AMR 1/10/2024

The key problems of mobile robotics

1. where am I?
2. how am I supposed to get to the goal?
3. how do I actually move?

1: localization (with or without initial guess, map,...)

2: path/trajectory/motion planning (respectively: only geometric motion, with time, among obstacles)

3: motion control (feedback techniques)

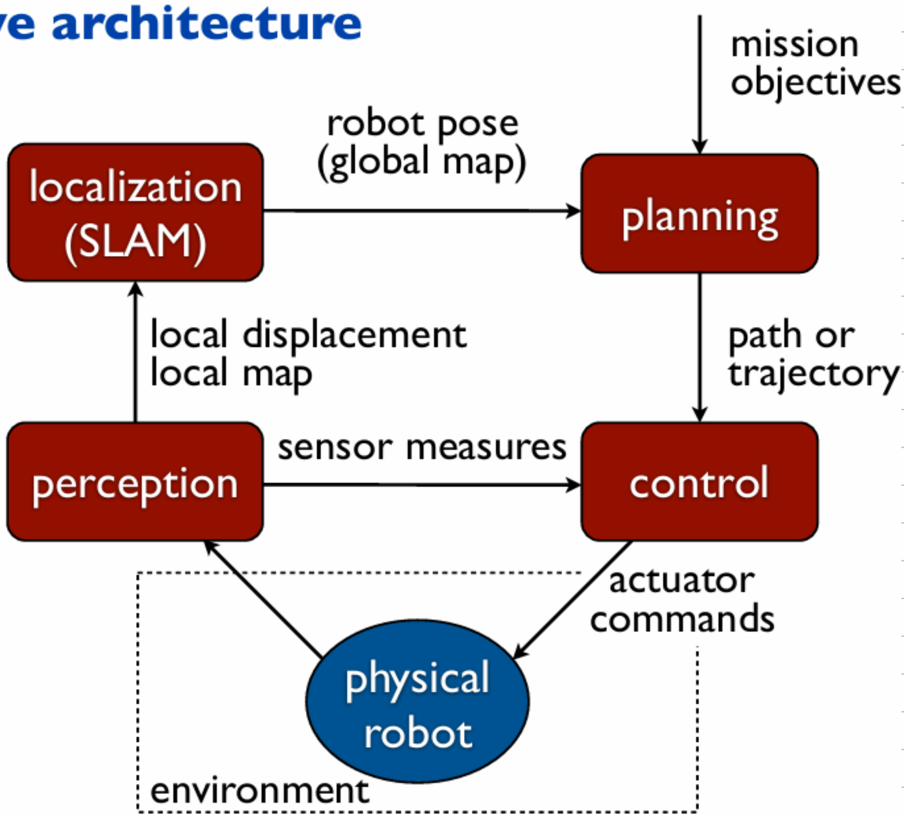
	fixed-base manipulators	single-body wheeled mobile robots
1. localization	easy (thanks to fixed-base and joint encoders)	difficult
2a. path/trajectory planning	easy (all paths are feasible)	difficult (not all paths are feasible)
2b. motion planning	difficult (many dof's)	more difficult (not all paths are feasible)
3. motion control	difficult (due to inertial couplings)	more difficult (nonlinear & no smooth stabilizer)

We need perception = sensing + interpretation. So actually there are 4 topics, localization, planning, motion control and perception htat is orthogonal to these ones.

deliberative architecture

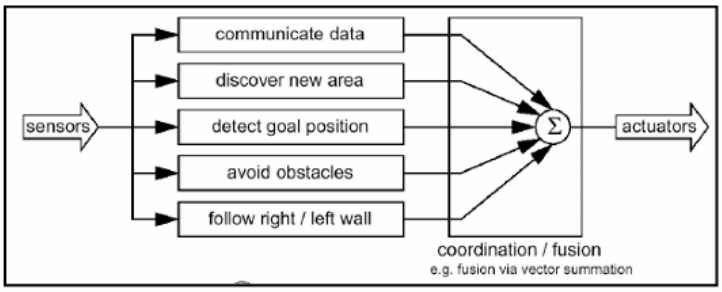
The simplest architecture

"think, then act"



other architectures

- reactive architecture ("don't think, (re)act")
- hybrid architecture ("think and act concurrently")
- behavior-based architecture ("think the way you act"), e.g.



taken from "Introduction to Autonomous Mobile Robots"

