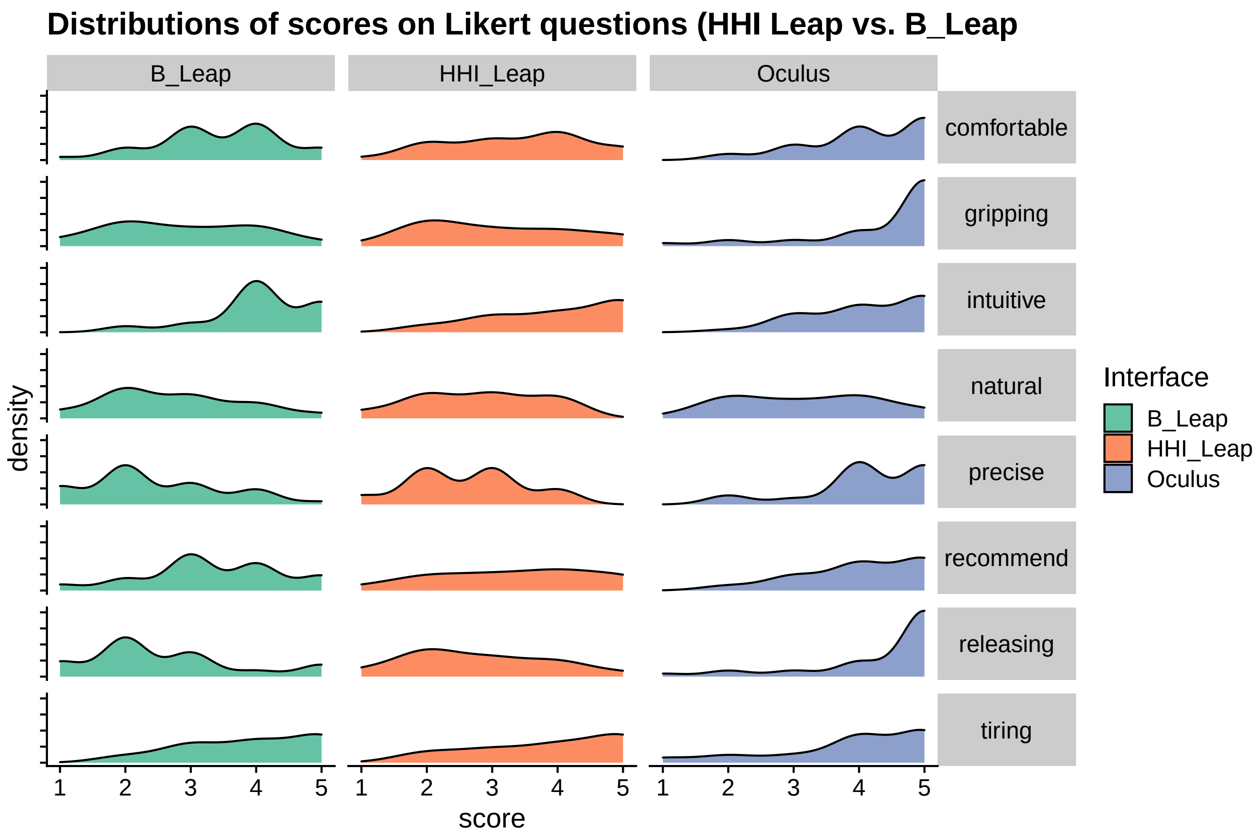

```
# plot distributions - 5 point
ggplot(likert_scores %>% filter(Interface != "Oculus", question != "agency" & question != "satisfaction
  geom_density()+
  facet_grid(question ~ .)+
  scale_alpha_manual(values=c(1,.2))+
  theme_cowplot()+ scale_fill_manual(values=c("green3", "orange"))+
  theme(strip.text.y = element_text(angle = 360))+#geom_histogram(bins=32)+
  labs(title="Distributions of scores on Likert questions (HHI Leap vs. B_Leap")+scale_y_continuous(lab
```



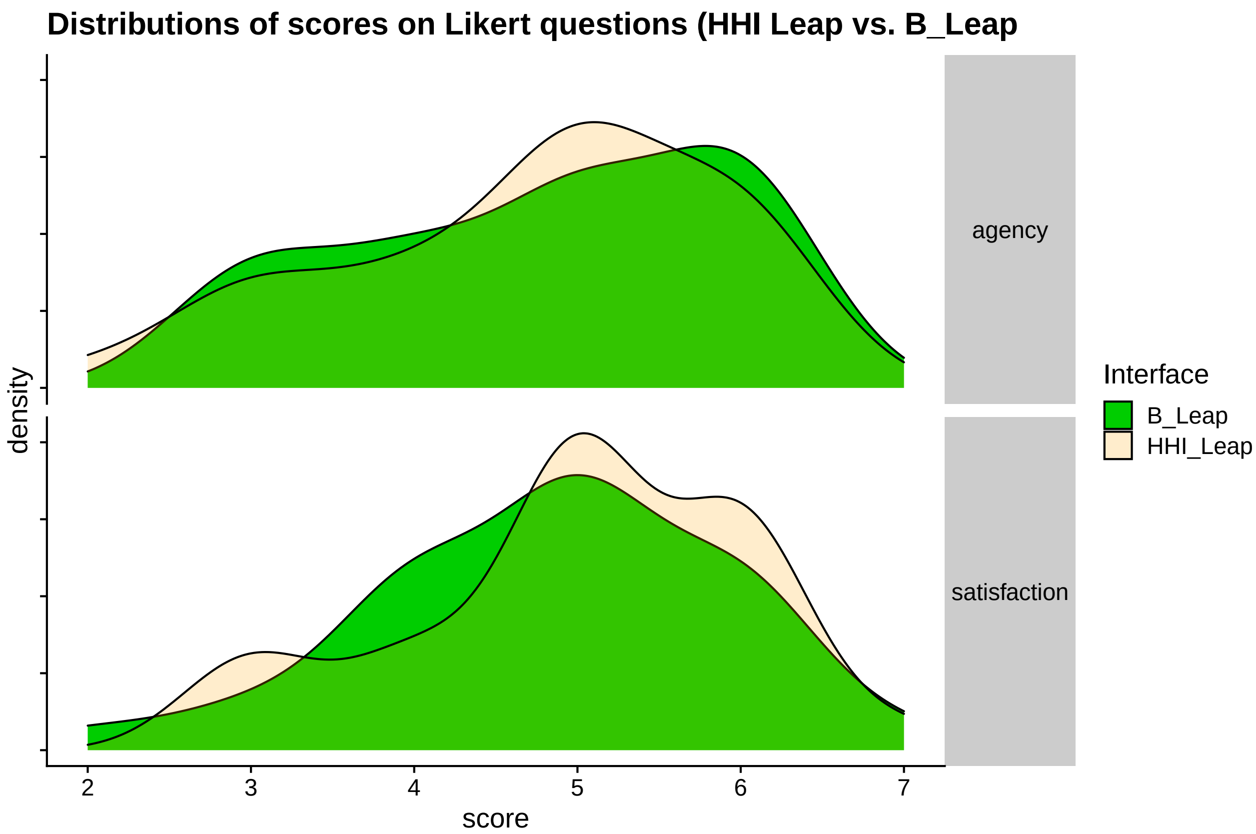

```r
ggplot(likert_scores %>% filter(Interface != "B_Leap", question != "agency" & question != "satisfaction
  geom_density()+
  facet_grid(question ~ .)+
  scale_alpha_manual(values=c(1,.2))+
  theme_cowplot()+ scale_fill_manual(values=c("green3","blue"))+
  theme(strip.text.y = element_text(angle = 360))+#geom_histogram(bins=32)+
  labs(title="Distributions of scores on Likert questions (HHI Leap vs. Oculus")+scale_y_continuous(labe
```



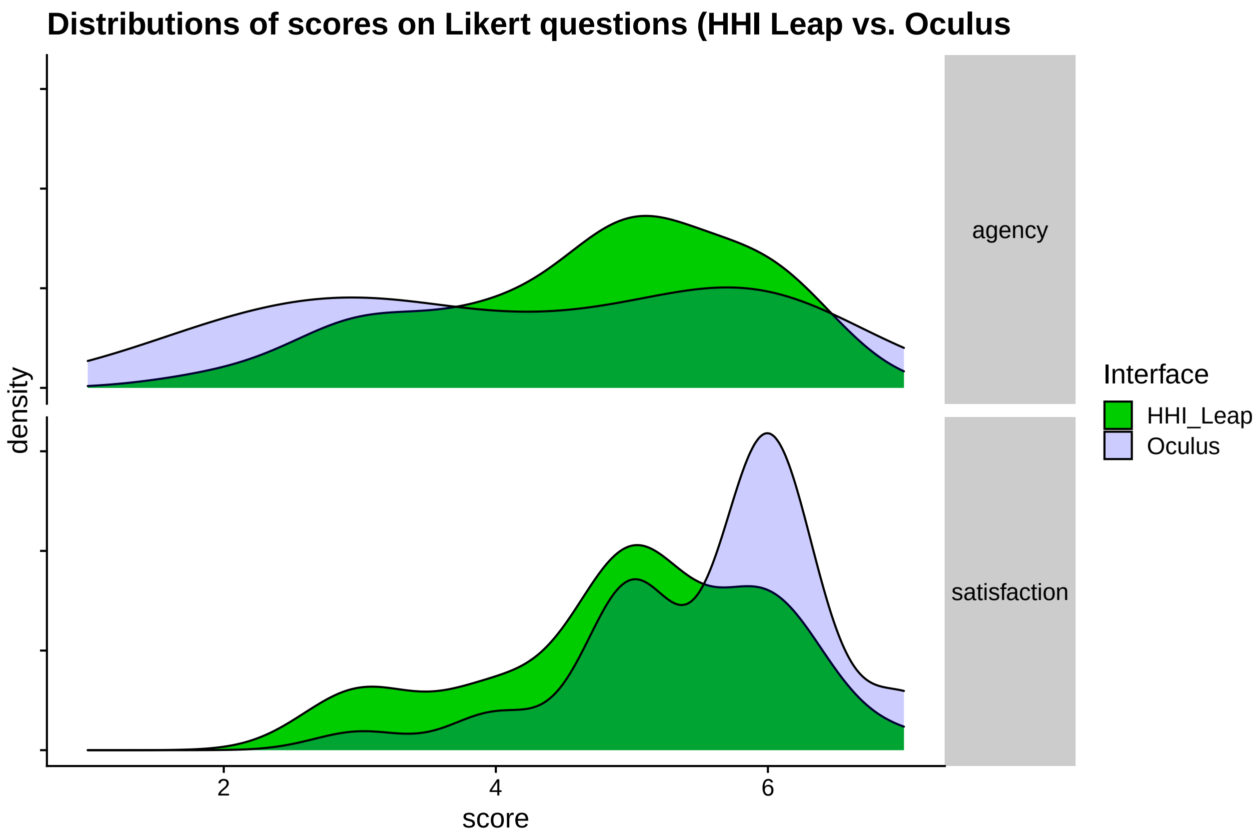

```
ggplot(likert_scores %>% filter(question != "agency" & question != "satisfaction"), aes(score, fill=Inte
  geom_density()+
  facet_grid(question ~ Interface)+ scale_fill_brewer(palette="Set2")+
  scale_alpha_manual(values=c(1,.2))+
  theme_cowplot()+
  theme(strip.text.y = element_text(angle = 360))+#geom_histogram(bins=32)+
  labs(title="Distributions of scores on Likert questions (HHI Leap vs. B_Leap")+scale_y_continuous(labe
```



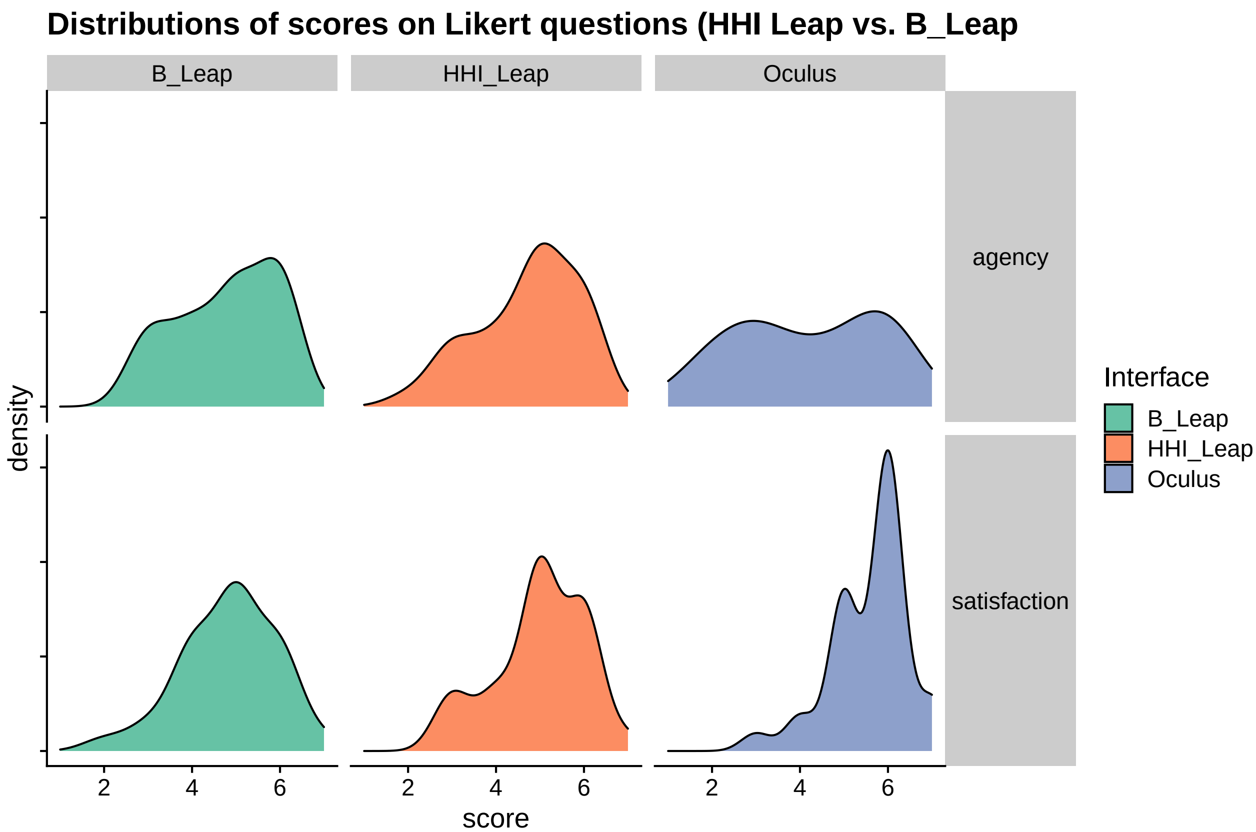

```
# plot distributions - 7 pt
ggplot(likert_scores %>% filter(Interface != "Oculus", question == "agency" | question == "satisfaction
  geom_density()+
  facet_grid(question ~ .)+
  scale_alpha_manual(values=c(1,.2))+
  theme_cowplot()+ scale_fill_manual(values=c("green3", "orange"))+
  theme(strip.text.y = element_text(angle = 360))+#geom_histogram(bins=32)+
  labs(title="Distributions of scores on Likert questions (HHI Leap vs. B_Leap")+scale_y_continuous(labe
```




```
ggplot(likert_scores %>% filter(Interface != "B_Leap", question == "agency" | question == "satisfaction
  geom_density()+
  facet_grid(question ~ .)+
  scale_alpha_manual(values=c(1,.2))+
  theme_cowplot()+ scale_fill_manual(values=c("green3","blue"))+
  theme(strip.text.y = element_text(angle = 360))+#geom_histogram(bins=32)+
  labs(title="Distributions of scores on Likert questions (HHI Leap vs. Oculus")+scale_y_continuous(labe
```




```
ggplot(likert_scores %>% filter(question == "agency" | question == "satisfaction"), aes(score, fill=Inte
  geom_density()+
  facet_grid(question ~ Interface)+ scale_fill_brewer(palette="Set2")+
  scale_alpha_manual(values=c(1,.2))+
  theme_cowplot()+
  theme(strip.text.y = element_text(angle = 360))+#geom_histogram(bins=32)+
  labs(title="Distributions of scores on Likert questions (HHI Leap vs. B_Leap")+scale_y_continuous(labe
```




```r
descriptives.subjective.all <-
  bind_rows(descriptives.subjective5pt, descriptives.subjective7pt) %>%
  filter(question != "all" & question != "means")
descriptives.subjective.all
```

```
## # A tibble: 40 x 10
##    interface question        n  mean    sd median   iqr   min   max SDs_from_mid
##    <chr>     <chr>       <dbl> <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl>        <dbl>
##  1 B_Leap    comfortable    32  3.44 0.982    3.5  1        1     5        0.430
##  2 HHI_Leap  comfortable    32  3.38 1.1      3.5  1.25     1     5        0.413
##  3 Oculus    comfortable    32  4.16 0.92     4    1        2     5        1.06
##  4 B_Leap    gripping       32  2.88 1.13     3    2        1     5       -0.141
##  5 HHI_Leap  gripping       32  3.09 1.17     3    2        1     5        0.110
##  6 Oculus    gripping       32  4.41 1.07     5    1        1     5        1.51
##  7 B_Leap    intuitive      32  4.09 0.818    4    1        2     5        0.895
##  8 HHI_Leap  intuitive      32  4.03 1.03     4    2        2     5        1.06
##  9 Oculus    intuitive      32  4.16 0.884    4    1.25     2     5        1.02
## 10 B_Leap    natural        32  2.75 1.08     3    1.25     1     5       -0.270
## # ... with 30 more rows
```

```r
# build df of mean and sd for t-test table
ttest_questions <- c("comfortable", "precise", "gripping", "releasing", "recommend", "satisfaction")

temp_df <- descriptives.subjective.all %>% filter(question %in% ttest_questions) %>%
  rename(Measure=question)

subjective_mean_sd <- data.frame(Measure=ttest_questions) %>%
```

```r
  left_join(temp_df %>% filter(interface=="HHI_Leap") %>% select(Measure, mean), by="Measure") %>%
  rename(HHI_Leap_Mean=mean) %>%
  left_join(temp_df %>% filter(interface=="HHI_Leap") %>% select(Measure, sd), by="Measure") %>%
  rename(HHI_Leap_SD=sd) %>%
  left_join(temp_df %>% filter(interface=="B_Leap") %>% select(Measure, mean), by="Measure") %>%
  rename(B_Leap_Mean=mean) %>%
  left_join(temp_df %>% filter(interface=="B_Leap") %>% select(Measure, sd), by="Measure") %>%
  rename(B_Leap_SD=sd) %>%
  left_join(temp_df %>% filter(interface=="Oculus") %>% select(Measure, mean), by="Measure") %>%
  rename(Oculus_Mean=mean) %>%
  left_join(temp_df %>% filter(interface=="Oculus") %>% select(Measure, sd), by="Measure") %>%
  rename(Oculus_SD=sd)

descriptives.subjective.all %>% filter(interface=="HHI_Leap") %>% select(interface, question, mean, sd,
```

```
## # A tibble: 10 x 5
##    interface question      mean    sd SDs_from_mid
##    <chr>     <chr>        <dbl> <dbl>        <dbl>
##  1 HHI_Leap  comfortable   3.38 1.1          0.413
##  2 HHI_Leap  gripping      3.09 1.17         0.110
##  3 HHI_Leap  intuitive     4.03 1.03         1.06
##  4 HHI_Leap  natural       2.78 0.975       -0.214
##  5 HHI_Leap  precise       2.59 0.875       -0.355
##  6 HHI_Leap  recommend     3.38 1.24         0.464
##  7 HHI_Leap  releasing     2.78 1.10        -0.241
##  8 HHI_Leap  tiring        3.91 1.12         1.01
##  9 HHI_Leap  agency        4.72 1.14         0.822
## 10 HHI_Leap  satisfaction  5    1.05         1.05
```

Stat tests

```r
# t tests likert q's: hhi_leap vs. oculus
ttests.likert.oculus <- likert_scores %>% ungroup(.) %>% group_by(question) %>% filter(question %in%
    c("comfortable", "precise", "gripping", "releasing", "recommend", "satisfaction")) %>%
    t_test(score ~ Interface, paired = TRUE, comparisons = list(c("HHI_Leap", "Oculus"))) %>%
    adjust_pvalue() %>% add_significance(p.col = "p.adj", output.col = "p.adj.signif",
    symbols = mysymbols)

ttests.likert.leap <- likert_scores %>% ungroup(.) %>% group_by(question) %>% t_test(score ~
    Interface, paired = TRUE, comparisons = list(c("HHI_Leap", "B_Leap"))) %>% adjust_pvalue() %>%
    add_significance(p.col = "p.adj", output.col = "p.adj.signif", symbols = mysymbols) %>%
    mutate(group1 = "HHI_Leap", group2 = "B_Leap", statistic = -1 * statistic)

# t.tests.likert.all.vs.hhi <- bind_rows(ttests.likert.oculus,
# ttests.likert.leap)

# t tests all vs. hhi - only for measures flagged as significant by ANOVA
ttests_subjective_all <- likert_scores %>% ungroup(.) %>% filter(question %in% c("comfortable",
    "precise", "gripping", "releasing", "recommend", "satisfaction")) %>% group_by(question) %>%
    pairwise_t_test(score ~ Interface, paired = TRUE, ref.group = "HHI_Leap") %>%
    add_significance(p.col = "p.adj", output.col = "p.adj.signif", symbols = mysymbols)
```

```r
# sink('subjective_stats.csv') write.csv(t.tests.likert.all.vs.hhi) sink()

t.tests.likert.all.vs.hhi <- bind_rows(ttests.likert.oculus, ttests.likert.leap) %>%
    left_join(likert_scores %>% ungroup(.) %>% cohens_d(score ~ Interface, paired = TRUE,
        ref.group = "HHI_Leap") %>% select(group1, group2, effsize, magnitude), by = c("group1",
        "group2")) %>% mutate(Interface = group1, test = "t-test")
t.tests.likert.all.vs.hhi
```

```
## # A tibble: 16 x 15
##    question .y.   group1 group2    n1    n2 statistic    df        p   p.adj
##    <chr>    <chr> <chr>  <chr>  <int> <int>     <dbl> <dbl>    <dbl>   <dbl>
##  1 comfort~ score HHI_L~ Oculus    32    32    -3.09     31 4.00e-3 1.20e-2
##  2 gripping score HHI_L~ Oculus    32    32    -5.21     31 1.16e-5 4.64e-5
##  3 precise  score HHI_L~ Oculus    32    32    -6.47     31 3.26e-7 1.96e-6
##  4 recomme~ score HHI_L~ Oculus    32    32    -2.66     31 1.20e-2 2.40e-2
##  5 releasi~ score HHI_L~ Oculus    32    32    -6.14     31 8.26e-7 4.13e-6
##  6 satisfa~ score HHI_L~ Oculus    32    32    -2.46     31 2.00e-2 2.40e-2
##  7 agency   score HHI_L~ B_Leap    32    32    -0.263    31 7.94e-1 1.00e+0
##  8 comfort~ score HHI_L~ B_Leap    32    32    -0.312    31 7.57e-1 1.00e+0
##  9 gripping score HHI_L~ B_Leap    32    32     0.909    31 3.70e-1 1.00e+0
## 10 intuiti~ score HHI_L~ B_Leap    32    32    -0.349    31 7.30e-1 1.00e+0
## 11 natural  score HHI_L~ B_Leap    32    32     0.133    31 8.95e-1 1.00e+0
## 12 precise  score HHI_L~ B_Leap    32    32     0.596    31 5.56e-1 1.00e+0
## 13 recomme~ score HHI_L~ B_Leap    32    32     0.166    31 8.69e-1 1.00e+0
## 14 releasi~ score HHI_L~ B_Leap    32    32     0.641    31 5.26e-1 1.00e+0
## 15 satisfa~ score HHI_L~ B_Leap    32    32     0.776    31 4.44e-1 1.00e+0
## 16 tiring   score HHI_L~ B_Leap    32    32    -0.183    31 8.56e-1 1.00e+0
## # ... with 5 more variables: p.adj.signif <chr>, effsize <dbl>,
## #   magnitude <ord>, Interface <chr>, test <chr>
```

```r
# anova for 5 pt. q's w/ grand means
anova.Interface.5ptgrand <- anova_summary(effect.size = "pes", aov(grand_mean ~ Interface +
    Error(id/Interface), data = likert_5pt_grand_scores))

# t-tests for 5pt q's w/ grand means
ttest.Interface.5ptgrand <- likert_5pt_grand_scores %>% ungroup(.) %>% pairwise_t_test(grand_mean ~
    Interface, paired = TRUE) %>% adjust_pvalue() %>% left_join(likert_5pt_grand_scores %>%
    cohens_d(grand_mean ~ Interface, paired = TRUE))

# anova for 7 pt. q's w/ grand means
anova.Interface.7ptgrand <- anova_summary(effect.size = "pes", aov(grand_mean ~ Interface +
    Error(id/Interface), data = likert_7pt_grand_scores))

# t-tests for 7 pt q's w/ grand means
ttest.Interface.7ptgrand <- likert_7pt_grand_scores %>% ungroup(.) %>% pairwise_t_test(grand_mean ~
    Interface, paired = TRUE) %>% adjust_pvalue() %>% left_join(likert_7pt_grand_scores %>%
    cohens_d(grand_mean ~ Interface, paired = TRUE))


# wilcox vs. B_Leap
wilcox.tests.likert.vs.B_Leap <- likert_scores %>% ungroup(.) %>% group_by(question) %>%
    pairwise_wilcox_test(score ~ Interface, paired = TRUE, comparisons = list(c("HHI_Leap",
        "B_Leap"))) %>% adjust_pvalue() %>% add_significance(p.col = "p.adj", output.col = "p.adj.signi
```

```r
    mutate(Interface = group1, test = "wilcox test")
wilcox.tests.likert.vs.B_Leap
```

```
## # A tibble: 10 x 12
##    question .y.   group1 group2    n1    n2 statistic     p p.adj p.adj.signif
##    <chr>    <chr> <chr>  <chr>  <int> <int>     <dbl> <dbl> <dbl> <chr>
##  1 agency   score B_Leap HHI_L~    32    32       139 0.987     1 ns
##  2 comfort~ score B_Leap HHI_L~    32    32       106 0.661     1 ns
##  3 gripping score B_Leap HHI_L~    32    32       150 0.512     1 ns
##  4 intuiti~ score B_Leap HHI_L~    32    32       115 0.705     1 ns
##  5 natural  score B_Leap HHI_L~    32    32        98 0.803     1 ns
##  6 precise  score B_Leap HHI_L~    32    32       154. 0.574     1 ns
##  7 recomme~ score B_Leap HHI_L~    32    32       102 0.922     1 ns
##  8 releasi~ score B_Leap HHI_L~    32    32       124. 0.467     1 ns
##  9 satisfa~ score B_Leap HHI_L~    32    32       128 0.509     1 ns
## 10 tiring   score B_Leap HHI_L~    32    32        40 0.968     1 ns
## # ... with 2 more variables: Interface <chr>, test <chr>
```

```r
# wilcox vs. oculus
wilcox.tests.likert.vs.oculus <- likert_scores %>% ungroup(.) %>% group_by(question) %>%
    pairwise_wilcox_test(score ~ Interface, paired = TRUE, comparisons = list(c("HHI_Leap",
        "Oculus"))) %>% mutate(Interface = group1, test = "wilcox test")
wilcox.tests.likert.vs.oculus
```

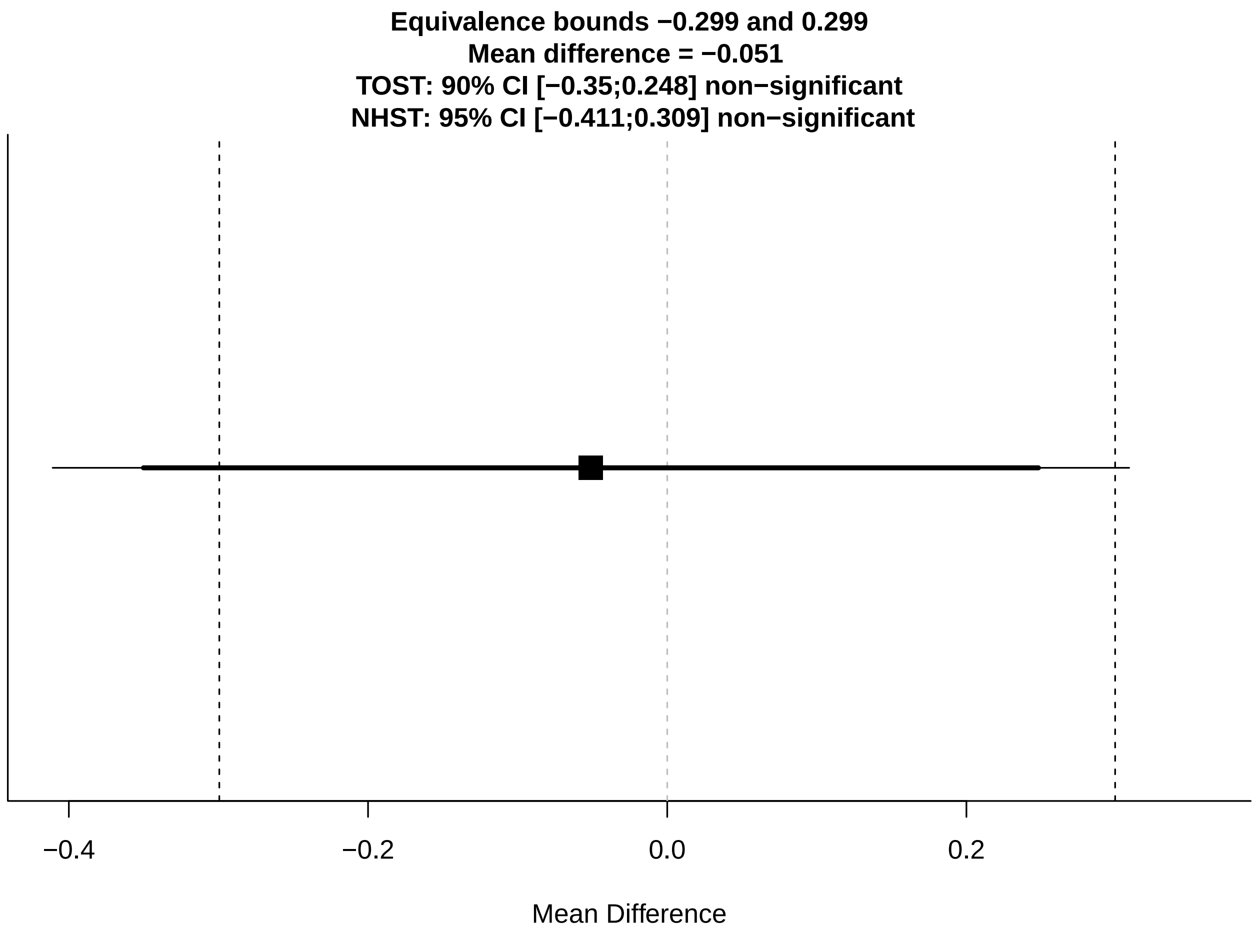
```
## # A tibble: 10 x 12
##    question .y.   group1 group2    n1    n2 statistic       p   p.adj
##    <chr>    <chr> <chr>  <chr>  <int> <int>     <dbl>   <dbl>   <dbl>
##  1 agency   score HHI_L~ Oculus    32    32      223   2.27e-1 2.27e-1
##  2 comfort~ score HHI_L~ Oculus    32    32       48.5 6.00e-3 6.00e-3
##  3 gripping score HHI_L~ Oculus    32    32       24   1.01e-4 1.01e-4
##  4 intuiti~ score HHI_L~ Oculus    32    32       92   6.31e-1 6.31e-1
##  5 natural  score HHI_L~ Oculus    32    32      126.  2.08e-1 2.08e-1
##  6 precise  score HHI_L~ Oculus    32    32       16.5 1.81e-5 1.81e-5
##  7 recomme~ score HHI_L~ Oculus    32    32       31   1.70e-2 1.70e-2
##  8 releasi~ score HHI_L~ Oculus    32    32       27.5 3.44e-5 3.44e-5
##  9 satisfa~ score HHI_L~ Oculus    32    32       20   2.30e-2 2.30e-2
## 10 tiring   score HHI_L~ Oculus    32    32       92.5 9.35e-1 9.35e-1
## # ... with 3 more variables: p.adj.signif <chr>, Interface <chr>, test <chr>
```

```r
# transform for data output
stat.test <- t.tests.likert.all.vs.hhi
stat.test <- stat.test %>% mutate(p = round(p, 4), p.adj = round(p.adj, 4), statistic = round(statistic,
    3), effsize = round(effsize, 4)) %>% select(question, group1, group2, stat = statistic,
    df, p, p.adj, p.a.sig = p.adj.signif, eff = effsize, mag = magnitude)

# anovas
anova.test <- anova_summary(effect.size = "pes", aov(score ~ Interface * question +
    Error(id/(Interface * question)), data = likert_scores %>% filter(question !=
    "agency", question != "satisfaction")))

anova.likert <- anova_summary(effect.size = "pes", aov(score ~ Interface + Error(id/Interface),
    data = likert_scores %>% filter(question == "comfortable"))) %>% mutate(question = "comfortable") %>%
```

```
    rbind(anova_summary(effect.size = "pes", aov(score ~ Interface + Error(id/Interface),
        data = likert_scores %>% filter(question == "precise"))) %>% mutate(question = "precise")) %>%
    rbind(anova_summary(effect.size = "pes", aov(score ~ Interface + Error(id/Interface),
        data = likert_scores %>% filter(question == "intuitive"))) %>% mutate(question = "intuitive")) %
    rbind(anova_summary(effect.size = "pes", aov(score ~ Interface + Error(id/Interface),
        data = likert_scores %>% filter(question == "tiring"))) %>% mutate(question = "tiring")) %>%
    rbind(anova_summary(effect.size = "pes", aov(score ~ Interface + Error(id/Interface),
        data = likert_scores %>% filter(question == "gripping"))) %>% mutate(question = "gripping")) %>
    rbind(anova_summary(effect.size = "pes", aov(score ~ Interface + Error(id/Interface),
        data = likert_scores %>% filter(question == "releasing"))) %>% mutate(question = "releasing")) %
    rbind(anova_summary(effect.size = "pes", aov(score ~ Interface + Error(id/Interface),
        data = likert_scores %>% filter(question == "natural"))) %>% mutate(question = "natural")) %>%
    rbind(anova_summary(effect.size = "pes", aov(score ~ Interface + Error(id/Interface),
        data = likert_scores %>% filter(question == "recommend"))) %>% mutate(question = "recommend")) %
    rbind(anova_summary(effect.size = "pes", aov(score ~ Interface + Error(id/Interface),
        data = likert_scores %>% filter(question == "agency"))) %>% mutate(question = "agency")) %>%
    rbind(anova_summary(effect.size = "pes", aov(score ~ Interface + Error(id/Interface),
        data = likert_scores %>% filter(question == "satisfaction"))) %>% mutate(question = "satisfacti
    select(question, everything(.))
```

Equivalence tests

```
#
r_val <- cor.test(x = subject_data_all_long[which(subject_data_all_long$Interface ==
    "HHI_Leap"), "totaltime"], y = subject_data_all_long[which(subject_data_all_long$Interface ==
    "B_Leap"), "totaltime"], method = "pearson", alternative = "two.sided")
r_val <- r_val$estimate[[1]]

TOSTpaired(n = 32, m1 = descriptives.Interface.totaltime[which(descriptives.Interface.totaltime$Interfac
    "HHI_Leap"), "mean"][[1]], m2 = descriptives.Interface.totaltime[which(descriptives.Interface.totalt
    "B_Leap"), "mean"][[1]], sd1 = descriptives.Interface.totaltime[which(descriptives.Interface.totalti
    "HHI_Leap"), "sd"][[1]], sd2 = descriptives.Interface.totaltime[which(descriptives.Interface.totalti
    "B_Leap"), "sd"][[1]], r12 = r_val, low_eqbound_dz = -0.3, high_eqbound_dz = 0.3,
    alpha = 0.05, plot = TRUE, verbose = TRUE)
```



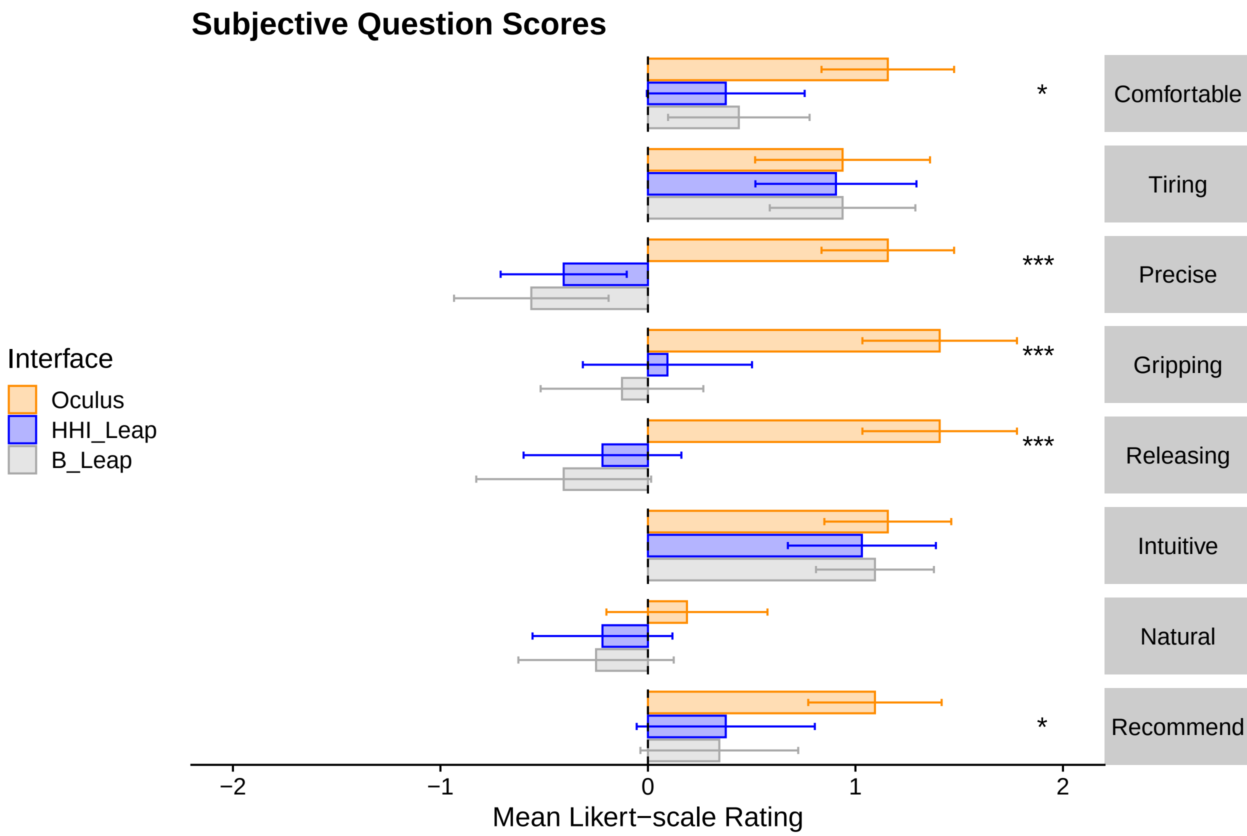

```
## TOST results:
## t-value lower bound: 1.41    p-value lower bound: 0.085
## t-value upper bound: -1.99   p-value upper bound: 0.028
## degrees of freedom : 31
##
## Equivalence bounds (Cohen's dz):
## low eqbound: -0.3
## high eqbound: 0.3
##
## Equivalence bounds (raw scores):
## low eqbound: -0.2994
## high eqbound: 0.2994
##
## TOST confidence interval:
## lower bound 90% CI: -0.35
## upper bound 90% CI:  0.248
##
## NHST confidence interval:
## lower bound 95% CI: -0.411
## upper bound 95% CI:  0.309
##
## Equivalence Test Result:
## The equivalence test was non-significant, t(31) = 1.408, p = 0.0845, given equivalence bounds of -0.
## Null Hypothesis Test Result:
```

```
## The null hypothesis test was non-significant, t(31) = -0.289, p = 0.774, given an alpha of 0.05.
## Based on the equivalence test and the null-hypothesis test combined, we can conclude that the observe
```

