Ontwerpdocumentatie simulatie robotarm

Max Janssen

Versie 1

Studentnummer: 1658262

Docent: Bram Knippenberg

Datum: 3 april 2024

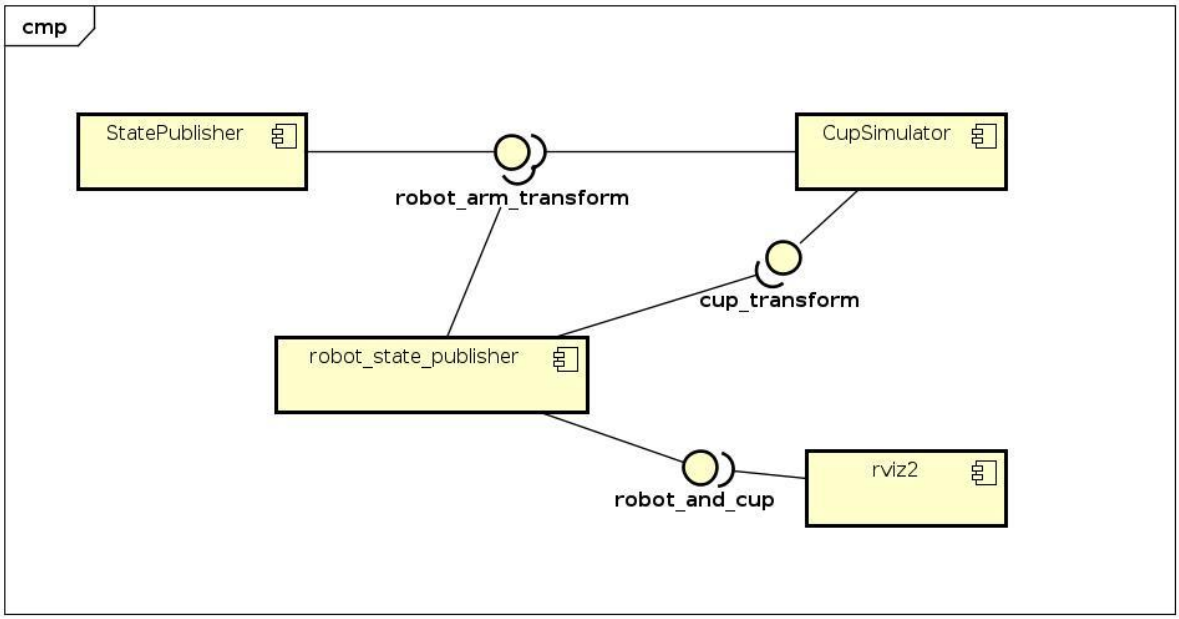
Inleiding

In dit document is het ontwerp van de gesimuleerde robotarm te lezen. Dit wordt gedaan met een component diagram en met klassendiagrammen. Ook wordt de communicaties tussen de ROS2-nodes van het programma weergegeven.

1. Ontwerp van het systeem

1.1 Component diagram

In de afbeelding hieronder is het component diagram van het systeem te zien:

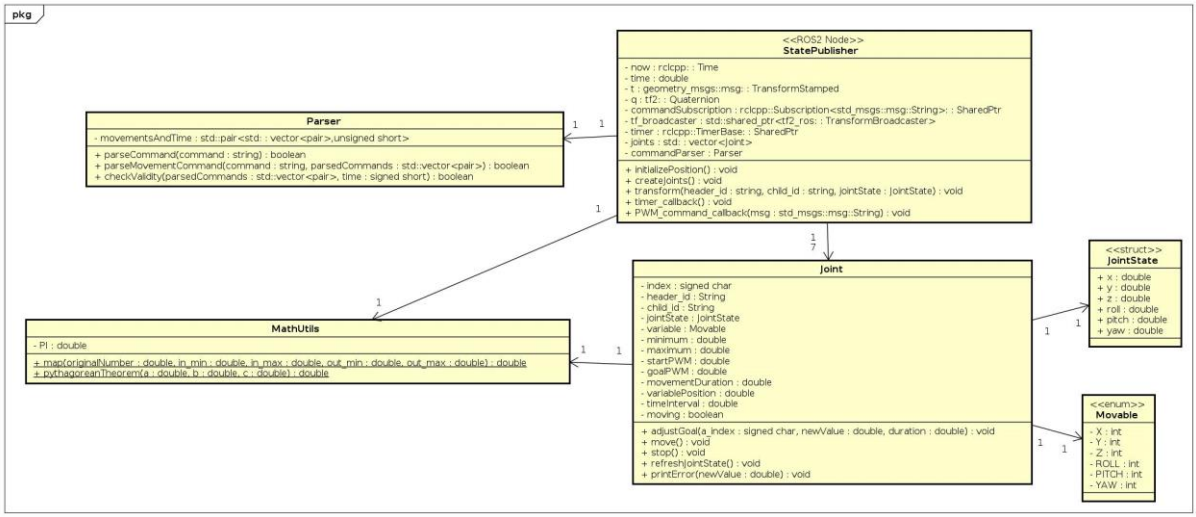


Hierin is te zien dat de StatePublisher de transforms (oftewel de posities en rotaties) van de servo's meegeeft. Deze wordt gebruikt door de CupSimulator om te kijken of de robotarm dicht genoeg bij het bekertje is om opgepakt te kunnen worden door de robotarm.