Configuration Space

Definition

Examples

Topology of the Configuration Space

Distance

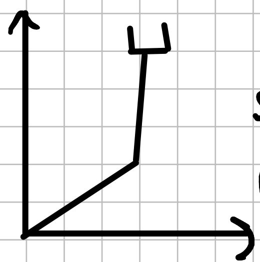
DEFINITION:

Def. of a ROBOT: a system of N bodies moving in a workspace $W = \mathbb{R}^2$ OR $\mathbb{R}^3$

$\mathbb{R}^2 \downarrow$ PLANAR ROBOTS

$\mathbb{R}^3 \downarrow$ SPATIAL ROBOTS

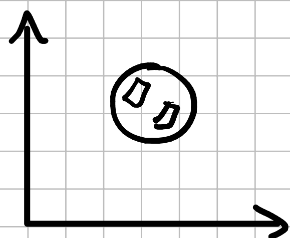
In $\mathbb{R}^2$



N=2 BODIES

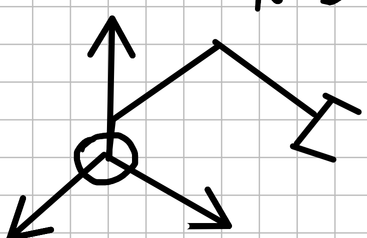
SO A PLANAR MULTI-BODY ROBOT

Roomba lives in the planar space and has just one body N=1

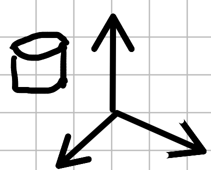


In $\mathbb{R}^3$

N=3



A quadrotor is a spatial single body robot



Then we can take the manipulator and put it on top of a roomba, for instance.

Definition of Configuration: a MINIMAL set of parameters which allow to place the robot in the workspace W

Place means to identify the posture, namely the position of all its points.
I can tell which part of the workspace is occupied by the robot

I can organize these paramers in a vector $q = \begin{pmatrix} q_1 \\ q_2 \\ \vdots \\ q_n \end{pmatrix}$ $n =$ minimum number of parameters

The q_i is called generalized coordinate. Why generalized? Because they are of two kinds: cartesian, to identify the position of points of the robot; angular coordinates, used to describe the orientation of bodies of the robot

Cartesian are in R_2 or R_3

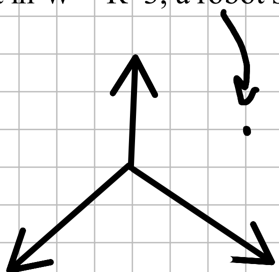
Angular coordinates are in the Special Orthonormal Group (SO2 or SO3). So basically rotation matrices.
This representation however is not minimal! I am using 4 elements in R^2 to represent a rotation, in R^3 for three rotations (3 axis) I am using 9 parameters. We want independent variables.

Euler angles are a way of describe orientation in a minimal way, just 1 angle in R^2 and 3 angles in R^3. Problems of Euler angles: they have singularities. To have a singularities-free representation I need more parameters, I need 4 parameters, so quaternions.

Configuration Space C: the set of all possible values of the configuration vector q. Since it's a space it has a dimension, its dimension is the number of parameters used.

EXAMPLES

A point robot in W = R^3, a robot so small that can be considered as a point

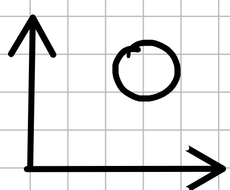


which paramers I need to describe its posture?

3 CARTESIAN COORD: $q = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \in \mathbb{R}^3$

The Configuration space is the same as the Workspace

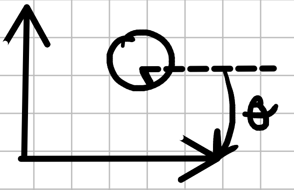
A disk robot in the plane, like Roomba



Consider the position of the center (x,y) we can place the disk and we know where the robot is. We may say that $q = \begin{pmatrix} x \\ y \end{pmatrix} \in \mathbb{R}^2$

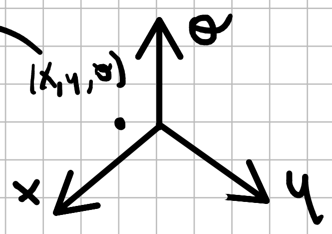
The configuration space is R^2 as the workspace W

Let's now add a tick to the disk robot, a specific orientation, useful for a camera for instance. If you want to know the field of vision you need to know where the tick is



Now besides the (x,y) coordinates of the center, we need to describe the orientation. We can choose the angle of the tick with respect to the horizontal axis θ

