Does spatial location represent keypress movements better than hand postures?

Pia Freiin von Boeselager

Department of Psychology, University of Twente

M12 BSc Thesis PSY (2023-2A)

Willem Verwey, 1st supervisor

Russel Chan, 2nd supervisor

July 26th, 2024

Abstract

This study aimed to ascertain whether hand postures, spatial locations, or both represent keypress movement in memory. To examine this, a reaction time (RT) task was administered to 32 participants. Each had to undergo three conditions: The spatial location, the hand posture, and the control condition. The former two were either congruent or incongruent with a key participants were instructed to press. It was hypothesised that both the congruent spatial condition as well as the congruent hand posture one should render shorter RTs than their incongruent counterparts. The data supported this only for the spatial condition. The study design may partially explain why the expected effect was not found for the hand postures (i.e. the postures may have not been distinct enough from one another). Whether this is the case, will have to be researched further.

Introduction

When we plan to carry out a movement, a certain feature will get activated in memory to represent this action. However, which features exactly represent movement remains uncertain. Our current understanding of the motor cortex suggests that movement is coded along many features, such as body postures, spatial relations, and body parts (Graziano, 2016). For instance, if a person wants to grab their glasses, the corresponding movement would be – before the action is carried out – represented as the hand posture of holding the glasses and the arm posture required to take the hand to the designated location (Graziano, 2016; Rosenbaum, 2017; Verwey, 2023; Verwey et al., 2016). As the existence of different action maps along the cortex for the above-mentioned features seems to be apparent (Graziano, 2016), it is plausible that memory codes movement using similar representations. Due to the growing interest in the functioning of the cognitive processes behind movement, the current study aims to examine whether keypress movement is coded as hand postures, spatial locations, or both.

Theoretical Background

The *common coding approach* argues that not only the preparation for a movement may activate related representations in memory but perception of one activates them as well (Stoet & Hommel, 2002; Vogt et al., 2003). Furthermore, *feature binding theory* indicates that a feature can only be part of one representation at the same time, meaning a stimulus occupying the same representations as a planned action with a dissimilar feature will interfere with said action due to feature overlap (Ding et al., 2017; Hommel, 2004; Stoet & Hommel, 2002).

To illustrate, if a person watches someone reach in a different direction after they planned to grab their glasses, it would slow their ensuing reach. This is because both action plan and perceived action share the same representations of the posture required to reach as well as of spatial location which, as they are bound to different locations, causes feature overlap. In other words, the perception of the differing action thus occupies the representation needed to carry out the previously planned action (Stoet & Hommel, 2002). However, this interference with the planned action occurs only if both the planned and perceived actions' features are fully integrated (Stoet & Hommel, 2002). If the time for integration for either is not given, the representation is merely activated, and its respective features do not block the relevant representation.

Hand posture and spatial location were chosen as likely representations as aforementioned action maps along the cortex point into their direction (Graziano, 2016). Spatial location in particular is expected as spatial memory seems to play an important role in planning and carrying out certain movements, such as reaching (Avraamides & Kelly, 2008) and psychological phenomena, such as the Simon effect (i.e. stimuli which spatially correspond to a response accelerate this response) demonstrate its relevance in facilitating motoric responses (Hommel, 2011, 2019; Simon & Rudell, 1967). The importance of hand postures in hand movement is well documented (Rosenbaum, 2017; Rosenbaum et al., 2001; Vogt et al., 2003). Rosenbaum et al. (2001) outlines how already learned movements, such as grasping, requires the recall of stored postures.

Current Study

In the current experiment, a reaction time task was conducted to assess whether two features – hand postures and spatial relation – represent key press movements in memory. Each participant performed in three conditions: the control condition, the spatial condition, and the motor condition. In each condition, they were instructed to press a number on a keyboard. After they prepared for this action – but before carrying it out – they were shown different pictures for a short amount of time. In the control condition, participants were shown random shapes which were not expected to have a significant effect on their response times. In the spatial condition, they were shown four squares, one of which was highlighted to indicate a spatial position which either did or did not correspond to the key they were expected to press. Similarly, in the motor condition, participants were confronted with photos of hand postures either consistent or inconsistent with their key pressing posture.

The interstimulus intervals (ISI) – the duration between the condition stimulus and the go-signal – were either 0ms, 300ms, 600ms, or 900ms. According to Vogt et al. (2003) at 0ms feature activation occurs, fastening the ensuing response, whether the stimulus corresponds or not; after 500ms, however, feature integration will take place, thus slowing the response if the condition stimulus is incongruent, as the mental space needed for action is occupied. The ISIs of 600ms and 900ms were added as the hand posture in the motor condition seemed to require more time to recognise. Then, the participants were shown the go-signal and pressed the previously prepared number while their response time was measured.

It was expected that in both the spatial as well as the posture condition, pictures that were incompatible with the key press movement would slow down participants' reaction time (RT). Longer RTs for incompatible pictures were expected when the pictures were shown long enough for the features to bind, and thus that the inconsistent spatial and posture features would slow the response.

Methods

Participants

The final sample consisted of 32 participants (6 Male, 26 Female; *M*(age) = 23.44; *SD*(age) = 5.25; age range: 18-40). Seven were Dutch, 23 German, and other nationalities included Lithuanian, Kazakh, Russian, Indonesian, Turkish, Bulgarian, and Pakistani. One participant was excluded as their response times were exceptionally slow, rendering them unusable for further analysis.

Participants were gathered through convenience sampling. Some were contacted by the researcher directly, while others signed up voluntarily through a website of the researchers' university. Thus, 27 were students at this university and received participation credits. Inclusion criteria were being between the ages of 18 and 40, sufficient English skills to understand the instructions, being right-handed, no alcohol consumption 24 hours prior to the experiment, and being a non-smoker. Prior to data collection, ethical approval was obtained from the BMS Ethics Committee at the University of Twente (No. 240215).

Materials

The study took place in the BMS Flex rooms of the University of Twente which consists of a small cubicle (ca. 8m^2) with a chair and table. It was equipped with an OptiPlex 7050 desktop computer (Dell Technologies Inc., Round Rock, TX, USA) – operating on Windows 10 – which was connected to an AOC FreeSync monitor with a 144Hz refresh rate, a USB (HP Wired Desktop 320K) keyboard. For the consent form as well as a demographics questionnaire Qualtrics was used (Appendix A) and for data collection E-Prime® 2.0 PST (Psychology Software Tools, 8 2023).

Task

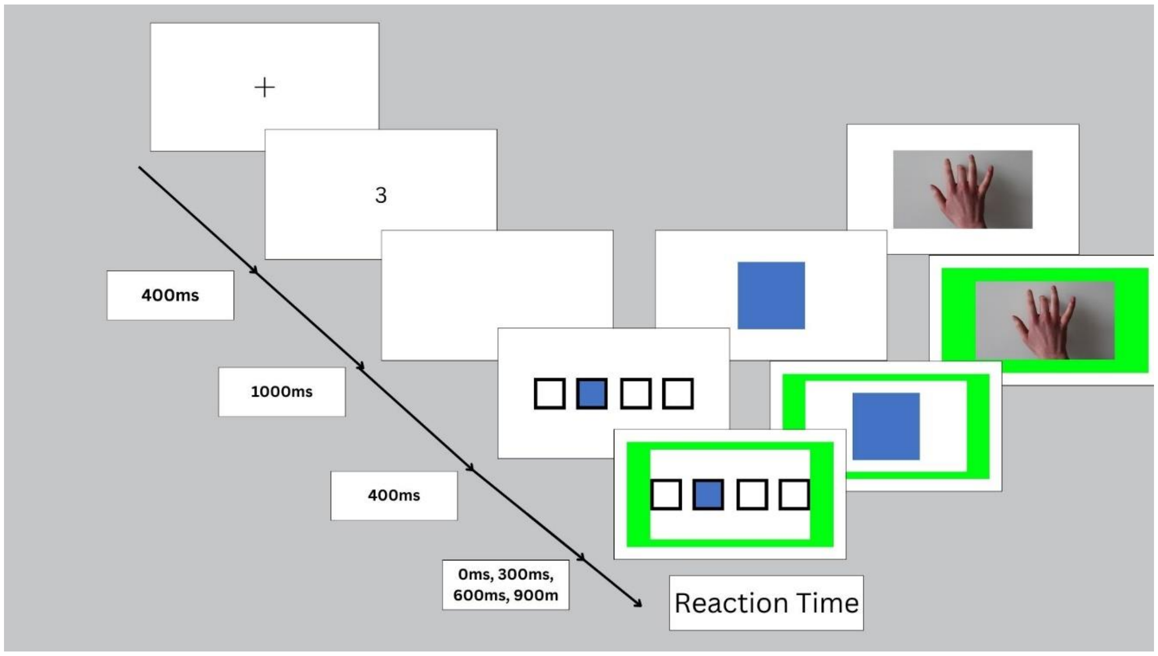
The reaction time task consisted of 4 sessions each including 120 trials. This was preceded by 8 familiarization trials. The exact timeline of a trial is illustrated in Figure 1. In each trial, participants fixated first on a fixation mark – X – for 400ms. Then they were shown the first stimulus, namely a number between 1 and 4 for 1000ms. Each of these numbers corresponded to a different key: 1 to H, 2 to J, 3 to K, and 4 to L. Participants were instructed to remember this number but do nothing yet until a go-signal appeared.

Next, the fixation mark occurred again for 400ms to ensure feature integration after which the second stimulus appeared which participants were instructed to ignore while waiting to for the go-signal. For this second stimulus, there were three conditions: the posture, the spatial, and the control condition. In the posture condition, one of four photos of a hand was shown, 70% of which corresponded to the hand posture they would have to assume to press the key corresponding to the number shown in the beginning and 30% of which showed a non-corresponding hand posture. Similarly, in the spatial condition, they were shown four squares one of which was highlighted. 70% of the time that square was consistent with the previously activated key and the other 30% it was inconsistent with it. In the control condition, random shapes – a square, a circle, a triangle, or a bar – were shown in the centre of the screen.

The interstimulus intervals (ISI; i.e., the duration between the condition stimulus and the go-signal) were either 0ms, 300ms, 600ms, or 900ms. The subsequent go-signal consisted of a green background while the second stimulus remained on the screen. Upon seeing the signal, participants had 3000ms to respond and press the key corresponding to the number they saw initially. Error trials were rerun.

