

Reasoning robot, involved tasks/modules and robot world representations

Václav Hlaváč

Czech Technical University in Prague

Czech Institute of Informatics, Robotics and Cybernetics

160 00 Prague 6, Jugoslávských partyzánů 1580/3, Czech Republic

<http://people.ciirc.cvut.cz/hlavac>, vaclav.hlavac@cvut.cz

Outline of the talk:

- ◆ Artificial intelligence view.
- ◆ Symbol grounding.
- ◆ Applied AI approaches.
- ◆ Need for a robot control architecture.
- ◆ Path planning / trajectory generation.
- ◆ Configuration space.
- ◆ Robot world models.

Robot reasoning/controlling at large

- ◆ We have studied a single feedback control loop.
- ◆ How do we put together multiple feedback control loops?
 - At which level of generality?
 - In what order?
 - In what priority?
- ◆ How do we generate reliable and correct behavior of a robot?

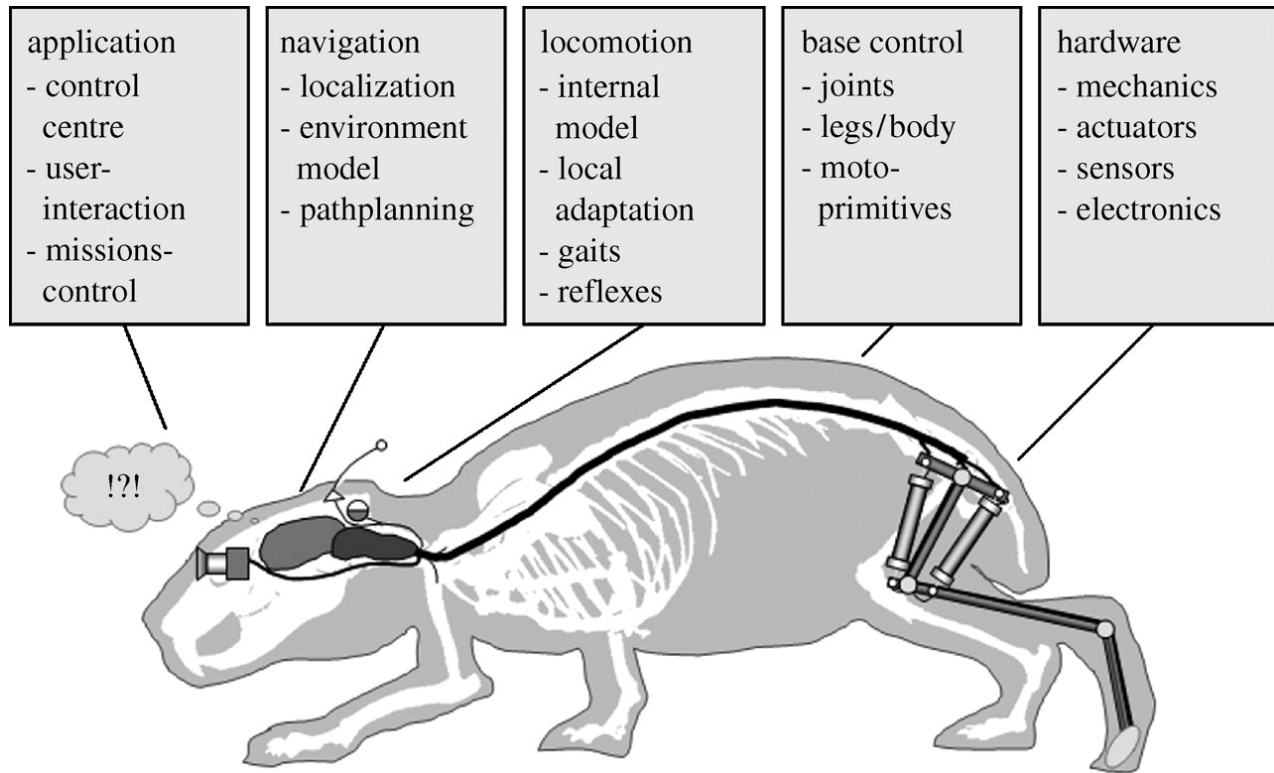
What is intelligence?

- ◆ How do we decide if a particular system or device (e.g., hardware and software combination) is intelligent?
- ◆ The Turing test (1950).
- ◆ ALVINN – Autonomous Land Vehicle in a Neural Network. Carnegie-Mellon Univ. 1993; Learning by observation of a human driver.
 - Inputs: stereo video, human control, laser range finders, radar, inertial navigation.
 - Outputs: steering, acceleration, braking.
 - Processing: neural network, 30×32 unit video input; 30 unit output (steering control), 5 hidden units, winner takes all output.



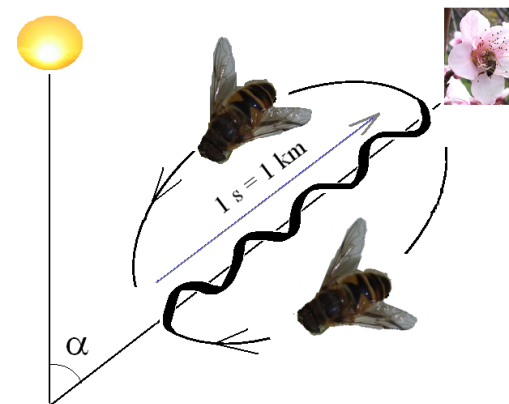
What behaviors are intelligent?

- ◆ Memory.
- ◆ Learning.
- ◆ Language.
- ◆ Planning.
- ◆ Eye gaze following.
- ◆ Predicting how someone will act based on her/his beliefs or knowledge.



A biological motivation

- ◆ Use animal biology as a metaphor for intelligence.
- ◆ Bees, rats, dolphins, slugs, chimpanzees, humans all exhibit forms of intelligence.
- ◆ A bee performs a waggle dance. (in Czech: osmíkový tanec)
- ◆ Dolphins and bats use the echolocation (sonar).
- ◆ Chimpanzees can track another's gaze.
- ◆ Rats are excellent at remembering locations.



Intelligence – an evolutionary metaphor

- ◆ We consider a bee, a snake, a rat, to have forms of intelligence because they exhibit behaviors (including learning behaviors) adapted to particular environments.
- ◆ We can also consider various rather specialized artificial systems to have forms of intelligence, e.g. the already mentioned autonomously driving military truck ALVINN from 1993.

Intelligence

There is a wide agreement among AI researcher that intelligence is required to do the following:

- ◆ Reason, use strategy, solve puzzles, and make judgments under uncertainty.
 - ◆ Represent knowledge, including commonsense knowledge.
 - ◆ Plan.
 - ◆ Learn.
 - ◆ Communicate in a natural language.
 - ◆ Integrate all these skills towards common goals.
-
- ◆ Ability to sense (e.g. see) and act (e.g. move and manipulate objects).
 - ◆ Salience (*in Czech významnost*) – the capacity to recognize importance.
-
- Interdisciplinary approaches to intelligence (cognitive science, computational intelligence) emphasise autonomy and imagination (ability to form concepts that were not programmed in).

Ethical issues in the connection of AI

The issues have been dealt in the science fiction literature for long. Recently, there have been attempts to formulate ethical dimensions of robotics, e.g. in a European parliament document [European Civil Law for Robotics](#) from October 2016.

- ◆ **Consciousness** – to have subjective experience and thought.
- ◆ **Self-awareness** – to be aware of oneself as a separate individual, especially to be aware of one's own thoughts.
- ◆ **Sentience** – ability to feel perceptions or emotions subjectively
- ◆ **Sapience** – the capacity for wisdom.

A Czech language corner

- ◆ Consciousness – *vědomí*.
- ◆ Self-awareness – *uvědomění si sama sebe*.
- ◆ Sentience – *uvědomování si*.
- ◆ Sapience – *prozíravost, často užíváno ironicky*.

Some philosophy

Rationalism

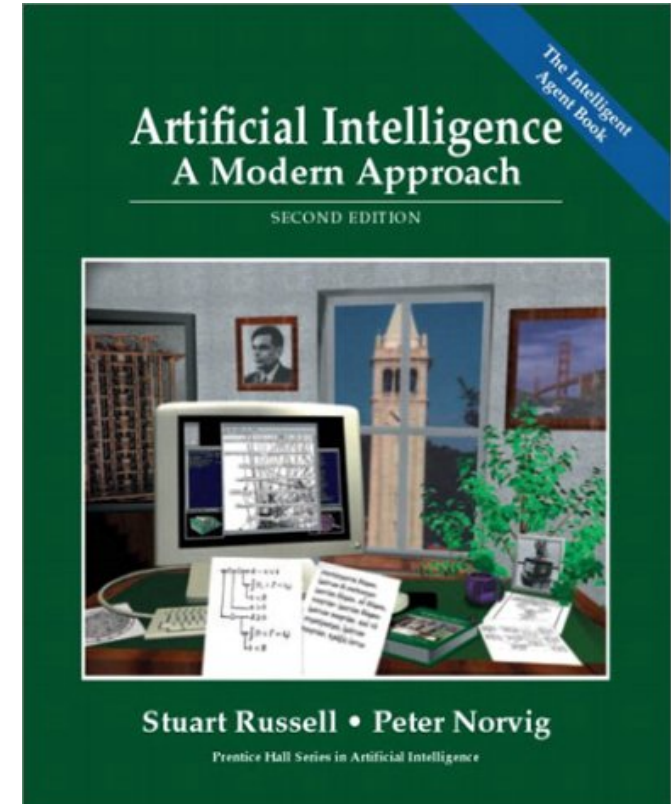
- ◆ “any view appealing to reason as a source of knowledge or justification”, “ the criterion of the truth is not sensory but intellectual and deductive”.
- ◆ Intelligence stems from (logical) reasoning.
- ◆ Many rationalists believe that some part of human knowledge is innate.
- ◆ *E.g., René Descartes (1596–1650), Baruch Spinoza (1632–1677), Gottfried Leibniz (1646–1716).*

Empiricism

- ◆ The idea that intelligence stems from the senses “An empiricist holds that experience is the source of all human knowledge”
- ◆ *E.g., all knowledge must be grounded or based in the sensory world.*
- ◆ *E.g. Aristotle (384 BC–322 BC), William of Ockham (also Occam, 1288–1348), Francis Bacon (1561–1626), John Locke (1632–1704), George Berkeley (1685–1753), David Hume (1711–1776), Hermann von Helmholtz (1821–1894), and Karl Popper (1902–1994).*

Classical AI

- ◆ Also GOFAI = Good Old-Fashioned Artificial Intelligence.
- ◆ Formal mathematical logic approach (“Laws of thought approach”).
- ◆ Making correct inferences.
- ◆ *Program input:*
Description of the problem in a formal, logical notation.
- ◆ *Reasoning:*
Logical inference. Find the solution to the problem, if the solution exists.



