Projektbeschreibung Armkontrolle Pointing & Gestures 03.06.24

Jobr1005@stud.hs-kl.de

Version V7

Ziel meines gewählten Projektes ist es,
die Roboter, die Nao´s mit Ihren Armen in bestimmte Richtungen oder auf Objekte deuten zu lassen.
Der Grund für die Implementierung dieser Funktion ist, dass das deuten auf bestimmte Objekte Später zum Testen genutzt werden kann z.B.

Ready Position Korrekt
Default Position Korrekt
Ball richtig erkannt

Im ersten Schritt meines Projektes soll umgesetzt werden:

Ein Roboter soll auf das eigene Tor zeigen können.
Erstellen einer scene mit einem Spieler → config/Scenes /Includes/1vsDummies.rsi2 1_Robot.ros2 erstellt. Ein Roboter auf dem Spielfeld.

Stand der Dinge:

Entsprechende Ansätze sind vorhanden (PointAt) → PointAtWithArm bevorzugen,
da der rechte Arm schlechter nach links zeigen kann.
Nao's wählen mit dem PointAt Skill den entsprechenden Arm aus mit dem besser gezeigt werden kann.

Dieser soll es ermöglichen die Nao´s auf bestimmte relative Koordinaten zeigen zu lassen.
Dies ist möglich, allerdings muss darauf geachtet werden, dass der Ball auch in einer deutbaren Richtung liegt (Anschläge der Gelenke).
Daher ist vorher der TurnToPoint Skill zu verwenden, dann sind alle denkbaren Richtungen möglich.

Nao's können auf die aktuelle Ballposition zeigen.
Die FieldBall.h enthält einige interessante Ballkoordinaten.
Es wird recentBallPositionRelative genutzt, um die Roboter spezifischen Ballkoordinaten zu erhalten.

recentBallPositionRelative zu `recentBallPositionOnField geändert`
Koordinaten Umrechnung in relative Koordinaten jetzt in PointingCard realisiert.
Können so einfach auf beliebige Koordinaten zeigen

Nao's drehen sich immer zum Ball. Das braucht viel Zeit.
Daher die Idee, die Nao's drehen sich nur zum Ball, wenn Sie nicht darauf zeigen können.
Wenn der Ball im Zeigebereich liegt, drehen sich die Roboter nicht.
Konzept Datei erstellt siehe Konzept Sichtfeld.cpp

Sichtfeld.cpp wurde implementiert. In der PointingCard kann jetzt ein Sichtfeld/ Zeigebereich eingestellt werden. Die Naos drehen dich nur in die Richtung des Balls, wenn der Ball nicht im zeigbaren Bereich liegt. PointingCard.cpp →V3

Vorhandene Ansätze:

- //The Robot Pose RobotPose.h SimulatedRobot.h Pose2f.h
- //GoalieDiveCard.cpp -> FieldBall.h

- Interessante Skills aus Skills.h:

```
/**
 * ACTION SKILL
 * This skill lets one arm point at some point.
 * @param localPoint The point in robot-relative coordinates
 */
SKILL_INTERFACE(PointAt, (const Vector3f&) localPoint);

/**
 * ACTION SKILL
 * This skill lets a specific arm point at some point.
 * @param localPoint The point in robot-relative coordinates
 * @param arm The arm that shall be used for pointing
 */
SKILL_INTERFACE(PointAtWithArm, (const Vector3f&) localPoint, (Arms::Arm) arm);
```

- Vielleicht weitere nützliche Skills:

```
/**
 * ACTION SKILL
 * This skill turns the robot to look forward at a target.
 * Does not move head
 * @param target The position in robot relative coordinates to turn to
 * @param margin The tolerance for the skill to be done
 */
SKILL_INTERFACE(TurnToPoint, (const Vector2f&) target, (Angle)(5_deg) margin);
```

Technische Umsetzung:

- Verwenden der PointAt Skills
- Zeit auslesen für Bestimmung der Dauer des Zeigens
  - game time
  - Unix System Zeit
  - Karten haben Zähler, Timer in Zustandsautomat z.B anstoß, theFrameInfo.getTimeSince()

theFrameInfo wurde genutzt. Die Karte hat jetzt eine Laufzeitbegrenzung und eine gewisse Sperrzeit, bis sie wieder aufgerufen werden kann.

Ablaufplan:

- relativen Koordinaten der Roboter verstehen, z.B. Wo ist Nullpunkt → Anstoßpunkt
  Globale Variablen, Consolenbefehl move mv zum verstehen
- Testen der vorhandenen Skills.
- Vergangene Zeit Messen (Zeit Auslesen)
- Testen im Simulator Develop, Hier können Infos einfach im Scene Graph angesehen werden
- Live-Tests

Konsolenbefehle: