Samenvatting RA

# Containerized ROS architectures

**Kan in eigen woorden het verschil uitleggen tussen containers en virtual machines**

Elke vm heeft zijn eigen OS terwijl containers hetzelfde OS-systeem gebruiken. Containers zijn light weighted en kunnen ervoor zorgen dat er minder overhead is. Nog een verschil is dat vm’s draaien op hypervisor en containers op Docker.

**Kan uitleggen wat docker is**

Docker is een set van Platform as a Service (PaaS) producten die op OS-niveau virtualisatie gebruiken om software te leveren in pakketten die containers worden genoemd. Containers zijn geïsoleerd van elkaar en bundelen hun eigen software, libraries en configuratiebestanden; ze kunnen met elkaar communiceren via goed gedefinieerde chanels. Alle containers worden beheerd door een enkele kernel van het besturingssysteem en zijn dus lichter dan virtuele machines.

**Kan in eigen woorden het verschil uitleggen tussen containers en images**

Docker images vormen de basis van containers. Images zijn eigenlijk frozen immutable snapshots van live containers. Containers zijn running instances van een image

**Kan beschrijven wat registries zijn**

Een Docker-register is een opslag- en distributiesysteem voor benoemde Docker images. Dezelfde images kunnen meerdere verschillende versies hebben, te herkennen aan hun tags. Een Docker-register is georganiseerd in Docker-repositories, waar een repository alle versies van een specifiek image bevat.

A Registry is a hosted service containing repositories of images which responds to the Registry API.

**Kan uitleggen wat docker volumes zijn**

Docker volumes dient om data op te slaan en te sharen tussen containers. Een volume is een directory die op het host os staat. Daarin worden dan wijzigingen opgelsagen en meegekopieerd.

# Week 2

**Kan uitleggen wat een docker container is**

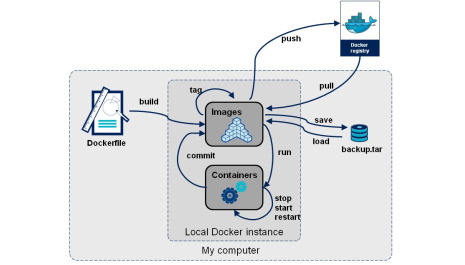
* Een container is een standaardeenheid van software die code en al zijn afhankelijkheden verpakt, zodat de toepassing snel en betrouwbaar van de ene computeromgeving naar de andere wordt uitgevoerd.
* Een Docker-containerimage is een lichtgewicht, stand-alone, uitvoerbaar softwarepakket dat alles bevat wat nodig is om een applicatie uit te voeren: code, runtime, systeemtools, systeembibliotheken en instellingen.
* Container images worden containers tijdens runtime en in het geval van Docker containers- images worden containers wanneer ze runnen op Docker engine
* Op elke machine werkt de containerized software hetzelfde omdat je de container in heel zijn staat overzet

**Kan de voordelen van een container beschrijven**

* Containers moeten niet booten staan direkt op
* Cloud ready
* Leightweighted
* Host isolation
* Images zijn self-contained
* Minimal overhead

**Kan elke docker stage beschrijven**

* Er wordt een docker file gemaakt
* Die file wordt gebuild en daaruit krijg je een immage tijdens het builden wordt er een tagname aan de image gegeven die kan je zelf inzetten door -t
* De image kan ook afgehaald worden vaan een docker repository of net pushen naar een docker repo
* Die image wordt ook ergens lokaal opgeslagen
* Daarna worden containers aangemaakt en gerunt van die image
* Die containers kunnen herstart worden gestopt worden,….
* Als er aanpassingen aan de containers worden gedaan kunnen deze eventueel meegecommit worden naar de image
* Anders ben je deze kwijt
* Deze update van image kan dan weer eventueel gepushed worden naar docker regestry



**Kan de link tussen decoupled applicaties en meerdere containers uitleggen**

Elke container moet maar één zorg hebben. Het ontkoppelen van applicaties in meerdere containers maakt het eenvoudiger om horizontaal te schalen en containers opnieuw te gebruiken. Een stapel webapplicaties kan bijvoorbeeld bestaan uit drie afzonderlijke containers, elk met een eigen unieke afbeelding, om de webapplicatie, de database en een in-memory cache op een ontkoppelde manier te beheren.

# Introduction to drones

**Kan in eigen woorden uitleggen wat een drone is in context van robotics**

Een onbemande op afstand bestuurbaar vliegtuig.

**Kan de afkorting UAV en RPAS verklaren**

UAV: Unmanned Aerial Vehicle (onbemand luchtvoertuig),

RPAS: Remotly piloted aircraft systems (op afstand bestuurbaar vliegtuig systeem).

**Kan een taxonomie voor UAV's / Drones opmaken**

DRONE-hardware: Fixed wings, mulyi rotor, hybrid

DRONE-category: Toy, Hobby, Professional

DRONE-application: GIS, security, movie, agriculture

DRONE-user: Professional, R&D

**Kan beschrijven waarom drones populair zijn geworden in de jaren 2010**

Prijzen zijn drastisch gedaald samen met het gewicht van de componenten

**Kan UAV's indelen in de overeenkomstige typen: fixed wing, multirotor / multicopter, quadcopter, hexacopter, hybride**

Fixed wings: Drones met vaste vleugels blijven langer in de lucht hangen. Als de moter uitvalt kunnen ze nog veilig zwevend aan de grond gezet worden.

Multirotor: Ziijn drones met meer dan 2 motoren. Zij kunnen niet zolang in de lucthblijven als fixed wings drones. Ook als de batterij plat is dan vallen ze gewoon rechtsreeks neer uit de lucht.

Quadcopter: een mulitrotor drone met 8 motors.

Hexacopter: multirotor drone met 6 motors.

Hybrid drones: mix van fixed wing en multicopter. Kan gewoon vliegen maar ook vliegen als een multi copter

**Kan de verschillende dronegebruiken vermelden en minstens één voorbeeld per gebruik geven**

* Military
  + Terroristische bendes stoppen vanuit een onbemand vliegtuig zodat geen piloot of schutter in gevaar is
* Foto en film
  + Vroeger was het heel duur om bv uw huis op hoge afstand te filmen je moest bv al een helikopter laten komen. Nu zijn er drones die dit allemaal kunnen doen
* Entertainment
  + Drons waar lampjes aangemaakt worden voor bv. sfeer te maken
* Infrastructuur inspectie
  + Inspecteren van grote bruggen met behulp van drones en AI
* Gevaarlijke plaatsen observeren
  + Drones laten vliegen bij brand om de omgeven te screenen
* Search en rescue
  + Om mensen op te sporen die vermist zijn
  + Als er een ongeval is gebeurd een drone sturen die een AED brengt
* Leveren van pakketjes
  + UPS die paketjes met drones laten bezorgen naar bv een eiland
* Inventory/stock control
  + Sommige opbergplaatsen zijn te enorm om door een klein team te laten controleren
  + Daarom kunnen drones gebruikt worden die dit in de mensen hun plaats doen en sneller
* Landbouw
  + Drones kunnen rondvliegen op regelmatig te controleren op boomziektes
* Hobby, sport, plezier
  + Drone race

**Kan verklaren waarom drone-veiligheid en wetgeving een noodzaak is.**

* Crashes
  + Als een drone crasht kan dit serieuze gevolgen hebben zoals
    - Vingers in stukken gesneden
    - Een deel uit het gezicht gesneden door de impact van een drone
    - Drones die in de propellors van de grote boeings graken en zo mogelijk vliegtuig crashen kunnen veroorzaken
* Privacy
  + Aan drones hangen vaak camera’s dit zorgt ervoor dat andere mensen worden gefilmd
  + Schending van privacy

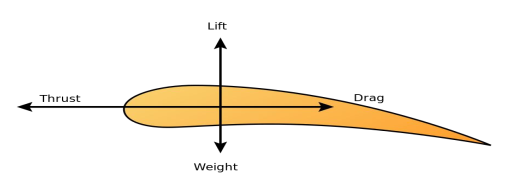
# Drone physics

**Kan het verschil beschrijven tussen een aerostatic en aerodyne vliegtuigen**

* Aerostatic
  + Lichter dan lucht
* Aerodyne
  + Zwaarder dan lucht
  + Krijgt zijn lift door beweging

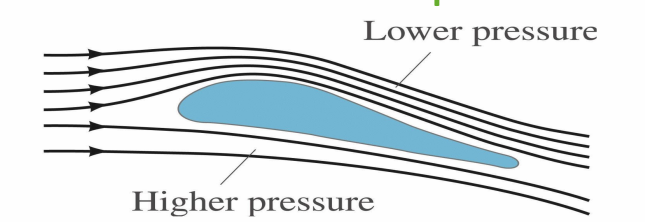
**Kan de vier verschillende krachten uitleggen die ontstaan tijden het vliegen**

