OptiVisT localization task analysis

Florian Pätzold, Ramon Zacharias

2023-07-25

Setup

Import all necessary packages.

```
library(dplyr)
library(tidyr)
library(ggplot2)
library(GGally)
library(ggcorrplot)
library(reshape2)
library(plotly)
```

Data preprocessing

Set working directory, load all the localization data, clean it and combine it into one data frame. Also load the grasping data for later comparisons.

```
# Set working directory to the folder containing the CSV files
HOME <- getwd()
SAVE <- paste0(getwd(), "/Plots/")
setwd(paste0(HOME, "/../Data"))

# Load combined grasping data
grasping_data <- read.csv("combined.csv", header = TRUE, sep = ",")
#grasping_data
```

```
# Create a list of file names that end with "_localization.csv"
setwd(paste0(HOME, "/../Data"))
file_list <- list.files(pattern = "_localization*")
# Create an empty data frame to store the combined data
localization_data <- data.frame()
# loop through each file in the list and add the data to combined_data
for (file in file_list) {
  # Extract the participant ID from the file name
  participant_id <- strsplit(file, "_")[[1]][1]
  # Get the name of the stimuli file
  stimuli_file <- paste0(participant_id, "_stimuli.csv")
  # Read the CSV files
  data_response <- read.csv(file, header = FALSE)
  data_stimuli <- read.csv(stimuli_file, header = FALSE)
```

```
  # the data was saved cumulative in each step, so we only take the last rows
  if(nrow(data_response) < 21){
    data_response <- tail(data_response, n=3)
    data_stimuli <- tail(data_stimuli, n=3)
  } else {
    data_response <- tail(data_response, n=6)
    data_stimuli <- tail(data_stimuli, n=6)
  }

  # Convert the data into a vector
  response <- as.vector(t(data_response))
  stimuli <- as.vector(t(data_stimuli))

  # Create Block column
  block <- c()
  for (i in 1:nrow(data_response)){
    curr_block <- c(rep(i, ncol(data_response[i,])))
    block <- c(block, curr_block)
  }

  # Combine everything into one data frame
  dummy <- data.frame(response)
  dummy$stimuli <- stimuli
  dummy$participant_id <- participant_id
  dummy$block <- block

  # Append the data to combined_data
  localization_data <- rbind(localization_data, dummy)
}

# Sort ascending by participantID
localization_data$participant_id <- as.numeric(localization_data$participant_id)
localization_data <- localization_data[order(localization_data$participant_id), ]

# Save the combined data frame as a CSV file
#write.csv(localization_data, "localization_combined.csv", row.names = FALSE)
#localization_data
```

Summary statistics & visualization

Mean accuracies

```
# Mean accuracy all trials
# Compute the number of lines where "response" and "stimuli" are equal
num_equal <- sum(localization_data$response == localization_data$stimuli)
# Compute the total number of lines
num_total <- nrow(localization_data)
# Compute the percentage of lines where "response" and "stimuli" are equal
percent_equal <- (num_equal / num_total) * 100
# Print the result
cat(sprintf("The accuracy over all trials is %.2f%%.", percent_equal))
```

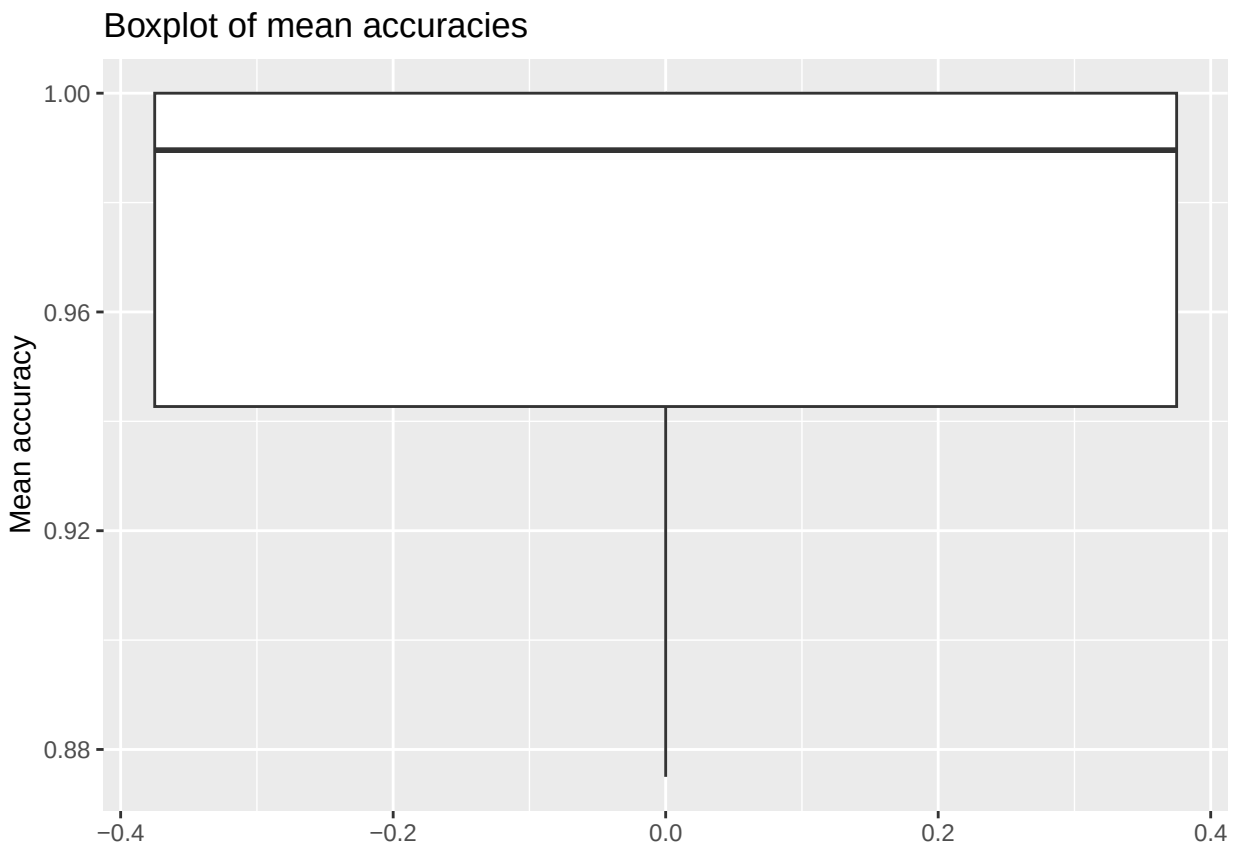
```
## The accuracy over all trials is 96.57%.
```

```
# Mean accuracy all trials per participant
# Compute the percentage of lines where "response" and "stimuli" are equal for each participant
accuracy <- aggregate(localization_data$response == localization_data$stimuli, by = list(localization_da
# Rename the columns of the result
colnames(accuracy) <- c("participant_id", "accuracy")
# Print the result
cat("\n\n")
```

```
summary(accuracy$accuracy)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.8750  0.9427  0.9896  0.9726  1.0000  1.0000
```

