If you retake the oral you have to retake the written test.
You can try what you have done on a real robo using matlab under appointment.
- 80% - written
- ± 20% - oral
- 10% - assignments

TODO Matlab

TODO zoleueb xrobotics 7 & every classes

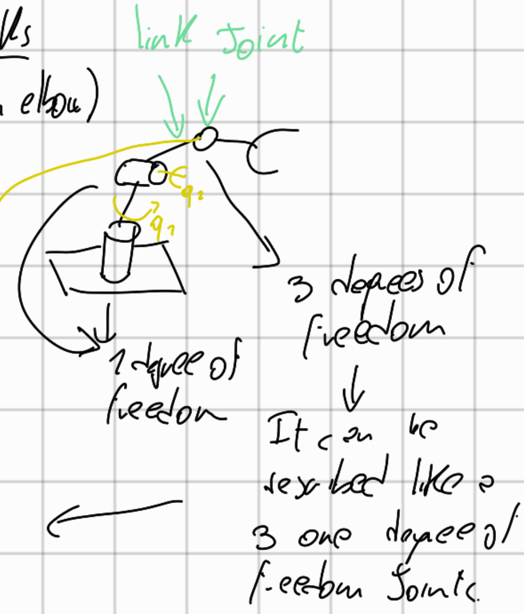
Monday 10:15

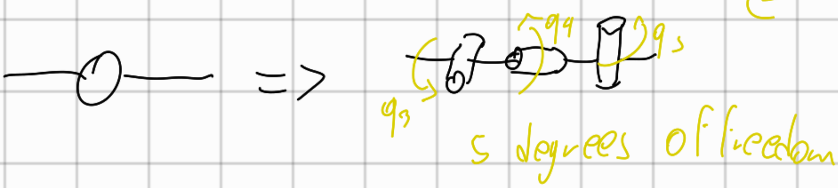
Introduction

physics of a manipulator

A manipulator is a connection between joints and links
A link is a 1 degree of freedom mechanism. (Like an elbow)

Degree of freedom is the number of variables needed to describe the movement of an object in space





The joints contributes to the movement of links and joints after it. Where the actuators are contributing to how the robot moves. You also have to take into consideration the physical stess on the robot.

It does not matter if you actuate a joint or not. If you have a joint then that contributes to adding another degree of freedom.

To have the position of arm of the robot you need to know a vector that describes the position of the arm.

$$\underline{q} = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_n \end{bmatrix}$$

→ This is called configuration vector or joint position vector

It describes the position of the joints

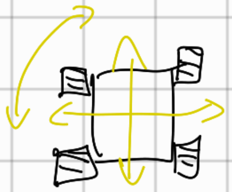
underline indicates a vector

n → number of degrees of freedom

$q \in \mathbb{R}^n$ $\quad$ n - dof system.

dof → degrees of freedom

The base of the robot has 3 degrees of freedom



6 degrees of freedom is the needed number to completely describe an object in space because we need 3 to describe rotation and 3 to describe translation.

If we increase to 7 for example then the degrees of freedom become redundant

For example our arm is a 7 degree of freedom system, so we have a redundant arm. We can show that by fixing the arm in a position and we can still move the elbow.

So 6 is the minimum to reach the end effector. wich is the final point to "touch" with your arm.

The end effector should be the tool at the tip of the arm and not the final point to touch

TODO

All of this also generates the WORKSPACE OF THE Robot.

It can be defined as a set of point that my manipulator can reach. with any orientation or in general.

We are also interested to know the dynamics of the robot. We need to know the velocity of every joints

$$\dot{\underline{q}} = \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \vdots \\ \dot{q}_n \end{bmatrix} \rightarrow \text{Joint velocity vector} \qquad \dot{\underline{q}} \in \mathbb{R}^n$$

We can also have the joints acceleration vectors.

$$\ddot{\underline{q}} = \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \\ \vdots \\ \ddot{q}_n \end{bmatrix} \rightarrow \text{Joint acceleration vector} \qquad \ddot{\underline{q}} \in \mathbb{R}^n$$

We stop at the second derivative because the generation of acceleration is generated by the torque of the motor. So the velocity is generated by the acceleration of the joint and this is generated by the torque of the motor.