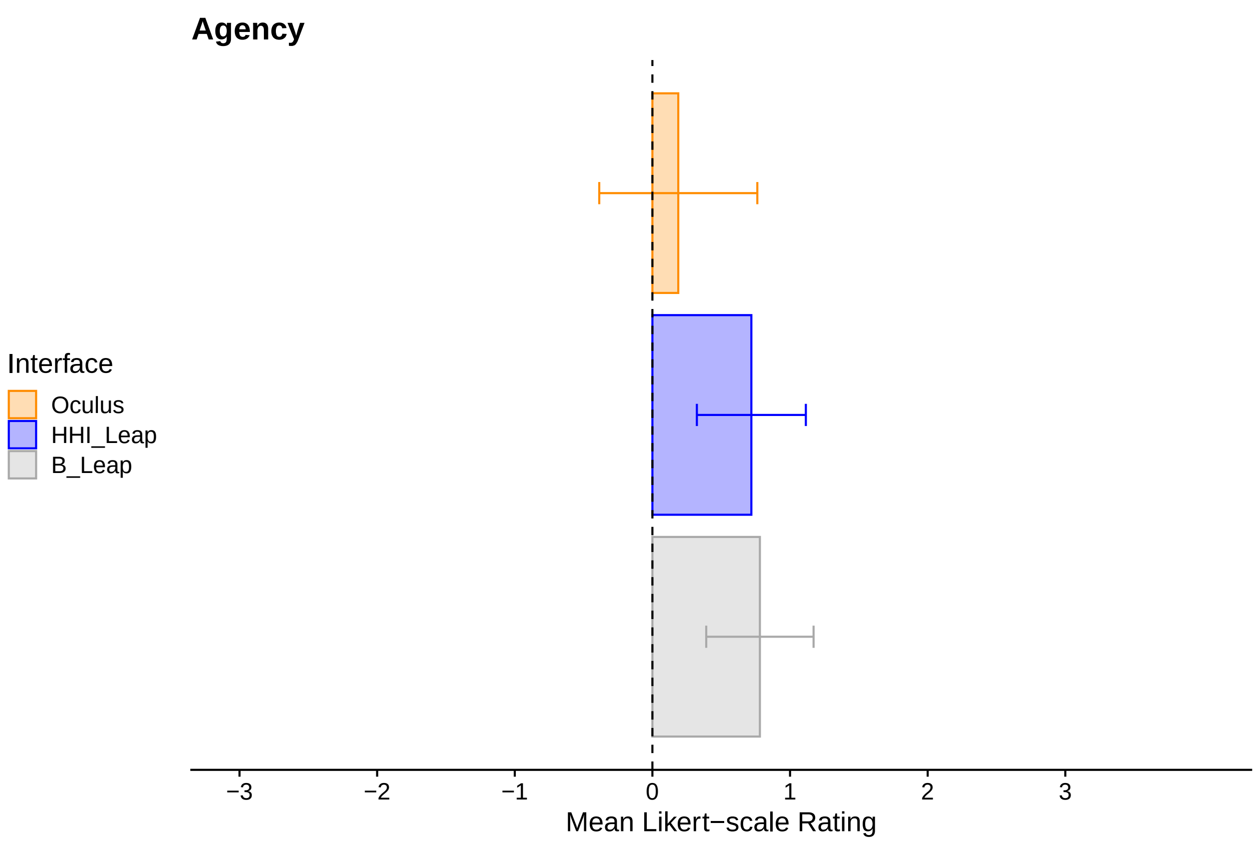
Likert Plots

```r
myquestions = c("satisfaction", "recommend", "agency", "natural", "intuitive", "releasing",  "gripping"
question_labels=c("Satisfaction", "Recommend", "Agency", "Natural", "Intuitive", "Releasing",  "Grippin

subjective_means <- likert_scores %>% group_by(Interface, question) %>% get_summary_stats(show=c("mean"
  mutate(question=factor(question, levels=myquestions, labels=question_labels))

subjective_means[which(subjective_means$question %in% c("Agency","Satisfaction")),"scale"] <- "7pt"
subjective_means[which(not(subjective_means$question %in% c("Agency","Satisfaction"))),"scale"] <- "5pt

# View(left_join(subjective_means, t.tests.likert.all.vs.hhi %>% mutate(question=factor(question, level

for (i in 1:nrow(subjective_means)){
  if (subjective_means[i,"scale"]=="7pt"){subjective_means[i,"converted_score"] <- subjective_means[i,"
}

#View(subjective_means)

# classify above or below avg.

# ggplot(subjective_means %>% filter(scale=="5pt"), aes(question, converted_score, color=Interface, fil
#   theme_minimal()+
#   geom_pointrange(aes(ymin=converted_score-ci, ymax=converted_score+ci), position=position_dodge(widt
#   #geom_point(data=likert_scores %>% mutate(question=factor(question, levels=myquestions, labels=ques
#   #geom_bar(stat="identity", position="dodge", alpha=myalpha)+
#   #geom_errorbar(aes(ymin=mean-ci, ymax=mean+ci), position="dodge", show.legend = FALSE)+
#   scale_color_manual(values=mycolors)+#, guide_legend(title="Interface", reverse = FALSE))+
#   scale_fill_manual(values=mycolors)+#, guide_legend(title="Interface", reverse=FALSE))+
#   coord_flip()+
#   theme(legend.position="bottom")+#, axis.text = element_text(size=likert_lab_size), plot.title = ele
#   scale_y_continuous(limits=c(-2,2))+
#   #scale_x_discrete(position = "top")+
#   #theme(strip.text.y = element_text(angle=0))+
#   geom_hline(aes(yintercept = 0), linetype=2)+
#   labs(title="Subjective Questionnaire Scores", y="Response (Mean)", x="")
# ggsave("subjective_scores_new.jpg")

# ggplot(subjective_means %>% filter(question=="Agency"), aes(question, mean, color=Interface, fill=Int
#   theme_minimal()+
#   geom_pointrange(aes(ymin=mean-ci, ymax=mean+ci), position=position_dodge(width=.5))+
#   #geom_bar(stat="identity", position="dodge", alpha=myalpha)+
#   #geom_errorbar(aes(ymin=mean-ci, ymax=mean+ci), position="dodge", show.legend = FALSE)+
#   scale_color_manual(values=mycolors)+#, guide_legend(title="Interface", reverse = FALSE))+
#   scale_fill_manual(values=mycolors)+#, guide_legend(title="Interface", reverse=FALSE))+
#   coord_flip()+
#   theme(legend.position="bottom", axis.text = element_text(size=likert_lab_size), plot.title = elemen
#   scale_y_continuous(limits=c(1,7), breaks=c(1:7))+
#   #theme(strip.text.y = element_text(angle=0))+
#   geom_hline(aes(yintercept = 4), linetype=2)+
```

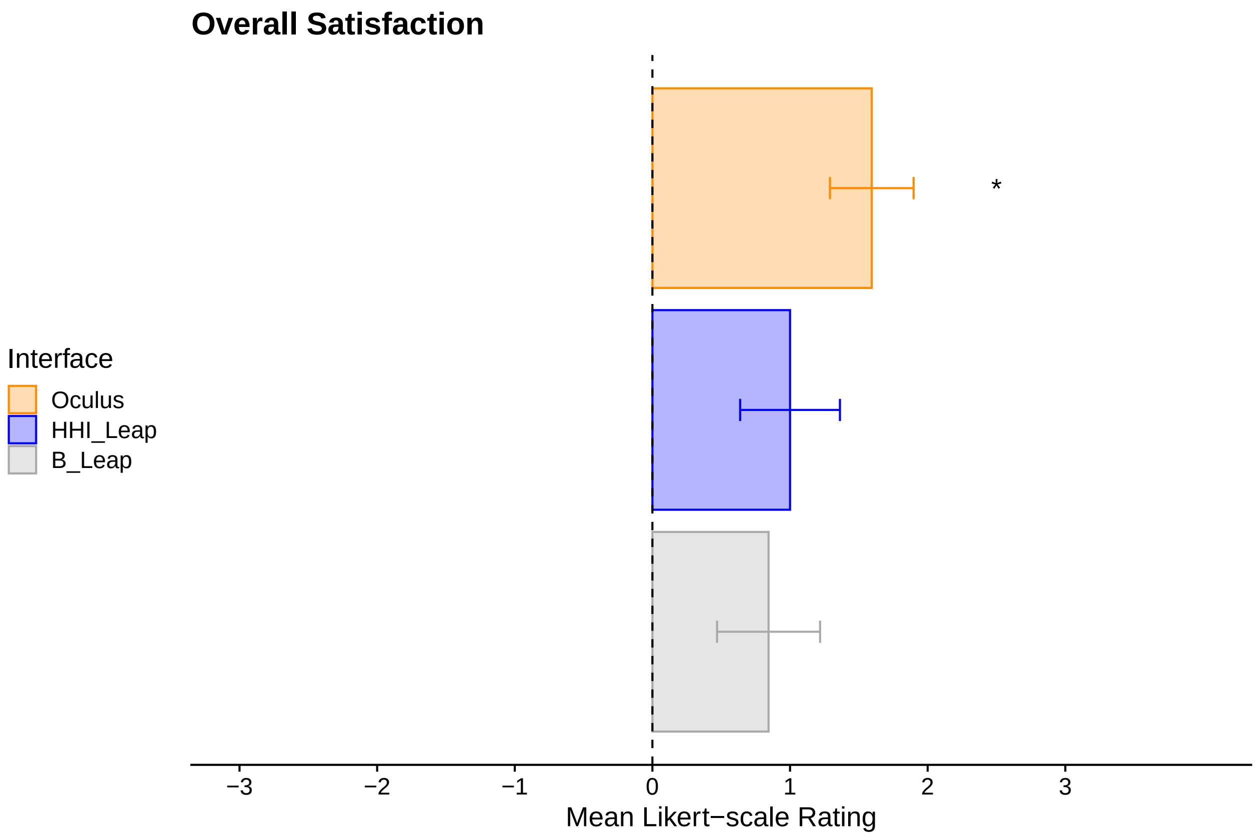
```
#   labs(title="Subjective Questionnaire Scores", y="Response (Mean)", x="")
# ggsave("subjective_scores_agency_new.jpg")

# ggplot(subjective_means %>% filter(question=="Satisfaction"), aes(question, mean, color=Interface, fi
#   theme_minimal()+
#   geom_pointrange(aes(ymin=mean-ci, ymax=mean+ci), position=position_dodge(width=.5))+
#   #geom_bar(stat="identity", position="dodge", alpha=myalpha)+
#   #geom_errorbar(aes(ymin=mean-ci, ymax=mean+ci), position="dodge", show.legend = FALSE)+
#   scale_color_manual(values=mycolors)+#, guide_legend(title="Interface", reverse = FALSE))+
#   scale_fill_manual(values=mycolors)+#, guide_legend(title="Interface", reverse=FALSE))+
#   coord_flip()+
#   theme(legend.position="bottom", axis.text = element_text(size=likert_lab_size), plot.title = elemen
#   scale_y_continuous(limits=c(1,7), breaks=c(1:7))+
#   #theme(strip.text.y = element_text(angle=0))+
#   geom_hline(aes(yintercept = 4), linetype=2)+
#   labs(title="Subjective Questionnaire Scores", y="Response (Mean)", x="")
# #ggsave("subjective_scores_satisfaction_new.jpg")

# ggplot(subjective_means %>% filter(scale=="5pt"), aes(question, converted_score, color=Interface, fil
#   geom_bar(stat="identity", position="dodge", alpha=myalpha)+
#   geom_errorbar(aes(ymin=converted_score-ci, ymax=converted_score+ci), position=position_dodge(width=
#   scale_color_manual(values=mycolors, guide_legend(title="Interface", reverse = TRUE))+
#   scale_fill_manual(values=mycolors, guide_legend(title="Interface", reverse=TRUE))+
#   scale_y_continuous(limits = c(-2,3))+
#   coord_flip()+
#   stat_pvalue_manual(data=t.tests.likert.all.vs.hhi %>% mutate(question=factor(question, levels=myques
#   scale_y_continuous(breaks=c(-2:3), limits=(c(-2,2)))+#labels = c(-2,-1,0,1,2,3))+
#   theme(legend.position = "bottom")+# strip.text.y = element_text(angle=0))+
#   geom_hline(aes(yintercept = 0), linetype=2)+
#   labs(title="Subjective Questionnaire Scores")#+facet_grid(question ~ .)

# good old fashioned bar chart
ggplot(subjective_means %>% mutate(question=fct_rev(question)) %>% filter(scale=="5pt"), aes(Interface,
  #theme_minimal()+
  geom_bar(stat="identity", position="dodge", alpha=myalpha)+
  geom_errorbar(aes(ymin=converted_score-ci, ymax=converted_score+ci), position=position_dodge(width=.9)
  scale_color_manual(values=mycolors, guide_legend(title="Interface", reverse = TRUE))+
  scale_fill_manual(values=mycolors, guide_legend(title="Interface", reverse=TRUE))+
  coord_flip()+

  stat_pvalue_manual(data=t.tests.likert.all.vs.hhi %>% mutate(question=factor(question, levels=myquest

  scale_y_continuous(breaks=c(-2:3), limits=(c(-2,2)))+#labels = c(-2,-1,0,1,2,3))+
  theme(legend.position = "left",  axis.text.y = element_blank(), axis.line.y=element_blank(), axis.tic
  geom_hline(aes(yintercept = 0), linetype=2)+
  labs(title="Subjective Question Scores", y="Mean Likert-scale Rating", x=NULL)+
  guides(fill = guide_legend(reverse = TRUE), color=guide_legend(reverse=TRUE))+
  facet_grid(question ~ .)
```



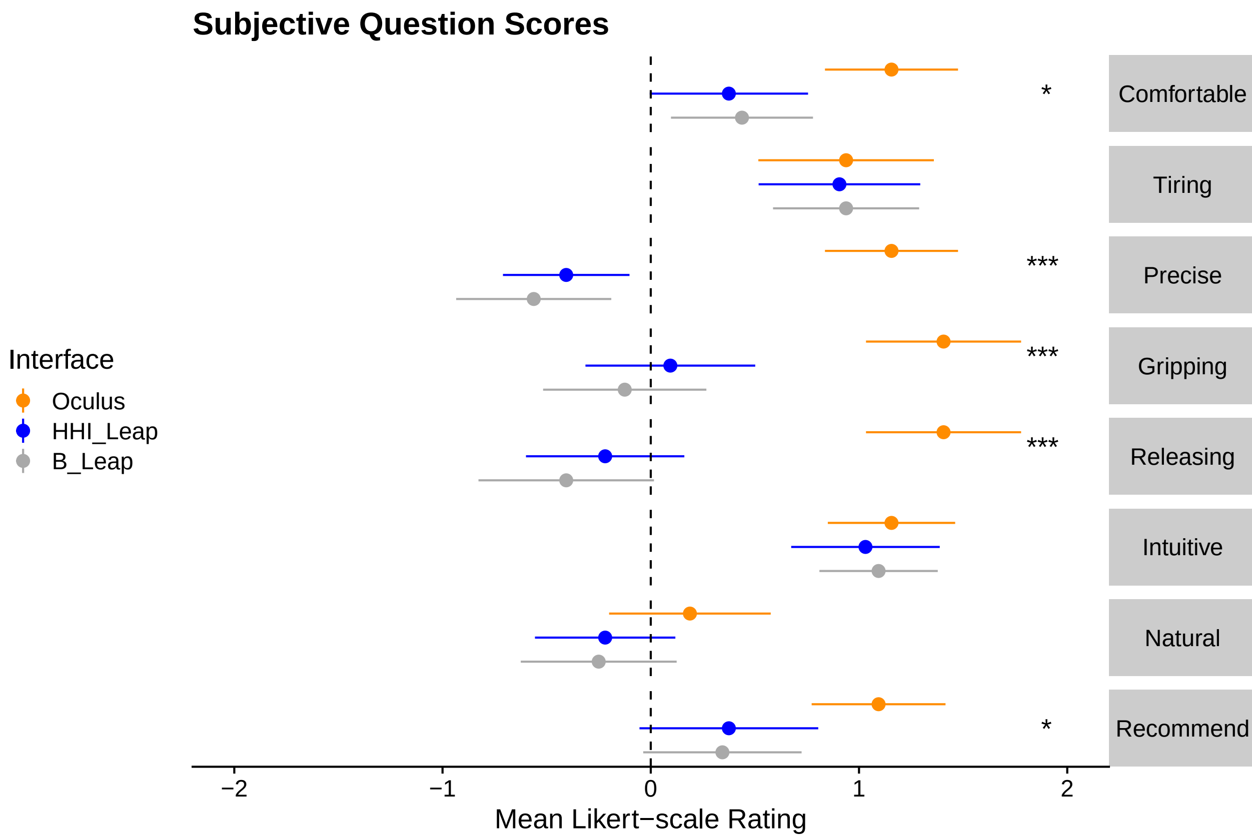

```
ggsave("subjective_questions_new_bar.jpg")

ggplot(subjective_means %>% filter(question=="Agency"), aes(Interface, converted_score, color=Interface
  #theme_minimal()+
  geom_bar(stat="identity", alpha=myalpha)+
  geom_errorbar(aes(ymin=converted_score-ci, ymax=converted_score+ci), width=.1, show.legend = FALSE)+
  scale_color_manual(values=mycolors, guide_legend(title="Interface", reverse = TRUE))+
  scale_fill_manual(values=mycolors, guide_legend(title="Interface", reverse=TRUE))+
  coord_flip()+
  #stat_pvalue_manual(data=t.tests.likert.all.vs.hhi %>% mutate(question=factor(question, levels=myques
  scale_y_continuous(breaks=c(-3:3), limits=(c(-3,4)))+#labels = c(-2,-1,0,1,2,3))+
  theme(legend.position = "left",  axis.text.y = element_blank(), axis.line.y=element_blank(), axis.ticl
  geom_hline(aes(yintercept = 0), linetype=2)+
  labs(title="Agency", y="Mean Likert-scale Rating", x=NULL)+
  guides(fill = guide_legend(reverse = TRUE), color=guide_legend(reverse=TRUE))#+facet_grid(question ~
```




```r
ggsave("subjective_questions_new_bar_agency.jpg", height=2, width=7)

ggplot(subjective_means %>% filter(question=="Satisfaction"), aes(Interface, converted_score, color=Inte
  #theme_minimal()+
  geom_bar(stat="identity", position="dodge", alpha=myalpha)+
  geom_errorbar(aes(ymin=converted_score-ci, ymax=converted_score+ci), position=position_dodge(width=.9)
  scale_color_manual(values=mycolors, guide_legend(title="Interface", reverse = TRUE))+
  scale_fill_manual(values=mycolors, guide_legend(title="Interface", reverse=TRUE))+
  coord_flip()+
  stat_pvalue_manual(data=t.tests.likert.all.vs.hhi %>% mutate(question=factor(question, levels=myquest
  scale_y_continuous(breaks=c(-3:3), limits=(c(-3,4)))+#labels = c(-2,-1,0,1,2,3))+
  theme(legend.position = "left",  axis.text.y = element_blank(), axis.line.y=element_blank(), axis.tic
  geom_hline(aes(yintercept = 0), linetype=2)+
  labs(title="Overall Satisfaction", y="Mean Likert-scale Rating", x=NULL)+
  guides(fill = guide_legend(reverse = TRUE), color=guide_legend(reverse=TRUE))#+facet_grid(question ~
```




```
ggsave("subjective_questions_new_bar_satisfaction.jpg", width=7, height=2)


# pointrange version
ggplot(subjective_means %>% mutate(question=fct_rev(question)) %>% filter(scale=="5pt"), aes(Interface,
  #theme_minimal()+
  geom_pointrange(aes(ymin=converted_score-ci, ymax=converted_score+ci), position=position_dodge(width=
  scale_color_manual(values=mycolors, guide_legend(title="Interface", reverse = TRUE))+
  scale_fill_manual(values=mycolors, guide_legend(title="Interface", reverse=TRUE))+
  coord_flip()+
  stat_pvalue_manual(data=t.tests.likert.all.vs.hhi %>% mutate(question=factor(question, levels=myquest
  scale_y_continuous(breaks=c(-2:3), limits=(c(-2,2)))+#labels = c(-2,-1,0,1,2,3))+
  theme(legend.position = "left", axis.text.y = element_blank(), axis.line.y=element_blank(), axis.ticks
  geom_hline(aes(yintercept = 0), linetype=2)+
  labs(title="Subjective Question Scores", y="Mean Likert-scale Rating", x=NULL)+
  guides(fill = guide_legend(reverse = TRUE), color=guide_legend(reverse=TRUE))+
  facet_grid(question ~ .)
```




```
ggsave("subjective_questions_new_point.jpg")
```

Raincloud plots

Dot plots, density distribution, means and CI's with a middle score indicated by a dotted line. Note: Using the R commind "adjust=" to smooth density plots (otherwise they would show divets in between points on the Likert scale). Adjustment is set by "mysmoothing" variable near the top of this code block.

```
# plot configs (local)
star_size=6


# SUBJECTIVE QUESTIONS
# question_text <-
#   c("Q1 - Comfortable","Q2 - Precise","Q3 - Intuitive",
#   "Q4 - Tiring for the hand (-)", "Q5 - Difficulty gripping (-)",
#   "Q6 - Difficulty releasing (-)", "Q7 - Gripping and releasing were natural",
#   "Q8 - Would recommend to friends")

# Comfortable
temp_plot_data <- subject_data_all_long %>%
  group_by(Interface)%>%
  get_summary_stats(Q_1_Score)

comfortable_raincloud <- ggplot(subject_data_all_long, aes(x=Interface,y=Q_1_Score, fill=Interface, col
  geom_hline(aes(yintercept=3), linetype=3, alpha=.5)+
```

```
  geom_violinhalf(position = position_nudge(x = 0, y = 0), alpha=myalpha, adjust=mysmoothing)+
  #geom_point(position = position_jitter(width = .15, height=.08), size = .25)+
  geom_dotplot(binaxis = "y", binwidth = 1, position=position_nudge(x=-.05), stackdir="down", dotsize=0
  geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(0), colour =
  geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=round(mean,2)), position = positi
  geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(se*
  ylab('Response (1-5)')+xlab(NULL)+theme_cowplot()+guides(fill = FALSE, colour = FALSE)+
  scale_color_manual(values=c(mycolors))+ #scale_colour_brewer(palette = "Set2")+
  scale_fill_manual(values=c(mycolors))+ #scale_fill_brewer(palette = "Set2")+
  scale_y_continuous(breaks=c(1:5), labels=c("1","2","3","4","5"))+
  #coord_flip()+
  #stat_pvalue_manual(data=t.tests.likert.all.vs.hhi %>% filter(question == "comfortable", p.adj < 0.05
  theme(plot.title = element_text(size=title_size*.75), axis.title = element_text(size=axis_text_size))+
  labs(title="Comfortable")
comfortable_raincloud
ggsave("likert_comfortable.jpg")

# Q2 - Precise
temp_plot_data <- subject_data_all_long %>%
  group_by(Interface)%>%
  get_summary_stats(Q_2_Score)

precise_raincloud <- ggplot(subject_data_all_long, aes(x=Interface,y=Q_2_Score, fill=Interface, color=In
  geom_hline(aes(yintercept=3), linetype=3, alpha=.5)+
  geom_flat_violin(position = position_nudge(x = 0, y = 0), alpha=myalpha, adjust=mysmoothing)+
  #geom_point(position = position_jitter(width = .15, height=.08), size = .25)+
  geom_dotplot(binaxis = "y", binwidth = 1, position=position_nudge(x=-.05), stackdir="down", dotsize=0
  geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(0), colour =
  geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=round(mean,2)), position = positi
  geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(se*
  ylab('Response (1-5)')+xlab(NULL)+theme_cowplot()+guides(fill = FALSE, colour = FALSE)+
  scale_color_manual(values=c(mycolors))+ #scale_colour_brewer(palette = "Set2")+
  scale_fill_manual(values=c(mycolors))+ #scale_fill_brewer(palette = "Set2")+
  scale_y_continuous(breaks=c(1:5), labels=c("1","2","3","4","5"))+
  #coord_flip()+
  stat_pvalue_manual(data=t.tests.likert.all.vs.hhi %>% filter(question == "precise", p.adj < 0.05), lab
  theme(plot.title = element_text(size=title_size*.75), axis.title = element_text(size=axis_text_size))+
  labs(title="Precise")
precise_raincloud
ggsave("likert_precise.jpg")


# Q3 - Intuitive
temp_plot_data <- subject_data_all_long %>%
  group_by(Interface)%>%
  get_summary_stats(Q_3_Score)

intuitive_raincloud <- ggplot(subject_data_all_long, aes(x=Interface,y=Q_3_Score, fill=Interface, color=
  geom_hline(aes(yintercept=3), linetype=3, alpha=.5)+
  geom_flat_violin(position = position_nudge(x = 0, y = 0), alpha=myalpha, adjust=mysmoothing)+
  #geom_point(position = position_jitter(width = .15, height=.08), size = .25)+
  geom_dotplot(binaxis = "y", binwidth = 1, position=position_nudge(x=-.05), stackdir="down", dotsize=0
  geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(0), colour =
```

```
  geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=round(mean,2)), position = positi
  geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(se*
  ylab('Response (1-5)')+xlab(NULL)+theme_cowplot()+guides(fill = FALSE, colour = FALSE)+
  scale_color_manual(values=c(mycolors))+ #scale_colour_brewer(palette = "Set2")+
  scale_fill_manual(values=c(mycolors))+ #scale_fill_brewer(palette = "Set2")+
  scale_y_continuous(breaks=c(1:5), limits=c(1,5), labels=c("1","2","3","4","5"))+
  #coord_flip()+
  #stat_pvalue_manual(data=t.tests.likert.all.vs.hhi %>% filter(question == "intuitive", p.adj < 0.05),
  theme(plot.title = element_text(size=title_size*.75), axis.title = element_text(size=axis_text_size))
  labs(title="Intuitive")
intuitive_raincloud
ggsave("likert_intuitive.jpg")

# Q4 - tiring for hand
temp_plot_data <- subject_data_all_long %>%
  group_by(Interface)%>%
  get_summary_stats(Q_4_Score)

tiring_raincloud <- ggplot(subject_data_all_long, aes(x=Interface,y=Q_4_Score, fill=Interface, color=Int
  geom_hline(aes(yintercept=3), linetype=3, alpha=.5)+
  geom_flat_violin(position = position_nudge(x = 0, y = 0), alpha=myalpha, adjust=mysmoothing)+
  #geom_point(position = position_jitter(width = .15, height=.08), size = .25)+
  geom_dotplot(binaxis = "y", binwidth = 1, position=position_nudge(x=-.05), stackdir="down", dotsize=0
  geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(0), colour =
  geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=round(mean,2)), position = positi
  geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(se*
  ylab('Response (5 = least tiring)')+xlab(NULL)+theme_cowplot()+guides(fill = FALSE, colour = FALSE)+
  scale_color_manual(values=c(mycolors))+ #scale_colour_brewer(palette = "Set2")+
  scale_fill_manual(values=c(mycolors))+ #scale_fill_brewer(palette = "Set2")+
  scale_y_continuous(breaks=c(1:5), labels=c("1","2","3","4","5"))+
  #coord_flip()+
  #stat_pvalue_manual(data=t.tests.likert.all.vs.hhi %>% filter(question == "tiring", p.adj < 0.05), la
  theme(plot.title = element_text(size=title_size*.75), axis.title = element_text(size=axis_text_size))
  labs(title="Tiring for the hand")
tiring_raincloud
ggsave("likert_tiring.jpg")

# Q5 - gripping
temp_plot_data <- subject_data_all_long %>%
  group_by(Interface)%>%
  get_summary_stats(Q_5_Score)

gripping_raincloud <- ggplot(subject_data_all_long, aes(x=Interface,y=Q_5_Score, fill=Interface, color=I
  geom_hline(aes(yintercept=3), linetype=3, alpha=.5)+
  geom_flat_violin(position = position_nudge(x = 0, y = 0), alpha=myalpha, adjust=mysmoothing)+
  #geom_point(position = position_jitter(width = .15, height=.08), size = .25)+
  geom_dotplot(binaxis = "y", binwidth = 1, position=position_nudge(x=-.05), stackdir="down", dotsize=0
  geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(0), colour =
  geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=round(mean,2)), position = positi
  geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(se*
  ylab('Response (1-5)')+xlab(NULL)+theme_cowplot()+guides(fill = FALSE, colour = FALSE)+
  scale_color_manual(values=c(mycolors))+ #scale_colour_brewer(palette = "Set2")+
  scale_fill_manual(values=c(mycolors))+ #scale_fill_brewer(palette = "Set2")+
```

```
  scale_y_continuous(breaks=c(1:5), labels=c("1","2","3","4","5"))+
  #coord_flip()+
  stat_pvalue_manual(data=t.tests.likert.all.vs.hhi %>% filter(question == "gripping", p.adj < 0.05), la
  theme(plot.title = element_text(size=title_size*.75), axis.title = element_text(size=axis_text_size))
  labs(title="Gripping")
gripping_raincloud
ggsave("likert_gripping.jpg")

# Q6 - difficulty releasing
temp_plot_data <- subject_data_all_long %>%
  group_by(Interface)%>%
  get_summary_stats(Q_6_Score)

releasing_raincloud <- ggplot(subject_data_all_long, aes(x=Interface,y=Q_6_Score, fill=Interface, color=
  geom_hline(aes(yintercept=3), linetype=3, alpha=.5)+
  geom_flat_violin(position = position_nudge(x = 0, y = 0), alpha=myalpha, adjust=mysmoothing)+
  #geom_point(position = position_jitter(width = .15, height=.08), size = .25)+
  geom_dotplot(binaxis = "y", binwidth = 1, position=position_nudge(x=-.05), stackdir="down", dotsize=0
  geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(0), colour =
  geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=round(mean,2)), position = positi
  geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(se*
  ylab('Response (1-5)')+xlab(NULL)+theme_cowplot()+guides(fill = FALSE, colour = FALSE)+
  scale_color_manual(values=c(mycolors))+ #scale_colour_brewer(palette = "Set2")+
  scale_fill_manual(values=c(mycolors))+ #scale_fill_brewer(palette = "Set2")+
  scale_y_continuous(breaks=c(1:5), labels=c("1","2","3","4","5"))+
  #coord_flip()+
  stat_pvalue_manual(data=t.tests.likert.all.vs.hhi %>% filter(question == "releasing", p.adj < 0.05), l
  theme(plot.title = element_text(size=title_size*.75), axis.title = element_text(size=axis_text_size))
  labs(title="Releasing")
releasing_raincloud
ggsave("likert_releasing.jpg")

# Q7 - grip and release was natural
temp_plot_data <- subject_data_all_long %>%
  group_by(Interface)%>%
  get_summary_stats(Q_7_Score)

natural_raincloud <- ggplot(subject_data_all_long, aes(x=Interface,y=Q_7_Score, fill=Interface, color=In
  geom_hline(aes(yintercept=3), linetype=3, alpha=.5)+
  geom_flat_violin(position = position_nudge(x = 0, y = 0), alpha=myalpha, adjust=mysmoothing)+
  #geom_point(position = position_jitter(width = .15, height=.08), size = .25)+
  geom_dotplot(binaxis = "y", binwidth = 1, position=position_nudge(x=-.05), stackdir="down", dotsize=0
  geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(0), colour =
  geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=round(mean,2)), position = positi
  geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(se*
  ylab('Response (1-5)')+xlab(NULL)+theme_cowplot()+guides(fill = FALSE, colour = FALSE)+
  scale_color_manual(values=c(mycolors))+ #scale_colour_brewer(palette = "Set2")+
  scale_fill_manual(values=c(mycolors))+ #scale_fill_brewer(palette = "Set2")+
  scale_y_continuous(breaks=c(1:5), labels=c("1","2","3","4","5"))+
  #coord_flip()+
  #stat_pvalue_manual(data=t.tests.likert.all.vs.hhi %>% filter(question == "natural", p.adj < 0.05), la
  theme(plot.title = element_text(size=title_size*.75), axis.title = element_text(size=axis_text_size))
  labs(title="Natural")
```

```
natural_raincloud
ggsave("likert_natural.jpg")

# Q8 - would recommend to friends
temp_plot_data <- subject_data_all_long %>%
  group_by(Interface)%>%
  get_summary_stats(Q_8_Score)

recommend_raincloud <- ggplot(subject_data_all_long, aes(x=Interface,y=Q_8_Score, fill=Interface, color=
  #geom_point(position = position_jitter(width = .15, height=.08), size = .25)+
  geom_hline(aes(yintercept=3), linetype=3, alpha=.5)+
  geom_flat_violin(position = position_nudge(x = 0, y = 0), alpha=myalpha, adjust=mysmoothing)+
  geom_dotplot(binaxis = "y", binwidth = 1, position=position_nudge(x=-.05), stackdir="down", dotsize=0
  geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(0), colour =
  geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=round(mean,2)), position = positi
  geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(se*
  ylab('Response (1-5)')+xlab(NULL)+theme_cowplot()+guides(fill = FALSE, colour = FALSE)+
  scale_color_manual(values=c(mycolors))+ #scale_colour_brewer(palette = "Set2")+
  scale_fill_manual(values=c(mycolors))+ #scale_fill_brewer(palette = "Set2")+
  scale_y_continuous(breaks=c(1:5), labels=c("1","2","3","4","5"))+
  #coord_flip()+
  #stat_pvalue_manual(data=t.tests.likert.all.vs.hhi %>% filter(question == "recommend", p.adj < 0.05),
  theme(plot.title = element_text(size=title_size*.75), axis.title = element_text(size=axis_text_size))
  labs(title="Recommend")
recommend_raincloud
ggsave("likert_recommend.jpg")

# agency
temp_plot_data <- subject_data_all_long %>%
  group_by(Interface) %>%
  get_summary_stats(agency)

agency_raincloud <- ggplot(subject_data_all_long, aes(x=Interface,y=agency, fill=Interface, color=Inter
  geom_hline(aes(yintercept=4), linetype=3, alpha=.5)+
  geom_flat_violin(position = position_nudge(x = 0, y = 0), alpha=myalpha, adjust=mysmoothing)+
  #geom_point(position = position_jitter(width = .15, height=.08), size = .25)+
  geom_dotplot(binaxis = "y", binwidth = 1, position=position_nudge(x=-.05), stackdir="down", dotsize=0
  geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(0), colour =
  geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=round(mean,2)), position = positi
  geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(se*
  ylab('Response (1-7)')+xlab(NULL)+theme_cowplot()+guides(fill = FALSE, colour = FALSE)+
  scale_color_manual(values=c(mycolors))+ #scale_colour_brewer(palette = "Set2")+
  scale_fill_manual(values=c(mycolors))+ #scale_fill_brewer(palette = "Set2")+
  scale_y_continuous(breaks=c(1:7), labels=c("1","2","3","4","5","6","7"))+
  #coord_flip()+
  #stat_pvalue_manual(data=t.tests.likert.all.vs.hhi %>% filter(question == "agency", p.adj < 0.05), la
  theme(plot.title = element_text(size=title_size*.75), axis.title = element_text(size=axis_text_size))
  labs(title="Agency")
agency_raincloud
ggsave(file="likert_agency.jpg", width=9, height=6)

# overall satisfaction
temp_plot_data <- subject_data_all_long %>%
```

```r
  group_by(Interface)%>%
  get_summary_stats(satisfaction)

satisfaction_raincloud <- ggplot(subject_data_all_long, aes(x=Interface,y=satisfaction, fill=Interface,
  geom_hline(aes(yintercept=4), linetype=3, alpha=.5)+
  geom_flat_violin(position = position_nudge(x = 0, y = 0), alpha=.4, adjust=1.5)+
  #geom_point(position = position_jitter(width = .15, height=.08), size = .25)+
  geom_dotplot(binaxis = "y", binwidth = 1, position=position_nudge(x=-.05), stackdir="down", dotsize=0
  geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(0), colour
  geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=round(mean,2)), position = posit
  geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(se*
  ylab('Response (1-7)')+xlab(NULL)+theme_cowplot()+guides(fill = FALSE, colour = FALSE)+
  scale_color_manual(values=c(mycolors))+ #scale_colour_brewer(palette = "Set2")+
  scale_fill_manual(values=c(mycolors))+ #scale_fill_brewer(palette = "Set2")+
  scale_y_continuous(breaks=c(1:7), limits=c(1,8.5), labels=c("1","2","3","4","5","6","7"))+
  #coord_flip()+
  #stat_pvalue_manual(data=t.tests.likert.all.vs.hhi %>% filter(question == "satisfaction", p.adj < 0.0
  theme(plot.title = element_text(size=title_size*.75), axis.title = element_text(size=axis_text_size))
  labs(title="Overall satisfaction")
satisfaction_raincloud
#ggsave(raincloud_satisfaction, file="raincloud_satisfaction.jpg")
ggsave("likert_satisfaction.jpg")

# preferred condition (code is in an earlier section)
preferred_plot
ggsave("preferred.jpg")


# all
plot_grid(comfortable_raincloud, tiring_raincloud, ncol=1, labels=c("A","B"))
ggsave("likert_plots1.jpg", width=8, height=8)
plot_grid(precise_raincloud, gripping_raincloud, releasing_raincloud, ncol=1, labels=c("C","D","E"))
ggsave("likert_plots2.jpg", width=8, height=12)
plot_grid(intuitive_raincloud, natural_raincloud, agency_raincloud, ncol=1, labels=c("F","G","H"))
ggsave("likert_plots3.jpg", width=8, height=12)
plot_grid(satisfaction_raincloud, recommend_raincloud, preferred_plot, ncol = 1, labels=c("I","J","K"))
ggsave("likert_plots4.jpg", width=8, height=12)

plot_grid(intuitive_raincloud, natural_raincloud, agency_raincloud, ncol=1, labels=c("A","B","C"))
ggsave("likert_naturalness_main.jpg", width=8, height=12)
```

Difference Scores & Plots

```r
myquestions = c("satisfaction", "recommend", "agency", "natural", "intuitive", "releasing",  "gripping"


#plot difference scores

# compile difference scores
diff_scores <- subject_data_all_long %>% group_by(id) %>% select(id, Interface, Q_1_Score) %>% spread(In
  left_join(subject_data_all_long %>% group_by(id) %>% select(id, Interface, Q_2_Score) %>% spread(Inte
  left_join(subject_data_all_long %>% group_by(id) %>% select(id, Interface, Q_3_Score) %>% spread(Inte
  left_join(subject_data_all_long %>% group_by(id) %>% select(id, Interface, Q_4_Score) %>% spread(Inte
```

```
  left_join(subject_data_all_long %>% group_by(id) %>% select(id, Interface, Q_5_Score) %>% spread(Inte
  left_join(subject_data_all_long %>% group_by(id) %>% select(id, Interface, Q_6_Score) %>% spread(Inte
  left_join(subject_data_all_long %>% group_by(id) %>% select(id, Interface, Q_7_Score) %>% spread(Inte
  left_join(subject_data_all_long %>% group_by(id) %>% select(id, Interface, Q_8_Score) %>% spread(Inte
  left_join(subject_data_all_long %>% group_by(id) %>% select(id, Interface, agency) %>% spread(Interfa
  left_join(subject_data_all_long %>% group_by(id) %>% select(id, Interface, satisfaction) %>% spread(I

# plots
# generate descriptives; add in t-test results

temp_plot_data <- diff_scores %>% group_by(Interface) %>% select(-id, -Interface) %>% get_summary_stats

diff_scores_vs_leap <- temp_plot_data %>% filter(Interface=="vs_Leap") %>% left_join(t.tests.likert.all
  mutate(question = factor(question, levels=myquestions))

# reorder question labels

diff_scores_vs_oculus <- temp_plot_data %>% filter(Interface=="vs_Oculus") %>% left_join(t.tests.likert
  mutate(question = factor(question, levels=myquestions))

likert_lab_size = 25
likert_title_size = 30
likert_interface_lab = 12
pointrange_size = 1.2
star_size=8


# plot difference scores vs. B_Leap -- note stat_pvalue_manual
ggplot(diff_scores_vs_leap %>% filter(question != "agency", question != "satisfaction"), aes(question, m
  geom_pointrange(aes(ymax=mean+1.96*se, ymin=mean-1.96*se), size=pointrange_size)+
  theme_cowplot()+
  theme(axis.text = element_text(size=likert_lab_size), axis.title = element_text(size=likert_lab_size)
  labs(title="Individual subjective questions (5-point)", y="Mean difference score", x="Question")+
  coord_flip()+guides(color=FALSE, fill=FALSE)+
  scale_y_continuous(limits=c(-2,2))+
  scale_color_manual(values=c("black", "orange"))+
  #scale_color_brewer(palette="Paired")+
  #stat_pvalue_manual(data=t.tests.likert.all.vs.hhi %>% filter(question != "agency", question != "sati
  geom_hline(yintercept=0, linetype=2)+
  geom_label(aes(x=8.3, y=-1, label="B_Leap"), fill=mycolors[1], alpha=.4, color="white", size=likert_i
  geom_label(aes(x=8.3, y=1.2, label="HHI Leap"), fill=mycolors[2], alpha=.4,color="white", size=likert
#ggsave("diff_scores_vs_leap.jpg", width=12, height =10)

# plot all difference scores vs. B_Leap
  x_guide = 10.35
ggplot(diff_scores_vs_leap, aes(question, mean))+
  geom_pointrange(aes(ymax=mean+1.96*se, ymin=mean-1.96*se), size=pointrange_size)+
  theme_cowplot()+
  theme(axis.text = element_text(size=likert_lab_size), axis.title = element_text(size=likert_lab_size)
  labs(title="Individual subjective questions", y="Mean difference score", x="Question")+
  coord_flip()+guides(color=FALSE, fill=FALSE)+
  scale_y_continuous(limits=c(-2,2))+
  scale_color_manual(values=c("black", "orange"))+
```

```
  #scale_color_brewer(palette="Paired")+
  stat_pvalue_manual(data=t.tests.likert.all.vs.hhi %>% filter (group2=="B_Leap"), x="question", label =
  geom_hline(yintercept=0, linetype=2)+
  geom_label(aes(x=x_guide, y=-1, label="B_Leap"), fill=mycolors[1], alpha=.4, color="white", size=liker
  geom_label(aes(x=x_guide, y=1.2, label="HHI Leap"), fill=mycolors[2], alpha=.4,color="white", size=lil
ggsave("diff_scores_vs_leap.jpg", width=12, height =10)


# plot difference scores vs. Oculus
ggplot(diff_scores_vs_oculus %>% filter(question != "agency", question != "satisfaction"), aes(question
  geom_pointrange(aes(ymax=mean+1.96*se, ymin=mean-1.96*se), size=pointrange_size)+
  #geom_pointrange(aes(ymax=mean+1.96*se, ymin=mean-1.96*se, color=p.adj<0.05), size=pointrange_size)+
  theme_cowplot()+
  theme(axis.text = element_text(size=likert_lab_size), axis.title = element_text(size=likert_lab_size)
  labs(title="Individual subjective questions (5-point)", y="Mean difference score", x="Question")+
  coord_flip(ylim=c(-2.5,2))+
  guides(color=FALSE)+
  scale_color_manual(values=c("black", "blue"))+
  scale_fill_manual(values=c("green3","blue"))+
  #scale_color_brewer(palette="Paired")+
  stat_pvalue_manual(data=t.tests.likert.all.vs.hhi %>% filter(question != "agency", question != "satis
  geom_hline(yintercept=0, linetype=2)+
  geom_label(aes(x=8.3, y=-1.5, label="Oculus"), fill=mycolors[3], alpha=.4, color="white",size=likert_
  geom_label(aes(x=8.3, y=1.2, label="HHI Leap"), fill=mycolors[2], alpha=.4, color="white",size=likert
#ggsave("diff_scores_vs_oculus.jpg", width=12, height = 10)

# plot all difference scores vs. Oculus
    x_guide = 10.35
ggplot(diff_scores_vs_oculus, aes(question, mean))+
  geom_pointrange(aes(ymax=mean+1.96*se, ymin=mean-1.96*se), size=pointrange_size)+
  theme_cowplot()+
  theme(axis.text = element_text(size=likert_lab_size), axis.title = element_text(size=likert_lab_size)
  labs(title="Individual subjective questions", y="Mean difference score", x="Question")+
  coord_flip(ylim=c(-2.5,2))+
  guides(color=FALSE)+
  stat_pvalue_manual(data=t.tests.likert.all.vs.hhi %>% filter(group2=="Oculus"), x="question", label =
  geom_hline(yintercept=0, linetype=2)+
  geom_label(aes(x=x_guide, y=-1.5, label="Oculus"), fill=mycolors[3], alpha=.4, color="white",size=like
  geom_label(aes(x=x_guide, y=1.2, label="HHI Leap"), fill=mycolors[2], alpha=.4, color="white",size=lil
ggsave("diff_scores_vs_oculus.jpg", width=12, height = 10)


# agency and satisfaction vs leap
ggplot(diff_scores_vs_leap %>% filter(question=="agency" | question=="satisfaction"), aes(question, mear
  geom_pointrange(aes(ymax=mean+1.96*se, ymin=mean-1.96*se), size = pointrange_size)+
  theme_cowplot()+
  theme(axis.text = element_text(size=likert_lab_size), plot.title = element_text(size=likert_title_size
  labs(title="Individual subjective questions (7-point)", y="Mean difference score", x="Question")+coord
  scale_color_manual(values=c("black", "orange"))+
  scale_y_continuous(limits=c(-2,2))+
  #scale_color_brewer(palette="Paired")+
  stat_pvalue_manual(data=t.tests.likert.all.vs.hhi %>% filter(question=="agency" | question=="satisfac
  geom_hline(yintercept=0, linetype=2)+
```

```r
  geom_label(aes(x=2.4, y=-.9, label="B_Leap"), fill=mycolors[1], alpha=.7, color="white", size=likert_i
  geom_label(aes(x=2.4, y=.8, label="HHI Leap"), fill=mycolors[2], alpha=.7, color="white",size=likert_i
#ggsave("diff_scores_agency_satisfaction_vs_leap.jpg", width=12, height=4.5)


# agency and satisfaction vs oculus
ggplot(diff_scores_vs_oculus %>% filter(question == "agency" | question == "satisfaction"), aes(questio
  #geom_boxplot(outlier.size = .25, outlier.alpha = .6)+
  geom_pointrange(aes(color=p.adj<0.05, ymax=mean+1.96*se, ymin=mean-1.96*se), size=pointrange_size)+
  #geom_violinhalf(data=likert_scores %>% filter(question == "agency" | question == "satisfaction", Int
   #geom_flat_violin(data=diff_scores %>% filter(Interface=="vs_Oculus") %>% gather(question, score, c(
  theme_cowplot()+
  theme(axis.text = element_text(size=likert_lab_size), plot.title = element_text(size=likert_title_size
  labs(title="Individual subjective questions (7-point)", y="Mean difference score", x="Question")+coord
  guides(color=FALSE, fill=FALSE)+
  scale_color_manual(values=c("black", "blue"))+#scale_fill_manual(values=c("grey","lightblue"))+
  #scale_fill_brewer(palette="Set2")+
  stat_pvalue_manual(data=t.tests.likert.all.vs.hhi %>% filter(question=="agency" | question=="satisfact
  geom_hline(yintercept=0, linetype=2)+
  geom_label(aes(x=2.4, y=-.9, label="Oculus"), fill=mycolors[3], alpha=.7, color="white",size=likert_i
  scale_y_continuous(limits=c(-2,2))+
  geom_label(aes(x=2.4, y=.9, label="HHI Leap"), fill=mycolors[2], alpha=.7, color="white", size=likert_
#ggsave("diff_scores_agency_satisfaction_vs_oculus.jpg", width=12, height =4.5)
```

SUS

Statistical test: Looks like the SUS scores are close enough to a normal distribution so that the means and medians are not radically different. While the Oculus data fails the shapiro test (probably due to its extreme outlier) and the data is generally very skewed, the mean and median are nearly the same. Therefore, we'll use the mean as the measure of central tendency for this data.