* Thrust
  + Stuwkracht
  + Ontstaat door een propellor of een straalmoter
  + Is de aerodynamische kracht die het vliegtuig door de ruimte naar voren duwt of trekt
  + Om een vliegtuig te laten vliegen moet de stuwkracht gelijk zijn of groter zijn dan de drag
  + Als om welke reden dan ook de drag groter is zal het vliegtuig vertraken
  + Als de thrust groter wordt zal het vliegtuig versnellen
* Drag
  + Weerstand
  + Tegenovergestelde kracht van Thrust
  + Of de wrijving die de beweging weerstaat van een object dat door een vloeistof beweegt (of immobiel in een bewegende vloeistof, zoals optreedt wanneer u met een vlieger vliegt)
  + Denk aan je hand uit het raam steken van een rijdende auto
  + Deze kracht heeft bepaalde factoren
    - De grote van het vliegtuig
    - De snelheid
    - De dichtheid
  + Omdat deze kracht enorm is trekken passagiersvliegtuigen hun landing gear in vlak nadat ze zijn opgestegen anders scheurt deze van het vliegtuig door de kracht
* Lift
  + Tegen gestelde kracht van weight
  + Dit zorgt ervoor dat een vliegtuig in de lucht houdt
  + Deze prestatie wordt bereikt met behulp van een vleugel (vleugelprofiel)
  + Net als draf kan lift alleen bestaan in een aanwezigheid van een bewegende vloeistof
  + Het maakt niet uit of het object stilstaat en de vloeistof beweegt (zoals bij een vlieger op een winderige)
  + Wat echt belangrijk is, is het relatieve verschil in snelheid tussen het object en de vloeistof
  + Wat de feitelijk liftmechanismen, treedt de kracht op wanneer een bewegende vloeistof wordt afgebogen door een vast voorwerp
  + De vleugel spplitst de luchtstroom in 2 richtingen
    - Op en over de vleugel
    - Naar beneden langs de onderkant van de vleugel
  + De vleugel is gevormd en gekanteld zodat de lucht die erover beweegt sneller reist dan de lucht die eronder beweegt
  + Wanneer bewegende lucht over een object stroomt en een obstakel tegenkomt (zoals een bobbel of een plotselinge toename van de vleugelhoek)
  + Wordt het pad smaller en versnelt de stroom terwijl alle moleculen zich haasten
  + Eenmaal voorbij het obstakel verbreedt het pad zich en neemt de stroom weer af
  + Hetzelfde principe als je op een tuinslang knijpt versmelt het pad en gaan de moleculen versneld wordn
  + Terwijl de lucht versneld daalt de druk
  + Dus de sneller bewegende lucht die over de vleugel beweegt, oefent er minder druk op uit dan de langzamere lucht
  + Resultaat is een opwaartse duw van lift
* Weight
  + De kracht waarmee het vliegtuig naar de aarde wordt getrokken
  + Product van masse en zwaartekracht



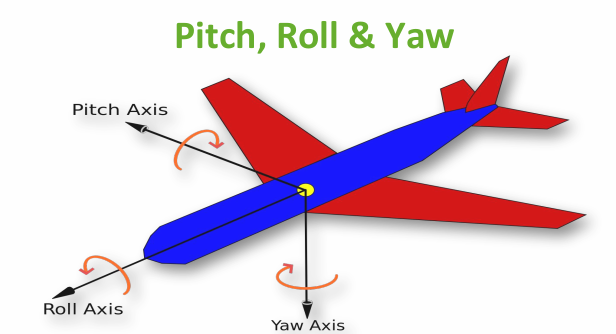
**Kan de Bernoulli’s principe uitleggen**

* Hoe sneller een vloeistof stroomt hoe minder druk erop bevindt
* Terug bij het verhaal van lifting
  + De vleugel is zo langs de bovenkant bol gemaakt zodat de lucht opeen wordt geperst
  + Wat maakt dat de lucht sneller gaat bewegen langs de bovenkant van de vleugel
  + Dus de vleugel deelt de lucht op langs 2 kanten van boven de sneller bewegende lucht (low pressure) en langs de onderkant de traag bewegende lucht (high pressure)
  + Dus de lucht langs de boven gaat gaat een lagere druk hebben omdat deze sneller beweegt wat maakt dat de onderkant van de vleugel een grotere druk heeft en dus het vliegtuig de lucht instuurt



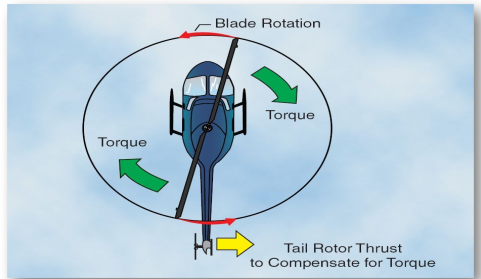
**Kan beschrijven wat pitch, roll en yaw betekent in een figuur om de verschillende assen te voorspellen**

* Pitch axis
  + Pitch is neus naar boven of benede
* Roll axis
  + Rollen
  + Celinder vormige draaiingen maken
* Yaw axis
  + Zeidelinkse beweging
  + Kan niet echt gebruikt worden bij vliegtuigen



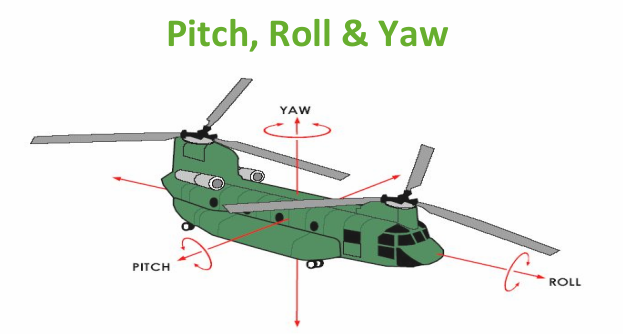
**Begrijpen van blade roation, torque en tail rotor thrust in context van helicopters**

* Blade rotation
  + Draaiing van de blades
  + Doordat de blades gaan draaien ontstaat er torque
* Torque
  + Voor elke actie is er een reactie
  + Omdat de blades gaan draaien ontstaat er een tegenovergestelde kracht (torque)
  + Als we geen tail rotor hebben zou de helicopter spinnen in de tegenovergestelde richting van waar de blades naar draaien
  + Oplossing
    - Tail rotor
    - Anti torque systeem
* Tail rotor
  + Gaat in dit geval de torque kracht probleem opvangen
  + Tail rotor gaat thrust maken
  + Dus hoe harder de blades gaan draaien hoe meer tail rotor thrust er nodig is om dit te counteren
  + Rotatie rond yaw as kan dus door meer of minder tail thrust te maken
  + (Sommige systemen hebben geen tail rotor dit maakt dat er 2 of meerdere blades zijn die elkaars torque opheffen)



**Begrijpen van pitch, roll en yaw in de context van helicopters**

* Yaw
  + Roteren draaien als torque kracht meer is naar de ene kant
  + Als torque kracht lager is dan naar de andere kant
* Roll
  + Rolbewegingen
  + Zijdelings
* Pitch
  + Forward
  + Backward
  + Bepaalde blades krijgen meer pitch dan anderen



**Kan lift, hovering uitleggen in context van helicopters**

* Lift
  + Airfoil principle
  + Hetzelfde verhaals als bij de vleugel van een vliegtuig
  + Turbo shaft engine
  + Compressor gaat koude lucht naar binnen zuigen en gaat deze compressen
  + Fuel wordt ingebrand in de compressed lucht
  + Dit verlaat de combustion kamer en gaat door een series van turbine stages
  + Wat maakt dat de blades gaan draaien
  + 2 turbine sets een voor de compressor en een voor de motor
* Hovering
  + Op 1 plaats blijven zweven
  + Heel moeilijk

**Kan collective pitch rotors helicopters van UAV uitleggen**

* Elke blade op rotor heeft zijn eigen pitch
* Meer manouvers
* Moeilijker

**Kan fixed pitch rotors uitleggen in context van UAV**

* Goedkoper
* Eenvoudiger
* De pitch van elke rotor staat vast is onverandelijk
* Dus om meer pitch in algemeen te krijgen moet je de achter motors meer pitch geven dan de andere

**Kan de verschillende electronische componenten van een uav uitleggen: autopilot, motors, ground station, GPS antenna, radio telemetry, remote control, battery, buzzer, ESC (electronic speed controls)**

* GPS atenne
  + Zorgt ervoor dat de uav gps heeft en zich kan navigeren
* Motors
  + Motors van de uav voor te vliegen
* Radio telemetry
  + Zorgt ervoor dat je de uav kan besturen met een ground station of met software op pc of laptop
* Batterij
  + Geeft de uav stroom
* Buzzer
  + Stuurt audiosignalen dat indiceert wat de uav aan het doen is
* Ground station
  + Een grondcontrolestation (GCS) is een land- of zeegebaseerd controlecentrum dat de voorzieningen biedt voor menselijke controle van onbemande luchtvaartuigen (UAV's of "drones")
* Remote control
* ESC
  + Electronic speed control
  + Gaat de snelheid van elke motor bepalen

**Kan de modes of motion van een quadcopter, directie en snelheid van de rotors uitleggen**

* Motors 1 en 3
  + Gaan tegen de klok in draaien
  + Ontstaan van torque
* Motors 2 en 4
  + Gaan met de klok mee draaien
  + Om torque kracht te onderdukken

**Kan de magnitudes of lift uitleggen**

* Ascend
  + Stijgen
* Descend
  + Dalen
* Hover
  + Zweven

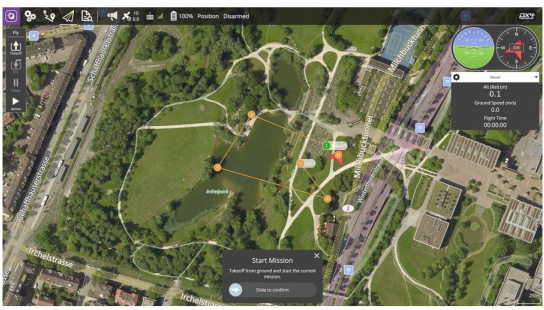
# Rapid drone software prototyping PX4

**Kan in eigenwoorden het px4 project en de software uitleggen**

* PX4 (autopilot) is een open source software voor flight control voor drones en andere onbemande voertuigen
* Het project biedt een flexibele set hulpmiddelen voor drone-ontwikkelaars om technologieën te delen om op maat gemaakte oplossingen voor drone-applicaties te creëren
* PX4 biedt een standaard voor ondersteuning van drone-hardware en softwarestack, waardoor een ecosysteem hardware en software op een schaalbare manier kan bouwen en onderhouden.