A physical symbol system hypothesis

- ◆ A set of arbitrary “physical tokens”, e.g. pen strokes on a paper, holes on a tape, events in a digital computer, that are manipulated on the basis of “explicit rules” that are likewise physical tokens and strings of tokens.
- ◆ The rule-governed symbol-token manipulation is based purely on the shape of the symbol tokens (not their “meaning”), i.e., it is purely syntactic, and consists of “rulefully combining” and recombining symbol tokens.
- ◆ There are primitive atomic symbol tokens and composite symbol-token strings.
- ◆ The entire system and all its parts – the atomic tokens, the composite tokens, the syntactic manipulations both actual and possible and the rules – are all “semantically interpretable”.
- ◆ The syntax can be systematically assigned a meaning e.g., as standing for objects, as describing states of affairs.

Symbol grounding

- ◆ Computationalism (one of cognition theories) states: cognition (i.e., thinking) is just a form of computation.
- ◆ Symbols are manipulated according to rules that are based on the symbols' shapes, not their meanings.
- ◆ How are those symbols (e.g., the words in our heads) connected to the things they refer to?
- ◆ The symbols in an autonomous hybrid symbolic+sensorimotor system would be grounded. (Importance of the perception-action cycle.)
- ◆ But whether its symbols would have meaning rather than just grounding is something cognitive science itself—cannot determine, or explain.

Harnad, S. (1990) The Symbol Grounding Problem. Physica D42: 335-346.

Weak and strong Artificial Intelligence

1. Weak Artificial Intelligence (also called Applied AI)

- ◆ The goal is to achieve practical outcomes:
- ◆ Examples: ALVINN, Deep blue – a chess program.

2. Strong Artificial Intelligence

- ◆ The goal is to make machines truly intelligent (to match or exceed human intelligence), to have minds and be conscious.
- ◆ Example: whole brain simulation (called also mind uploading).

A distinction between two positions in the philosophy of AI is given by John Searle as part of his Chinese room argument.

Chinese room argument, J. Searle 1980

- ◆ There is a program that gives a computer the ability to carry on an intelligent conversation in written Chinese.
- ◆ If the program is given to someone who speaks only English to execute the instructions of the program by hand, then in theory, the English speaker would also be able to carry on a conversation in written Chinese.
- ◆ However, the English speaker would not be able to understand the conversation because she/he does not know how to interpret symbols.
- ◆ J. Searle concludes, a computer executing the program would not understand the conversation either.

The argument is directed against the philosophical positions of functionalism and computationalism, which hold that the mind may be viewed as an information processing system operating on formal symbols. The argument applies only to digital computers and does not apply to machines in general.

Practical AI approaches

1. Engineering approach

- ◆ Building intelligent entities.
- ◆ The most important goal is often behavior and not mechanism.

2. Cognitive modeling approach

- ◆ Tries to achieve “psychological reality”, e.g. use studies from psychology on how humans think.
- ◆ Trace of programs reasoning steps should follow human steps in reasoning. E.g. General Problem Solver (A. Newell & H.A. Simon, 1959), the first computer program which separated its knowledge of problems (rules represented as input data) from its strategy of how to solve problems (a generic solver engine, mathematical logic).

AI issues in robotics

- ◆ A **state space** – a set of possible configurations of a robot, usually a discrete one, e.g. a position on a occupancy grid.
- ◆ **Representation of a robot environment** (a practical implementation of the state space), e.g. the **robot world model**.
- ◆ A **reasoning engine** – usually some type of symbol manipulation mechanism used for search/planning in a state space.
- ◆ **Ontology** (*next slide*).

Ontology

◆ In philosophy:

Def: Ontology is the philosophical study of the nature of being, existence, or reality, as well as the basic categories of being and their relations.

◆ In computer and information science:

Def: Ontology formally represents knowledge as a set of concepts within a domain, and the relationships between pairs of concepts.

- The ontology can be used to model a selected application domain and support reasoning about its entities.
- The notion of a semantically relevant object enables to simplify the issues. The objects and their detections provide a simplistic version of the symbol grounding.
- Ontologies are the structural frameworks for organizing information and are used in artificial intelligence, the Semantic Web, systems engineering, software engineering, biomedical informatics, library science, enterprise bookmarking, and information architecture as a form of knowledge representation about the world or some part of it.

Need for a robot control architecture

- ◆ A robot control architecture provides **principles for organizing** robot reasoning and robot world representation, sensing and perception, and actuation modules (both hardware and software), which are called jointly “**a robot control architecture**”.
- ◆ The control architecture specifies building blocks and their interrelations.
- ◆ The control architecture provides:
 - Structure.
 - Constraints.
- ◆ There have been **four control architectures** mainly used:
 1. Deliberative.
 2. Reactive.
 3. Behavior-based.
 4. Hybrid.

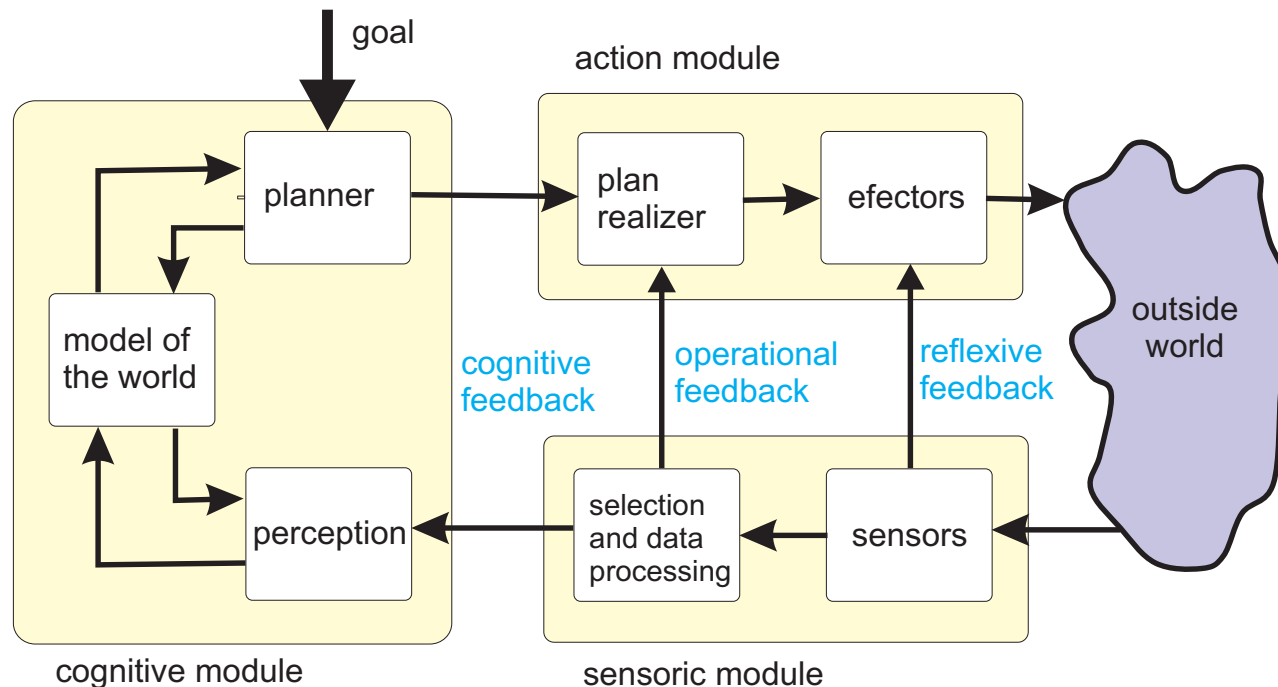
Brief comparison of four control architectures

<i>Architecture</i>	<i>Properties</i>
Deliberative	<p>Think hard, act later.</p> <p>Lots of states.</p> <p>Maps of the robot environment.</p> <p>Look ahead.</p>
Reactive	<p>Do not think, react.</p> <p>No states.</p> <p>No maps.</p> <p>No look ahead.</p>
Behavior-based	<p>Think the way you act.</p> <p>Some states.</p> <p>Look ahead only while acting.</p> <p>Reactive + state.</p>
Hybrid	<p>Think and act independently, in parallel.</p> <p>States.</p> <p>Look ahead but act.</p> <p>Combines long and short time scales.</p>

Motivating example, modules:

A cognitive robot with a deliberative architecture

The deliberative architecture, the most sophisticated and oldest one from the artificial intelligence point of view, is used here to motivate main involved building blocks.



This lecture introduces two issues only

Only two blocks will be introduced in this lecture and explicated in more detail in coming lectures:

- ◆ **Motion planner** (only briefly, to motivate the second block).

The higher level task planning, needed in some more complicated scenarios, is not discussed here for a moment.

E.g. a mobile robot, which should play chess and be one of the players physically acting on a chessboard.

