Ontwerp Documentatie

HAN Arnhem

Versie 1

WOR-World

René van Eendenburg 561378
Derk Wiegerinck 567665

10 juni 2020

Inhoudsopgave

1 Inleiding

In dit document wordt het ontwerp van de uitwerking van de simulatieopdracht beschreven. Hier zal worden ingegaan op de structuur van de packages, Dit houdt in dat alle nodes, topics en messages worden beschreven. Ook wordt beschreven wat de samenhang is van de broncode. Dit wordt met behulp van class diagrams en beschrijvingen van de classes gedaan. Daarnaast wordt er beschreven hoe het gedrag van de belangrijke componenten gerealiseerd is. Tot slot wordt de API van alle publieke interfaces beschreven.

2 Beschrijving van de packages

De uitwerking bestaat uit twee ROS-packages: al5d_simulation en cup. In de volgende paragrafen worden de packages beschreven.

2.1 al5d_simulation

De al5d_simulation package bevat een node die de al5d controller simuleert en de visualisatie van de lynxmotion al5d.

De controller accepteert berichten van het SSC32U-format op een rostopic. Het rostopic, genaamd "SSC32U_request_topic", handelt alle mogelijke requests af. Responses zijn op dit moment niet geïmplementeerd, omdat dit buiten de scope van de opdracht ligt.

Bij een request zal de al5d_simulation de SSC32U requests converteren en vervolgens publiceren op het "joint_state_message_topic". Deze worden op hun beurt weer afgehandeld door de "robot_state_publisher". Dit topic zal de robot in de Rviz-visualisatie aansturen.

2.2 cup

De cup package bevat een node die een beker kan simuleren. De cup luistert naar de frames die de al5d_simulation publiceert. In de volgende sectie wordt uitgelegd wat de relevante frames zijn en waarom.

2.2.1 Relevante frames

De cup maakt gebruik van de door al5d_simulation gepubliceerde transform-frames van de robot. De belangrijke frames waar de cup geïnteresseerd in is, zijn: “odom”, “gripper_right” en “gripper_left”.

“odom” is het frame van de simulatie dat altijd vast staat. Met behulp van dit frame, worden de posities van de beker in de gesimuleerde wereld berekend.

De andere twee frames zijn de gripper-frames die worden gebruikt om de positie van de beker ten opzichte van de gripper te bepalen. Ook worden deze frames gebruikt om sensoren aan de gripper te koppelen (dit wordt verderop in het document uitgelegd).

2.2.2 Verschillende onderdelen van de cup

De beker bestaat uit een marker en een transform-frame. De marker is een visuele representatie van de beker. Dit bepaalt hoe de beker eruitziet. Aan de hand van het transform-frame wordt bepaalt waar de beker precies is in de gesimuleerde wereld.

Om te bepalen wanneer de beker opgepakt is, zijn er "sensoren"gerealiseerd op de gripper. Dit zijn markers die de positie van de grippers met een bepaalde offset volgen. Als de sensoren de beker detecteren, is de cup opgepakt. Ook zijn er twee sensoren gerealiseerd aan de buitenkant van de grippers. Op deze manier kan de beker verschoven worden. Deze sensoren bepalen uiteindelijk de toestand van de beker. De volgende toestanden zijn mogelijk: opgepakt, niet opgepakt, schuiven naar links en schuiven naar rechts.

Na het bepalen van de toestand, kunnen er een aantal zaken worden gedaan. Als de beker niet opgepakt is, wordt er een zwaartekracht gesimuleerd. Dit zorgt ervoor dat als de positie van de beker boven de grond is, de positie wordt verminderd met een factor totdat de beker de grond heeft bereikt. Als de beker opgepakt is, wordt het transform-frame van de beker gekoppeld aan de gripper, zodat de beker de gripper volgt. Dit wordt ook gedaan in de overige toestanden.

Elke cup publiceert zijn snelheid, versnelling en oriëntatie op topics in de vorm van "cup" + id + "_" + ("speed" / "velocity" of "orientation"). Daarnaast wordt de marker van de cup wordt gepubliceerd zodat deze zichtbaar is in de wereld en het transform-frame wordt gepubliceerd. De cup publiceert ook de sensors (deze bestaan ook uit visuele markers en transform-frames). De markers zijn in de simulatie te zien als "left_sensor", "right_sensor", "left_outer_sensor" en "right_outer_sensor".

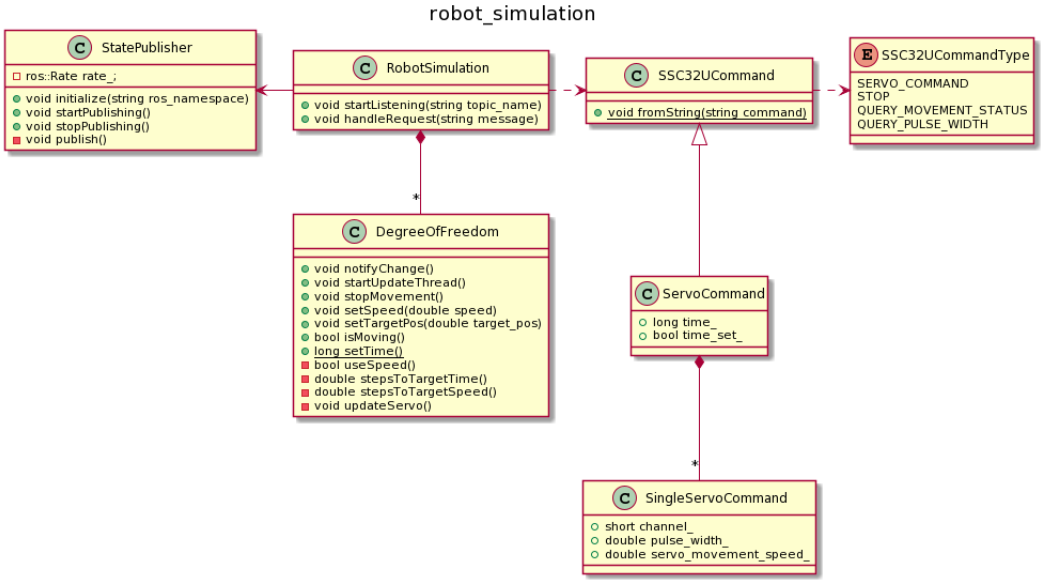
2.2.3 Starten van de cup

Om een beker in de simulatie wereld te weergeven, moeten de volgende argumenten worden meegegeven. Een numeriek ID, X locatie, Y locatie en een Z locatie.

3 Structuur van de packages

3.1 al5d_simulation

De al5d is opgezet in de volgende onderdelen. Een command parser, servo aansturing, een publisher en een controller klasse. Na het klassediagram zal er een kleine beschrijving van elk onderdeel staan.



Figuur 1: al5d_simulation class diagram

3.1.1 SSC32UCommand

Deze klasse parsed SSC32U string commando's en geeft een SSC32U object terug. Dit object kan een ServoCommand zijn, deze kan meerdere SingleServoCommands bevatten. Hierin zitten het kanaal voor de servo, de gewenste positie en de snelheid voor de servo. In een ServoCommand kan ook een globale tijd worden gezet. De keus voor individuele servo snelheid of de globale tijd wordt gebruikt, wordt afgehandeld in DegreeOfFreedom.

3.1.2 DegreeOfFreedom

Een object van deze klasse dient als een enkele servo. De servo ontvangt een target positie waar de servo naar toe moet bewegen. De servo zal vervolgens berekenen, op basis van de gegeven snelheid, tijd of limitaties van de servo, in hoe veel stappen er bewogen wordt naar de target positie. De limitaties van de servo worden opgehaald van de Ros Parameter server. Deze parameter server wordt geladen met een xarco file.