The scores are normally distributed. Therefore, even though the dependent variable is not continuous, a T-Test seems appropriate. T-Tests have been found to be more robust than Wilcox tests, even when some assumptions are violated (Norman, 2010; Meed et al., 2010).

We can use the guidelines offered by Bangor, Kortum and Miller (2009): <50: Not acceptable
50–70: Marginal >70: Acceptable

4 out of 32 people rated the HHI Leap as "not acceptable" on the SUS. Only 1 out of 32 did this for the Leap. How do I tell if this is statistically significant?

```r
#SUS score means and medians
temp_plot_data <- subject_data_all_long %>%
  group_by(Interface) %>%
  summarise(mean=mean(SUS), sd=sd(SUS), se=sd/sqrt(sample_size), median=median(SUS))

stat.test <- subject_data_all_long %>%
  ungroup(.) %>%
  t_test(SUS ~ Interface, paired=TRUE, comparisons=list(c("B_Leap","HHI_Leap"),c("HHI_Leap","Oculus")))
  adjust_pvalue() %>% mutate(Interface=group1) %>%
  left_join(subject_data_all_long %>% ungroup(.) %>% cohens_d(SUS ~ Interface, paired=TRUE) %>% select(g
stat.test
```

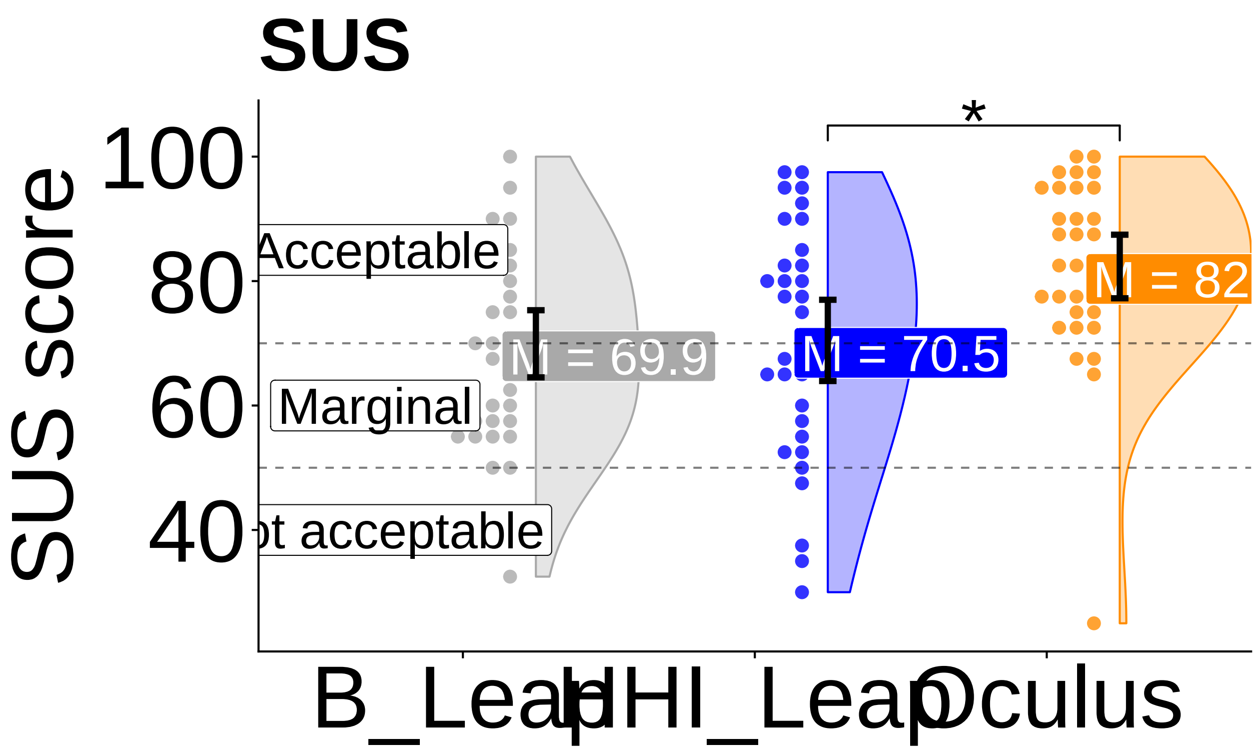
```
## # A tibble: 2 x 13
```

```
##   .y.   group1 group2    n1    n2 statistic    df     p p.adj p.adj.signif
##   <chr> <chr>  <chr>  <int> <int>     <dbl> <dbl> <dbl> <dbl> <chr>
## 1 SUS   B_Leap HHI_L~    32    32    -0.187    31 0.853 0.853 ns
## 2 SUS   HHI_L~ Oculus    32    32    -2.69     31 0.011 0.022 *
## # ... with 3 more variables: Interface <chr>, effsize <dbl>, magnitude <ord>
```

```
anova.test <-
  anova_summary(effect.size="pes",aov(SUS ~ Interface + Error(id/Interface), data=subject_data_all_long
anova.test
```

```
##      Effect DFn DFd     F     p p<.05   pes
## 1 Interface   2  62 7.132 0.002     * 0.187
```

```
# for export
ttest.SUS <- stat.test %>% select(-Interface)
anova.SUS <- anova.test
descriptives.sus <- subject_data_all_long %>% group_by(Interface) %>% get_summary_stats(SUS, type = "co


# SUS scoring bin
SUS_scoring_bins <- subject_data_all_long %>%
  filter(SUS<50) %>% mutate(SUS_bin="Not acceptable") %>%
  bind_rows(subject_data_all_long %>%
            filter(SUS>=50 & SUS < 70) %>% mutate(SUS_bin="Marginal")) %>%
  bind_rows(subject_data_all_long %>%
            filter(SUS>=70) %>% mutate(SUS_bin="Acceptable")) %>%
  select(id, Interface, SUS, SUS_bin) %>%
  group_by(Interface) %>%
  count(SUS_bin) %>% mutate(SUS_bin_ratio=n/sample_size)
  # add plot position
  SUS_scoring_bins <- SUS_scoring_bins %>%
  filter(SUS_bin=="Not acceptable") %>% mutate(plot_pos=40) %>%
  bind_rows(SUS_scoring_bins %>% filter(SUS_bin=="Marginal") %>% mutate(plot_pos=60)) %>%
    bind_rows(SUS_scoring_bins %>% filter(SUS_bin=="Acceptable") %>% mutate(plot_pos=85))

# plot
SUS_raincloud <- ggplot(subject_data_all_long, aes(x=Interface, y=SUS, fill=Interface, color=Interface)
  geom_flat_violin(position = position_nudge(x = .25, y = 0), alpha=myalpha,adjust=mysmoothing)+
  geom_dotplot(binaxis = "y", stackratio=1.4, binwidth = 1, position=position_nudge(x=.2), stackdir="do
  scale_color_manual(values=mycolors)+#scale_color_brewer(palette="Set2")+
  scale_fill_manual(values=mycolors)+#scale_fill_brewer(palette = "Set2")+#coord_flip()+
  geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(.25), colou
  geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=paste0("M = ",round(mean,1))), p
  geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(se*
  ylab('SUS score')+xlab('')+theme_cowplot()+guides(colour = FALSE, label = FALSE, text=FALSE)+
  labs(title="SUS")+
  geom_hline(aes(yintercept=50), linetype=2, alpha=.5)+geom_hline(aes(yintercept=70), linetype=2, alpha
  geom_label(data=SUS_scoring_bins %>% filter(Interface=="B_Leap"), aes(x=.7, y=plot_pos, label=SUS_bin
  #geom_label(data=SUS_scoring_bins, aes(x=Interface, y=plot_pos, label=n), alpha=.6, size=9, color="wh
  stat_pvalue_manual(data=stat.test %>% filter(p.adj < 0.05), xmin="group1", xmax="group2", label = "p.
  theme(plot.title = element_text(size=title_size*1.25), axis.title = element_text(size=axis_text_size*

# export
SUS_raincloud
```




```r
ggsave(last_plot(), filename = "SUS_plot.jpg", width=14, height=8)

# are they normally distributed?
#B_Leap: yes
shapiro <- subject_data_all_long %>% filter(Interface=="B_Leap") %>%
ungroup(.) %>%
shapiro_test(SUS)
cat("\nShapiro says Leap data are normally distributed: ", shapiro$p>.05, "\n")
```

```
##
## Shapiro says Leap data are normally distributed:  TRUE
```

```r
#HHI_Leap: no
shapiro <- subject_data_all_long %>% filter(Interface=="HHI_Leap") %>%
ungroup(.) %>%
shapiro_test(SUS)
cat("\nShapiro says HHI data are normally distributed: ", shapiro$p>.05, "\n")
```

```
##
## Shapiro says HHI data are normally distributed:  TRUE
```

```r
#Oculus
shapiro <- subject_data_all_long %>% filter(Interface=="Oculus") %>%
ungroup(.) %>%
shapiro_test(SUS)
cat("\nShapiro says Oculus data are normally distributed: ", shapiro$p>.05, "\n")
```

```
##
## Shapiro says Oculus data are normally distributed:  FALSE
```

```r
# check out summary stats, including skewness and kurtosis
describeBy(subject_data_all_long %>% ungroup(.) %>% select(Interface, SUS), group = "Interface")
```

```
##
##  Descriptive statistics by group
## group: B_Leap
##            vars  n  mean    sd median trimmed   mad  min max range skew
## Interface*    1 32  1.00  0.00      1       1  0.00  1.0   1   0.0  NaN
## SUS           2 32 69.92 15.57     70      70 18.53 32.5 100  67.5 -0.1
##            kurtosis   se
## Interface*      NaN 0.00
## SUS           -0.69 2.75
## ------------------------------------------------------------
## group: HHI_Leap
##            vars  n  mean    sd median trimmed   mad min  max range  skew
## Interface*    1 32  2.00  0.00    2.0    2.00  0.00   2  2.0   0.0   NaN
## SUS           2 32 70.47 18.84   72.5   71.63 20.39  30 97.5  67.5 -0.38
##            kurtosis   se
## Interface*      NaN 0.00
## SUS           -0.86 3.33
## ------------------------------------------------------------
## group: Oculus
##            vars  n  mean    sd median trimmed   mad min max range  skew
## Interface*    1 32  3.00  0.00    3.0    3.00  0.00   3   3     0   NaN
## SUS           2 32 82.34 14.74   82.5   83.85 12.97  25 100    75 -1.69
##            kurtosis   se
## Interface*      NaN 0.00
## SUS            4.61 2.61
```

```r
# do Interfaces have equal variances?
levene <- subject_data_all_long %>% ungroup(.) %>%
levene_test(SUS ~ Interface, center=mean)
cat("\nLevene says data have equal variances: ", levene$p>.05, "\n")
```

```
##
## Levene says data have equal variances:  TRUE
```

```r
# anova
print("ANOVA: SUS - Interface")
```

```
## [1] "ANOVA: SUS - Interface"
```

```r
summary(aov(SUS ~ Interface + Error(id/Interface), data=subject_data_all_long))
```

```
##
## Error: id
##           Df Sum Sq Mean Sq F value Pr(>F)
## Residuals 31  11556   372.8
##
## Error: id:Interface
##           Df Sum Sq Mean Sq F value  Pr(>F)
```

```
## Interface  2   3153    1577   7.132 0.00163 **
## Residuals 62  13705     221
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova_summary(effect.size="pes",aov(SUS ~ Interface + Error(id/Interface), data=subject_data_all_long))
```

```
##      Effect DFn DFd     F     p p<.05   pes
## 1 Interface   2  62 7.132 0.002     * 0.187
```

```r
# t test and wilcox
compare_means(SUS ~ Interface, data=subject_data_all_long, method = "t.test", paired = TRUE)
```

```
## # A tibble: 3 x 8
##   .y.   group1   group2         p  p.adj p.format p.signif method
##   <chr> <chr>    <chr>      <dbl>  <dbl> <chr>    <chr>    <chr>
## 1 SUS   B_Leap   HHI_Leap 0.853   0.85   0.8528   ns       T-test
## 2 SUS   B_Leap   Oculus   0.00190 0.0057 0.0019   **       T-test
## 3 SUS   HHI_Leap Oculus   0.0114  0.023  0.0114   *        T-test
```

```r
compare_means(SUS ~ Interface, data=subject_data_all_long, method = "wilcox", paired = TRUE)
```

```
## # A tibble: 3 x 8
##   .y.   group1   group2         p  p.adj p.format p.signif method
##   <chr> <chr>    <chr>      <dbl>  <dbl> <chr>    <chr>    <chr>
## 1 SUS   B_Leap   HHI_Leap 0.581   0.580  0.5808   ns       Wilcoxon
## 2 SUS   B_Leap   Oculus   0.00121 0.0036 0.0012   **       Wilcoxon
## 3 SUS   HHI_Leap Oculus   0.00906 0.018  0.0091   **       Wilcoxon
```

```r
# normality check for t test
# leap HHI vs. B_Leap
group1<-"HHI_Leap"
group2<-"B_Leap"
t_test_dataset <- subject_data_all_long %>%
  filter(Interface==group1 | Interface==group2) %>%
  select(id, Interface, SUS)
# Shapiro-Wilk normality test for the differences
t_diff_dataset <- with(t_test_dataset,
        SUS[Interface == group1] - SUS[Interface == group2])
#hist(t_diff_dataset)
shapiro.test(t_diff_dataset)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  t_diff_dataset
## W = 0.95572, p-value = 0.209
```

```r
# t test
#t.test(SUS ~ Interface, data = t_test_dataset, paired = TRUE)
```

```r
# leap HHI vs. Oculus
group1<-"HHI_Leap"
group2<-"Oculus"
t_test_dataset <- subject_data_all_long %>%
  filter(Interface==group1 | Interface==group2) %>%
  select(id, Interface, SUS)
# Shapiro-Wilk normality test for the differences
t_diff_dataset <- with(t_test_dataset,
        SUS[Interface == group1] - SUS[Interface == group2])
#hist(t_diff_dataset)
shapiro.test(t_diff_dataset)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  t_diff_dataset
## W = 0.95435, p-value = 0.1914
```

```r
# t test
#t.test(SUS ~ Interface, data = t_test_dataset, paired = TRUE)

# individual subject sus scores
  SUS_subject_plot <- ggplot(subject_data_all_long, aes(id, SUS, fill=Interface, color=Interface))+
    #geom_bar(stat="identity", position="dodge")+
    geom_point(aes(shape=Interface), size=3)+
    scale_fill_brewer(palette="Set2")+scale_color_brewer(palette="Set2")+theme_minimal()+
    #facet_grid(. ~ Interface)+
    ggtitle("SUS scores by subject")
#  SUS_subject_plot

# chi sq goodness of fit using oculus as expected counts
SUS_scoring_bins %>% arrange(Interface)
```

```
## # A tibble: 9 x 5
## # Groups:   Interface [3]
##   Interface SUS_bin            n SUS_bin_ratio plot_pos
##   <fct>     <chr>          <int>         <dbl>    <dbl>
## 1 B_Leap    Not acceptable     1        0.0312       40
## 2 B_Leap    Marginal          14        0.438        60
## 3 B_Leap    Acceptable        17        0.531        85
## 4 HHI_Leap  Not acceptable     4        0.125        40
## 5 HHI_Leap  Marginal          11        0.344        60
## 6 HHI_Leap  Acceptable        17        0.531        85
## 7 Oculus    Not acceptable     1        0.0312       40
## 8 Oculus    Marginal           3        0.0938       60
## 9 Oculus    Acceptable        28        0.875        85
```

```r
chi_counts <- SUS_scoring_bins %>% filter(Interface=="HHI_Leap")
chi_expected <- SUS_scoring_bins %>% filter(Interface=="Oculus")
chisq.test(chi_counts$n, chi_expected$SUS_bin_ratio)
```

```
##
```

```
##  Pearson's Chi-squared test
##
## data:  chi_counts$n and chi_expected$SUS_bin_ratio
## X-squared = 6, df = 4, p-value = 0.1991
```

```
# build df of mean and sd for HHI Leap
SUS_mean_sd <- #left_join(get_summary_stats(temp_df, Distance,))
  data.frame(Measure="SUS",
      HHI_Leap_Mean = mean(subject_data_all_long[which(subject_data_all_long$Interface=="HHI_Leap"),"SU
      HHI_Leap_SD = sd(subject_data_all_long[which(subject_data_all_long$Interface=="HHI_Leap"),"SUS"])
      B_Leap_Mean = mean(subject_data_all_long[which(subject_data_all_long$Interface=="B_Leap"),"SUS"])
      B_Leap_SD = sd(subject_data_all_long[which(subject_data_all_long$Interface=="HHI_Leap"),"SUS"]),
      Oculus_Mean = mean(subject_data_all_long[which(subject_data_all_long$Interface=="Oculus"),"SUS"])
      Oculus_SD = sd(subject_data_all_long[which(subject_data_all_long$Interface=="Oculus"),"SUS"])) %>
```

Plot compilation export

```
# intuitive
plot_grid(intuitive_raincloud, natural_raincloud, agency_raincloud)  #, trainingtime_raincloud)
ggsave(last_plot(), filename = "intuitive_plots.jpg", width = 14, height = 8.5)

# ease of use w/ grab and release times plot_grid(tiring_raincloud,
# gripping_raincloud, releasing_raincloud, comfortable_raincloud,
# grabtime_raincloud, releasetime_Interface_raincloud) ggsave(last_plot(),
# filename='ease_of_use_plots.jpg', width=14, height=8.5)

# ease of use w / only subjective metrics
plot_grid(tiring_raincloud, gripping_raincloud, releasing_raincloud, comfortable_raincloud,
    SUS_raincloud)
ggsave(last_plot(), filename = "ease_of_use_plots.jpg", width = 14, height = 8.5)

# preference
plot_grid(recommend_raincloud, satisfaction_raincloud, preferred_plot)
ggsave(last_plot(), filename = "preference_plots.jpg", width = 14, height = 8.5)

# perception of performance
plot_grid(precise_raincloud, distance_Interface_raincloud + scale_y_reverse(), gripping_raincloud,
    grabtime_Interface_raincloud, releasing_raincloud, releasetime_Interface_raincloud,
    nrow = 3, ncol = 2)
ggsave(last_plot(), filename = "perception_of_performance_plots.jpg", width = 14,
    height = 8.5)

# SUS
SUS_raincloud
ggsave(last_plot(), filename = "SUS_plot.jpg", width = 14, height = 8)
```

Post-hoc exploratory

Non-equivalence

```
# TOSTpaired(32, m1, m2, sd1, sd2, r12, low_eqbound_dz, high_eqbound_dz, alpha,
# plot = TRUE, verbose = TRUE)
```

Time to learn

Practice time

```
# generate descriptives for raincloud
temp_plot_data <- subject_data_all_long %>% group_by(Interface) %>% get_summary_stats(practice_time)

stat.test.anova <-
anova_summary(effect.size="pes",aov(practice_time ~ Interface + Error(id/Interface), data=subject_data_a
stat.test.anova
```

```
##      Effect DFn DFd     F        p p<.05   pes
## 1 Interface   2  62 9.349 0.000283     * 0.232
```

```
#write.csv(stat.test.anova, file="dropcount_anova.csv")

stat.test.anova2 <-
anova_summary(effect.size="pes",aov(practice_time ~ Interface + Error(id/Interface), data=subject_data_a
stat.test.anova2
```

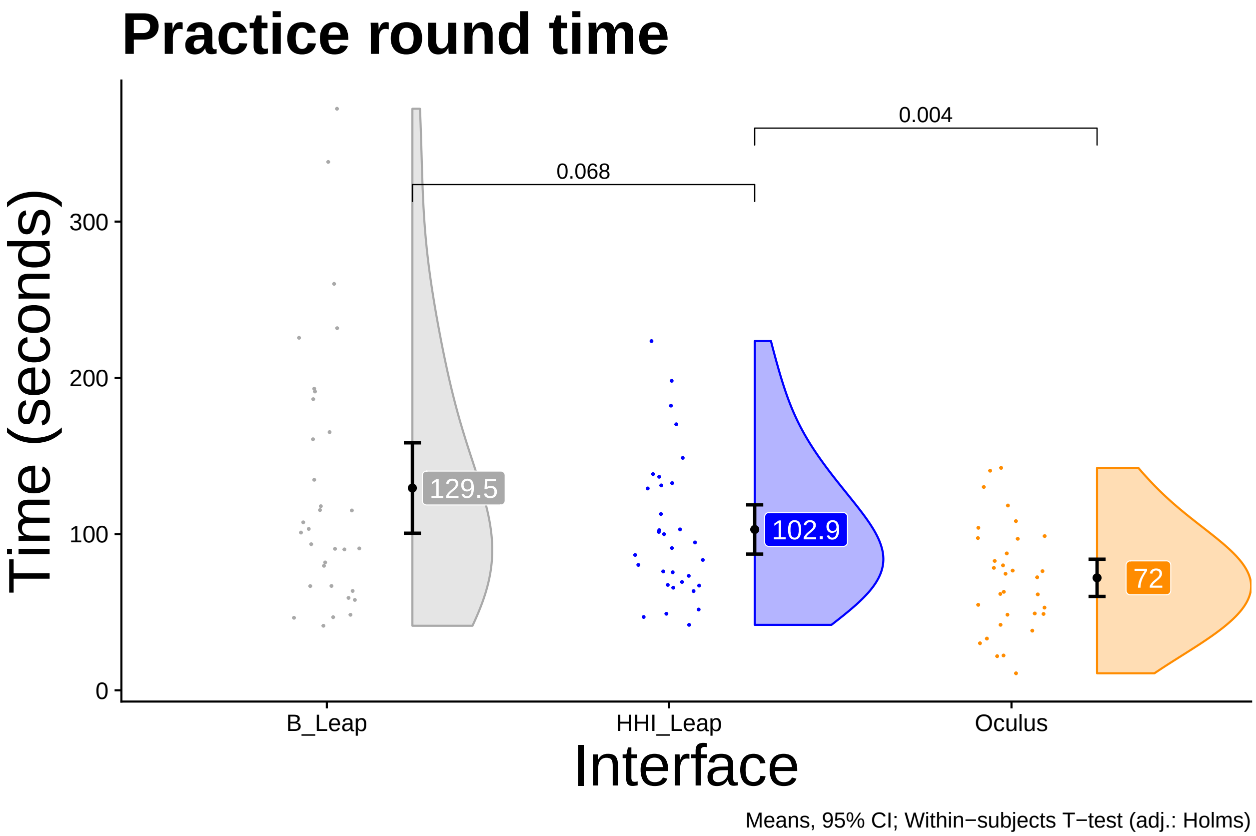
```
##      Effect DFn DFd     F     p p<.05   pes
## 1 Interface   1  31 3.573 0.068       0.103
```

```
stat.test <- subject_data_all_long %>%
  ungroup(.) %>%
  pairwise_t_test(practice_time ~ Interface, paired=TRUE, comparisons=list(c("B_Leap","HHI_Leap"),c("HHI
  mutate(Interface=group1, test="T-test")
stat.test
```

```
## # A tibble: 2 x 12
##   .y.   group1 group2    n1    n2 statistic    df     p p.adj p.adj.signif
##   <chr> <chr>  <chr>  <int> <int>     <dbl> <dbl> <dbl> <dbl> <chr>
## 1 prac~ B_Leap HHI_L~    32    32      1.89    31 0.068 0.068 ns
## 2 prac~ HHI_L~ Oculus    32    32      3.33    31 0.002 0.004 **
## # ... with 2 more variables: Interface <chr>, test <chr>
```

```
# raincloud training time
trainingtime_raincloud<-ggplot(subject_data_all_long, aes(x=Interface, y=practice_time, fill=Interface,
  #geom_flat_violin(position = position_nudge(x = .25, y = 0), alpha=.7, draw_quantiles=c(.25, .75))+#,
  geom_violinhalf(position = position_nudge(x = .25, y = 0), alpha=myalpha, draw_quantiles=c(.25, .75),
  geom_point(position = position_jitter(width = 0.1, height=0), size = .25)+ # the "rain"
geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=round(mean,1)), position = positio
  geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(.25), colou
```

```
  geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(se*
  ylab('Time (seconds)')+xlab('Interface')+theme_cowplot()+guides(fill = FALSE, colour = FALSE) +
  scale_color_manual(values=mycolors)+#scale_colour_brewer(palette = "Set2")+
  scale_fill_manual(values=mycolors)+#scale_fill_brewer(palette = "Set2", direction=1)+
  stat_pvalue_manual(data=stat.test, xmin="group1", xmax="group2", label = "p.adj", step.increase=.1, y
  labs(title="Practice round time", caption="Means, 95% CI; Within-subjects T-test (adj.: Holms)")+
  theme(plot.title = element_text(size=title_size), axis.title = element_text(size=axis_text_size))
trainingtime_raincloud
```




```
ggsave(last_plot(), filename="trainingtime_plot.jpg", width=12, height=7)

# transform for data output
stat.test <- stat.test %>% select(-.y., -n1, -n2, -Interface)%>% mutate(p=round(p, 4), p.adj=round(p.ad

stat.test.anova <- stat.test.anova %>% mutate(p=round(p, 4))

anova.Interface.trainingtime <- stat.test.anova
ttest.Interface.trainingtime <- stat.test
descriptives.Interface.trainingtime <- subject_data_all_long %>% group_by(Interface) %>% get_summary_sta
descriptives.Interface.trainingtime
```

```
## # A tibble: 3 x 7
##   Interface variable       mean    sd   min   max   iqr
##   <fct>     <chr>         <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 B_Leap    practice_time 129.   83.5  41.3  372. 104.
## 2 HHI_Leap  practice_time 103.   45.5  41.9  224.  62.6
```

```
## 3 Oculus    practice_time  72.0  34.4  10.9  142.  48.3
```

Learning Curve (slope)

```r
# What is the lowest number of trials remaining?
min_n <- unity_data_clean %>% group_by(id, Interface) %>% summarise(n=n()) %>% ungroup %>% summarise(mi
min_n <- min_n[[1]]
min_n
```

```
## [1] 20
```

```r
# # Resample all to that number, but keep order of SpawnOrder
# unity_data_downsampled <- unity_data_clean %>% group_by(id, Interface) %>% sample_n(min_n) %>%
# # Change SpawnOrder to 1-20
# mutate(SpawnOrder=dense_rank(SpawnOrder))

# the problem with resampling is that it is inconsistent, you get a different statistical result each t

# Instead of downsampling, just pick first min_n data points
unity_data_downsampled <- unity_data_clean %>% group_by(id, Interface) %>%
  mutate(SpawnOrder=dense_rank(SpawnOrder)) %>% #re-number starting at 1
  filter(SpawnOrder <= min_n) %>%
  ungroup %>% mutate(Interface=as.factor(Interface))#, SpawnOrder=as.factor(SpawnOrder))
```

Learning curve: Accuracy

```r
# anova to see if there is an effect of SpawnOrder*Interface
#summary(aov(Distance ~ Interface*SpawnOrder + Error(id/(Interface*SpawnOrder)), data=unity_data_downsa

anova_summary(aov(Distance ~ Interface*SpawnOrder + Error(id/(Interface*SpawnOrder)), data=unity_data_d
```

```
##                 Effect DFn DFd      F        p p<.05   ges
## 1            Interface   2  62 33.168 1.60e-10     * 0.344
## 2           SpawnOrder   1  31  0.782 3.83e-01       0.006
## 3 Interface:SpawnOrder   2  62  0.793 4.57e-01       0.007
```

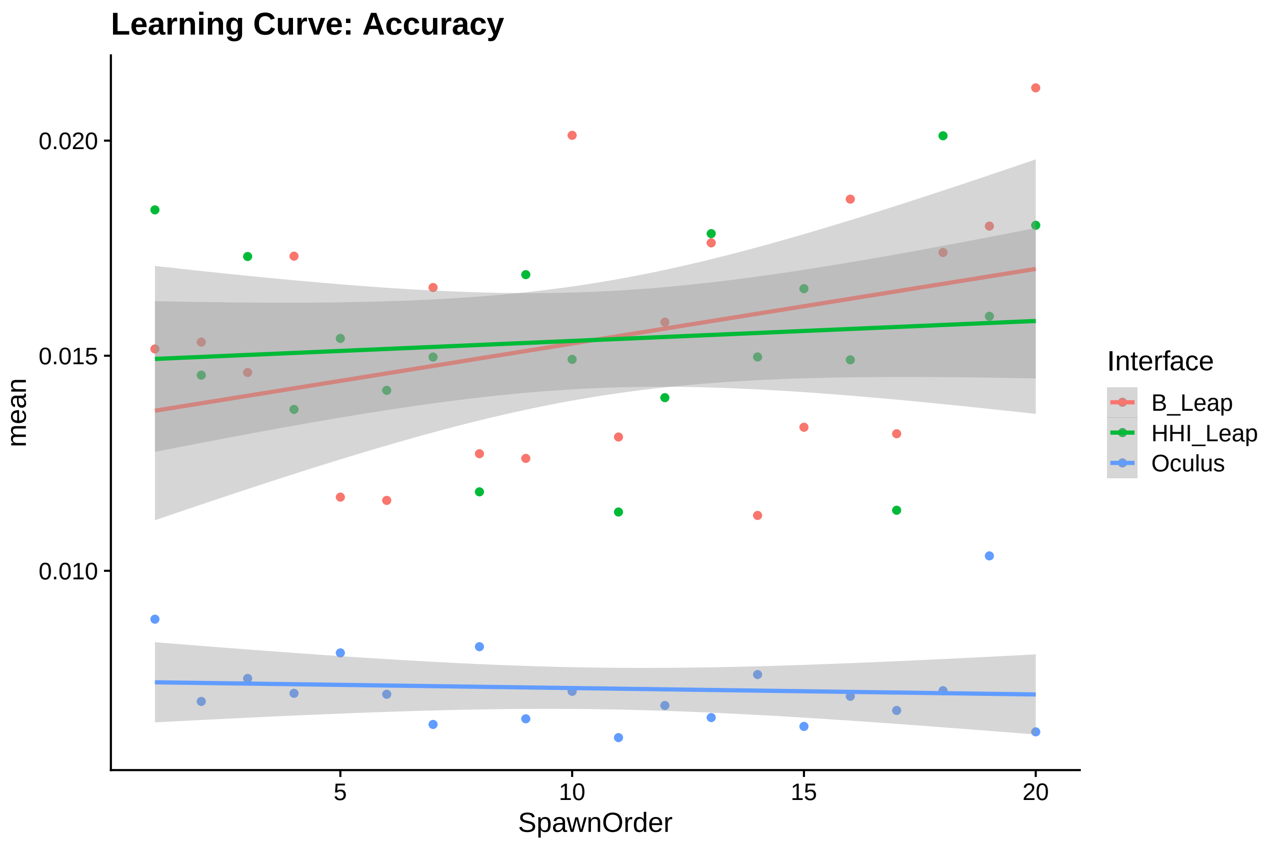
```r
# correlation test - do people get more accurate as they go?
corr_data <- unity_data_downsampled %>% select(SpawnOrder, Distance) %>% arrange(SpawnOrder)
cor.test(corr_data$SpawnOrder, corr_data$Distance)
```

```
##
##  Pearson's product-moment correlation
##
## data:  corr_data$SpawnOrder and corr_data$Distance
## t = 1.1046, df = 1918, p-value = 0.2695
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.01954346  0.06986970
## sample estimates:
##        cor
## 0.02521355
```

```r
# linear regression modeling
summary(lm(Distance ~ SpawnOrder*Interface, data=unity_data_downsampled))
```

```
##
## Call:
## lm(formula = Distance ~ SpawnOrder * Interface, data = unity_data_downsampled)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.016198 -0.007856 -0.003434  0.002344  0.145662
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     0.0135475  0.0012461  10.872  < 2e-16 ***
## SpawnOrder                      0.0001735  0.0001040   1.668  0.09546 .
## InterfaceHHI_Leap               0.0013309  0.0017622   0.755  0.45018
## InterfaceOculus                -0.0061254  0.0017622  -3.476  0.00052 ***
## SpawnOrder:InterfaceHHI_Leap -0.0001271  0.0001471  -0.864  0.38785
## SpawnOrder:InterfaceOculus   -0.0001884  0.0001471  -1.281  0.20051
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01517 on 1914 degrees of freedom
## Multiple R-squared:  0.06114,    Adjusted R-squared:  0.05868
## F-statistic: 24.93 on 5 and 1914 DF,  p-value: < 2.2e-16
```

```r
#summary(glm(Distance ~ Interface + SpawnOrder + Interface:SpawnOrder, data=unity_data_downsampled))

plot_data <- unity_data_downsampled %>%
  #filter(TimeFromSpawnToGrabLoss < 15) %>% # get rid of absurd times
  group_by(Interface, SpawnOrder) %>% summarise(mean=mean(Distance))

learning_curve_accuracy <- ggplot(plot_data, aes(SpawnOrder, mean, color=Interface)) +
  geom_point()+#geom_line()+
  geom_smooth(method="lm")+
  labs(title="Learning Curve: Accuracy")
learning_curve_accuracy
```




```r
sink("learning_curve_stat_output.csv")
cat("ANOVA: Learning Curve: Accuracy")
```

```
## ANOVA: Learning Curve: Accuracy
```

```r
write.csv(anova_summary(aov(Distance ~ Interface*SpawnOrder + Error(id/(Interface*SpawnOrder)), data=un
```

```
## "","Effect","DFn","DFd","F","p","p<.05","ges"
## "1","Interface",2,62,33.168,1.6e-10,"*",0.344
## "2","SpawnOrder",1,31,0.782,0.383,"",0.006
## "3","Interface:SpawnOrder",2,62,0.793,0.457,"",0.007
```

```r
sink()
```

Learning Curve: Total time

```r
# anova to see if there is an effect of SpawnOrder*Interface
#summary(aov(TimeFromSpawnToGrabLoss ~ Interface*SpawnOrder + Error(id/(Interface*SpawnOrder)), data=un

anova_summary(aov(TimeFromSpawnToGrabLoss ~ Interface*SpawnOrder + Error(id/(Interface*SpawnOrder)), da
```

```
##                 Effect DFn DFd      F        p p<.05   ges
## 1            Interface   2  62 32.072 2.74e-10     * 0.357
## 2           SpawnOrder   1  31  9.304 5.00e-03     * 0.022
## 3 Interface:SpawnOrder   2  62  2.286 1.10e-01       0.028
```