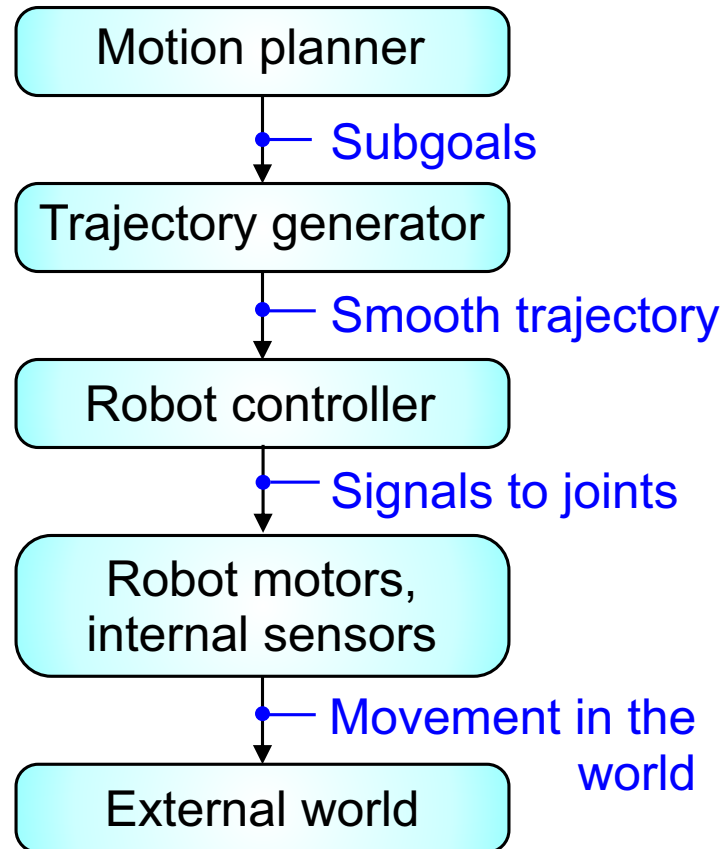
Only motion of a robot would be of concern in this lecture.

- ◆ **Model of the robot world** (called often a map in mobile robotics).

Planning for manipulators and mobile robots

- ◆ The intention is to use a single formalism for planning in both most practically useful tasks:
 - Open kinematic chain manipulators.
 - Mobile robots.
- ◆ Involved tasks:
 - **Localization**. *Where am I?*
E.g. odometry, localization with respect to external sensors (as GPS), beacons or landmarks.
 - **Planning/trajectory generation**.
Where am I going? (= goal)
How do I get there? (= plan or trajectory).
- ◆ Complications: errors in maps, sensing, control, changing world, unexpected obstacles, ...

Motion planning



Global path vs. local trajectory

- ◆ Finding a path/trajectory between the robot initial configuration and its desired final configuration can be formulated as a **single optimization task** in theory.
- ◆ However, this task has a **prohibitively high computational complexity** in most cases.
Why? The robot has several degrees of freedom, which induces a high dimensional search space. The search algorithms (as finding the shortest path in a graph) have exponential complexity.
- ◆ The **pragmatic solution** (divide and conquer) is to **separate the path/trajectory finding** into two subtasks:
 - Global path planning.
 - Local trajectory generation (local obstacle avoidance in mobile robotics).

Terminology: path vs. trajectory

- ◆ *Note: Terms path, trajectory are often confused. They are used as synonyms informally.*
- ◆ **Path** is an ordered locus of points in the space (either joint or operational), which the robot should follow.
 - Path provides a pure geometric description of motion.
 - Path is usually planned globally taking into account obstacle avoidance, traversing a complicated maze, etc.
- ◆ **Trajectory** is a path augmented with velocities and accelerations in its each point.
 - A design of a trajectory does not need global information, which simplifies the task significantly.
 - The trajectory is specified and designed locally. Parts of a path are covered by individual trajectories.
 - It is often required that pieces of trajectories join smoothly, which induces that a single trajectory design takes into account neighboring trajectories from the path only.

Path/trajectory and a robot world map

- ◆ **Path planning** is strongly related with the global world map and is usually performed **off-line** (because of the above mentioned high computational complexity).
- ◆ **Trajectory generation** (or local obstacle avoidance) can be regarded locally as independent of a global map. It is usually performed **on-line**.
- ◆ For global path planning, the robot world map should be given (e.g. by a floor plan of a building).
 - If it is not given, a map is obtained off-line before path planning.
 - Sometimes, map building and localization can be done simultaneously. It is called as SLAM (Simultaneous Localization and Mapping).
- ◆ With industrial robots, path planning is not usually solved by a computer because a human operator plans the paths.

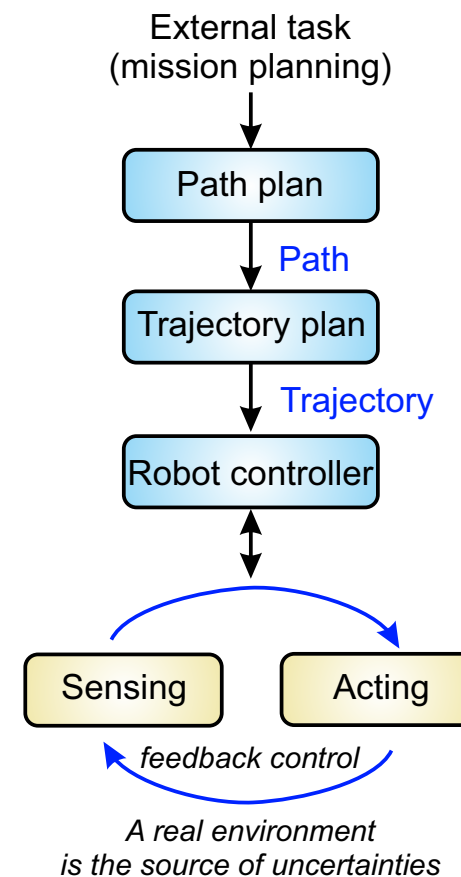
Robot motion planning, an overview

Path planning

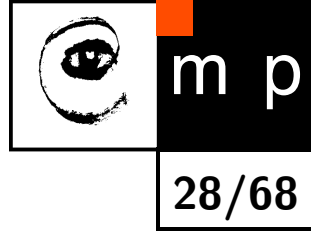
- ◆ Geometric path.
- ◆ Issues: obstacle avoidance, shortest path.

Trajectory generating

- ◆ The path planning provides the input – the chunk of a path usually given as a set of points defining the trajectory.
- ◆ “Approximate” the desired path chunk by a class of polynomial functions and
- ◆ generate a sequence of time-based “control set points” for the control of manipulator from the initial configuration to its destination.



Robot & representations



- ◆ What does the robot know and keep in its brain?
- ◆ Representation is the form in which information is stored or encoded in the robot.
- ◆ The robot may need to remember what happened in the past or predict what will happen in the future.
- ◆ The robot may need to store maps of the environment, images of people or places.

Spatial memory

- ◆ Spatial memory is the world representation that the robot uses.
- ◆ Provides data structures and methods for storing/retrieving processed sensory input, a priori information about the environment (as a road map for a self-driving car).
 - E.g. “Drive down the corridor to the third yellow door on the right.”
- ◆ Organized to support methods, which can extract relevant expectations about the navigational task.
 - E.g. “Drive down the corridor to the third yellow door on the right.”
 - Robot can query its spatial memory and notice that the odd numbered doors are blue and on the left side, and even numbered doors are yellow and on the right side.

Spatial memory supports four basic functions

1. **Attention:**

What features, landmarks to seek next?

2. **Reasoning:**

Can I fit through that door?

3. **Path planning:**

What is the best way through this space?

4. **Collecting information:**

- ◆ What does this place look like?
- ◆ Have I ever seen it before?
- ◆ What has changed since I was here last time?

Two forms of spatial memory

Qualitative (route):

- ◆ Expresses space in terms of connection between landmarks.
- ◆ Dependent upon perspective of the robot.
- ◆ Orientation clues are egocentric.
- ◆ Usually, it cannot be used to generate quantitative (metric/layout) representations.

Quantitative (metric, layout):

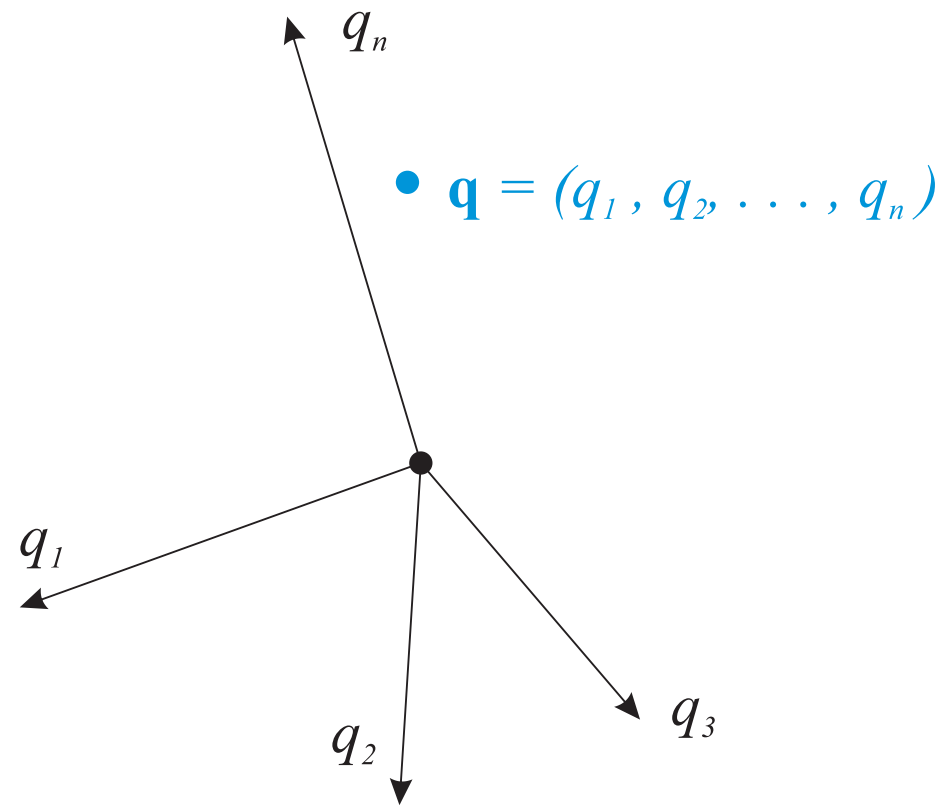
- ◆ Expresses space in terms of physical distances of travel.
- ◆ Bird's view of the world.
- ◆ Does not depend on the perspective of the robot.
- ◆ Independent of the position and orientation of the robot.
- ◆ Can be used to generate qualitative (route) representations.