Figure 1

Timeline of a trial



Note. Sequence of stimuli as they appeared to participants. At fourth place, the three experimental conditions are shown from left to right: the spatial, the control, and the posture condition.

Procedure

Participants were led into a quiet room where each of them filled out a demographics questionnaire and an online informed consent form outlining their voluntary participation, data management procedures, the study's objectives, and their rights, including the option to withdraw at any time. After filling out both, the researcher gave them an instruction sheet and explained the experiment after which the participants read the instructions on their screen. Once all doubts were cleared up, the participants were left alone to start the task. Furthermore, they were asked to give their phone to the researcher to avoid distractions. All participants took part in all conditions. In between sessions, participants had a minute of break during which they could do as they pleased, most looked out of the window to relax their eyes. The experiment lasted about 40 minutes. When the task ended, they were asked to report any issues they experienced while completing the experiment after which they were allowed to leave.

Analysis

The main dependent variable was reaction time (RT) of accurately pressed keys while errors acted as a second dependent variable to account for speed-accuracy trade-off. A key was pressed accurately if the previously instructed key was pressed after the go-signal appeared. The independent variables included Primes (Symbol, Posture, Spatial location), Interstimulus Intervals (ISI; 0, 300, 600, 900), and Correspondence (corresponding, non-corresponding). To answer the research question, several analyses were run in RStudio. First, the data was cleaned: One participant was removed as their RT-scores were too slow for further analysis, then the mean RT for each condition (0ms spatial corresponding, 300ms posture non-corresponding, etc.) were computed for each participant. For errors, first the proportions of each condition were calculated per participant after which arcsine transformation was conducted to perform an ANOVA (Winer et al. 1991). For the ANOVAs the Afex package in R was used. For p-values below α=.05, statistical significance was assumed. Greenhouse-Geisser correction was applied to the p-values of the *F* tests when Mauchly's sphericity was significant. The R-script can be found in Appendix B.

Results

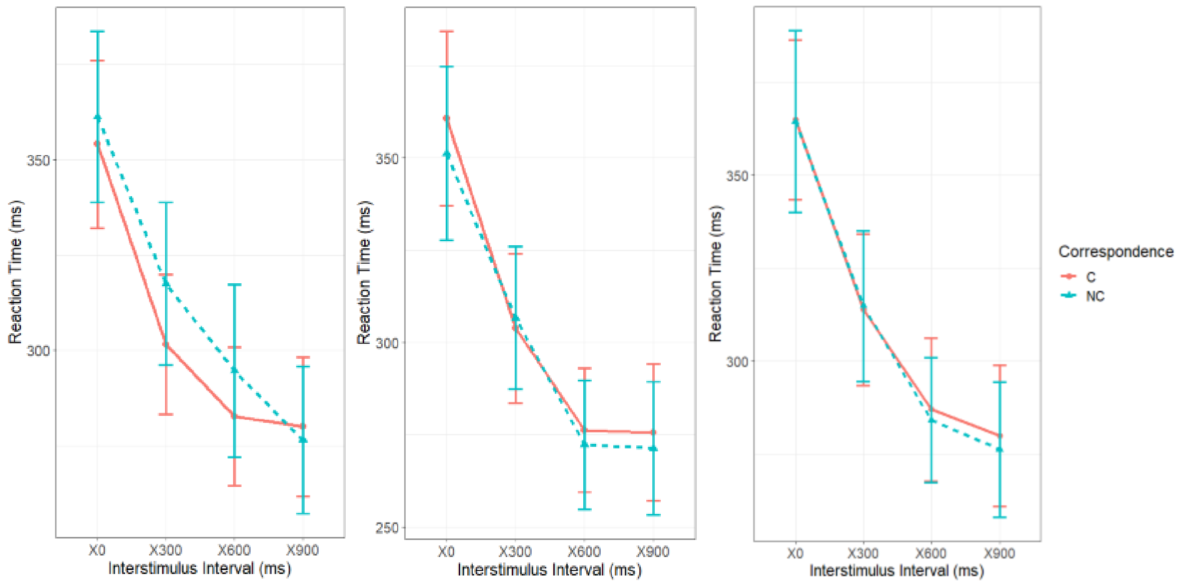
Reaction Times

RTs were analysed with a 2 (Correspondence: yes/no) x 4 (ISI: 0, 300, 600, 900ms) x 3 (Prime: Symbol, Hand posture photo, Spatial location) repeated-measures design ANOVA and – as this a within-subject design – the variable Subject was treated as a random effect.

A Correspondence x Prime interaction, *F*(2,62)=4.79, *p*=.01, η_p^2=.13, showed the participants responded faster in the spatial corresponding condition than in the spatial non-corresponding one (305 vs 312ms), *t*(31)=-2.01, *p*=.05, CI[-15.82, .122]. Figure 2 illustrates that this disparity in RTs occurred specifically at interstimulus intervals (ISI) of 300ms and 600ms which was supported by a planned comparison within the spatial condition across the 300ms and 600ms conditions, *t*(31)=-2.86, *p*=.008, CI[-47.6, -7.96]. There were no significant differences in RTs between the posture corresponding and non-corresponding conditions, *t*(31)=1.18, *p*=.25, CI[-2.71, 10.13], nor between the control corresponding and non-corresponding one, *t*(31)=.58, *p*=.57, CI[-3.83, 6.86].

Figure 2

Mean RTs in all Prime Conditions Divided by Correspondence Separately across the Different ISIs



Note. From left to right: the spatial, posture, and control conditions. Error bars show a 95% confidence interval.

Further, Interstimulus Intervals, ε=.65, F(1.95,60.45)=159.95, p<.001, η_p^2=.84, indicated a decrease of RT as the intervals grew longer with the slowest responses in the 0ms condition, gradually faster ones in the 300ms and 600ms conditions, and the fastest in the 900ms condition (359ms vs 310ms vs 283ms vs 277ms).

Additionally, Prime, F(2,62)=11.65, p<.001, η_p^2=.27, suggested participants were slowest in the control condition, slightly faster in the spatial, and fastest in the posture condition (311 vs 309 vs 302). Correspondence, F(1,31)=0.141, p=.7, on its own did not render significant effects.

Errors

Arcsine transformed proportions of the sums of accuracy and timing errors were analysed with the same 2x4x3 mixed ANOVA as the RTs. Interstimulus Intervals, ε=.67, F(2.01,62.31)=39.32, p<.001, η_p^2=.56, showed the error rate seemed to increase the longer the intervals between stimuli were (0ms=1% vs 300ms=3% vs 600ms=6% vs 900ms=8%). Further, Prime, F(2,62)=13.85, p<.001, η_p^2 = .31, indicated that the least errors were observed in the control condition, the spatial and posture condition rendered slightly more (3% vs 5% vs 6%). The Correspondence main effect, ε=.76, F(4.56,141.36)=39.32, p<.001, η_p^2=.73, implied participants made fewer mistakes when the conditions corresponded to the initial stimulus than when it did not correspond (4.5% vs 4.8%). Additionally, Interstimulus Interval x Prime interaction, F(6,186)=5.57, p<.001, η_p^2=.15, showed that the increase in errors with the length of the intervals holds true across all conditions. Lastly, Correspondence x Interstimulus Interval x Prime interaction, F(6,186)=2.79, p=.01, η_p^2=.08, was found.

Discussion

The current experiment investigated whether keypress movements are coded as hand postures, spatial locations, or both. For these purposes, participants performed a reaction time (RT) task in which they retained and pressed certain keys. Between the instruction and action of pressing the key, they viewed one of three different stimuli: Either a picture of a hand posture, one of four squares – one of which was highlighted to indicate the spatial location of a key –, or a picture of a random symbol – the control condition. The hand postures and relative spatial location of the squares were either congruent or incongruent with the keys they needed to press.

According to feature binding theory, if either represents such a movement in memory, the RTs should increase if they are incongruent (Ding et al., 2017; Hommel, 2004; Stoet & Hommel, 2002). Thus, the first hypothesis stated that inconsistent hand postures should render longer RTs than consistent ones. This was not supported by the data. The second hypothesis stated that inconsistent spatial locations should render longer RTs than consistent ones. This appeared to be the case.

As expected, in the spatial condition, participants responded faster in the corresponding than in the non-corresponding condition at ISIs of 300ms and 600ms. This supports the notion that keypress movement is represented as spatial locations. To break it down, the image of a square signifying a consistent spatial location with the subsequent key to press activates a certain feature in memory which facilitates movement (Stoet & Hommel, 2002; Vogt et al., 2003). An inconsistent spatial location, however, will interfere once it is integrated as a ‘wrong’ location representation is taking up the feature needed to press the instructed key. The integration appeared to take place at the ISIs of 300ms and 600ms, a time-line consistent with both Vogt et al. (2003) who argue integration may take place at 300-700ms with a peak at 500ms and Stoet and Hommel (2002) who indicate it begins after 250-500ms. Furthermore, this finding is consistent with research highlighting the importance of spatial memory in action planning (Avraamides & Kelly, 2008) as well as well-known psychological phenomena, such as the Simon effect (i.e. stimuli which spatially correspond to a response facilitate this response), which support the presence of spatial representations for movement in memory (Hommel, 2011, 2019; Simon & Rudell, 1967). Furthermore, studies on cortical action maps indicate the existence of spatial maps which interact with maps of body parts and body postures to code and initiate actions (Graziano, 2016).

In the posture condition, RTs did not differ between the corresponding and non-corresponding conditions which contradicts the expectation that hand postures code for keypress movements. One potential explanation may lie in the nature of the task itself. According to the *intentional weighting principle* within Theory of Event Coding (TEC; Hommel, 2019; Memelink & Hommel, 2013), features that are assumed to be more relevant to the task at hand are more likely used for coding it than irrelevant features. When typing we usually do not look at our hands and thus, may not consider what specific postures are relevant for pressing specific keys. The specific location of a key may be far more important; this is even considered in the design of a keyboard through e.g., nobs one can orientate oneself on to know the location of other keys. Especially, in a task in which one has to press four keys on a horizontal line, it may be more relevant to know where these keys are than what posture is required to press this key. However, at this stage this reasoning is highly theoretical and would need to be researched further.