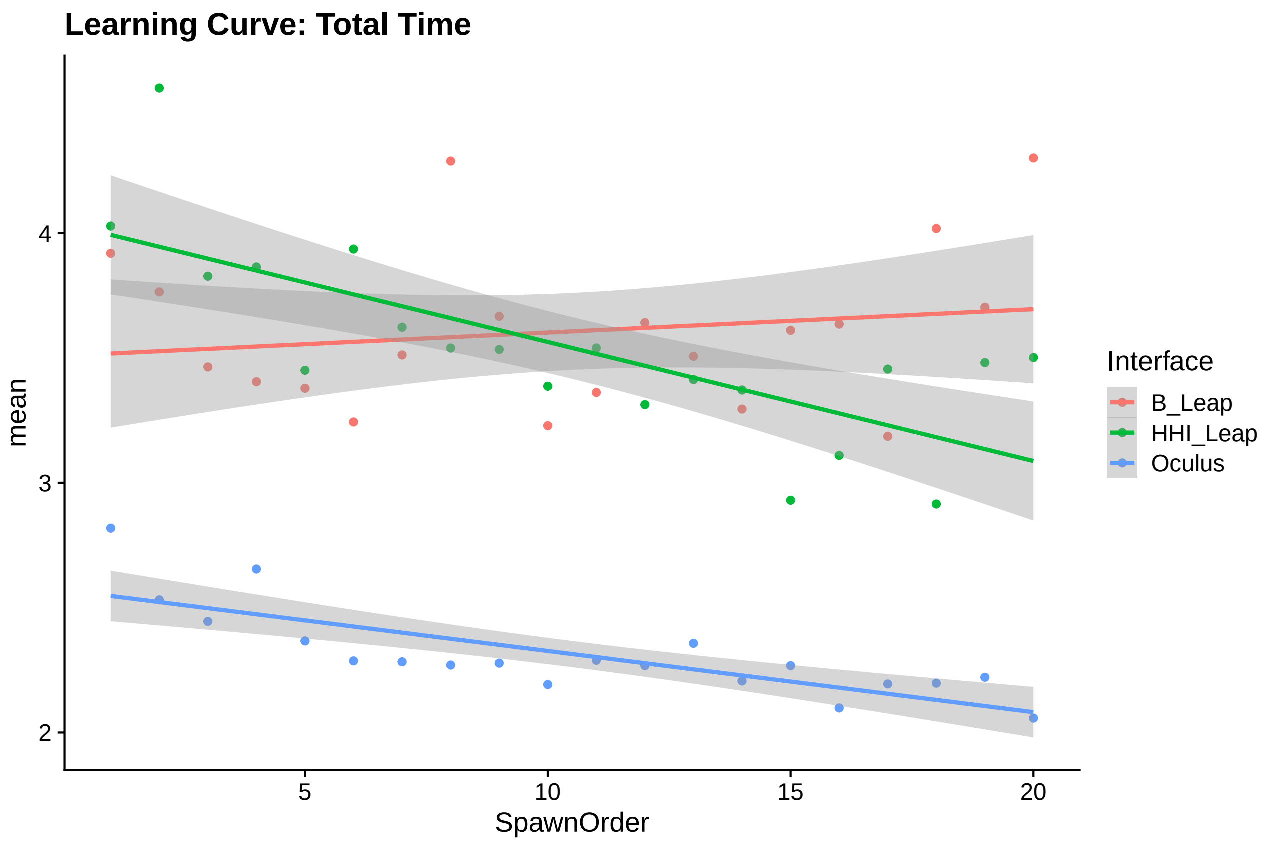
```r
# correlation test - do times get faster as they go?
corr_data <- unity_data_downsampled %>% select(SpawnOrder, TimeFromSpawnToGrabLoss) %>% arrange(SpawnOrd
cor.test(corr_data$SpawnOrder, corr_data$TimeFromSpawnToGrabLoss)
```

```
##
##  Pearson's product-moment correlation
##
## data:  corr_data$SpawnOrder and corr_data$TimeFromSpawnToGrabLoss
## t = -2.5868, df = 1918, p-value = 0.009761
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.10342518 -0.01426569
## sample estimates:
##         cor
## -0.05896302
```

```r
# linear regression modeling
summary(lm(TimeFromSpawnToGrabLoss ~ SpawnOrder*Interface, data=unity_data_downsampled))
```

```
##
## Call:
## lm(formula = TimeFromSpawnToGrabLoss ~ SpawnOrder * Interface,
##     data = unity_data_downsampled)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.6081 -0.9674 -0.4024  0.4825 31.0008
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  3.508013   0.160456  21.863  < 2e-16 ***
## SpawnOrder                   0.009347   0.013395   0.698  0.48537
## InterfaceHHI_Leap            0.532284   0.226920   2.346  0.01909 *
## InterfaceOculus             -0.936795   0.226920  -4.128 3.81e-05 ***
## SpawnOrder:InterfaceHHI_Leap -0.057013   0.018943  -3.010  0.00265 **
## SpawnOrder:InterfaceOculus   -0.033835   0.018943  -1.786  0.07423 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.954 on 1914 degrees of freedom
## Multiple R-squared:  0.09202,    Adjusted R-squared:  0.08964
## F-statistic: 38.79 on 5 and 1914 DF,  p-value: < 2.2e-16
```

```r
#summary(glm(Distance ~ Interface + SpawnOrder + Interface:SpawnOrder, data=unity_data_downsampled))

plot_data <- unity_data_downsampled %>%
  #filter(TimeFromSpawnToGrabLoss < 15) %>% # get rid of absurd times
  group_by(Interface, SpawnOrder) %>% summarise(mean=mean(TimeFromSpawnToGrabLoss))

learning_curve_total_time <- ggplot(plot_data, aes(SpawnOrder, mean, color=Interface)) +
  geom_point()+#geom_line()+
  geom_smooth(method="lm")+labs(title="Learning Curve: Total Time")
learning_curve_total_time
```




```r
sink("learning_curve_stat_output.csv", append = TRUE)
cat("\nANOVA: Learning Curve: Total Time")
```

```
##
## ANOVA: Learning Curve: Total Time
```

```r
write.csv(anova_summary(aov(TimeFromSpawnToGrabLoss ~ Interface*SpawnOrder + Error(id/(Interface*SpawnO
```

```
## "","Effect","DFn","DFd","F","p","p<.05","ges"
## "1","Interface",2,62,32.072,2.74e-10,"*",0.357
## "2","SpawnOrder",1,31,9.304,0.005,"*",0.022
## "3","Interface:SpawnOrder",2,62,2.286,0.11,"",0.028
```

```r
sink()

#------#------#

# # Bin method: bin into 5 bins --> find means of bins --> use as data points
# unity_data_downsampled <- unity_data_clean %>% group_by(id, Interface) %>% mutate(bin=ntile(SpawnOrde
#   # transform times into mean of bins
#   group_by(id, Interface, bin) %>% summarise_at(c("TimeFromSpawnToGrabLoss", "Distance"), mean) %>%
#       # can use the above for a more general data set using the binning method later
#   rename(SpawnOrder = bin) # rename for use with ANOVA
#
# # anova to see if there is an effect of SpawnOrder*Interface
```

```
# summary(aov(TimeFromSpawnToGrabLoss ~ Interface*SpawnOrder + Error(id/(Interface*SpawnOrder)), data=u
#
# anova_summary(aov(TimeFromSpawnToGrabLoss ~ Interface*SpawnOrder + Error(id/(Interface*SpawnOrder)),
#
# ggplot(unity_data_downsampled, aes(SpawnOrder, TimeFromSpawnToGrabLoss, color=Interface)) +
#   geom_point()+#geom_line()+
#   geom_smooth(method="lm")

# # make a data set of average total time per trial for each interface
# trial_data_means <- unity_data_downsampled %>%
#   #filter(TimeFromSpawnToGrabLoss < 15) %>% # get rid of absurd times
#   group_by(Interface, SpawnOrder) %>% summarise(Total_Time=mean(TimeFromSpawnToGrabLoss))
#
# ggplot(trial_data_means, aes(SpawnOrder, Total_Time, color=Interface)) +
#   geom_point()+#geom_line()+
#   geom_smooth(method="lm")
#
#
# # only small cubes
# small_cubes <- unity_data_clean %>% filter(Cube_Size=="Small") %>%
#   select(id, Interface, SpawnOrder, Cube_Size, TimeFromSpawnToGrabLoss)
#
# ggplot(small_cubes %>% filter(Interface=="B_Leap", TimeFromSpawnToGrabLoss<12, id%in%c(1,2,3)),
#        aes(SpawnOrder, TimeFromSpawnToGrabLoss, color=id))+
#   geom_point()+geom_line()
#
# # renumber spawnorder to start at 1
# small_cubes <- small_cubes %>% group_by(Interface, id) %>% mutate(Order=dense_rank(SpawnOrder))
#
#   # filter(SpawnOrder==max(SpawnOrder) | SpawnOrder==min(SpawnOrder)) %>%
#   # mutate(SpawnOrder=c(1,2))
#
# ggplot(small_cubes %>% filter(Interface=="B_Leap", TimeFromSpawnToGrabLoss < 13, as.integer(id)<10),
#   geom_point()+geom_line()
```

Learning Curve - grab time

```
# summary(aov(TimeFromSpawnToGrab ~ Interface*SpawnOrder + Error(id/(Interface*SpawnOrder)), data=unity

anova_summary(aov(TimeFromSpawnToGrab ~ Interface*SpawnOrder + Error(id/(Interface*SpawnOrder)), data=un
```

```
##                 Effect DFn DFd      F        p p<.05   ges
## 1            Interface   2  62 22.948 3.47e-08     * 0.224
## 2          SpawnOrder   1  31 10.428 3.00e-03     * 0.099
## 3 Interface:SpawnOrder   2  62  3.660 3.10e-02     * 0.032
```

```
# correlation test - do times get faster as they go?
corr_data <- unity_data_downsampled %>% select(SpawnOrder, TimeFromSpawnToGrab) %>% arrange(SpawnOrder)
cor.test(corr_data$SpawnOrder, corr_data$TimeFromSpawnToGrab)
```

```
##
```

```
##  Pearson's product-moment correlation
##
## data:  corr_data$SpawnOrder and corr_data$TimeFromSpawnToGrab
## t = -5.4833, df = 1918, p-value = 4.727e-08
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.16803524 -0.07994348
## sample estimates:
##        cor
## -0.1242341
```

```r
# hhi_leap
corr_data <- unity_data_downsampled %>% filter(Interface=="HHI_Leap") %>%
select(SpawnOrder, TimeFromSpawnToGrab) %>% arrange(SpawnOrder)
cor.test(corr_data$SpawnOrder, corr_data$TimeFromSpawnToGrab)
```

```
##
##  Pearson's product-moment correlation
##
## data:  corr_data$SpawnOrder and corr_data$TimeFromSpawnToGrab
## t = -4.1978, df = 638, p-value = 3.079e-05
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.23841372 -0.08755361
## sample estimates:
##        cor
## -0.1639421
```

```r
# b_leap
corr_data <- unity_data_downsampled %>% filter(Interface=="B_Leap") %>%
select(SpawnOrder, TimeFromSpawnToGrab) %>% arrange(SpawnOrder)
cor.test(corr_data$SpawnOrder, corr_data$TimeFromSpawnToGrab)
```

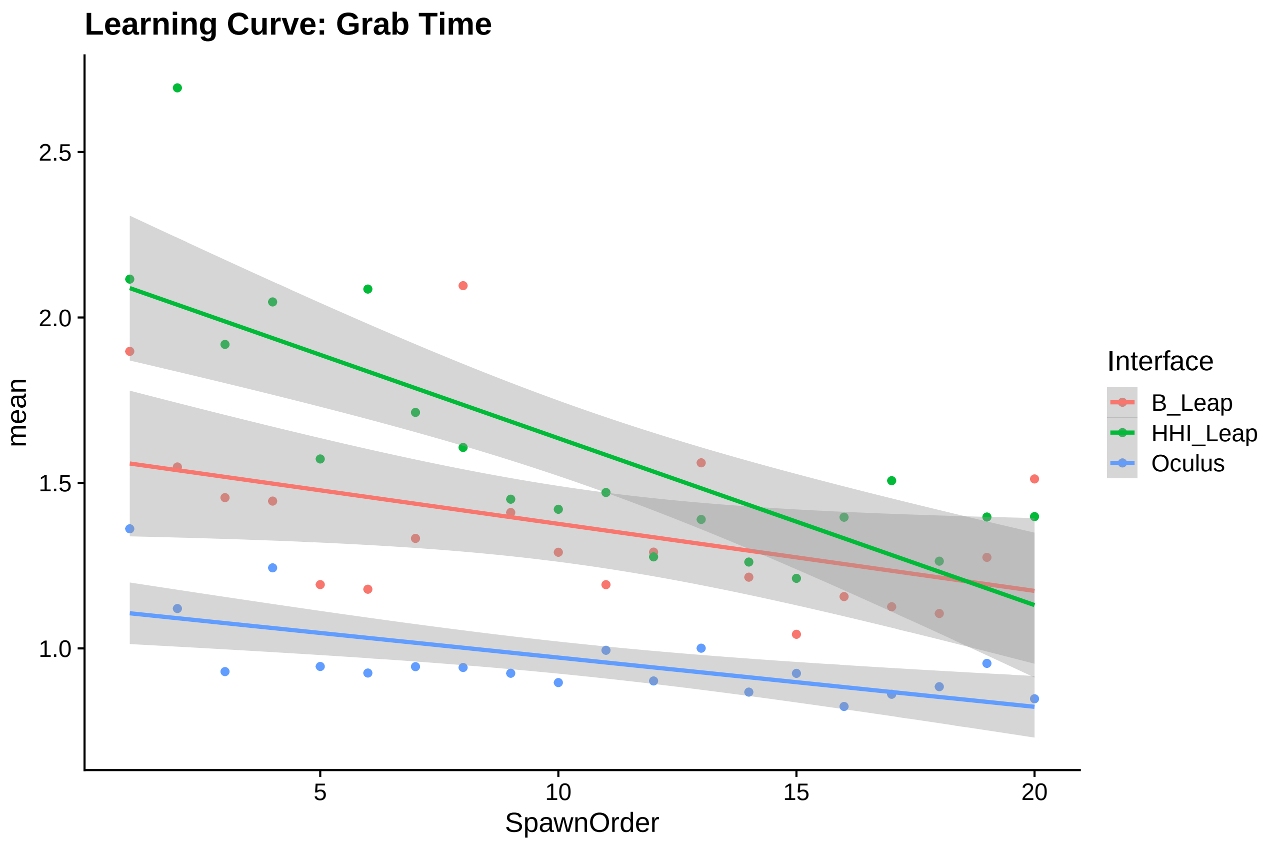
```
##
##  Pearson's product-moment correlation
##
## data:  corr_data$SpawnOrder and corr_data$TimeFromSpawnToGrab
## t = -2.2756, df = 638, p-value = 0.0232
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.16607593 -0.01231474
## sample estimates:
##         cor
## -0.08972997
```

```r
# linear regression modeling
summary(lm(TimeFromSpawnToGrab ~ SpawnOrder*Interface, data=unity_data_downsampled))
```

```
##
## Call:
## lm(formula = TimeFromSpawnToGrab ~ SpawnOrder * Interface, data = unity_data_downsampled)
##
```

```
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -1.8759 -0.4436 -0.1838  0.1108 30.1849 
## 
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)    
## (Intercept)                  1.579055   0.105517  14.965  < 2e-16 ***
## SpawnOrder                  -0.020260   0.008808  -2.300 0.021548 *  
## InterfaceHHI_Leap            0.560185   0.149223   3.754 0.000179 ***
## InterfaceOculus             -0.457966   0.149223  -3.069 0.002178 ** 
## SpawnOrder:InterfaceHHI_Leap -0.030153   0.012457  -2.421 0.015590 *  
## SpawnOrder:InterfaceOculus    0.005386   0.012457   0.432 0.665505    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 1.285 on 1914 degrees of freedom
## Multiple R-squared:  0.06045,    Adjusted R-squared:  0.05799 
## F-statistic: 24.63 on 5 and 1914 DF,  p-value: < 2.2e-16
```

```r
#summary(glm(Distance ~ Interface + SpawnOrder + Interface:SpawnOrder, data=unity_data_downsampled))

plot_data <- unity_data_downsampled %>%
  #filter(TimeFromSpawnToGrabLoss < 15) %>% # get rid of absurd times
  group_by(Interface, SpawnOrder) %>% summarise(mean=mean(TimeFromSpawnToGrab))

learning_curve_grab_time <- ggplot(plot_data, aes(SpawnOrder, mean, color=Interface)) +
  geom_point()+#geom_line()+
  geom_smooth(method="lm")+
  labs(title="Learning Curve: Grab Time")
learning_curve_grab_time
```




```r
sink("learning_curve_stat_output.csv", append = TRUE)
cat("\nANOVA: Learning Curve: Grab Time")
```

```
##
## ANOVA: Learning Curve: Grab Time
```

```r
write.csv(anova_summary(aov(TimeFromSpawnToGrab ~ Interface*SpawnOrder + Error(id/(Interface*SpawnOrder)
```

```
## "","Effect","DFn","DFd","F","p","p<.05","ges"
## "1","Interface",2,62,22.948,3.47e-08,"*",0.224
## "2","SpawnOrder",1,31,10.428,0.003,"*",0.099
## "3","Interface:SpawnOrder",2,62,3.66,0.031,"*",0.032
```

```r
sink()
```

Learning Curve: Release Time

```r
# summary(aov(TimeFromGrabToGrabLoss ~ Interface*SpawnOrder + Error(id/(Interface*SpawnOrder)), data=un

anova_summary(aov(TimeFromGrabToGrabLoss ~ Interface*SpawnOrder + Error(id/(Interface*SpawnOrder)), dat
```

```
##                 Effect DFn DFd      F        p p<.05   ges
## 1            Interface   2  62 27.993 2.17e-09     * 0.284
## 2           SpawnOrder   1  31  0.890 3.53e-01       0.006
## 3 Interface:SpawnOrder   2  62  2.187 1.21e-01       0.025
```

```r
# correlation test - do times get faster as they go?
corr_data <- unity_data_downsampled %>% select(SpawnOrder, TimeFromGrabToGrabLoss) %>% arrange(SpawnOrd
cor.test(corr_data$SpawnOrder, corr_data$TimeFromGrabToGrabLoss)
```

```
##
##  Pearson's product-moment correlation
##
## data:  corr_data$SpawnOrder and corr_data$TimeFromGrabToGrabLoss
## t = 1.4272, df = 1918, p-value = 0.1537
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.01218261  0.07719260
## sample estimates:
##        cor
## 0.03257011
```

```r
  # how about just for the B_Leap?
  corr_data <- unity_data_downsampled %>% filter(Interface=="B_Leap") %>%
    select(SpawnOrder, TimeFromGrabToGrabLoss) %>% arrange(SpawnOrder)
  cor.test(corr_data$SpawnOrder, corr_data$TimeFromGrabToGrabLoss)
```

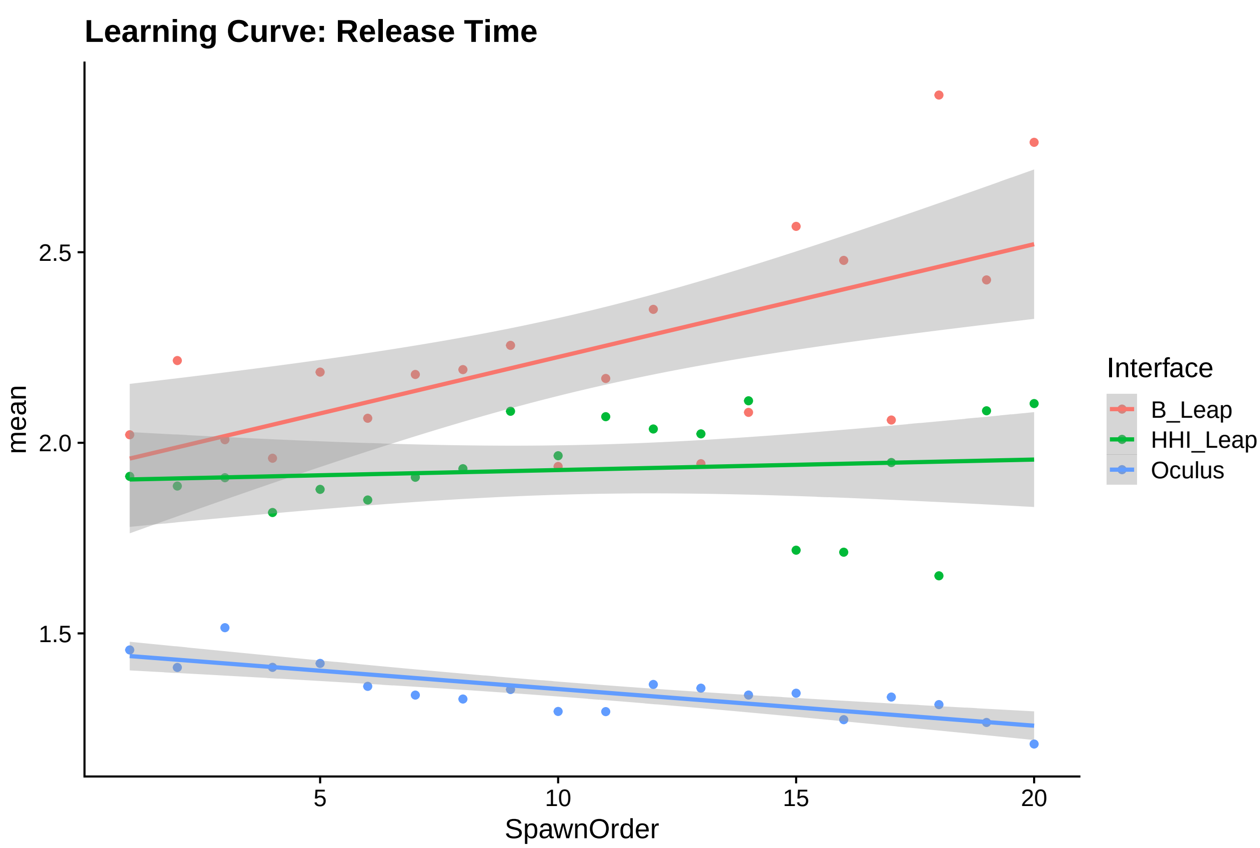
```
##
##  Pearson's product-moment correlation
##
## data:  corr_data$SpawnOrder and corr_data$TimeFromGrabToGrabLoss
## t = 2.3235, df = 638, p-value = 0.02047
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.01420159 0.16791048
## sample estimates:
##        cor
## 0.09160164
```

```r
# linear regression modeling
summary(lm(TimeFromGrabToGrabLoss ~ SpawnOrder*Interface, data=unity_data_downsampled))
```

```
##
## Call:
## lm(formula = TimeFromGrabToGrabLoss ~ SpawnOrder * Interface,
##     data = unity_data_downsampled)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.0398 -0.6491 -0.2597  0.3313 21.8142
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     1.92896    0.10578  18.236  < 2e-16 ***
## SpawnOrder                      0.02961    0.00883   3.353 0.000815 ***
## InterfaceHHI_Leap              -0.02790    0.14959  -0.187 0.852061
## InterfaceOculus                -0.47883    0.14959  -3.201 0.001392 **
## SpawnOrder:InterfaceHHI_Leap -0.02686    0.01249  -2.151 0.031602 *
```

```
## SpawnOrder:InterfaceOculus   -0.03922    0.01249  -3.141 0.001711 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.288 on 1914 degrees of freedom
## Multiple R-squared:  0.08168,    Adjusted R-squared:  0.07928
## F-statistic: 34.05 on 5 and 1914 DF,  p-value: < 2.2e-16
```

```
#summary(glm(Distance ~ Interface + SpawnOrder + Interface:SpawnOrder, data=unity_data_downsampled))

plot_data <- unity_data_downsampled %>%
  #filter(TimeFromSpawnToGrabLoss < 15) %>% # get rid of absurd times
  group_by(Interface, SpawnOrder) %>% summarise(mean=mean(TimeFromGrabToGrabLoss))

learning_curve_release_time <- ggplot(plot_data, aes(SpawnOrder, mean, color=Interface)) +
  geom_point()+#geom_line()+
  geom_smooth(method="lm")+
  labs(title="Learning Curve: Release Time")
learning_curve_release_time
```




```
sink("learning_curve_stat_output.csv", append = TRUE)
cat("\nANOVA: Learning Curve: Release Time")
```

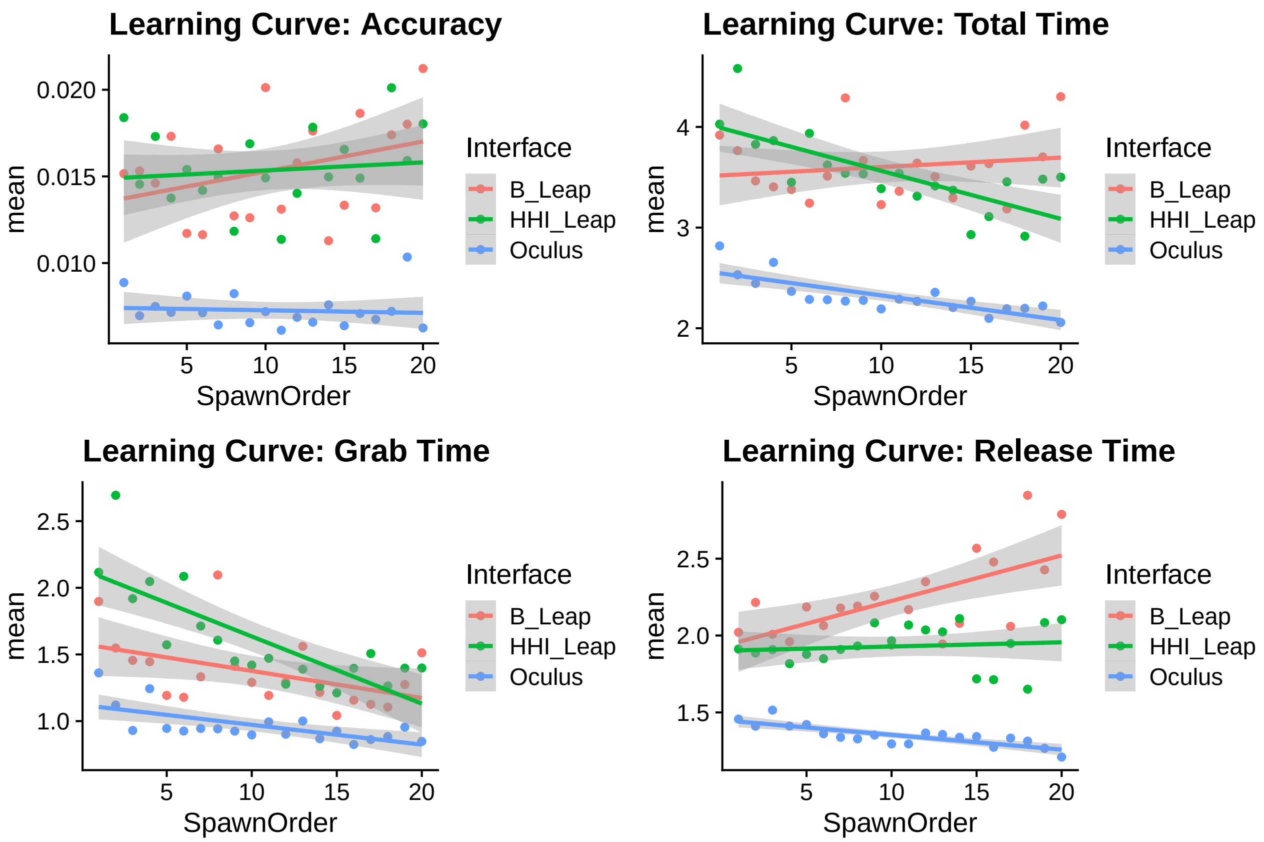
```
##
## ANOVA: Learning Curve: Release Time
```

```r
write.csv(anova_summary(aov(TimeFromGrabToGrabLoss ~ Interface*SpawnOrder + Error(id/(Interface*SpawnOr
```

```
## "","Effect","DFn","DFd","F","p","p<.05","ges"
## "1","Interface",2,62,27.993,2.17e-09,"*",0.284
## "2","SpawnOrder",1,31,0.89,0.353,"",0.006
## "3","Interface:SpawnOrder",2,62,2.187,0.121,"",0.025
```

```r
sink()
```

```r
plot_grid(learning_curve_accuracy,# + theme(legend.position = "none"),
          learning_curve_total_time,# + theme(legend.position = "none"),
          learning_curve_grab_time,# + theme(legend.position = "none"),
          learning_curve_release_time)# + theme(legend.position = "none"))
```




```r
ggsave("learning_curve_compilation.jpg", width = 12, height=9)
```

Interface order

This section looks for signs of systematic effects due to the order that subjects used the interfaces in this study. Some information about the variables: - Interface order contains all subjects. It is one per subject. For the ANOVA, the Interface factor is within-subjects but the Interface Order, Leap_Group and Oculus_Group factors are not. - The Interface x InterfaceOrder ANOVA contains all three interfaces in the Interface factor; the other two ANOVAs use only the Leaps - Leap_Group contains all Interface orders: 3 for Leap first, 3 for HHI first. - Oculus_Group contains only 4 of 6 Interface order: 2 for when Oculus came first, 2 for when

Oculus came last. The logic is that this allows comparison for outcome measures for both Leaps when the Oculus had already been seen tried out and before it had been tried. If Oculus had an effect on ratings for either or both Leap interfaces, it would happen only after the subject had been exposed to the Oculus. Lower ratings for Leaps when Oculus came first would indicate this sort of effect.

I.O. Performance

Accuracy - I.O.

```
# Accuracy
# ANOVAs
Interface.order.anova <-
  anova_summary(effect.size="pes",aov(Distance ~ Interface*InterfaceOrder + Error(id/Interface), data=su
Interface.order.anova
```

```
##                      Effect DFn DFd      F        p p<.05   pes
## 1            InterfaceOrder   5  26  0.314 9.00e-01       0.057
## 2                 Interface   2  52 40.106 2.90e-11     * 0.607
## 3 Interface:InterfaceOrder  10  52  0.761 6.64e-01       0.128
```

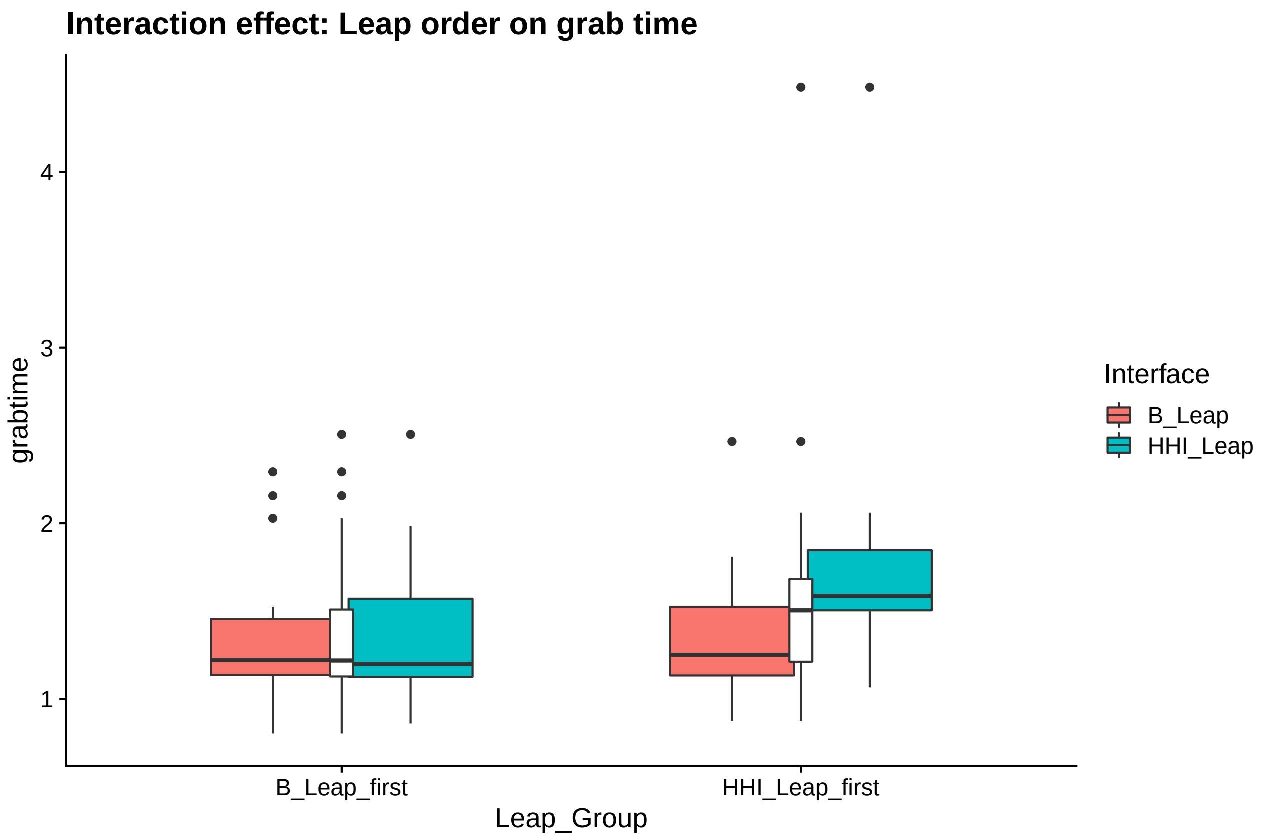
```
  #Leap group
  Interface.order.anova2 <-
    anova_summary(effect.size="pes",aov(Distance ~ Interface*Leap_Group + Error(id/Interface), data=sub

  #Oculus group -- Interface orders, filtered
  oculus.group.anova <-
  anova_summary(effect.size="pes",aov(Distance ~ Interface*Oculus_Group + Error(id/Interface),
      data=subject_data_all_long%>%ungroup%>%
        filter(is.na(Oculus_Group)==FALSE, Interface!="Oculus") %>% mutate(Interface=factor(Interface))

Interface_order_output<- Interface.order.anova %>% rbind(Interface.order.anova2) %>% rbind(oculus.group

  # t-tests
  stat.test <- subject_data_all_long %>% ungroup(.) %>% filter(Interface=="B_Leap" | Interface=="HHI_Lea
    group_by(Leap_Group) %>%
    t_test(Distance ~ Interface, paired=TRUE) %>% # paired b/c it's within Leap group
    mutate(test="Within-subjects T-test") %>% rename(Group=Leap_Group)

  # plot and test, split by Interface (leap vs. leap, HHI vs. HHI)
  stat.test2 <- subject_data_all_long %>% ungroup(.) %>% filter(Interface=="B_Leap" | Interface=="HHI_Le
    t_test(Distance ~ Leap_Group, paired=FALSE) %>% # NOT paired b/c it's between Leap groups
    mutate(test="Between-subjects T-test")%>% rename(Group=Interface)
  stat.test <-rbind(stat.test, stat.test2)

  # Interfaces when first
  stat.test3 <- subject_data_all_long %>% ungroup(.) %>% filter((Leap_Group=="B_Leap_first" & Interface=
    t_test(Distance ~ Interface, paired=FALSE) %>% # NOT paired b/c it's between Leap groups
    mutate(test="Between-subjects T-test", Group="when first")
  stat.test <- stat.test %>% bind_rows(stat.test3)

  #Interfaces when second
  stat.test4 <- subject_data_all_long %>% ungroup(.) %>% filter((Leap_Group=="B_Leap_first" & Interface=
```

```r
    t_test(Distance ~ Interface, paired=FALSE) %>% # NOT paired b/c it's between Leap groups
    mutate(test="Between-subjects T-test", Group="when second")
  stat.test <- stat.test %>% bind_rows(stat.test4)

  # adjust p value
  stat.test <- stat.test %>% adjust_pvalue() %>% add_significance("p.adj") %>%
    mutate(Interface=group1) #to make the stat.pvalue.manual ggplot item happy
  stat.test
```

```
## # A tibble: 6 x 13
##   Group .y.   group1 group2    n1    n2 statistic    df     p test  p.adj
##   <chr> <chr> <chr>  <chr>  <int> <int>     <dbl> <dbl> <dbl> <chr> <dbl>
## 1 B_Le~ Dist~ B_Leap HHI_L~    17    17     1.27   16   0.223 With~ 1
## 2 HHI_~ Dist~ B_Leap HHI_L~    15    15    -1.55   14   0.144 With~ 0.864
## 3 B_Le~ Dist~ B_Lea~ HHI_L~    17    15     0.537  28.0 0.595 Betw~ 1
## 4 HHI_~ Dist~ B_Lea~ HHI_L~    17    15    -1.15   29.3 0.261 Betw~ 1
## 5 when~ Dist~ B_Leap HHI_L~    17    15    -0.657  22.4 0.518 Betw~ 1
## 6 when~ Dist~ B_Leap HHI_L~    15    17     0.278  28.8 0.783 Betw~ 1
## # ... with 2 more variables: p.adj.signif <chr>, Interface <chr>
```

Grab time - I.O.

```r
#grabtime
Interface.order.anova <-
  anova_summary(effect.size="pes",aov(grabtime ~ Interface*InterfaceOrder + Error(id/Interface), data=su

  #Leap group
  Interface.order.anova2 <-
    anova_summary(effect.size="pes",aov(grabtime ~ Interface*Leap_Group + Error(id/Interface), data=sub
      ggplot(subject_data_all_long %>% filter(Interface!="Oculus"), aes(Leap_Group, grabtime))+
        labs(title="Grab time by Leap order and interface")+
        geom_boxplot(aes(fill=Interface), width=.6)+labs(title="Interaction effect: Leap order on grab
        geom_boxplot(width=.05, fill="white", position=position_nudge(0))
```




```
  #Oculus group
  oculus.group.anova <-
  anova_summary(effect.size="pes",aov(Distance ~ Interface*Oculus_Group + Error(id/Interface),
      data=subject_data_all_long%>%ungroup%>%
        filter(is.na(Oculus_Group)==FALSE, Interface!="Oculus") %>% mutate(Interface=factor(Interface))

Interface_order_output <- Interface_order_output %>% rbind(Interface.order.anova %>% rbind(Interface.or

    # t-tests - grab time d.o. - leap group
    stat.test <- subject_data_all_long %>% ungroup(.) %>% filter(Interface=="B_Leap" | Interface=="HHI_
      group_by(Leap_Group) %>%
      t_test(grabtime ~ Interface, paired=TRUE) %>% # paired b/c it's within Leap group
      mutate(test="Within-subjects T-test") %>% rename(Group=Leap_Group)

    # plot and test, split by Interface (leap vs. leap, HHI vs. HHI)
    stat.test2 <- subject_data_all_long %>% ungroup(.) %>% filter(Interface=="B_Leap" | Interface=="HHI
      t_test(grabtime ~ Leap_Group, paired=FALSE) %>% # NOT paired b/c it's between Leap groups
      mutate(test="Between-subjects T-test")%>% rename(Group=Interface)
    stat.test <-rbind(stat.test, stat.test2)

    # Interfaces when first
    stat.test3 <- subject_data_all_long %>% ungroup(.) %>% filter((Leap_Group=="B_Leap_first" & Interfa
      t_test(grabtime ~ Interface, paired=FALSE) %>% # NOT paired b/c it's between Leap groups
      mutate(test="Between-subjects T-test", Group="when first")
    stat.test <- stat.test %>% bind_rows(stat.test3)

    #Interfaces when second
```