There are also cartesian robots wich translates in space instead of rotating.

As a side note the motor does not need to be embedded in the joint. It can happen that the joint can be tendon driven. Also some motors can have embedded controllers and everything is in a joint, and then a bus to deliver information.

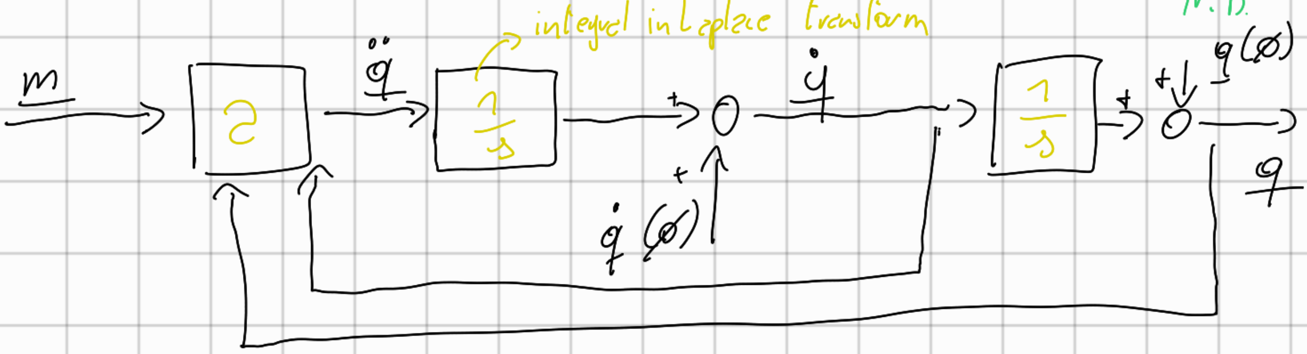
$$\underline{m} = \begin{bmatrix} m_1 \\ m_2 \\ \vdots \\ m_n \end{bmatrix} \rightarrow \text{Joint torque vector}$$

From the torque $(\underline{m})$ to the acceleration $(\ddot{\underline{q}})$ we have the dynamics. From velocity $(\dot{\underline{q}})$ to position $(\underline{q})$ we have an integral. One example of dynamics is the moment of inertia.

Dynamic are not covered in this class.

Dynamics of the manipulator.

To apply laplace integral we need to know the initial condition. N.B.



In block a we have the equations describing the dynamics of the system.

Block a corresponds to:

$$\boxed{A(q)\,\ddot{q} + B(q,\ddot{q})\,\dot{q} + C(q) = m}$$

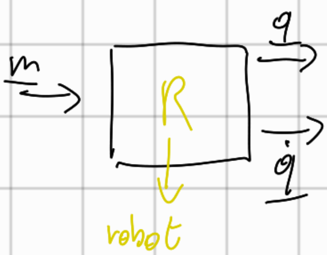
this should be m + external forces

inertia metrics

This is the basic equation for Robotics systems.

A represents inertia effects
B represents coriolis effects
C represents gravitational effects

Most of the times the previous equation can be simplified:



we will put into a grey box the dynamics of the robot.

Controlling the Robot

We will use a backstepping approach.

It's a controlling technique with starting from an input m adds through various layers other variables. We go backwards from the physics and we backstep towards the desired variables.

The main problem gets subdivided

I take m and add a block that regulates acceleration well. Once I have that I start designing what regulates speed. ~~TODO~~ 12:33

The first layer is:

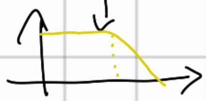
Dynamic control layer (DCL)

The objective of this layer is to make the robot controllable with joint velocity reference signals.

This layer can be done in a model based approach that requires knowing the inertia of the system or non-model based system, for example PID controller.