An alternative explanation is that the hand postures were too difficult to recognise. As was already expected before starting the experiment, one participant indicated that they did not really grasp which posture the hand was doing before they had to respond. Thus, maybe a task with more distinct hand postures may be needed in the future. For this speaks, a great body of research indicating the existence of (hand) posture features when it comes to (hand) movement (Graziano, 2016; Rosenbaum, 2017; Rosenbaum et al., 2001; Vogt et al., 2003). Therefore, to dismiss hand postures outright as possible features coding for keypress or hand movements at this stage would be premature.

Thus, one limitation of this study may have been the task design. The aim was to ascertain features along which keypress movements are represented, however, whether these findings are generalisable to (hand) movements would need to be researched while widening the movement scope and moving to tasks in which participants would be more likely consider their (hand) postures or perform more distinct hand postures. One such task could be playing the piano which requires participants to retain certain hand postures (Liu et al., 2023, Ogawa et al., 2019). A second one may have been the short practice period, which only consisted of 8 trials. While the researcher tried to instruct the participants as well as possible, some reported afterwards that they still needed a while to understand what exactly they had to do. This may have been due to language barriers in some cases as the participants' English levels differed and at times participants may have indicated that they understood everything when this was not actually the case. This could be mitigated by more practice trials as the task was reported to be quite intuitive once the instructions were understood.

In conclusion, the current study found support that keypress movements are represented as spatial features in memory as incongruent spatial relations interfered with the instructed action. This aligns with previous research emphasising the role of spatial memory in action planning, psychological phenomena such as the Simon effect, and the existence of spatial maps along the cortex's action maps which are involved in generating and coding movement. Although no support was found that hand postures code keypress movements, this might be due to the specific task design and context. The typing component of this experiment potentially made spatial location more pertinent than hand posture. Furthermore, the hand postures may have not been distinct enough from one another to distinguish between them. However, to ascertain whether this is the case, further research into this topic is necessary.

References

Avraamides, M. N., & Kelly, J. W. (2008). Multiple systems of spatial memory and action. *Cognitive Processing*, *9*(2), 93–106. https://doi.org/10.1007/S10339-007-0188-5

Ding, S., Meng, L., Han, Y., & Xue, Y. (2017). A Review on Feature Binding Theory and Its Functions Observed in Perceptual Process. *Cognitive Computation*, *9*(2), 194–206. https://doi.org/10.1007/s12559-016-9446-0

Graziano, M. S. A. (2016). Ethological Action Maps: A Paradigm Shift for the Motor Cortex. *Trends in Cognitive Sciences*, *20*(2), 121–132. https://doi.org/10.1016/J.TICS.2015.10.008

Hommel, B. (2004). Event files: feature binding in and across perception and action. *Trends in Cognitive Sciences*, *8*(11), 494–500. https://doi.org/10.1016/J.TICS.2004.08.007

Hommel, B. (2011). The Simon effect as tool and heuristic. *Acta Psychologica*, *136*(2), 189–202. https://doi.org/10.1016/J.ACTPSY.2010.04.011

Hommel, B. (2019). Theory of Event Coding (TEC) V2.0: Representing and controlling perception and action. *Attention, Perception, and Psychophysics*, *81*(7), 2139–2154. https://doi.org/10.3758/S13414-019-01779-4

Liu, R., Wu, E., Liao, C., Nishioka, H., Furuya, S., & Koike, H. (2023). PianoSyncAR: Enhancing Piano Learning through Visualizing Synchronized Hand Pose Discrepancies in Augmented Reality. *2023 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 859-868. https://doi.org/10.1109/ISMAR59233.2023.00101.

Psychology Software Tools. (2023). E-Prime® | Psychology Software Tools. Retrieved from https://www.pstnet.com

Ogawa, K., Mitsui, K., Imai, F., & Nishida, S. (2019). Long-term training-dependent representation of individual finger movements in the primary motor cortex. *NeuroImage*, 202. https://doi.org/10.1016/j.neuroimage.2019.116051.

Rosenbaum, D. A. (2017). *Knowing Hands* Cambridge University Press. https://doi.org/10.1017/9781316148525

Rosenbaum, D. A., Meulenbroek, R. J., Vaughan, J., & Jansen, C. (2001). Posture-based motion planning: Applications to grasping. *Psychological Review*, *108*(4), 709-734. https://doi.org/10.1037/0033-295x.108.4.709

Simon, J. R., & Rudell, A. P. (1967). Auditory S-R compatibility: The effect of an irrelevant cue on information processing. *Journal of Applied Psychology*, *51*(3), 300–304. https://doi.org/10.1037/H0020586

Stoet, G., & Hommel, B. (2002). Interaction between feature binding in perception and action. In *Common Mechanisms in Perception and Action* (pp. 538–552). Oxford University PressOxford. https://doi.org/10.1093/oso/9780198510697.003.0026

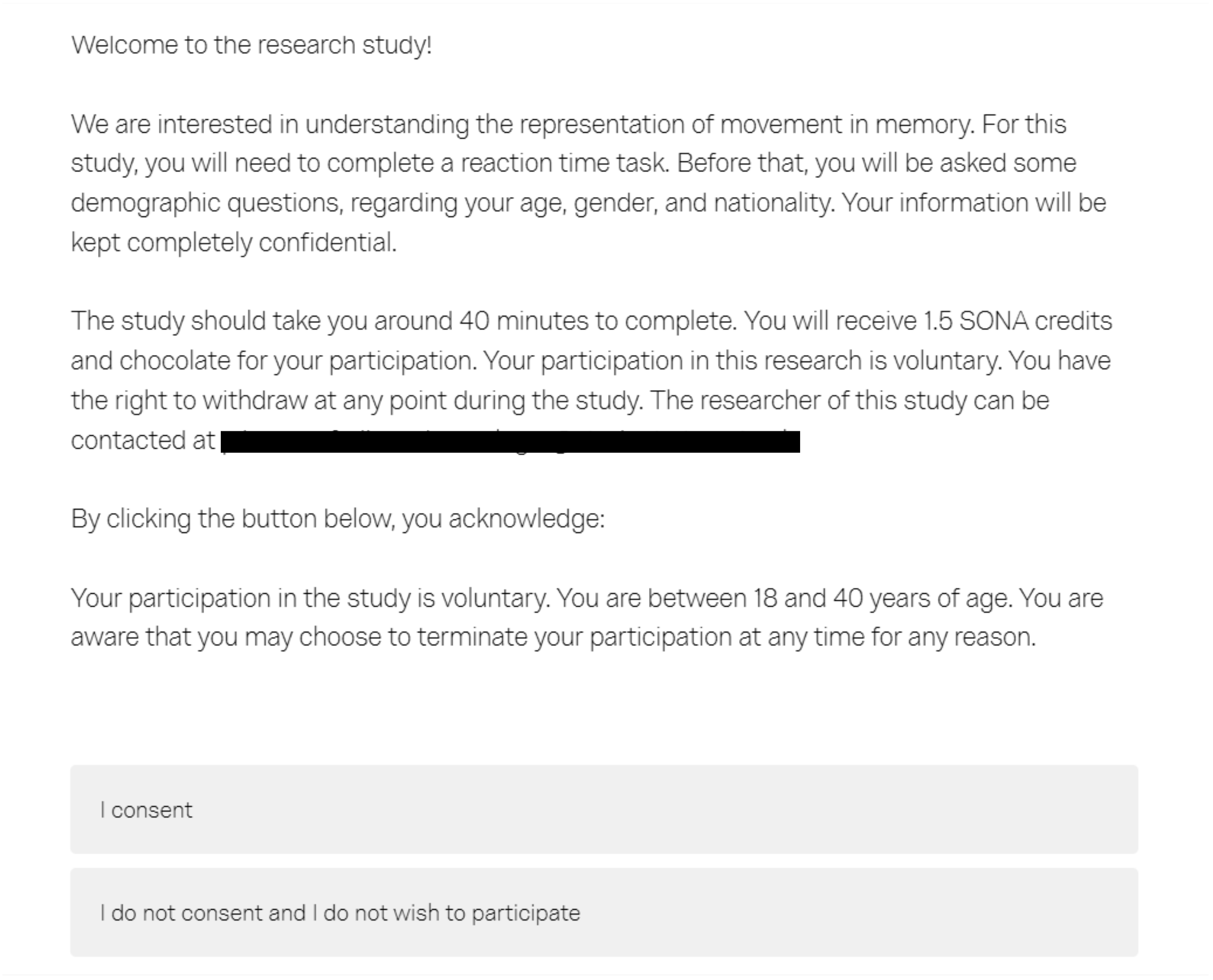
Verwey, W. B. (2023). Chord skill: learning optimized hand postures and bimanual coordination. *Experimental Brain Research*, *241*(6), 1643–1659. https://doi.org/10.1007/S00221-023-06629-2

Verwey, W. B., Groen, E. C., & Wright, D. L. (2016). The stuff that motor chunks are made of: Spatial instead of motor representations? *Experimental Brain Research*, *234*(2), 353–366. https://doi.org/10.1007/S00221-015-4457-8

Vogt, S., Taylor, P., & Hopkins, B. (2003). Visuomotor priming by pictures of hand postures: perspective matters. *Neuropsychologia*, *41*(8), 941–951. https://doi.org/10.1016/S0028-3932(02)00319-6

Winer, B. J., Brown, D. R., & Michels, K. M. (1991). Statistical Principles in Experimental Design. McGraw-Hill Humanities, Social Sciences & World Languages.

