|  |  |
| --- | --- |
| Abschnitt | Inhalt |
| Name | Mikrofonfunktionstest |
| Autor(en) | Philipp Otto |
| Priorität | hoch |
| Kritikalität | hoch |
| Quelle |  |
| Verantwortlicher | Philipp Otto |
| Kurzbeschreibung | Mit diesem Test soll die fehlerfreie Funktion der Mikrofone getestet werden |
| Auslösendes Ereignis | Eingabe Konsole:  „sudo ./RecordMicToRAM“ |
| Akteure | Mirkofonpaar, Rasberry Pi, Jumperkabel |
| Vorbedingung | Raspberry eingerichtet, Mikrofone und Raspberry mit Kabel verbunden, Raspberry eingeschaltet |
| Nachbedingung | „RecordMicToRAM“ beendet,  Prüfung der .txt-Datei ist erfolgt |
| Ergebnis | .txt- Datei erstellt |
| Hauptszenario | 1. Über die Konsoleneingabe wird das Programm gestartet 2. Das Programm zeichnet ca. 8 Sekunden lang den Mikrofonausgang auf 3. Das Programm erstellt eine .txt-Datei mit allen Messwerten 4. Die .txt-Datei wird auf Vollständigkeit überprüft |
| Alternativszenario | 4.b Die .txt-Datei enthält keine Messwerte  4.c Kabelverbindung wird überprüft |
| Ausnahmeszenario |  |
| Qualitäten |  |

|  |  |
| --- | --- |
| Abschnitt | Inhalt |
| Name | Kabelfunktionstest Nr.1 |
| Autor(en) | Philipp Otto |
| Priorität | hoch |
| Kritikalität | hoch |
| Quelle |  |
| Verantwortlicher | Philipp Otto |
| Kurzbeschreibung | Mit diesem Test soll die fehlerfreie Funktion der Kabel getestet werden |
| Auslösendes Ereignis | Eingabe Konsole:  „sudo ./RecordMicToRAM“ |
| Akteure | gesamtes Mirkofonarray, Rasberry Pi, Kabel(lang) |
| Vorbedingung | Das Array ist vollständig mit Kabeln und Raspberry verbunden, Raspberry eingeschaltet |
| Nachbedingung | Messwerte grafisch auf Plausibilität überprüft |
| Ergebnis | Plot der Messwerte |
| Hauptszenario | 1. Über die Konsoleneingabe wird das Programm gestartet 2. Das Programm zeichnet ca. 8 Sekunden lang den Mikrofonausgang auf 3. Das Programm erstellt eine .txt-Datei mit allen Messwerten 4. Die .txt-Datei wird mit python-script in korrektes Datenformat gebracht und in .csv konvertiert 5. .csv wird mit Matlab eingelesen und geplottet 6. Plot wird auf Plausibilität überprüft |
| Alternativszenario | 4.b Die .txt-Datei enthält keine Messwerte  4.c Kabelverbindung wird überprüft  6.b Plot weist fehlerhafte Messwerte auf |
| Ausnahmeszenario |  |
| Qualitäten |  |

|  |  |
| --- | --- |
| Abschnitt | Inhalt |
| Name | Kabelfunktionstest Nr.2 |
| Autor(en) | Philipp Otto |
| Priorität | hoch |
| Kritikalität | hoch |
| Quelle |  |
| Verantwortlicher | Philipp Otto |
| Kurzbeschreibung | Mit diesem Test soll die fehlerfreie Funktion der Kabel bei halbiertem SPI-Takt getestet werden |
| Auslösendes Ereignis | Eingabe Konsole:  „sudo ./RecordMicToRAM64“ |
| Akteure | gesamtes Mirkofonarray, Rasberry Pi, Kabel(lang) |
| Vorbedingung | Das Array ist vollständig mit Kabeln und Raspberry verbunden, Raspberry eingeschaltet |
| Nachbedingung | Messwerte grafisch auf Plausibilität überprüft |
| Ergebnis | Plot der Messwerte |
| Hauptszenario | 1. Über die Konsoleneingabe wird das Programm gestartet 2. Das Programm zeichnet ca. 8 Sekunden lang den Mikrofonausgang auf 3. Das Programm erstellt eine .txt-Datei mit allen Messwerten 4. Die .txt-Datei wird mit python-script in korrektes Datenformat gebracht und in .csv konvertiert 5. .csv wird mit Matlab eingelesen und geplottet 6. Plot wird auf Plausibilität überprüft |
| Alternativszenario | 4.b Die .txt-Datei enthält keine Messwerte  4.c Kabelverbindung wird überprüft  6.b Plot weist fehlerhafte Messwerte auf |
| Ausnahmeszenario |  |
| Qualitäten |  |