It's all a matter of bandwidth of a system. TODO WTF is this

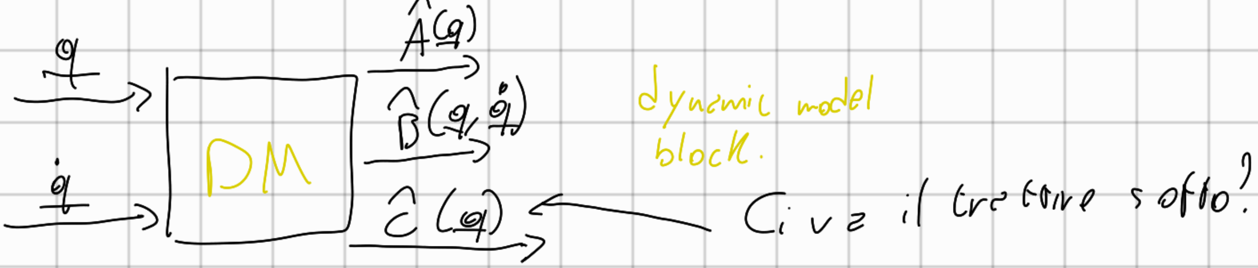
If you give a signal wich is outside of the bandwidth of a system this will be filtered out.



For example if you oscillate a robotic arm outside the bandwidth of the system then the Amplitude will be lowered.

First of all to build a model based system you need all the parameters that affect the inertia of the system. (geometric and physical parameters). These parameters are estimated because you have some uncertainties in the building of the arm. (system)

The first model block wich we can call dynamics model block is:



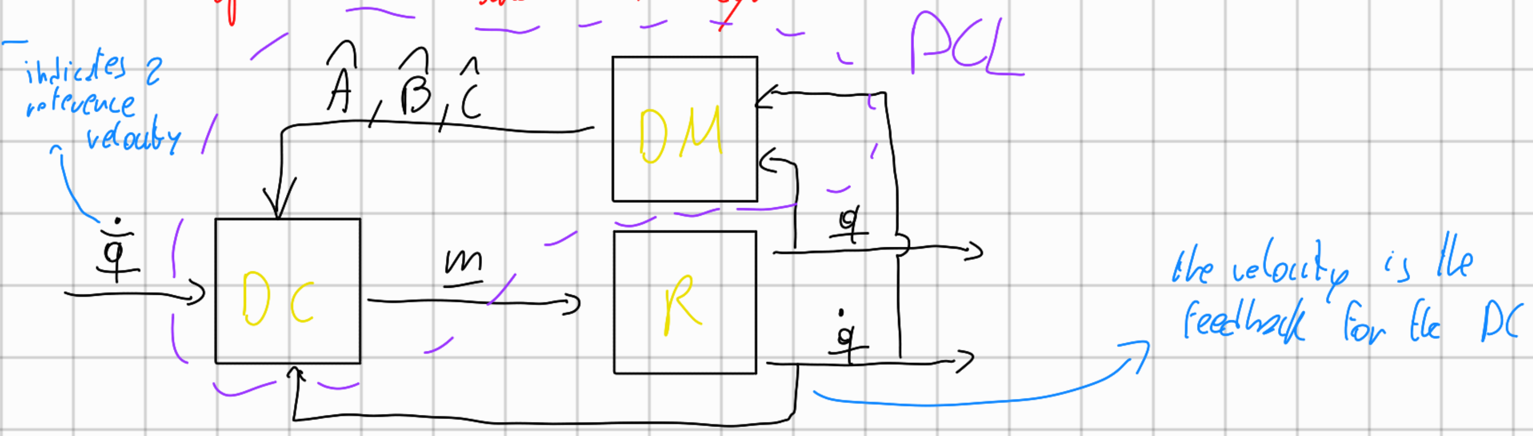
$\hat{A}$, $\hat{B}$ and $\hat{C}$ are computed in the simulation to help with projecting.

The ABC model is an exact model in some cases and approximated one in others like the underwater robot.

22/04/23

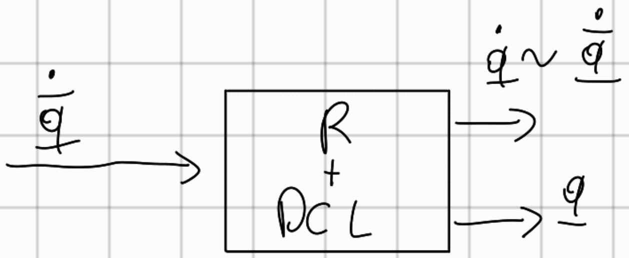
DM allows to have an estimate of the parameters related to the dynamic model equation. Whenever you have a model, in practice you also need parameters. The approx. can be more or less precise. Once you have that you can build a controller.