State, state space

- ◆ **State**: a collection of parameters sufficient for a robot (system) description.
 - Observable state: the robot knows its states all the time.
 - Hidden state, inaccessible/unobservable state, partially observable state.
 - ◆ **State space**: all possible states a robot can be in.
 - ◆ **External state**: sensed by robot sensors or known as night/day, at home.
- ◆ **Internal state**:
 - Sensed: e.g., the robot position, velocity.
 - Remembered: can be used to store/recall information about the robot world, called the representation or the model of the robot world.
 - ◆ The nature and complexity of the representation influences significantly planning and the robot control.

Robot configuration, configuration space \mathcal{C}

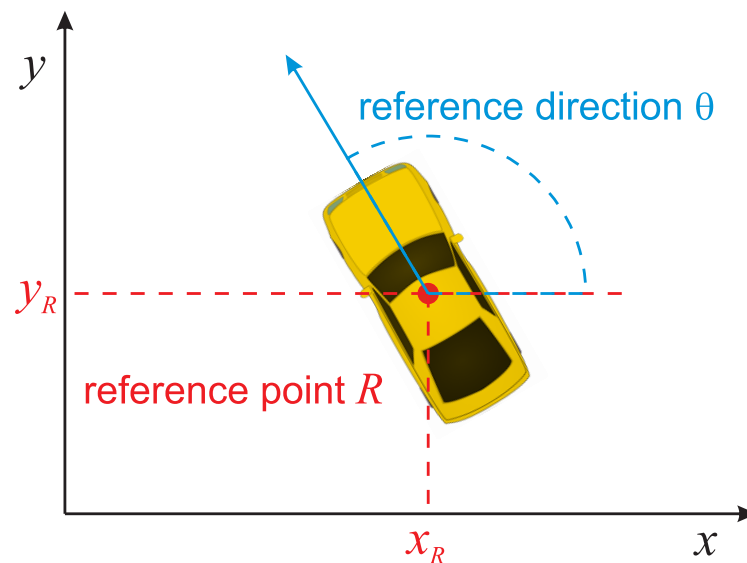
- ◆ A robot configuration is a specification of the positions of all robot points relative to a fixed coordinate system.
- ◆ A configuration is usually expressed as a vector of position and orientation parameters.
- ◆ The dimension of the configuration vector equals the number of parameters $\mathbf{q} = (q_1, q_2, \dots, q_n)$.
- ◆ The configuration space \mathcal{C} is the set of all possible configurations.
- ◆ A configuration is a point in \mathcal{C} .



Configuration space, continued

- ◆ Configuration space (\mathcal{C} -space for short) = the space of all possible configurations of an object.
- ◆ Typically represents “occupied” and “free” space.
- ◆ The dimension of the \mathcal{C} -space equals to the number of the parameters representing a configuration (degrees of freedom).
- ◆ The greater the dimension of the \mathcal{C} -space, the more complex the motion planning problem will be. A “good \mathcal{C} -space” reduces the number of dimensions that the planner has to deal with.
- ◆ Origin:
 - Tomas Lozano-Perez: Spatial Planning: A configuration space approach, MIT Artificial Intelligence Lab Memo. 605, 1980.
 - Tomas Lozano-Perez: Spatial planning: A configuration space approach. IEEE Transaction on Computers, Vol. C-32, No. 2, 1983, pp. 108-120.

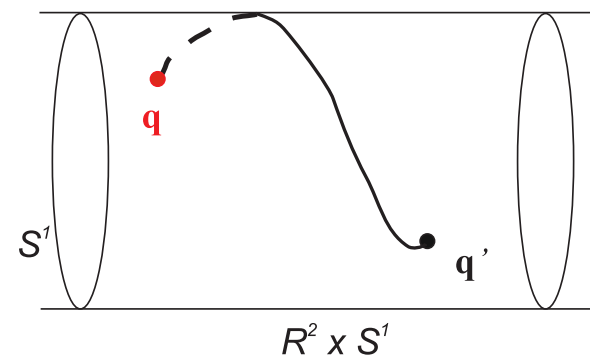
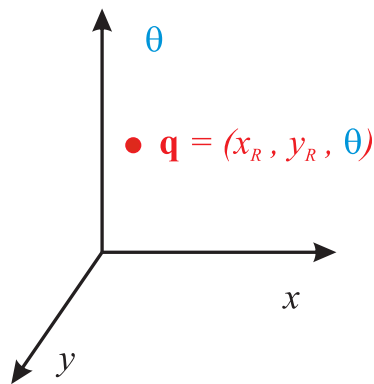
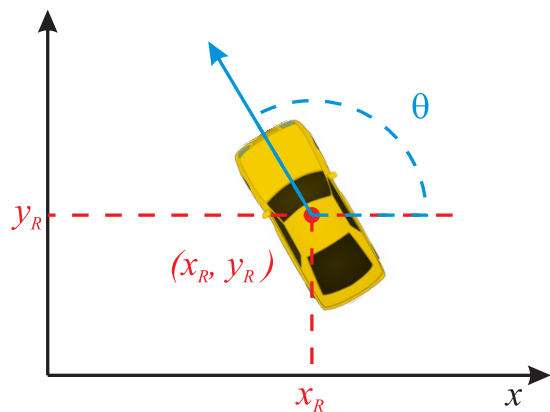
Configuration space example, a 2D rigid robot (1)



- ◆ In 2D (Euclidean) space: a 3-parameter representation, e.g. for a car in a planar world $q = x_R, y_R, \Theta$. See figure above.
- ◆ In 3D (Euclidean) space: a 6-parameter representation, e.g. for an airplane $q = x_R, y_R, z_R, \alpha, \beta, \gamma$, where α, β, γ are Cardan angles yaw, pitch, roll.

Configuration space example, a 2D rigid robot (2)

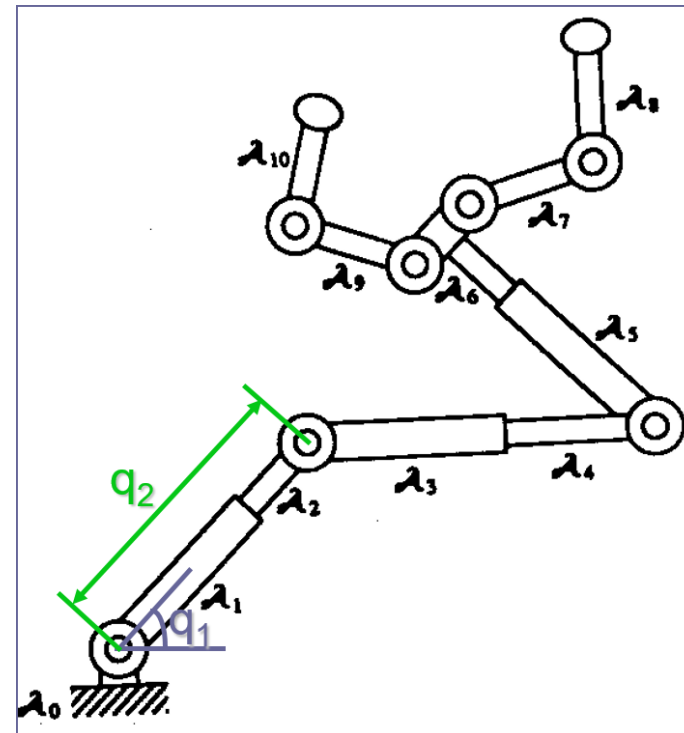
- ◆ We like to illustrate that the topology of the configuration space is not that of a Euclidean (Cartesian) space in our 2D rigid robot case.
- ◆ One configuration of the robot in 2D space (figure, left) and the related Cartesian space (figure, middle).



- ◆ A set of all possible configurations q of this rigid robot in 2D Euclidean space leads to a 3D cylinder embedded in 4D space. The possible configurations are only on the surface of this cylinder. One possible change of the configuration q to q' is shown on (figure, right). We explain later what S^1 and $R^2 \times S^1$ are.

Configuration of an articulated robot

- ◆ 10 degrees of freedom, 5 rotational joints, 5 prismatic joints.
- ◆ $\mathbf{q} = (q_1, q_2, \dots, q_{10})$

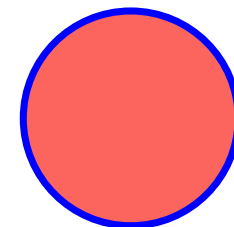


Metric space; to have something to start from

- ◆ Consider a **metric space** \mathcal{M} , i.e. a set, for which a **distance** d between all members of the set are defined. Those distances, taken together, are called a metric on the set. A metric on a space induces topological properties like open and closed sets, which lead to the study of more abstract topological spaces.
- ◆ The most familiar metric space is 3-dimensional Euclidean space.
- ◆ A metric space \mathcal{M} is a special case of a topological space (to be introduced), and therefore all definitions and theorems about general topological spaces also apply to all metric spaces.

