IU

Mobile Robotics

DLBROESR01_E

Dr. Florian Simroth

If you do not wish to be credited, please indicate it below:

_____

Learning Objectives

The initial reason computers were created was to assist humanity in the calculation of data. The idea was simple: computers will perform these tasks with greater certainty and speed than a human being. A computer does not have a human being's propensity to become tired or bored. It does not make simple mathematical errors, no matter how many times it executes the same equation – that is, as long as the computer is programmed and designed correctly.

You will begin your **Mobile Robotics** by looking at the most basic concept of computing: data. Interpreting numerical data as an abstract human concept is the goal of man-to-machine cooperation. Data are processed by hardware, firmware, and software working together.

To create a computing machine, it is not only essential to assemble the parts, but also to program them. This programming involves a complex language that was created to be understood by humans and machines alike – you will be introduced to the concepts of programming languages, data types, syntax, and semantics.

From hardware to advanced software, computer logic and structure permeate the essence of computer science. You will encounter algorithms, data structure, circuit design, propositional logic, and basic computer architecture. The invention of networks revolutionized computing, and we will be examining these concepts along with TCP/IP, fault tolerance, and network topologies.

This overview of data, hardware, software, programming, logic, and networking will prepare you, the student, with a foundational knowledge of the vocabulary and concepts necessary for studying the discipline of computer science.

Unit 1 – Locomotion

Study Goals

On completion of this unit, you will be able to …

… define locomotion.

… decide if a mobile robot is governed by holonomic of non-holonomic constraints.

… recognize different concepts for locomotion and their advantages.

… distinguish different mobility parameters.

Unit 1 – Locomotion

Introduction

"A mobile robot can be defined as a mechanical system capable of moving in its environment in an autonomous manner" (Jaulin, 2019, p. IX).

The field of mobile robotics involves a variety of disciplines including mechatronic design, control systems, signal processing, robot vision, operation, and many more. The spectrum of tasks for which mobile robots are constructed is broad: Within the logistics industry mobile robots are particularly useful for transporting goods in an autonomous way. On the other, flying robots are involved in search and rescue missions, where humans would be exposed to great dangers, and support rescue forces for instance with aerial reconnaissance. Even in space, such as the Curiosity mars rover, mobile robots are present and often need to carve their way through rough and uneven terrain, millions of kilometers away from earth.

In order to achieve this, they must be equipped with a broad range of sensors, to perceive and process their environment. To this end, Lidars, Cameras, Sonars, GPS systems, Accelerometers and many more come into action and the gathered data needs to be filtered and fused together. To make use of this information, algorithms need to be put in place and the necessary computational capacities must be available – all this taking into account, that energy is a limited resource for a mobile robot as it is often supplied by means of batteries or a finite amount of fuel.

After all, mobile robots obviously need to move around for which reason they require actuators to be controlled and operated. As mobile robots are deployed not only on smooth terrain, but also in water or air, their means of motion must be as diverse as their areas of application. To this end, scientists and engineers have come up with most different modes of **locomotion**. Robot locomotion is the process of how a robot is moving itself from one place to

Locomotion

Moving or transporting something from one place to another.

another. Types of locomotion can be rolling, hopping, flying, swimming, walking, or even a snake-like slithering motion. Some common types of robot locomotion will be covered in the upcoming sections.

1.1 Basics

During design phase, an adequate type of locomotion must be chosen, for the robot to be able to fulfill its goals within the task space. Some of the typical means for mobility are:

- Legs: Like humas, animals, or insects, robots can be equipped with legs. While control and construction are more sophisticated, legs usually make untreated terrain accessible, that could not be passed with wheels.
- Tracks: Tracks are usually designed for rough terrain and can handle inclinations quite well. Though, from an efficiency point of view they are still more energy demanding than wheels.
- Wheels: Wheeled robots are widely in use, as they allow for a simple construction and control. Usually designed for treated environments, there are also exist types of wheels designed for rougher terrain.
- Aerial: Aerial robots require close attention to light weight construction and power supply. Furthermore, usually more degrees of freedom than ground-based robots need to be considered. On the upside, they are mostly independent of the local terrain and can reach high velocities.