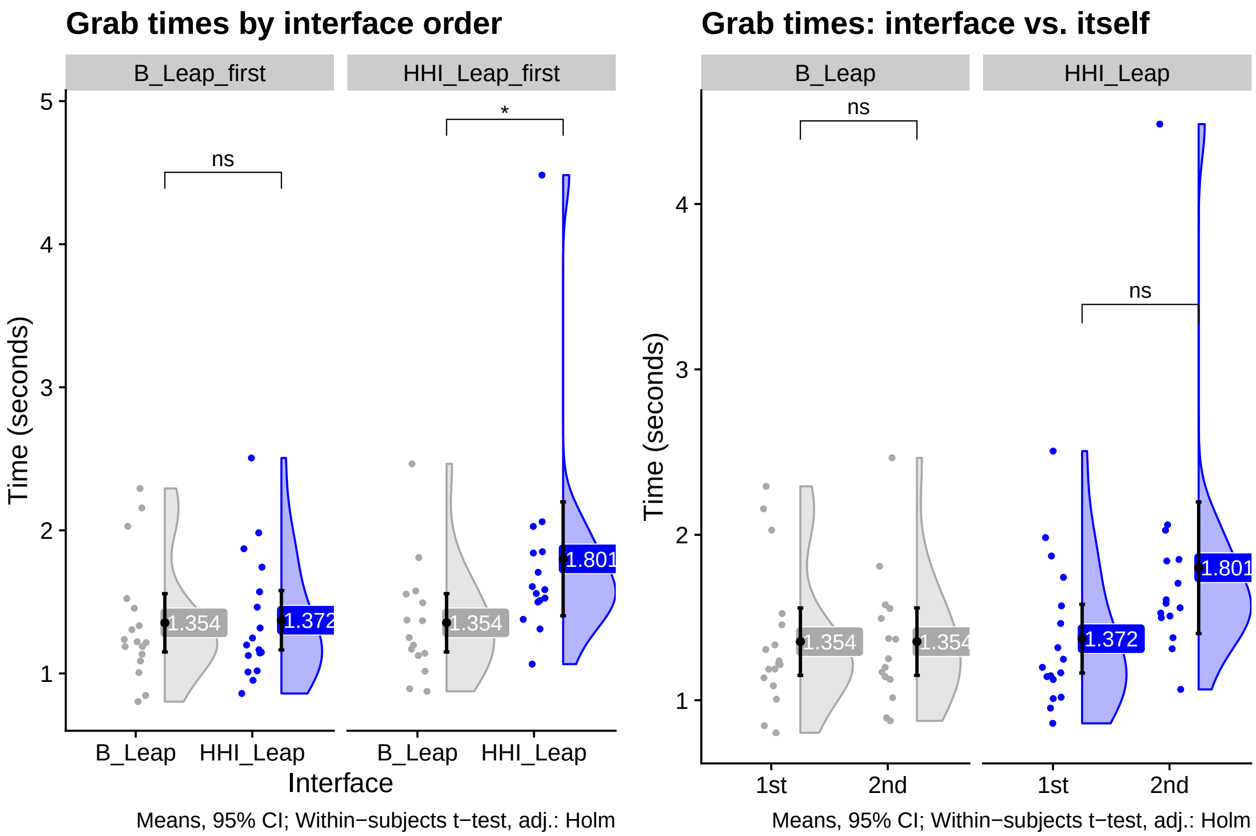
```r
stat.test4 <- subject_data_all_long %>% ungroup(.) %>% filter((Leap_Group=="B_Leap_first" & Interfa
  t_test(grabtime ~ Interface, paired=FALSE) %>% # NOT paired b/c it's between Leap groups
  mutate(test="Between-subjects T-test", Group="when second")
stat.test <- stat.test %>% bind_rows(stat.test4)

# adjust p value
stat.test <- stat.test %>% adjust_pvalue() %>% add_significance("p.adj") %>%
  mutate(Interface=group1) #to make the stat.pvalue.manual ggplot item happy
stat.test
```

```
## # A tibble: 6 x 13
##   Group .y.   group1 group2    n1    n2 statistic    df       p test   p.adj
##   <chr> <chr> <chr>  <chr>  <int> <int>     <dbl> <dbl>   <dbl> <chr>  <dbl>
## 1 B_Le~ grab~ B_Leap HHI_L~    17    17  -0.140    16   0.891   With~ 1
## 2 HHI_~ grab~ B_Leap HHI_L~    15    15  -3.40     14   0.00434 With~ 0.0260
## 3 B_Le~ grab~ B_Lea~ HHI_L~    17    15   0.00252  29.9 0.998   Betw~ 1
## 4 HHI_~ grab~ B_Lea~ HHI_L~    17    15  -1.87     21.2 0.0752  Betw~ 0.32
## 5 when~ grab~ B_Leap HHI_L~    17    15  -1.96     21.0 0.064   Betw~ 0.32
## 6 when~ grab~ B_Leap HHI_L~    15    17  -0.120    29.9 0.905   Betw~ 1
## # ... with 2 more variables: p.adj.signif <chr>, Interface <chr>
```

```r
    # make labels for plots
Leap_Group_labs<-c(paste0("HHI first n=", length((subject_data_all_long %>%
    select(id, Interface, grabtime,Leap_Group)%>%filter(Leap_Group=="HHI_Leap_first")%>%select(id)%

# by leap group (B_Leap_first, HHI_Leap_first)
temp_set <- subject_data_all_long %>% filter(Interface=="B_Leap" | Interface=="HHI_Leap") %>%
  select(id, Interface, grabtime, Leap_Group)
temp_plot_data <- subject_data_all_long %>% group_by(Interface, Leap_Group) %>%
  filter(Interface=="B_Leap" | Interface=="HHI_Leap") %>% get_summary_stats(grabtime)

p1<- ggplot(temp_set, aes(x=Interface, y=grabtime, fill=Interface, colour = Interface))+
geom_flat_violin(position = position_nudge(x = .25, y = 0), alpha=myalpha,adjust=mysmoothing)+
geom_point(position = position_jitter(width = 0.1, height=0), size = 1)+ # the "rain"
geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=round(mean, 3)), position = po
geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(.25), col
geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(s
ylab('Time (seconds)')+xlab('Interface')+theme_cowplot()+guides(fill = FALSE, colour = FALSE) +
scale_color_manual(values=mycolors)+#scale_colour_brewer(palette = "Set2")+
scale_fill_manual(values=mycolors)+#scale_fill_brewer(palette = "Set2", direction=1)+
  stat_pvalue_manual(data=stat.test%>%slice(1:2)%>%rename(Leap_Group=Group), xmin="group1", xmax="g
labs(title="Grab times by interface order", caption="Means, 95% CI; Within-subjects t-test, adj.: H
facet_grid(. ~ Leap_Group)
#ggsave(last_plot(), filename="trainingtime_leapgroup_raincloud.jpg", width=9, height=5)


# plot, split by Interface (leap vs. leap, HHI vs. HHI)
temp_set <- subject_data_all_long %>% filter(Interface=="B_Leap" | Interface=="HHI_Leap") %>%
  select(id, Interface, grabtime, Leap_Group)

temp_plot_data <- subject_data_all_long %>% group_by(Interface, Leap_Group) %>%
  filter(Interface=="B_Leap" | Interface=="HHI_Leap") %>% get_summary_stats(grabtime)
```

```
p2<- ggplot(temp_set, aes(x=Leap_Group, y=grabtime, fill=Interface, colour = Interface))+
geom_flat_violin(position = position_nudge(x = .25, y = 0), alpha=myalpha, adjust=mysmoothing)+
geom_point(position = position_jitter(width = 0.1, height=0), size = 1)+ # the "rain"
geom_label(data = temp_plot_data, aes(x = Leap_Group, y = mean, label=round(mean, 3)), position = po
geom_point(data = temp_plot_data, aes(x = Leap_Group, y = mean), position = position_nudge(.25), col
geom_errorbar(data = temp_plot_data, aes(x = Leap_Group, y = mean, ymin=mean-(se*1.96), ymax=mean+(s
ylab('Time (seconds)')+theme_cowplot()+guides(fill = FALSE, colour = FALSE) +
scale_color_manual(values=mycolors)+#scale_colour_brewer(palette = "Set2")+
scale_fill_manual(values=mycolors)+#scale_fill_brewer(palette = "Set2", direction=1)+
  xlab(NULL)+
stat_pvalue_manual(data=stat.test%>%slice(3:4)%>%mutate(Interface=Group), xmin="group1", xmax="grou
labs(title="Grab times: interface vs. itself", caption="Means, 95% CI; Within-subjects t-test, adj.
scale_x_discrete(labels=c("1st","2nd"))+
facet_grid(. ~ Interface)#+scale_x_discrete(labels=c("HHI-->Leap","Leap-->HHI"))
#ggsave(last_plot(), filename="trainingtime_leapgroup_raincloud.jpg", width=9, height=5)


# each Interface when first
temp_set <- subject_data_all_long %>% ungroup(.) %>% filter((Leap_Group=="B_Leap_first" & Interface=
temp_plot_data <- temp_set %>% group_by(Interface)%>% get_summary_stats(grabtime)

p3<- ggplot(temp_set, aes(x=Interface, y=grabtime, fill=Interface, colour = Interface))+
  geom_flat_violin(position = position_nudge(x = .25, y = 0), alpha=.7)+#,adjust =2)+
  geom_point(position = position_jitter(width = 0.1, height=0), size = 1)+ # the "rain"
geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=round(mean, 3)), position = pos
  geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(.25), co
  geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+
  ylab('Time (seconds)')+xlab(NULL)+theme_cowplot()+guides(fill = FALSE, colour = FALSE) +
  scale_colour_brewer(palette = "Set2")+#coord_flip()+
  scale_fill_brewer(palette = "Set2")+
  # stat_compare_means(method="t.test", paired=FALSE, label.x.npc="center")+
  # stat_compare_means(method="wilcox", paired=FALSE, label.x.npc="right")+
  stat_pvalue_manual(data=stat.test%>%slice(5)%>%mutate(Interface="B_Leap"), xmin="group1", xmax="gr
  labs(title="Grab times when 1st", caption="Means, 95% CI; Within-subjects t-test, adj.: Holm")#+fa
#ggsave(last_plot(), filename="Grabtime_both_first.jpg", width=6, height=4)


# Grab times when second (plot) - BETWEEN SUBJECTS
temp_set <- subject_data_all_long %>% ungroup(.) %>% filter((Leap_Group=="B_Leap_first" & Interface=
temp_plot_data <- temp_set %>% group_by(Interface)%>% get_summary_stats(grabtime)

p4<- ggplot(temp_set, aes(x=Interface, y=grabtime, fill=Interface, colour = Interface))+
  geom_flat_violin(position = position_nudge(x = .25, y = 0), alpha=.7)+#,adjust =2)+
  geom_point(position = position_jitter(width = 0.1, height=0), size = 1)+ # the "rain"
geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=round(mean, 3)), position = pos
  geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(.25), co
  geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+
  ylab('Time (seconds)')+xlab('Interface')+theme_cowplot()+guides(fill = FALSE, colour = FALSE) +
  scale_colour_brewer(palette = "Set2")+#coord_flip()+
  scale_fill_brewer(palette = "Set2")+
  # stat_compare_means(method="t.test", paired=FALSE, label.x.npc="center")+
  # stat_compare_means(method="wilcox", paired=FALSE, label.x.npc="right")+
  stat_pvalue_manual(data=stat.test%>%slice(5)%>%mutate(Interface="B_Leap"), xmin="group1", xmax="gr
```

```
  labs(title="Grab times when 2nd", subtitle = , caption="Means, 95% CI; Within-subjects t-test, ad
ggsave(last_plot(), filename="Grabtime_both_first.jpg", width=6, height=4)

plot_grid(p1, p2)#, p3, p4)
```




```
ggsave("grabtimes_closer_look.jpg", width=12, height=10)

do_grabtime<-p1
```

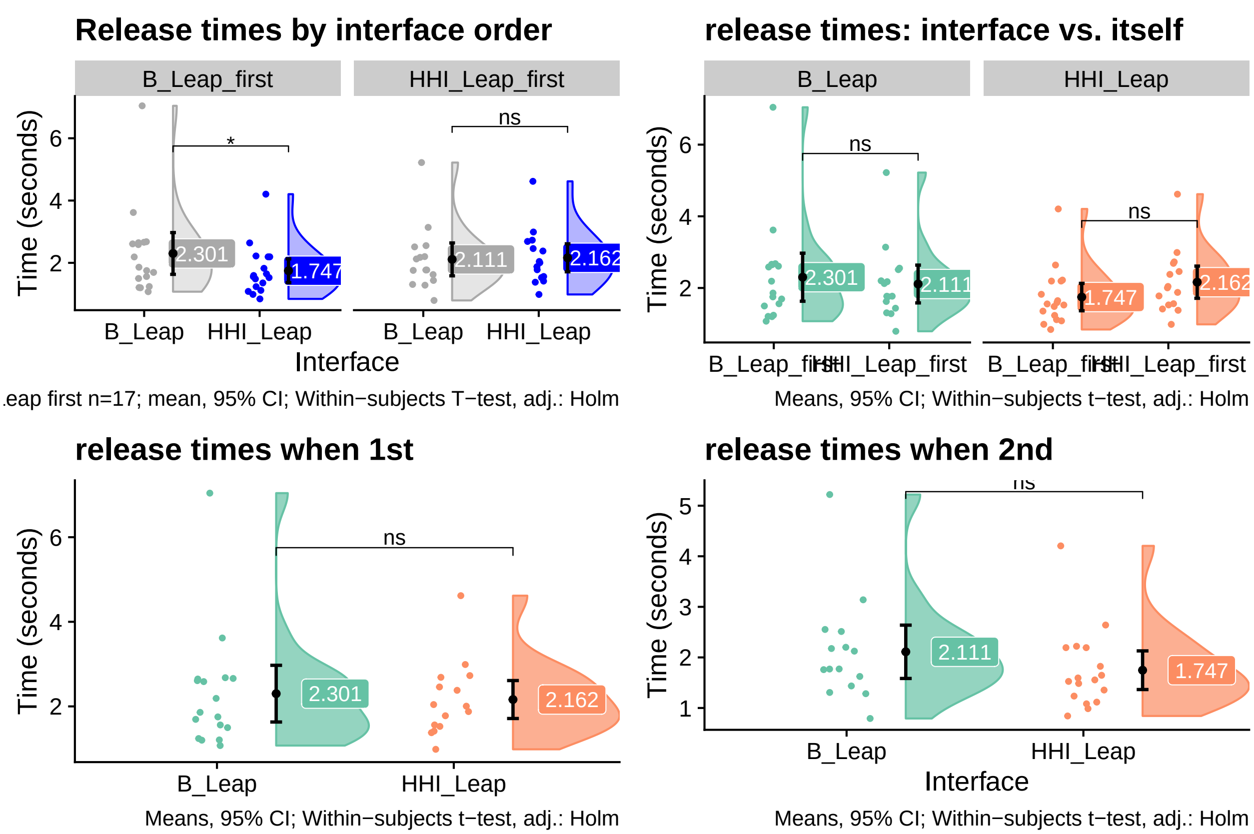
Release time

```
# release time
  Interface.order.anova <-
    anova_summary(effect.size="pes",aov(releasetime ~ Interface*InterfaceOrder + Error(id/Interface), da
  #Interface.order.anova

  #Leap group
  Interface.order.anova2 <-
    anova_summary(effect.size="pes",aov(releasetime ~ Interface*Leap_Group + Error(id/Interface), data=
  #Interface.order.anova2

  #Oculus group
  oculus.group.anova <-
  anova_summary(effect.size="pes",aov(releasetime ~ Interface*Oculus_Group + Error(id/Interface),
      data=subject_data_all_long%>%ungroup%>%
```

```
        filter(is.na(Oculus_Group)==FALSE, Interface!="Oculus") %>% mutate(Interface=factor(Interface))

Interface_order_output <- Interface_order_output %>% rbind(Interface.order.anova %>% rbind(Interface.or

      # t-tests - release time d.o. - leap group
    stat.test <- subject_data_all_long %>% ungroup(.) %>% filter(Interface=="B_Leap" | Interface=="HHI_
      group_by(Leap_Group) %>%
      t_test(releasetime ~ Interface, paired=TRUE) %>% # paired b/c it's within Leap group
      mutate(test="Within-subjects T-test") %>% rename(Group=Leap_Group)

    # plot and test, split by Interface (leap vs. leap, HHI vs. HHI)
    stat.test2 <- subject_data_all_long %>% ungroup(.) %>% filter(Interface=="B_Leap" | Interface=="HHI_
      t_test(releasetime ~ Leap_Group, paired=FALSE) %>% # NOT paired b/c it's between Leap groups
      mutate(test="Between-subjects T-test")%>% rename(Group=Interface)
    stat.test <-rbind(stat.test, stat.test2)

    # Interfaces when first
    stat.test3 <- subject_data_all_long %>% ungroup(.) %>% filter((Leap_Group=="B_Leap_first" & Interfa
      t_test(releasetime ~ Interface, paired=FALSE) %>% # NOT paired b/c it's between Leap groups
      mutate(test="Between-subjects T-test", Group="when first")
    stat.test <- stat.test %>% bind_rows(stat.test3)

    #Interfaces when second
    stat.test4 <- subject_data_all_long %>% ungroup(.) %>% filter((Leap_Group=="B_Leap_first" & Interfa
      t_test(releasetime ~ Interface, paired=FALSE) %>% # NOT paired b/c it's between Leap groups
      mutate(test="Between-subjects T-test", Group="when second")
    stat.test <- stat.test %>% bind_rows(stat.test4)

    # adjust p value
    stat.test <- stat.test %>% adjust_pvalue() %>% add_significance(p.col="p.adj", output.col="p.adj.si


    #stat.test<- stat.test %>% mutate(Interface=group1, p.adj.signif=p.signif)




        # make labels for plots
    Leap_Group_labs<-c(paste0("HHI first n=", length((subject_data_all_long %>%
        select(id, Interface, releasetime,Leap_Group)%>%filter(Leap_Group=="HHI_Leap_first")%>%select(i

    # by leap group (B_Leap_first, HHI_Leap_first)
    temp_set <- subject_data_all_long %>% filter(Interface=="B_Leap" | Interface=="HHI_Leap") %>%
      select(id, Interface, releasetime, Leap_Group)
    temp_plot_data <- subject_data_all_long %>% group_by(Interface, Leap_Group) %>%
      filter(Interface=="B_Leap" | Interface=="HHI_Leap") %>% get_summary_stats(releasetime)

    p1<- ggplot(temp_set, aes(x=Interface, y=releasetime, fill=Interface, colour = Interface))+
    geom_flat_violin(position = position_nudge(x = .25, y = 0), alpha=myalpha, adjust=mysmoothing)+
    geom_point(position = position_jitter(width = 0.1, height=0), size = 1)+ # the "rain"
    geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=round(mean, 3)), position = po
    geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(.25), col
    geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(s
    ylab('Time (seconds)')+xlab('Interface')+theme_cowplot()+guides(fill = FALSE, colour = FALSE) +
```

```r
scale_color_manual(values=mycolors)+#scale_colour_brewer(palette = "Set2")+
scale_fill_manual(values=mycolors)+#scale_fill_brewer(palette = "Set2", direction=1)+
stat_pvalue_manual(data=stat.test%>%slice(1:2)%>%rename(Leap_Group=Group), xmin="group1", xmax="grou
labs(title="Release times by interface order", caption=paste0(Leap_Group_labs[1],", ", Leap_Group_la
facet_grid(. ~ Leap_Group)
#ggsave(last_plot(), filename="trainingtime_leapgroup_raincloud.jpg", width=9, height=5)


# plot, split by Interface (leap vs. leap, HHI vs. HHI)
temp_set <- subject_data_all_long %>% filter(Interface=="B_Leap" | Interface=="HHI_Leap") %>%
  select(id, Interface, releasetime, Leap_Group)

temp_plot_data <- subject_data_all_long %>% group_by(Interface, Leap_Group) %>%
  filter(Interface=="B_Leap" | Interface=="HHI_Leap") %>% get_summary_stats(releasetime)

p2<- ggplot(temp_set, aes(x=Leap_Group, y=releasetime, fill=Interface, colour = Interface))+
geom_flat_violin(position = position_nudge(x = .25, y = 0), alpha=.7)+#,adjust =2)+
geom_point(position = position_jitter(width = 0.1, height=0), size = 1)+ # the "rain"
geom_label(data = temp_plot_data, aes(x = Leap_Group, y = mean, label=round(mean, 3)), position = po
geom_point(data = temp_plot_data, aes(x = Leap_Group, y = mean), position = position_nudge(.25), col
geom_errorbar(data = temp_plot_data, aes(x = Leap_Group, y = mean, ymin=mean-(se*1.96), ymax=mean+(s
ylab('Time (seconds)')+theme_cowplot()+guides(fill = FALSE, colour = FALSE) +
scale_colour_brewer(palette = "Set2")+#coord_flip()+
scale_fill_brewer(palette = "Set2")+xlab(NULL)+
stat_pvalue_manual(data=stat.test%>%slice(3:4)%>%mutate(Interface=Group), xmin="group1", xmax="grou
labs(title="release times: interface vs. itself", caption="Means, 95% CI; Within-subjects t-test, ad
facet_grid(. ~ Interface)#+scale_x_discrete(labels=c("HHI-->Leap","Leap-->HHI"))
#ggsave(last_plot(), filename="trainingtime_leapgroup_raincloud.jpg", width=9, height=5)


# each Interface when first
temp_set <- subject_data_all_long %>% ungroup(.) %>% filter((Leap_Group=="B_Leap_first" & Interface=
temp_plot_data <- temp_set %>% group_by(Interface)%>% get_summary_stats(releasetime)

p3<- ggplot(temp_set, aes(x=Interface, y=releasetime, fill=Interface, colour = Interface))+
  geom_flat_violin(position = position_nudge(x = .25, y = 0), alpha=.7)+#,adjust =2)+
  geom_point(position = position_jitter(width = 0.1, height=0), size = 1)+ # the "rain"
geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=round(mean, 3)), position = pos
  geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(.25), co
  geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+
  ylab('Time (seconds)')+xlab(NULL)+theme_cowplot()+guides(fill = FALSE, colour = FALSE) +
  scale_colour_brewer(palette = "Set2")+#coord_flip()+
  scale_fill_brewer(palette = "Set2")+
  # stat_compare_means(method="t.test", paired=FALSE, label.x.npc="center")+
  # stat_compare_means(method="wilcox", paired=FALSE, label.x.npc="right")+
  stat_pvalue_manual(data=stat.test%>%slice(5)%>%mutate(Interface="B_Leap"), xmin="group1", xmax="g
  labs(title="release times when 1st", caption="Means, 95% CI; Within-subjects t-test, adj.: Holm")
#ggsave(last_plot(), filename="releasetime_both_first.jpg", width=6, height=4)


# release times when second (plot) - BETWEEN SUBJECTS
temp_set <- subject_data_all_long %>% ungroup(.) %>% filter((Leap_Group=="B_Leap_first" & Interface=
temp_plot_data <- temp_set %>% group_by(Interface)%>% get_summary_stats(releasetime)
```

```
p4<- ggplot(temp_set, aes(x=Interface, y=releasetime, fill=Interface, colour = Interface))+
  geom_flat_violin(position = position_nudge(x = .25, y = 0), alpha=.7)+#,adjust =2)+
  geom_point(position = position_jitter(width = 0.1, height=0), size = 1)+ # the "rain"
geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=round(mean, 3)), position = pos
  geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(.25), co
  geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+
  ylab('Time (seconds)')+xlab('Interface')+theme_cowplot()+guides(fill = FALSE, colour = FALSE) +
  scale_colour_brewer(palette = "Set2")+#coord_flip()+
  scale_fill_brewer(palette = "Set2")+
  # stat_compare_means(method="t.test", paired=FALSE, label.x.npc="center")+
  # stat_compare_means(method="wilcox", paired=FALSE, label.x.npc="right")+
  stat_pvalue_manual(data=stat.test%>%slice(5)%>%mutate(Interface="B_Leap"), xmin="group1", xmax="g
  labs(title="release times when 2nd", subtitle = , caption="Means, 95% CI; Within-subjects t-test,
ggsave(last_plot(), filename="releasetime_both_first.jpg", width=6, height=4)

plot_grid(p1, p2, p3, p4)
```




```
ggsave("releasetimes_closer_look.jpg", width=12, height=10)
do_releasetime<-p1
```

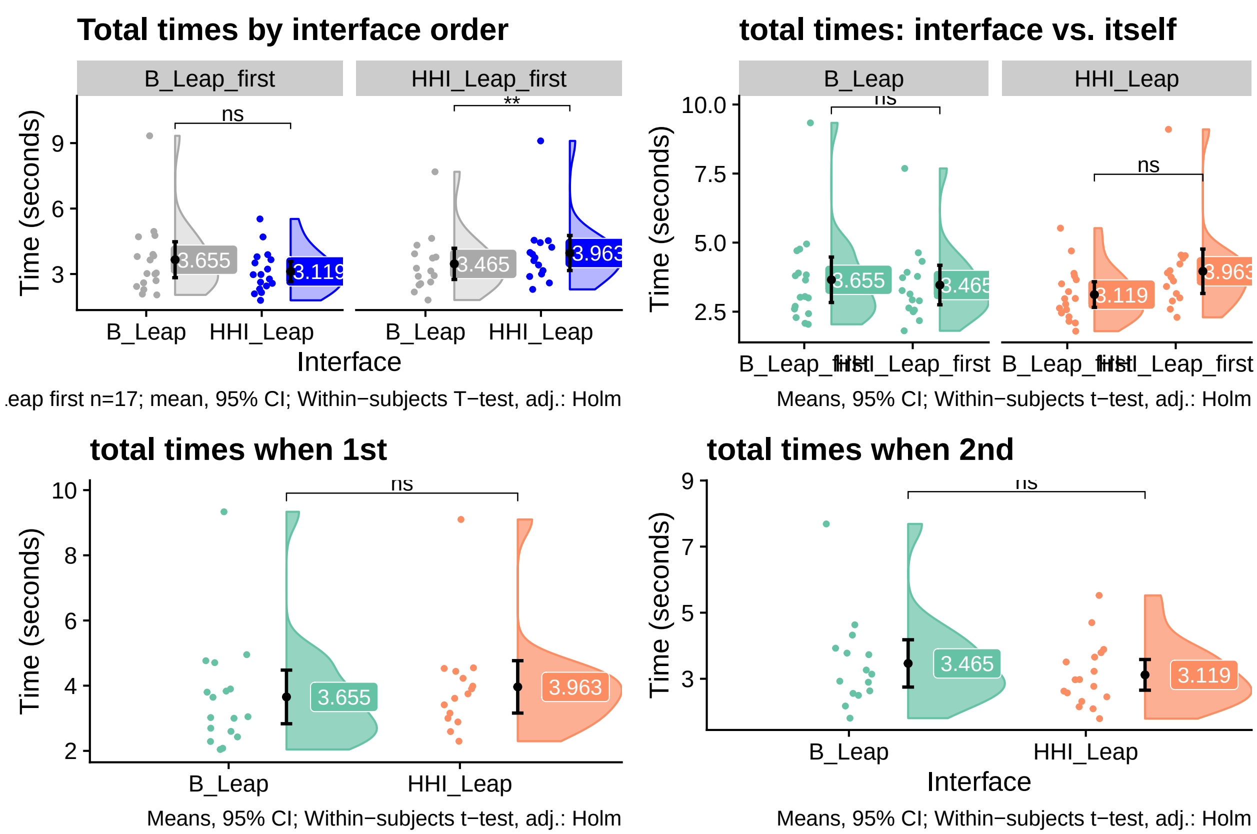
Total time I.O.

```
# total time
  Interface.order.anova <-
    anova_summary(effect.size="pes",aov(totaltime ~ Interface*InterfaceOrder + Error(id/Interface), data
```

```r
#Interface.order.anova

#Leap group
Interface.order.anova2 <-
  anova_summary(effect.size="pes",aov(totaltime ~ Interface*Leap_Group + Error(id/Interface), data=sul
#Interface.order.anova2

#Oculus group
oculus.group.anova <-
anova_summary(effect.size="pes",aov(totaltime ~ Interface*Oculus_Group + Error(id/Interface),
    data=subject_data_all_long%>%ungroup%>%
      filter(is.na(Oculus_Group)==FALSE, Interface!="Oculus") %>% mutate(Interface=factor(Interface))

Interface_order_output <- Interface_order_output %>% rbind(Interface.order.anova %>% rbind(Interface.

    # t-tests - total time d.o. - leap group
  stat.test <- subject_data_all_long %>% ungroup(.) %>% filter(Interface=="B_Leap" | Interface=="HHI_
    group_by(Leap_Group) %>%
    t_test(totaltime ~ Interface, paired=TRUE) %>% # paired b/c it's within Leap group
    mutate(test="Within-subjects T-test") %>% rename(Group=Leap_Group)

  # plot and test, split by Interface (leap vs. leap, HHI vs. HHI)
  stat.test2 <- subject_data_all_long %>% ungroup(.) %>% filter(Interface=="B_Leap" | Interface=="HHI_
    t_test(totaltime ~ Leap_Group, paired=FALSE) %>% # NOT paired b/c it's between Leap groups
    mutate(test="Between-subjects T-test")%>% rename(Group=Interface)
  stat.test <-rbind(stat.test, stat.test2)

  # Interfaces when first
  stat.test3 <- subject_data_all_long %>% ungroup(.) %>% filter((Leap_Group=="B_Leap_first" & Interfac
    t_test(totaltime ~ Interface, paired=FALSE) %>% # NOT paired b/c it's between Leap groups
    mutate(test="Between-subjects T-test", Group="when first")
  stat.test <- stat.test %>% bind_rows(stat.test3)

  #Interfaces when second
  stat.test4 <- subject_data_all_long %>% ungroup(.) %>% filter((Leap_Group=="B_Leap_first" & Interfac
    t_test(totaltime ~ Interface, paired=FALSE) %>% # NOT paired b/c it's between Leap groups
    mutate(test="Between-subjects T-test", Group="when second")
  stat.test <- stat.test %>% bind_rows(stat.test4)

  # adjust p value
  stat.test <- stat.test %>% adjust_pvalue() %>% add_significance("p") %>% mutate(Interface=group1, p


  #stat.test<- stat.test %>% mutate(Interface=group1, p.adj.signif=p.signif)

      # make labels for plots
  Leap_Group_labs<-c(paste0("HHI first n=", length((subject_data_all_long %>%
      select(id, Interface, totaltime,Leap_Group)%>%filter(Leap_Group=="HHI_Leap_first")%>%select(id)

  # by leap group (B_Leap_first, HHI_Leap_first)
  temp_set <- subject_data_all_long %>% filter(Interface=="B_Leap" | Interface=="HHI_Leap") %>%
    select(id, Interface, totaltime, Leap_Group)
  temp_plot_data <- subject_data_all_long %>% group_by(Interface, Leap_Group) %>%
```

```
  filter(Interface=="B_Leap" | Interface=="HHI_Leap") %>% get_summary_stats(totaltime)

p1<- ggplot(temp_set, aes(x=Interface, y=totaltime, fill=Interface, colour = Interface))+
geom_flat_violin(position = position_nudge(x = .25, y = 0), alpha=myalpha,adjust=mysmoothing)+
geom_point(position = position_jitter(width = 0.1, height=0), size = 1)+ # the "rain"
geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=round(mean, 3)), position = pos
geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(.25), colo
geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(se
ylab('Time (seconds)')+xlab('Interface')+theme_cowplot()+guides(fill = FALSE, colour = FALSE) +
scale_color_manual(values=mycolors)+#scale_colour_brewer(palette = "Set2")+
scale_fill_manual(values=mycolors)+#scale_fill_brewer(palette = "Set2", direction=1)+
stat_pvalue_manual(data=stat.test%>%slice(1:2)%>%rename(Leap_Group=Group), xmin="group1", xmax="grou
labs(title="Total times by interface order", caption=paste0(Leap_Group_labs[1],", ", Leap_Group_labs
facet_grid(. ~ Leap_Group)
#ggsave(last_plot(), filename="trainingtime_leapgroup_raincloud.jpg", width=9, height=5)


# plot, split by Interface (leap vs. leap, HHI vs. HHI)
temp_set <- subject_data_all_long %>% filter(Interface=="B_Leap" | Interface=="HHI_Leap") %>%
  select(id, Interface, totaltime, Leap_Group)

temp_plot_data <- subject_data_all_long %>% group_by(Interface, Leap_Group) %>%
  filter(Interface=="B_Leap" | Interface=="HHI_Leap") %>% get_summary_stats(totaltime)

p2<- ggplot(temp_set, aes(x=Leap_Group, y=totaltime, fill=Interface, colour = Interface))+
geom_flat_violin(position = position_nudge(x = .25, y = 0), alpha=.7)+#,adjust =2)+
geom_point(position = position_jitter(width = 0.1, height=0), size = 1)+ # the "rain"
geom_label(data = temp_plot_data, aes(x = Leap_Group, y = mean, label=round(mean, 3)), position = po
geom_point(data = temp_plot_data, aes(x = Leap_Group, y = mean), position = position_nudge(.25), col
geom_errorbar(data = temp_plot_data, aes(x = Leap_Group, y = mean, ymin=mean-(se*1.96), ymax=mean+(s
ylab('Time (seconds)')+theme_cowplot()+guides(fill = FALSE, colour = FALSE) +
scale_colour_brewer(palette = "Set2")+#coord_flip()+
scale_fill_brewer(palette = "Set2")+xlab(NULL)+
stat_pvalue_manual(data=stat.test%>%slice(3:4)%>%mutate(Interface=Group), xmin="group1", xmax="grou
labs(title="total times: interface vs. itself", caption="Means, 95% CI; Within-subjects t-test, adj
facet_grid(. ~ Interface)#+scale_x_discrete(labels=c("HHI-->Leap","Leap-->HHI"))
#ggsave(last_plot(), filename="trainingtime_leapgroup_raincloud.jpg", width=9, height=5)


# each Interface when first
temp_set <- subject_data_all_long %>% ungroup(.) %>% filter((Leap_Group=="B_Leap_first" & Interface=
temp_plot_data <- temp_set %>% group_by(Interface)%>% get_summary_stats(totaltime)

p3<- ggplot(temp_set, aes(x=Interface, y=totaltime, fill=Interface, colour = Interface))+
  geom_flat_violin(position = position_nudge(x = .25, y = 0), alpha=.7)+#,adjust =2)+
  geom_point(position = position_jitter(width = 0.1, height=0), size = 1)+ # the "rain"
geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=round(mean, 3)), position = pos
  geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(.25), co
  geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+
  ylab('Time (seconds)')+xlab(NULL)+theme_cowplot()+guides(fill = FALSE, colour = FALSE) +
  scale_colour_brewer(palette = "Set2")+#coord_flip()+
  scale_fill_brewer(palette = "Set2")+
  # stat_compare_means(method="t.test", paired=FALSE, label.x.npc="center")+
```

```
  # stat_compare_means(method="wilcox", paired=FALSE, label.x.npc="right")+
  stat_pvalue_manual(data=stat.test%>%slice(5)%>%mutate(Interface="B_Leap"), xmin="group1", xmax="g
  labs(title="total times when 1st", caption="Means, 95% CI; Within-subjects t-test, adj.: Holm")#+
#ggsave(last_plot(), filename="totaltime_both_first.jpg", width=6, height=4)


# total times when second (plot) - BETWEEN SUBJECTS
temp_set <- subject_data_all_long %>% ungroup(.) %>% filter((Leap_Group=="B_Leap_first" & Interface
temp_plot_data <- temp_set %>% group_by(Interface)%>% get_summary_stats(totaltime)

p4<- ggplot(temp_set, aes(x=Interface, y=totaltime, fill=Interface, colour = Interface))+
  geom_flat_violin(position = position_nudge(x = .25, y = 0), alpha=.7)+#,adjust =2)+
  geom_point(position = position_jitter(width = 0.1, height=0), size = 1)+ # the "rain"
geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=round(mean, 3)), position = po
  geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(.25), c
  geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+
  ylab('Time (seconds)')+xlab('Interface')+theme_cowplot()+guides(fill = FALSE, colour = FALSE) +
  scale_colour_brewer(palette = "Set2")+#coord_flip()+
  scale_fill_brewer(palette = "Set2")+
  # stat_compare_means(method="t.test", paired=FALSE, label.x.npc="center")+
  # stat_compare_means(method="wilcox", paired=FALSE, label.x.npc="right")+
  stat_pvalue_manual(data=stat.test%>%slice(5)%>%mutate(Interface="B_Leap"), xmin="group1", xmax="g
  labs(title="total times when 2nd", subtitle = , caption="Means, 95% CI; Within-subjects t-test, a
ggsave(last_plot(), filename="totaltime_both_first.jpg", width=6, height=4)

plot_grid(p1, p2, p3, p4)
```



```r
    ggsave("totaltimes_closer_look.jpg", width=12, height=10)
    do_totaltime<-p1
```

Accidental drop I.O.

```r
# accidental drops
Interface.order.anova <- anova_summary(effect.size = "pes", aov(Drop_Count ~ Interface *
    InterfaceOrder + Error(id/Interface), data = subject_data_all_long))
# Interface.order.anova

# Leap group
Interface.order.anova2 <- anova_summary(effect.size = "pes", aov(Drop_Count ~ Interface *
    Leap_Group + Error(id/Interface), data = subject_data_all_long %>% ungroup %>%
    filter(Interface == "B_Leap" | Interface == "HHI_Leap")))
# Interface.order.anova2

# Oculus group
oculus.group.anova <- anova_summary(effect.size = "pes", aov(Drop_Count ~ Interface *
    Oculus_Group + Error(id/Interface), data = subject_data_all_long %>% ungroup %>%
    filter(is.na(Oculus_Group) == FALSE, Interface != "Oculus") %>% mutate(Interface = factor(Interface)

Interface_order_output <- Interface_order_output %>% rbind(Interface.order.anova %>%
    rbind(Interface.order.anova2) %>% rbind(oculus.group.anova) %>% mutate(metric = "accidental drops",
    p = round(p, 4)) %>% select(metric, everything()))
```

Practice time: Interface order

The plots below may be confusing, but basically they compare every iteration of Interface (B_Leap or HHI Leap) and Interface order (B_Leap first or HHI Leap first) with Holm-adjusted p values. While each Interface is slower when first, not all differences are statistically significant. The significant effects between Interfaces when B_Leap comes first (HHI is faster), and between HHI and itself when it comes first vs. when it comes second.

In other words, when B_Leap precedes HHI Leap, the amount of training time required for a user to feel ready to use the HHI Leap is lower, but NOT the other way around. This suggests that the B_Leap trains the user for the HHI Leap better than the other way around, which makes sense, because the HHI Leap is the B_Leap with extra features (the highlighting and grab delay).

The trend towards a significant effect in the Training Time by Interface plot above is likely due to effects of Interface order, rather than something intrinsic to the Interface. This is therefore not evidence that the HHI Leap is more intuitive (that the user feels ready to use it sooner).

The ANOVA below, only between Leaps and Leap Order, offers a confusing addition: there is a main effect of Interface, suggesting that Interface alone does impact training time.

The interaction effect supports the Interface Order theory, that Interface type and order work together to impact training time– that a Interface impacts training time differently when it comes first than when it comes second.

```r
# interaction: training time: Interface*leap_group
Interface.order.anova <-
  anova_summary(effect.size="pes",aov(practice_time ~ Interface*InterfaceOrder + Error(id/Interface), da
Interface.order.anova
```

```
##                        Effect DFn DFd      F        p p<.05   pes
## 1           InterfaceOrder   5  26  1.476 2.31e-01       0.221
## 2                Interface   2  52 12.833 2.95e-05     * 0.330
## 3 Interface:InterfaceOrder  10  52  3.310 2.00e-03     * 0.389
```

```
stat.test.anova <-
anova_summary(effect.size="pes",aov(practice_time ~ Interface*Leap_Group + Error(id/Interface), data=sul
stat.test.anova
```

```
##                  Effect DFn DFd      F        p p<.05   pes
## 1            Leap_Group   1  30  0.037 0.849000       0.001
## 2             Interface   1  30  5.346 0.028000     * 0.151
## 3 Interface:Leap_Group   1  30 16.385 0.000335     * 0.353
```

```
write.csv(stat.test.anova, file="trainingtime_by_leapgroup_anova.csv")
# There was an interaction between leap group and Interface. What was driving this interaction?

 #Leap group
  Interface.order.anova2 <-
    anova_summary(effect.size="pes",aov(practice_time ~ Interface*Leap_Group + Error(id/Interface), data
  #Interface.order.anova2

  #Oculus group
  oculus.group.anova <-
  anova_summary(effect.size="pes",aov(practice_time ~ Interface*Oculus_Group + Error(id/Interface),
      data=subject_data_all_long%>%ungroup%>%
        filter(is.na(Oculus_Group)==FALSE, Interface!="Oculus") %>% mutate(Interface=factor(Interface))


Interface_order_output <- Interface_order_output %>% rbind(Interface.order.anova %>% rbind(Interface.or


###
# group all t tests then adjust p value
# t tests
stat.test <- subject_data_all_long %>% ungroup(.) %>% filter(Interface=="B_Leap" | Interface=="HHI_Leap
  group_by(Leap_Group) %>%
  t_test(practice_time ~ Interface, paired=TRUE) %>% # paired b/c it's within Leap group
  mutate(test="Within-subjects T-test") %>% rename(Group=Leap_Group)

# plot and test, split by Interface (leap vs. leap, HHI vs. HHI)
stat.test2 <- subject_data_all_long %>% ungroup(.) %>% filter(Interface=="B_Leap" | Interface=="HHI_Leap
  t_test(practice_time ~ Leap_Group, paired=FALSE) %>% # NOT paired b/c it's between Leap groups
  mutate(test="Between-subjects T-test")%>% rename(Group=Interface)
stat.test <-rbind(stat.test, stat.test2)

# Interfaces when first
stat.test3 <- subject_data_all_long %>% ungroup(.) %>% filter((Leap_Group=="B_Leap_first" & Interface==
  t_test(practice_time ~ Interface, paired=FALSE) %>% # NOT paired b/c it's between Leap groups
  mutate(test="Between-subjects T-test", Group="when first")
stat.test <- stat.test %>% bind_rows(stat.test3)

#Interfaces when second
```