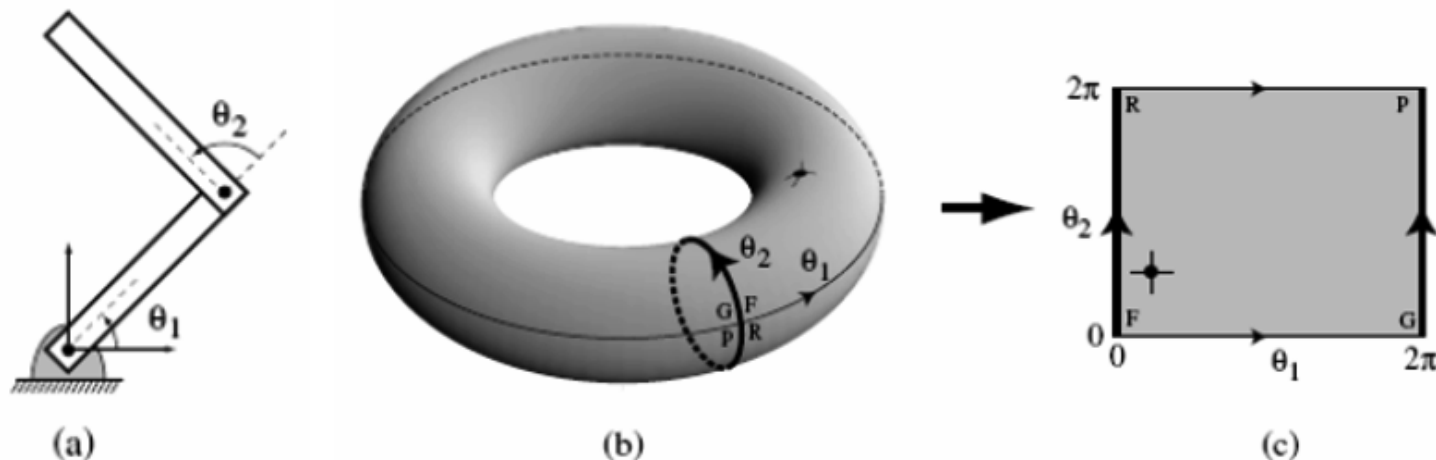
Neighborhood, open/closed set

- ◆ About any point $p \in \mathcal{M}$ we define the **open ball** with radius r :
 $B_r(p) = \{p' \in \mathcal{M} \mid d(p, p') < r\}$. Note: The complement of an open set is called closed.
- ◆ The **neighborhood** \mathcal{U} of a point $p \in \mathcal{M}$ such a set that for every $p' \in \mathcal{U}$, $B_r(p') \subset \mathcal{U}$.
- ◆ An open set is an abstract concept in topology. We introduce it in the metric space which is more intuitive.
- ◆ An **open set** can be defined as a set containing a ball around each of their points.
- ◆ Example: The red disk represents the set of points (x, y) satisfying $x^2 + y^2 < r^2$. The blue circle represents the set of points (x, y) satisfying $x^2 + y^2 = r^2$. The red set is an **open set**, the blue set is its **boundary set**, and the union of the red and blue sets is a **closed set**.



Plan towards manifolds; need for topology

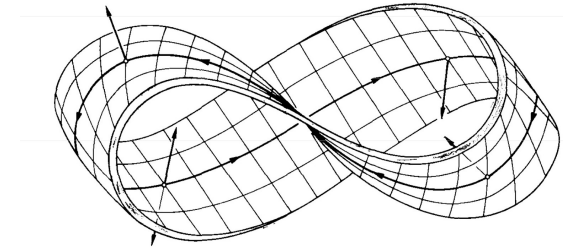
- ◆ We like to define a manifold, a mathematical concept useful to express a subspace of n -dimensional configuration space. Manifolds serve, e.g. to represent eligible points, e.g. reachable by the tip of robot arm.
- ◆ Consider a 2-DoF manipulator with rotational joints, $\theta_1, \theta_2 \in \langle 0, 2\pi \rangle$.



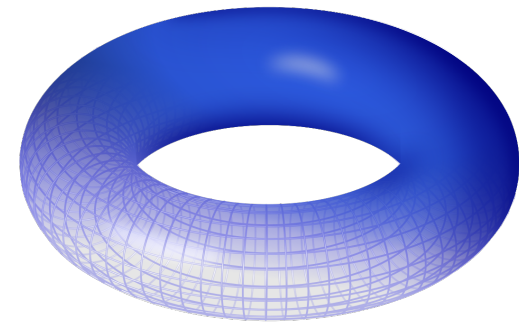
- ◆ The shape of 2-DoF rotational manipulator workspace = surface of a toroid, Fig. (b), motivates for topological considerations in relation to \mathcal{C} -space. The pictorial example above shows that the workspace topology differs from the Cartesian space.

Topology of a (configuration) space

- ◆ Topology is the intrinsic characteristic of a space.
- ◆ Two spaces have a different topology if cutting and pasting is required to make them the same (e.g. a sheet of paper vs. a Mobius strip) vs. torus.
- ◆ Think of a rubber object. If we can stretch and reshape (“continuously” without tearing) one object into the other object, they have the same topology.
- ◆ A basic mathematical tool for talking about topology is the homeomorphism.
- ◆ The distance does not play a role in topology. Neighborhood does.



Mobius strip



Torus

Topology, topological space

- ◆ Topology (in mathematics) is concerned with the properties of space that are preserved under continuous deformations, such as stretching, crumpling and bending, but not tearing or gluing.
- ◆ This can be studied by considering open sets, a collection of subsets.
- ◆ A topological space relies only upon set theory and is the most general notion of a mathematical space that allows for the definition of concepts such as continuity, connectedness, and convergence.
- ◆ Other spaces, such as manifolds and metric spaces, are specializations of topological spaces with extra structures or constraints.

Rehearsal of needed mathematical concepts/notations

- ◆ \mathbb{R} – real numbers (represent quantities along a line, a 1-Cartesian space).
- ◆ \mathbb{R}^n – n -dimensional Cartesian space.
- ◆ A, B are sets; Cartesian product $A \times B = \{(a, b) | a \in A \text{ and } b \in B\}$;
power notation $A^n = \underbrace{A \times A \times \dots \times A}_{n\text{-times}}$
- ◆ \mathcal{S}_1 – boundary (abbreviated ∂) of a circle in 2D
- ◆ \mathcal{S}_2 – boundary (abbreviated ∂) of a sphere in 3D
- ◆ $SO(2), SO(3)$ – a set of 2D, 3D rotation (a special orthogonal group). It was explained in the lecture *Geometry for kinematics*.
- ◆ $SE(2), SE(3)$ – a set of rigid 2D, 3D translations (a special Euclidean group)

Homeomorphism and diffeomorphism

◆ Mapping $\phi: S \rightarrow T$, rehearsal

- If each element of S maps to a unique T then ϕ is an **injective** or one-to-one mapping.
- If each element of T has a corresponding preimage in S , then ϕ is a **surjective** (or onto) mapping.
- If ϕ is surjective and injective then it is **bijective**. The inverse mapping ϕ^{-1} exists.
- Mapping ϕ is **smooth** if derivatives of all orders exist. We say that ϕ is C^∞ **continuous**.

◆ If $\phi: S \rightarrow T$ is a bijection, and both ϕ and ϕ^{-1} are continuous then ϕ is a **homeomorphism**. If such a ϕ exists, spaces S and T are homeomorphic.

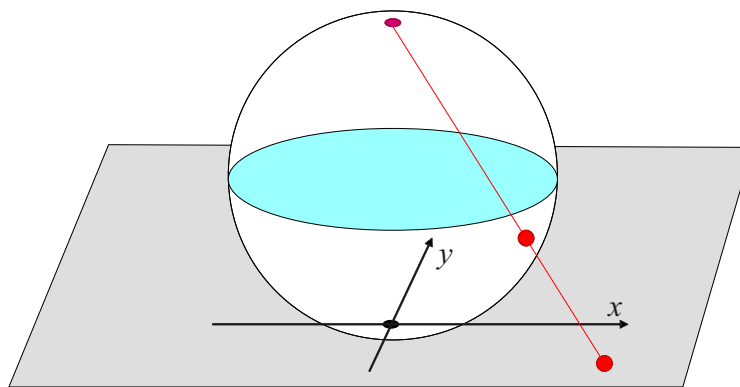
◆ **Diffeomorphism** (an infinitely differentiable homeomorphism) is a special case of homeomorphism where both ϕ and ϕ^{-1} are smooth.

Three homomorphism/diffeomorphism examples

1. The rectangle and the square in 2D are diffeomorphic.
How would you show that?
2. The circle and the ellipse are diffeomorphic.
How would you show that?
3. A “racetrack” (composed of two straight segments and two semicircular segment) is not diffeomorphic to a circle. At four connections between straight lines and semicircular segments, there is C^0 continuity and discontinuities from C^1 on. Smoothness is violated. It is homeomorphic, though.
How would you show that?

Manifolds

- ◆ Informally: a manifold is a topological space that locally resembles Euclidean space near each point.
-
- ◆ A space S is locally diffeomorphic (homeomorphic) to a space T if for each point $p \in S$ there is a neighborhood containing it, for which a diffeomorphism (homeomorphism) to some neighborhood of T exists.
 - ◆ The sphere is locally diffeomorphic to the plane (as is the torus).



- ◆ A set S is a k -dimensional manifold if it is locally homeomorphic to \mathbb{R}^n .

Examples of manifolds

- ◆ A surface in a 3D Euclidean space is a manifold.
- ◆ More generally, a manifold embedded in n -D Euclidean space locally looks like a $(n - 1)$ -D Euclidean space.
For example Earth (a big sphere) is a big manifold embedded in 3D space. But, we as tiny entities living on its surface can only see flat 2D land. So locally at every point on a sphere, it looks like a 2D plane.
- ◆ Curves in 3D space are also manifolds. However, they locally look like 1D vector space (a line). To be a little more precise, a manifold embedded in n -dimensional Euclidean space looks locally like k -dimensional vector space ($k \leq n$) at every point on it.
- ◆ A manifold can be thought of a smooth geometric figure of a certain dimension. “Smooth” means it has no kinks, self-crossings and such. It may or may not be allowed to have corners (like a cube, or a square), depending on whether it’s a “smooth manifold” or just a “topological manifold”.

Chart, atlas, differentiable manifolds

- ◆ A **chart** is a pair (U, ϕ) such that U is an open set in a k -dimensional manifold and ϕ is a diffeomorphism from U to some open set in \mathbb{R}^n .
 - Consider the chart as a “coordinate system” for U (e.g. lines of latitude and longitude away from the poles).
 - The inverse map is a parameterization of the manifold.
- ◆ Many manifolds require more than one chart to cover (e.g. the circle requires at least 2).
- ◆ An **atlas** is a set of charts that
 - cover a manifold
 - are smooth where they overlap (the theory defines the notion of C^∞ related for this; we will take this for granted).
- ◆ A set S is a **differentiable manifold** of dimension n if there exists an atlas from S to \mathbb{R}^n . For example, this is what allows us (locally) to view the (spherical) earth as flat and talk about translational velocities upon it.