De StatePublisher en de CupSimulator geven respectievelijk hun posities mee aan de robot_state_publisher, om deze vervolgens mee te geven aan Rviz2, die de bewegingen van de robotarm en het bekertje toont.

1.2 Klassendiagram state publisher

Hieronder is het klassendiagram van de state publisher te zien (die de robotarm simuleert).



De functies van dit deel van het programma zijn het accepteren van commando's voor de robotarm en het beheren van de posities en rotaties van de servo's van de robotarm.

1.2.1 StatePublisher

Tijdens het draaien van het programma blijft deze node de posities en rotaties van al zijn joints broadcasten naar de robot_state_publisher. Daarnaast luistert deze naar het topic genaamd /arm_command om commando's voor de robotarm te registreren. Deze roept dan de functie adjustGoal in de joints aan om ervoor te zorgen dat de joints een nieuwe doelpositie krijgen. In een loop (timer_callback) wordt dan constant de move-functie van de joints aangeroepen om de servo's van de robotarm te bewegen.

1.2.2 Parser

Als de StatePublisher een commando ontvangt, controleer de Parser of deze daadwerkelijk in het SSC-32U-formaat zijn geschreven. Als dit niet het geval is, wordt deze niet geaccepteerd en wordt er een error geprint.

1.2.3 Joint

De StatePublisher bevat een aantal joints die bijhouden wat hun positie en rotatie is in een struct genaamd JointState. Ook staat hierin geregisreerd in welke richting de joint kan bewegen. Dit wordt bepaald door een variabele met het type Movable. Als dan de functie 'move' wordt aangeroepen, kan alleen de variabele in Jointstate die 'movable' is veranderen, zodat de servo's niet alle kanten op kunnen bewegen.

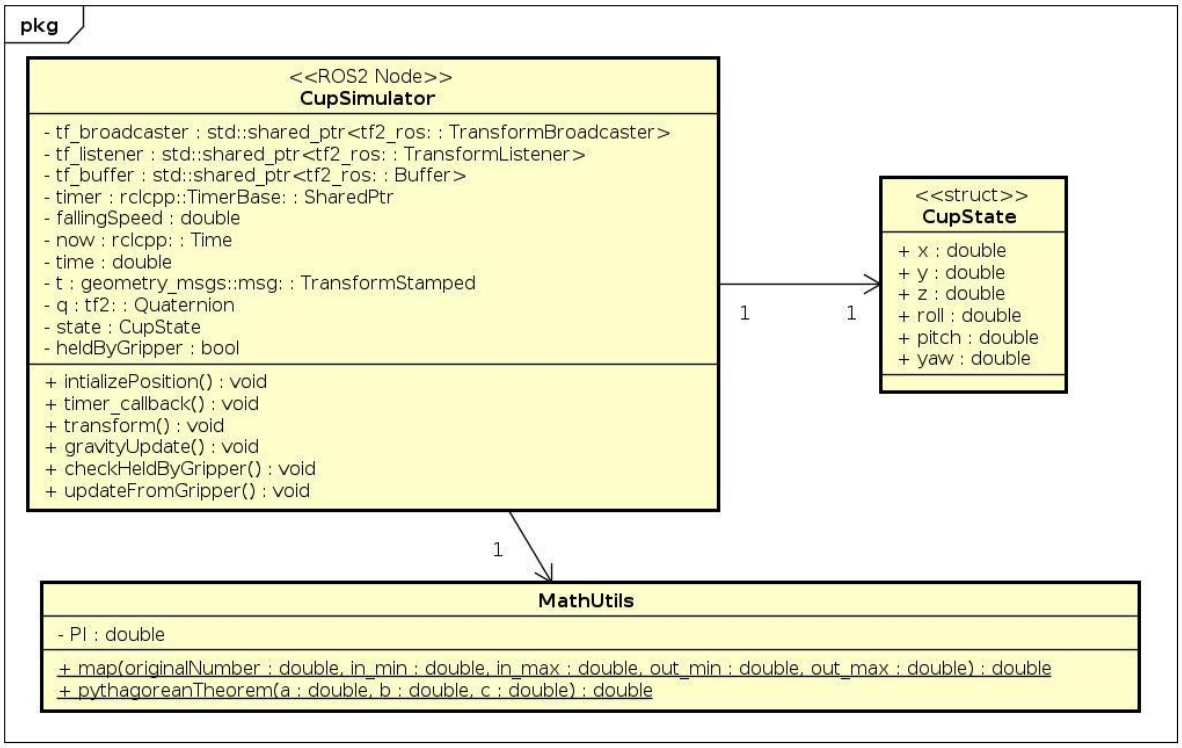
Mocht er verder een commando binnenkomen die kleiner is dan 500 of groter is dan 2500, wordt daar een error voor geprint.