A fundamental property to understand is the concept of holonomic and non-holonomic locomotion, which has profound impact on the underlying mathematical derivations of equations of motion.

Pose

Holistic term for the position and orientation of an object in space.

Holonomic and Non-Holonomic Locomotion

In the most general sense, the **pose**, i.e., the position and orientation, of an object requires six independent coordinates to be uniquely defined. By adding constraints, this number is reduced. For a mobile platform moving on a ground plane, for

instance, three coordinates (x, y, φ) would be sufficient to uniquely describe its position and orientation within that plane. Simply put, if the mobile platform is able to directly move to any locally "neighboring" pose under consideration of its type of locomotion, the locomotion is referred to as holonomic and otherwise as non-holonomic. For instance, a mobile robot with tracks, such as a tank, is only able to move straight and to rotate about its vertical axis. Since it cannot move sideways, it is unable to directly reach any pose to its side and its locomotion is considered as non-holonomic. From a kinematics point of view, there exist some implicit constraints which do not only depend on position coordinates alone, but also on the velocity coordinates. As these constraints cannot be integrate, special care is required, when deriving the equations of motion.

Self-Check Questions

1. Which kind of systems does a car resemble – holonomic or non-holonomic?

Non-holonomic.

2. What does the term "pose" refer to?

Position and orientation of an object in space.

1.2 Legged Mobile Robots

Legged mobile robots have limbs that enable them to move around in unstructured environment making them particularly suitable to terrains that can't be passed by wheeled robots easily. Yet, they are usually more complex to design and control compared to wheeled robots and often involve rather advanced control algorithms for achieving a stable and efficient locomotion. Depending on the number of legs, these robots are referred to as bipedal, quadrupedal, or hexapods for two, four, or six legs respectively. One key aspect of this type of locomotion is **gait**, which can be defined as follows:

Gait

Motion pattern of legs during locomotion

"The sequence and way of placing and lifting each foot (in time and space), synchronized with the body motion so as to move from one place to another, is called gait" (Tzafestas, 2014, p. 10).

There are several properties particularly inherent to the characteristics of legged robots that will be covered in the following subsections.

Static and Dynamic Gait

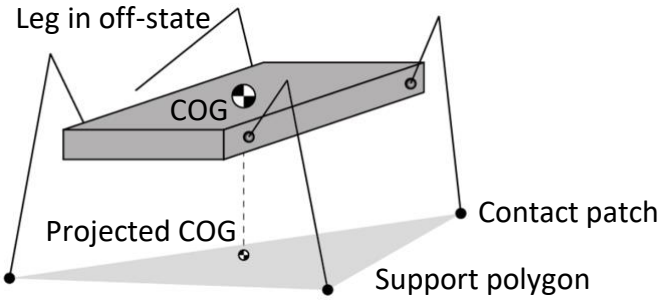
The locomotion of legged mobile robots can be distinguished between static and dynamic gait. Static gait is present, if the motion of the legged robot could be "frozen" at any time, with the robot being in balance meaning that it would maintain its posture without falling such as the walking forest harvester shown in [FIGURE 1a]. This would not be true for a robot with dynamic gait – here the dynamics play an important role, and momentums need to be considered to achieve a consistent locomotion. An impressive example for dynamic gait is the biped Atlas robot from the company Boston Dynamics, shown in the figure below.

Hexapod Walking Forest Harvester With Static Gait (a) and a Biped Robot Atlas from Boston Dynamics with Dynamic Gait (b)

Source: (a) Florian Simroth (2023), based on hackschnitzelharvester.de (n.d.). [currently copyrighted]. (b) Florian Simroth (2023), based on Boston Dynamics (n.d.). [currently copyrighted].

