Dokumentation der Monster- und Morse-Aufgaben

27. November 2025

1 Abhängigkeiten

Python 3.11.3[1]**:** Die gesamten automatisierten Teile der Verarbeitung sind in Python geschrieben.

pygame[2] **2.6.1:** Da dies bei diesen Programmen auch graphischer Output und Audio gebraucht werden, wird pygame genutzt um all dies zu managen.

PyQt5[3] **5.15.11:** Dies ist nötig um Dialogefenster darzustellen.

Installierte Versionen:

- Python 3.11.3
- pygame 2.6.1
- PyQt5 5.15.11
- PyQt5-Qt5 5.15.2
- PyQt5_sip 12.16.0
- setuptools 65.5.0

2 Installation

Bevor mit der Installation des eigentlichen Programms begonnen werden kann, müssen zuerst die in Abschnitt 1 beschriebenen Abhängigkeiten installiert werden. Da es sich bei pygame um Paket für Python selbst handelt, muss Python auch schon vor der Installation von pygame installiert sein.

Sind die Abhängigkeiten installiert, kann der Quellcode auch schon von einem Git-Repository[4] heruntergeladen und an einem beliebigen Ort entpackt werden. Anschließend ist das Programm startbereit und sollte getestet werden.

Definition 1 (Top-Level)

[1] `https://python.org/downloads/release/python-3113/`
[2] `https://pygame.org/download.shtml`
[3] `https://www.riverbankcomputing.com/software/pyqt/download`
[4] `https://github.com/jasserdhaouadi/Monster_Morse`

Im weiteren Verlaufe wird der Ordner, in dem sich das eigentliche Programm, also die Pythondateien monster1.py, morsel.py usw., *befindet, als Top-Level bezeichnet.*

Später kann Top-Level nach belieben verschoben werden, da ja keine Installation im eigentlichen Sinne stattgefunden hat. Jedoch sollte darauf geachtete werden, dass die Ordnerstruktur in Top-Level nicht verändert wird.

3 Allgemeine Informationen

3.1 Start der Programme

Um eines der Programme zu starten, muss nur die entsprechende Datei auf dem Top-Level ausgeführt werden. Dies kann entweder von einer Kommandozeile oder durch ein graphisches Programm wie beispielsweise den Arbeitsplatz unter Windows geschehen. Am generellen Ablauf des Programms ändert sich hierbei nichts, allerdings haben graphische Programme gegen über einer Kommandozeile den klaren Nachteil, dass gegeben Falls auftretende Fehlermeldungen zwar „angezeigt"werden, aber nicht lesbar sind, da sie sofort nach anzeigen der Fehlermeldung verschwinden.

Anschließend erscheint ein Dialog, indem ein Probandencode abgefragt wird. Dieser muss nicht ausgefüllt werden, um mit dem Programm fortzufahren, da im Falle eines leeren Eingabefeldes der Probandencode „testängenommen wird. Anschließend wird überprüft, ob der genannte Probandencode schon in den Log-Files existiert und, sollte dies der Fall sein, gefragt, ob die Daten überschrieben werden sollen. Bei beiden Dialogen führt eine negative Antwort zum Beenden des Programms. Ansonsten startet der eigentliche Teil des Programms mit einem Sartscreen. Von diesem aus kann mit einem Druck auf <ENTER> mit den Aufgaben begonnen werden.

3.2 Hilfreiche Tastenbefehle

Es gibt ein paar Tastenbefehle in den Programmen, die dem Nutzer die Arbeit ein wenig erleichtern können. Die wichtigsten seien hier aufgezählt:

- **<F12>:** Einige der Instruktionen lassen sich durch einen Druck auf <F12> beenden. Das Programm läuft danach normal weiter als wäre die Instruktion beendet.
- **<ESC>:** Hiermit lässt sich das Programm zu jeder Zeit beenden. An manchen Stellen, besonders bei bestimmten Instruktionen, kann das Programm nicht sofort beendet werden, sollte jedoch während dessen <ESC> gedrückt werden, wird zum nächsten möglichen Zeitpunkt beendet.

4 Monster 1 & 2

Da sich die Aufgaben Monster 1 und 2 nur in der Form des visuellen Stimulus voneinander unterscheiden, der Rest aber identisch ist, werden sie hier gemeinsam behandelt.

4.1 Durchführung

Das Programm wird wie in Unterabschnitt 3.1 beschrieben gestartet. Monster 1 und 2 bestehen aus zwei Teilen: Einer Lernphase und dem anschließenden eigentlichen Test. Als

Eingabegerät ist eine Maus vorgesehen, wobei die linke Maustaste als Eingabe für Falsch und die rechte für Richtig genutzt wird. Dies sollte auf der Maus durch einen roten und einen grünen Aufkleber signalisiert werden.

Der primäre Unterschied zwischen der Lernphase und dem Test ist, dass während der Lernphase eine Rückmeldung zu den gegebenen Antworten gegeben wird und gegeben Falls der Trial wiederholt wird. Beim Test geschieht dies nicht. Abgesehen davon, sollten die Programme reibungslos und komplett automatisiert ablaufen.

4.2 Auswertung

Um einer automatisierte Auswertung zu ermöglichen, werden im Ordner `helpers/trial_log/<PROBCODE` jeweils ein Ordner für monster1 und monster2 angelegt, in welchen sich wiederum jeweils eine Datei learn und test befinden. Darin werden für jeden Trial die Trial-Nr., die gedrückte Taste und der entsprechende Response gespeichert. Zusätzlich wird auch die Reaktionszeit zwischen dem Präsentieren des Stimulus und dem Drücken der Taste gespeichert.