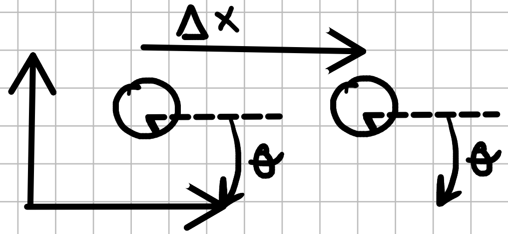
The configuration space here is the composition of three coordinates, the first two in R^2 and the second one in SO2, so the configuration space is the cartesian product of these two spaces $C = \mathbb{R}^2 \times SO2$



This above is a naive representation, not the best one. The entire robot shrinks to a single point in this configuration

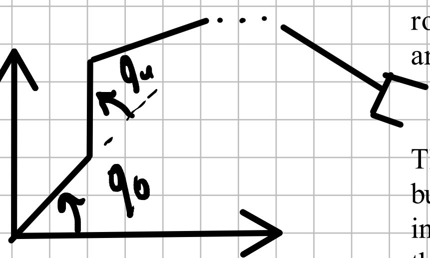
If I change one of these three coordinates, let's say I change x, adding a displacement Δx

So $x' = x + \Delta x$



Imagine a big humanoid, and being able to shrinking it to a single point in a suitable configuration space, that's powerful. If I have to plan the motion, I have to take into account all the bodies in the workspace, while in the configuration space the robot becomes a single point, and the point must avoid the obstacle, so much easier to develop algorithm in this space.

PLANAR MANIPULATOR



We have N joints, the workspace is R^2. How do we define the configuration of this robot? One way: for each body, let's take the body center and orientation of the link around the barycenter -> 3*N parameters. Is this minimal?

The problem is that this would be a good representation if the link were separated, but we have joints here, so links have relationship between them, or better, they introduce contraints. What kind of constraints? Each joint introduces two constraints: the x the y must be aligned (???), the number of free parameters is 3*N - 2*N = N

We have subtracted the number of constraints for the whole robot.

I would use joint variables, that are N, to represent the configuration.

$$q = \begin{pmatrix} q_0 \\ \vdots \\ q_n \end{pmatrix}$$

So in a manipulator, the generalized coordinates are the joint variables. What is the dimension of this space? Each rotation is in the plane so 1 parameters so we have C = SO(2) x ... x SO(2) = (SO(2))^N

$n = N$ → dimension of the conf. space

Let's move to R^3. We will have 6 parameters for each body taking position and orientation of each body. Is this 6*N minimal? No, of course, since joints introduces contraints. How many constraints does an elementary joint introduce?

It has just 1 DoF, since you can rotate only in one direction, we can just move one axis. The number of constraints is 5 out of the 6 parameters. So we have 6*N - 5*N = N as in the plane. We use joint variables.

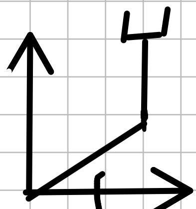
The configuration space C = (SO(2))^N

Each rotation is still a planar rotation, so still SO(2), we are still rotating in the plane, since the other two rotations are fixed, as we said.

Topology

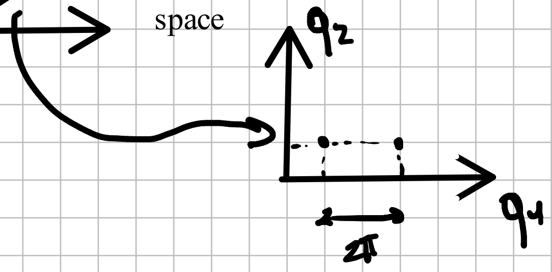
property of a space which are invarint under continous deformation, so it's not the shape but it's the nature of the space

In general, is not an euclidean space, since it's includes SO(2) and SO(3). In an euclidean space you have a vector, you multiply it and get another vector, so it's close w.r.t sum and moltiplication with scalar. You can't do that with angular coordinates, you can't multiply an orientation with a scalar, so it's not euclidean (in general, e.g. the point robot is in an euclidean space)



C = SO(2) x SO(2) and generalized coordinates are the joint variables, n = N = 2

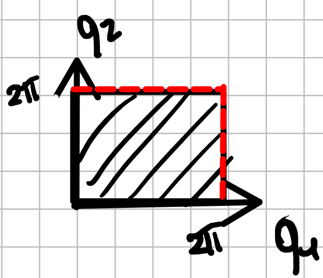
I want a representation for this. Let's start with the naive representation of the configuration space



Problem: injectivity. Let's perform a complete rotation with the first joint of the robot. The robot goes back to the same posture. But in the configuration space we are in another point. So we have multiple representation for the same robot posture.
It's not easy to work with these situations.