A simple criterium for static gait is shown in figure 2. The shape of contact points of the legs currently touching the ground form a polygon. If at any time during the locomotion, the projection of the robot's center of gravity (COG) onto the ground plane lies within the spanned support polygon, the robot will not tip over, and the gait is considered to be static.

Static Gait Illustrated For A Quadruped Robot With Projected Center Of Gravity Within Support Polygon



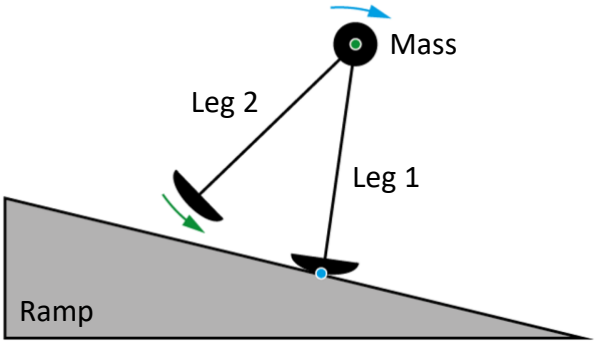
Source: Florian Simroth (2023), based on Siciliano et al. (2016, p. 437).

This model can directly be adapted to robots with four and more legs. But also biped robots can demonstrate static gait if each leg caps off with some sort of platform, effectively expanding a contact point into a contact patch that forms the support polygon. In the same way as acrobats are able to perform complex figures at very slow speeds by balancing out their center of gravity, for instance, biped robots with static gait need to add non-contributing counter-balancing moves to their gait, to assure static equilibrium at any time. This overhead of movements makes this type of gait usually less efficient than its dynamic counterpart. Also, accelerations and therefore maximum movement speed needs to be limited, to ensure that dynamic effects do not prevail.

Passive and Active Gait

It is worth mentioning, that there exist walking machines that perform a walking locomotion passively, i.e., without any active actuators. The motion is intrinsic to the mechanism and is exhibited, when placed on an inclined plane, whose principle in its simplest form is shown in the figure below. While one leg has contact to the ground, it acts as a pivot about which the lumped mass rotates. In the meantime, the other leg swings around the center mass. The mechanism tips to the front until the “swing-leg” touches ground again and then takes over the role of the pivot. The energy for motion originates from the potential energy slowly declining while walking downwards the slope whereby the mechanism needs to be designed in a way such that an equal amount of energy is dissipated during the gait to keep constant speed.

Simplified Passive Dynamic Gait Model (a) and Mechanical Passive Biped Walker (b)



Source: (a) Florian Simroth (2023), based on Siciliano et al. (2016, p. 428). (b) Florian Simroth (2023), based on Collins et al. (2001) [currently copyrighted].

Advantages according to McGeer (1990) are the high efficiency of the gait, that no active control required for the gait, and that the mechanism itself forms a subject to better understand gait and its parameters.

Modeling Gait

In order to design controllers and to describe the motion of a legged robot, a substitution model can be used to model the fundamental mechanics. Depending on the mechanics of the robot and the degree of detail required, these models can become quite complex. In [FIGURE 2], the linear inverted pendulum (LIP) model is shown as the simplest representative to model the forward motion (x-direction) of a biped robot using just a single lumped mass m and the leg's contact point p. The three equations for equilibrium of forces and moments are:

$$m\ddot{x} = F_x$$

$$m\ddot{z} + mg = F_z$$

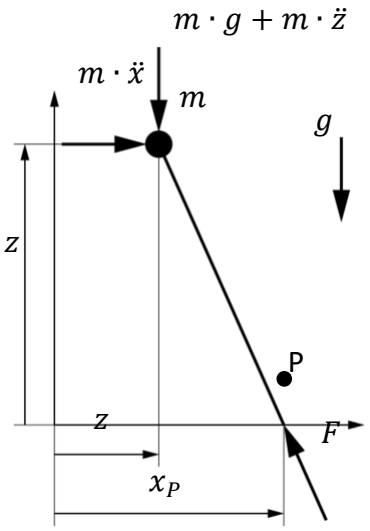
$$F_x\, z = F_z(x_P - x)$$

Inserting the first two equations into the third and assuming neglectable movements in z-direction yields the dynamics into forward x-direction:

$$\ddot{x} = \frac{g}{z}(x_P - x)$$

This equation corresponds to a linearized inverted pendulum, thus the name for the model. Since it assumes contact with ground, only walking can be modeled – for running, the spring-loaded inverted pendulum (SLIP) model as presented in Blickhan (1989) can be used instead.

Linear Inverted Pendulum Model For Describing Gait



Source: Florian Simroth (2023), based on Siciliano et al. (2016, p. 430).

Terms and Properties of Legged Locomotion

When exploring the field of legged locomotion, some gait-specific terms and properties will be encountered. As such, the on-state and off-state describe whether a leg is in contact with ground or not. The stride refers to a complete cycle from lifting a leg over off- and on-state until lifting it again. Consequently, the cycle time is the time it takes for one full stride. The support period is the part of a stride in which the leg is in on-state. Furthermore, walking and running are differentiated: during

walking at least one leg is always in contact with ground, while there exist periods where all legs are airborne during running. To compare different types of legged locomotion or different robots with the same type of legged locomotion, some properties were introduced of which two are explained below:

Duty factor

The duty factor β is a dimensionless measure for the fraction the leg is in on-state, i.e., has contact with the ground, and the cycle time as in the equation below.

$$\beta = \frac{support\ period}{cycle\ time}$$

Form this simple relationship, several conclusions can be drawn. For instance, walking is present for $\beta \geq 0.5$ (Alexander, 1984) such that at least one leg is in ground contact at any time. Also, to achieve static gait for robots with more than three legs, $\beta \geq \frac{3}{number\ of\ legs}$ (Tzafestas, 2014, p. 11).

Specific resistance

The specific resistance ϵ is the dimensionless ratio of the total resistance and weight of a vehicle according to [EQUATION] (Gabrielli & Karman, 1950).

$$\epsilon = \frac{P}{W\ V}$$

Here, P is the power of required propulsion to move the vehicle with weight $W = m \cdot g$ corresponding to mass times gravitational acceleration, with velocity V. When integrating the right hand side with respect to time, the specific resistance can also be seen as the ratio of energy required to move a vehicle over a certain distance. This gives information about how efficiently a certain locomotion economizes the energy of a mobile robot.

Self-Check Questions

3. Determine the minimum duty factor for static gate for a three-legged robot – what does the result imply?

Since the duty factor $\beta = 1$, the support period equals the cycle time. This result implies, that no static gate is possible.

4. Please mark the correct statement(s).

☐ *The LIP model is not applicable to running*.

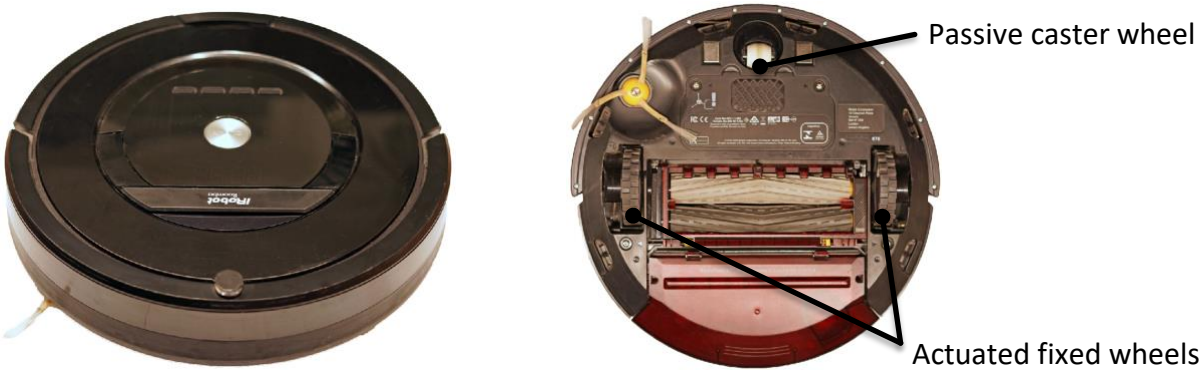
☐ The terms running and walking for gait can be arbitrarily exchanged by the terms dynamic and static gait, respectively.