Appendix A



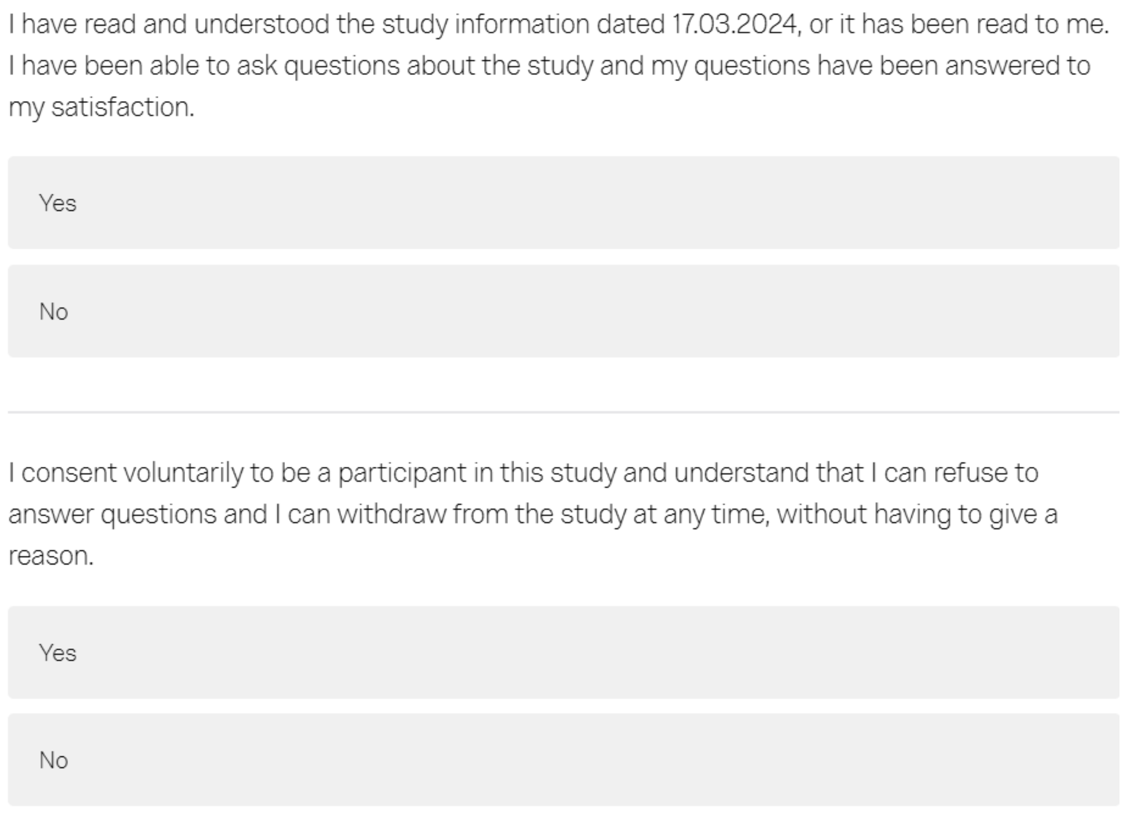
Welcome to the research study!

We are interested in understanding the representation of movement in memory. For this study, you will need to complete a reaction time task. Before that, you will be asked some demographic questions, regarding your age, gender, and nationality. Your information will be kept completely confidential.

The study should take you around 40 minutes to complete. You will receive 1.5 SONA credits and chocolate for your participation. Your participation in this research is voluntary. You have the right to withdraw at any point during the study. The researcher of this study can be contacted at [redacted]

By clicking the button below, you acknowledge:

Your participation in the study is voluntary. You are between 18 and 40 years of age. You are aware that you may choose to terminate your participation at any time for any reason.



I have read and understood the study information dated 17.03.2024, or it has been read to me. I have been able to ask questions about the study and my questions have been answered to my satisfaction.

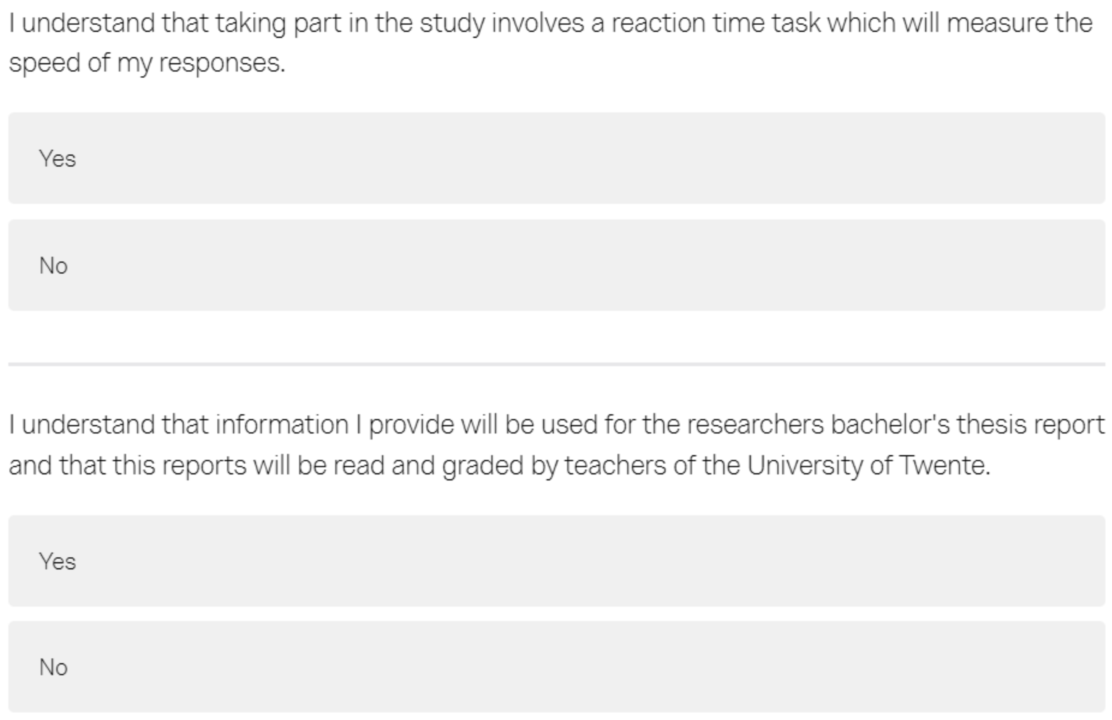
Yes

No

I consent voluntarily to be a participant in this study and understand that I can refuse to answer questions and I can withdraw from the study at any time, without having to give a reason.

Yes

No



I understand that taking part in the study involves a reaction time task which will measure the speed of my responses.

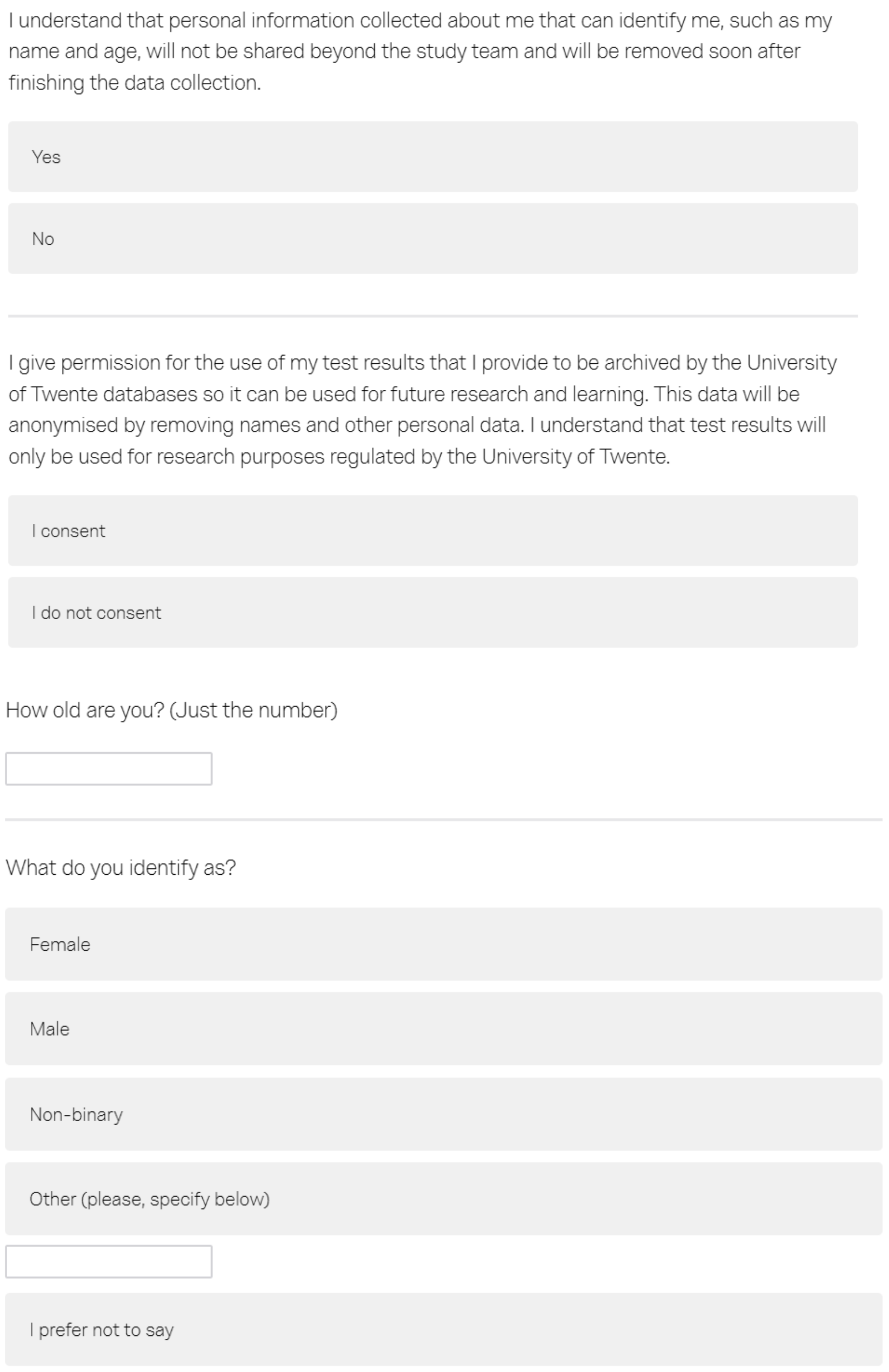
Yes

No

I understand that information I provide will be used for the researchers bachelor's thesis report and that this reports will be read and graded by teachers of the University of Twente.

Yes

No



I understand that personal information collected about me that can identify me, such as my name and age, will not be shared beyond the study team and will be removed soon after finishing the data collection.

Yes

No

I give permission for the use of my test results that I provide to be archived by the University of Twente databases so it can be used for future research and learning. This data will be anonymised by removing names and other personal data. I understand that test results will only be used for research purposes regulated by the University of Twente.

I consent

I do not consent

How old are you? (Just the number)

What do you identify as?