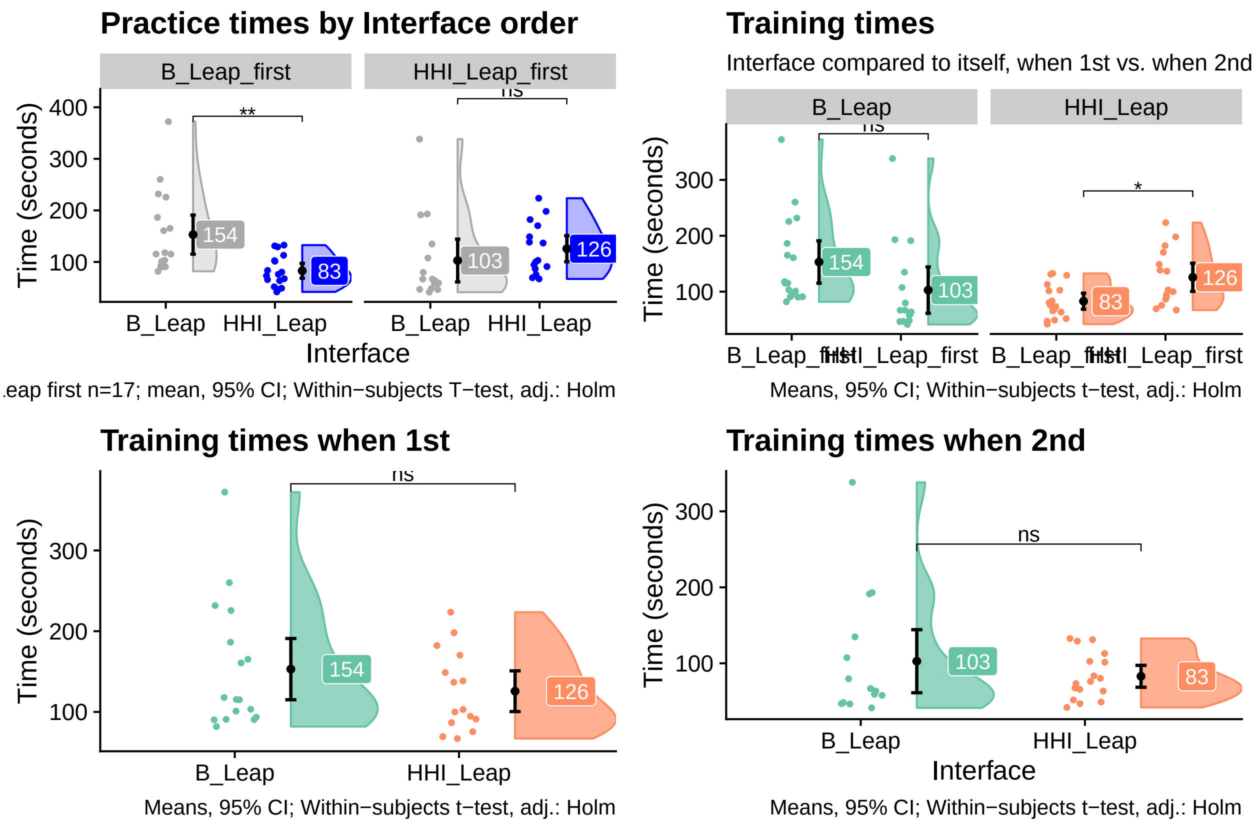
```r
stat.test4 <- subject_data_all_long %>% ungroup(.) %>% filter((Leap_Group=="B_Leap_first" & Interface==
  t_test(practice_time ~ Interface, paired=FALSE) %>% # NOT paired b/c it's between Leap groups
  mutate(test="Between-subjects T-test", Group="when second")
stat.test <- stat.test %>% bind_rows(stat.test4)

# adjust p value
stat.test <- stat.test %>% adjust_pvalue() %>% add_significance("p.adj") %>%
  mutate(Interface=group1) #to make the stat.pvalue.manual ggplot item happy
stat.test
```

```
## # A tibble: 6 x 13
##   Group .y.   group1 group2    n1    n2 statistic    df        p test    p.adj
##   <chr> <chr> <chr>  <chr>  <int> <int>     <dbl> <dbl>    <dbl> <chr>   <dbl>
## 1 B_Le~ prac~ B_Leap HHI_L~    17    17     4.17   16   7.26e-4 With~ 0.00436
## 2 HHI_~ prac~ B_Leap HHI_L~    15    15    -1.50   14   1.57e-1 With~ 0.471
## 3 B_Le~ prac~ B_Lea~ HHI_L~    17    15     1.75   29.3 9.01e-2 Betw~ 0.360
## 4 HHI_~ prac~ B_Lea~ HHI_L~    17    15    -2.89   22.5 8.45e-3 Betw~ 0.0422
## 5 when~ prac~ B_Leap HHI_L~    17    15     1.18   27.2 2.49e-1 Betw~ 0.498
## 6 when~ prac~ B_Leap HHI_L~    15    17     0.887  17.4 3.87e-1 Betw~ 0.498
## # ... with 2 more variables: p.adj.signif <chr>, Interface <chr>
```

```r
###

# make labels for plots
Leap_Group_labs<-c(paste0("HHI first n=", length((subject_data_all_long %>%
    select(id, Interface, practice_time,Leap_Group)%>%filter(Leap_Group=="HHI_Leap_first")%>%select(id)

# by leap group (B_Leap_first, HHI_Leap_first)
temp_set <- subject_data_all_long %>% filter(Interface=="B_Leap" | Interface=="HHI_Leap") %>%
  select(id, Interface, practice_time, Leap_Group)
temp_plot_data <- subject_data_all_long %>% group_by(Interface, Leap_Group) %>%
  filter(Interface=="B_Leap" | Interface=="HHI_Leap") %>% get_summary_stats(practice_time)

p1<- ggplot(temp_set, aes(x=Interface, y=practice_time, fill=Interface, colour = Interface))+
geom_flat_violin(position = position_nudge(x = .25, y = 0), alpha=myalpha, adjust=mysmoothing)+
geom_point(position = position_jitter(width = 0.1, height=0), size = 1)+ # the "rain"
geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=ceiling(mean)), position = positio
geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(.25), colour
geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(se*1.
ylab('Time (seconds)')+xlab('Interface')+theme_cowplot()+guides(fill = FALSE, colour = FALSE) +
scale_color_manual(values=mycolors)+#scale_colour_brewer(palette = "Set2")+
scale_fill_manual(values=mycolors)+#scale_fill_brewer(palette = "Set2", direction=1)+
stat_pvalue_manual(data=stat.test%>%slice(1:2)%>%rename(Leap_Group=Group), xmin="group1", xmax="group2"
labs(title="Practice times by Interface order", caption=paste0(Leap_Group_labs[1],", ", Leap_Group_labs
facet_grid(. ~ Leap_Group)
ggsave(last_plot(), filename="trainingtime_leapgroup_raincloud.jpg", width=9, height=5)


# plot, split by Interface (leap vs. leap, HHI vs. HHI)
temp_set <- subject_data_all_long %>% filter(Interface=="B_Leap" | Interface=="HHI_Leap") %>%
  select(id, Interface, practice_time, Leap_Group)

temp_plot_data <- subject_data_all_long %>% group_by(Interface, Leap_Group) %>%
```

```r
  filter(Interface=="B_Leap" | Interface=="HHI_Leap") %>% get_summary_stats(practice_time)

p2<- ggplot(temp_set, aes(x=Leap_Group, y=practice_time, fill=Interface, colour = Interface))+
geom_flat_violin(position = position_nudge(x = .25, y = 0), alpha=.7)+#,adjust =2)+
geom_point(position = position_jitter(width = 0.1, height=0), size = 1)+ # the "rain"
geom_label(data = temp_plot_data, aes(x = Leap_Group, y = mean, label=ceiling(mean)), position = positi
geom_point(data = temp_plot_data, aes(x = Leap_Group, y = mean), position = position_nudge(.25), colour
geom_errorbar(data = temp_plot_data, aes(x = Leap_Group, y = mean, ymin=mean-(se*1.96), ymax=mean+(se*1
ylab('Time (seconds)')+theme_cowplot()+guides(fill = FALSE, colour = FALSE) +
scale_colour_brewer(palette = "Set2")+#coord_flip()+
scale_fill_brewer(palette = "Set2")+xlab(NULL)+
stat_pvalue_manual(data=stat.test%>%slice(3:4)%>%mutate(Interface=Group), xmin="group1", xmax="group2",
labs(title="Training times", subtitle="Interface compared to itself, when 1st vs. when 2nd", caption="Me
facet_grid(. ~ Interface)#+scale_x_discrete(labels=c("HHI-->Leap","Leap-->HHI"))
ggsave(last_plot(), filename="trainingtime_leapgroup_raincloud.jpg", width=9, height=5)


# each Interface when first
temp_set <- subject_data_all_long %>% ungroup(.) %>% filter((Leap_Group=="B_Leap_first" & Interface=="B
temp_plot_data <- temp_set %>% group_by(Interface)%>% get_summary_stats(practice_time)

p3<- ggplot(temp_set, aes(x=Interface, y=practice_time, fill=Interface, colour = Interface))+
  geom_flat_violin(position = position_nudge(x = .25, y = 0), alpha=.7)+#,adjust =2)+
  geom_point(position = position_jitter(width = 0.1, height=0), size = 1)+ # the "rain"
geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=ceiling(mean)), position = positio
  geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(.25), colou
  geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(se*
  ylab('Time (seconds)')+xlab(NULL)+theme_cowplot()+guides(fill = FALSE, colour = FALSE) +
  scale_colour_brewer(palette = "Set2")+#coord_flip()+
  scale_fill_brewer(palette = "Set2")+
  # stat_compare_means(method="t.test", paired=FALSE, label.x.npc="center")+
  # stat_compare_means(method="wilcox", paired=FALSE, label.x.npc="right")+
  stat_pvalue_manual(data=stat.test%>%slice(5)%>%mutate(Interface="B_Leap"), xmin="group1", xmax="group
  labs(title="Training times when 1st", caption="Means, 95% CI; Within-subjects t-test, adj.: Holm")#+f
ggsave(last_plot(), filename="trainingtime_both_first.jpg", width=6, height=4)


# training times when second (plot) - BETWEEN SUBJECTS
temp_set <- subject_data_all_long %>% ungroup(.) %>% filter((Leap_Group=="B_Leap_first" & Interface=="HH
temp_plot_data <- temp_set %>% group_by(Interface)%>% get_summary_stats(practice_time)

p4<- ggplot(temp_set, aes(x=Interface, y=practice_time, fill=Interface, colour = Interface))+
  geom_flat_violin(position = position_nudge(x = .25, y = 0), alpha=.7)+#,adjust =2)+
  geom_point(position = position_jitter(width = 0.1, height=0), size = 1)+ # the "rain"
geom_label(data = temp_plot_data, aes(x = Interface, y = mean, label=ceiling(mean)), position = positio
  geom_point(data = temp_plot_data, aes(x = Interface, y = mean), position = position_nudge(.25), colou
  geom_errorbar(data = temp_plot_data, aes(x = Interface, y = mean, ymin=mean-(se*1.96), ymax=mean+(se*
  ylab('Time (seconds)')+xlab('Interface')+theme_cowplot()+guides(fill = FALSE, colour = FALSE) +
  scale_colour_brewer(palette = "Set2")+#coord_flip()+
  scale_fill_brewer(palette = "Set2")+
  # stat_compare_means(method="t.test", paired=FALSE, label.x.npc="center")+
  # stat_compare_means(method="wilcox", paired=FALSE, label.x.npc="right")+
  stat_pvalue_manual(data=stat.test%>%slice(5)%>%mutate(Interface="B_Leap"), xmin="group1", xmax="group
```

```
  labs(title="Training times when 2nd", subtitle = , caption="Means, 95% CI; Within-subjects t-test, ad
ggsave(last_plot(), filename="trainingtime_both_first.jpg", width=6, height=4)

plot_grid(p1, p2, p3, p4)
```




```
ggsave("practice_times_closer_look.jpg", width=12, height=10)
do_trainingtime<-p1
```

I.O. Subjective

```
# Basic stats for Interface order subgroups
table(subject_data_all_wide$InterfaceOrder)
```

```
##
## LHO LOH HLO HOL OLH OHL
##   5   6   7   3   6   5
```

```
table(subject_data_all_wide$Leap_Group)
```

```
##
##   B_Leap_first HHI_Leap_first
##             17             15
```

```r
table(subject_data_all_wide$Oculus_Group)
```

```
##
## Oculus_first  Oculus_last
##           11           12
```

```r
cat("leap order within oculus order groups")
```

```
## leap order within oculus order groups
```

```r
cat("Oculus first")
```

```
## Oculus first
```

```r
table((subject_data_all_wide %>% filter(Oculus_Group == "Oculus_first"))$Leap_Group)
```

```
##
##   B_Leap_first HHI_Leap_first
##              6              5
```

```r
cat("Oculus last")
```

```
## Oculus last
```

```r
table((subject_data_all_wide %>% filter(Oculus_Group == "Oculus_last"))$Leap_Group)
```

```
##
##   B_Leap_first HHI_Leap_first
##              5              7
```

```r
# Interface order for subjective questions

cat("\n\nSubjective Q's -- Interface Order\n")
```

```
##
##
## Subjective Q's -- Interface Order
```

```r
# comfortable
cat("\nComfortable\n")
```

```
##
## Comfortable
```

```r
Interface.order.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface *
    InterfaceOrder + Error(id/Interface), data = likert_scores %>% filter(question ==
    "comfortable")))
Interface.order.anova %>% filter(`p<.05` == "*")
```

```
##                        Effect DFn DFd      F        p p<.05   pes
## 1                   Interface   2  52 10.433 0.000155     * 0.286
## 2 Interface:InterfaceOrder  10  52  2.977 0.005000     * 0.364
```

```
# Leap group
leap.group.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface * Leap_Group +
    Error(id/Interface), data = likert_scores %>% ungroup %>% filter(Interface ==
    "B_Leap" | Interface == "HHI_Leap", question == "comfortable")))
leap.group.anova %>% filter(`p<.05` == "*")
```

```
##                   Effect DFn DFd     F     p p<.05   pes
## 1 Interface:Leap_Group   1  30 5.586 0.025     * 0.157
```

```
# Oculus Oculus group
oculus.group.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface *
    Oculus_Group + Error(id/Interface), data = likert_scores %>% ungroup %>% filter(Oculus_Group ==
    "Oculus_first" | Oculus_Group == "Oculus_last", Interface != "Oculus", question ==
    "comfortable") %>% mutate(Interface = factor(Interface))))
oculus.group.anova %>% filter(p < 0.1)
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```
Interface_order_output <- Interface_order_output %>% rbind(Interface.order.anova %>%
    rbind(Interface.order.anova2) %>% rbind(oculus.group.anova) %>% mutate(metric = "comfortable",
    p = round(p, 4)) %>% select(metric, everything()))


# t test
t.test <- likert_scores %>% ungroup %>% filter(Oculus_Group == "Oculus_first" | Oculus_Group ==
    "Oculus_last", Interface != "Oculus", question == "comfortable") %>% mutate(Interface = factor(Inte
    Oculus_Group = factor(Oculus_Group)) %>% group_by(Oculus_Group) %>% pairwise_t_test(score ~
    Interface, paired = TRUE)
# t.test

# precise
cat("\nPrecise\n")
```

```
##
## Precise
```

```
Interface.order.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface *
    InterfaceOrder + Error(id/Interface), data = likert_scores %>% filter(question ==
    "precise")))
Interface.order.anova %>% filter(`p<.05` == "*")
```

```
##                        Effect DFn DFd      F        p p<.05   pes
## 1                   Interface   2  52 45.532 3.74e-12     * 0.637
## 2 Interface:InterfaceOrder  10  52  4.911 5.57e-05     * 0.486
```

```r
# Leap group
leap.group.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface * Leap_Group +
    Error(id/Interface), data = likert_scores %>% ungroup %>% filter(Interface ==
    "B_Leap" | Interface == "HHI_Leap", question == "precise")))
leap.group.anova %>% filter(`p<.05` == "*")
```

```
##                  Effect DFn DFd      F        p p<.05   pes
## 1 Interface:Leap_Group   1  30 22.918 4.25e-05     * 0.433
```

```r
# oculus group
oculus.group.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface *
    Oculus_Group + Error(id/Interface), data = likert_scores %>% ungroup %>% filter(Oculus_Group ==
    "Oculus_first" | Oculus_Group == "Oculus_last", Interface != "Oculus", question ==
    "precise") %>% mutate(Interface = factor(Interface))))
oculus.group.anova %>% filter(`p<.05` == "*")
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```r
oculus.group.anova %>% filter(p < 0.1)
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```r
Interface_order_output <- Interface_order_output %>% rbind(Interface.order.anova %>%
    rbind(Interface.order.anova2) %>% rbind(oculus.group.anova) %>% mutate(metric = "precise",
    p = round(p, 4)) %>% select(metric, everything()))


# intuitive
cat("\nIntuitive\n")
```

```
##
## Intuitive
```

```r
Interface.order.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface *
    InterfaceOrder + Error(id/Interface), data = likert_scores %>% filter(question ==
    "intuitive")))
Interface.order.anova %>% filter(`p<.05` == "*")
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```r
# Leap group
leap.group.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface * Leap_Group +
    Error(id/Interface), data = likert_scores %>% ungroup %>% filter(Interface ==
    "B_Leap" | Interface == "HHI_Leap", question == "intuitive")))
leap.group.anova %>% filter(`p<.05` == "*")
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```
# oculus group
oculus.group.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface *
    Oculus_Group + Error(id/Interface), data = likert_scores %>% ungroup %>% filter(Oculus_Group ==
    "Oculus_first" | Oculus_Group == "Oculus_last", Interface != "Oculus", question ==
    "intuitive") %>% mutate(Interface = factor(Interface))))
oculus.group.anova %>% filter(`p<.05` == "*")
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```
oculus.group.anova %>% filter(p < 0.1)
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```
Interface_order_output <- Interface_order_output %>% rbind(Interface.order.anova %>%
    rbind(Interface.order.anova2) %>% rbind(oculus.group.anova) %>% mutate(metric = "intuitive",
    p = round(p, 4)) %>% select(metric, everything()))


# tiring
cat("\nTiring\n")
```

```
##
## Tiring
```

```
Interface.order.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface *
    InterfaceOrder + Error(id/Interface), data = likert_scores %>% filter(question ==
    "tiring")))
Interface.order.anova %>% filter(`p<.05` == "*")
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```
# Leap group
leap.group.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface * Leap_Group +
    Error(id/Interface), data = likert_scores %>% ungroup %>% filter(Interface ==
    "B_Leap" | Interface == "HHI_Leap", question == "tiring")))
leap.group.anova %>% filter(`p<.05` == "*")
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```
# oculus group
oculus.group.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface *
    Oculus_Group + Error(id/Interface), data = likert_scores %>% ungroup %>% filter(Oculus_Group ==
    "Oculus_first" | Oculus_Group == "Oculus_last", Interface != "Oculus", question ==
    "tiring") %>% mutate(Interface = factor(Interface))))
oculus.group.anova %>% filter(`p<.05` == "*")
```

```
## [1] Effect DFn    DFd    F      p      p<.05   pes
## <0 rows> (or 0-length row.names)
```

```
oculus.group.anova %>% filter(p < 0.1)
```

```
## [1] Effect DFn    DFd    F      p      p<.05   pes
## <0 rows> (or 0-length row.names)
```

```
Interface_order_output <- Interface_order_output %>% rbind(Interface.order.anova %>%
    rbind(Interface.order.anova2) %>% rbind(oculus.group.anova) %>% mutate(metric = "tiring",
    p = round(p, 4)) %>% select(metric, everything()))


# gripping
cat("\nGripping\n")
```

```
##
## Gripping
```

```
Interface.order.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface *
    InterfaceOrder + Error(id/Interface), data = likert_scores %>% filter(question ==
    "gripping")))
Interface.order.anova %>% filter(`p<.05` == "*")
```

```
##                      Effect DFn DFd      F        p p<.05   pes
## 1                 Interface   2  52 33.199 5.12e-10     * 0.561
## 2 Interface:InterfaceOrder  10  52  3.589 1.00e-03     * 0.408
```

```
# Leap group
leap.group.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface * Leap_Group +
    Error(id/Interface), data = likert_scores %>% ungroup %>% filter(Interface ==
    "B_Leap" | Interface == "HHI_Leap", question == "gripping")))
leap.group.anova %>% filter(`p<.05` == "*")
```

```
## [1] Effect DFn    DFd    F      p      p<.05   pes
## <0 rows> (or 0-length row.names)
```

```
# oculus group
oculus.group.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface *
    Oculus_Group + Error(id/Interface), data = likert_scores %>% ungroup %>% filter(Oculus_Group ==
    "Oculus_first" | Oculus_Group == "Oculus_last", Interface != "Oculus", question ==
    "gripping") %>% mutate(Interface = factor(Interface))))
oculus.group.anova %>% filter(`p<.05` == "*")
```

```
## [1] Effect DFn    DFd    F      p      p<.05   pes
## <0 rows> (or 0-length row.names)
```

```
oculus.group.anova %>% filter(p < 0.1)
```

```
##         Effect DFn DFd     F     p p<.05   pes
## 1 Oculus_Group   1  21 3.094 0.093       0.128
```

```r
Interface_order_output <- Interface_order_output %>% rbind(Interface.order.anova %>%
    rbind(Interface.order.anova2) %>% rbind(oculus.group.anova) %>% mutate(metric = "gripping",
    p = round(p, 4)) %>% select(metric, everything()))


# releasing
cat("\nReleasing\n")
```

```
##
## Releasing
```

```r
Interface.order.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface *
    InterfaceOrder + Error(id/Interface), data = likert_scores %>% filter(question ==
    "releasing")))
Interface.order.anova %>% filter(`p<.05` == "*")
```

```
##                      Effect DFn DFd      F        p p<.05   pes
## 1                 Interface   2  52 34.807 2.55e-10     * 0.572
## 2 Interface:InterfaceOrder  10  52  3.239 3.00e-03     * 0.384
```

```r
# Leap group
leap.group.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface * Leap_Group +
    Error(id/Interface), data = likert_scores %>% ungroup %>% filter(Interface ==
    "B_Leap" | Interface == "HHI_Leap", question == "releasing")))
leap.group.anova %>% filter(`p<.05` == "*")
```

```
##                Effect DFn DFd      F        p p<.05   pes
## 1 Interface:Leap_Group   1  30 14.405 0.000668     * 0.324
```

```r
# oculus group
oculus.group.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface *
    Oculus_Group + Error(id/Interface), data = likert_scores %>% ungroup %>% filter(Oculus_Group ==
    "Oculus_first" | Oculus_Group == "Oculus_last", Interface != "Oculus", question ==
    "releasing") %>% mutate(Interface = factor(Interface))))
oculus.group.anova %>% filter(`p<.05` == "*")
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```r
oculus.group.anova %>% filter(p < 0.1)
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```r
Interface_order_output <- Interface_order_output %>% rbind(Interface.order.anova %>%
    rbind(Interface.order.anova2) %>% rbind(oculus.group.anova) %>% mutate(metric = "releasing",
    p = round(p, 4)) %>% select(metric, everything()))


# natural
cat("\nNatural\n")
```

```
##
## Natural
```

```r
Interface.order.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface *
    InterfaceOrder + Error(id/Interface), data = likert_scores %>% filter(question ==
    "natural")))
Interface.order.anova %>% filter(`p<.05` == "*")
```

```
##                       Effect DFn DFd     F     p p<.05   pes
## 1 Interface:InterfaceOrder  10  52 2.205 0.032     * 0.298
```

```r
# Leap group
leap.group.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface * Leap_Group +
    Error(id/Interface), data = likert_scores %>% ungroup %>% filter(Interface ==
    "B_Leap" | Interface == "HHI_Leap", question == "natural")))
leap.group.anova %>% filter(`p<.05` == "*")
```

```
##                   Effect DFn DFd     F     p p<.05   pes
## 1 Interface:Leap_Group   1  30 7.721 0.009     * 0.205
```

```r
# oculus group
oculus.group.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface *
    Oculus_Group + Error(id/Interface), data = likert_scores %>% ungroup %>% filter(Oculus_Group ==
    "Oculus_first" | Oculus_Group == "Oculus_last", Interface != "Oculus", question ==
    "natural") %>% mutate(Interface = factor(Interface))))
oculus.group.anova %>% filter(`p<.05` == "*")
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```r
oculus.group.anova %>% filter(p < 0.1)
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```r
Interface_order_output <- Interface_order_output %>% rbind(Interface.order.anova %>%
    rbind(Interface.order.anova2) %>% rbind(oculus.group.anova) %>% mutate(metric = "natural",
    p = round(p, 4)) %>% select(metric, everything()))


# recommend
cat("\nRecommend\n")
```

```
##
## Recommend
```

```r
Interface.order.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface *
    InterfaceOrder + Error(id/Interface), data = likert_scores %>% filter(question ==
    "recommend")))
Interface.order.anova %>% filter(`p<.05` == "*")
```

```
##      Effect DFn DFd     F     p p<.05   pes
## 1 Interface   2  52 7.467 0.001     * 0.223
```

```
# Leap group
leap.group.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface * Leap_Group +
    Error(id/Interface), data = likert_scores %>% ungroup %>% filter(Interface ==
    "B_Leap" | Interface == "HHI_Leap", question == "recommend")))
leap.group.anova %>% filter(`p<.05` == "*")
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```
# oculus group
oculus.group.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface *
    Oculus_Group + Error(id/Interface), data = likert_scores %>% ungroup %>% filter(Oculus_Group ==
    "Oculus_first" | Oculus_Group == "Oculus_last", Interface != "Oculus", question ==
    "recommend") %>% mutate(Interface = factor(Interface))))
oculus.group.anova %>% filter(`p<.05` == "*")
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```
oculus.group.anova %>% filter(p < 0.1)
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```
Interface_order_output <- Interface_order_output %>% rbind(Interface.order.anova %>%
    rbind(Interface.order.anova2) %>% rbind(oculus.group.anova) %>% mutate(metric = "recommend",
    p = round(p, 4)) %>% select(metric, everything()))


# agency
cat("\nAgency\n")
```

```
##
## Agency
```

```
Interface.order.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface *
    InterfaceOrder + Error(id/Interface), data = likert_scores %>% filter(question ==
    "agency")))
Interface.order.anova %>% filter(`p<.05` == "*")
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```
# Leap group
leap.group.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface * Leap_Group +
    Error(id/Interface), data = likert_scores %>% ungroup %>% filter(Interface ==
    "B_Leap" | Interface == "HHI_Leap", question == "agency")))
leap.group.anova %>% filter(`p<.05` == "*")
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```r
# oculus group
oculus.group.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface *
    Oculus_Group + Error(id/Interface), data = likert_scores %>% ungroup %>% filter(Oculus_Group ==
    "Oculus_first" | Oculus_Group == "Oculus_last", Interface != "Oculus", question ==
    "agency")))
oculus.group.anova %>% filter(`p<.05` == "*")
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```r
oculus.group.anova %>% filter(p < 0.1)
```

```
##                    Effect DFn DFd     F     p p<.05   pes
## 1            Oculus_Group   1  21 3.021 0.097       0.126
## 2 Interface:Oculus_Group   1  21 3.319 0.083       0.136
```

```r
Interface_order_output <- Interface_order_output %>% rbind(Interface.order.anova %>%
    rbind(Interface.order.anova2) %>% rbind(oculus.group.anova) %>% mutate(metric = "agency",
    p = round(p, 4)) %>% select(metric, everything()))



# satisfaction
cat("\nSatisfaction\n")
```

```
##
## Satisfaction
```

```r
Interface.order.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface *
    InterfaceOrder + Error(id/Interface), data = likert_scores %>% filter(question ==
    "satisfaction")))
Interface.order.anova %>% filter(`p<.05` == "*")
```

```
##                      Effect DFn DFd     F     p p<.05   pes
## 1                 Interface   2  52 7.136 0.002     * 0.215
## 2 Interface:InterfaceOrder  10  52 2.298 0.025     * 0.306
```

```r
# Leap group
leap.group.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface * Leap_Group +
    Error(id/Interface), data = likert_scores %>% ungroup %>% filter(Interface ==
    "B_Leap" | Interface == "HHI_Leap", question == "satisfaction")))
leap.group.anova %>% filter(`p<.05` == "*")
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```r
# oculus group
oculus.group.anova <- anova_summary(effect.size = "pes", aov(score ~ Interface *
    Oculus_Group + Error(id/Interface), data = likert_scores %>% ungroup %>% filter(Oculus_Group ==
    "Oculus_first" | Oculus_Group == "Oculus_last", Interface != "Oculus", question ==
    "satisfaction") %>% mutate(Interface = factor(Interface))))
oculus.group.anova %>% filter(`p<.05` == "*")
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```r
oculus.group.anova %>% filter(p < 0.1)
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```r
Interface_order_output <- Interface_order_output %>% rbind(Interface.order.anova %>%
    rbind(Interface.order.anova2) %>% rbind(oculus.group.anova) %>% mutate(metric = "satisfaction",
    p = round(p, 4)) %>% select(metric, everything()))


# SUS
cat("\nSUS score\n")
```

```
##
## SUS score
```

```r
Interface.order.anova <- anova_summary(effect.size = "pes", aov(SUS ~ Interface *
    InterfaceOrder + Error(id/Interface), data = subject_data_all_long))
Interface.order.anova %>% filter(`p<.05` == "*")
```

```
##                      Effect DFn DFd     F        p p<.05   pes
## 1                 Interface   2  52 8.808 0.000508     * 0.253
## 2 Interface:InterfaceOrder  10  52 2.457 0.017000     * 0.321
```

```r
# Leap group
leap.group.anova <- anova_summary(effect.size = "pes", aov(SUS ~ Interface * Leap_Group +
    Error(id/Interface), data = subject_data_all_long %>% ungroup %>% filter(Interface ==
    "B_Leap" | Interface == "HHI_Leap") %>% mutate(Interface = factor(Interface))))
leap.group.anova %>% filter(`p<.05` == "*")
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```r
# oculus group
oculus.group.anova <- anova_summary(effect.size = "pes", aov(SUS ~ Interface * Oculus_Group +
    Error(id/Interface), data = subject_data_all_long %>% ungroup %>% filter(Oculus_Group ==
    "Oculus_first" | Oculus_Group == "Oculus_last", Interface != "Oculus") %>% mutate(Interface = facto
oculus.group.anova %>% filter(`p<.05` == "*")
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```r
oculus.group.anova %>% filter(p < 0.1)
```

```
## [1] Effect DFn    DFd    F      p      p<.05  pes
## <0 rows> (or 0-length row.names)
```

```r
Interface_order_output <- Interface_order_output %>% rbind(Interface.order.anova %>%
    rbind(Interface.order.anova2) %>% rbind(oculus.group.anova) %>% mutate(metric = "SUS",
    p = round(p, 4)) %>% select(metric, everything()))



# pairwise t-tests
likert_scores %>% filter(Interface == "HHI_Leap", Oculus_Group != "NA") %>% group_by(question) %>%
    pairwise_t_test(score ~ Oculus_Group) %>% adjust_pvalue() %>% add_significance(p.col = "p.adj",
    output.col = "p.adj.signif")
```

```
## # A tibble: 10 x 10
##    question  .y.   group1  group2    n1    n2      p p.signif p.adj p.adj.signif
##    <chr>     <chr> <chr>   <chr>  <int> <int>  <dbl> <chr>    <dbl> <chr>
##  1 agency    score Oculus~ Oculu~    11    12 0.0224 *        0.224 ns
##  2 comforta~ score Oculus~ Oculu~    11    12 0.136  ns       0.952 ns
##  3 gripping  score Oculus~ Oculu~    11    12 0.0688 ns       0.619 ns
##  4 intuitive score Oculus~ Oculu~    11    12 0.804  ns       1     ns
##  5 natural   score Oculus~ Oculu~    11    12 0.465  ns       1     ns
##  6 precise   score Oculus~ Oculu~    11    12 0.749  ns       1     ns
##  7 recommend score Oculus~ Oculu~    11    12 0.291  ns       1     ns
##  8 releasing score Oculus~ Oculu~    11    12 0.868  ns       1     ns
##  9 satisfac~ score Oculus~ Oculu~    11    12 0.0926 ns       0.741 ns
## 10 tiring    score Oculus~ Oculu~    11    12 0.142  ns       0.952 ns
```

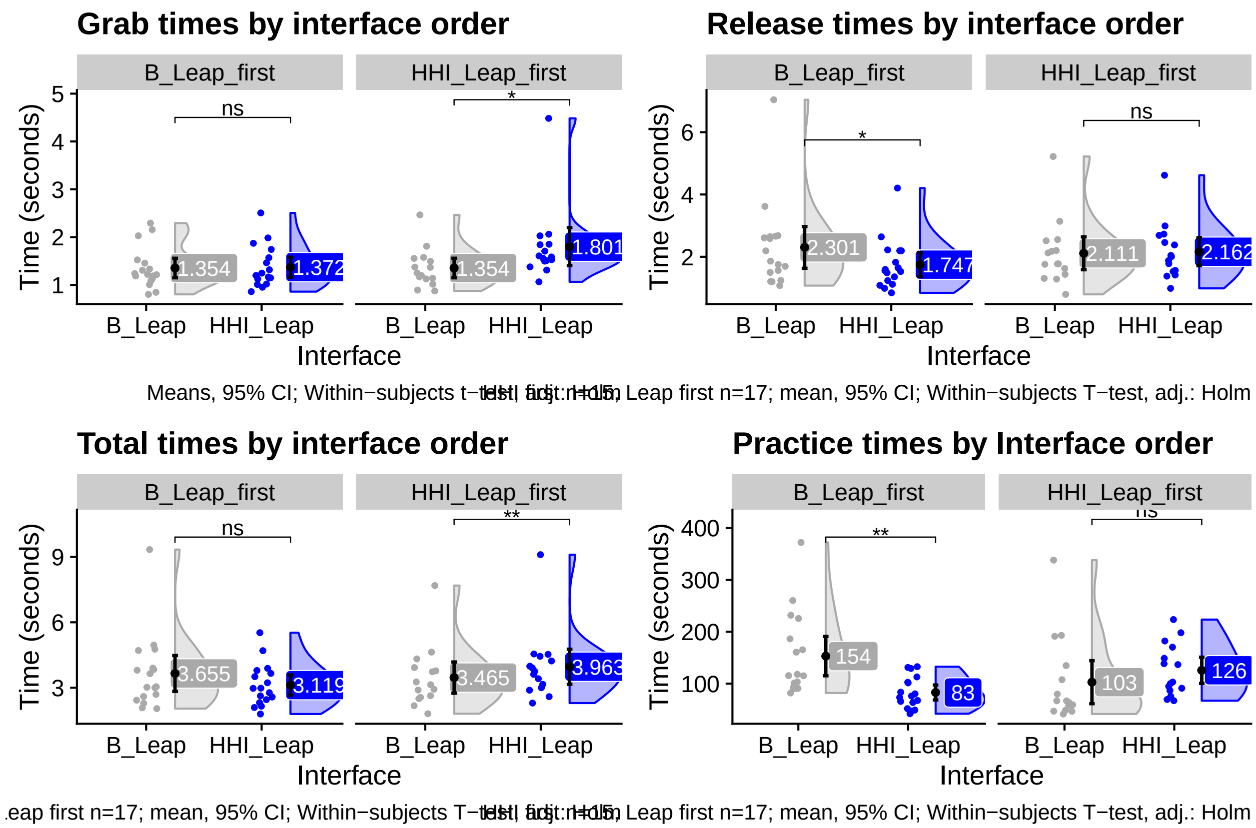
```r
# Overall preference
cat("\nOverall preference\n")
```

```
##
## Overall preference
```

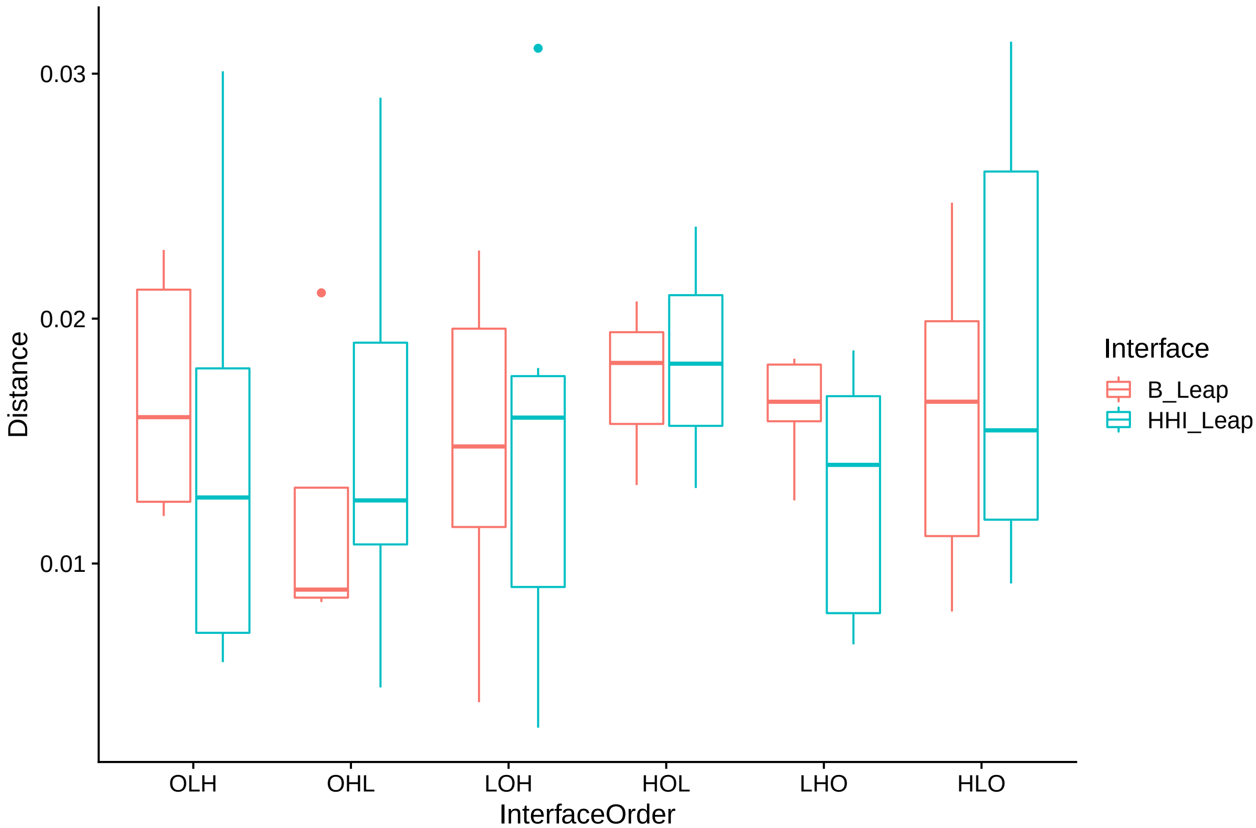
```r
# Interface order and preferred condition note: counts may be too low for chi sq
chi <- chisq.test(table(subject_data_all_wide$InterfaceOrder, subject_data_all_wide$PrefCondition))
# chi$expected chi$observed chi

chi <- chisq.test(table(subject_data_all_wide$Oculus_Group, subject_data_all_wide$PrefCondition))
# chi$expected chi$observed chi
```

I.O. Plots

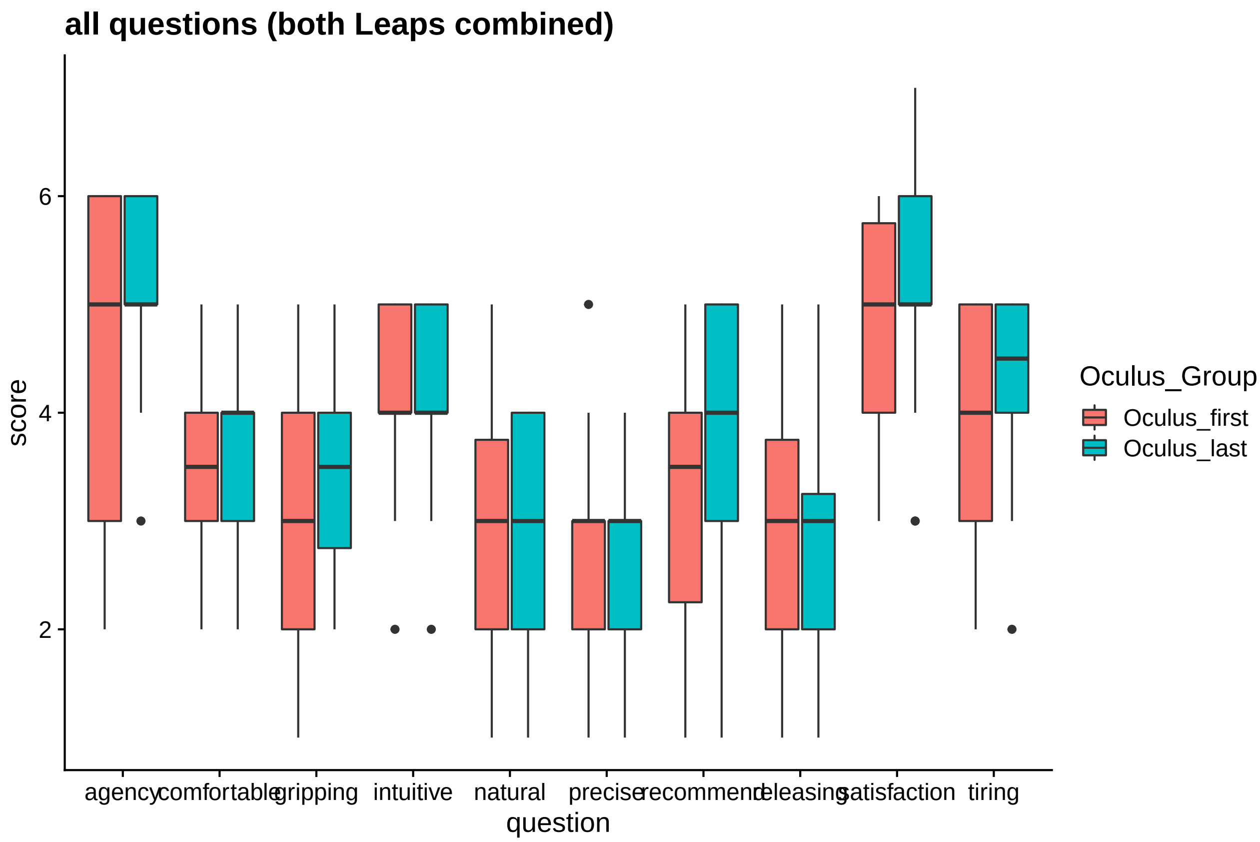
```r
# leap group and times
plot_grid(do_grabtime, do_releasetime, do_totaltime, do_trainingtime)
```



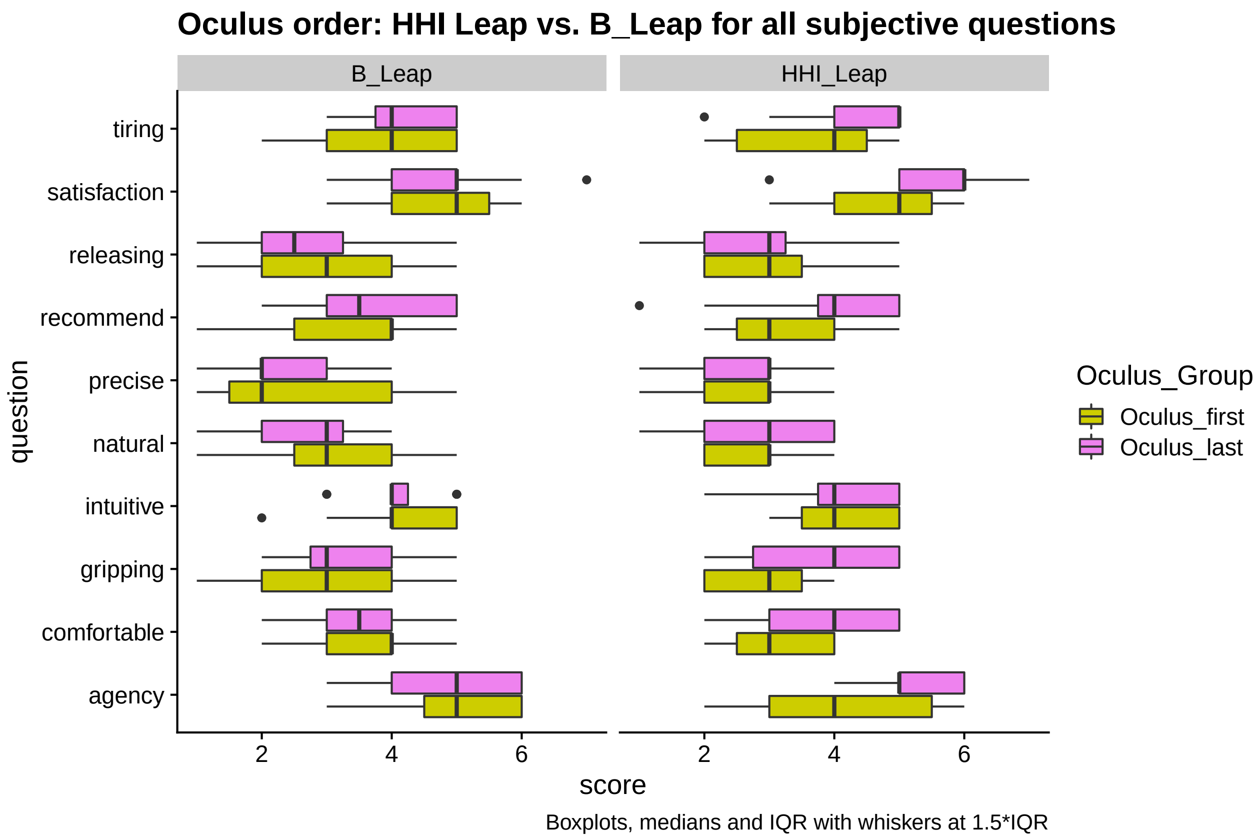

```r
ggsave("Interface_order_leapgroup_times.jpg", width = 10, height = 10)

# performance
ggplot(subject_data_all_long %>% filter(Interface != "Oculus") %>% mutate(InterfaceOrder = factor(Inter
    levels = c("OLH", "OHL", "LOH", "HOL", "LHO", "HLO")), Interface = factor(Interface)),
    aes(InterfaceOrder, Distance, color = Interface)) + geom_boxplot()
```