☐ For both, passive gait and passive walking, the center of mass projected to ground is always within the support polygon.

1.3 Wheeled Mobile Robots

Wheeled mobile robots use wheels as their primary means of locomotion, which, depending on the terrain, allows for a smooth and relatively efficient type of locomotion. Realizations with certain types of wheels or special arrangement of wheels open up possibilities to maneuver in narrow and congested spaces. As a result, wheeled mobile robots already have wide range of applications, and are already used within warehouses for intralogistics and even find their way into many households nowadays to carry out vacuuming or sweeping tasks (see figure below).

Roomba Vacuum Cleaning Robot With Differential Drive Mode



Source: Florian Simroth (2023).

Kinds of Wheels

As wheels form the interface of wheeled mobile robots with their environment, the choice of wheel type determines, how well the robot's tasks may be achieved. Essential design decisions concerning wheels are:

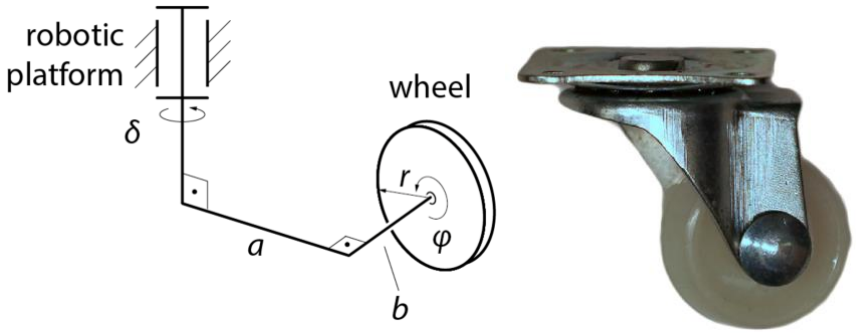
- Wheel architecture: conventional or special
- Mounting type: fixed or mobile
- Actuation: Passive or active
- Plasticity: Rigid or deformable

To understand why to choose one or the other kind of wheel, the individual aspects are discussed hereafter.

Conventional wheels

When developing a wheeled mobile robot, there exists broad range of wheel architectures to choose from. Conventional wheels are most commonly composed of a cylindrical body with radius r which is mounted with a low-friction hub or bearing at its center and a high-friction tread to provide traction on the ground surface. There are several important geometric mounting parameters that have a significant impact on the robot's kinematics when mounting a conventional wheel to the robot as shown in [FIGURE].

Geometric Mounting Parameters of a Conventional Wheel and Passive Caster Wheel with b=0



Source: Florian Simroth (2023).

Besides the wheel radius itself, offset a defines the distance of the wheel axis within the ground plain with respect to the vertical mounting axis. The offset b indicates how much the wheel is off centered. A non-zero value for the off-center distance can achieve a pure rolling contact of the wheel on the ground surface (Siciliano et al., 2016, p. 576). Though, commonly the parameter will be set to $|b| = 0$ and will be omitted hereafter. The angles δ and φ refer to the steering and the rotation angle of the wheel, respectively.