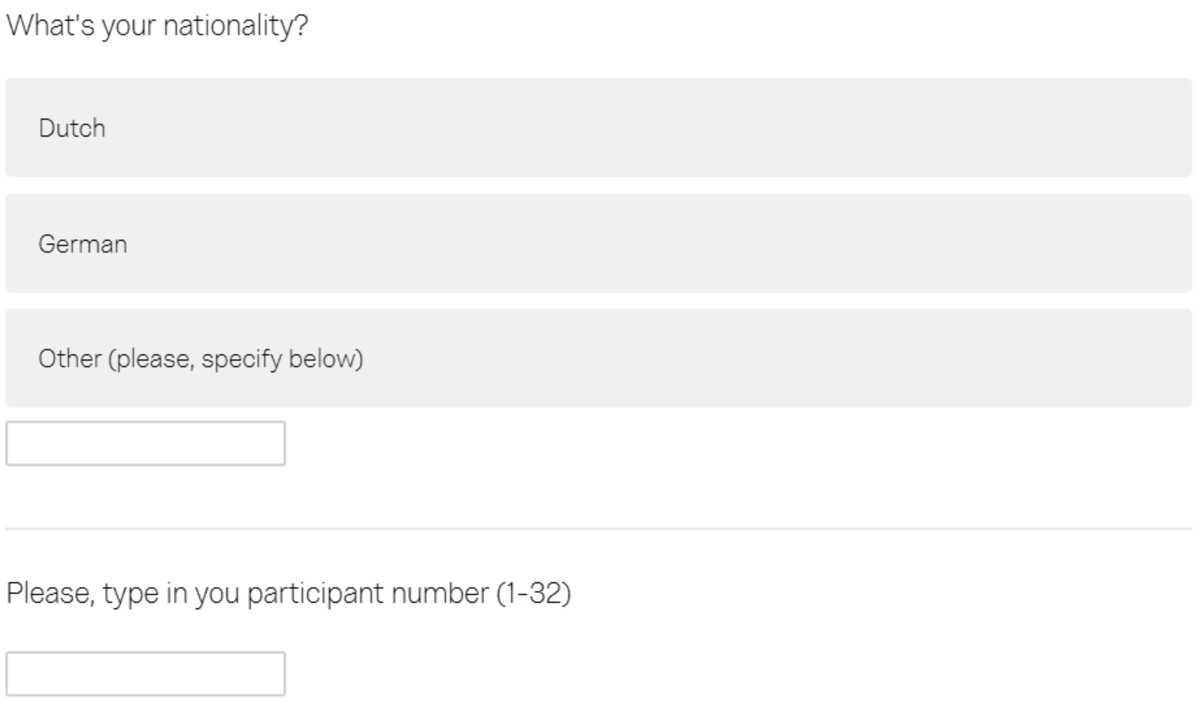
Female

Male

Non-binary

Other (please, specify below)

I prefer not to say



What's your nationality?

- Dutch
- German
- Other (please, specify below)

Please, type in you participant number (1-32)