Mathematical concepts/notations continued

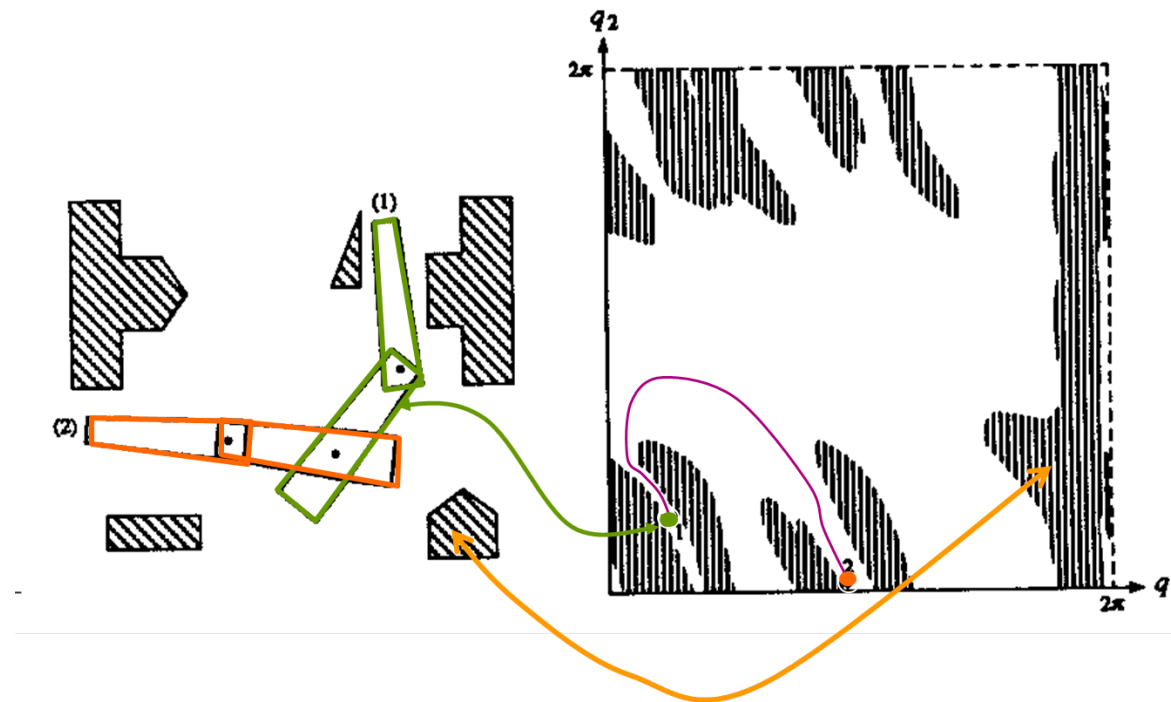
- ◆ $\mathbb{R}^1 \times \mathbb{R}^1 \times \dots \times \mathbb{R}^1 = \mathbb{R}^n$ – n -dim Euclidean space.
- ◆ $\mathcal{S}^1 \times \mathcal{S}^1 \times \dots \times \mathcal{S}^1 \neq \mathcal{S}^n$, i.e. n -dimensional sphere in \mathbb{R}^{n+1} .
- ◆ $\mathcal{S}^1 \times \mathcal{S}^1 \times \dots \times \mathcal{S}^1 = \mathcal{T}^n$, i.e. n -dimensional torus.
- ◆ Although \mathcal{S}^n is an n -dimensional manifold, it is not a manifold of a single chart. There is no single, smooth, invertible mapping from \mathcal{S}^n to \mathbb{R}^n .
- ◆ $\mathcal{S}^1 \times \mathcal{S}^1 \times \mathcal{S}^1 \neq SO(3)$.
- ◆ $SO(2) \neq \mathbb{R}^3$.
- ◆ $SO(3) \neq \mathbb{R}^6$.

Robot examples and their configurations spaces

<i>Type of a robot</i>	<i>Representation of Q</i>
Mobile robot translating in a plane	\mathbb{R}^2
Mobile robot translating and rotating in a plane	$\mathbb{R}^2 \times \mathcal{S}^1$ or $SE(2)$
An aircraft in a 3D space	$\mathbb{R}^3 \times SO(3)$ or $SE(3)$
An j -joint open-chain revolute arm	\mathcal{T}^n
A planar mobile rotating, translating with n -joint arm	$SE^3 \times \mathcal{T}^n$

Obstacles in \mathcal{C} -space, motivating picture

Robot arm with two DOFs. The task is to move from the point (1) to the point (2) not touching the obstacles.

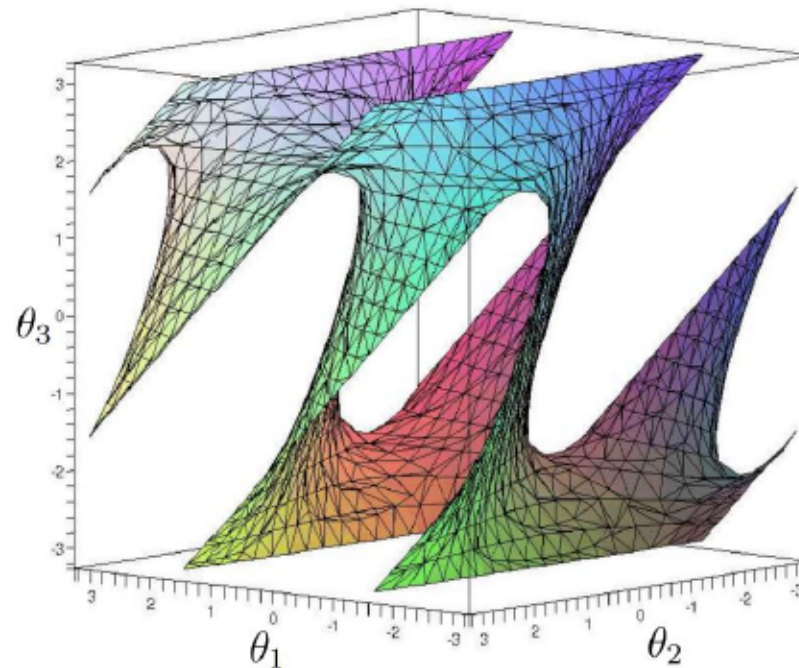
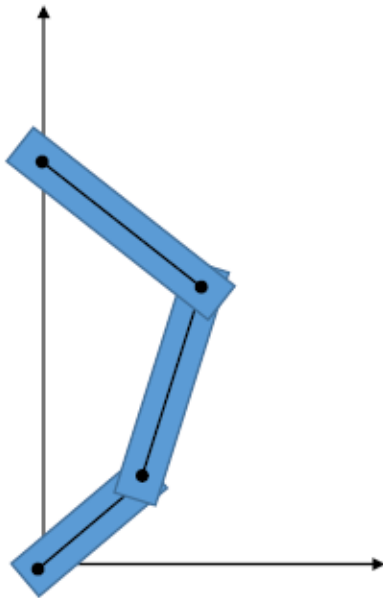


Euclidean (Cartesian) space

\mathcal{C} -space

3 DoF robot arm, \mathcal{C} -space visualization

A 3-chain line in 2D with one end on the origin and the other on y axis.



\mathcal{C} -space between two rigid objects

- ◆ Given two rigid bodies \mathcal{A} (object), \mathcal{B} (obstacles).
 - ◆ The configuration space (\mathcal{C} -space) between them is a high-dimensional non-Euclidean space composed of all possible configurations of objects \mathcal{A} , \mathcal{B} .
 - ◆ Any configuration in \mathcal{C} -space corresponds to a relative position and orientation of objects \mathcal{A} , \mathcal{B} .
 - ◆ In robot motion planning, \mathcal{C} -space is used to find a collision-free path of an object \mathcal{A} among set of obstacles \mathcal{B} .
- ◆ Path planning reduces to finding a free path for a point in a higher-dimensional \mathcal{C} -space.
 - ◆ The calculation complexity of the exact path computation in \mathcal{C} -space is exponential in number of dimensions (degrees of freedom).
 - ◆ Approximative calculations are used.

\mathcal{C} -space between two rigid objects (2)

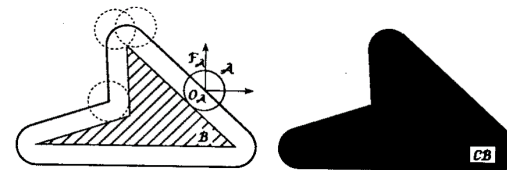
- ◆ \mathcal{C} -space is denoted \mathcal{C} for brevity in this slide only.
 - ◆ Any configuration $\mathbf{q} = (q_1, q_2, \dots, q_n)$ corresponds to a relative position and orientation of \mathcal{A} with respect to \mathcal{B} in \mathcal{C} .
 - ◆ A configuration \mathbf{q} is represented as a point in \mathcal{C} .
-
- ◆ For 2D objects and translation only, the \mathcal{C} has 2 DOFs.
 - ◆ For 2D objects and translation and rotation, the \mathcal{C} has 3 DOFs.
-
- ◆ For 3D objects and translation only, the \mathcal{C} has 3 DOFs.
 - ◆ For 3D objects and translation and rotation, the \mathcal{C} has 6 DOFs.
-
- ◆ For articulated objects, e.g. a manipulator arm, \mathcal{C} has a higher dimension.

\mathcal{C} -space between two rigid objects (3)

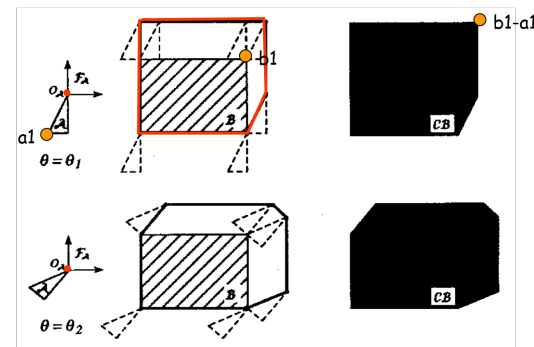
- ◆ It is assumed that object \mathcal{A} is movable and obstacles \mathcal{B} are fixed.
- ◆ $\mathcal{A}(\mathbf{q})$ means an object \mathcal{A} located at the configuration \mathbf{q} .
- ◆ \mathcal{C} -space is decomposed into two subspaces:
 - Free \mathcal{C} -space
 $\mathcal{C}_{free} = \{\mathbf{q}: \mathcal{A}(\mathbf{q}) \cap \mathcal{B} = \emptyset\}.$
 - Obstacle \mathcal{C} -space
 $\mathcal{C}_{obs} = \{\mathbf{q}: \mathcal{A}(\mathbf{q}) \cap \mathcal{B} \neq \emptyset\}.$
- ◆ The boundary of \mathcal{C}_{free} is denoted $\partial \mathcal{C}_{free}$.