Another aspect is the actuation, i.e., the selection of which angles are actively driven. It is possible to actuate the steering angle, to achieve active steering as well as the wheel's rotational angle to generate propulsion. Resultingly, four combinations are possible:

- No actuation at all, the wheel is completely passive
- Actuated steering but passive rolling
- Actuated rolling, but passive steering
- Actuated steering and rolling

The actuation modes can be combined with either setting the wheel axis distance $a = 0$ or $a \geq 0$. From all possible combinations the following are most commonly used (Siciliano et al., 2016, p. 577):

- Completely passive castor wheel with $a \geq 0$

- Active steering and passive rolling wheel with $a = 0$
- Actively steered power wheel with $a = 0´$
- Actively steered power wheel with $a \geq 0$

Another important design choice is the plasticity of the wheel, namely, whether the wheel can be regarded as rigid or soft in relation to the ground structure it is used on. Each of the four combinations, rigid/rigid, rigid/soft, soft/rigid, and soft/soft for the wheel requires a different wheel-terrain interaction model, to appropriately predict the occurring wheel forces. This course will only be concerned with the rigid/rigid combination for which the well-known Coulomb friction force model can be used. Once the lateral and longitudinal friction coefficients μ_{lat} and μ_{long} of a wheel are known, the friction forces F_{lat} and F_{long} result from the normal force F_N.

$$F_{lat} = \mu_{lat} \cdot F_N$$

$$F_{long} = \mu_{long} \cdot F_N$$

Rigidity of the wheels can be assumed for solid material such as hard rubber or firmly inflated pneumatic tires.

Omnidirectional wheels

Besides conventional wheels, engineers came up with special wheel architectures which allow for better maneuverability. This is often achieved by allowing passive motion in one direction, while realizing traction into another direction. One example of a ball shaped wheel is shown in [FIGURE]. It is composed of two hemispheres, which can independently rotate about a common axis. The rotation axis itself is perpendicularly mounted on a drive shaft, which rotates both hemispheres simultaneously.

Ball-Shaped Omnidirectional Wheel

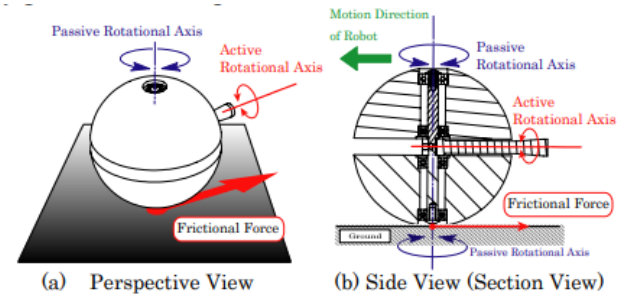


Fig.7: Section Views of "Omni-Ball"

Source: Florian Simroth (2023), based on Tadakuma et al. (2007). [currently coprighted]

Another common type is the Mecanum or Swedish wheel, which was developed by the Swede Bengt Ilon in 1973 (Diegel et al., 2002). This type of wheel consists of many small rollers that are evenly distributed among along the circumference of the wheel. The rollers' axes are angled with respect to the wheel axis. Common designs realize either an angle of 45° as shown in [FIGURE], or a 90° angle with the wheel axis. The rollers are capable of passively rotating, while the wheel itself is usually actively driven about the wheel axis.

Mecanum Wheel With Passive Rollers Mounted At 45° Degree Angles

Source: Florian Simroth (2023).

Usually, one or two of the rollers are in contact with ground simultaneously, and the contact patch shifts from one end of the roller to the other during the rotation. The overall motion is comparable to that of a screw with the rollers replacing the thread. For a detailed derivation of the kinematics and dynamics of the Mecanum wheel, see Tlale et al. (2008).

While offering more versatility, the mechanical design is more cumbersome and expensive than conventional wheels. Generally, Mecanum wheels perform well in indoor environments with plane concrete floors – even minor landings give raise to problems, as the small rollers are not capable to overcome small steps. Also, with more moving parts involved, maintenance costs increase, and the design is bulkier when higher load capacity are to be realized. Another aspect are noise and vibrations induced by the roller impingements with ground.