3.1.3 StatePublisher

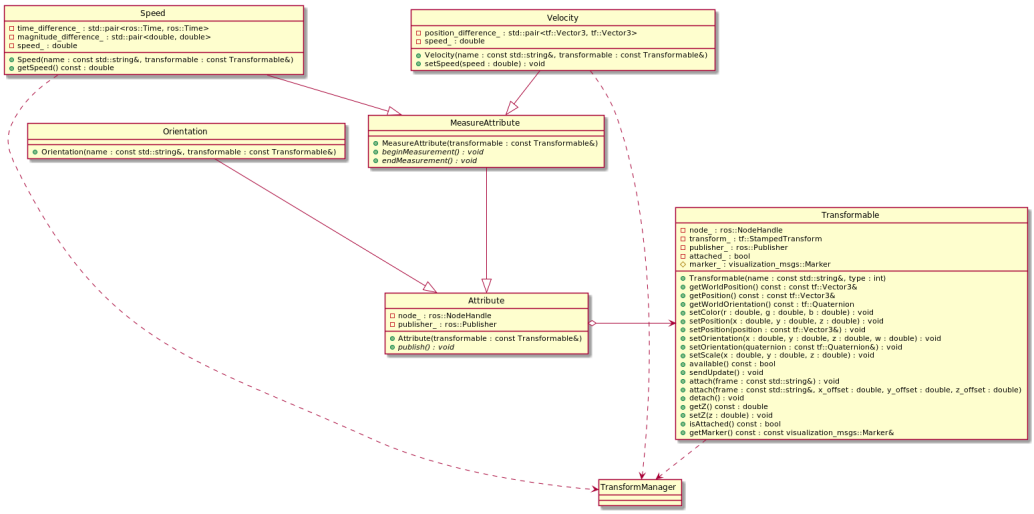
Deze klasse benodigd een lijst met alle servo's. Vervolgens zal deze met een vaste rate door de lijst heen loopen in de functie Publish en daarmee de configuratie van de robot arm doorsturen naar het "joint_state_message_topic".

3.1.4 RobotSimulation

De RobotSimulation is de controller van de al5d_simulation. Deze handelt de requests af die binnen komen op het rostopic.

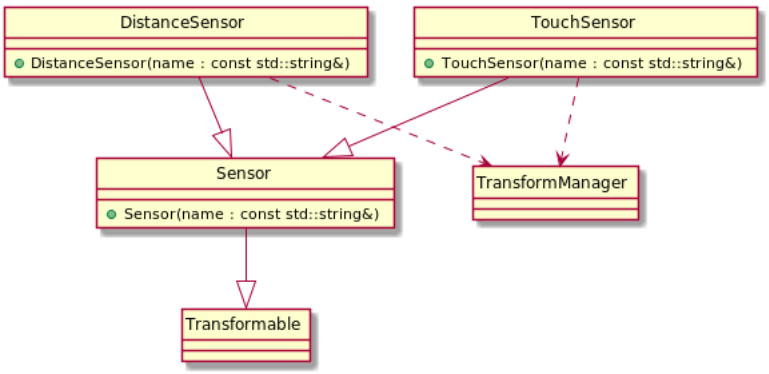
3.2 cup

De cup bestaat uit een aantal onderdelen: de classes voor de marker, classes voor de sensor en de overige classes. In de code zijn deze classes onder namespaces verdeeld. De lege classes zijn verwijzingen naar classes die behoren tot een andere namespace.



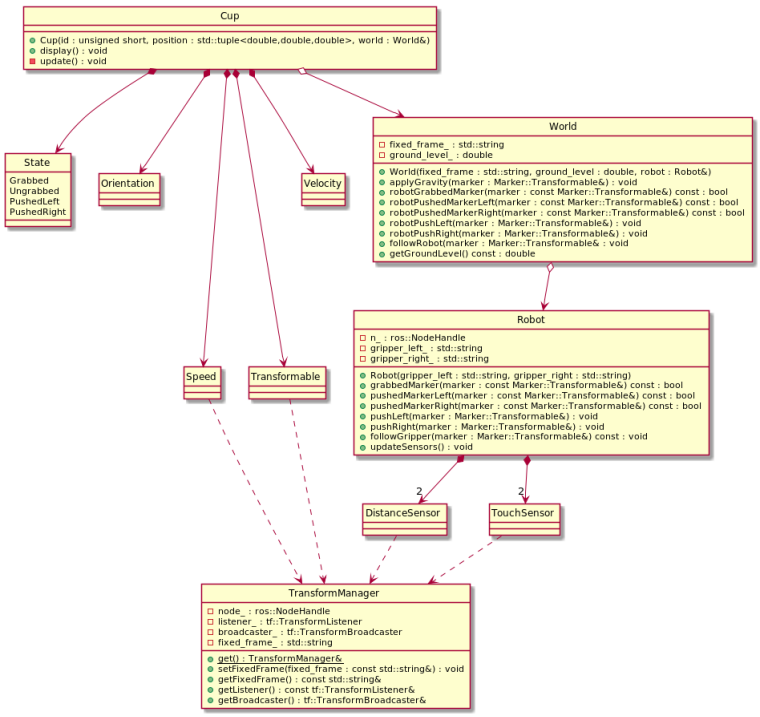
Figuur 2: cup class diagram van de marker