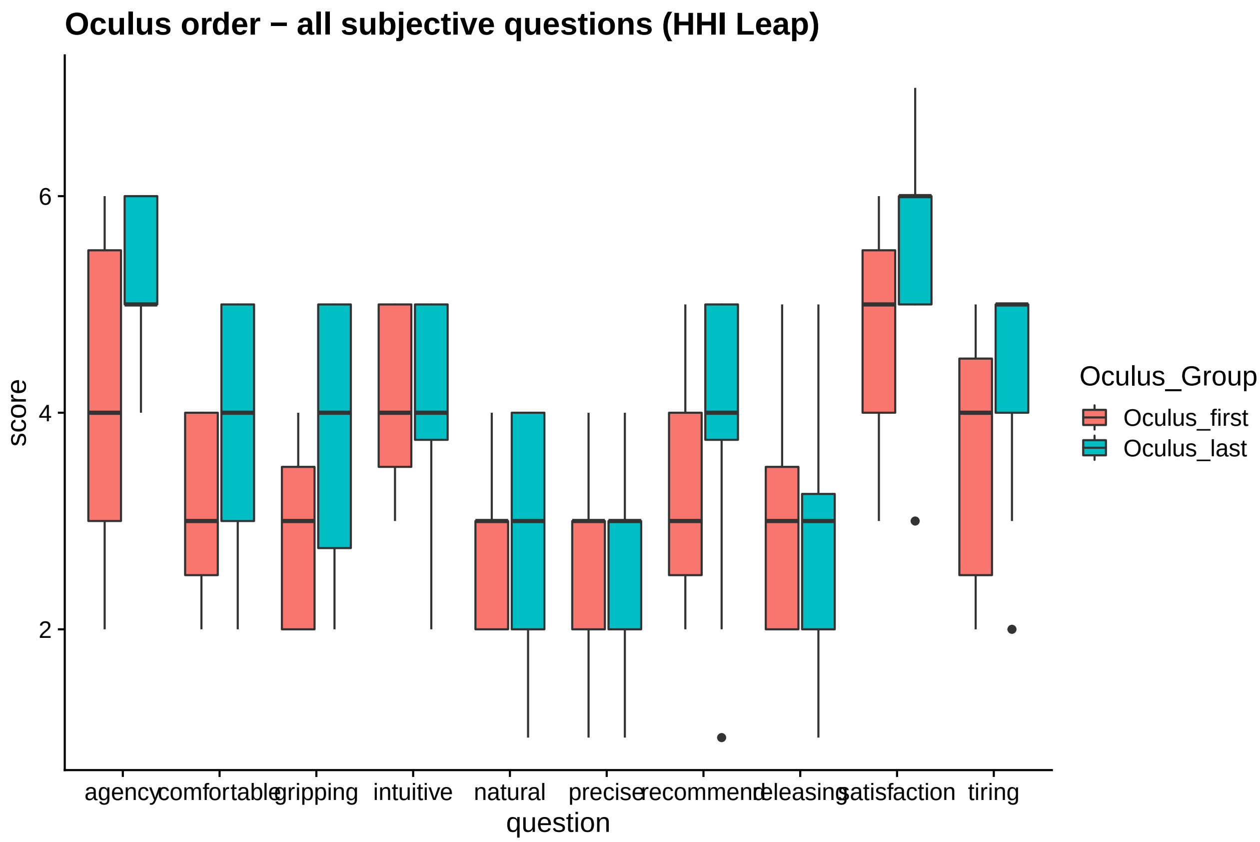

```r
# subjective box plots of scores on subjective questions, comparing oculus groups
# (for leaps)

# leaps together
ggplot(likert_scores %>% filter(Interface != "Oculus", Oculus_Group != "NA"), aes(question,
    score, fill = Oculus_Group)) + geom_boxplot() + # facet_grid(.~Oculus_Group)+
labs(title = "all questions (both Leaps combined)")  #+coord_flip()
```




```r
# leaps separate -- seems that only HHI Leap is different
ggplot(likert_scores %>% filter(Interface != "Oculus", Oculus_Group != "NA"), aes(question,
    score, fill = Oculus_Group)) + geom_boxplot() + scale_fill_manual(values = c("yellow3",
    "violet")) + facet_grid(. ~ Interface) + coord_flip() + labs(title = "Oculus order: HHI Leap vs. B_L
    caption = "Boxplots, medians and IQR with whiskers at 1.5*IQR")
```




```
ggsave("oculus_order_plot.jpg", width = 10, height = 7)

# just HHI (close-up of above) these have higher medians for Oculus_last: tiring,
# satisfaction, recommend, gripping, agency, comfortable
ggplot(likert_scores %>% filter(Interface == "HHI_Leap", Oculus_Group != "NA") %>%
    group_by(question), aes(question, score, fill = Oculus_Group)) + geom_boxplot() +
    # facet_grid(.~Interface)+
labs(title = "Oculus order - all subjective questions (HHI Leap)")  #+coord_flip()
```



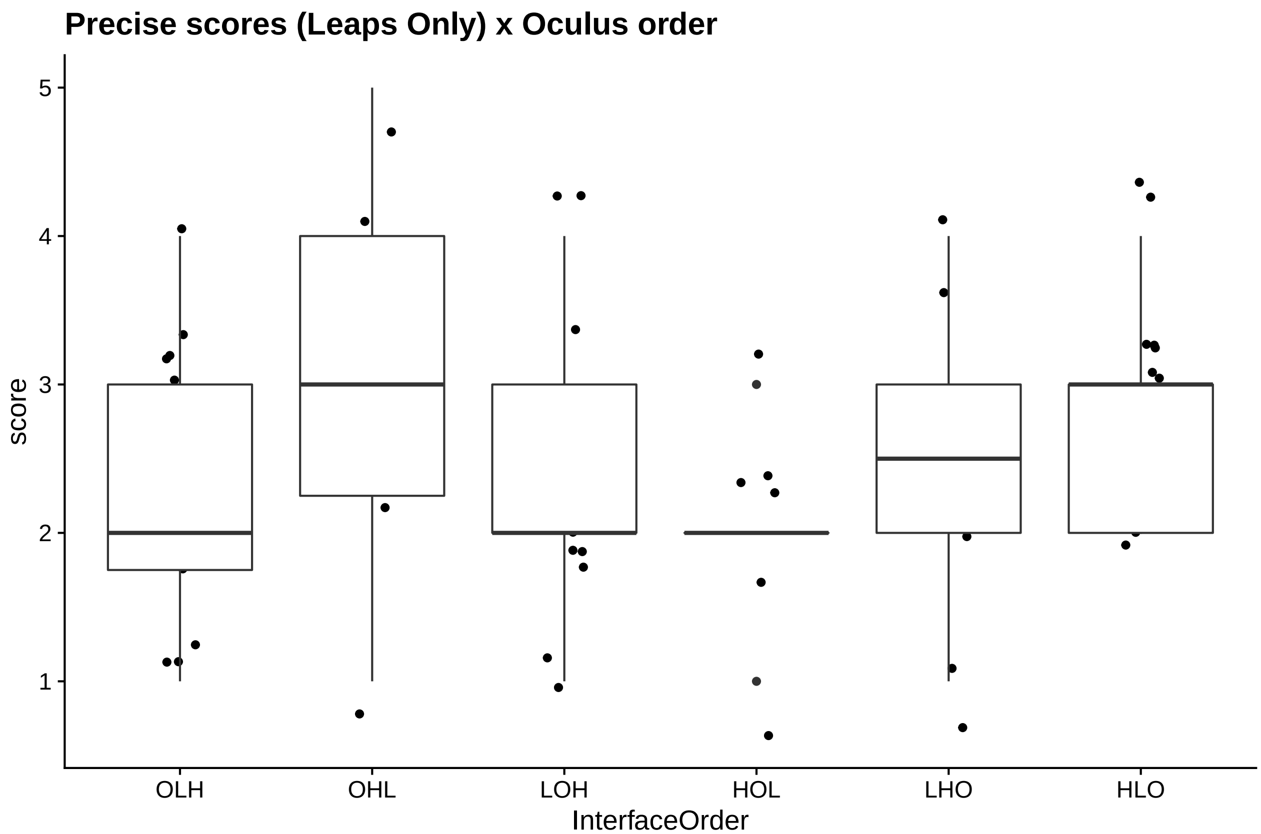

```
# looks like there might be an effect purely on the Oculus. This should have
# manifested itself as an interaction in the ANOVA, but perhaps ANOVA was not
# powerful enough to detect it. t-tests would have the same problem, I suppose.

likert_scores %>% filter(Interface == "HHI_Leap", question == "agency") %>% wilcox_test(score ~
    Oculus_Group)
```

```
## # A tibble: 1 x 9
##   .y.   group1       group2         n1    n2 statistic     p p.adj p.adj.signif
## * <chr> <chr>        <chr>       <int> <int>     <dbl> <dbl> <dbl> <chr>
## 1 score Oculus_first Oculus_last    11    12        37 0.066 0.066 ns
```

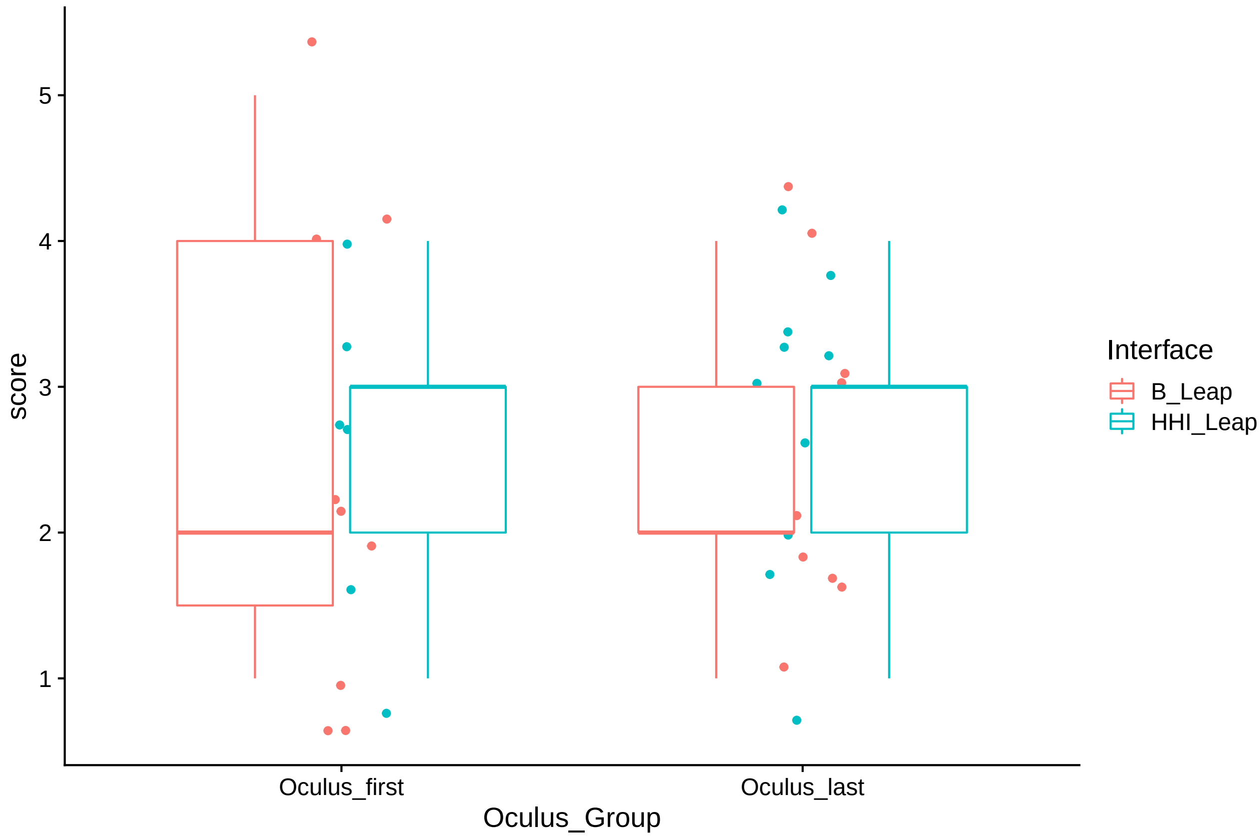
```
# t-test is significant for agency
likert_scores %>% filter(Interface == "HHI_Leap", question == "agency") %>% t_test(score ~
    Oculus_Group)
```

```
## # A tibble: 1 x 10
##   .y.   group1     group2      n1    n2 statistic    df     p p.adj p.adj.signif
## * <chr> <chr>      <chr>    <int> <int>     <dbl> <dbl> <dbl> <dbl> <chr>
## 1 score Oculus_fi~ Oculus_~    11    12     -2.39  13.5 0.032 0.032 *
```

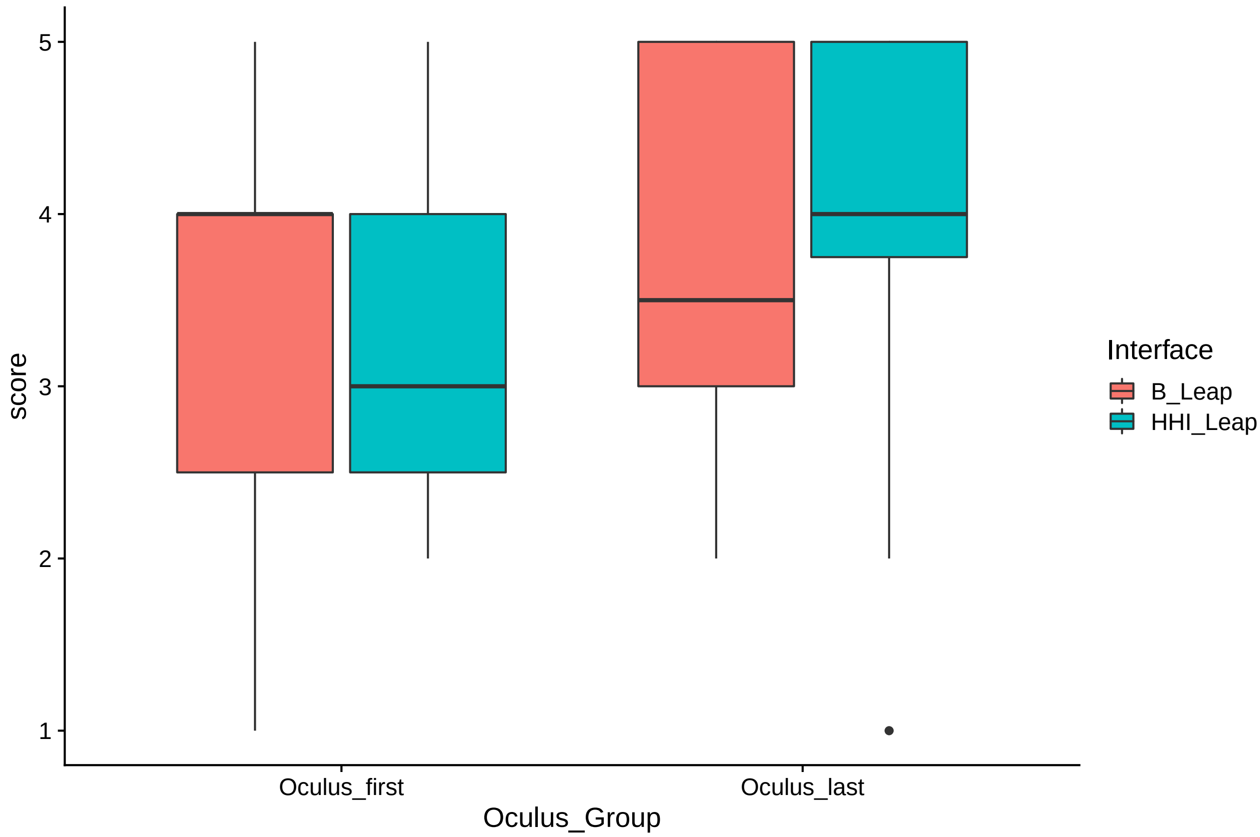
```
ggplot(likert_scores %>% filter(Interface != "Oculus", question == "precise") %>%
    mutate(InterfaceOrder = factor(InterfaceOrder, levels = c("OLH", "OHL", "LOH",
        "HOL", "LHO", "HLO")), Interface = factor(Interface)), aes(InterfaceOrder,
    score)) + geom_point(position = position_jitter(width = 0.1)) + geom_boxplot() +
    labs(title = "Precise scores (Leaps Only) x Oculus order")
```



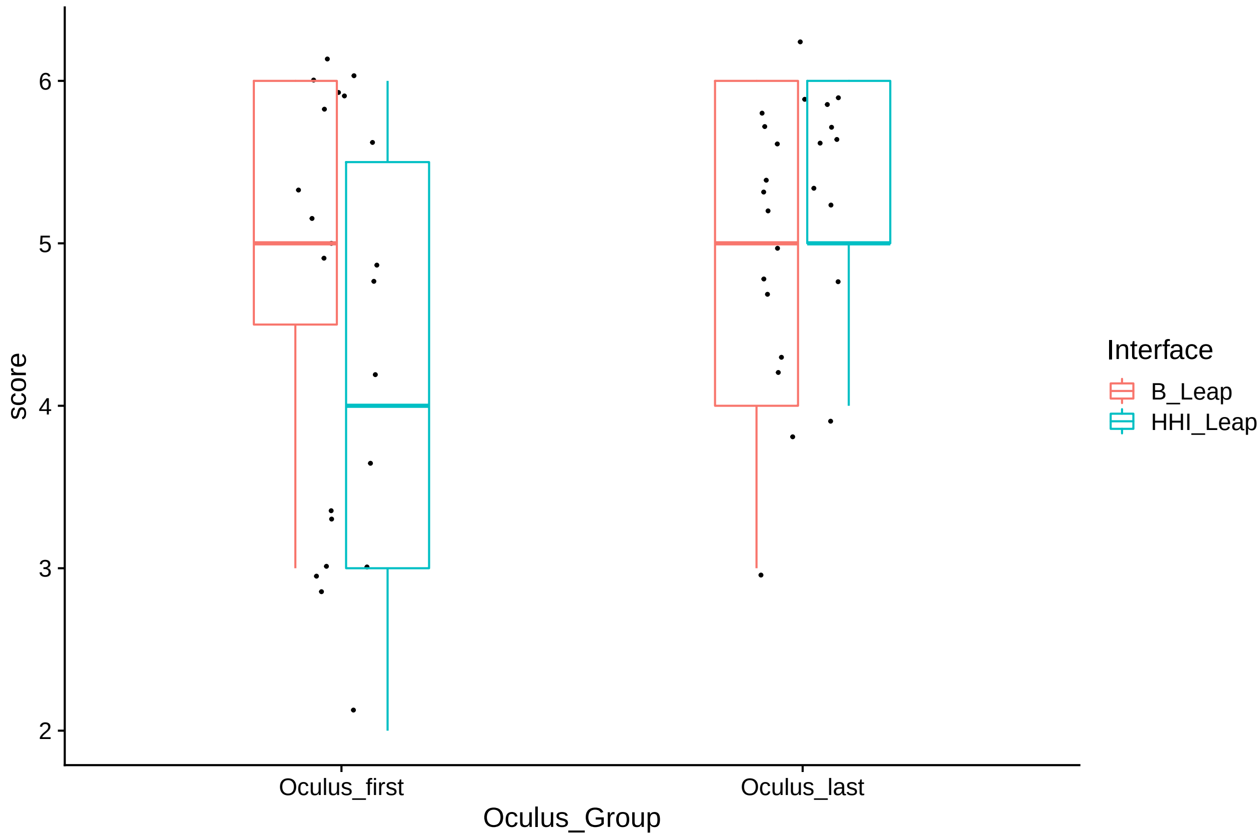

```
ggplot(likert_scores %>% filter(Interface != "Oculus", Oculus_Group != "NA", question ==
    "precise") %>% mutate(Interface = factor(Interface)), aes(Oculus_Group, score,
    color = Interface)) + geom_point(position = position_jitter(width = 0.1)) + geom_boxplot()
```



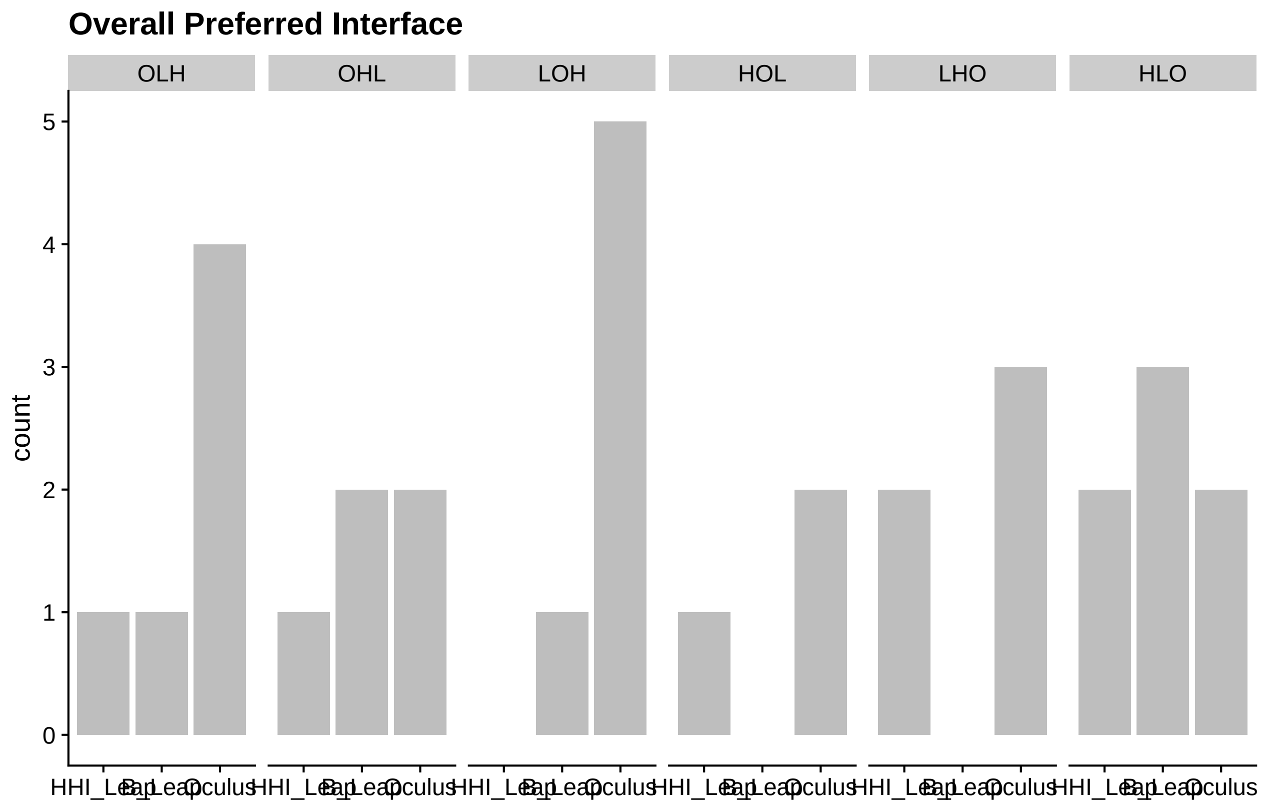

```r
ggplot(likert_scores %>% filter(Interface != "Oculus", Oculus_Group != "NA", question ==
    "recommend") %>% mutate(Interface = factor(Interface)), aes(Oculus_Group, score,
    fill = Interface)) + # geom_point(position=position_jitter(width=.1))+
geom_boxplot()
```



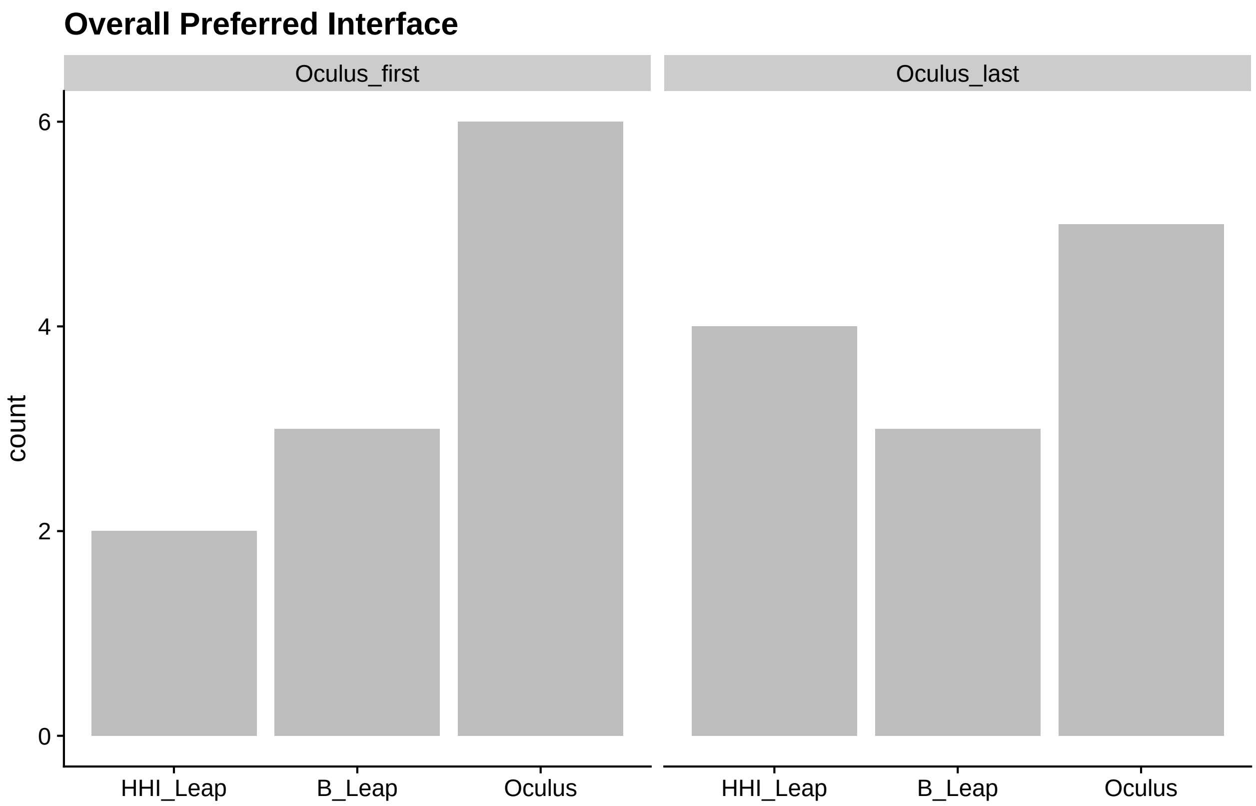

```
ggplot(likert_scores %>% filter(Interface != "Oculus", Oculus_Group != "NA", question ==
    "agency") %>% mutate(Interface = factor(Interface)), aes(Oculus_Group, score,
    color = Interface)) + geom_point(position = position_jitter(width = 0.1), color = "black",
    size = 0.5) + geom_boxplot(aes(middle = mean(score)), width = 0.4, fill = "transparent")
```




```r
# preferred condition
subject_data_all_wide <- subject_data_all_wide %>% mutate(InterfaceOrder = factor(InterfaceOrder,
    levels = c("OLH", "OHL", "LOH", "HOL", "LHO", "HLO")))

ggplot(subject_data_all_wide, aes(factor(PrefCondition, levels = c("HHI_Leap", "B_Leap",
    "Oculus")))) + geom_bar(fill = "grey") + scale_fill_brewer(palette = "Set2") +
    facet_grid(. ~ InterfaceOrder) + labs(x = "", title = "Overall Preferred Interface")
```




```
ggplot(subject_data_all_long %>% filter(Oculus_Group != "NA") %>% select(id, PrefCondition,
    Oculus_Group) %>% distinct(.), aes(factor(PrefCondition, levels = c("HHI_Leap",
    "B_Leap", "Oculus")))) + geom_bar(fill = "grey") + scale_fill_brewer(palette = "Set2") +
    facet_grid(. ~ Oculus_Group) + labs(x = "", title = "Overall Preferred Interface")
```



All tables

```r
# demographics
cat("Demographics")
```

```
## Demographics
```

```r
demographics
```

```
## $descriptives
## # A tibble: 6 x 8
##   variable              n   mean    sd   max   min median   iqr
##   <chr>             <dbl>  <dbl> <dbl> <dbl> <dbl>  <dbl> <dbl>
## 1 Age                  32   27.8  3.37    36    22   27.5  5.25
## 2 Arm                  31   78.2  5.74    89    63   79    6
## 3 Height               32  178.   8.88   194   159  179   12
## 4 SkillController      32    3    1.05     5     2    3    2
## 5 SkillGames           32    3.72  1.35     5     1    4    2.25
## 6 SkillVR              32    2.25  1.14     5     1    2    2
##
## $gender
##
##   Male    N/A Female
##     23      1      8
```

```
##
## $handedness
##
##  Left Right
##     1    31
```

```r
# anovas - Interface level
cat("\nDistance")
```

```
##
## Distance
```

```r
anova.Interface.distance
```

```
##      Effect DFn DFd      F        p p<.05   pes
## 1 Interface   2  62 41.711 3.33e-12     * 0.574
```

```r
cat("\nTotal time")
```

```
##
## Total time
```

```r
print(anova.Interface.totaltime)
```

```
##      Effect DFn DFd      F        p p<.05   pes
## 1 Interface   2  62 36.619 3.16e-11     * 0.542
```

```r
cat("\nGrab time")
```

```
##
## Grab time
```

```r
print(anova.Interface.grabtime)
```

```
##      Effect DFn DFd      F        p p<.05   pes
## 1 Interface   2  62 29.057 1.25e-09     * 0.484
```

```r
cat("\nRelease time")
```

```
##
## Release time
```

```r
print(anova.Interface.releasetime)
```

```
##      Effect DFn DFd      F        p p<.05   pes
## 1 Interface   2  62 30.342 6.48e-10     * 0.495
```

```r
cat("\n")
```