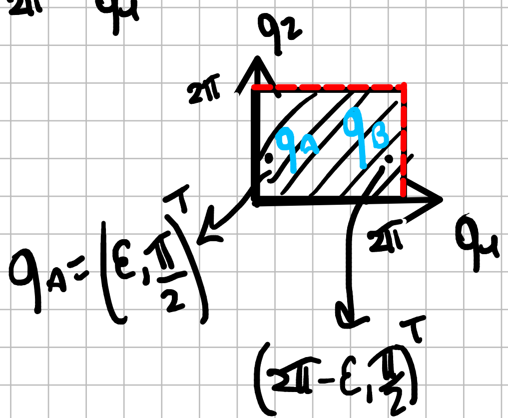
How to get rid of this problem? Let's take a square of ...(???) 2pi

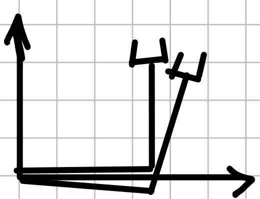
We just consider 0 to 2pi interval



I take one side only, since otherwise I will have the same problem of injectivity

Does this represent distances correct? To discuss it let's consider the configuration very close to the left and right boundaries, q_A and q_B

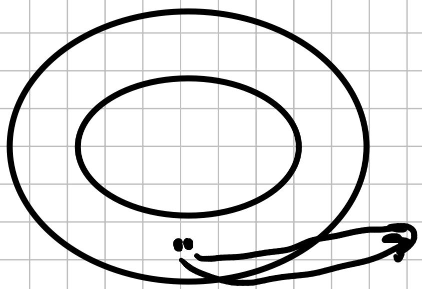




q_A and q_B are far in the configuration space but close in posture. The more I push them apart in the configuration space the more they become the same

Distance is not well represented in this way

We need to go to 3 dimensions to solve this problem. We need to take the square in 2D and tell the representation that the right and left boundaries are the same. In the same way, we need to tell that the upper and lower boundaries are the same, folding the square and getting a donut.



This object is a torus in geometry. It's not an euclidean space, it's a manifold (varietà in italian) in differential geometry.

q_A AND q_B ARE NOW CLOSER

In a manifold, a neighborood of a point looks like the regular neighborood in the euclidean space, so there is a invertible mapping of the two spaces, an homeomorphism. Locally the taurus looks like an euclidean space.
Basically if you see from very close the curvature is not visible, it seems a plain euclidean space, the global structure is completely different.

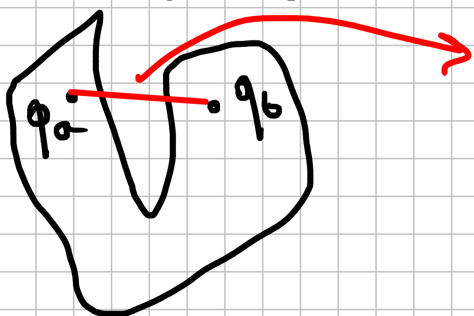
3/10/2024

Distance

How we define it in the configuration space. Is needed to control and planning, since it quantifies how far a configuration is from another one. Mathematically, a function d(q_a, q_b) must sasfisfy this axioms
- iff is positive,
- iff = 0 -> q_a = q_b
- iff d(q_a, q_b) = d(q_b, q_a) symmetric
- iff d(q_a, q_b) <= d(q_a, q_c) + d(q_c, q_b) triangle inequality

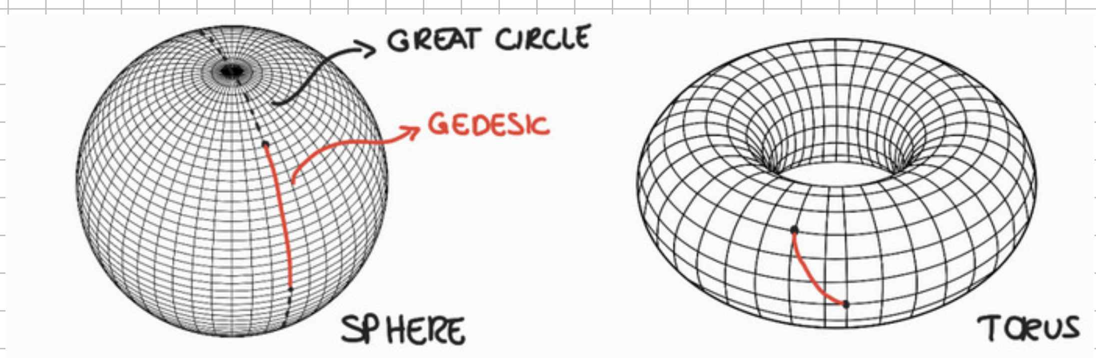
in order to be called a distance

Can we use the euclidean space in the configuration space?
d(q_a, q_b) = ‖q_a - q_b‖
No, since the configuration space is not an euclidean space



IT'S NOT ON THE CONFIGURATION SPACE!