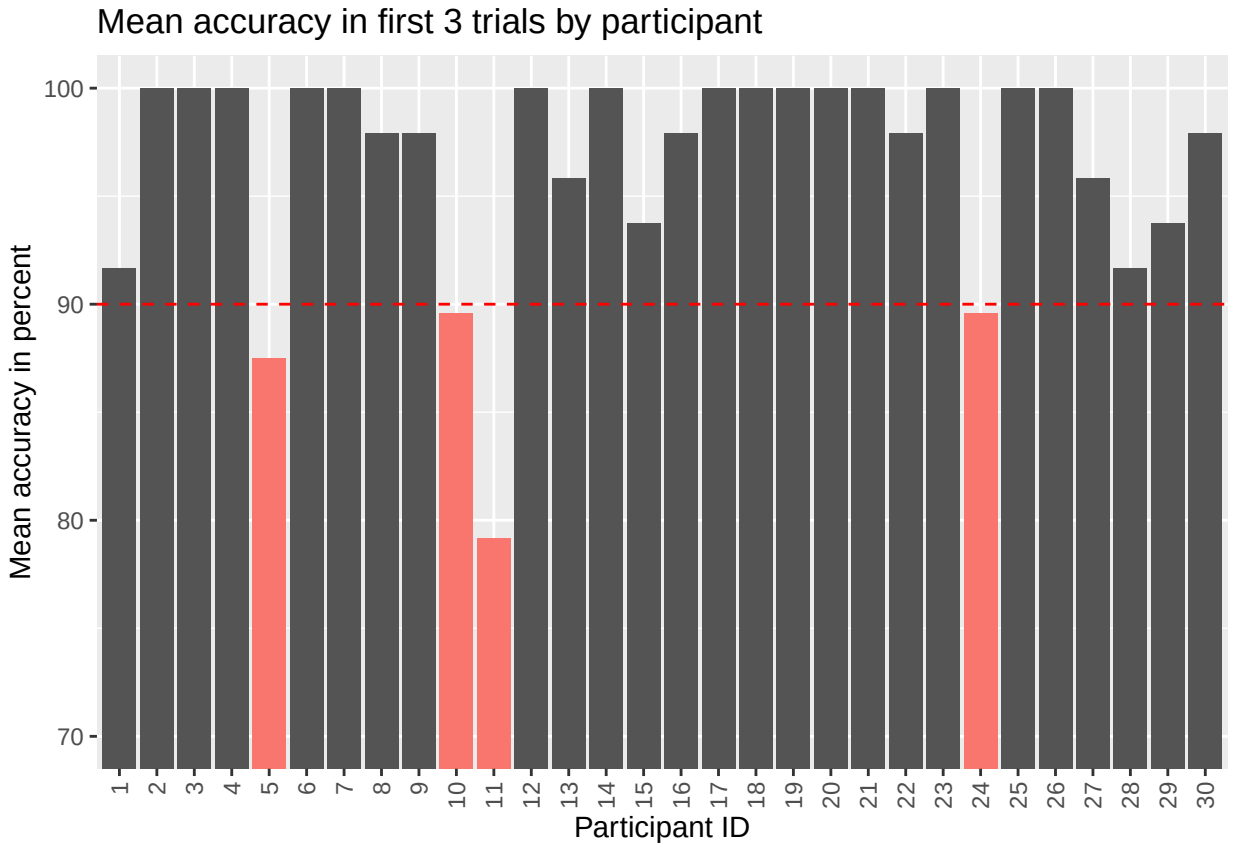
```
# Boxplot mean accuracy
ggplot(accuracy, aes(y = accuracy)) +
  geom_boxplot() +
  labs(y = "Mean accuracy") +
  ggtitle("Boxplot of mean accuracies")
```



Calculate and visualize accuracies for first three blocks per participant.

```
# Compute the maximum block number for each participant
max_blocks <- aggregate(localization_data$block, by = list(localization_data$participant_id), max)
# Compute the percentage of rows for each participant where stimuli and response are equal in the first
accuracy_first3 <- NULL
for (i in 1:nrow(max_blocks)) {
  curr_participant_id <- max_blocks[i, "Group.1"]
  selected_rows <- subset(localization_data, participant_id == curr_participant_id & block <= 3)
```

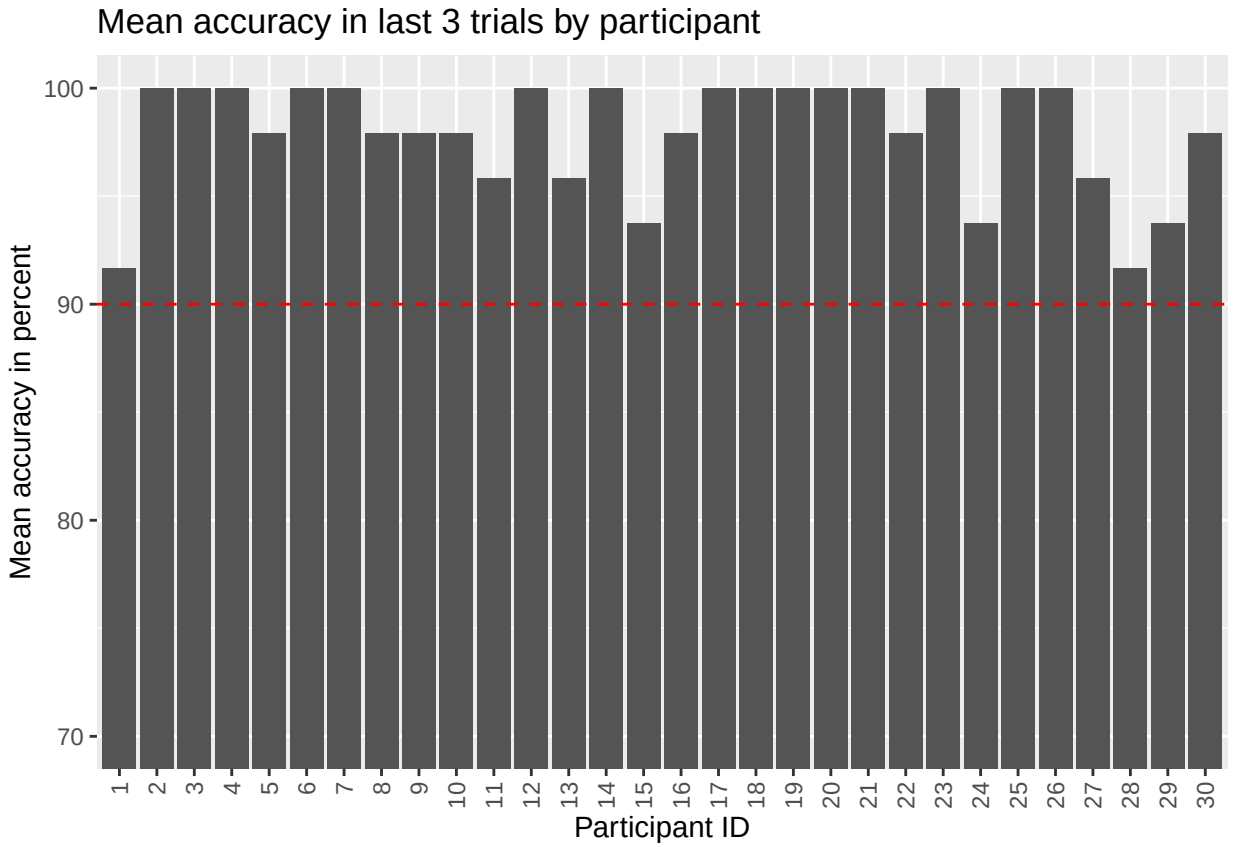
```
  equal_rows_count <- sum(selected_rows$response == selected_rows$stimuli)
  total_rows_count <- nrow(selected_rows)
  equal_rows_percent <- equal_rows_count / total_rows_count * 100
  accuracy_first3 <- rbind(accuracy_first3, data.frame(participant_id = curr_participant_id, accuracy =
}

# Create a new variable to mark participant with accuracy below 90
accuracy_first3$color <- ifelse(accuracy_first3$accuracy < 90, "red", "default")

# Visualize the accuracies of the first 3 blocks
ggplot(accuracy_first3, aes(x = as.factor(participant_id), y = accuracy, fill = color)) +
  geom_bar(stat = "identity") +
  scale_fill_manual(values = c("red" = "#F8766D", "default" = "#545454")) +
  labs(title = "Mean accuracy in first 3 trials by participant", x = "Participant ID", y = "Mean accura
  coord_cartesian(ylim = c(70,100)) +
  geom_hline(yintercept = 90, color = "red", linetype = "dashed") +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
  guides(fill = FALSE)
```



Calculate and visualize accuracies for last three blocks per participant.