**Kan in eigenwoorden MAVLINK, QGroundControl, MAVSDK uitleggen**

* MAVLIK
  + Lightwieghted messaging protocol
  + Voor communicatie met drones (en tussen onboard drone-componenten)
  + Volgt een modern hybride pu-sub en poin-to-point design pattern
    - Data streams worden als onderwerpen verzonden/gepubliceerd
    - Terwijl configuratie sub-protocols zoals het mission protocol of parameter protocol point-to-point zijn bij hertransmissie
  + Berichten worden gedefinieerd in XML-bestanden
  + Elk XML-bestand definieert de berichtenset die wordt ondersteund door een bepaald MAVLINK-systeem (dialect)
  + De reference message set die wordt geimplementeerd door de meeste ground control stations en autopilot wordt gedefinieerd in common.xml (de meeste dialecten worden gebouwd bovenop deze definitie)
  + MIT license
* QGroundControl
  + Biedt volledige flight control en missie planning voor elke drone met MAVLink
  + Primaire doel is voor gebruiks gemak voor professionele gebruikers en ontwikkelaars
  + Open source

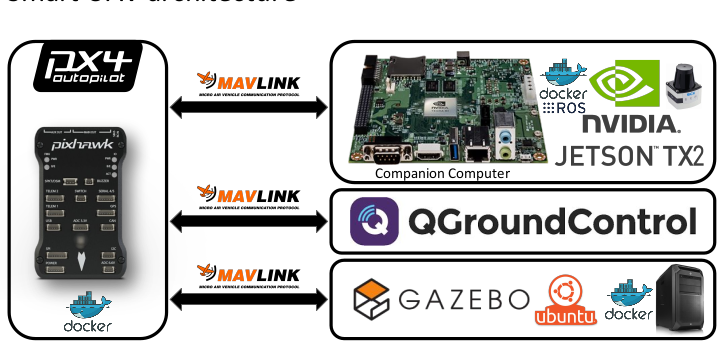


* MAVSKD
  + Nieuwe naam voor de dronecode sdk
  + Standaardenconfome MAVLINK librarues met api’s
  + Gemakkelijkste manier om te integreren met drones via MAVLink

**Kan in eigenwoorden companion computer uitleggen**

* Moederbord dat op de vehicle zit
* Communiceert met de autopilot
* Haalt alle mavlink gegevens op die er op die er geproduceerd worden door de autopilot
* En kan deze gebruiken om intelligente beslissingen te nemen tijdens de vlucht
* Maakt hoge functionaliteiten waard zoals vision werking
* Bevat ROS

**Kan een compleet smart UAV-architectuur tekenen en uitleggen**



# Simultaneous localization and mapping

**Kan in eigenwoorden het SLAM probleem samen met zijn componenten (mapping en localization) uitleggen**

* Voor een roboto autonoom te laten rijden hebben ze 2 belangrijke dingen nodig
  + Mapping
    - Wat is de wereld rondom me
    - Dit is inclusief het gevoel voor verschillende posities, metingen toevoegen om een map te maken en aanname dat hij perfect zijn locatie weet.
    - Gevoel vanuit verschillende posities
    - Integreer metingen om kaart te produceren
    - Veronderstelt perfecte kennis van positie
  + Localization
    - Waar ben ik ergens in de wereld
    - Heeft betrekking tot een sensor die zijn waarden vertaald naar een wereldmodel
    - Er wordt ook de huidige locatie tot de relatieve locatie in het model berekend.
    - Sensiegebonden sensorwaarden aan een wereldmodel
    - Locatie berekenen ten opzichte van het model
    - Gaat uit van een perfect wereldmodel
    - We nemen hier aan dat we een perfect model van de wereld hebben.
* Samen zijn ze SLAM (simultaneous localization and mapping)
* SLAM is het simultaan localiseren en mappen van de onbekende omgeving.

**Kan de link uitleggen tussen SLAM en het “chicken-egg” probleem**

* Wat was er eerst de map of de motion?

**Kan de term accumulation of uncertainty uitleggen in de context van sensors**

* Het berekenen van de verwachte foutmarge (covariantie) tussen frames die de relatieve locatie voorstellen
* Als men een fout heeft bij de eerste frame, wordt de fout van de tweede frame hierbij opgeteld
* Na een reeks frames, is de fout uit de hand aan het lopen
* Oplossing is meerdere referentiepunten.

**Kan long-term position estimation zonder priori information uitleggen**

* Lange termijn positiebepaling zonder voorhandse informatie is niet zo makkelijk
* Om precies te bewegen, moeten we een accurate map hebben
* Maar om een accurate map te hebben, moeten de locaties die de sensors detecteren, precies gekend zijn

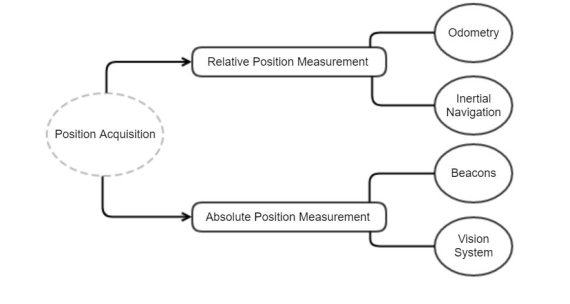
**Kan de accumulation of uncertainty in de context van SLAM uitleggen**

* Indien de initiële locatie van de robot verkeerd wordt geschat
* Wordt dit altijd bij opgeteld bij de volgende berekening
* Hierdoor gaat de voorspelde locatie altijd meer en meer afwijken van de echte locatie.

**Weten dat er meerdere schatting methodes zijn zoals kalman en Particle**

* Kalman
* Particle

**kan lokalisatiemethoden verdelen vanaf de positie-acquisitie (positie registratie).**

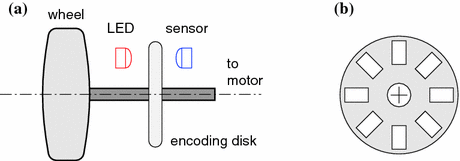


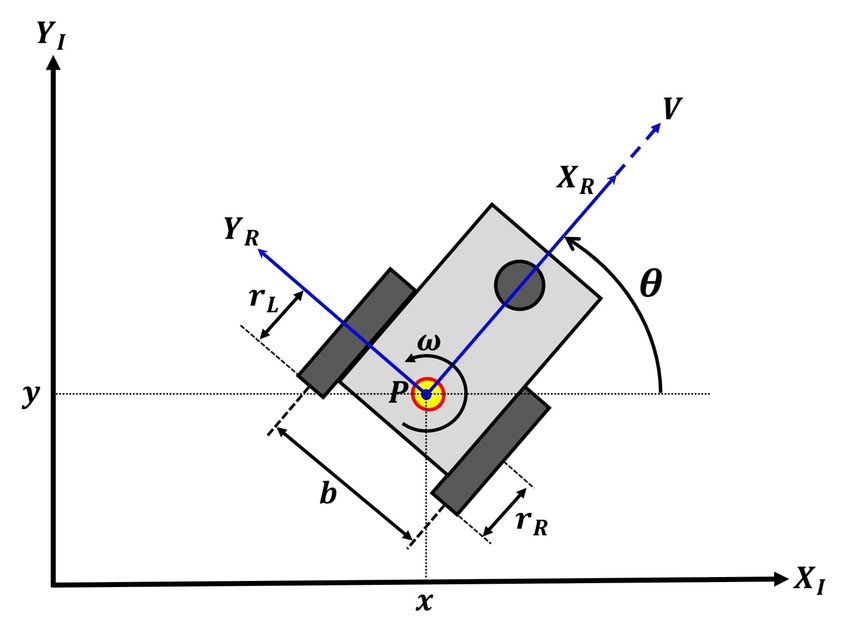
**Kan in eigenwoorden relative en absolute positioning uitleggen als ook de verschillen**