Appendix B

```
#Pia Boeselager
#Bachelor thesis
#08/04/2024 - 26/07/2024

update.packages()
install.packages("afex", dependencies = TRUE)
install.packages("remotes")

install.packages("lme4")
install.packages("car")
install.packages("pbkrtest")
install.packages("rlang")
install.packages("estimability")
install.packages("multcomp")
install.packages("emmeans", dependencies = TRUE)
install.packages("contrast")

library(tidyverse)
library(readxl)
library(stats)
library(ggpubr)
library(foreign)
library(dplyr)
library(ggplot2)
library(broom)
library(afex)
library(multcomp)
library(emmeans)
library(car)
library(contrast)

#loading DEMOGRAPHIC DATA
Demo_data <- read_excel("Demographics_numbers.xlsx")

#mean age + sd age
david <- as.numeric(Demo_data$Q9)
mean_value <- mean(david, na.rm = TRUE)
sd_value <- sd(david, na.rm = FALSE)

#gender distribution
hans <- as.numeric(Demo_data$Q10)
summary_table <- table(hans)
print(summary_table)

#nationality
jane <- as.numeric(Demo_data$Q11)
hilde <- table(jane)
print(hilde)
```

```
#loading EXPERIMENT DATA
#RT as DV

Rick <- read_excel("Hope.xlsx")

Finally <- separate(Rick,
                   col = Prime,
                   into = c("C_NC", "Number", "Category"),
                   sep = "_",
                   extra = "merge",
                   remove = FALSE)


#Anova
model <- aov_car(RT ~ C_NC * Number * Category +
Error(Subject/(C_NC*Number*Category)), data = Finally)
summary(model)

#partial eta square
effectsize::eta_squared(model)

#planned contrast spatial 300ms + 600ms
# Get the estimated marginal means (EMMs) for the C_NC variable within
specific conditions
emmeans_model <- emmeans(model, ~ C_NC | Category * Number)

# Inspect the EMMs object to ensure it includes the desired levels
print(emmeans_model)

# Get EMMs for specific interaction conditions
specific_conditions <- subset(emmeans_model, Category == "Spatial" & (Number
== "X300" | Number == "X600"))

# Define the custom contrast
custom_contrast <- list("C_vs_NC_300_and_600" = c(1, -1, 1, -1)) # Combining C
vs NC at Number = 300 and 600

# Apply the custom contrast and print the results
combined_contrasts <- contrast(specific_conditions, custom_contrast)
summary(combined_contrasts, infer = c(TRUE, TRUE))



#contrast analysis
#interaction C_NC * Category
emm <- emmeans(model, ~ C_NC * Category)

# Define contrasts for comparing C_NC levels within each Category level
contrasts <- list(
  "C_NC1 vs C_NC2 in Cat1" = c(1, 0, -1, 0, 0, 0), #C_NC in control
  "C_NC1 vs C_NC2 in Cat2" = c(0, 1, 0, -1, 0, 0), #C_NC in posture
  "C_NC1 vs C_NC2 in Cat3" = c(0, 0, 0, 0, 1, -1) #C_NC in spatial
)
```

```
# Run the contrasts
contrast_results <- contrast(emm, contrasts)
summary(contrast_results)

#contrast numbers and categories
emma <- emmeans(model, ~ Number)
emmi <- emmeans(model, ~ Category)

# Define contrasts for the 'Number' variable (4 levels)
contrast_number <- list(
  Number_contrast1 = c(1, -1, 0, 0),  # Comparing level 1 (0) with level 2
(300)
  Number_contrast2 = c(1, 0, -1, 0),   # Comparing level 1(0) with level 3
(600)
  Number_contrast3 = c(1, 0, 0, -1),   # Comparing level 1 (0) with level 4
(900)
  Number_contrast4 = c(0, 1, -1, 0),   # Comparing level 2 (300)with level 3
(600)
  Number_contrast5 = c(0, 1, 0, -1),   # Comparing level 2 (300) with level 4
(900)
  Number_contrast6 = c(0, 0, 1, -1)    # Comparing level 3 (600) with level 4
(900)
)

# Define contrasts for the 'Category' variable (3 levels)
contrast_category <- list(
  Category_contrast1 = c(1, -1, 0),    # Comparing level 1 (control) with
level 2 (Posture)
  Category_contrast2 = c(1, 0, -1),     # Comparing level 1 (control) with
level 3 (Spatial)
  Category_contrast3 = c(0, 1, -1)      # Comparing level 2 (posture) with
level 3 (spatial)
)

# Conduct the contrast analysis for 'Number' variable
contrast_result_number <- contrast(emma, contrast_number)

# Conduct the contrast analysis for 'Category' variable
contrast_result_category <- contrast(emmi, contrast_category)

# View the results
contrast_result_number
contrast_result_category




#MEAN OF DIF CONDITIONS
#Correspondence and Posture
values_to_delete_category <- c("Control", "Spatial")
value_to_delete_c_nc <- "NC"
CP <- Finally[!(Finally$Category %in% values_to_delete_category), ]
```

```
CPF <- CP[!(CP$C_NC %in% value_to_delete_c_nc), ]
mean(CP$RT, na.rm = TRUE)

value_to_delete_c <- "C"
NCP <- CP[!(CP$C_NC %in% value_to_delete_c), ]
mean(NCP$RT, na.rm = TRUE)

#Correspondence Spatial
values_to_delete_category <- c("Control", "Posture")
value_to_delete_c_nc <- "NC"
SP <- Finally[!(Finally$Category %in% values_to_delete_category), ]
SPF <- SP[!(SP$C_NC %in% value_to_delete_c_nc), ]
mean(SPF$RT, na.rm = TRUE)

value_to_delete_c <- "C"
NSP <- SP[!(SP$C_NC %in% value_to_delete_c), ]
mean(NSP$RT, na.rm = TRUE)

#Correspondence Control
values_to_delete_category <- c("Posture", "Spatial")
value_to_delete_c_nc <- "NC"
CC <- Finally[!(Finally$Category %in% values_to_delete_category), ]
CCF <- CC[!(CC$C_NC %in% value_to_delete_c_nc), ]
mean(CCF$RT, na.rm = TRUE)

value_to_delete_c <- "C"
NCC <- CC[!(CC$C_NC %in% value_to_delete_c), ]
mean(NCC$RT, na.rm = TRUE)


#interstimulus interval 0ms
values_to_delete_number <- c(300, 600, 900)
II0 <- Finally[!(Finally$Number %in% values_to_delete_number), ]
mean(II0$RT, na.rm = TRUE)

#interstimulus interval 300ms
values_to_delete_number <- c(0, 600, 900)
II300 <- Finally[!(Finally$Number %in% values_to_delete_number), ]
mean(II300$RT, na.rm = TRUE)

#interstimulus interval 600ms
values_to_delete_number <- c(300, 0, 900)
II600 <- Finally[!(Finally$Number %in% values_to_delete_number), ]
mean(II600$RT, na.rm = TRUE)

#interstimulus interval 900ms
values_to_delete_number <- c(300, 600, 0)
II900 <- Finally[!(Finally$Number %in% values_to_delete_number), ]
mean(II900$RT, na.rm = TRUE)

#prime spatial
mean(SP$RT, na.rm = TRUE)
```

```
#prime posture
mean(CP$RT, na.rm = TRUE)

#prime control
values_to_delete_category <- c("Spatial", "Posture")
PC <- Finally[!(Finally$Category %in% values_to_delete_category), ]
mean(PC$RT, na.rm = TRUE)


#plot it
#Spatial
# Get the estimated marginal means (EMMs) for the interaction of interest
emmeans_results1 <- emmeans(model, ~ C_NC * Number | Category)
# Convert the emmeans results to a data frame
emmeans_df <- as.data.frame(emmeans_results1)
# Subset specific conditions
specific_conditions1 <- subset(emmeans_df, Category == "Spatial")
# Extract necessary columns for plotting
plot_spa_data <- specific_conditions1[, c("C_NC", "Number", "emmean",
"lower.CL", "upper.CL")]
colnames(plot_spa_data) <- c("C_NC", "Number", "Mean_RT", "Lower_CI",
"Upper_CI")
# Create the interaction plot with ggplot2
ggplot(data = plot_spa_data, aes(x = factor(Number), y = Mean_RT, group =
C_NC, color = C_NC)) +
  geom_line(aes(linetype = C_NC), size = 1.3) +
  geom_point(aes(shape = C_NC), size = 2.5) +
  geom_errorbar(aes(ymin = Lower_CI, ymax = Upper_CI), width = 0.2, size =
1.1) +
  labs(x = "Interstimulus Interval (ms)", y = "Reaction Time (ms)", color =
"Correspondence", linetype = "Correspondence", shape = "Correspondence", size
= 2) +
  theme_bw() +
  theme(
    legend.position = "right",
    plot.title = element_text(size = 16, face = "bold"),
    axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14),
    axis.text = element_text(size = 12),
    legend.title = element_text(size = 14),
    legend.text = element_text(size = 12)
  )

#Posture
# Get the estimated marginal means (EMMs) for the interaction of interest
emmeans1 <- emmeans(model, ~ C_NC * Number | Category)
# Convert the emmeans results to a data frame
emmeans_dfp <- as.data.frame(emmeans1)
# Subset specific conditions
specific_conditionsp <- subset(emmeans_dfp, Category == "Posture")
# Extract necessary columns for plotting
plot_post_data <- specific_conditionsp[, c("C_NC", "Number", "emmean",
"lower.CL", "upper.CL")]
```

```
colnames(plot_post_data) <- c("C_NC", "Number", "Mean_RT", "Lower_CI",
"Upper_CI")
# Create the interaction plot with ggplot2
ggplot(data = plot_post_data, aes(x = factor(Number), y = Mean_RT, group =
C_NC, color = C_NC)) +
  geom_line(aes(linetype = C_NC), size = 1.3) +
  geom_point(aes(shape = C_NC), size = 2.5) +
  geom_errorbar(aes(ymin = Lower_CI, ymax = Upper_CI), width = 0.2, size =
1.1) +
  labs(x = "Interstimulus Interval (ms)", y = "Reaction Time (ms)", color =
"Correspondence", linetype = "Correspondence", shape = "Correspondence", size
= 2) +
  theme_bw() +
  theme(
    legend.position = "right",
    plot.title = element_text(size = 16, face = "bold"),
    axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14),
    axis.text = element_text(size = 12),
    legend.title = element_text(size = 14),
    legend.text = element_text(size = 12)
  )


#Control
# Get the estimated marginal means (EMMs) for the interaction of interest
emmeans2 <- emmeans(model, ~ C_NC * Number | Category)
# Convert the emmeans results to a data frame
emmeans_dfc <- as.data.frame(emmeans2)
# Subset specific conditions
specific_conditionsc <- subset(emmeans_dfc, Category == "Control")
# Extract necessary columns for plotting
plot_con_data <- specific_conditionsc[, c("C_NC", "Number", "emmean",
"lower.CL", "upper.CL")]
colnames(plot_con_data) <- c("C_NC", "Number", "Mean_RT", "Lower_CI",
"Upper_CI")
# Create the interaction plot with ggplot2
ggplot(data = plot_con_data, aes(x = factor(Number), y = Mean_RT, group =
C_NC, color = C_NC)) +
  geom_line(aes(linetype = C_NC), size = 1.3) +
  geom_point(aes(shape = C_NC), size = 2.5) +
  geom_errorbar(aes(ymin = Lower_CI, ymax = Upper_CI), width = 0.2, size =
1.1) +
  labs(x = "Interstimulus Interval (ms)", y = "Reaction Time (ms)", color =
"Correspondence", linetype = "Correspondence", shape = "Correspondence", size
= 2) +
  theme_bw() +
  theme(
    legend.position = "right",
    plot.title = element_text(size = 16, face = "bold"),
    axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14),
    axis.text = element_text(size = 12),
    legend.title = element_text(size = 14),
    legend.text = element_text(size = 12)
  )



#why is posture the fastest?
# prime across isis
#try 1
par(mar = c(5, 5, 4, 10))  # Adjust the right margin to make room for the
legend

interaction.plot(x.factor = Finally$Number,
                 trace.factor = Finally$Category,
                 response = Finally$RT,
                 type = "b",
                 legend = TRUE,
                 xlab = "ISI",
                 ylab = "RT")

#try again
# Get the estimated marginal means (EMMs) for the interaction of interest
emmeansag <- emmeans(model, ~ Category * Number)
# Convert the emmeans results to a data frame
emmeans_dfag <- as.data.frame(emmeansag)

# Extract necessary columns for plotting
plot_ag_data <- emmeans_dfag[, c("Category", "Number", "emmean", "lower.CL",
"upper.CL")]
colnames(plot_ag_data) <- c("Category", "Number", "Mean_RT", "Lower_CI",
"Upper_CI")
# Create the interaction plot with ggplot2
ggplot(data = plot_ag_data, aes(x = factor(Number), y = Mean_RT, group =
Category, color = Category)) +
  geom_line(aes(linetype = Category), size = 1.3) +
  geom_point(aes(shape = Category), size = 2.5) +
  geom_errorbar(aes(ymin = Lower_CI, ymax = Upper_CI), width = 0.2, size =
1.1) +
  labs(x = "Interstimulus Interval (ms)", y = "Reaction Time (ms)", color =
"Prime", linetype = "Prime", shape = "Prime", size = 2) +
  theme_bw() +
  theme(
    legend.position = "right",
    plot.title = element_text(size = 16, face = "bold"),
    axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14),
    axis.text = element_text(size = 12),
    legend.title = element_text(size = 14),