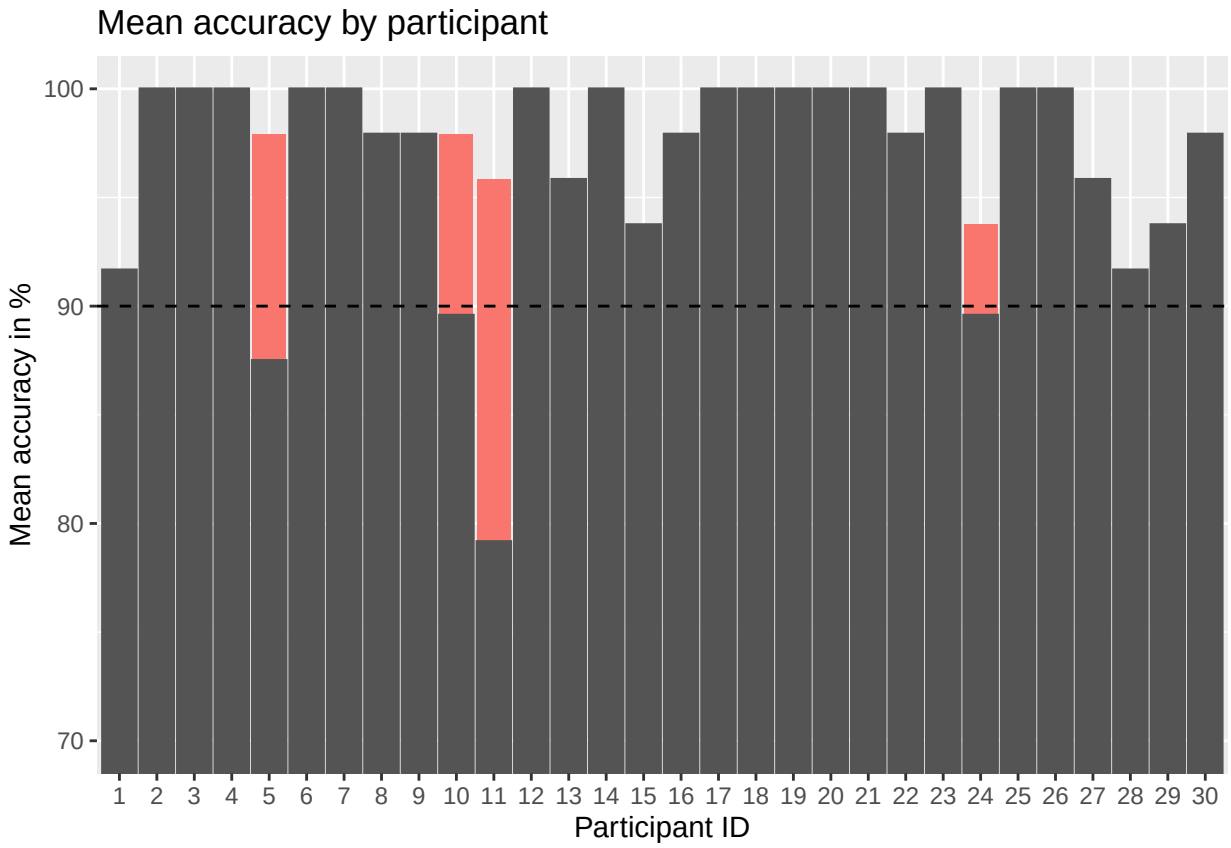
```
# Compute the percentage of rows for each participant where stimuli and response are equal in the last
accuracy_last3 <- NULL
for (i in 1:nrow(max_blocks)) {
  curr_participant_id <- max_blocks[i, "Group.1"]
  max_block <- max_blocks[i, "x"]
  selected_rows <- subset(localization_data, participant_id == curr_participant_id & block >= (max_bloc
```

```
  equal_rows_count <- sum(selected_rows$response == selected_rows$stimuli)
  total_rows_count <- nrow(selected_rows)
  equal_rows_percent <- (equal_rows_count / total_rows_count) * 100
  accuracy_last3 <- rbind(accuracy_last3, data.frame(participant_id = curr_participant_id, accuracy = ec
}

# Create a new variable to mark participant with accuracy below 90
accuracy_last3$color <- ifelse(accuracy_last3$accuracy < 90, "red", "default")

# Visualize the accuracies of the first 3 blocks
ggplot(accuracy_last3, aes(x = as.factor(participant_id), y = accuracy, fill = color)) +
  geom_bar(stat = "identity") +
  scale_fill_manual(values = c("red" = "#F8766D", "default" = "#545454")) +
  labs(title = "Mean accuracy in last 3 trials by participant", x = "Participant ID", y = "Mean accuracy
  coord_cartesian(ylim = c(70,100)) +
  geom_hline(yintercept = 90, color = "red", linetype = "dashed") +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
  guides(fill = FALSE)
```




```
# filter to combine plots
filter <- accuracy_last3 %>% mutate(acc_first3 = accuracy_first3$accuracy, participant_id = as.factor(pa

# Create the barplot
stacked_bar <- ggplot(filter, aes(x = participant_id, y = accuracy)) +
  geom_col(stat = "identity", fill = "#F8766D") +
  geom_col(aes(y = acc_first3), stat = "identity", color = "#545454", fill = "#545454", position = "sta
  labs(title = "Mean accuracy by participant", x = "Participant ID", y = "Mean accuracy in %") +
```

```
  geom_hline(yintercept = 90, color = "black", linetype = "dashed") +
  coord_cartesian(ylim = c(70,100))
stacked_bar
```




```
#ggsave("stacked_barplot.jpeg", plot = stacked_bar, dpi = 600)
```

