HAND-AUGE-KALIBRIERUNG

Dokumentation

Inhalt

Hauptordner der App 2
Anbindung eines Roboters und einer Simulation 2
Anbindung Roboter 2
getRobotPose → Schritt 3 3
moveRobotToJointAngles → Schritt 3 3
takePictureRobot → Schritt 3 3
rosConnection → Schritt 2 3
Anbindung Simulation 3
connectToSimulation → Schritt 2 3
getSimTcpPose → Schritt 3 3
setTcpPose → Schritt 3 3
startSimulation → Schritt 1 3
stopSimulation → Schritt 4 3

Hauptordner der App

Im Hauptordner „app“ sind alle benötigten Funktionen definiert und bilden die Schnittstelle zur der App. Folgende Unterordner sind verfügbar:

- **Calibration**: Alle Funktionen für die Berechnung der Hand Auge Kalibrierung
- **Configuration**: Alle Funktionen um *.ini Dateien zu lesen und schreiben
- **Icons**: Icons und Bilder
- **Robot**: Alle Funktionen für Anbindung des Roboters
- **Scenes**: Szenen der Simulation
- **Simulation**: Alle Funktionen für die Anbindung der Simulation
- **Utils**: Nützliche Funktionen

Siehe Kommentare für weitere Informationen der einzelnen Funktionen.

Anbindung eines Roboters und einer Simulation

Die Anbindung sowohl des Roboters als auch der Simulation benötigt einige Schritte. Es existieren 4 gemeinsame Schritte:

1. Start der Simulation/Anwendung für den Roboter
2. Verbindung mit Simulation/Roboter
3. Ausführung der Simulation/Roboterbewegung
4. Trennen von der Simulation/Roboter

Schritt 1 wird von der APP behandelt und muss nicht neu implementiert werden, solange ein Programm mit Argumenten gestartet werden soll. Anderenfalls müssen die Funktionen startSimExecutbale und startCaptureServer in der APP neu implementiert werden. Schritt 4 wird aktuell für den Roboter ignoriert.

Die APP speichert alle Pose (Kartesisch und Gelenkwinkel) als Vektoren ab, mit (X, Y, Z, Roll, Pitch, Yaw) bzw. (J1,..Jn). Für die Simulation und realen Roboter kann eine andere Darstellung als RPY gewählt werden. Sie muss bei der Anbindung berücksichtig werden. Ohne zusätzliches Eingreifen in die APP ist es nicht möglich die Visualisierung von RPY in Grad in eine andere Visualisierung zu ändern. Diese Änderung ist mit dem App Designer leicht zu machen.

Das vorgegebene Interface muss genutzt werden, fall keine Änderungen an der APP gemacht werden sollen.

Alle Anmerkungen sind im Folgenden für die aktuelle Implementierung gedacht und können ggf. vom aktuellen Simulations-/Robotersystem abweichen!

Anbindung Roboter

Im Verzeichnis app\robot müssen die Hauptfunktionen:

- getRobotPose → Schritt 3
- moveRobotToJointAngles → Schritt 3
- rosConnection → Schritt 2
- takePictureRobot → Schritt 3

an einen neuen Roboter angepasst werden. Die restlichen Funktionen werden von den Hauptfunktionen verwendet, Ausnahme moveRobotToCartPose. Für Implementierungsdetails sei an dieser Stelle auf die Funktionen selbst verwiesen.

getRobotPose → Schritt 3

Hole die aktuelle Roboter Pose und speichere sie. Es wird eine kartesische und GelenkWinkel Pose gespeichert

moveRobotToJointAngles → Schritt 3

Roboter mittels Gelenkwinkel bewegen. Diese Funktion macht busy waiting bis Roboter die Gelenkwinkel erreicht. Siehe rechedRobotJointAngles

takePictureRobot → Schritt 3

Mache ein Bild mit der am Roboter angebrachten Kamera. Dafür wird ein Capture Server von der APP gestartet → Schritt 1.

rosConnection → Schritt 2

Verbindung mit ROS

Anbindung Simulation

Im Verzeichnis app\robot müssen die Hauptfunktionen:

- connectToSimulation → Schritt 2
- getSimTcpPose → Schritt 3
- setTcpPose → Schritt 3
- startSimulation → Schritt 1
- stopSimulation → Schritt 4

an eine neue Simulation angepasst werden. Die restlichen Funktionen werden von den Hauptfunktionen verwendet. Für Implementierungsdetails sei an dieser Stelle auf die Funktionen selbst verwiesen.

connectToSimulation → Schritt 2

Verbindung mit Simulation basierend auf einer API zu Matlab

getSimTcpPose → Schritt 3

Hole aktuell simulierte TCP Pose aus der Simulation

setTcpPose → Schritt 3

Lege eine zu simulierende TCP Pose fest

startSimulation → Schritt 1

Starte Simulation, nicht die Anwendung!!!

stopSimulation → Schritt 4

Stoppe Simulation und trenne