    legend.text = element_text(size = 12)
  )
```

```
#all of it
emmeans_results3 <- emmeans(model, ~ C_NC * Number * Category)

# Convert the emmeans results to a data frame
emmeans_dfy <- as.data.frame(emmeans_results3)

# Extract necessary columns for plotting
plot_datay <- emmeans_dfy[, c("C_NC", "Number", "Category", "emmean",
"lower.CL", "upper.CL")]
colnames(plot_datay) <- c("C_NC", "Number", "Category", "Mean_RT", "Lower_CI",
"Upper_CI")

ggplot(data = plot_datay, aes(x = factor(Number), y = Mean_RT, group =
interaction(Category, C_NC), color = Category)) +
  geom_line(aes(linetype = C_NC), size = 1.3) +
  geom_point(aes(shape = C_NC), size = 2.5) +
  labs(x = "Interstimulus Interval (ms)", y = "Reaction Time", color =
"Prime", linetype = "Correspondence", shape = "Correspondence") +
  theme_bw()+
  theme(
    legend.position = "right",
    plot.title = element_text(size = 16, face = "bold"),
    axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14),
    axis.text = element_text(size = 12),
    legend.title = element_text(size = 14),
    legend.text = element_text(size = 12)
  )




#ERROR as DV
Mick <- read_excel("Error1.xlsx")

Nick <- separate(Mick,
                    col = Prime,
                    into = c("C_NC", "Number", "Category"),
                    sep = "_",
                    extra = "merge",
                    remove = FALSE)

#Anova
Tick <- aov_car(Arc_Error ~ C_NC * Number * Category +
Error(Subject/(C_NC*Number*Category)), data = Nick)
summary(Tick)

#partial eta square
effectsize::eta_squared(Tick)


#maybe not so necessary??
```

```
#Define contrast for C_NC
# Compute EMMs
emm_c_nc <- emmeans(Tick, specs = ~ C_NC)

# Pairwise contrasts
pairwise_c_nc <- pairs(emm_c_nc)
summary(pairwise_c_nc)

#Contrast analysis
emmo <- emmeans(Tick, ~ Number)
emme <- emmeans(Tick, ~ Category)


# Define contrasts for the 'Number' variable (4 levels)
contrast_numbers <- list(
  Number_contrast1 = c(1, -1, 0, 0),  # Comparing level 1 (0) with level 2
(300)
  Number_contrast2 = c(1, 0, -1, 0),   # Comparing level 1(0) with level 3
(600)
  Number_contrast3 = c(1, 0, 0, -1),   # Comparing level 1 (0) with level 4
(900)
  Number_contrast4 = c(0, 1, -1, 0),   # Comparing level 2 (300)with level 3
(600)
  Number_contrast5 = c(0, 1, 0, -1),   # Comparing level 2 (300) with level 4
(900)
  Number_contrast6 = c(0, 0, 1, -1)    # Comparing level 3 (600) with level 4
(900)
)

# Define contrasts for the 'Category' variable (3 levels)
contrast_categories <- list(
  Category_contrast1 = c(1, -1, 0),    # Comparing level 1 (control) with
level 2 (Posture)
  Category_contrast2 = c(1, 0, -1),     # Comparing level 1 (control) with
level 3 (Spatial)
  Category_contrast3 = c(0, 1, -1)      # Comparing level 2 (posture) with
level 3 (spatial)
)

# Conduct the contrast analysis for 'Number' variable
contrast_result_number <- contrast(emmo, contrast_numbers)

# Conduct the contrast analysis for 'Category' variable
contrast_result_category <- contrast(emme, contrast_categories)

# View the results
contrast_result_number
contrast_result_category

# Compute EMMs for the interaction between Number and Category
emm_interaction <- emmeans(Tick, specs = ~ Number:Category)

# Pairwise contrasts for the interaction
```

```
pairwise_interaction <- pairs(emm_interaction)
summary(pairwise_interaction)



#mean of dif conditions
#Spatial
values_to_delete_category <- c("Control", "Posture")
SC <- Nick[!(Nick$Category %in% values_to_delete_category), ]
mean(SC$Prop_Error, na.rm = TRUE)

#Posture
values_to_delete_category <- c("Control", "Spatial")
PC <- Nick[!(Nick$Category %in% values_to_delete_category), ]
mean(PC$Prop_Error, na.rm = TRUE)

#Control
values_to_delete_category <- c("Spatial", "Posture")
CC <- Nick[!(Nick$Category %in% values_to_delete_category), ]
mean(CC$Prop_Error, na.rm = TRUE)

#correspondence
value_to_delete_nc <- "NC"
CN <- Nick[!(Nick$C_NC %in% value_to_delete_nc), ]
mean(CN$Prop_Error, na.rm = TRUE)

value_to_delete_c <- "C"
nCN <- Nick[!(Nick$C_NC %in% value_to_delete_c), ]
mean(nCN$Prop_Error, na.rm = TRUE)

#interstimulus interval 0ms
values_to_delete_number <- c(300, 600, 900)
II0 <- Nick[!(Nick$Number %in% values_to_delete_number), ]
mean(II0$Prop_Error, na.rm = TRUE)

#interstimulus interval 300ms
values_to_delete_number <- c(0, 600, 900)
II300 <- Nick[!(Nick$Number %in% values_to_delete_number), ]
mean(II300$Prop_Error, na.rm = TRUE)

#interstimulus interval 600ms
values_to_delete_number <- c(300, 0, 900)
II600 <- Nick[!(Nick$Number %in% values_to_delete_number), ]
mean(II600$Prop_Error, na.rm = TRUE)

#interstimulus interval 900ms
values_to_delete_number <- c(300, 600, 0)
II900 <- Nick[!(Nick$Number %in% values_to_delete_number), ]
mean(II900$Prop_Error, na.rm = TRUE)

#cor isi0 post
values_to_delete_all <- c("C_0_Control", "C_0_Spatial",
                          "C_300_Control", "C_300_Posture", "C_300_Spatial",
                          "C_600_Control", "C_600_Posture", "C_600_Spatial",
                          "C_900_Control", "C_900_Posture", "C_900_Spatial",
                          "NC_0_Control", "NC_0_Posture", "NC_0_Spatial",
                          "NC_300_Control", "NC_300_Posture",
"NC_300_Spatial",
                          "NC_600_Control", "NC_600_Posture",
"NC_600_Spatial",
                          "NC_900_Control", "NC_900_Posture",
"NC_900_Spatial")
CISI0Post <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(CISI0Post$Prop_Error)

#ncor isi0 post
values_to_delete_all <- c("C_0_Control", "C_0_Posture","C_0_Spatial",
                          "C_300_Control", "C_300_Posture", "C_300_Spatial",
                          "C_600_Control", "C_600_Posture", "C_600_Spatial",
                          "C_900_Control", "C_900_Posture", "C_900_Spatial",
                          "NC_0_Control", "NC_0_Spatial",
                          "NC_300_Control", "NC_300_Posture",
"NC_300_Spatial",
                          "NC_600_Control", "NC_600_Posture",
"NC_600_Spatial",
                          "NC_900_Control", "NC_900_Posture",
"NC_900_Spatial")
NCISI0Post <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(NCISI0Post$Prop_Error)

#cor isi300 post
values_to_delete_all <- c("C_0_Control", "C_0_Posture","C_0_Spatial",
                          "C_300_Control", "C_300_Spatial",
                          "C_600_Control", "C_600_Posture", "C_600_Spatial",
                          "C_900_Control", "C_900_Posture", "C_900_Spatial",
                          "NC_0_Control", "NC_0_Posture", "NC_0_Spatial",
                          "NC_300_Control", "NC_300_Posture",
"NC_300_Spatial",
                          "NC_600_Control", "NC_600_Posture",
"NC_600_Spatial",
                          "NC_900_Control", "NC_900_Posture",
"NC_900_Spatial")
CISI300Post <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(CISI300Post$Prop_Error)

#ncor isi300 post
values_to_delete_all <- c("C_0_Control", "C_0_Posture","C_0_Spatial",
                          "C_300_Control", "C_300_Posture", "C_300_Spatial",
                          "C_600_Control", "C_600_Posture", "C_600_Spatial",
                          "C_900_Control", "C_900_Posture", "C_900_Spatial",
                          "NC_0_Control", "NC_0_Posture", "NC_0_Spatial",
                          "NC_300_Control", "NC_300_Spatial",
                          "NC_600_Control", "NC_600_Posture",
"NC_600_Spatial",
                          "NC_900_Control", "NC_900_Posture",
"NC_900_Spatial")
```

```
nCISI300Post <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(nCISI300Post$Prop_Error)

#cor isi600 post
values_to_delete_all <- c("C_0_Control", "C_0_Posture","C_0_Spatial",
                          "C_300_Control", "C_300_Posture", "C_300_Spatial",
                          "C_600_Control", "C_600_Spatial",
                          "C_900_Control", "C_900_Posture", "C_900_Spatial",
                          "NC_0_Control", "NC_0_Posture", "NC_0_Spatial",
                          "NC_300_Control", "NC_300_Posture",
"NC_300_Spatial",
                          "NC_600_Control", "NC_600_Posture",
"NC_600_Spatial",
                          "NC_900_Control", "NC_900_Posture",
"NC_900_Spatial")
CISI600Post <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(CISI600Post$Prop_Error)

#ncor isi600 post
values_to_delete_all <- c("C_0_Control", "C_0_Posture","C_0_Spatial",
                          "C_300_Control", "C_300_Posture", "C_300_Spatial",
                          "C_600_Control", "C_600_Posture", "C_600_Spatial",
                          "C_900_Control", "C_900_Posture", "C_900_Spatial",
                          "NC_0_Control", "NC_0_Posture", "NC_0_Spatial",
                          "NC_300_Control", "NC_300_Posture",
"NC_300_Spatial",
                          "NC_600_Control", "NC_600_Spatial",
                          "NC_900_Control", "NC_900_Posture",
"NC_900_Spatial")
nCISI600Post <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(nCISI600Post$Prop_Error)

#cor isi900 post
values_to_delete_all <- c("C_0_Control", "C_0_Posture","C_0_Spatial",
                          "C_300_Control", "C_300_Posture", "C_300_Spatial",
                          "C_600_Control", "C_600_Posture", "C_600_Spatial",
                          "C_900_Control", "C_900_Spatial",
                          "NC_0_Control", "NC_0_Posture", "NC_0_Spatial",
                          "NC_300_Control", "NC_300_Posture",
"NC_300_Spatial",
                          "NC_600_Control", "NC_600_Posture",
"NC_600_Spatial",
                          "NC_900_Control", "NC_900_Posture",
"NC_900_Spatial")
CISI900Post <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(CISI900Post$Prop_Error)

#ncor isi900 post
values_to_delete_all <- c("C_0_Control", "C_0_Posture","C_0_Spatial",
                          "C_300_Control", "C_300_Posture", "C_300_Spatial",
                          "C_600_Control", "C_600_Posture", "C_600_Spatial",
                          "C_900_Control", "C_900_Posture", "C_900_Spatial",
                          "NC_0_Control", "NC_0_Posture", "NC_0_Spatial",
                          "NC_300_Control", "NC_300_Posture",
"NC_300_Spatial",
                          "NC_600_Control", "NC_600_Posture",
"NC_600_Spatial",
                          "NC_900_Control", "NC_900_Spatial")
nCISI900Post <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(nCISI900Post$Prop_Error)

#cor isi0 spat
values_to_delete_all <- c("C_0_Control", "C_0_Posture",
                          "C_300_Control", "C_300_Posture", "C_300_Spatial",
                          "C_600_Control", "C_600_Posture", "C_600_Spatial",
                          "C_900_Control", "C_900_Posture", "C_900_Spatial",
                          "NC_0_Control", "NC_0_Posture", "NC_0_Spatial",
                          "NC_300_Control", "NC_300_Posture",
"NC_300_Spatial",
                          "NC_600_Control", "NC_600_Posture",
"NC_600_Spatial",
                          "NC_900_Control", "NC_900_Posture",
"NC_900_Spatial")
CISI0Spat <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(CISI0Spat$Prop_Error)

#ncor isi0 spat
values_to_delete_all <- c("C_0_Control", "C_0_Posture","C_0_Spatial",
                          "C_300_Control", "C_300_Posture", "C_300_Spatial",
                          "C_600_Control", "C_600_Posture", "C_600_Spatial",
                          "C_900_Control", "C_900_Posture", "C_900_Spatial",
                          "NC_0_Control", "NC_0_Posture",
                          "NC_300_Control", "NC_300_Posture",
"NC_300_Spatial",
                          "NC_600_Control", "NC_600_Posture",