We use geodesics in manifolds. It's a minimum lenght curve on a surface. The lenght of the geodesic will be the distance



In the sphere, the geodesic of two points lies on the great circle (i.e. the circle with the biggest radius that passes through the two points). In general, is difficult to find the geodesic. We only know geodesic for a limited set of simple manifolds. This is not practical

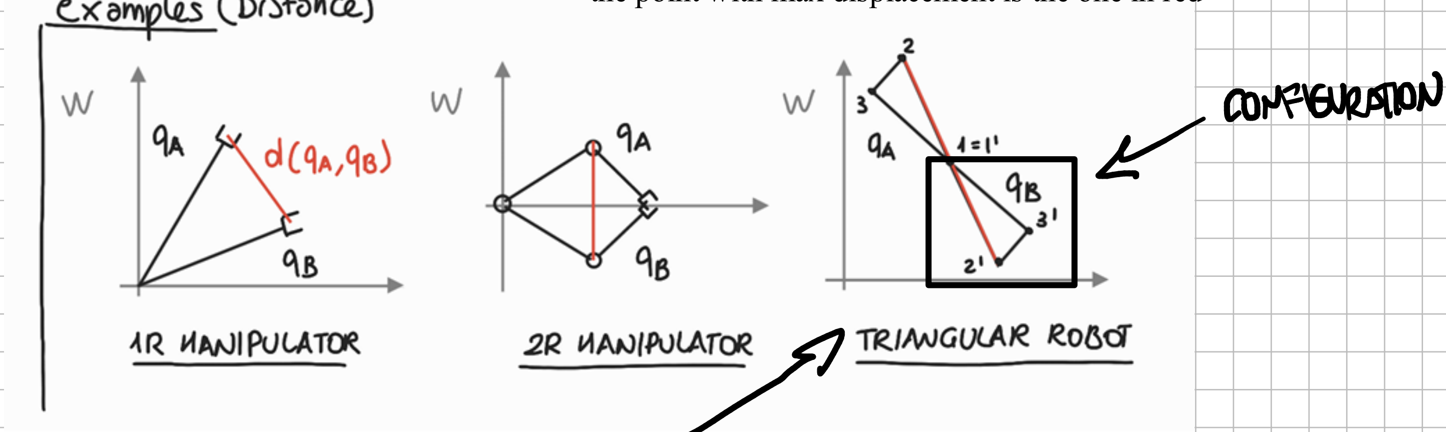
In robotics, we use the displacement metric

- p a point of the robot
- p(q) is the position of point p in W when the robot is in q

$$\boxed{d(q_A, q_B) = \max_{p \in robot} \| p(q_A) - p(q_B) \|}$$

This is the euclidean space in the workspace, not in the configuration space, so we can take it, since the workspace is an euclidean space. We are looking at the same point p, taking its position in configuration q_a and q_b.
Let's do an example

the point with max displacement is the one in red



This works also with mobile robots

It's complicated to compute, you need to compute the maximum over an infinite set. In general robots are not so simple as above.

We also need to compute the gradient of the distance, because we want to move away from a obstacle or closer to the goal. This distance is non-differentiable, since we have a maximum so there is a discontinuity

Practical alternative:

let's get rid of the maximum over the infinite set of points and compute the maximum over a finite set of points called control points.

- Define N control points p_1, ..., p_n, pre-defined. For instance barycenters of the links, end-effector etc.
- d(q_a, q_b) is replaced with

$$d(q_a, q_b) = \sum_{i=1}^{N} \| p(q_a) - p(q_b) \|$$

get rid of the
discontinuity using sum instead of max. Moreover, it's a finite set of points.

If I make stupid choise for the control points, the distance will be bad. The control points must be properly selected, tipically distributing them along the whole robot.

Sometimes we will make an even stronger simplification, using an euclidean distance:

$$d(q_a, q_b) = \| q_a - q_b \|$$

If we are local, so q_a and q_b are in the same locality of the configuration space, it's okay.