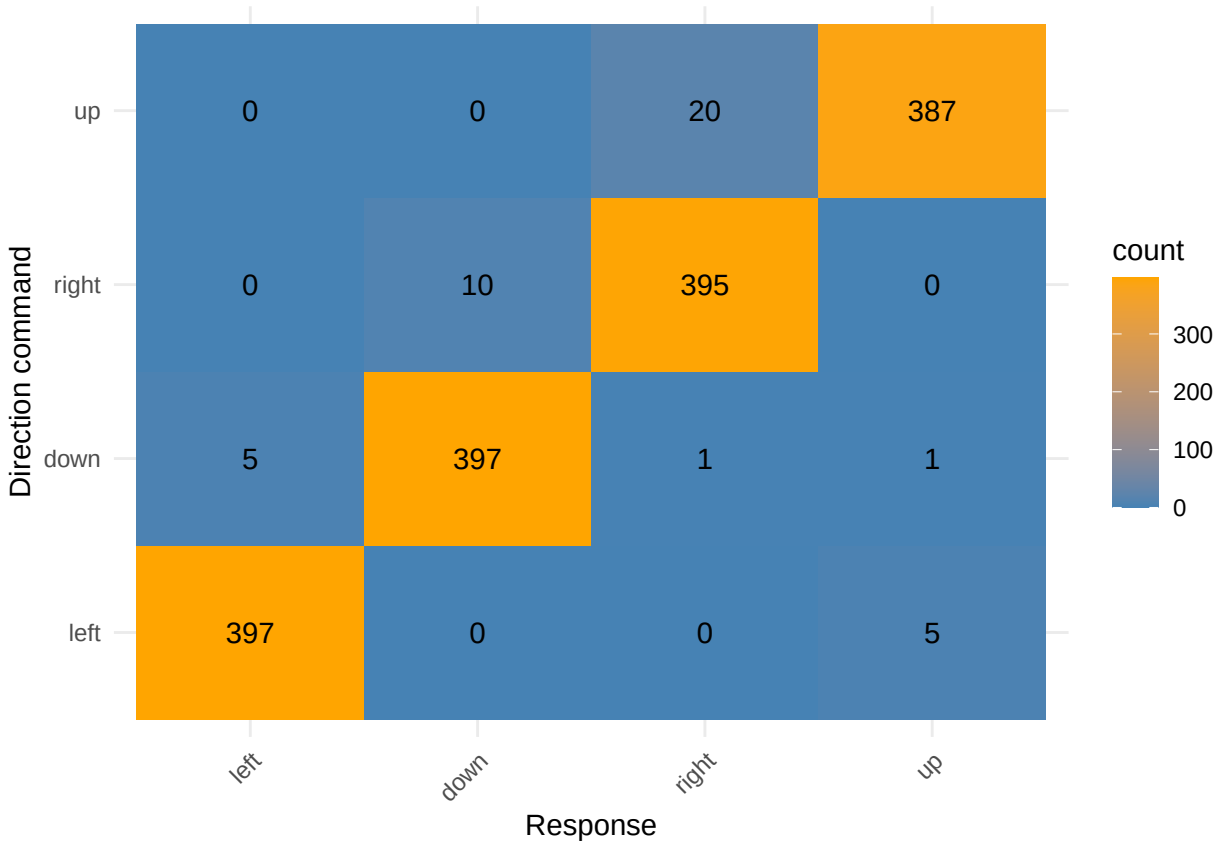
Direction confusion

Create a confusion matrix with counts and probabilities.

```
# Create a table of counts for each combination of stimuli and response
count_table <- table(localization_data$stimuli, localization_data$response)

# Put "no response" column last
count_table <- count_table[,c(1,2,4,5,3)]

# Print the result
print(count_table)
```

```
##
##         down left right  up no response
##   down   397    5     1   1           4
##   left     0  397     0   5           6
##   right   10    0   395   0           3
##   up       0    0    20 387           1
```

```r
# Convert the table into a data frame for ggplot2
ct_table <- count_table[, -5] # remove "no response"
count_df <- as.data.frame(melt(ct_table))

# Convert Var1 and Var2 to character
count_df$Var1 <- as.character(count_df$Var1)
count_df$Var2 <- as.character(count_df$Var2)

# Reorder the factor levels in Var1 and 2
count_df$Var1 <- factor(count_df$Var1, levels = c("left", "down", "right", "up", "no response"))
count_df$Var2 <- factor(count_df$Var2, levels = c("left", "down", "right", "up", "no response"))

# Add a new column specifying whether or not each cell is on the diagonal
# Exclude "no response" column from the diagonal
count_df$on_diagonal <- count_df$Var1 == count_df$Var2 & count_df$Var1 != "no response" & count_df$Var2
count_df <- count_df %>% rename(count = value)

# Create the plot
confusion_counts <- ggplot(count_df, aes(Var2, Var1, fill = count)) +
  geom_tile() +
  scale_fill_gradient(low = "steelblue", high = "orange") +
  geom_text(aes(label = round(count, 3)), size = 4) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(x = "Response", y = "Direction command") #+
  #ggtitle("Count of vibration signal misinterpretation")
confusion_counts
```




```
#ggsave(paste0(SAVE,"confusion_counts.jpeg"), plot = confusion_counts, dpi = 600)
```

```
# Calculate how often 2 locations got confused.
up_right <- count_table["up","right"] + count_table["right","up"]
up_down <- count_table["up","down"] + count_table["down","up"]
up_left <- count_table["up","left"] + count_table["left","up"]
right_down <- count_table["right","down"] + count_table["down","right"]
right_left <- count_table["right","left"] + count_table["left","right"]
down_left <- count_table["down","left"] + count_table["left","down"]
noresponse <- sum(count_table[, "no response"])
right <- count_table["right", "right"]
left <- count_table["left", "left"]

cat(sprintf("Up and Right got confused %i times. \nUp and Down got confused %i times. \nUp and Left got
            up_right, up_down, up_left, right_down, right_left, down_left, noresponse, right, left))
```

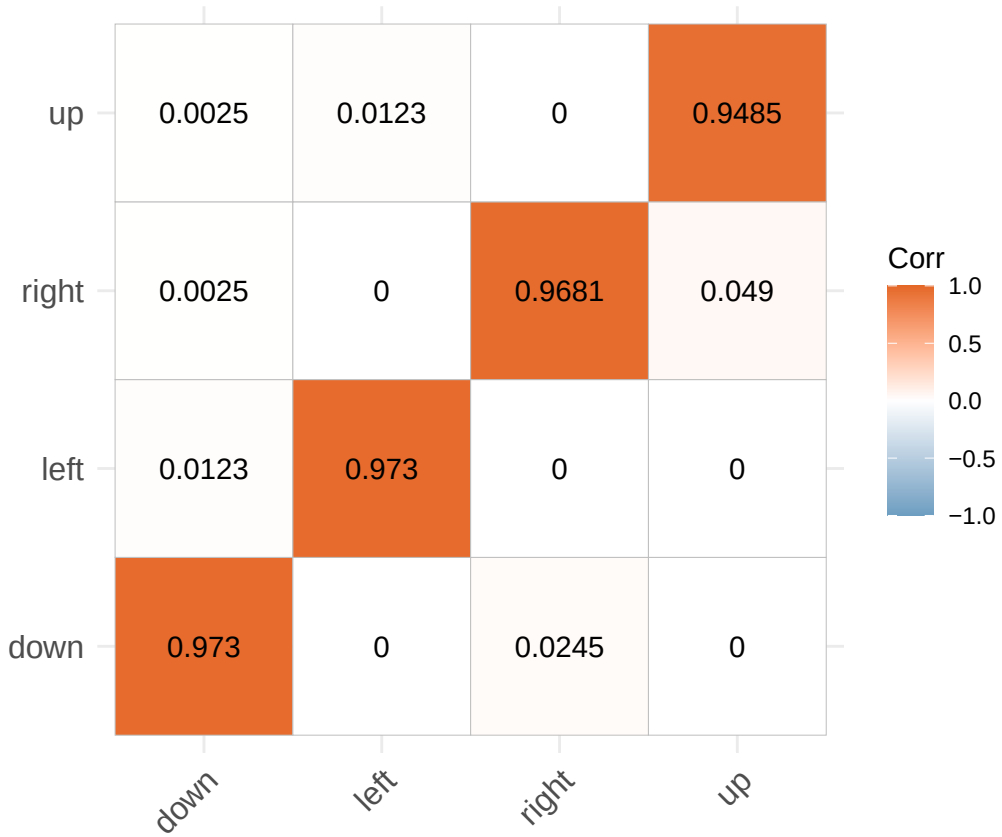
```
## Up and Right got confused 20 times.
## Up and Down got confused 1 times.
## Up and Left got confused 5 times.
## Right and Down got confused 11 times.
## Right and Left got confused 0 times.
## Down and Left got confused 5 times.
## In 14 trials the participant answered with no response.
## The Right vibration motor was identified correctly most frequently, namely 395 times.
## Overall participants correclty identified the Left motor the least. It was identified correctly 397
```

Now for probabilities.

```r
# Create a confusion matrix by dividing the count table by the row sums
confusion_matrix <- prop.table(count_table, 1)

# Print the result
confusion_matrix
```

```
##
##                down        left       right          up no response
##   down  0.973039216 0.012254902 0.002450980 0.002450980 0.009803922
##   left  0.000000000 0.973039216 0.000000000 0.012254902 0.014705882
##   right 0.024509804 0.000000000 0.968137255 0.000000000 0.007352941
##   up    0.000000000 0.000000000 0.049019608 0.948529412 0.002450980
```

```r
# Convert the matrix into a data frame for ggplot2
confusion_df <- as.data.frame(melt(confusion_matrix))

# Convert Var1 and Var2 to character
confusion_df$Var1 <- as.character(confusion_df$Var1)
confusion_df$Var2 <- as.character(confusion_df$Var2)

# Reorder the factor levels in Var2
confusion_df$Var2 <- factor(confusion_df$Var2, levels = c("down", "left", "right", "up", "no response"))

# Add a new column specifying whether or not each cell is on the diagonal
# Exclude "no response" column from the diagonal
confusion_df$on_diagonal <- confusion_df$Var1 == confusion_df$Var2 & confusion_df$Var2 != "no response"

# create heatmap with ggcorrplot
confusion_mat <- confusion_matrix[, -5] # remove "no response"
confusions <- ggcorrplot(confusion_mat, lab=TRUE, digits=4, colors=c("#6D9EC1", "white", "#E46726"))
confusions
```




```
#ggsave(paste0(SAVE,"confusion_mat_cc.jpeg"), plot = confusions, dpi = 600)
```