"NC_600_Spatial",
                          "NC_900_Control", "NC_900_Posture",
"NC_900_Spatial")
NCISI0spat <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(NCISI0spat$Prop_Error)

#cor isi300 spat
values_to_delete_all <- c("C_0_Control", "C_0_Posture","C_0_Spatial",
                          "C_300_Control", "C_300_Posture",
                          "C_600_Control", "C_600_Posture", "C_600_Spatial",
                          "C_900_Control", "C_900_Posture", "C_900_Spatial",
                          "NC_0_Control", "NC_0_Posture", "NC_0_Spatial",
                          "NC_300_Control", "NC_300_Posture",
"NC_300_Spatial",
                          "NC_600_Control", "NC_600_Posture",
"NC_600_Spatial",
                          "NC_900_Control", "NC_900_Posture",
"NC_900_Spatial")
CISI300spat <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(CISI300spat$Prop_Error)
```

```
#ncor isi300 spat
values_to_delete_all <- c("C_0_Control", "C_0_Posture","C_0_Spatial",
                          "C_300_Control", "C_300_Posture", "C_300_Spatial",
                          "C_600_Control", "C_600_Posture", "C_600_Spatial",
                          "C_900_Control", "C_900_Posture", "C_900_Spatial",
                          "NC_0_Control", "NC_0_Posture", "NC_0_Spatial",
                          "NC_300_Control", "NC_300_Posture",
                          "NC_600_Control", "NC_600_Posture",
"NC_600_Spatial",
                          "NC_900_Control", "NC_900_Posture",
"NC_900_Spatial")
nCISI300spat <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(nCISI300spat$Prop_Error)

#cor isi600 spat
values_to_delete_all <- c("C_0_Control", "C_0_Posture","C_0_Spatial",
                          "C_300_Control", "C_300_Posture", "C_300_Spatial",
                          "C_600_Control", "C_600_Posture",
                          "C_900_Control", "C_900_Posture", "C_900_Spatial",
                          "NC_0_Control", "NC_0_Posture", "NC_0_Spatial",
                          "NC_300_Control", "NC_300_Posture",
"NC_300_Spatial",
                          "NC_600_Control", "NC_600_Posture",
"NC_600_Spatial",
                          "NC_900_Control", "NC_900_Posture",
"NC_900_Spatial")
CISI600spat <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(CISI600spat$Prop_Error)

#ncor isi600 spat
values_to_delete_all <- c("C_0_Control", "C_0_Posture","C_0_Spatial",
                          "C_300_Control", "C_300_Posture", "C_300_Spatial",
                          "C_600_Control", "C_600_Posture", "C_600_Spatial",
                          "C_900_Control", "C_900_Posture", "C_900_Spatial",
                          "NC_0_Control", "NC_0_Posture", "NC_0_Spatial",
                          "NC_300_Control", "NC_300_Posture",
"NC_300_Spatial",
                          "NC_600_Control", "NC_600_Posture",
                          "NC_900_Control", "NC_900_Posture",
"NC_900_Spatial")
nCISI600spat <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(nCISI600spat$Prop_Error)

#cor isi900 spat
values_to_delete_all <- c("C_0_Control", "C_0_Posture","C_0_Spatial",
                          "C_300_Control", "C_300_Posture", "C_300_Spatial",
                          "C_600_Control", "C_600_Posture", "C_600_Spatial",
                          "C_900_Control", "C_900_Posture",
                          "NC_0_Control", "NC_0_Posture", "NC_0_Spatial",
                          "NC_300_Control", "NC_300_Posture",
"NC_300_Spatial",
                          "NC_600_Control", "NC_600_Posture",
"NC_600_Spatial",
                          "NC_900_Control", "NC_900_Posture",
"NC_900_Spatial")
CISI900spat <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(CISI900spat$Prop_Error)

#ncor isi900 spat
values_to_delete_all <- c("C_0_Control", "C_0_Posture","C_0_Spatial",
                          "C_300_Control", "C_300_Posture", "C_300_Spatial",
                          "C_600_Control", "C_600_Posture", "C_600_Spatial",
                          "C_900_Control", "C_900_Posture", "C_900_Spatial",
                          "NC_0_Control", "NC_0_Posture", "NC_0_Spatial",
                          "NC_300_Control", "NC_300_Posture",
"NC_300_Spatial",
                          "NC_600_Control", "NC_600_Posture",
"NC_600_Spatial",
                          "NC_900_Control", "NC_900_Posture")
nCISI900spat <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(nCISI900spat$Prop_Error)

#cor isi0 con
values_to_delete_all <- c("C_0_Posture", "C_0_Spatial",
                          "C_300_Control", "C_300_Posture", "C_300_Spatial",
                          "C_600_Control", "C_600_Posture", "C_600_Spatial",
                          "C_900_Control", "C_900_Posture", "C_900_Spatial",
                          "NC_0_Control", "NC_0_Posture", "NC_0_Spatial",
                          "NC_300_Control", "NC_300_Posture",
"NC_300_Spatial",
                          "NC_600_Control", "NC_600_Posture",
"NC_600_Spatial",
                          "NC_900_Control", "NC_900_Posture",
"NC_900_Spatial")
CISI0c <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(CISI0c$Prop_Error)

#ncor isi0 con
values_to_delete_all <- c("C_0_Control", "C_0_Posture","C_0_Spatial",
                          "C_300_Control", "C_300_Posture", "C_300_Spatial",
                          "C_600_Control", "C_600_Posture", "C_600_Spatial",
                          "C_900_Control", "C_900_Posture", "C_900_Spatial",
                          "NC_0_Posture", "NC_0_Spatial",
                          "NC_300_Control", "NC_300_Posture",
"NC_300_Spatial",
                          "NC_600_Control", "NC_600_Posture",
"NC_600_Spatial",
                          "NC_900_Control", "NC_900_Posture",
"NC_900_Spatial")
NCISI0c <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(NCISI0c$Prop_Error)

#cor isi300 con
values_to_delete_all <- c("C_0_Control", "C_0_Posture","C_0_Spatial",
                          "C_300_Posture", "C_300_Spatial",
                          "C_600_Control", "C_600_Posture", "C_600_Spatial",
                          "C_900_Control", "C_900_Posture", "C_900_Spatial",
                          "NC_0_Control", "NC_0_Posture", "NC_0_Spatial",
                          "NC_300_Control", "NC_300_Posture",
"NC_300_Spatial",
                          "NC_600_Control", "NC_600_Posture",
"NC_600_Spatial",
                          "NC_900_Control", "NC_900_Posture",
"NC_900_Spatial")
CISI300c <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(CISI300c$Prop_Error)

#ncor isi300 con
values_to_delete_all <- c("C_0_Control", "C_0_Posture","C_0_Spatial",
                          "C_300_Control", "C_300_Posture", "C_300_Spatial",
                          "C_600_Control", "C_600_Posture", "C_600_Spatial",
                          "C_900_Control", "C_900_Posture", "C_900_Spatial",
                          "NC_0_Control", "NC_0_Posture", "NC_0_Spatial",
                          "NC_300_Posture", "NC_300_Spatial",
                          "NC_600_Control", "NC_600_Posture",
"NC_600_Spatial",
                          "NC_900_Control", "NC_900_Posture",
"NC_900_Spatial")
nCISI300c <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(nCISI300c$Prop_Error)

#cor isi600 con
values_to_delete_all <- c("C_0_Control", "C_0_Posture","C_0_Spatial",
                          "C_300_Control", "C_300_Posture", "C_300_Spatial",
                          "C_600_Posture", "C_600_Spatial",
                          "C_900_Control", "C_900_Posture", "C_900_Spatial",
                          "NC_0_Control", "NC_0_Posture", "NC_0_Spatial",
                          "NC_300_Control", "NC_300_Posture",
"NC_300_Spatial",
                          "NC_600_Control", "NC_600_Posture",
"NC_600_Spatial",
                          "NC_900_Control", "NC_900_Posture",
"NC_900_Spatial")
CISI600c <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(CISI600c$Prop_Error)

#ncor isi600 con
values_to_delete_all <- c("C_0_Control", "C_0_Posture","C_0_Spatial",
                          "C_300_Control", "C_300_Posture", "C_300_Spatial",
                          "C_600_Control", "C_600_Posture", "C_600_Spatial",
                          "C_900_Control", "C_900_Posture", "C_900_Spatial",
                          "NC_0_Control", "NC_0_Posture", "NC_0_Spatial",
                          "NC_300_Control", "NC_300_Posture",
"NC_300_Spatial",
                          "NC_600_Posture", "NC_600_Spatial",
                          "NC_900_Control", "NC_900_Posture",
"NC_900_Spatial")
nCISI600c <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(nCISI600c$Prop_Error)
```

```
#cor isi900 con
values_to_delete_all <- c("C_0_Control", "C_0_Posture","C_0_Spatial",
                          "C_300_Control", "C_300_Posture", "C_300_Spatial",
                          "C_600_Control", "C_600_Posture", "C_600_Spatial",
                          "C_900_Posture", "C_900_Spatial",
                          "NC_0_Control", "NC_0_Posture", "NC_0_Spatial",
                          "NC_300_Control", "NC_300_Posture",
"NC_300_Spatial",
                          "NC_600_Control", "NC_600_Posture",
"NC_600_Spatial",
                          "NC_900_Control", "NC_900_Posture",
"NC_900_Spatial")
CISI900c <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(CISI900c$Prop_Error)

#ncor isi900 con
values_to_delete_all <- c("C_0_Control", "C_0_Posture","C_0_Spatial",
                          "C_300_Control", "C_300_Posture", "C_300_Spatial",
                          "C_600_Control", "C_600_Posture", "C_600_Spatial",
                          "C_900_Control", "C_900_Posture", "C_900_Spatial",
                          "NC_0_Control", "NC_0_Posture", "NC_0_Spatial",
                          "NC_300_Control", "NC_300_Posture",
"NC_300_Spatial",
                          "NC_600_Control", "NC_600_Posture",
"NC_600_Spatial",
                          "NC_900_Posture", "NC_900_Spatial")
nCISI900c <- Nick[!(Nick$Prime %in% values_to_delete_all), ]
mean(nCISI900c$Prop_Error)




#plot it
# Get the estimated marginal means (EMMs) for the interaction of interest
emmeans_results2 <- emmeans(Tick, ~ C_NC * Number * Category)

# Convert the emmeans results to a data frame
emmeans_df1 <- as.data.frame(emmeans_results2)

# Subset specific conditions if needed (example here filters for "Spatial"
Category)
# specific_conditions1 <- subset(emmeans_df, Category == "Spatial")

# Extract necessary columns for plotting
plot_data <- emmeans_df1[, c("C_NC", "Number", "Category", "emmean",
"lower.CL", "upper.CL")]
colnames(plot_data) <- c("C_NC", "Number", "Category", "Mean_Error",
"Lower_CI", "Upper_CI")

# Create the interaction plot with ggplot2
```

```
ggplot(data = plot_data, aes(x = factor(Number), y = Mean_Error, group =
interaction(Category, C_NC), color = Category)) +
  geom_line(aes(linetype = C_NC), size = 1.1) +
  geom_point(aes(shape = C_NC), size = 2) +
  geom_errorbar(aes(ymin = Lower_CI, ymax = Upper_CI), width = 0.2, size =
1.1) +
  labs(x = "Interstimulus Interval (ms)", y = "Error Proportions", color =
"Prime", linetype = "Correspondence", shape = "Correspondence") +
  theme_bw() +
  theme(legend.position = "right")


#w/o error plots
ggplot(data = plot_data, aes(x = factor(Number), y = Mean_Error, group =
interaction(Category, C_NC), color = Category)) +
  geom_line(aes(linetype = C_NC), size = 1.1) +
  geom_point(aes(shape = C_NC), size = 2) +
  labs(x = "Interstimulus Interval (ms)", y = "Error Proportions", color =
"Prime", linetype = "Correspondence", shape = "Correspondence") +
  theme_bw() +
  theme(legend.position = "right")
```