Closed loop control scheme for dynamics



The overall goal is to regulate the velocity of the model
DC needs to compute the error between the reference velocity and the actual velocity.
This istance of PCL exploits the model between $\hat{A}, \hat{B}, \hat{C}$.
This is only a possibility. An other one could be to attach a PID to every joint.

The goal is to have $\dot{\underline{q}} = \dot{\underline{q}}$. Whenever this level of detail is not important we can omit the PCL in detail and just represent:



If we simulate the complete scheme we have the dynamics of the system and given velocity in the bandwidth of the system velocity follows very closely. If we do a kinematic simulation we assume that the input immediately goes to the output so second scheme.

Now that we have a sufficient DCL now the first step of the backstepping approach is complete. First we put a subproblem with putting a reference velocity $\dot{\bar{q}}$. Once this is solved we need to find the velocity needed to reach an object. It's harder to immediately parametrize so we isolate dynamics and approach the problem only with velocities. You must not forget that dynamics exist so there is bandwidth. We are not able to accelerate extremely fast. We need to keep into consideration how fast $\dot{q}$ changes. Another thing is saturation of actuators that limit the speed (non-linear thing).

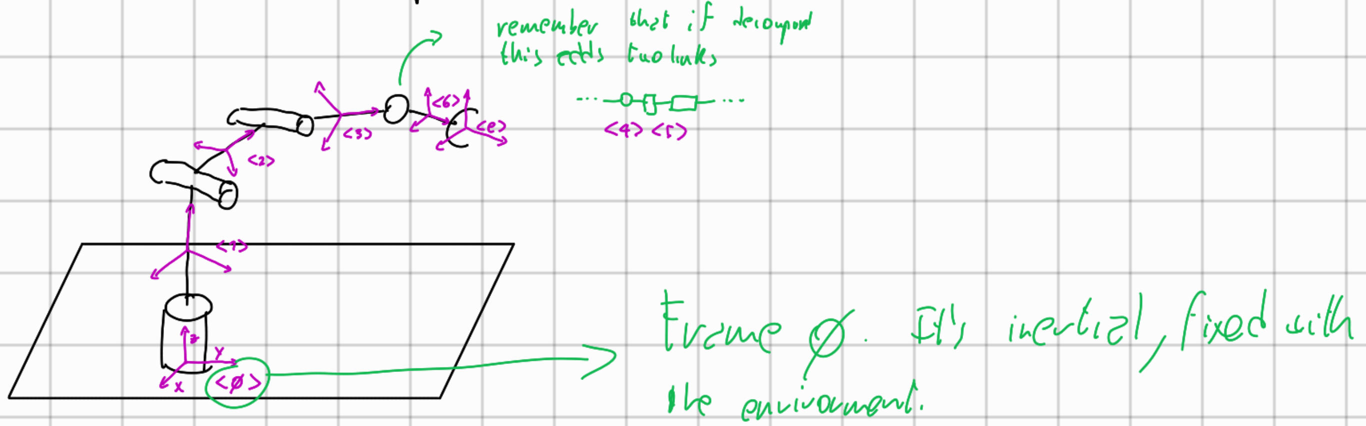
Kinematic Control Layer (KCL)

Is the input the goal position?

This layer can be very complicated. This layer needs to achieve the desired velocity for a joint.

The goal of this layer is to reach a $\overbrace{\text{desired position and orientation}}^{\text{pose}}$ in the Cartesian space with either the end effector or tool.

The first thing that we need is to know where the end effector is. And for that we need the geometry of the robot and the configuration of the robot (time dependent). Then also the end position of the E.E.



If I know position and value of the joint I will know the EF. This is done iteratively.

When we talk about position of a joint we introduce the concept of Frame

Placed the frame $\emptyset$ then we start placing frames on every link.

Note that <e> is placed where the manipulator closes so that is usefull in gripping.