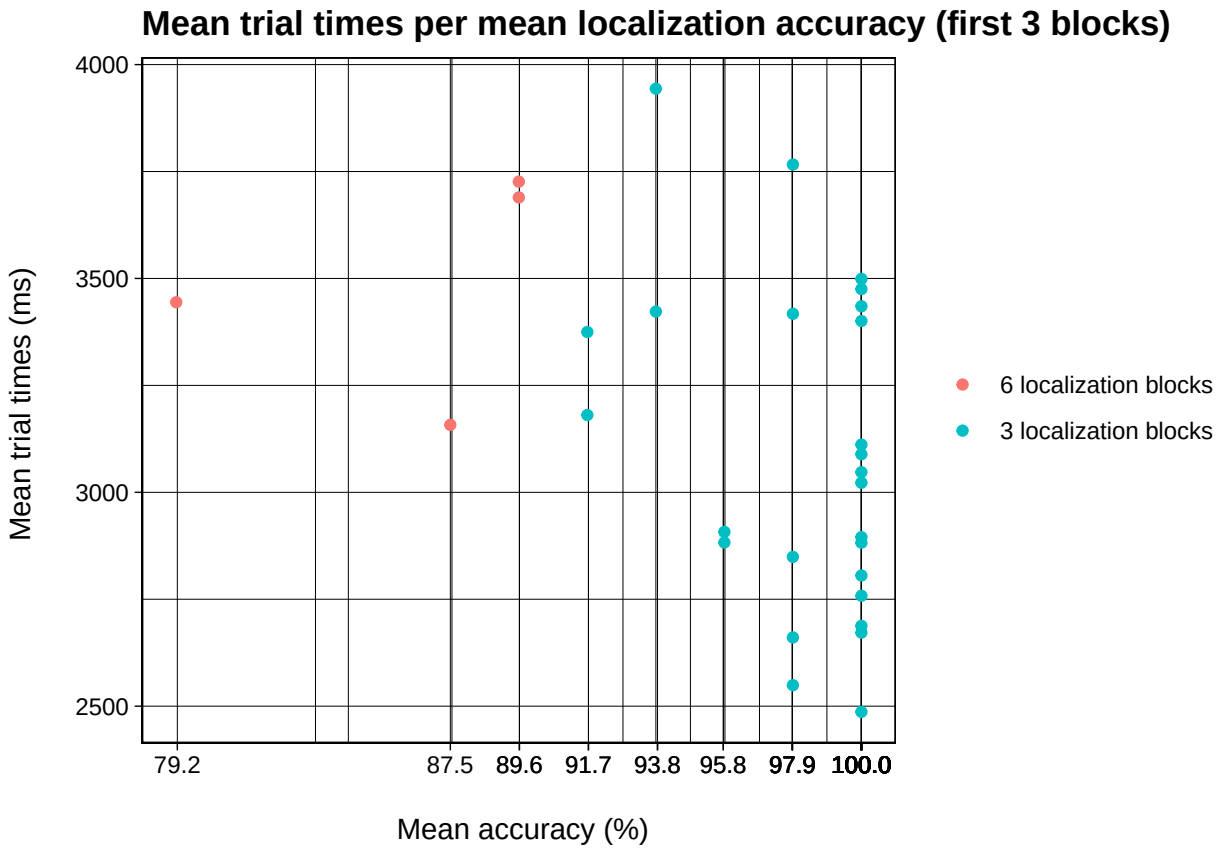
```
# Calculate how often 2 locations got confused.
up_right <- confusion_matrix["up","right"] + confusion_matrix["right","up"]
up_down <- confusion_matrix["up","down"] + confusion_matrix["down","up"]
up_left <- confusion_matrix["up","left"] + confusion_matrix["left","up"]
right_down <- confusion_matrix["right","down"] + confusion_matrix["down","right"]
right_left <- confusion_matrix["right","left"] + confusion_matrix["left","right"]
down_left <- confusion_matrix["down","left"] + confusion_matrix["left","down"]
noresponse <- sum(confusion_matrix[, "no response"])
right <- confusion_matrix["right", "right"]
left <- confusion_matrix["left", "left"]

cat(sprintf("Up and Right got confused ~%.2f%% of the time. \nUp and Down got confused ~%.2f%% of the ti
```

```
## Up and Right got confused ~0.05% of the time.
## Up and Down got confused ~0.00% of the time.
## Up and Left got confused ~0.01% of the time.
## Right and Down got confused ~0.03% of the time.
## Right and Left got confused ~0.00% of the time.
## Down and Left got confused ~0.01% of the time.
## In ~0.03% of trials the participant answered with no response.
## The Right vibration motor was identified with the highest accuracy of ~0.97%.
## Overall participants identified the Left motor with the lowest acccuracy of ~0.97%.
```

Predicting grasping performance with localization performance

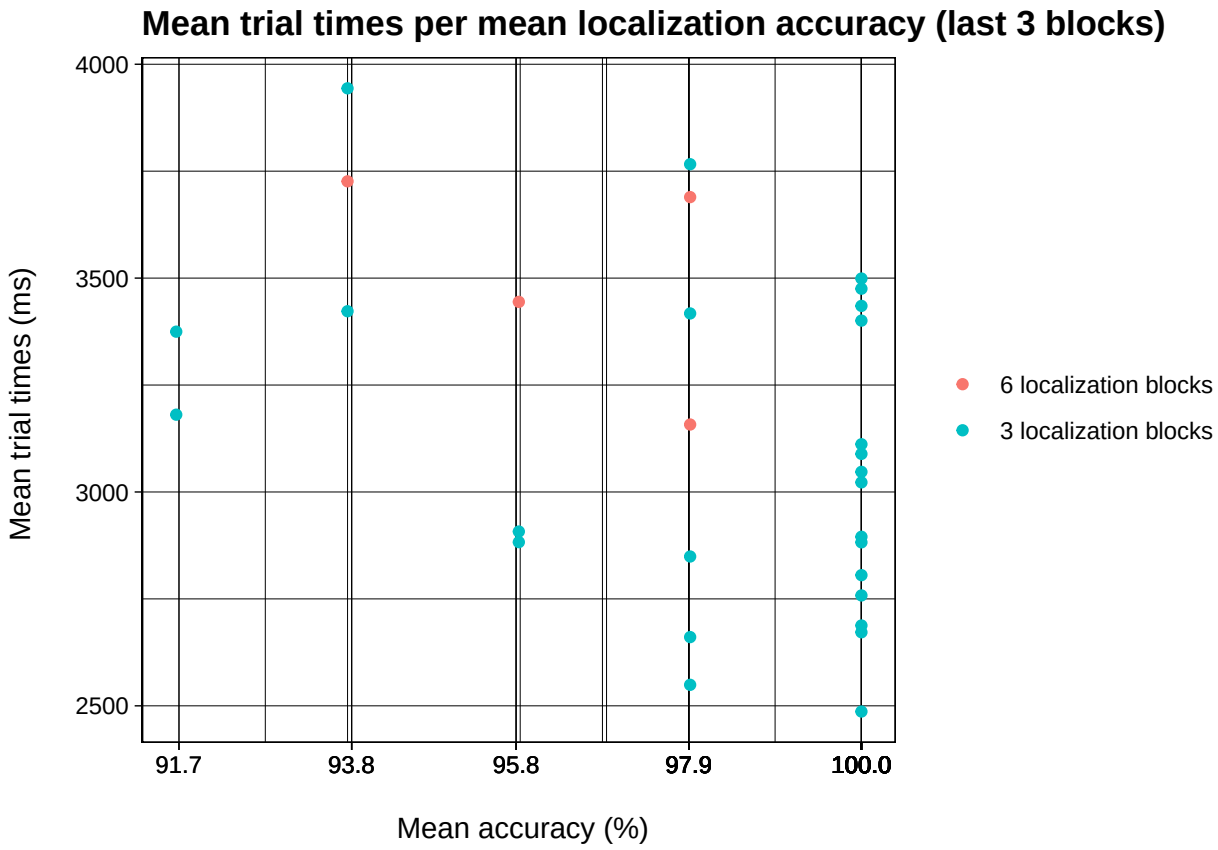
Reaction times

```
# filter for only successes in tactile condition
mixed_data <- grasping_data %>% filter(success=="success", condition=="tactile") %>% group_by(participar
mixed_data$acc_first3 <- accuracy_first3$accuracy
mixed_data$acc_last3 <- accuracy_last3$accuracy
mixed_data$only3 <- ifelse(mixed_data$acc_first3 == mixed_data$acc_last3, "yes", "no")

# scatterplot: y = grasping_RTs, x = mean acc (first 3 blocks)
mixed_data %>%
  ggplot(aes(x = acc_first3, y = mean_time, color=only3)) +
  geom_point() +
  labs(
    title = "Mean trial times per mean localization accuracy (first 3 blocks)",
    x = "\n Mean accuracy (%)",
    y = "Mean trial times (ms) \n"
    ) +
  theme_linedraw() +
  theme(plot.title = element_text(face = "bold"), legend.position = "right", legend.title = element_bla
  scale_x_continuous(breaks = round(mixed_data$acc_first3, 1)) +
  scale_color_discrete(labels = c("6 localization blocks", "3 localization blocks"))
```




```
# scatterplot: y = grasping_RTs, x = mean acc (last 3 blocks)
mixed_data %>%
  ggplot(aes(x = acc_last3, y = mean_time, color=only3)) +
```

```
  geom_point() +
  labs(
    title = "Mean trial times per mean localization accuracy (last 3 blocks)",
    x = "\n Mean accuracy (%)",
    y = "Mean trial times (ms) \n"
    ) +
  theme_linedraw() +
  theme(plot.title = element_text(face = "bold"), legend.position = "right", legend.title = element_bla
  scale_x_continuous(breaks = round(mixed_data$acc_last3, 1)) +
  scale_color_discrete(labels = c("6 localization blocks", "3 localization blocks"))
```