* Relative position measurement
  + Een van de simpelste methoden is gebruiken van wheel odemetry methoden die afhankelijk zijn van encoders om de hoeveelheid rotatie van de wielen te meten
    - Rotatie metingen worden stapsgewijs gebruikt in combinatie met het bewegingsmodel van de robot om de huidige locatie van de robot te vinden ten opzichte van een global reference coordinate system
    - De belangrijkste foutbron is wielslip op oneffen terreinen
    - Positiebepalingen worden gedaan door al de verkregen informatie te integreren
    - Eenmaal voor het verkrijgen van de snelheid
    - Tweemaal voor het verkrijgen van de afgelegde afstand van de robot
    - Systemen zijn onafhankelijk van externe informatiebronnen
    - Aangezien metingen worden uitgevoerd door integratie kan dit leiden tot verhoogde fouten
  + Onafhankelijk van externe informatiebronnen
  + Nadeel
    - Als de schatting een kleine fout heeft wordt deze groter en groter
    - Naar verlang van tijd
* Absolute position measurement
  + Zijn wel afhankelijk voor externe informatie
  + Deze systemen leveren de locatie van de robot dat is onafhankelijk van zijn vorige locaties
  + De locatie is niet gekend door een serie van metingen (voorgaande metingen) maar door slechts 1 meting
  + Voordeel
    - Als er een foutmarge is wordt deze niet uitvergroot naargelang het aantal metingen stijgt

**Kan odometry en intertial navigation uitleggen in de context van absolute positioning measurement**

* odometry
  + Een van de simpelste methoden is gebruiken van wheel odemetry methoden die afhankelijk zijn van encoders om de hoeveelheid rotatie van de wielen te meten
    - Rotatie metingen worden stapsgewijs gebruikt in combinatie met het bewegingsmodel van de robot om de huidige locatie van de robot te vinden ten opzichte van een global reference coordinate system
    - De belangrijkste foutbron is wielslip op oneffen terreinen





* Intertial navigation
  + Net als bij odometry worden positiebepalingen van intertial navigation verkregen door de verkregen informatie van de sensoren te integreren
  + Eenmaal voor het verkrijgen van de snelheid
  + Tweemaal voor het verkrijgen van de afgelegde afstand van de robot
  + Dead recognition
    - Berekenen van een positie aan de hand van vorige posities en bevorderen van die positie door rekening te houden met snelheid en afgelgegde tijd en course
* Net zoals bij odometry worden deze pebaald door integratie wat maakt dat fouten uit de vorige metingen vergroot worden
* Deze metingen zijn onafhankelijk van externe informatiebronnen

**Kan beacons en vision systemen in de context van absolute position measurement uitleggen**

* Worden gebruikt om een nauwkerig positiebepaling te krijgen
* Zijn afhankelijk van elkaar

**Kan odometry uitleggen en een voorbeeld geven**

* odometry
  + Een van de simpelste methoden is gebruiken van wheel odemetry methoden die afhankelijk zijn van encoders om de hoeveelheid rotatie van de wielen te meten
    - Rotatie metingen worden stapsgewijs gebruikt in combinatie met het bewegingsmodel van de robot om de huidige locatie van de robot te vinden ten opzichte van een global reference coordinate system
    - De belangrijkste foutbron is wielslip op oneffen terreinen of slippery flours

**Kan dead reckogning uitleggen**

* Het proces van het berekenen van iemands huidige positie met behulp van een vooraf bepaalde positie
* En het bevorderen van die positie op basis van bekende of geschatte snelheden over verstreken tijd en koers

**Kan intertial navigation uitleggen samen met de term IMU**

* Intertial navigation
  + Net als bij odometry worden positiebepalingen van intertial navigation verkregen door de verkregen informatie van de sensoren te integreren
  + Eenmaal voor het verkrijgen van de snelheid
  + Tweemaal voor het verkrijgen van de afgelegde afstand van de robot
  + Dead recognition
    - Berekenen van een positie aan de hand van vorige posities en bevorderen van die positie door rekening te houden met snelheid en afgelgegde tijd en cours
  + IMU
    - Wordt ook gebruikt om de lineaire en rotatieversnelling van de robots te meten. Het kan echter nog steeds last hebben van factoren zoals uitgebreide drift en gevoeligheid voor hobbelige grond.
    - inertial measurement unit (traagheidsmeeteenheid)

**Kan de belangrijkste foutbron van roterende encoders aangeven**

* De belangrijkste foutbron is wielslip op oneffen terreinen of slippery flours

**Kan drive en andere problemen met IMU's verklaren**

**Kan de link uitleggen tussen LIDAR, optical sensors, wifi, gps en absolute positioning measurement**

* Absolute position measurement
  + Absolute positie meting is afhankelijk van meerdere sensoren
  + LIDAR, optical sensors
    - Afstand meten tussen robot en object
    - Kijken welke objecten er rond hem bevinden
  + Wifi localization
    - Op grafieken gebaseerde wifi-kaart door de signaalsterkte over het veld te verzamelen
    - In deze benadering, worden de gemiddelde en standaardafwijking van wifi RSSI-waarnemingen benaderd door lineaire interpolatie op een grafiek
    - Dit leidt tot een rekenkundig efficiënte observatie waarschijnlijkheidsfunctie en de geoptimaliseerde locatie kan worden afgeleid uit de waarschijnlijkheidsfunctie
  + GPS
    - Stel je voor dat je ergens op aarde staat met drie satellieten in de lucht boven je. Als u weet hoe ver u van satelliet A bent, weet u dat u zich ergens op de rode cirkel moet bevinden. Als u hetzelfde doet voor satellieten B en C, kunt u uw locatie bepalen door te kijken waar de drie cirkels elkaar kruisen. Dit is precies wat uw GPS-ontvanger doet, hoewel deze overlappende bollen gebruikt in plaats van cirkels.
    - Hoe meer satellieten zich boven de horizon bevinden, hoe nauwkeuriger uw GPS-apparaat kan bepalen waar u bent.
    - Accurancy
      * GPS saltalieten zenden hun signalen uit met een zekere nauwkeurigheid
      * Hangt af van bepaalde factoren
        + Satellite geometry
        + Signal blockage
        + …
      * Als voorbeeld
        + GPS op smartphone zijn accuraat tot 4.9 m

Hoewel dit kan verergeren bij een boom of een gebouw of een brug

* + - * High-end users boosts gps kunnen tot centimeters nauwkeurig zijn
  + Al deze metingen samen komen tot een nauwkeurige positiebepaling

**Kan uitleggen waarom wifi niet een goede sensor is in de context van SLAM**

* Beperkt tot het gebied met wifi-signaal
* Vooraf aangeleerde wifi-grafiek

**Kan minimum 2 problemen geven van GPS’s sensors**

* Signalen die geblokt worden door wolkenkrabbers
* Signalen die gereflecteerd worden op glas bv en zo een iets ander resultaat bekomen
* Geen indoor GPS

**Kan in eigenwoorden de term visual odometry (VO) uitleggen**

* Is het proces van het bepalen van equivalente odometrie-informatie met behulp van opeenvolgende camerabeelden om de afgelegde afstand te schatten
* Zorgt voor betere navigatienauwkeurigheid voor robots of voertuigen met elk type voortbeweging op elk oppervlak

**Kan het verschil beschrijven tussen monocular en stereocamera’s in de context van VO**

* Hangt van de camera setup vast
  + Monocular
    - Single camera
  + Stereocamera's
    - Two cameras in stereo setup

**Kan het verschil uitleggen tussen feature based en directed method in de context van VO**

* Feature based
  + Traditionele VO’s visual information is gekregen door feature based
  + Deze gaat extract image feature points and tracking them in the image sequence
* Recente ontwikkeling in VO-onderzoek bood een alternatief, Direct Method genaamd
  + Dat pixelintensiteit in de beeldreeks direct als visuele invoer gebruikt
  + Er zijn ook hybride methoden.

**Kan visual inertial odometry beschrijven in eigen woorden**

* Als een traagheidsmeeteenheid (IMU) binnen het VO-systeem wordt gebruikt, wordt dit meestal Visual Inertial Odometry (VIO) genoemd.

**Kan een 6 stap generec VO algorithme beschrijven**

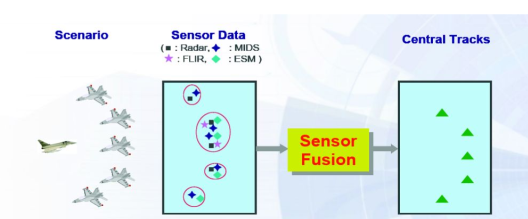
1. Verkrijg input images door single camera’s, stereo camera’s of omnidirectional camera’s
2. Immage correction: immage processing technieken toepassen voor lens distortion te verwijderen
3. Feature detection: definieer interesseoperatoren en combineer features over frames en construeer een optische stroom (kijken welke features er belangrijk zijn om de afstand te bepalen)
   1. Gebruik correlatie om correspondentie (overeenkomsten) tot stand te brengen, en geen feature trakcing op lang termijn
   2. Feature extraction en corralation
   3. Construcxt optical flow field (lucas-kanade method)
4. Controleren van flow field vectoren voor potentieel tracking van errors en outliers verwijderen
5. Schatting van de camerabeweging van de optische stroom
   1. Optie 1: Kalman filter voor state estimate distribution maintenance.
   2. Optie 2: zoek de geometrische en 3D-eigenschappen van de functies die een kostenfunctie minimaliseren op basis van de herprojectiefout tussen twee aangrenzende afbeeldingen. Dit kan worden gedaan door wiskundige minimalisatie of willekeurige steekproeven.
6. Periodieke herpopulatie van trackpoints om coverage in de afbeelding te behouden.