\mathcal{C} -space examples:

- ◆ Disc robot in 2-D workspace



- ◆ Rigid Robot Translating in 2-D



\mathcal{C} -space as a simplification; The mobile robotics motivating example

- ◆ Introducing \mathcal{C} -space helps reducing the number of DOFs in order to simplify planning.
- ◆ Typical use for indoor mobile robots.

Assumptions:

- \mathcal{C} -space represents occupied and free spaces.
- The robot is considered small (represented as a point) and holonomic, i.e. the robot rotation is not considered.
- Polygonal approximation of the robot and obstacles.
- Obstacles are dilated (grown) by the shape of the robot instead.

Outcomes:

- Two DoFs \mathcal{C} -space suffice in the case of planar environment.
- Three DoFs \mathcal{C} -space is needed for 3D environment.
- Navigation then simplifies to finding routes through freespace.

Robot world model

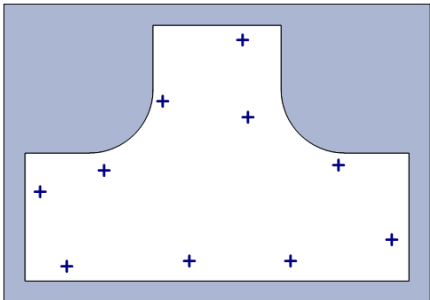
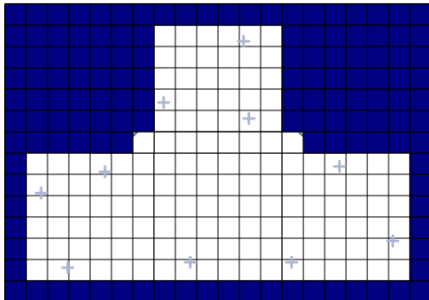
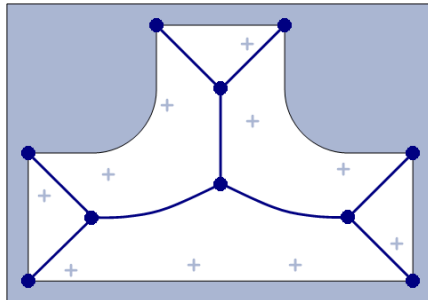
(Called also robot environment model.)

- ◆ Robot representation of its environment is called **robot world model** (or robot environment model).
- ◆ Robot world model, **representation types**:
 - **Continuous**: dynamic relations between subsets of a space (e.g. a 3D Euclidean space), i.e. a function of several time-dependent variables.
 - **Discrete**: e.g., composed of cells as an occupancy grid representing a floor plan. A particular cell x, y can tell that it is a (part of) obstacle.
 - **Symbolic**: e.g., this particular manipulation task needs that type of a gripper.
Relations between symbols are usually represented by a graph (graphs).

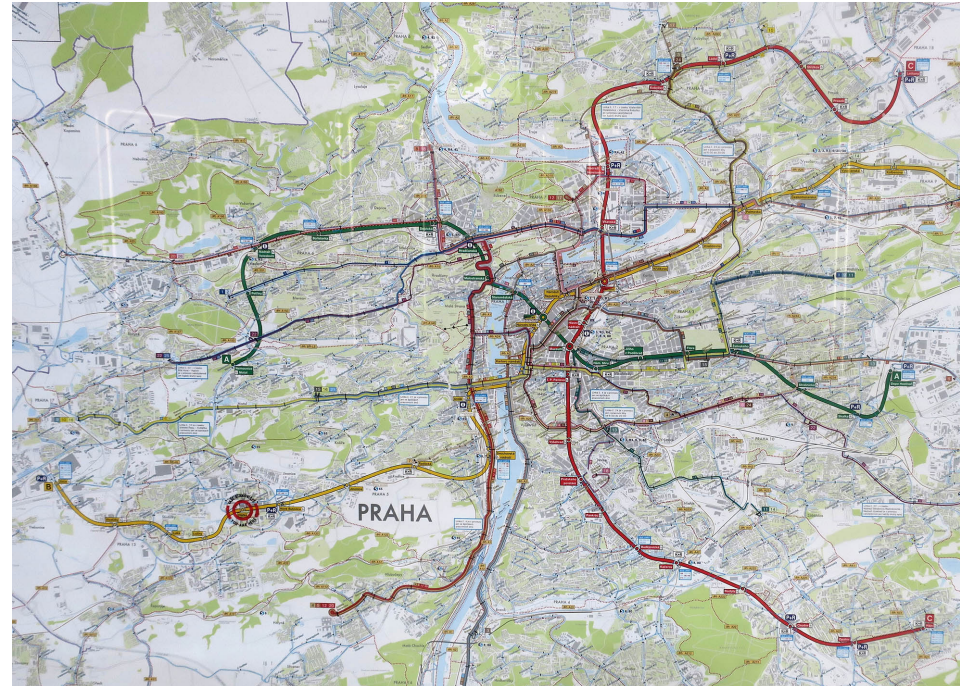
Examples: world maps in mobile robotics

- ◆ An **odometric path** to store/recall the route traveled.
- ◆ A **landmark-based map** (also feature-based), which tells the robot what to do at each landmark regardless of the order.
- ◆ A **metric map**, also geometric map, e.g. a robot may recall a maze by drawing it by using exact lengths of corridors and distances to walls.
 - It can describe a location of a robot and objects in its world as a configuration space. For mobile robots, it collapses 6 DOF to 2 DOF usually by assuming planar world and that the robot can rotate on a spot (so the robot direction is not important). Only the obstacle location matters in such a case.
 - A discrete approximation leads to a **discrete map** (also a grid map). a sub-division into elements of equal size (e.g., squares). If obstacles are represented \Rightarrow **occupancy grid**.
- ◆ A **topological map** is subdivided into elements of differing sizes (e.g., rooms in a building with connections between these elements). Alternatively, it can be a collection of landmarks with links.

Comparison of three map models

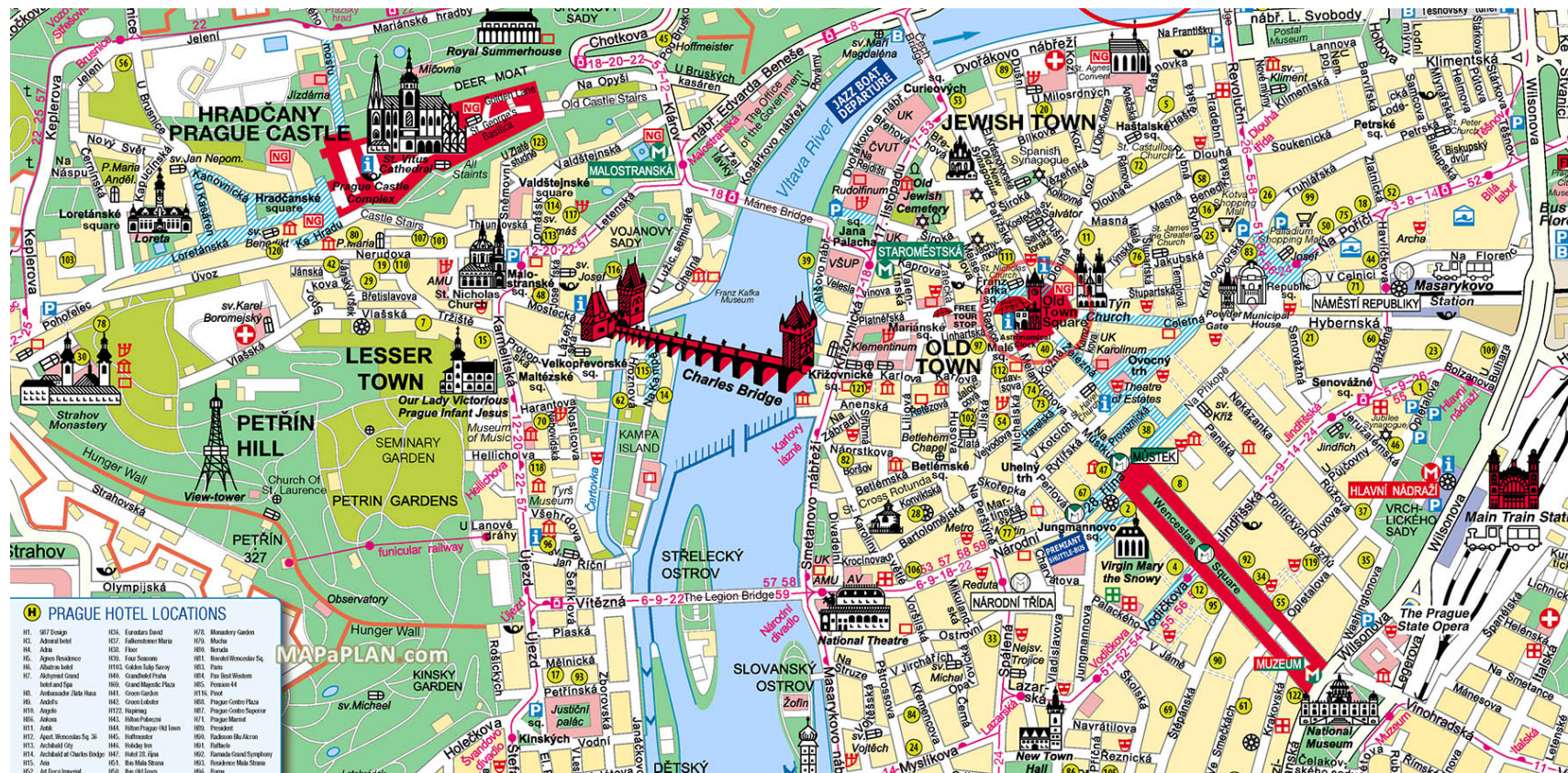
	Landmark-based	Grid-based	Topological
	<p>Collection of landmark locations</p> 	<p>Collection of discretized obstacle/free-space cells</p> 	<p>Collection of graph nodes and their interconnections</p> 
Resolution	Arbitrary localization	Discrete localization	Localize to graph nodes
Complexity	# of landmarks	Grows with resolution	Minimal loc. complexity
Strategies	No inherent exploration	Frontier exploration	Graph exploration

Prague metro, metric and topological maps



Prague, metric and semantic map

61/68



Graph representation of maps

- ◆ Discrete maps invite to use a **graph representation**.
- ◆ Every semantically relevant element of the map corresponds to a **vertex** (also called a graph node). Graph vertices are connected by **edges**.
- ◆ For example, a road map is a topological map, where road intersections are vertices and graph edges are roads connecting them.
- ◆ **Graph theory** provides algorithms for solving tasks on graphs, e.g. finding a shortest path between two vertices of a graph (e.g. a road map).