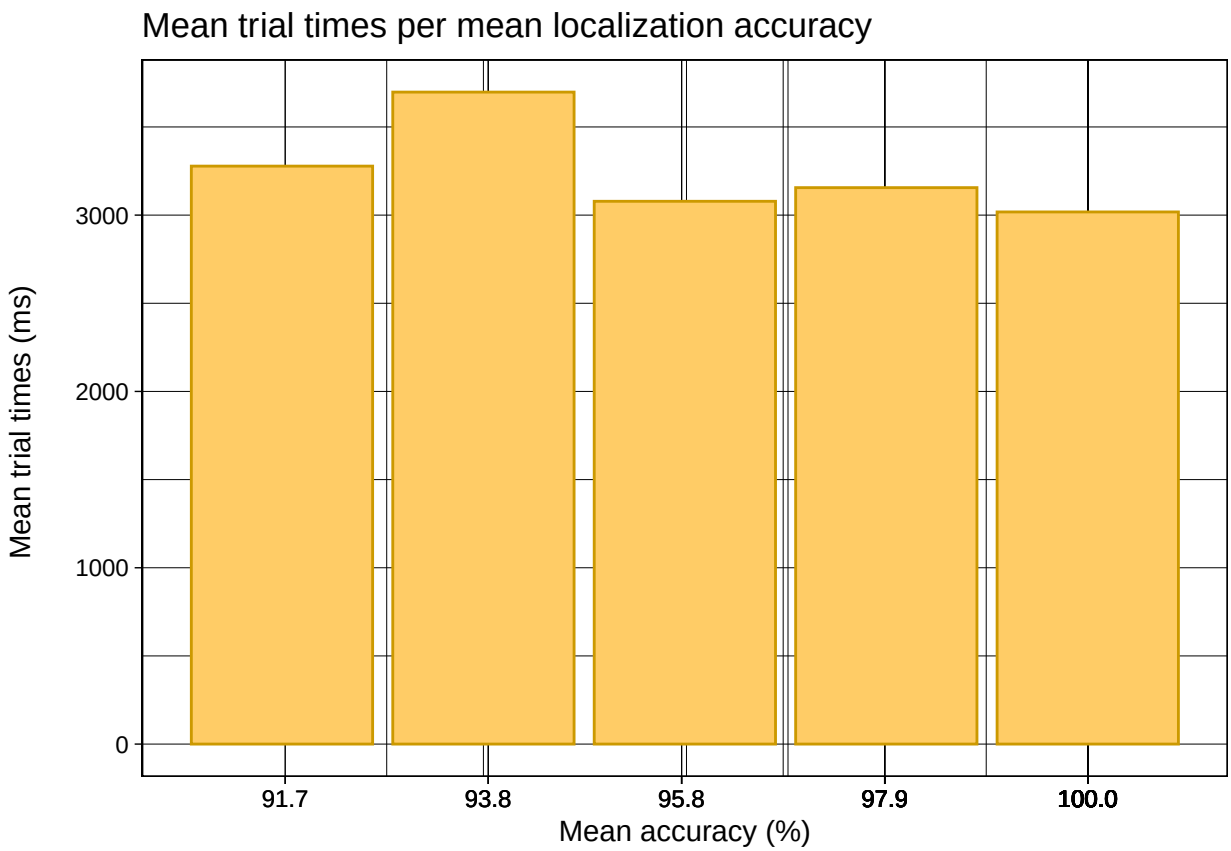



```
# barplot: y = mean RT, x = mean accuracy (last 3 blocks)
mixed_data %>% group_by(acc_last3) %>% summarize(mean_time = mean(mean_time))
```

```
## # A tibble: 5 x 2
##   acc_last3 mean_time
##       <dbl>     <dbl>
## 1      91.7     3278.
## 2      93.8     3697.
## 3      95.8     3078.
## 4      97.9     3156.
## 5     100       3018.
```

```
mixed_data %>% group_by(acc_last3) %>% summarize(mean_time = mean(mean_time)) %>%
  ggplot(aes(x = acc_last3, y = mean_time)) +
  geom_bar(stat = "identity", position = "dodge", colour="#CC9900", fill="#FFCC66") +
  labs(
```

```
    title = "Mean trial times per mean localization accuracy",
    x = "Mean accuracy (%)",
    y = "Mean trial times (ms) \n",
    ) +
  theme_linedraw() +
  scale_x_continuous(breaks = round(mixed_data$acc_last3, 1)) +
  scale_y_continuous(breaks = scales::pretty_breaks())
```



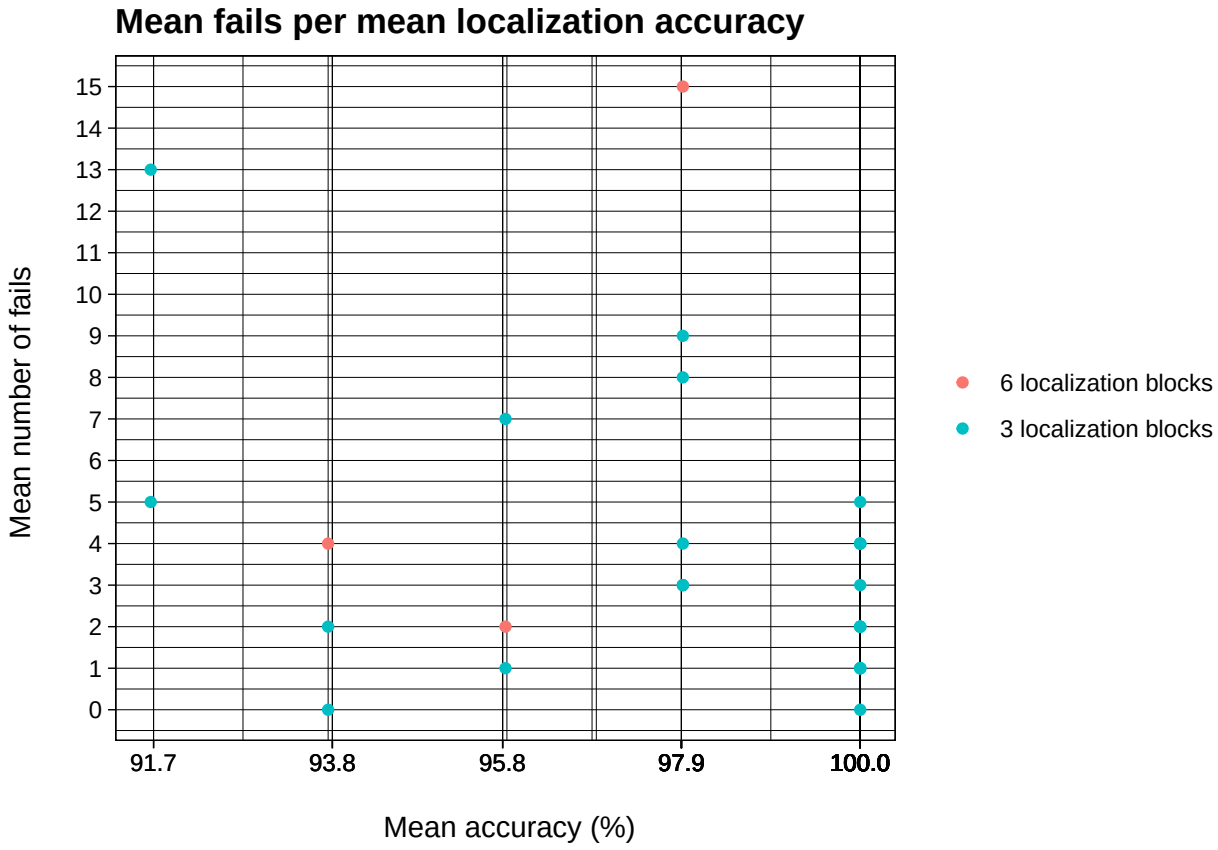

```
# test for task performance correlation
cor.test(mixed_data$mean_time, mixed_data$acc_last3)
```

```
##
##  Pearson's product-moment correlation
##
## data:  mixed_data$mean_time and mixed_data$acc_last3
## t = -2.3025, df = 28, p-value = 0.02895
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.66384486 -0.04523542
## sample estimates:
##        cor
## -0.3990021
```

Interpretation: positive correlation means more fails mean slower trials -> probably because people that have trouble interpreting tactile signals also go slower, but still fail more.