|  |  |
| --- | --- |
| Abschnitt | Inhalt |
| Name | Kabelfunktionstest Nr.3 |
| Autor(en) | Philipp Otto |
| Priorität | hoch |
| Kritikalität | hoch |
| Quelle |  |
| Verantwortlicher | Philipp Otto |
| Kurzbeschreibung | Mit diesem Test soll die fehlerfreie Funktion der Kabel bei halbiertem SPI-Takt und kürzeren Kabeln getestet werden |
| Auslösendes Ereignis | Eingabe Konsole:  „sudo ./RecordMicToRAM64“ |
| Akteure | gesamtes Mirkofonarray, Rasberry Pi, Kabel(kurz) |
| Vorbedingung | Das Array ist vollständig mit Kabeln und Raspberry verbunden, Raspberry eingeschaltet |
| Nachbedingung | Messwerte grafisch auf Plausibilität überprüft |
| Ergebnis | Plot der Messwerte |
| Hauptszenario | 1. Über die Konsoleneingabe wird das Programm gestartet 2. Das Programm zeichnet ca. 8 Sekunden lang den Mikrofonausgang auf 3. Das Programm erstellt eine .txt-Datei mit allen Messwerten 4. Die .txt-Datei wird mit python-script in korrektes Datenformat gebracht und in .csv konvertiert 5. .csv wird mit Matlab eingelesen und geplottet 6. Plot wird auf Plausibilität überprüft |
| Alternativszenario | 4.b Die .txt-Datei enthält keine Messwerte  4.c Kabelverbindung wird überprüft  6.b Plot weist fehlerhafte Messwerte auf |
| Ausnahmeszenario |  |
| Qualitäten |  |

|  |  |
| --- | --- |
| Abschnitt | Inhalt |
| Name | Algorithmus-Test (Matlab) |
| Autor(en) | Philipp Otto |
| Priorität | hoch |
| Kritikalität | hoch |
| Quelle |  |
| Verantwortlicher | Philipp Otto |
| Kurzbeschreibung | Mit diesem Test soll die fehlerfreie Funktion des Lokalisierungs-algorithmus getestet werden |
| Auslösendes Ereignis | Eingabe Konsole:  „sudo ./RecordMicToRAM64“ |
| Akteure | gesamtes Mirkofonarray, Rasberry Pi, Kabel(kurz) |
| Vorbedingung | Das System ist vollständig montiert,  Raspberry eingeschaltet |
| Nachbedingung | Lokalisierungsergebnisse grafisch auf Plausibilität überprüft. |
| Ergebnis | Plot der Lokalisierungsergebnisse |
| Hauptszenario | 1. Über die Konsoleneingabe wird das Programm gestartet 2. Das Programm zeichnet ca. 8 Sekunden lang den Mikrofonausgang auf 3. Das Programm erstellt eine .txt-Datei mit allen Messwerten 4. Die .txt-Datei wird mit python Skript in korrektes Datenformat gebracht und in .csv konvertiert 5. .csv wird in Matlab eingelesen 6. Lokalisierung wird mithilfe des Skripts „testDelayAndSum.m“ berechnet und geplottet 7. Plot wird auf Plausibilität überprüft |
| Alternativszenario | 7.b Lokalisierung entspricht nicht dem erwarteten Winkelbereich  7.c Lokalisierung ist uneindeutig  8. Messung wird in anderem Umfeld wiederholt |
| Ausnahmeszenario |  |
| Qualitäten |  |

|  |  |
| --- | --- |
| Abschnitt | Inhalt |
| Name | Gesamt-Funktionstest |
| Autor(en) | Philipp Otto |
| Priorität | hoch |
| Kritikalität | hoch |
| Quelle |  |
| Verantwortlicher | Philipp Otto |
| Kurzbeschreibung | Mit diesem Test soll die fehlerfreie Funktion des gesamten Programms getestet werden |
| Auslösendes Ereignis | Eingabe Konsole:  „sudo ./mic\_handler\_pipe64“,  Start des Programms über Netbeans |
| Akteure | gesamtes Mirkofonarray, Rasberry Pi, Kabel(kurz) |
| Vorbedingung | Das System ist vollständig montiert,  Raspberry eingeschaltet |
| Nachbedingung | Lokalisierungsergebnisse grafisch auf Plausibilität überprüft. |
| Ergebnis | Bearbeitetes .png-Datei |
| Hauptszenario | 1. Über „run“ wird das Hauptprogramm in der IDE gestartet 2. 4 Sekunden warten 3. Mit „sudo ./mic\_handler\_pipe64“ wird die pipe geöffnet 4. Geräuschevent wird erzeugt 5. Event wird erkannt und auf Konsole angezeigt 6. Wenn das Event im Bildbereich erzeugt wurde, wird ein Bild aufgezeichnet 7. Bild wird bearbeitet und abgespeichert 8. Bild wird auf Zweitgerät geladen 9. Bild wird auf Plausibilität überprüft |
| Alternativszenario | 5.b Event wird nicht korrekt erkannt  5.c Pegelerkennung wird angepasst  5.d Event wird wiederholt  6.b Event wurde im Bildbereich erzeugt aber nicht dort lokalisiert  6.c Event wird wiederholt  9.b Bearbeitetes Bild entspricht nicht den Erwartungen  9.c Lokalisierungsparameter (Buffer wird angepasst)  9.d Event wird wiederholt |
| Ausnahmeszenario |  |
| Qualitäten |  |