Metric, topological maps and their advantages

◆ Metric maps:

- Enable extrapolating between known locations.
- Supports fusing sensor/motor data.

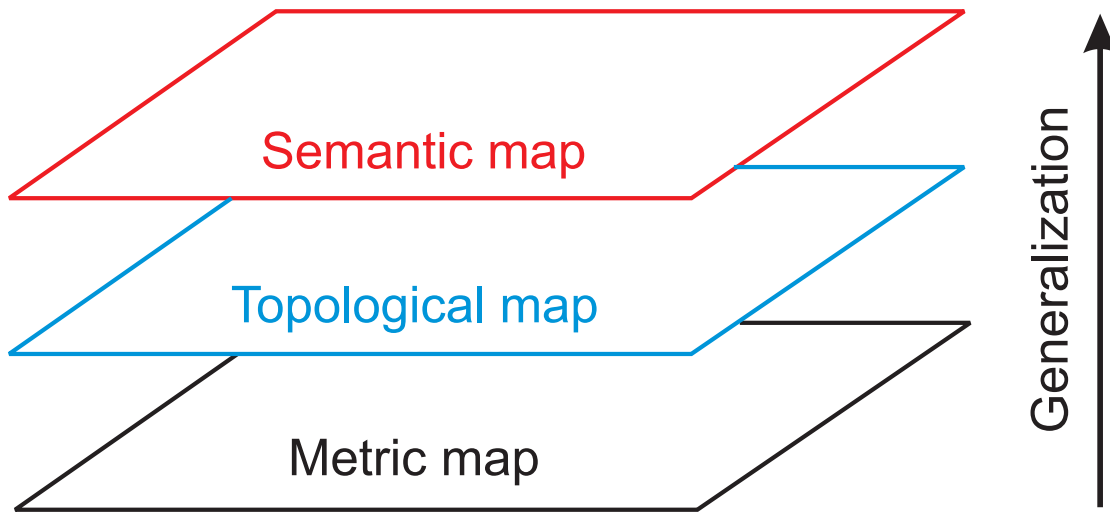
◆ Topological maps:

- The outcome generalizing the metric map. A sparse representation.
- The representation matches the problem description, e.g. it hints the robot to move between discrete locations.
- Allowed actions (e.g. a traversal) do not require the metric accuracy.
- Standard AI graph search methods are used to find a route from the start to the goal.

◆ Semantic maps:

- Introduces (measured) data interpretation and takes the advantage of a domain specific knowledge.
- A natural representation for the landmarks-based navigation.

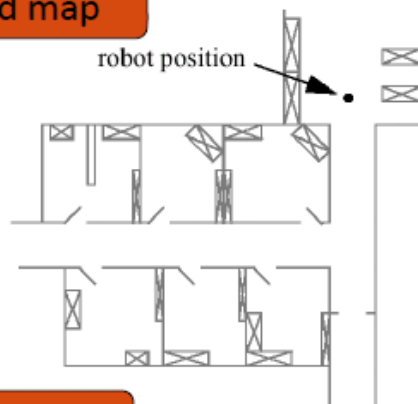
The natural hierarchy of metric, topological and semantic maps



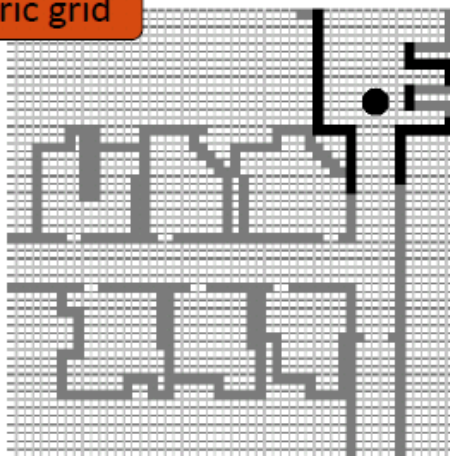
- ◆ Maps are aligned each to other. In other words, they are in correspondence.
- ◆ The hierarchy of maps implements the generalization of concepts, i.e. only the needed knowledge is retained and the less useful facts are discarded at a particular level.
- ◆ If all maps are kept, it allows to look into more detailed data if needed.

Robot world maps illustrated pictorially 1

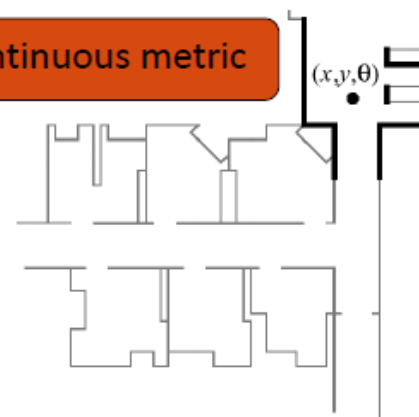
world map



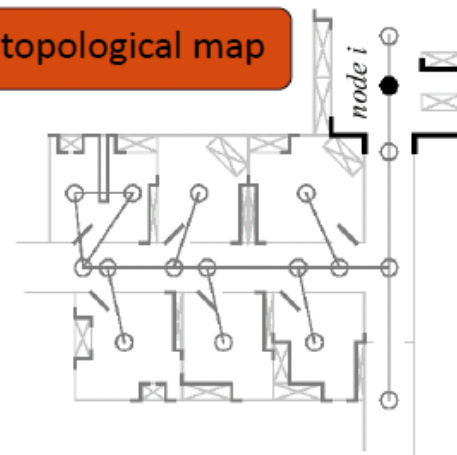
metric grid



continuous metric

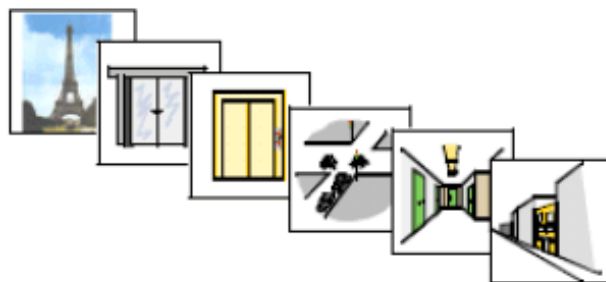


topological map

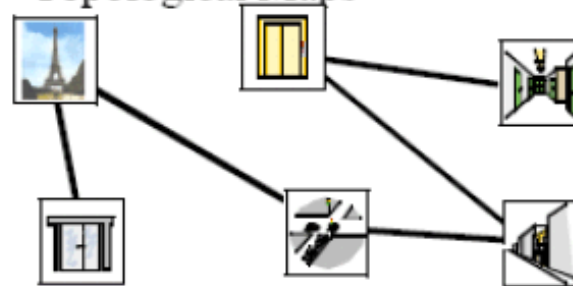


Robot world maps illustrated pictorially 2

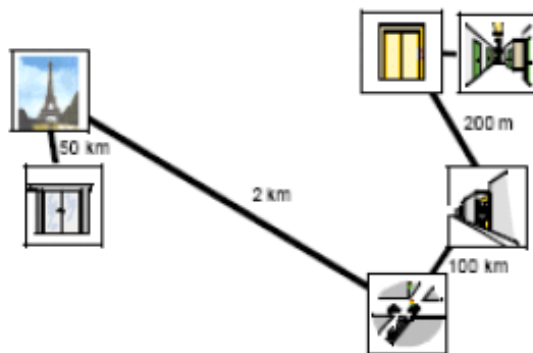
- Recognizable Locations



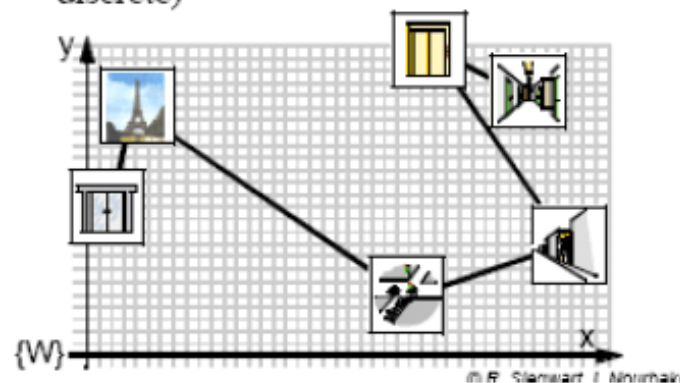
- Topological Maps



- Metric Topological Maps



- Fully Metric Maps (continuous or discrete)



Other representations used in robotics

- ◆ Robot itself

Proprioception, self-limitations, goals, sensors, intentions, plans.

- ◆ Environment

Navigable spaces, structures, maps.

- ◆ Objects

People, doors, other robots, detectable things in the world.

- ◆ Actions

Outcomes of specific actions in the environment.

- ◆ Tasks

What needs to be done, where, in what order, how fast.

Adding a timeline to a robotic representation

- ◆ Keeping a model updated requires sensing, computation and memory.
- ◆ Some models take a long time to construct and may be kept around for the lifetime of the robot task (e.g. detailed metric maps).
- ◆ Some models may be quickly constructed and soon discarded (e.g., the odometric path).