Fails

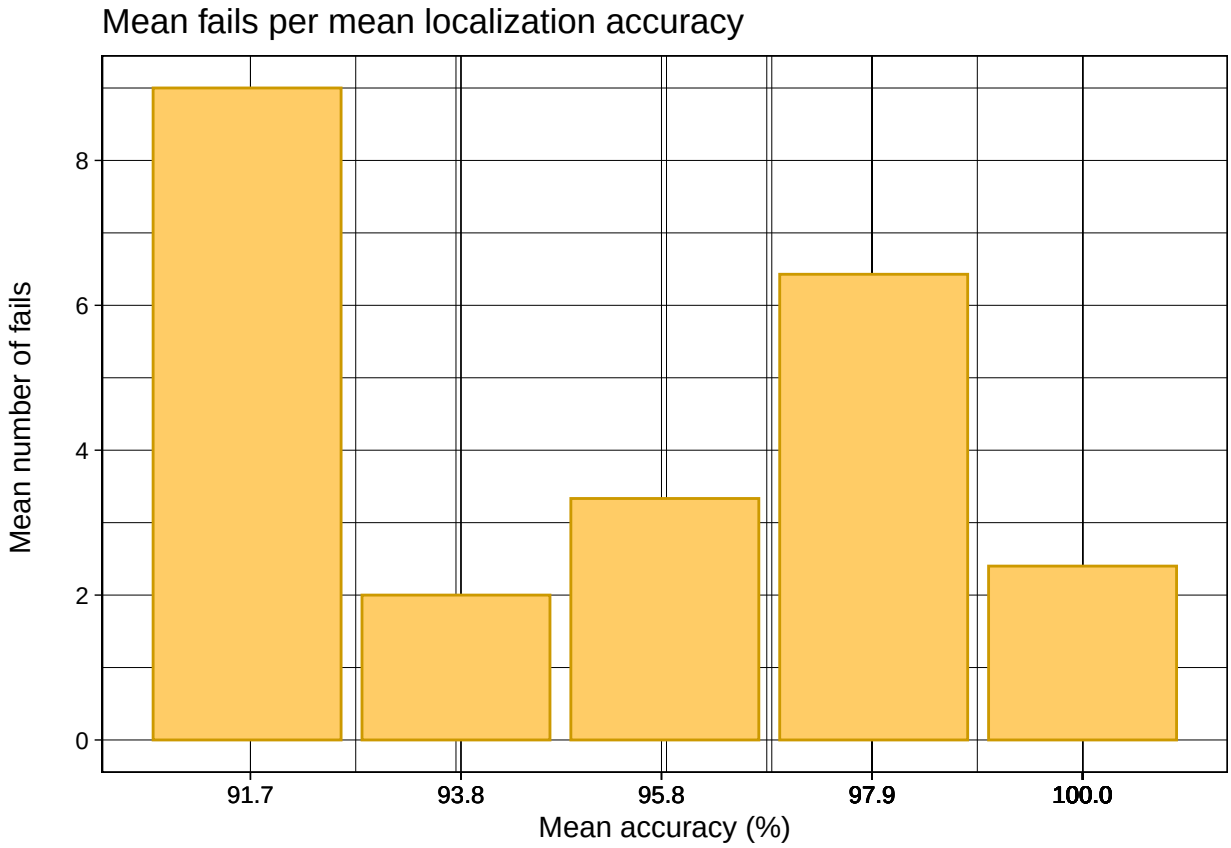
```
# have a look at the errors per accuracy
fails_df <- grasping_data %>%
  select(-1) %>%
  filter(success=="fail", condition=="tactile") %>%
  group_by(participant_id) %>%
  count() %>%
  rename_at('n', ~'fails') %>%
  rbind(data.frame(participant_id = c(23,29), fails = c(0,0))) %>%
  arrange(participant_id)

# add to mixed data
mixed_data$fails <- fails_df$fails

# participants scatterplot: y = mean fails, x = mean acc (last 3 blocks)
mixed_data %>%
  ggplot(aes(x = acc_last3, y = fails, colour=only3)) +
  geom_point() +
  labs(
    title = "Mean fails per mean localization accuracy",
    x = "\n Mean accuracy (%)",
    y = "Mean number of fails \n"
    ) +
  theme_linedraw() +
  theme(plot.title = element_text(face = "bold"), legend.position = "right", legend.title = element_bla
  scale_x_continuous(breaks = round(mixed_data$acc_last3, 1)) +
  scale_y_continuous(breaks = c(0:15)) +
  scale_color_discrete(labels = c("6 localization blocks", "3 localization blocks"))
```




```
# barplot: y = mean fails, x = mean acc
mixed_data %>% group_by(acc_last3) %>% summarize(mean_fails = mean(fails))
```

```
## # A tibble: 5 x 2
##   acc_last3 mean_fails
##       <dbl>      <dbl>
## 1      91.7       9
## 2      93.8       2
## 3      95.8       3.33
## 4      97.9       6.43
## 5     100         2.4
```

```
mixed_data %>% group_by(acc_last3) %>% summarize(mean_fails = mean(fails)) %>%
  ggplot(aes(x = acc_last3, y = mean_fails)) +
  geom_bar(stat = "identity", position = "dodge", colour="#CC9900", fill="#FFCC66") +
  labs(
    title = "Mean fails per mean localization accuracy",
    x = "Mean accuracy (%)",
    y = "Mean number of fails \n",
    ) +
  theme_linedraw() +
  scale_x_continuous(breaks = round(mixed_data$acc_last3, 1)) +
  scale_y_continuous(breaks = scales::pretty_breaks())
```



Let's test the correlation again between grasping accuracy and localization accuracy / mean trial times.

```
# calc acc exclude experimenter fails
# have a look at the errors per accuracy
total_trials <- grasping_data %>%
  select(-1) %>%
  filter(success!="exFail", condition=="tactile") %>%
  group_by(participant_id) %>%
  count() %>%
  rename_at('n', ~'trial_count')

# add to mixed data
mixed_data$grasp_acc <- 1 - (mixed_data$fails / total_trials$trial_count)

# correlation testing
cor.test(mixed_data$grasp_acc, mixed_data$acc_last3)
```

```
##
##  Pearson's product-moment correlation
##
## data:  mixed_data$grasp_acc and mixed_data$acc_last3
## t = 1.6951, df = 28, p-value = 0.1012
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.06201723  0.59945340
## sample estimates:
##       cor
## 0.3050681
```

```r
#cor.test(mixed_data$grasp_acc, mixed_data$mean_time)
```