1.2.4 MathUtils

MathUtils is een statische klasse die twee functies bevat. In dit programma wordt de map-functie gebruikt om een PWM van het SSC-32U-formaat te vertalen naar radialen. De andere functie wordt door de cup simulator gebruikt om de afstand tot de gripper van de robotarm bij te houden.

1.3 Klassendiagram cup simulator

In de afbeelding hieronder is het klassendiagram van de simulator van het bekertje te zien.



1.3.1 CupSimulator

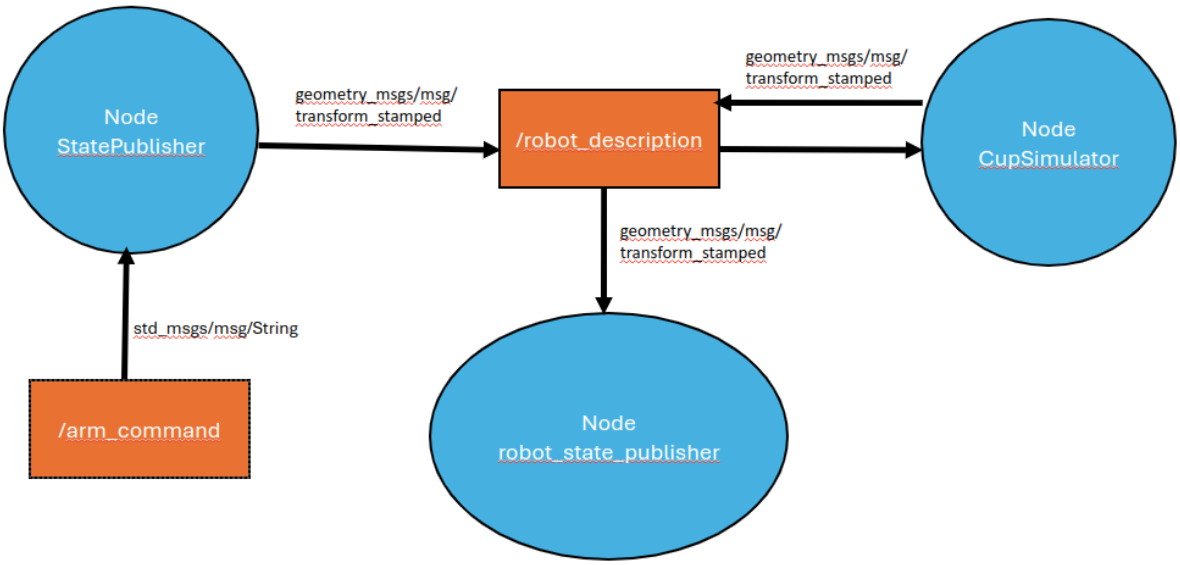
De CupSimulator houdt de positie van het bekertje bij en controleert of deze opgepakt kan worden door de robotarm of niet. De positie en rotatie van het bekertje worden bijgehouden in een instantie van CupState. Verder bevat deze node een TransformListener om bij te houden of de gripper van de robotarm dicht genoeg bij het bekertje is om opgepakt te kunnen worden. Deze TransformListener wordt ook gebruikt om het bekertje mee te laten bewegen met de hand van de robotarm als deze is opgepakt.

1.3.2 MathUtils

MathUtils wordt hier gebruikt via de functie ‘pythagoreanTheorem’. Deze gebruikt de stelling van Pythagoras in een driedimensionale ruimte om te berekenen wat de afstand tussen het bekertje en de gripper van de robotarm is, zodat kan worden gecontroleerd of het bekertje kan worden opgepakt.

1.4 Nodes en topics

In de afbeelding hieronder is te zien hoe het programma communiceert via nodes en topics.



De StatePublisher luistert via /arm_command naar berichten die alleen een string bevatten. De string moet een commando van het SSC-32U-formaat zijn, zodat de robotarm kan bewegen. De commando's kunnen via de terminal worden gestuurd, bijvoorbeeld met:

```
ros2 topic pub -1 /arm_command std_msgs/msg/String "{data: '#0 P1500 #1 P1300 #2 P500 #3 P1500 #5 P1500 T2000'}"
```

De StatePublisher en de CupSimulator kunnen allebei een transform_stamped sturen met de positie en rotatie van elke joint. De CupSimulator ontvangt deze ook nog, omdat deze de positie van de gripper van de robotarm nodig heeft, zodat er kan worden gecontroleerd of het bekertje wordt vastgehouden door de robotarm.

De ingebouwde robot_state_publisher ontvangt transform_stamped-berichten vanuit de StatePublisher en de CupSimulator, zodat de bewegingen van de robotarm en het bekertje in Rviz kunnen worden getoond.