## Bedienungsanleitung

Da es sich bei dem entwickelten System um einen Prototyp handelt, ist die Bedienung noch nicht benutzerfreundlich gestaltet und bedarf hier einer Einführung.

Das System könnte zwar autonom funktionieren, da aber auf die Verwendung eines Displays verzichtet wurde ist es notwendig ein zweites Endgerät zu verwenden.

Zunächst müssen sich beide Geräte in einem gemeinsamen Netzwerk befinden. Dies kann über den Access-Point, den der Raspberry Pi hostet stattfinden oder über beliebige umgebende Netzwerke auf das beide Geräte Zugriff haben. Der Name des AP lautet „wifipi“ und es wird das Passwort „RaspberryWiFi“ verwendet.

Befinden sich beide Geräte im Netzwerk, so kann eine SSH-Verbindung hergestellt werden. Besonders geeignet hierfür ist das Tool „PuTTY“, da es für eine Vielzahl von Betriebssystemen verfügbar ist.

Alternativ dazu kann auch ein Remote Zugriff mit grafischem Desktop wie „VNC Viewer“ verwendet werden. Dieser belastet allerding den Arbeitsspeicher des Raspberry Pi stark, sodass dieser nicht für den laufenden Betrieb des Systems empfohlen wird.

Besteht eine gültige SSH-Verbindung so gibt es nun zwei Betriebsvarianten.

Die Hauptfunktion unseres Systems wird mithilfe von 2 Programmen realisiert. Das eine Programm definiert die SPI-Schnittstelle und liest die Mikrofondaten ein. Diese werden dann über eine ~~Pipe~~ (Pipeline) an das Hauptprogramm gesendet. Für den Entwicklungsprozess wurde das Hauptprogramm über einen Remote Zugriff in der „Netbeans IDE“ entwickelt. So können sämtliche Änderungen am Programm wie etwa die Kalibrierung der Geräuschempfindlichkeit direkt über das zusätzliche Endgerät stattfinden. Diese Empfindlichkeit muss je nach Umgebungslautstärke oder des zu messenden Geräusches eingestellt werden. Zu Testzwecken wurde das System z.B. so eingestellt, dass es in einem ruhigen Umfeld ein Klatschen erkennt. Für die Inbetriebnahme ist es zwingend notwendig, dass zuerst das Hauptprogramm in der IDE ausgeführt wird. Dieses benötigt dann mindestens vier Sekunden, um die Kamera und die (Pipesiehe oben) zu stabilisieren. Im Anschluss daran kann das Skript „mic\_handler\_pipe64“ über die Kommandozeile gestartet werden. Die Auswertung der Mikrofondaten beginnt und eventuell auftretende Events werden erkannt und berechnet. Befindet sich das Hauptprogramm nicht im Modus „Debug“ läuft diese Auswertung endlos, bis das Programm händisch terminiert wird. Der Prozess wird korrekt beendet, wenn zuerst das Hauptprogramm in der IDE und anschließend das Skript beendet werden.

In der anderen Variante können alle für die Funktion notwendigen Programme mit dem Skript „script.sh“ gleichzeitig gestartet werden. Das Skript muss sich jedoch im gleichen Verzeichnis wie die von der IDE ausführbar kompilierte Version des Hauptprogrammes und das Skript zum starten der (Pipesiehe oben). Alle drei Skripte befinden sich im /home- Verzeichnis des Raspberry Pi und könnten z.B. mit dem Befehl „bash script.sh“ ausgeführt werden. Je nach Konfiguration des Hauptprogramms läuft der Prozess nun wieder unendlich, in diesem Fall können aber alle Unterprogramme mit einem „stg+C“ Kommando gleichzeitig beendet werden. Wurden in der IDE Änderungen am Hauptprogramm durchgeführt, so muss die aktualisierte Version des kompilierten Hauptprogramms händisch in das korrekte Verzeichnis kopiert werden.

Für einen einfacheren Zugriff auf den Dateiexplorer um z.B. die kompilierten Programme verschieben zu können, empfiehlt sich die Verwendung von „WinSCP“. Hier kann direkt über SSH auf die Ordnerstruktur des Raspberry Pi zugegriffen werden. So lassen sich Kopier- und Bearbeitungsvorgänge sowie die Sicherung und Sichtung der Bearbeiteten Beweisbilder direkt über das verwendete, zweite Endgerät durchgeführt werden.