**Kan egomotion beschrijven**

* Egomotion wordt gedefinieerd als de 3D-beweging van een camera in een omgeving
* Op het gebied van computer vision verwijst egomotion naar het schatten van de beweging van een camera ten opzichte van een starre scène
* Een voorbeeld van egomotionschatting is het schatten van de bewegende positie van een auto ten opzichte van lijnen op de weg of straatborden die vanuit de auto zelf worden waargenomen. De schatting van egomotion is belangrijk in autonome robotnavigatietoepassingen.

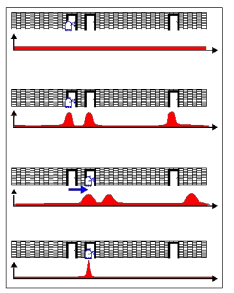
**Kan het voordeel uitleggen van multi sensor fusion en kan een voorbeeld tekenen voor de uitleg te verduidelijken**

* Minder onzekerheden dan wanneer bronnen afzonderlijk gebruikt zouden worden



**Kan het basisidee van markov localization uitleggen en een simpel figuur van tekenen voor de uitleg te verklaren**

* Verhelpt het probleem van toestandsschatting van sensor gegevens
* Is een probalisitsisch algoritme
  + In plaats van een enkele hyptose te handhaven over waar ter wereld een robot zou kunnen zijn
  + Handhaaft Markov localisatie een waarschijnlijkheids hypothesen
  + De probabilistische weergave stelt het in stat deze verschillende hypothesen op een wiskundige manier te wegen
* Voorbeeld
  + Overweeg de omgeving die wordt voorgesteld in de figuur
  + Laten we ervan uitgaan dat de wereld eendimensionaal is
  + Dat wil zeggen dat de robot alleen maar horizontaal kan bewegen (hij mag niet roteren)
  + Stel nu dat de robot ergens in deze omgeving wordt geplaatst, maar de locatie niet wordt verteld
  + Markov-localisatie vertegenwoordigd deze onzekerheid door een uniforme verdeling over alle posities, zoals weergegeven in de grafiek in het laatste diagram
  + Laten we nu aannemen dat de robot zijn sensoren opvraagt en ontdekt dat hij zich naast een deur bevindt
  + Markov-localisatie wijzigt het geloof door de kans op plaatsen naast een deur te vergroten en ergens anders te verlagen
  + Dit wordt geillustreerd in het tweede diagram
  + Merk op dat de resulterende overtuigingen multimodaal is, wat het feit weerspiegelt dat de beschikbare informatie onvoldoende is voor wereldwijde lokalisatie
  + Merk op dat plaatsen die niett naast een deur liggen nog steeds een niet nul kans hebben
  + Dit komt omdat de sensorwaarden noisy zijn en een enkel zicht op een deur meestal onvoldoende is om de mogelijkheid uit te sluiten dat u niet naast een deur bent
  + Laten we nu aannemen dat de robot een meter voorruit beweegt
  + Markoc-localistatie neemt deze informatie op door de geloofsverdeling dienovereenkomstig te verschuiven zoals gevisualiseerd in het derde diagram
  + Om rekening te houden met inherent ruis in de robotbeweging, wat onvermijdelijk leidt tot informatie verlies, is het nieuwe geloof vloeiender (en minder zeker) dan de vorige
  + Laten we ten slotte aanemen dat de robot een tweede keer detecteert, en opnieuw bevindt hij zich naast een deur
  + Nu wordt deze waarneming vermenigvuldigd met de huidige (niet uniforme) overtuiging
  + Wat leidt tot de uiteindelijke overtuiging die getoont wordt in het laatste diagram
  + De waarschijnlijkheid is nu gebaseerd rond een enkele locatie
  + De robot is nu vrijzeker van zijn positie



**Kan 2 SLAM-applicaties opnoemen**

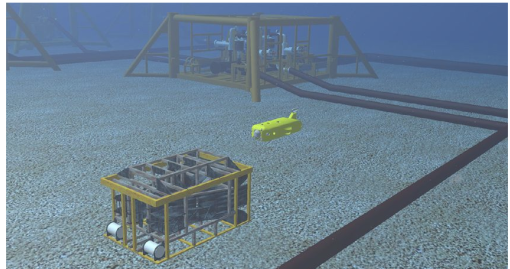
* Automatische pilootbesturing op niet-gerepeteerde of off-road terreinen
* Reddingstaken voor omgevingen met een hoog risico of moeilijk navigeren
* Planetaire, lucht-, terrestrische en oceanische verkenning
* Augmented reality-toepassingen waarbij virtuele objecten betrokken zijn bij scènes uit de echte wereld
* Visuele bewakingssystemen
* Geneeskunde

**Kan het verschil beschrijven tussen interoceptive en exteroceptive sensoren en hun link met relative of absolute position metingen**

* Interoceptive sensors
  + Zoals wheel odometers en IMU’s, genereren een relatieve positie
  + Ze zijn onderhevig aan niet-systematische fouten als gevolg van externe oorzaken zoals mensen interventie en systematische fouten vanwege onvolkomenheden in de structuur van de robot
* Exteroceptive sensors
  + Waaronder camera’s en lasers, leveren een absolute positiemeting
  + Als ze naast elkaar worden gebruikt, kunnen ze fouten zoals odometrische drift compenseren
  + De drie belangrijkste soorten sensoren die worden toegepast op de huidige SLAM-technologie zijn acoustic sensors, laserafstandsmeters en visuele sensors

**Kan uitleggen hoe acoustic sensors afstand meten; waar zij van toepassing zijn; en hun tekortkomingen**

* Hoe werken ze
  + Gebruikt de time of flight (TOF) -techniek om de locatiete meten
    - Bv een time of flight camera is een soort sonar techniek
    - De camera schiet een infrarood signaal uit, die vervolgens op objecten stuiteren en terugkeren naar de hardware
    - De tijd die dit licht nodig heeft om de camera te verlaten en er weer naar toe gaan, wordt gemeten
    - Waardoor een 3D kaart van een ruimte kan gemaakt worden
  + Sonarsensoren worden meestal onder water gebruikt, waarbij laserafstandsmeters en visuele sensoren zijn uitgesloten
  + Ultrasone sensoren zijn over het algemeen de goedkoopste beschikbare bron van ruimtelijke detectie voor mobiele robots
* Waar zijn ze van toepassing
  + Ze zijn compatibel met de meeste oppervlaktetypen, metaal of niet metaal, Doorzichtig of ondoorzichtig
  + Zolang de gemeten oppervlakte voldoende akoestische reflectie heeft
* Hun tekortkomingen
  + Lage ruimtelijke resolutie en detectiebereik
  + Gevoelig voor omgevingsfactoren en lage reactiesnelheden belemmeren het gebruik
  + Is de mogelijkheid om oppervlakte-eigenschappen te gebruiken om objecten te lokaliseren en te identificeren
  + Met kleuren- en grijswaardenafbeeldingen kunnen robots meer informatie gebruiken om functies in de omgeving te identificeren en te lokaliseren.



**Kan uitleggen hoe lasor range finders afstand meten; waar zij van toepassing zijn; en hun tekortkomingen**

* Hoe werken ze
  + Gebruikt TOF en phase shift technieken om de positite te meten
  + Dankzij de hoge snelheid en nauwkeurigheid van laserafstandmeters kunnen robots nauwkeurige afstandmetingen genereren
  + Laserscanner is de beste sensor voor het extraheren van planaire kenmerken zoals muren vanwege de verstrekte gegevens over het dichte bereik
* Tekortkomingen
  + De prijs is echter het struikelblok
  + Is de mogelijkheid om oppervlakte-eigenschappen te gebruiken om objecten te lokaliseren en te identificeren
  + Met kleuren- en grijswaardenafbeeldingen kunnen robots meer informatie gebruiken om functies in de omgeving te identificeren en te lokaliseren.