Now I want to know how frame <2> is placed in respect to frame <0> or frame <e> is placed in respect to <0>. This is represented by a 4x4 matrix called transformation matrix.

with respect to $\rightarrow$ $^{\emptyset}_{e}T(q)$ or $^{\emptyset}_{e}H(q)$

↳ homogeneous

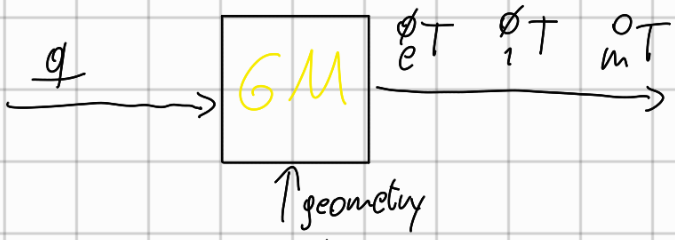
How e is placed according to $\emptyset$.

With the knowledge of q we can build all the transformation matrixes with respect to any other frames. In particular we want to compute every frame position in respect to frame $\emptyset$ (<$\emptyset$>).

q allows me to compute $^{\emptyset}_{e}T$, $^{\emptyset}_{1}T$, $^{\emptyset}_{2}T$, $^{\emptyset}_{3}T$

Geometric Model

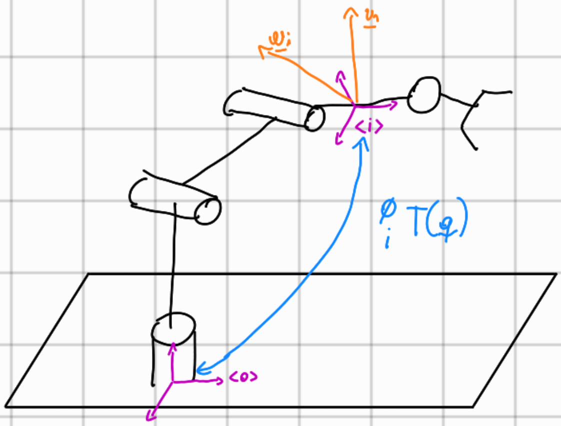
As an output gives the various T



We can build a library in which parameters of the geometry of the robot are given and we can write the Transformation Matrixes

We need to make the T.M. of the post exactly equal to the TM of the E.E. To do that we need to change $\dot{q}$. To change q we need to operate on $\dot{q}$.

If we need to move ee we should think about the velocity of the frame of the ee.
We want to reach a certain position so we need to learn how the various frames move.



Two 3 dimensional vectors are needed to describe the movement of an object in space: - velocity vector
- angular velocity vector.

The combination of these two vectors is $\underline{\dot{X}}_{i/0}$

$$\underline{\dot{X}}_{i/\phi} = \begin{bmatrix} \underline{\omega}_{i/\phi} \\ \underline{v}_{i/\phi} \end{bmatrix} \in \mathbb{R}^6$$ geometric vector N.B.

This is the velocity and ang.vel. of the frame $\overset{<i>}{i}$ in respect to an observer placed in frame $\underset{<\phi>}{\phi}$.

$${}^{\phi}\underline{\dot{X}}_{i/\phi} = \begin{bmatrix} {}^{\phi}\underline{\omega}_{i/\phi} \\ {}^{\phi}v_{i/\phi} \end{bmatrix}$$ algebraic vector N.B.

this means to wich reference point everything is calculated. TODO
riguardo diff of ϕ and $_{i/\phi}$

To better understand we have a geometric relationship:

$$v_{obs/boat} + v_{drone/obs} = v_{obs/drone}$$ ← These are only vectors

Now to do the calculation I need a reference plane on with doing the calculations.

$$v^{\phi}_{obs/boat} + v^{\phi}_{drone/obs} = v^{\phi}_{obs/drone}$$ ← these are vectors of numbers used in computers.

Now we need to know the relationship between $\dot{x}$ and the movement of joints and that is a jacobian relationship between the velocity of joints and position of joints

linear

$$\underline{\dot{x}}_{i/\phi} = J_{i/\phi}(q)\,\underline{\dot{q}} \qquad J_{i/0} \in \mathbb{R}^{6 \times n}$$

6: 3 trans $+$ 3 rot

n: number of frames

The motion of a frame depends only on the motion of links previous to that particular link. So usually J has ϕ in the columns representing the last joints. You could use a smaller matrix at that point but for computers is practically irrelevant.