Output to file

```r
sink("results_tables.csv")

cat("\nDemographics\n")
```

```
##
## Demographics
```

```r
write.csv(demographics$descriptives)
```

```
## "","variable","n","mean","sd","max","min","median","iqr"
## "1","Age",32,27.812,3.374,36,22,27.5,5.25
## "2","Arm",31,78.194,5.735,89,63,79,6
## "3","Height",32,177.75,8.875,194,159,179,12
## "4","SkillController",32,3,1.047,5,2,3,2
## "5","SkillGames",32,3.719,1.35,5,1,4,2.25
## "6","SkillVR",32,2.25,1.136,5,1,2,2
```

```r
cat("\nGender")
```

```
##
## Gender
```

```r
write.csv(demographics$gender)
```

```
## "","Var1","Freq"
## "1","Male",23
## "2","N/A",1
## "3","Female",8
```

```r
cat("\nHandedness")
```

```
##
## Handedness
```

```r
write.csv(demographics$handedness)
```

```
## "","Var1","Freq"
## "1","Left",1
## "2","Right",31
```

```r
cat("\nExperience\n")
```

```
##
## Experience
```

```r
write.csv(exp_counts)
```

```
## "","Response","controller_count","vr_count","allgames_count"
## "1","1",0,10,2
## "2","2",13,10,6
## "3","3",10,7,4
## "4","4",5,4,7
## "5","5",4,1,13
```

```r
cat("\nInterface Order")
```

```
##
## Interface Order
```

```r
table(Interface_order$InterfaceOrder)
```

```
##
## LHO LOH HLO HOL OLH OHL
##   5   6   7   3   6   5
```

```r
cat("\nLeap groups")
```

```
##
## Leap groups
```

```r
table(subject_data_all_wide$Leap_Group)
```

```
##
##   B_Leap_first HHI_Leap_first
##             17             15
```

```r
cat("\nPerformance measures\n")
```

```
##
## Performance measures
```

```r
cat("\nAccuracy\n")
```

```
##
## Accuracy
```

```r
cat("\nby Interface")
```

```
##
## by Interface
```

```r
cat("\nDescriptives")
```

```
##
## Descriptives
```

```r
write.csv(descriptives.Interface.distance)
```

```
## "","Interface","variable","mean","sd","min","max","iqr"
## "1","B_Leap","Distance",0.015,0.005,0.004,0.025,0.007
## "2","HHI_Leap","Distance",0.016,0.008,0.003,0.031,0.009
## "3","Oculus","Distance",0.007,0.004,0.002,0.016,0.005
```

```r
cat("\nANOVA")
```

```
##
## ANOVA
```

```r
write.csv(anova.Interface.distance)
```

```
## "","Effect","DFn","DFd","F","p","p<.05","pes"
## "1","Interface",2,62,41.711,3.33e-12,"*",0.574
```

```r
cat("\nt-tests")
```

```
##
## t-tests
```

```r
write.csv(ttest.Interface.distance)
```

```
## "","group1","group2","statistic","df","p","p.adj","p.adj.signif","effsize","magnitude"
## "1","B_Leap","HHI_Leap",-0.293361476687807,31,0.771,0.771,"",0.0519,"negligible"
## "2","HHI_Leap","Oculus",7.12684351797544,31,0,0,"*** ",1.2599,"large"
```

```r
cat("\nby Cube Size and Interface")
```

```
##
## by Cube Size and Interface
```

```r
cat("\nDescriptives")
```

```
##
## Descriptives
```

```r
write.csv(descriptives.Interface.cubesize.distance)
```

```
## "","Interface","Cube_Size","variable","mean","sd","min","max","iqr"
## "1","B_Leap","Small","Distance",0.014,0.007,0.002,0.031,0.009
## "2","B_Leap","Medium","Distance",0.015,0.007,0.003,0.034,0.011
## "3","B_Leap","Large","Distance",0.017,0.007,0.007,0.033,0.01
## "4","HHI_Leap","Small","Distance",0.015,0.01,0.002,0.053,0.008
## "5","HHI_Leap","Medium","Distance",0.015,0.009,0.003,0.046,0.01
## "6","HHI_Leap","Large","Distance",0.018,0.013,0.003,0.061,0.014
## "7","Oculus","Small","Distance",0.006,0.004,0.002,0.018,0.005
## "8","Oculus","Medium","Distance",0.007,0.005,0.002,0.023,0.004
## "9","Oculus","Large","Distance",0.008,0.004,0.002,0.02,0.005
```

```r
cat("\nANOVA")
```

```
##
## ANOVA
```

```r
write.csv(anova.Interface.cubesize.distance)
```

```
## "","Effect","DFn","DFd","F","p","p<.05","pes"
## "1","Interface",2,62,41.437,0,"*",0.572
## "2","Cube_Size",2,62,4.055,0.022,"*",0.116
## "3","Interface:Cube_Size",4,124,0.237,0.917,"",0.008
```

```r
cat("\nt-tests")
```

```
##
## t-tests
```

```r
write.csv(ttest.Interface.cubesize.distance)
```

```
## "","Cube_Size","group1","group2","statistic","df","p","p.adj","p.adj.sig","eff","mag"
## "1","Small","B_Leap","HHI_Leap",-0.638,31,0.528,1,"",0.035,"negligible"
## "2","Small","HHI_Leap","Oculus",5.54,31,0,0,"*** ",0.8774,"large"
## "3","Medium","B_Leap","HHI_Leap",0.355,31,0.725,1,"",0.035,"negligible"
## "4","Medium","HHI_Leap","Oculus",4.755,31,0,2e-04,"***",0.8774,"large"
## "5","Large","B_Leap","HHI_Leap",-0.283,31,0.779,1,"",0.035,"negligible"
## "6","Large","HHI_Leap","Oculus",4.659,31,1e-04,2e-04,"***",0.8774,"large"
```

```r
cat("\nTotal time")
```

```
##
## Total time
```

```r
cat("\nby Interface")
```

```
##
## by Interface
```

```r
cat("\nDescriptives")
```

```
##
## Descriptives
```

```r
write.csv(descriptives.Interface.totaltime)
```

```
## "","Interface","variable","mean","sd","min","max","iqr"
## "1","B_Leap","totaltime",3.566,1.569,1.808,9.335,1.317
## "2","HHI_Leap","totaltime",3.515,1.347,1.794,9.101,1.307
## "3","Oculus","totaltime",2.271,0.753,1.359,4.438,0.817
```

```r
cat("\nANOVA")
```

```
##
## ANOVA
```

```r
write.csv(anova.Interface.totaltime)
```

```
## "","Effect","DFn","DFd","F","p","p<.05","pes"
## "1","Interface",2,62,36.619,3.16e-11,"*",0.542
```

```r
cat("\nt-tests")
```

```
##
## t-tests
```

```r
write.csv(ttest.Interface.totaltime)
```

```
## "","group1","group2","statistic","df","p","p.adj","p.adj.signif","effsize","magnitude"
## "1","B_Leap","HHI_Leap",0.292633640776283,31,0.772,0.772,"",0.0517,"negligible"
## "2","HHI_Leap","Oculus",7.88422920826133,31,0,0,"*** ",1.3937,"large"
```

```r
cat("\nby Cube Size and Interface")
```

```
##
## by Cube Size and Interface
```

```r
cat("\nDescriptives")
```

```
##
## Descriptives
```

```r
write.csv(descriptives.Interface.cubesize.totaltime)
```

```
## "","Interface","Cube_Size","variable","mean","sd","min","max","iqr"
## "1","B_Leap","Small","totaltime",3.636,1.625,1.867,8.225,1.744
## "2","B_Leap","Medium","totaltime",3.543,1.515,1.658,8.059,1.548
## "3","B_Leap","Large","totaltime",3.564,1.839,1.509,11.306,1.519
## "4","HHI_Leap","Small","totaltime",4.118,2.046,1.937,13.322,2.037
## "5","HHI_Leap","Medium","totaltime",3.308,1.273,1.495,7.703,1.218
## "6","HHI_Leap","Large","totaltime",3.141,1.13,1.759,7.403,1.163
## "7","Oculus","Small","totaltime",2.503,0.911,1.45,5.453,0.899
## "8","Oculus","Medium","totaltime",2.205,0.701,1.347,4.08,0.827
## "9","Oculus","Large","totaltime",2.109,0.724,1.279,4.013,0.808
```

```r
cat("\nANOVA")
```

```
##
## ANOVA
```

```r
write.csv(anova.Interface.cubesize.totaltime)
```

```
## "","Effect","DFn","DFd","F","p","p<.05","pes"
## "1","Interface",2,62,35.794,0,"*",0.536
## "2","Cube_Size",2,62,14.115,0,"*",0.313
## "3","Interface:Cube_Size",4,124,5.232,6e-04,"*",0.144
```

```r
cat("\nt-tests")
```

```
##
## t-tests
```

```r
write.csv(ttest.Interface.cubesize.totaltime)
```

```
## "","Cube_Size","group1","group2","statistic","df","p","p.adj","p.adj.sig","eff","mag"
## "1","Small","B_Leap","HHI_Leap",-1.666,31,0.106,0.212,"",0.0438,"negligible"
## "2","Small","HHI_Leap","Oculus",5.89,31,0,0,"*** ",1.1033,"large"
## "3","Medium","B_Leap","HHI_Leap",1.502,31,0.143,0.212,"",0.0438,"negligible"
## "4","Medium","HHI_Leap","Oculus",7.679,31,0,0,"*** ",1.1033,"large"
## "5","Large","B_Leap","HHI_Leap",1.881,31,0.069,0.207,"",0.0438,"negligible"
## "6","Large","HHI_Leap","Oculus",7.206,31,0,0,"*** ",1.1033,"large"
```

```r
cat("\nGrab time")
```

```
##
## Grab time
```

```r
cat("\nby Interface")
```

```
##
## by Interface
```

```r
cat("\nDescriptives")
```

```
##
## Descriptives
```

```r
write.csv(descriptives.Interface.grabtime)
```

```
## "","Interface","variable","mean","sd","min","max","iqr"
## "1","B_Leap","grabtime",1.354,0.41,0.803,2.465,0.369
## "2","HHI_Leap","grabtime",1.573,0.652,0.86,4.483,0.606
## "3","Oculus","grabtime",0.946,0.248,0.697,1.771,0.274
```

```r
cat("\nANOVA")
```

```
##
## ANOVA
```

```r
write.csv(anova.Interface.grabtime)
```

```
## "","Effect","DFn","DFd","F","p","p<.05","pes"
## "1","Interface",2,62,29.057,1.25e-09,"*",0.484
```

```r
cat("\nt-test")
```

```
##
## t-test
```

```r
write.csv(ttest.Interface.grabtime)
```

```
## "","group1","group2","statistic","df","p","p.adj","p.adj.signif","effsize","magnitude"
## "1","B_Leap","HHI_Leap",-2.25290643213319,31,0.032,0.032,"*",0.3983,"small"
## "2","HHI_Leap","Oculus",6.93934039986717,31,0,0,"*** ",1.2267,"large"
```

```r
cat("\nby Cube Size and Interface")
```

```
##
## by Cube Size and Interface
```

```r
cat("\nDescriptives")
```

```
##
## Descriptives
```

```r
write.csv(descriptives.Interface.cubesize.grabtime)
```

```
## "","Interface","Cube_Size","variable","mean","sd","min","max","iqr"
## "1","B_Leap","Small","grabtime",1.739,0.917,0.783,4.803,0.649
## "2","B_Leap","Medium","grabtime",1.292,0.357,0.786,2.011,0.439
## "3","B_Leap","Large","grabtime",1.11,0.304,0.704,1.992,0.284
## "4","HHI_Leap","Small","grabtime",2.062,1.493,0.963,9.243,1.11
## "5","HHI_Leap","Medium","grabtime",1.471,0.579,0.771,3.454,0.607
## "6","HHI_Leap","Large","grabtime",1.226,0.287,0.844,2.077,0.434
## "7","Oculus","Small","grabtime",1.048,0.349,0.735,2.48,0.306
## "8","Oculus","Medium","grabtime",0.92,0.253,0.653,1.818,0.276
## "9","Oculus","Large","grabtime",0.874,0.214,0.622,1.532,0.279
```

```r
cat("\nANOVA")
```

```
##
## ANOVA
```

```r
write.csv(anova.Interface.cubesize.grabtime)
```

```
## "","Effect","DFn","DFd","F","p","p<.05","pes"
## "1","Interface",2,62,23.828,0,"*",0.435
## "2","Cube_Size",2,62,17.216,0,"*",0.357
## "3","Interface:Cube_Size",4,124,5.27,6e-04,"*",0.145
```

```r
cat("\nt-test")
```

```
##
## t-test
```

```r
write.csv(ttest.Interface.cubesize.grabtime)
```

```
## "","Cube_Size","group1","group2","statistic","df","p","p.adj","p.adj.sig","eff","mag"
## "1","Small","B_Leap","HHI_Leap",-1.433,31,0.162,0.184,"",0.2491,"small"
## "2","Small","HHI_Leap","Oculus",4.576,31,1e-04,3e-04,"***",0.7736,"moderate"
## "3","Medium","B_Leap","HHI_Leap",-1.738,31,0.092,0.184,"",0.2491,"small"
## "4","Medium","HHI_Leap","Oculus",7.094,31,0,0,"*** ",0.7736,"moderate"
## "5","Large","B_Leap","HHI_Leap",-2.006,31,0.054,0.162,"",0.2491,"small"
## "6","Large","HHI_Leap","Oculus",6.73,31,0,0,"*** ",0.7736,"moderate"
```

```r
cat("\nRelease time")
```

```
##
## Release time
```

```r
cat("\nby Interface")
```

```
##
## by Interface
```

```r
cat("\nDescriptives")
```

```
##
## Descriptives
```

```r
write.csv(descriptives.Interface.releasetime)
```

```
## "","Interface","variable","mean","sd","min","max","iqr"
## "1","B_Leap","releasetime",2.212,1.236,0.792,7.042,1.109
## "2","HHI_Leap","releasetime",1.942,0.856,0.842,4.618,0.855
## "3","Oculus","releasetime",1.324,0.575,0.582,3.122,0.593
```

```r
cat("\nANOVA")
```

```
##
## ANOVA
```

```r
write.csv(anova.Interface.releasetime)
```

```
## "","Effect","DFn","DFd","F","p","p<.05","pes"
## "1","Interface",2,62,30.342,6.48e-10,"*",0.495
```

```r
cat("\nt-test")
```

```
##
## t-test
```

```r
write.csv(ttest.Interface.releasetime)
```

```
## "","group1","group2","statistic","df","p","p.adj","p.adj.signif","effsize","magnitude"
## "1","B_Leap","HHI_Leap",2.31713221956506,31,0.027,0.027,"*",0.4096,"small"
## "2","HHI_Leap","Oculus",6.70134523648657,31,0,0,"*** ",1.1846,"large"
```

```r
cat("\nby Cube Size and Interface")
```

```
##
## by Cube Size and Interface
```

```r
cat("\nDescriptives")
```

```
##
## Descriptives
```

```r
write.csv(descriptives.Interface.cubesize.releasetime)
```

```
## "","Interface","Cube_Size","variable","mean","sd","min","max","iqr"
## "1","B_Leap","Small","releasetime",1.898,0.913,0.824,5.056,1.203
## "2","B_Leap","Medium","releasetime",2.252,1.277,0.839,6.378,1.069
## "3","B_Leap","Large","releasetime",2.454,1.621,0.728,9.314,1.096
## "4","HHI_Leap","Small","releasetime",2.057,0.869,0.87,4.079,1.127
## "5","HHI_Leap","Medium","releasetime",1.837,0.883,0.608,4.791,0.761
## "6","HHI_Leap","Large","releasetime",1.915,0.928,0.861,5.327,1.033
## "7","Oculus","Small","releasetime",1.455,0.65,0.62,3.662,0.639
## "8","Oculus","Medium","releasetime",1.285,0.538,0.575,2.991,0.741
## "9","Oculus","Large","releasetime",1.236,0.578,0.552,2.798,0.544
```

```
cat("\nANOVA")
```

```
##
## ANOVA
```

```
write.csv(anova.Interface.cubesize.releasetime)
```

```
## "","Effect","DFn","DFd","F","p","p<.05","pes"
## "1","Interface",2,62,32.023,0,"*",0.508
## "2","Cube_Size",2,62,0.884,0.418,"",0.028
## "3","Interface:Cube_Size",4,124,9.721,0,"*",0.239
```

```
cat("\nt-test")
```

```
##
## t-test
```

```
write.csv(ttest.Interface.cubesize.releasetime)
```

```
## "","Cube_Size","group1","group2","statistic","df","p","p.adj","p.adj.sig","eff","mag"
## "1","Small","B_Leap","HHI_Leap",-1.506,31,0.142,0.142,"",0.3089,"small"
## "2","Small","HHI_Leap","Oculus",5.497,31,0,0,"*** ",1.0274,"large"
## "3","Medium","B_Leap","HHI_Leap",3.417,31,0.002,0.006,"**",0.3089,"small"
## "4","Medium","HHI_Leap","Oculus",5.568,31,0,0,"*** ",1.0274,"large"
## "5","Large","B_Leap","HHI_Leap",2.853,31,0.008,0.016,"*",0.3089,"small"
## "6","Large","HHI_Leap","Oculus",6.255,31,0,0,"*** ",1.0274,"large"
```

```
cat("\nAccidental drops")
```

```
##
## Accidental drops
```

```
cat("\nby Interface")
```

```
##
## by Interface
```

```r
cat("\nDescriptives")
```

```
##
## Descriptives
```

```r
write.csv(descriptives.Interface.drops)
```

```
## "","Interface","variable","mean","sd","median","mad","min","max","iqr"
## "1","B_Leap","Drop_Count",4.75,2.676,5,2.965,0,10,5
## "2","HHI_Leap","Drop_Count",2.406,2.092,2,2.965,0,8,3
## "3","Oculus","Drop_Count",0.125,0.336,0,0,0,1,0
```

```r
cat("\nANOVA")
```

```
##
## ANOVA
```

```r
write.csv(anova.Interface.drops)
```

```
## "","Effect","DFn","DFd","F","p","p<.05","pes"
## "1","Interface",2,62,44.383,1.09e-12,"*",0.589
```

```r
cat("\nt-test")
```

```
##
## t-test
```

```r
write.csv(ttest.Interface.drops)
```

```
## "","group1","group2","statistic","df","p","p.adj","p.adj.signif","effsize","magnitude"
## "1","B_Leap","HHI_Leap",3.86269651968454,31,5e-04,5e-04,"***",0.6828,"moderate"
## "2","HHI_Leap","Oculus",6.24269329915146,31,0,0,"*** ",1.1036,"large"
```

```r
cat("\nby Cube Size and Interface")
```

```
##
## by Cube Size and Interface
```

```r
cat("\nDescriptives")
```

```
##
## Descriptives
```

```r
write.csv(descriptives.Interface.cubesize.drops)
```

```
## "","Interface","Cube_Size","variable","mean","sd","min","max","iqr"
## "1","B_Leap","Small","Drop_Count",2.281,1.651,0,6,2
## "2","B_Leap","Medium","Drop_Count",1.344,1.125,0,4,1.25
## "3","B_Leap","Large","Drop_Count",1.125,1.008,0,4,2
## "4","HHI_Leap","Small","Drop_Count",0.719,0.958,0,3,1
## "5","HHI_Leap","Medium","Drop_Count",0.688,1.061,0,4,1
## "6","HHI_Leap","Large","Drop_Count",1,1.164,0,4,1.25
## "7","Oculus","Small","Drop_Count",0.125,0.336,0,1,0
## "8","Oculus","Medium","Drop_Count",0,0,0,0,0
## "9","Oculus","Large","Drop_Count",0,0,0,0,0
```

```r
cat("\nANOVA")
```

```
##
## ANOVA
```

```r
write.csv(anova.Interface.cubesize.drops)
```

```
## "","Effect","DFn","DFd","F","p","p<.05","pes"
## "1","Interface",2,62,44.383,1.09e-12,"*",0.589
## "2","Cube_Size",2,62,5.369,0.007,"*",0.148
## "3","Interface:Cube_Size",4,124,5.842,0.000243,"*",0.159
```

```r
cat("\nt-test")
```

```
##
## t-test
```

```r
write.csv(ttest.Interface.cubesize.drops)
```

```
## "","Cube_Size",".y.","group1","group2","n1","n2","statistic","df","p","p.adj","p.adj.signif","effsiz
## "1","Small","Drop_Count","B_Leap","HHI_Leap",32,32,4.31830270011461,31,0.00015,0.00075,"***",0.43081
## "2","Small","Drop_Count","HHI_Leap","Oculus",32,32,3.6875212378654,31,0.000863,0.003452,"**",0.72162
## "3","Medium","Drop_Count","B_Leap","HHI_Leap",32,32,2.21319776722762,31,0.034,0.068,"",0.43081615077
## "4","Medium","Drop_Count","HHI_Leap","Oculus",32,32,3.66666666666667,31,0.000914,0.003452,"**",0.721
## "5","Large","Drop_Count","B_Leap","HHI_Leap",32,32,0.502028405894729,31,0.619,0.619,"",0.43081615077
## "6","Large","Drop_Count","HHI_Leap","Oculus",32,32,4.85994317035165,31,3.21e-05,0.0001926,"***",0.72
```

```r
cat("\nDifference: HHI Leap to Oculus\n")
```

```
##
## Difference: HHI Leap to Oculus
```

```r
write.csv(oculus_diffs)
```

```
## "","Difference","Type","cohen's d"
## "Accuracy",2.143,"Ratio",NA
## "Grab time",1.431,"Ratio",NA
## "Release time",1.67,"Ratio",NA
## "Total time",1.571,"Ratio",NA
## "Accidental drops",4.625,"Mean difference",NA
```

```r
cat("\n\nSubjective\n")
```

```
##
##
## Subjective
```

```r
cat("\n(by Interface)")
```

```
##
## (by Interface)
```

```r
cat("\nSUS")
```

```
##
## SUS
```

```r
cat("\nDescriptives")
```

```
##
## Descriptives
```

```r
write.csv(descriptives.sus)
```

```
## "","Interface","variable","n","mean","sd","min","max","iqr"
## "1","B_Leap","SUS",32,69.922,15.574,32.5,100,25
## "2","HHI_Leap","SUS",32,70.469,18.842,30,97.5,26.25
## "3","Oculus","SUS",32,82.344,14.742,25,100,20
```

```r
cat("\nANOVA")
```

```
##
## ANOVA
```

```r
write.csv(anova.SUS)
```

```
## "","Effect","DFn","DFd","F","p","p<.05","pes"
## "1","Interface",2,62,7.132,0.002,"*",0.187
```

```r
cat("\nt-test")
```

```
##
## t-test
```

```r
write.csv(ttest.SUS)
```

```
## "",".y.","group1","group2","n1","n2","statistic","df","p","p.adj","p.adj.signif","effsize","magnitud
## "1","SUS","B_Leap","HHI_Leap",32,32,-0.187136775943861,31,0.853,0.853,"ns",0.0330814208198229,"neglig
## "2","SUS","HHI_Leap","Oculus",32,32,-2.68874100005461,31,0.011,0.022,"*",0.475306748498229,"small"
```

```r
cat("\n\n5-pt Likert Questions")
```

```
##
##
## 5-pt Likert Questions
```

```r
cat("\nDescriptives")
```

```
##
## Descriptives
```

```r
write.csv(descriptives.subjective5pt)
```

```
## "","interface","question","n","mean","sd","median","iqr","min","max","SDs_from_mid"
## "1","B_Leap","comfortable",32,3.438,0.982,3.5,1,1,5,0.430116
## "2","HHI_Leap","comfortable",32,3.375,1.1,3.5,1.25,1,5,0.4125
## "3","Oculus","comfortable",32,4.156,0.92,4,1,2,5,1.06352
## "4","B_Leap","gripping",32,2.875,1.129,3,2,1,5,-0.141125
## "5","HHI_Leap","gripping",32,3.094,1.174,3,2,1,5,0.110356
## "6","Oculus","gripping",32,4.406,1.073,5,1,1,5,1.508638
## "7","B_Leap","intuitive",32,4.094,0.818,4,1,2,5,0.894892
## "8","HHI_Leap","intuitive",32,4.031,1.031,4,2,2,5,1.062961
## "9","Oculus","intuitive",32,4.156,0.884,4,1.25,2,5,1.021904
## "10","B_Leap","natural",32,2.75,1.078,3,1.25,1,5,-0.2695
## "11","HHI_Leap","natural",32,2.781,0.975,3,2,1,4,-0.213525
## "12","Oculus","natural",32,3.188,1.12,3,2,1,5,0.21056
## "13","B_Leap","precise",32,2.438,1.076,2,1,1,5,-0.604712
## "14","HHI_Leap","precise",32,2.594,0.875,3,1,1,4,-0.35525
## "15","Oculus","precise",32,4.156,0.92,4,1,2,5,1.06352
## "16","B_Leap","recommend",32,3.344,1.096,3,1,1,5,0.377024
## "17","HHI_Leap","recommend",32,3.375,1.238,3.5,2,1,5,0.46425
## "18","Oculus","recommend",32,4.094,0.928,4,1.25,2,5,1.015232
## "19","B_Leap","releasing",32,2.594,1.214,2,1,1,5,-0.492884
## "20","HHI_Leap","releasing",32,2.781,1.099,3,2,1,5,-0.240681
## "21","Oculus","releasing",32,4.406,1.073,5,1,1,5,1.508638
## "22","B_Leap","tiring",32,3.938,1.014,4,2,2,5,0.951132
## "23","HHI_Leap","tiring",32,3.906,1.118,4,2,2,5,1.012908
## "24","Oculus","tiring",32,3.938,1.216,4,1.25,1,5,1.140608
## "25","all","comfortable",96,3.656,1.055,4,1,1,5,0.69208
## "26","all","gripping",96,3.458,1.305,4,3,1,5,0.59769
## "27","all","intuitive",96,4.094,0.907,4,1,2,5,0.992258
## "28","all","natural",96,2.906,1.067,3,2,1,5,-0.100298
## "29","all","precise",96,3.062,1.23,3,2,1,5,0.0762599999999998
## "30","all","recommend",96,3.604,1.138,4,2,1,5,0.687352
## "31","all","releasing",96,3.26,1.386,3,3,1,5,0.36036
## "32","all","tiring",96,3.927,1.107,4,2,1,5,1.026189
## "33","all","all",768,3.496,1.215,4,3,1,5,0.60264
## "34","B_Leap","all",256,3.184,1.192,3,2,1,5,0.219328
## "35","HHI_Leap","all",256,3.242,1.177,3,2,1,5,0.284834
## "36","Oculus","all",256,4.062,1.072,4,1.25,1,5,1.138464
## "37","means","means",24,3.496,0.644,3.407,1.243,2.438,4.406,0.319424
```

```r
cat("\nGrand ANOVA 5pt")
```

```
##
## Grand ANOVA 5pt
```

```r
write.csv(anova.Interface.5ptgrand)
```

```
## "","Effect","DFn","DFd","F","p","p<.05","pes"
## "1","Interface",2,62,15.827,2.8e-06,"*",0.338
```

```r
cat("\nIndividual ANOVAs")
```

```
##
## Individual ANOVAs
```

```r
write.csv(anova.likert %>% filter(question != "agency" & question != "satisfaction"))
```

```
## "","question","Effect","DFn","DFd","F","p","p<.05","pes"
## "1","comfortable","Interface",2,62,7.911,0.000871,"*",0.203
## "2","precise","Interface",2,62,27.921,2.26e-09,"*",0.474
## "3","intuitive","Interface",2,62,0.198,0.821,"",0.006
## "4","tiring","Interface",2,62,0.012,0.989,"",0.000372
## "5","gripping","Interface",2,62,23.42,2.65e-08,"*",0.43
## "6","releasing","Interface",2,62,25.571,7.98e-09,"*",0.452
## "7","natural","Interface",2,62,1.608,0.209,"",0.049
## "8","recommend","Interface",2,62,6.556,0.003,"*",0.175
```

```r
cat("\nGrand t-tests 5pt")
```

```
##
## Grand t-tests 5pt
```

```r
write.csv(ttest.Interface.5ptgrand)
```

```
## "",".y.","group1","group2","n1","n2","statistic","df","p","p.adj","p.adj.signif","effsize","magnitude
## "1","grand_mean","B_Leap","HHI_Leap",32,32,-0.442220442777235,31,0.661,0.661,"ns",0.0781742684667751
## "2","grand_mean","B_Leap","Oculus",32,32,-5.3899623047961,31,7.02e-06,2.106e-05,"****",0.95281972401
## "3","grand_mean","HHI_Leap","Oculus",32,32,-4.12873747030021,31,0.000255,0.00051,"***",0.72986456574
```

```r
cat("\nIndividual t-tests 5pt")
```

```
##
## Individual t-tests 5pt
```

```r
write.csv(t.tests.likert.all.vs.hhi %>% filter(question != "agency" & question !=
    "satisfaction") %>% select(-Interface))
```

```
## "","question",".y.","group1","group2","n1","n2","statistic","df","p","p.adj","p.adj.signif","effsize
## "1","comfortable","score","HHI_Leap","Oculus",32,32,-3.08863005998846,31,0.004,0.012,"*",0.39539967228
## "2","gripping","score","HHI_Leap","Oculus",32,32,-5.21334119748093,31,1.16e-05,4.64e-05,"*** ",0.3953
## "3","precise","score","HHI_Leap","Oculus",32,32,-6.46889884507546,31,3.26e-07,1.956e-06,"*** ",0.3953
## "4","recommend","score","HHI_Leap","Oculus",32,32,-2.65924473839805,31,0.012,0.024,"*",0.39539967228
## "5","releasing","score","HHI_Leap","Oculus",32,32,-6.13927092430481,31,8.26e-07,4.13e-06,"*** ",0.395
## "6","comfortable","score","HHI_Leap","B_Leap",32,32,-0.311734956951542,31,0.757,1,"",0.0448853300824
## "7","gripping","score","HHI_Leap","B_Leap",32,32,0.908841233201343,31,0.37,1,"",0.0448853300824245,"
## "8","intuitive","score","HHI_Leap","B_Leap",32,32,-0.348666929104239,31,0.73,1,"",0.0448853300824245
## "9","natural","score","HHI_Leap","B_Leap",32,32,0.132754095232589,31,0.895,1,"",0.0448853300824245,"
## "10","precise","score","HHI_Leap","B_Leap",32,32,0.595832218818465,31,0.556,1,"",0.0448853300824245,
## "11","recommend","score","HHI_Leap","B_Leap",32,32,0.166443102962189,31,0.869,1,"",0.044885330082424
## "12","releasing","score","HHI_Leap","B_Leap",32,32,0.641013556775943,31,0.526,1,"",0.044885330082424
## "13","tiring","score","HHI_Leap","B_Leap",32,32,-0.182869374687206,31,0.856,1,"",0.0448853300824245,
```

```r
cat("\n\n7-pt Likert Questions")
```

```
##
##
## 7-pt Likert Questions
```

```r
cat("\nDescriptives")
```

```
##
## Descriptives
```

```r
write.csv(descriptives.subjective7pt)
```

```
## "","interface","question","n","mean","sd","median","iqr","min","max","SDs_from_mid"
## "1","B_Leap","agency",32,4.781,1.128,5,2,3,6,0.880968
## "2","HHI_Leap","agency",32,4.719,1.143,5,2,2,6,0.821817
## "3","Oculus","agency",32,4.188,1.655,4,3,1,7,0.31114
## "4","B_Leap","satisfaction",32,4.844,1.081,5,2,2,7,0.912364
## "5","HHI_Leap","satisfaction",32,5,1.047,5,1.25,3,7,1.047
## "6","Oculus","satisfaction",32,5.594,0.875,6,1,3,7,1.39475
## "7","all","agency",96,4.562,1.344,5,3,1,7,0.755328
## "8","all","satisfaction",96,5.146,1.046,5,1,2,7,1.198716
## "9","all","all",192,4.854,1.236,5,2,1,7,1.055544
## "10","B_Leap","all",64,4.812,1.097,5,2,2,7,0.890764
## "11","HHI_Leap","all",64,4.859,1.096,5,2,2,7,0.941464
## "12","Oculus","all",64,4.891,1.492,5,2,1,7,1.329372
## "13","means","means",6,4.854,0.455,4.812,0.227,4.188,5.594,0.38857
```

```r
cat("\nGrand ANOVA 7pt")
```

```
##
## Grand ANOVA 7pt
```

```r
write.csv(anova.Interface.7ptgrand)
```

```
## "","Effect","DFn","DFd","F","p","p<.05","pes"
## "1","Interface",2,62,1.602,0.21,"",0.049
```

```r
cat("\nIndividual ANOVAs")
```

```
##
## Individual ANOVAs
```

```r
write.csv(anova.likert %>% filter(question == "agency" | question == "satisfaction"))
```

```
## "","question","Effect","DFn","DFd","F","p","p<.05","pes"
## "1","agency","Interface",2,62,1.602,0.21,"",0.049
## "2","satisfaction","Interface",2,62,5.901,0.005,"*",0.16
```

```r
cat("\nGrand t-tests 7pt")
```

```
##
## Grand t-tests 7pt
```

```r
write.csv(ttest.Interface.7ptgrand)
```

```
## "",".y.","group1","group2","n1","n2","statistic","df","p","p.adj","p.adj.signif","effsize","magnitude
## "1","grand_mean","B_Leap","HHI_Leap",32,32,0.263346217791439,31,0.794,0.794,"ns",0.046553474100039,"
## "2","grand_mean","B_Leap","Oculus",32,32,1.3872655077504,31,0.175,0.525,"ns",0.245236211959127,"small
## "3","grand_mean","HHI_Leap","Oculus",32,32,1.33127932788822,31,0.193,0.525,"ns",0.235339160100808,"sm
```

```r
cat("\nIndividual t-tests 7pt")
```

```
##
## Individual t-tests 7pt
```

```r
write.csv(t.tests.likert.all.vs.hhi %>% filter(question == "agency" | question ==
    "satisfaction") %>% select(-Interface))
```

```
## "","question",".y.","group1","group2","n1","n2","statistic","df","p","p.adj","p.adj.signif","effsize
## "1","satisfaction","score","HHI_Leap","Oculus",32,32,-2.46150657490488,31,0.02,0.024,"*",0.3953996722
## "2","agency","score","HHI_Leap","B_Leap",32,32,-0.263346217791439,31,0.794,1,"",0.0448853300824245,"
## "3","satisfaction","score","HHI_Leap","B_Leap",32,32,0.775999743669805,31,0.444,1,"",0.044885330082
```

```r
cat("\n\nOverall preference\n")
```

```
##
##
## Overall preference
```

```r
write.csv(preference)
```

```
## "","Var1","Freq"
## "1","B_Leap",7
## "2","Oculus",18
## "3","HHI_Leap",7
```

```r
cat("\nDATA CLEANING\n")
```

```
##
## DATA CLEANING
```

```r
cat("Cut and add to accidental drop count:
- Distance >=0.2
- Distance >= 0.1 & <= 0.2 & TimeFromGrabToGrabLoss < 0.5
- LandOnTable==FALSE\n")
```

```
## Cut and add to accidental drop count:
## - Distance >=0.2
## - Distance >= 0.1 & <= 0.2 & TimeFromGrabToGrabLoss < 0.5
## - LandOnTable==FALSE
```

```r
cat("\nTotal trials before clean: ", length(data_set$id))
```

```
##
## Total trials before clean:  2880
```

```r
cat("\nTotal trials AFTER clean: ", length(unity_data_clean$id))
```

```
##
## Total trials AFTER clean:  2647
```

```r
cat("\n\nACCIDENTAL DROP REPORT")
```

```
##
##
## ACCIDENTAL DROP REPORT
```

```r
cat("\nTotal accidental drops: ", accidental_drops_total)
```

```
##
## Total accidental drops:  233
```

```r
cat("\nManual detection: ", accidental_drops_manual, " (", round(accidental_drops_manual.percent,
    1), "% of original trials)")
```

```
##
## Manual detection:  199  ( 6.9 % of original trials)
```

```r
cat("\nAutomatic detection: ", accidental_drops_auto_detect, " (", round(accidental_drops_auto_detect.pe
    1), "% of original trials)")
```

```
##
## Automatic detection:  202  ( 7 % of original trials)
```

```r
cat("\nAuto, not detected by manual: ", accidental_drops_auto_only)
```

```
##
## Auto, not detected by manual:  34
```

```r
cat("\nManual, not detected by auto: ", accidental_drops_manual_only)
```

```
##
## Manual, not detected by auto:  31
```

```r
sink()
```

```r
# output all anovas in one place
all_anovas <- bind_rows(anova.Interface.distance %>% mutate(Measure = "Accuracy") %>%
    select(Measure, everything()), anova.Interface.totaltime %>% mutate(Measure = "Total Time") %>%
    select(Measure, everything()), anova.Interface.grabtime %>% mutate(Measure = "Grab Time") %>%
    select(Measure, everything()), anova.Interface.releasetime %>% mutate(Measure = "Release Time") %>%
    select(Measure, everything()), anova.Interface.drops %>% mutate(Measure = "Accidental Drops") %>%
    select(Measure, everything()), anova.SUS %>% mutate(Measure = "SUS") %>% select(Measure,
    everything()), anova.likert %>% rename(Measure = question) %>% select(Measure,
    everything())) %>% select(-`p<.05`, -Effect)  #%>% mutate(F=round(F,2))#, p='<.001')
# then set p values to be 'p < .001' (if true) and remove the *
all_anovas[which(all_anovas$p < 0.001), "p"] <- "p < .001"
all_anovas
```

```
##              Measure DFn DFd      F        p      pes
## 1           Accuracy   2  62 41.711 p < .001 0.574000
## 2         Total Time   2  62 36.619 p < .001 0.542000
## 3          Grab Time   2  62 29.057 p < .001 0.484000
## 4       Release Time   2  62 30.342 p < .001 0.495000
## 5   Accidental Drops   2  62 44.383 p < .001 0.589000
## 6                SUS   2  62  7.132    0.002 0.187000
## 7        comfortable   2  62  7.911 p < .001 0.203000
## 8            precise   2  62 27.921 p < .001 0.474000
## 9          intuitive   2  62  0.198    0.821 0.006000
## 10            tiring   2  62  0.012    0.989 0.000372
## 11          gripping   2  62 23.420 p < .001 0.430000
## 12         releasing   2  62 25.571 p < .001 0.452000
## 13           natural   2  62  1.608    0.209 0.049000
## 14         recommend   2  62  6.556    0.003 0.175000
## 15            agency   2  62  1.602     0.21 0.049000
## 16      satisfaction   2  62  5.901    0.005 0.160000
```

```r
sink("anova_data.csv")
cat("ANOVAs\n")
```

```
## ANOVAs
```

```r
write.csv(all_anovas)
```

```
## "","Measure","DFn","DFd","F","p","pes"
## "1","Accuracy",2,62,41.711,"p < .001",0.574
## "2","Total Time",2,62,36.619,"p < .001",0.542
## "3","Grab Time",2,62,29.057,"p < .001",0.484
## "4","Release Time",2,62,30.342,"p < .001",0.495
## "5","Accidental Drops",2,62,44.383,"p < .001",0.589
## "6","SUS",2,62,7.132,"0.002",0.187
## "7","comfortable",2,62,7.911,"p < .001",0.203
## "8","precise",2,62,27.921,"p < .001",0.474
## "9","intuitive",2,62,0.198,"0.821",0.006
## "10","tiring",2,62,0.012,"0.989",0.000372
## "11","gripping",2,62,23.42,"p < .001",0.43
## "12","releasing",2,62,25.571,"p < .001",0.452
## "13","natural",2,62,1.608,"0.209",0.049
## "14","recommend",2,62,6.556,"0.003",0.175
## "15","agency",2,62,1.602,"0.21",0.049
## "16","satisfaction",2,62,5.901,"0.005",0.16
```

```r
sink()

# output table for LaTeX
stargazer(all_anovas, summary = FALSE, rownames = FALSE, title = "caption here")
```

```
##
## % Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harva
## % Date and time: Sun, Oct 18, 2020 - 22:57:03
## \begin{table}[!htbp] \centering
##   \caption{caption here}
##   \label{}
## \begin{tabular}{@{\extracolsep{5pt}} cccccc}
## \\[-1.8ex]\hline
## \hline \\[-1.8ex]
## Measure & DFn & DFd & F & p & pes \\
## \hline \\[-1.8ex]
## Accuracy & $2$ & $62$ & $41.711$ & p \textless  .001 & $0.574$ \\
## Total Time & $2$ & $62$ & $36.619$ & p \textless  .001 & $0.542$ \\
## Grab Time & $2$ & $62$ & $29.057$ & p \textless  .001 & $0.484$ \\
## Release Time & $2$ & $62$ & $30.342$ & p \textless  .001 & $0.495$ \\
## Accidental Drops & $2$ & $62$ & $44.383$ & p \textless  .001 & $0.589$ \\
## SUS & $2$ & $62$ & $7.132$ & 0.002 & $0.187$ \\
## comfortable & $2$ & $62$ & $7.911$ & p \textless  .001 & $0.203$ \\
## precise & $2$ & $62$ & $27.921$ & p \textless  .001 & $0.474$ \\
## intuitive & $2$ & $62$ & $0.198$ & 0.821 & $0.006$ \\
## tiring & $2$ & $62$ & $0.012$ & 0.989 & $0.0004$ \\
## gripping & $2$ & $62$ & $23.420$ & p \textless  .001 & $0.430$ \\
## releasing & $2$ & $62$ & $25.571$ & p \textless  .001 & $0.452$ \\
## natural & $2$ & $62$ & $1.608$ & 0.209 & $0.049$ \\
## recommend & $2$ & $62$ & $6.556$ & 0.003 & $0.175$ \\
## agency & $2$ & $62$ & $1.602$ & 0.21 & $0.049$ \\
## satisfaction & $2$ & $62$ & $5.901$ & 0.005 & $0.160$ \\
## \hline \\[-1.8ex]
## \end{tabular}
## \end{table}
```

```
Measure=c("Accuracy (m)","Total Time (s)","Grab Time (s)","Release Time (s)","Accidental Drops (#)")

# build df of mean and sd for HHI Leap performance metrics
temp_df <- subject_data_all_long %>% ungroup() %>% filter(Interface=="HHI_Leap")
hhi_leap_mean_sd <- #left_join(get_summary_stats(temp_df, Distance,))
  data.frame(Measure=Measure,
      HHI_Leap_Mean=c(mean(temp_df$Distance), mean(temp_df$totaltime), mean(temp_df$grabtime), mean(temp
      HHI_Leap_SD=c(sd(temp_df$Distance), sd(temp_df$totaltime), sd(temp_df$grabtime), sd(temp_df$releas
             )
# build df of mean and sd for Oculus performance metrics
temp_df <- subject_data_all_long %>% ungroup() %>% filter(Interface=="Oculus")
oculus_mean_sd <-
  data.frame(Measure=Measure,
      Oculus_Mean=c(mean(temp_df$Distance), mean(temp_df$totaltime), mean(temp_df$grabtime), mean(temp_d
      Oculus_SD=c(sd(temp_df$Distance), sd(temp_df$totaltime), sd(temp_df$grabtime), sd(temp_df$release
             )
# combine hhi and oculus means and sd's
hhi_leap_oculus_mean_sd <- left_join(hhi_leap_mean_sd, oculus_mean_sd)

# build df of mean and sd for HHI and b_leap performance metrics
temp_df <- subject_data_all_long %>% ungroup() %>% filter(Interface=="B_Leap")
b_leap_mean_sd <-
  data.frame(Measure=Measure,
      B_Leap_Mean=c(mean(temp_df$Distance), mean(temp_df$totaltime), mean(temp_df$grabtime), mean(temp_d
      B_Leap_SD=c(sd(temp_df$Distance), sd(temp_df$totaltime), sd(temp_df$grabtime), sd(temp_df$release
             )
# combine hhi_leap and b_leap performance metric means and sd's
hhi_leap_b_leap_mean_sd <- left_join(hhi_leap_mean_sd, b_leap_mean_sd)

# performance t test results (add "measure" column)
all_ttests_interface <- bind_rows(
  ttest.Interface.distance %>% mutate(Measure="Accuracy (m)"),
  ttest.Interface.totaltime %>% mutate(Measure="Total Time (s)"),
  ttest.Interface.grabtime %>% mutate(Measure = "Grab Time (s)"),
  ttest.Interface.releasetime %>% mutate(Measure = "Release Time (s)"),
  ttest.Interface.drops %>% mutate(Measure = "Accidental Drops (#)")
)

# build performance t-test table for HHI vs Oculus
all_ttests_interface_hhiVsOculus <-
  all_ttests_interface %>% filter(group2 == "Oculus") %>%
  left_join(hhi_leap_oculus_mean_sd, by="Measure") %>%
  select(Measure, HHI_Leap_Mean, HHI_Leap_SD, Oculus_Mean, Oculus_SD, t=statistic, df, `p(Holm)`=p.adj)
  # add SUS
  bind_rows(
    left_join(SUS_mean_sd %>% select(Measure, HHI_Leap_Mean, HHI_Leap_SD,  Oculus_Mean, Oculus_SD),
              ttest.SUS %>% filter(group2=="Oculus") %>% mutate(Measure=`.y.`) %>%
                select(Measure, t=statistic, df, `p(Holm)`=p.adj), by="Measure")) %>%
  # add subjective scores
  bind_rows(
    # combine means/sd's and t-test results
    left_join(
      subjective_mean_sd %>% select(Measure, HHI_Leap_Mean, HHI_Leap_SD, Oculus_Mean, Oculus_SD),
```

```r
      ttests_subjective_all %>% filter(group2=="Oculus") %>% rename(Measure=question) %>%
        select(Measure, t=statistic, df, `p(Holm)`=p.adj), by="Measure")) %>%
  # round numbers (rounds down)
  mutate_if(is.numeric, round, 4)
# change p =0 to p <.001
all_ttests_interface_hhiVsOculus[which(all_ttests_interface_hhiVsOculus$`p(Holm)`<.001),"p(Holm)"] <- "p

# build performance t-test table for HHI vs B_Leap
all_ttests_interface_hhiVsBLeap <-
  all_ttests_interface %>% filter(group1== "B_Leap") %>%
  left_join(hhi_leap_b_leap_mean_sd, by="Measure") %>%
  select(Measure, HHI_Leap_Mean, HHI_Leap_SD, B_Leap_Mean, B_Leap_SD, t=statistic, df, `p(Holm)`=p.adj)
  mutate(t=-t) %>%
    # add SUS
  bind_rows(
    left_join(SUS_mean_sd %>% select(Measure, HHI_Leap_Mean, HHI_Leap_SD,  B_Leap_Mean, B_Leap_SD),
              ttest.SUS %>% filter(group1=="B_Leap") %>% mutate(Measure=`.y.`) %>%
                select(Measure, t=statistic, df, `p(Holm)`=p.adj), by="Measure")) %>% # add subjective
  bind_rows(
    left_join(
      subjective_mean_sd %>% select(Measure, HHI_Leap_Mean, HHI_Leap_SD,  B_Leap_Mean, B_Leap_SD),
      ttests_subjective_all %>% filter(group2=="B_Leap") %>% mutate(Measure=question) %>%
        select(Measure, t=statistic, df, `p(Holm)`=p.adj), by="Measure")
  ) %>%
  # round numbers (rounds down)
  mutate_if(is.numeric, round, 4)
# change p =0 to p <.001
all_ttests_interface_hhiVsBLeap[which(all_ttests_interface_hhiVsBLeap$`p(Holm)`<.001),"p(Holm)"] <- "p

stargazer(all_ttests_interface_hhiVsOculus, summary=FALSE, rownames = FALSE, title="caption")
```

```
##
## % Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harva
## % Date and time: Sun, Oct 18, 2020 - 22:57:03
## \begin{table}[!htbp] \centering
##   \caption{caption}
##   \label{}
## \begin{tabular}{@{\extracolsep{5pt}} cccccccc}
## \\[-1.8ex]\hline
## \hline \\[-1.8ex]
## Measure & HHI\_Leap\_Mean & HHI\_Leap\_SD & Oculus\_Mean & Oculus\_SD & t & df & p(Holm) \\
## \hline \\[-1.8ex]
## Accuracy (m) & 0.0158 & 0.0081 & 0.0071 & 0.0038 & 7.1268 & 31 & p \textless  .001 \\
## Total Time (s) & 3.5145 & 1.3465 & 2.2705 & 0.753 & 7.8842 & 31 & p \textless  .001 \\
## Grab Time (s) & 1.573 & 0.6522 & 0.9462 & 0.2479 & 6.9393 & 31 & p \textless  .001 \\
## Release Time (s) & 1.9416 & 0.8561 & 1.3243 & 0.5745 & 6.7013 & 31 & p \textless  .001 \\
## Accidental Drops (\#) & 2.4062 & 2.0924 & 0.125 & 0.336 & 6.2427 & 31 & p \textless  .001 \\
## SUS & 70.4688 & 18.8418 & 82.3438 & 14.7416 & -2.6887 & 31 & 0.022 \\
## comfortable & 3.375 & 1.1 & 4.156 & 0.92 & -3.0886 & 31 & 0.008 \\
## precise & 2.594 & 0.875 & 4.156 & 0.92 & -6.4689 & 31 & p \textless  .001 \\
## gripping & 3.094 & 1.174 & 4.406 & 1.073 & -5.2133 & 31 & p \textless  .001 \\
## releasing & 2.781 & 1.099 & 4.406 & 1.073 & -6.1393 & 31 & p \textless  .001 \\
## recommend & 3.375 & 1.238 & 4.094 & 0.928 & -2.6592 & 31 & 0.025 \\
```

```
## satisfaction & 5 & 1.047 & 5.594 & 0.875 & -2.4615 & 31 & 0.039 \\
## \hline \\[-1.8ex]
## \end{tabular}
## \end{table}
```

```r
stargazer(all_ttests_interface_hhiVsBLeap, summary=FALSE, rownames = FALSE, title="caption")
```

```
##
## % Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harva
## % Date and time: Sun, Oct 18, 2020 - 22:57:03
## \begin{table}[!htbp] \centering
##   \caption{caption}
##   \label{}
## \begin{tabular}{@{\extracolsep{5pt}} cccccccc}
## \\[-1.8ex]\hline
## \hline \\[-1.8ex]
## Measure & HHI\_Leap\_Mean & HHI\_Leap\_SD & B\_Leap\_Mean & B\_Leap\_SD & t & df & p(Holm) \\
## \hline \\[-1.8ex]
## Accuracy (m) & 0.0158 & 0.0081 & 0.0154 & 0.0052 & 0.2934 & 31 & 0.771 \\
## Total Time (s) & 3.5145 & 1.3465 & 3.5661 & 1.5687 & -0.2926 & 31 & 0.772 \\
## Grab Time (s) & 1.573 & 0.6522 & 1.3543 & 0.4097 & 2.2529 & 31 & 0.032 \\
## Release Time (s) & 1.9416 & 0.8561 & 2.2118 & 1.2365 & -2.3171 & 31 & 0.027 \\
## Accidental Drops (\#) & 2.4062 & 2.0924 & 4.75 & 2.6761 & -3.8627 & 31 & p \textless  .001 \\
## SUS & 70.4688 & 18.8418 & 69.9219 & 18.8418 & -0.1871 & 31 & 0.853 \\
## comfortable & 3.375 & 1.1 & 3.438 & 0.982 & -0.3117 & 31 & 0.757 \\
## precise & 2.594 & 0.875 & 2.438 & 1.076 & 0.5958 & 31 & 0.556 \\
## gripping & 3.094 & 1.174 & 2.875 & 1.129 & 0.9088 & 31 & 0.37 \\
## releasing & 2.781 & 1.099 & 2.594 & 1.214 & 0.641 & 31 & 0.526 \\
## recommend & 3.375 & 1.238 & 3.344 & 1.096 & 0.1664 & 31 & 0.869 \\
## satisfaction & 5 & 1.047 & 4.844 & 1.081 & 0.776 & 31 & 0.444 \\
## \hline \\[-1.8ex]
## \end{tabular}
## \end{table}
```

```r
# # output all performance t-tests
# all_performance_ttests_oculus <- bind_rows(
#   subject_data_all_long %>% t_test(Distance ~ Interface, comparisons = list(c("HHI_Leap","Oculus")), p
#   subject_data_all_long %>% t_test(totaltime ~ Interface, comparisons = list(c("HHI_Leap","Oculus")),
#   subject_data_all_long %>% t_test(grabtime ~ Interface, comparisons = list(c("HHI_Leap","Oculus")), p
#   subject_data_all_long %>% t_test(releasetime ~ Interface, comparisons = list(c("HHI_Leap","Oculus"))
#   subject_data_all_long %>% t_test(Drop_Count ~ Interface, comparisons = list(c("HHI_Leap","Oculus"))
# ) %>% adjust_pvalue(p.col="p", output.col="p.adj") %>% cbind(Measure) %>% select(Measure, t=statistic
# all_performance_ttests_oculus
#
# all_performance_ttests_leap <- bind_rows(
#   subject_data_all_long %>% t_test(Distance ~ Interface, comparisons = list(c("HHI_Leap","B_Leap")), p
#   subject_data_all_long %>% t_test(totaltime ~ Interface, comparisons = list(c("HHI_Leap","B_Leap")),
#   subject_data_all_long %>% t_test(grabtime ~ Interface, comparisons = list(c("HHI_Leap","B_Leap")), p
#   subject_data_all_long %>% t_test(releasetime ~ Interface, comparisons = list(c("HHI_Leap","B_Leap"))
#   subject_data_all_long %>% t_test(Drop_Count ~ Interface, comparisons = list(c("HHI_Leap","B_Leap"))
# ) %>% adjust_pvalue(p.col="p", output.col="p.adj") %>% mutate(statistic=-1*statistic) %>% select(Meas
# all_performance_ttests_leap
#
```

```r
# # output to file
# sink("ttest_stats.csv")
# cat("t-tests: Performance Measures (HHI_Leap and Oculus)\n")
# write.csv(all_performance_ttests_oculus)
# cat("\nt-tests: Performance Measures (HHI_Leap and B_Leap)\n")
# write.csv(all_performance_ttests_leap)
# sink()

# # output table for LaTeX
# stargazer(all_performance_ttests %>%
#             select(-p) %>%
#             mutate(`p(Holm)`=round(`p(Holm)`, 3), t=round(t, 3)), summary=FALSE, rownames = FALSE, ti
```

Output Interface order

```r
write.csv(file = "Interface_order_anova_tables.csv", Interface_order_output)
```