**Kan uitleggen hoe monocular camera’s afstand meten**

* De hardware voor te implementeren is veel eenvoudiger, wat leidt tot systemen die goedkoper en fysiek kleiner zijn
  + Wat maakt dat SLAM plots toegankelijk is op GSM’s
* De benodigde algoritmen en software zijn veel complexer vanwege het ontbreken van directe diepte informatie van een 2D afbeelding
* Omdat de diepten van punten niet rechtstreeks worde waargenomen, zijn het geschatte punt en de camera positie gerelateerd aan de reële posities door een gemeenschappelijke, onbekende schaalfactor
* De kaart wordt daarom een dimensie loze kaart zonder een echte betekenis verbonden aan een kaarteenheid

**Kan de unknown scale factor issue uitleggen alsook de dimensionless maps**

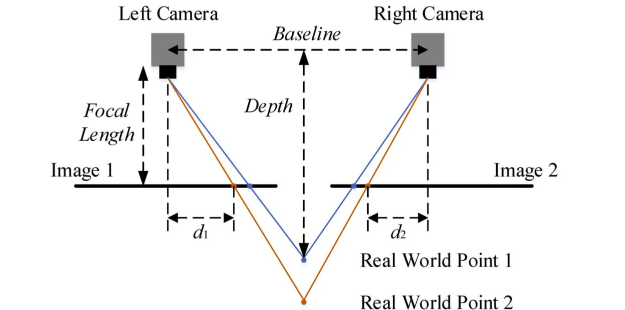
* Omdat de diepten van punten niet rechtstreeks worde waargenomen, zijn het geschatte punt en de camera positie gerelateerd aan de reële posities door een gemeenschappelijke, onbekende schaalfactor
* De kaart wordt daarom een dimensie loze kaart zonder een echte betekenis verbonden aan een kaarteenheid

**Kan een oplossing beschrijven voor het unknown scale factor probleem**

* Een Diepte camera toevoegen
* Metrische schaal introduceren door een externe schaalreferentie in de vorm van een vooraf gespecificeerd object of een set met een bekende grootte die kan worden herkend tijdens het in kaart brengen

**Kan in eigenwoorden beschrijven hoe stereocamera’s werken**

* Het dieptemeet bereik wordt beperkt door de basislijn en de resolutie
* In het algemeen geldt dat hoe breder de basislijn, hoe beter de diepteschatting
* Hoewel gebruiken van een bredere basislijn een grotere ruimte nodig heeft
* De basislijn op een AR-headset is meestal slechts 20 cm, en veel minder op een mobile gsm
* Gezien de hoge rekenbelasting, wordt FPGA de belangrijkste kracht om de hoge invoersnelheid te verwerken



**Kan de term FPGA uitleggen**

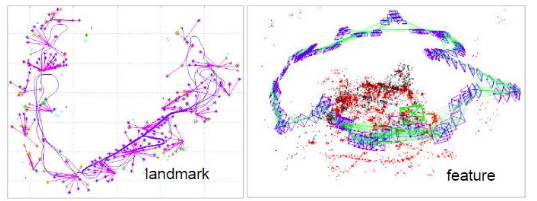
* Field programmable gate array
* Is een geïntegreerde schakeling die is ontworpen om te worden geconfigureerd door een klant of een ontwerper na productie

**Kan in eigenwoorden beschrijven hoe RGB-D-camera’s werken alsook het verschil tussen gestructureerd licht en time-of-flight en hun tekortkomingen**

* Genereren 3D afbeeldingen door structured light of tof
  + Die beide rechtstreeks diepte-informatie kunnen bieden.
* Structured
  + ToF-sensoren overspoelen de hele scène met licht en berekenen diepte tegen de tijd die nodig is om elk foton terug te brengen naar de sensor. Gestructureerd licht daarentegen houdt een bekend patroon in dat wordt geprojecteerd door de IF-sensor. De manier waarop het patroon vervormt, is de sleutel tot het construeren van een dieptekaart.
  + Berekent de deformatie van het pattern
* Tekortkomingen
  + De gestructureerde lichtsensoren bv kinetic voor Xbox zijn gevoelig voor verlichting
  + Dit beperkt hun toepasbaarheid in direkt zonlicht
  + RGB-D-camera's bieden geen betrouwbare bereikgegevens voor semi-transparant of zeer reflecterend oppervlakken, en hebben ook een beperk effectief bereik

**Kan het verschil tussen feature maps en occupancy**

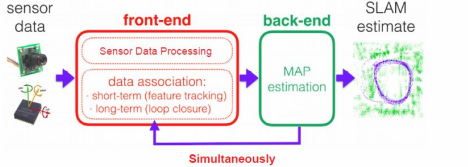
* Feature maps
  + Aangezien deze benadering een beperkt aantal schaarse objecten gebruikt om een kaart weer te geven
  + Kunnen de berekeningskosten relatief laag gehouden worden
  + Is een goede oplossing voor huidige applicaties
  + Nadeel
    - Gevoeligheid voor valse data associations



* Occupancy
  + Nuttig in pathplanning en exploratie algoritmen
  + Waarin de waarschijnlijkheidsinformatie de complexiteit van de path planning task kan verminderen
  + Nadeel
    - Zeer rekenintensief vooral voor grote omgevingen



**Kan een simpel architectuur van een SLAM-systeem tekenen ondergescheid in front-end en back-end**



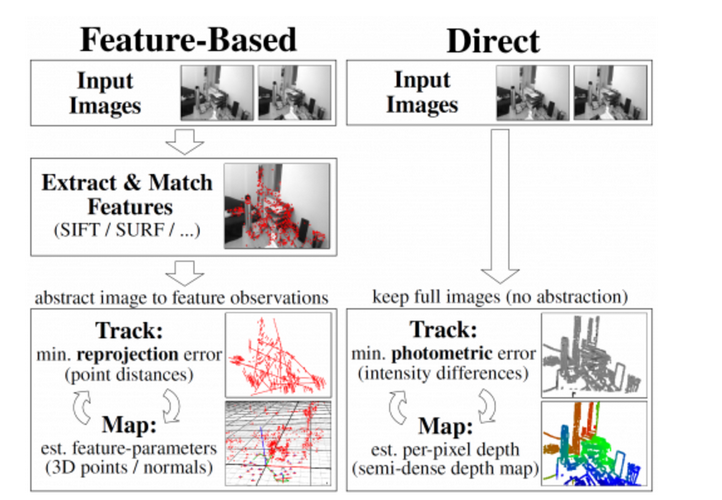
* Front-end
  + Abstracts sensorgegevens in modellen die vatbaar zijn voor schatting
* Back-end
  + Terwijl back-end interferentie uitvoert op de abtracted data geproduceerd door de front-end

**Kan het verschil tussen sparse en dense visual SLAM-methoden uitleggen**

* Sparse
  + Gebruiken slechts een kleine geselecteerde subsets van pixels in een beeldframe
* Dense
  + Gaan de meeste of alle pixels gebruiken in elke frame
  + Bieden meer details
  + Nadeel
    - Krachtigere hardware nodig

**Kan het verschil uitleggen tussen feature-based en direct visual SLAM-methoden**

* Feature based
  + Input image
  + Daar worden een set van sparse features uitgehaald
  + Match de features die krijgt vanuit verschillende poses
  + Los het SLAM-probleem op door de feature reprojection error
    - Het verschil tussen de bijgehouden locatie van een punt en waar verwacht wordt dat de camera een schatting van de pose geeft, over alle punten)
* Direct
  + Vergelijke hele afbeeldingen met elkaar door te zoeken welke delen samengaan
  + Ze kunnen semi-dense 3D-kaarten maken in realtime op een smartphone met behulp van semi-dense filteralgoritme
  + Dit betekent dat ze meer informatie over de omgeving bieden waardoor het interessanter is om te gebruiken in robotica of AR, en een betekenisvollere weergave geeft aan het menselijk oog
  + Nadelen
    - Kunnen niet goed omgaan met outliers, omdat ze altijd zullen proberen om ze te verwerken en in de definitieve kaart te willen gaan implementeren
    - Langzamer dan feature based methoden

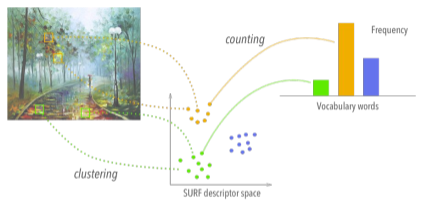


**Kan feature extraction en feature matching in de context van visual SLAM uitleggen**

* Feature extraction
  + Worden de meest bruikbare informatie uit de afbeelding gehaald
  + Features die van belang zijn variëren van eenvoudige punt features zoals hoeken tot meer uitgebreidere features zoals randen en zelfs complexe objecten zoals ramen, deuropeningen
  + Het gebied rond elk object wordt omgezet in een compacte descriptor die kan worden vergeleken met andere descriptors
  + De eenvoudigste descriptor van een feature is het uiterlijk of de intensiteit van de pixels in een patch rond het feature point
* Feature matching
  + Is het proces waarbij features afzonderlijk worden uitgehaald (descriptors) en over meerdere frames worden vergeleken
  + Is handig wanneer significante veranderingen in het uiterlijk van de features optreden na het observeren ervan over lange reeksen
  + De eenvoudigste manier om features tussen 2 afbeeldingen te matchen, is om alle feature descriptors in de eerste afbeelding te vergelijken met alle andere feature descriptors in de 2de afbeelding met behulp van een gelijkenismaat
  + De pose-schatting wordt berekend op basis van de overeenkomende features via een techniek genaamd RANdom SAmple consensus (RANSAC)
  + Nadeel
    - Een frame waarvan de meeste features in een klein gebied zijn geconcentreerd is voor het algoritme minder interessant, dan een frame met veel verspreide features over een groot gebied, omdat de features minder snel overlappen
    - Opslaan van de verwerkte features erg duur kan worden, aangezien deze methoden echter alle gegevens elimineert die niet kunnen worden gebruikt
  + Voordeel
    - Deze methode is wel sneller dan de directe methoden