Die gedrückte Taste und der Response sind wie folgt kodiert:

	0	1
gedrückte Taste	linke Maustaste	rechte Maustaste
Response	falsch	richtig

Tabelle 1: Kodierung des Tastendrucks und des Response in Log-Dateien

5 Monster 3

Monster 3 ist nun eine Weiterführung der Aufgaben Monster 1 und 2.

5.1 Durchführung

Die Durchführung lehnt sich im Großen und Ganzen an die von Monster 1 und 2 wie in Unterabschnitt 4.1 beschrieben an.

5.2 Auswertung

Wie in für Monster 1 und 2 in Unterabschnitt 4.2 schon beschrieben, wird auch hier ein Ordner monster3 in `helpers/trial_log/<PROBCODE>` angelegt, in welchem die erstellten Log-Dateien gespeichert werden. Da diese Aufgabe aber dreigeteilt ist, werden auch drei Log-Dateien gespeichert: Zum einen die Dateien cookie und monster für die beiden Ubungsaufgaben und zum anderen test für den eigentlichen Test. Die Einträge entsprächen dabei den schon in Unterabschnitt 4.2 beschriebenen.

6 Morse 1

Morse 1 ist wiederum ein komplett anderes Szenario. Zur Durchführung ist neben einem separaten Programm zur Aufnahme von Audio (hier sei Audacity[5] empfohlen) auch ein

[5] `https://audacity.sourceforge.net/`

Mikrofon und ein Setup, welches sowohl Audio ausgeben als auch aufnehmen kann, von Noten.

6.1 Durchführung

Da es während der Durchführung es Programms nicht möglich ist, zu einem anderen Programm zu wechseln, muss die Audioaufnahme vor Beginn gestartet werden. Anschließend ist das Programm wie in Unterabschnitt 3.1 beschrieben zu starten.

Die Aufgabe besteht aus **zwei Hauptteilen**:

Teil 1:

- Es werden zwei Zeichen eingeführt
- Eine Lernphase mit Rückmeldung
- Ein Test ohne Rückmeldung

Teil 2:

- Nach Abschluss von Teil 1 werden zusätzliche neue Zeichen eingeführt (zusammen mit den ursprünglichen zwei Zeichen)
- Eine Lernphase mit Rückmeldung
- Ein abschließender Test ohne Rückmeldung

Während der Testphasen soll der Proband die auf dem Bildschirm präsentierten Zeichen vorlesen. Zwischen den einzelnen Trials wird mit <SPACE> und <ENTER> die Aussage des Kindes bestätigt, wobei ersteres eine falsche und letzteres eine richtige Aussage markieren.

6.2 Auswertung

Die Auswertung von Morse 1 birgt noch einige Probleme. Da es keine Möglichkeit gibt, Audio direkt im Programm Morse 1 aufzuzeichnen, sondern ein externes Programm genutzt werden muss, müssen die Zeiten der Audioaufnahme und die von Morse 1 ermittelten im Nachhinein synchronisiert werden. Dies geschieht über die Vorstellung des Zeichens *ta*. Aber die bisherigen Durchführungen haben gezeigt, dass dies problematisch ist, da die Zeit in unterschiedlichen Programmen unterschiedlich schnell gezählt werden.

Im eigentlich geplanten Verlauf der Auswertung müssen die Zeit des Punktes der Synchronisation und der Beginn der finalen Aussagen des Probanden ausgelesen werden und anschließend werden letztere in eine Datei in im Ordner `helpers/trial_log/<PROBANDENCODE>/morsel` abgespeichert. Anschließend wird das Programm `auswertung.py` ausgeführt, welches die Synchronisationszeit abfragt und anschließend eine Datei `test_resptime` in genanntem Ordner anlegt, in welcher dann folgende Daten abgespeichert werden:

- *trial*: die laufende Nummer des Trials
- *image*: das präsentierte Bild
- *correct*: der nach Tabelle 1 kodierte Response
- *time_m*: ???

- *time_r*: ???
- *time_response*: die Zeit zwischen Präsentation des Stimulus und der finalen Äußerung

Generell ist das Problem der unzuverlässigen Synchronisation aber ein Problem, welches vor einer erneuten Verwendung nochmals überdacht und nach Möglichkeit gelöst werden sollte.

7 Morse 2

Morse 2 ist nun das genaue Gegenteil von Morse 1: hier soll der Proband die vorgelesenen Zeichen aufschreiben.

7.1 Durchführung

Das Programm muss wie in Unterabschnitt 3.1 beschrieben gestartet werden. Nach einer kleinen Einführung soll der Proband dann die präsentierten Zeichen wie in der Einführung beschrieben aufschreiben. Jeder Trail wird mit <SPACE> und <ENTER> bestätigt, wobei, wie auch schon bei Morse 1, ersteres für einen falschen und letzteres für einen richtigen Response steht.

7.2 Auswertung

Die Auswertung von Morse 2 findet zum größten Teil außerhalb des Programms statt. So wird in Morse 2 nur die Trial-Nummer und der gegebene Response geloggt. Die entsprechende Log-Datei ist in `helpers/trial_log/<PROBANDENCODE>/morse2` zu finden.