In Figuur 2 zijn de classes weergeven die behoren tot de namespace marker. Dit bevat alle functionaliteit van een marker: de visuele representatie, het transform-frame en attributen van een marker als snelheid, oriëntatie en versnelling.



Figuur 3: cup class diagram van de sensor

In Figuur 3 zijn de classes weergeven die behoren tot de namespace sensor. Dit bevat alle functionaliteit van een sensor. Een sensor is een marker met een vaste grootte en vorm en kan worden gebruikt om een andere marker (zoals een beker) te detecteren.



Figuur 4: cup class diagram van de overige classes van de cup

In Figuur 4 zijn de overige classes weergeven. Deze zijn niet onderverdeeld in een specifieke namespace. Dit bevat de classes die voornamelijk gebruik maken van de classes van andere namespaces en op zichzelf niet veel functionaliteit toevoegen.

In de volgende secties worden de individuele classes beschreven.

3.2.1 Attribute

Deze class definieert een interface om een attribuut van een marker te kunnen maken. Een attribuut is een onderdeel van een marker dat gepubliceerd kan worden. Het bevat het absolute minimum wat benodigd is voor het publiceren van een attribuut.

3.2.2 MeasureAttribute

Deze class definieert een interface om een attribuut van een marker te kunen maken, waarbij iets (zoals tijd) gemeten moet worden.

3.2.3 Transformable

Deze class bevat een visuele representatie van de marker en het transform-frame van de marker. Het houdt beide representaties bij elkaar. Dit biedt het voordeel als het transform-frame wordt geüpdatet, de visuele represenatie ook wordt geüpdatet.

3.2.4 Speed

Deze class kan worden gebruikt om de snelheid te meten van een marker en om dit te publiceren.

3.2.5 Orientation

Deze class kan worden gebruikt om de oriëntatie van de marker te meten en te publiceren.

3.2.6 Velocity

Deze class kan worden gebruikt om de versnelling van een marker te meten en te publiceren.

3.2.7 Sensor

Deze class definieert een interface om specifiekere sensoren te kunnen maken. Het bevat het absolute minimum om een sensor te definiëren. De sensor is een marker van vaste grootte en vorm.

3.2.8 TouchSensor

Deze class kan worden gebruikt om te detecteren of een marker de TouchSensor aanraakt.

3.2.9 DistanceSensor

Deze class kan worden gebruikt om te detecteren of een marker het midden van de DistanceSensor aanraakt. Het verschil tussen de TouchSensor en DistanceSensor is dat de TouchSensor een collision detecteert als een marker de sensor aanraakt. De DistanceSensor detecteert een collision op basis van het midden van de sensor.

3.2.10 Cup

De Cup bevat een marker, verschillende marker-attributen en de toestand van de beker (opgepakt, niet opgepakt, verschoven naar links of rechts). Ook heeft het een referentie naar de World om zo te controleren of de Robot iets met de beker heeft gedaan en om zwaartekracht op de beker toe te kunnen passen.

3.2.11 World

De World is een toegangsluik van de Cup naar de Robot. De beker weet in principe niets van de robot, maar weet wel van de wereld. De World heeft een vast transform-frame en weet ook waar de grond is. Daarnaast kan de World zwaartekracht op de Cup toepassen.

3.2.12 Robot

De Robot is een representatie van de Robot in het perspectief van de Cup. Het bevat dus ook alleen de voor de Cup relevante transform-frames. Om te kunnen detecteren of de beker is opgepakt of is verschoven, heeft de robot twee DistanceSensors aan de binnenkant van de gripper en twee TouchSensors aan de buitenkant van de gripper.

3.2.13 TransformManager

De TransformManager is een singleton zodat classes makkelijker bij de TransformListener en TransformBroadcaster kunnen komen. Ook weet de TransformManager wat het vaste transform-frame is.