**Kan loop closure in de context van SLAM uitleggen**

* Laatste verfijningsstap en is van vitaal belang voor het verkrijgen van een wereldwijd consistente SLAM-oplossing, met name bij het lokaliseren en in kaart brengen van gedurende lange perioden
* Is het proces van observeren van dezelfde scene door niet aangrenzende frames en het toevoegen van een beperking daartussen, waardoor de geaccumuleerde drift in de pose-schatting aanzienlijk wordt verminderd
* De meest basic vorm van loop closure detection is om het huidige frame aan alle voorgaande frames aan te passen met behukp van feature matching technieken
* Deze benadering is rekenkundig intensief vanwege het feit dat het aantal frames in de loop van de tijd dramatisch toeneemt, zodat het matchen van het huidige frame met alle voorgaande frames niet geschikt is voor real-time oplossingen
* Een oplossing is om key frames te definieren (een subset van alle voorgaande frames) en het huidige frame te vergelijken met alleen die key frame
* De meest gebruikelijke manier om kandidaten voor het loop closure frame te filteren, is om een plaatsherkenningsbenadering te gebuiken op basis van vocabulairestructuur waarin de feature descriptors van de kandidaat key frames hiërarchisch worden gekwantificeerd weergegeven door een back of visual words



* Feature descriptors zijn geclusterd rond de woorden in een visuele vocabulary
* De clustering beperkt het probleem tot een kwestie van tellen hoe vaak elk woord in de vocabulary voorkomy
* Ten slotte kan het beeld worden weergegeven met behulp van het resulterende histogram van frequenties
* Overeenkomsten van afbeeldingen worden vergeleken met histogrammen
* Problemen
  + False positives
    - Waarbij 2 verschillende plaatsen als hetzelfde gezien worden
  + False negatives
    - Waarbij 1 plzzts als 2 verschillende plaatsen wordt gezien
* Oplossing
  + Precisie-recall-curve kan de prestaties van het systeemn beter kwantificieren
  + De cirve markeert de wisselwerking tussen precisie (afwezigheid van false positives in de detectie) en recall (voorspellingskracht)

**Kan uitleggen waarom back-end optimalisatie nodig is, alsook het verschil tussen camera pose optimization en bundle adjustment**

* Omdat drift van de pose-schatting onvermijdelijk is wordt camera pose optimization cruciaal om de beweging van camera’s te achterhalen
* Traditioneel kan extended kalman filter (EKF) worden geïntroduceerd om de noise in de beweging (een schatting van de toekomstige positie van de robot) en observatie (de werkelijke meting) modellen te mnimaliseren
* Dit is nog steeds de eerste keuze voor kleinschalige schattingen vanwege de eenvaoud van de implementatie
* Een alternative methode os om bundle adjustment (graph optimization) te gebrijken
* Waarbij de camerapositie en de 3D-structuur parameters die worden bekeken en gekoppeld over meerdere frames gezamenlijk worden geoptimaliseerd door een kostenfunctie te minimaliseren
* Het trekt ideeën uit de kruising van numerieke methoden en grafentheorie.
* Bundle adjustment heeft steeds meer de voorkeur boven filteren, deels vanwege de inherente inconsistentie van de laatste
* In combinatie met submapping leidt deze methode tot een hogere efficiëntie.

# Multi Agent Systems, Swarms en ROS2

**Kan de term multi agent systemen uitleggen**

* Is een subdeel van distributed AI

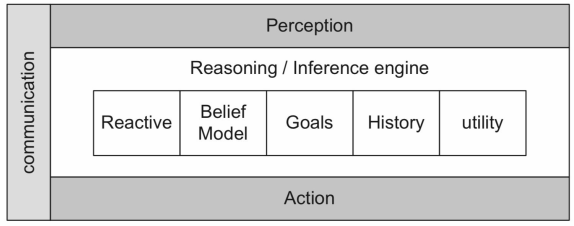
**Kan distributed AI; parallel AI, distributed problem solving en multi-agent-systems (MAS) uitleggen**

* Distributed AI
  + Subdeel van AI
  + Oplossen van real-world problemen
  + Parallel AI
    - Gedistribueerde hardware-architecturen zoals multiprocessor of clustergebaseerd computergebruik
    - Verhoog de snelheid van operation en werk op parallel threads om een globale oplossing te komen voor een bepaald probleem
  + Distributed problem solving
    - Overweegt hoe een probleem kan worden opgelost door de middelen en kennis te delen tussen een groot aantal samenwerkende modules, de Computing-entiteit.
    - Bij gedistribueerde probleemoplossing, communicatie tussen computerentiteiten, hoeveelheid gedeelde informatie wordt vooraf bepaald en embeddedin ontwerp van computerentiteit
    - Gedistribueerde probleemoplossing is star vanwege de embedded strategieën en biedt bijgevolg weinig of geen flexibiliteit
  + Multi-agent systemen
    - Omgaan met het gedrag van beschikbare computerentiteiten om een bepaald probleem op te lossen
    - Elke computerentiteit wordt een agent genoemd
    - Kan worden gedefinieerd als een netwerk van indviduele agents die kennis delen en met elkaar communiceren om een probleem op te lossen dat buiten het bereik van een enkele agent valt

**Kan een agent beschrijven in de context van MAS**

* Flexibele autonome entiteit die de omgeving kan waarnemen via de sensoren die erop zijn aangesloten
* Deze werken in op het environment via actuatoren
* Sitautedness
  + Gesitueerdheid
  + De interactie van een agent met de omgeving door het gebruik van sensoren en de resulterende acties van de actuatoren.
  + Alle inputs worden rechtstreeks ontvangen als gevolg van de interacties van agents met hun omgeving
  + Werk direkt op het environment via de actuatoren figureren niet alleen als een metaniveau-advisor
* Autonomy
  + Het vermogen van een agent om zijn acties onafhankelijk van buitenaf te kiezen zonder externe interventie van andere agents in het netwerk of menselijke inmenging
  + Dit kenmerk beschermt de interne statussen van agents tegen extern invloed
  + Het isoleert de agent tegen instabiliteit veroorzaakt door externe stoornissen
  + Andere agents kunnen invoer activeren, zie sociaal gedrag
* inferential capability
  + Het vermogen van een agent om te werken aan abstracte doelspecificaties, zoals het afleiden van een observatie door de informatie te generaliseren
* Responsiveness
  + Het vermogen om de toestand van de omgeving waar te nemen en er tijdig op te reageren om rekening te houden met eventuele veranderingen in de omgeving
  + Deze laatste eigenschap is van cruciaal belang in realtime-toepassingen
* Pro-activeness
  + Agents moeten een goede reactue op opportunistusch gedrag vertonen
  + Dit is om acties te verbeteren die doelgericht zijn in plaats van alleen te reageren op een specifieke verandering in de omgeving
  + Het moet de mogelijkheid hebben zich aan te passen aan veranderingen in de dynamische omgeving
* Social behaviour
  + In staat zijn om te communiceren met de externe bronnen wanneer de noodzaak zich voordoet om een specifiek doel te bereiken
  + Het moet ook deze keknnis kunnen delen en andere agents (MAS) helpen een specifiek probleem op te lossen
  + Dit wil zeggen dat de agents moeten kunnen leren van der ervaring van andere communicerende entiteiten die menselijk kunnen zijn, adnere agents in het netwerk of statische controllers

**Kan de typische building blocks van een autonome agent tekenen en beschrijven**



**Kan de link tussen agent technology en MAS uitleggen**

* Uitbreiding van de agent technology waarbij een groep losse verbonden autonome agents in een omgeving optreden om een gemeenschappelijk doel te bereiken
* Dit gebeurt door samen te werken of te concurreren, kennis te delen of niet met elkaar te delen

**Kan op zijn minst 2 voordelen van MAS geven**

* Een toename van de snelheid en efficiëntie van de operatie als gevolg van parallel berekeningen en asynchrone bewerkingen
* Een sierlijke degradatie van het systeem als een of meerdere agents falen. Het verhoogt daarmee de betrouwbaarheid en de robuustheid van het systeem
* Schaalbaarheid en flexibiliteit
  + Agents kunnen worden toegevoegd wanneer nodig
* Lagere kosten
  + Individuele agents zijn goedkoper dan gecentraliseerde architectuur
* Herbuikbaarheid-agents hebben een modulaire structuur en kunnen eenvoudig worden vervangen in andere systemen of gemakkelijker worden opgewaardeerd dan een monolitisch systeem