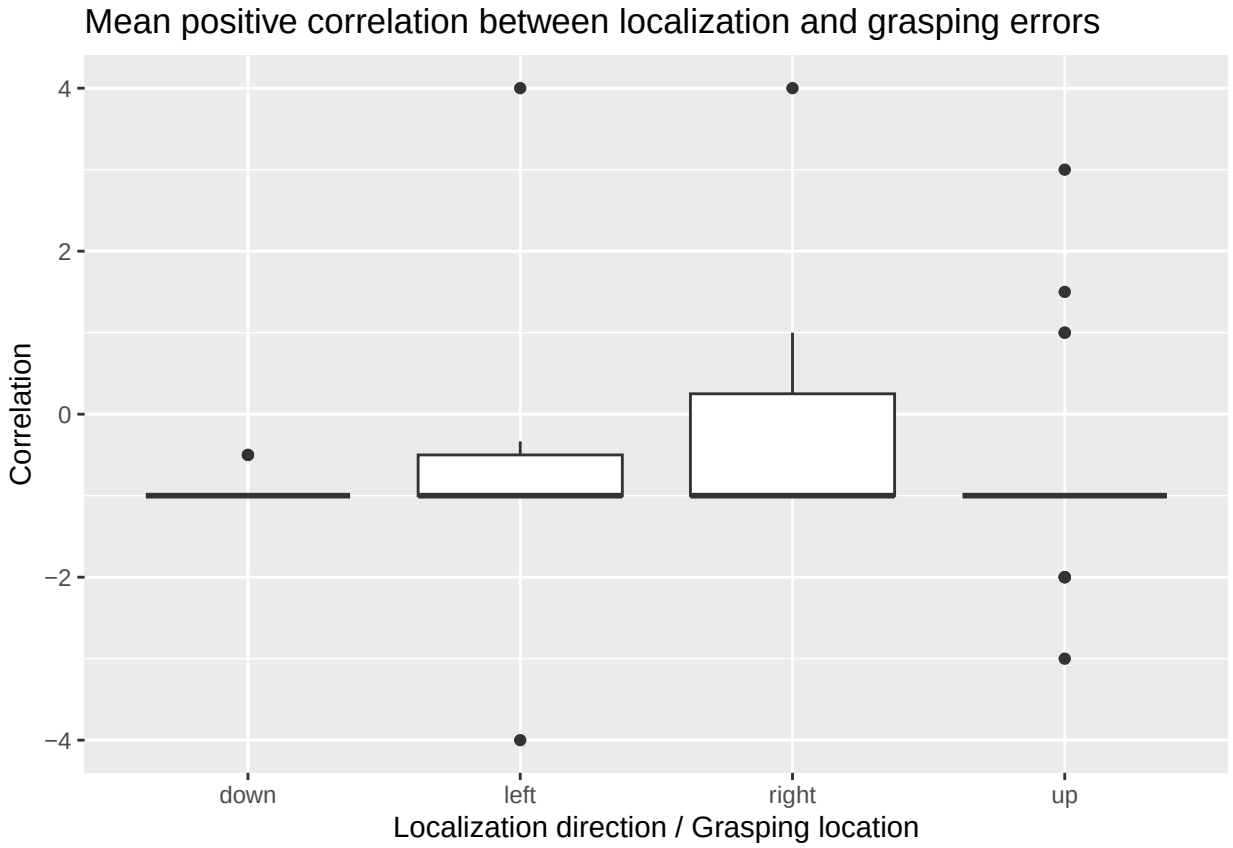
```r
# idea: create a scatterplot in 3D to observe correlation between errors in both tasks in form of clust

# 1. Create grasping error counts per object location
# rename locations
pos_data <- grasping_data # dummy
pos_data$location[pos_data$location == "rep_1"] = "1"
pos_data$location[pos_data$location == "rep_2"] = "2"
pos_data$location[pos_data$location == "rep_3"] = "3"
pos_data$location[pos_data$location == "rep_4"] = "4"
pos_data$location[pos_data$location == "rep_5"] = "5"
pos_data$location[pos_data$location == "rep_6"] = "6"
pos_data$location[pos_data$location == "rep_7"] = "7"
pos_data$location[pos_data$location == "rep_8"] = "8"
pos_data$location[pos_data$location == "rep_9"] = "9"

# create a failed column (necessary for 0 counts in the code below)
pos_data$failed <- ifelse(pos_data$success == "fail", 1, 0)

# grasping error counts per participant per object location (left, right, up, down)
pos_data <- pos_data %>% filter(condition=="tactile") %>% complete(participant_id, location, fill = lis

# rename locations to localization task directions for merging
pos_data$location[pos_data$location == 2] = "up"
pos_data$location[pos_data$location == 4] = "left"
pos_data$location[pos_data$location == 6] = "right"
pos_data$location[pos_data$location == 8] = "down"


# 2. Create localization error counts
# only last 3 blocks for all participants are taken into account
loc_data <- localization_data %>% filter(!(participant_id==5 & (block==1 | block==2 | block==3)),
                              !(participant_id==10 & (block==1 | block==2 | block==3)),
                              !(participant_id==11 & (block==1 | block==2 | block==3)),
                              !(participant_id==24 & (block==1 | block==2 | block==3)))

# determine failed directions (location) in localization trials
loc_data$location <- case_when(
  (loc_data$stimuli == "left" & loc_data$response != "left") ~ "left",
  (loc_data$stimuli == "right" & loc_data$response != "right") ~ "right",
  (loc_data$stimuli == "up" & loc_data$response != "up") ~ "up",
  (loc_data$stimuli == "down" & loc_data$response != "down") ~ "down",
  TRUE ~ "none")

# create a failed column (necessary for 0 counts in the code below)
loc_data$failed <- ifelse(loc_data$response == loc_data$stimuli, 0, 1)

# localization error counts per participant per command direction (left, right, up, down)
loc_data <- loc_data %>% filter(failed==1) %>% complete(participant_id, location, fill = list(failed = 


# 3. Merge grasping and localization errors per participant and location/direction
cluster_data <- merge(pos_data, loc_data, by = c("participant_id", "location"), all = TRUE)
```

```r
cluster_data[is.na(cluster_data)] <- 0
cluster_data <- cluster_data %>% filter(!(grasping_errors == 0 & localization_errors == 0))
cluster_data
```

```
##    participant_id location grasping_errors localization_errors
## 1               1     down               0                   1
## 2               1     left               4                   1
## 3               1       up               3                   2
## 4               2       up               1                   0
## 5               4     left               1                   0
## 6               5    right               0                   1
## 7               5       up               1                   0
## 8               6       up               2                   0
## 9               7     down               1                   0
## 10              7       up               1                   0
## 11              8    right               4                   1
## 12              9     down               1                   0
## 13              9    right               1                   1
## 14              9       up               3                   0
## 15             10     left               4                   0
## 16             10       up               3                   1
## 17             11     down               0                   1
## 18             11    right               1                   0
## 19             11       up               1                   1
## 20             12       up               1                   0
## 21             13     left               0                   2
## 22             14       up               1                   0
## 23             15     down               0                   2
## 24             15     left               0                   1
## 25             16     down               0                   1
## 26             16       up               2                   0
## 27             17    right               1                   0
## 28             17       up               1                   0
## 29             19       up               1                   0
## 30             20       up               1                   0
## 31             22    right               1                   0
## 32             22       up               1                   1
## 33             24     down               1                   0
## 34             24     left               0                   1
## 35             24    right               0                   2
## 36             24       up               2                   0
## 37             27     down               0                   1
## 38             27     left               0                   1
## 39             28     down               0                   2
## 40             28     left               0                   2
## 41             28       up               1                   0
## 42             29     left               0                   3
## 43             30     down               0                   1
```

```r
# boxplot correlation coefficients between localization and grasping errors
# calculation: graspingERR / localizationERR (transformed 0's to -1's) --> values close to 1 are positi
coeffs <- cluster_data %>% mutate_all(~ ifelse(. == 0, -1, .)) %>% select(-participant_id) %>% group_by

coeffs %>%
```

```
  ggplot(aes(x = location, y = coeff)) +
  geom_boxplot() +
  labs(x = "Localization direction / Grasping location", y = "Correlation") +
  ggtitle("Mean positive correlation between localization and grasping errors")
```




```
cor(cluster_data$grasping_errors, cluster_data$localization_errors)
```

```
## [1] -0.3084282
```

Interpretation: The plot shows that the errors made in both tasks seem to be unrelated. The correlation between both error types is -0.308, slightly negatively correlated, which means that for errors made in one task the error count in the other task is going into the other direction. This might mean that participants in the localization task efficiently learned to identify vibration directions (0 was swapped with -1 values for calculation, so overall negative correlation values mean overall low error counts).