**kan ten minste twee kritieke MAS-uitdagingen vermelden**

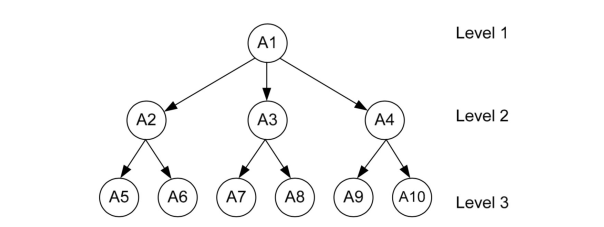
* Environment
  + De actie van een agent wijzigt niet alle zijn eigen omgeving maar ook die van de buren
  + Dit vereist dat elke agent de actie van de andere agents moet gaan voorspellen om de optimale actie te bepalen
  + Dit type van gelijktijdig leren kan leiden tot onstabiel gedrag en kan mogelijk chaos veroorzaken
  + Het probleem is verder gecompliceerd, als de omgeving dynamisch is
  + Vervolgens moet elke agent onderscheid maken tussen de effecten veroorzaakt door andere agents en variaties in de omgeving zelf
* Perceptie (zintuigelijke informatie)
  + Agents zijn verspreid over de hele omgeving
  + Elke agent heeft een beperkte detectiecapaciteit vanwege het beperkt bereik en de dekken van de sensoren die erop zijn aangesloten
  + Dit beperkt de beschikbare weergave voor elk van de agenten in de omgeving
  + Daarom kunnen beslissingen op basis van de gedeeltelijke waarnemingen door elk van de agents suboptimaal zijn en wordt het bereiken van een globale oplossing hierdoor onhandelbaar
* Abstractie
  + Er wordt vanuit gegaan dat een agent zijn hele action space en mapping van de state space tot actoion space kan gedaan worden door ervaring
  + In MAS ervaart elke agent niet alle states
  + Om een kaart te maken moet deze kunnen leren van de ervaringen van andere agents met vergelijkbare capaciteiten of beslissingsbevoegdheden
  + In het geval van concurrerende agent is het niet mogelijk om de informatie te delen, aangezien elk van de agents zijn eigen kans om te winnen probeert te vergroten
  + Het is daarom essentieel om te kwantificeren hoeveel van de lokale informatie en de mogelijkheden van andere agents bekend moeten zijn om een verbeterde modellering van de omgeving te creëren
* Conflictenoplossing
  + Conflicten komen voort uit het ontbreken van een globaal beeld dat beschikbaar is voor elk van de agents
  + Een actie die door een agent is geselecteerd om een specifieke interne status te wijzigen, kan slecht zijn voor een andere agents
  + Deze omstandigheden moet informatie over de beperking, actievoorkeuren en doelprioriteiten van agents worden gedeeld om de samenwerking te verbeteren
  + Een groot probleem is te weten wanneer deze informatie moet worden gecommuniceerd en aan welke van de agents
* Inference
  + Inferentie: een conclusie op basis van bewijs en redenering
  + Dit is moeilijk omdat de omgeving wordt gewijzigd door meerdere agents die al dan niet met elkaar communiceren
  + Verder kan het MAS bestaan uit heterogene agents, dat wil zeggen agenten met verschillende doelen en mogelijkheden
  + Deze kunnen niet samenwerken en met elkaar concurreren
  + Het identificeren van een geschikt afleidingsmechanisme in overeenstemming met de mogelijkheden van elke agent is cruciaal voor het bereiken van een wereldwijde optimale oplossing

**Kan het verschil uitleggen tussen homogeneous structure en heteregeneous structure in de context van MAS**

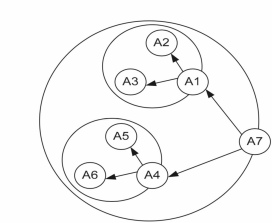
* Homegeneous structure
  + Alle agents van het multi-agent systeem hebben dezelfde interne architectuur
  + Interne architectuur refereert naar de local goals, sensor mogelijkheden, interne states, inference machanisme en mogelijke acties
  + Het verschil tussen de agents zijn hun specifieke locatie en het deel van de environment waar de actie wordt gedaan
* Heterogeneous structure
  + Agents verschillen in ability, structure en functionaliteit (predator-prey game)

**Kan het verschil uitleggen tussen hierarchical organization, holonic agent organization, coalitions teams in context of MAS en kan een figuur tekenen per type**

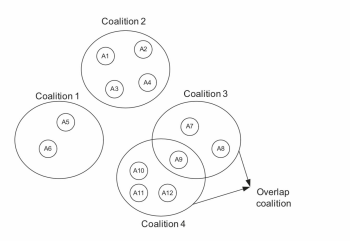
* Hierarchical organization



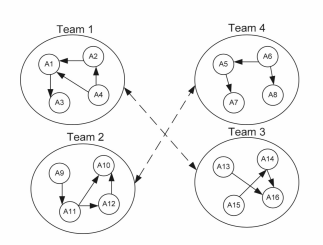
* Holonic organization
  + Een agent die wordt weergegeven als één entiteit, maar die kan bestaan uit vele subagents die aan elkaar zijn gebonden door verplichtingen



* Coalitions
  + Een groep van agents die samenkomen voor een korte tijd om de utility of de performance te verhogen van de individuele agents in een groep

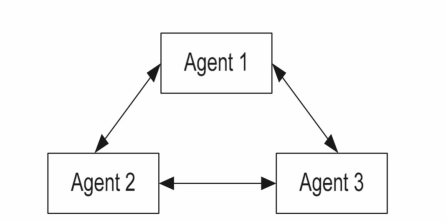


* Teams
  + Hetzelde als coalitie architectuur in design behalve dat de agents in een team samenwerken om de performance van de hele groep de verhogen

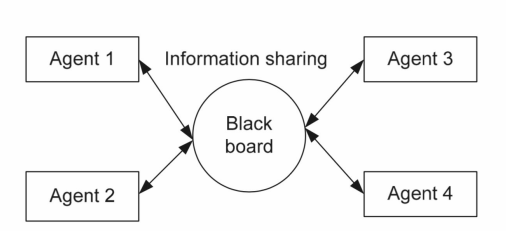


**Kan het verschil tussen local communication en blackboards in de context van MAS uitleggen door behulp van eigen figuren te maken om de uitleg te verduidelijken**

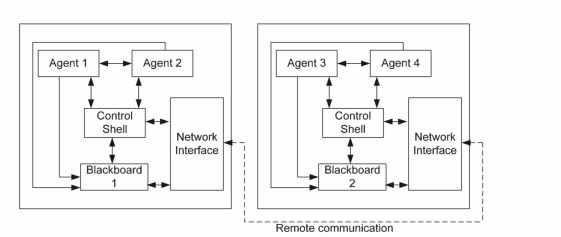
* Local communication
  + Lokale communicatie heeft geen plaats om de informatie op te slaan en er zijn geen tussenliggende communicatiemedia aanwezig om als facilitator op te treden
  + De term messaging passing wordt gebruikt om de directe communicatie tussen de agents te benadrukken.



* Blackboards
  + Een groep van agents delen een data repository die is voorzien voor het efficiënt opslaan en ophalen van gegevens die actief tussen de agenten worden gedeeld
  + De repository kan zowel de ontwerpgegevens als de besturingskennis bevatten waartoe de agenten toegang hebben



**Kan backboards in een multi coomputer setup uitleggen en kan een schets tekenen om het antwoord te verduidelijken**



**Kan een hierarchische state machine beschrijven**

**Kan active learning, reactive learning en learning based on consequence in context van MAS uitleggen**

* Active learning
  + Analyse van de observaties om een geloof of intern model van de omgeving van de overeenkomstige gesitueerde agent te creëren
* Reactive learning
  + Een geloof bijwerken zonder de werkelijke kennis te hebben van wat er moet worden geleerd of waargenomen
  + Deze methode is met name handig wanneer het onderliggende model van de agent of de omgeving niet duidelijk bekend is en als black box wordt aangeduid
  + Is te zien bij middelen die connectionistische systemen gebruiken, zoals neurale netwerken
* Learning based on consequence
  + Gebaseerd op de evalutie van hoe goed een geselecteerde actie is. Dit kan gebeuren door reinforcement learning

**Kan de term swarms beschrijven**

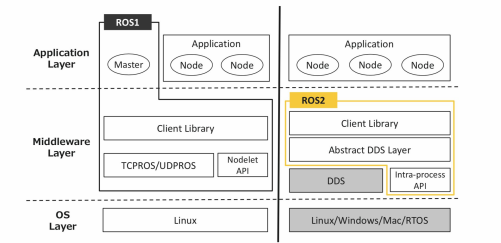
* Zijn ontworpen naar mieren, bijen en dergelijke wezens
* Het idee gaat daarom niet alleen over meerdere robots die samenwerken maar over veel robots die heel eenvoudig zijn
* Je zou verwachten dat elke robot te dom is om mogelijk zelfs te weten waarom hij iets doet
* Hun eenvoudige functie zou echter resulteren in een complex algoritme wanneer een aantal robots dit doen
* Met swarms kan elke robot niets zinvols doen. Het gewenste gedrag komt pas naar voren als er veel zijn
* Meestal zien robots er allemaal hetzelfde uit

**Kan een simpele representatie van zowel ROS1 als ROS2 tekenen om de verschillen duidelijk te maken**

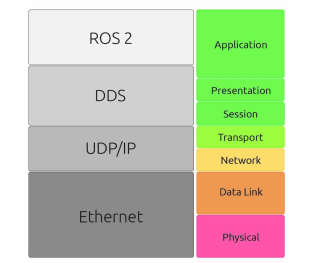
**Kan het verschil tussen ROS1 en ROS2 architectuur uitleggen**

* ROS1
  + Heeft een master als die master uitvalt valt heel het systeem uit
  + Client library
  + UDPROS
  + TCPROS
  + Nodelet api
  + Alleen Linux
* ROS2
  + Heeft geen master elke node werkt afzonderlijk zonder master
  + Wat maakt dat interne communicatie iets moeilijker is
  + Abstracted DDS layer
  + DDS
    - Data distribution service
  + Intraprocess api
  + Linux, windows, mac, RTOS

**Kan een figuur tekenen met de ROS2-, DDS-, UDP / IP- en Ethernet-stack en een OSI 7-laags model**



**Kan de voordelen van DDS uitleggen met behulp van OSI 7 layer model**



**Kan de ros\_bridge uitleggen om ROS1 te connecteren met ROS2**