Drive Modes

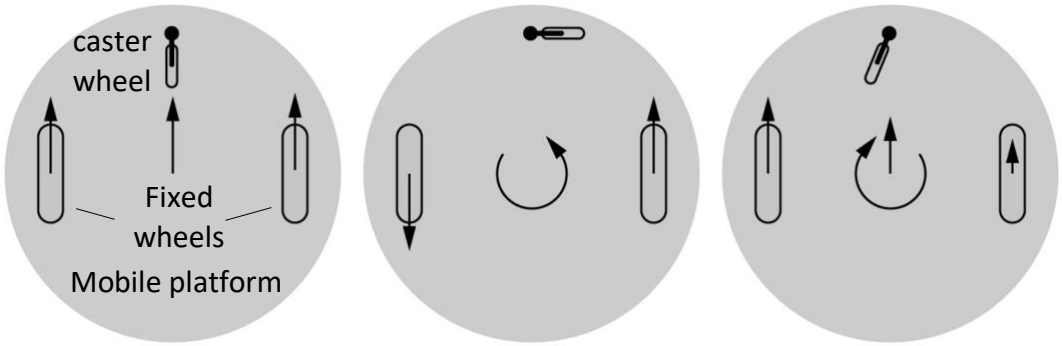
There are endless combinations of different wheel types, mounting positions, and actuation modes imaginable. Depending on the use-case, some of these configurations are more suitable than others and it is the task of the engineer to select an appropriate one. To get an overview of the different concepts and commonly used combinations of actuated conventional and omnidirectional wheels, some fundamental drive modes are described.

Differential drive

The differential drive mode of robots distinguishes itself by its simplicity. In its simplest statically determined form, a mobile platform with differential drive consists of two fixed actuated conventional wheels with coinciding axes and one passive conventional caster wheel with a non-zero offset. This design eases the mechanical effort, since both actuated wheels are fixed, allowing a cost-effective and simple realization.

If both power wheels are driven with an equal angular velocity in the same direction, a straight motion of the mobile platform results. On the other hand, if both power wheels are driven with an equal angular velocity but in opposite directions, a pure rotation about the midpoint of the common rotation axis between the power wheels is caused. In the general case, where the angular velocities are different for both powered wheels, the robot performs a curve with a radius depending on the difference of velocities. The drive modes are illustrated in [FIGURE].

Drive Modes of a Differential Drive Mobile Platform



Straight motion Pure rotation General motion

Source: Florian Simroth (2023).

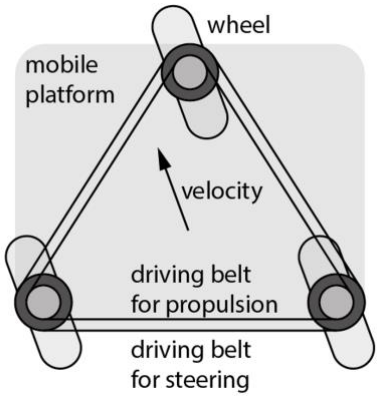
Another quite familiar representative for differential drive platforms, is a tank-like robot with tracks on both sides, also referred to as skid-steer robot. The drive modes are equal to the ones presented above, though, much more friction is involved due to the large contact area of the tracks.

A downside of wheeled differential drive platform is a certain limitation in regard to maximum velocity as stability decreases at high speeds. Also, uneven terrain can be challenging, as the angular velocity of the platform is directly impacted, if one of the powered wheels loses traction.

Synchronous drive

Some designs make the synchronous actuation of multiple wheels necessary. A simple example of a mobile platform with three powered wheels with active steering is shown in [FIGURE]. Kinematically, this platform is only able to move, if all wheel axes are parallel as otherwise, the resulting forces would counteract resulting in friction losses. In the shown example, also the angular velocities of the wheels are equal. From a design point of view, this is realized by timing belts, such that it is possible to drive the whole platform with only two motors.

Mobile Platform With Three Synchronously Powered and Actively Steered Wheels



Source: Florian Simroth (2023).

The platform can move into any direction, by steering the wheels accordingly. Though, only translational motions can be produced, such that the platform cannot change its orientation. Furthermore, the design of the powered steerable wheels is mechanically complex resulting in higher costs and tendentially more maintenance.

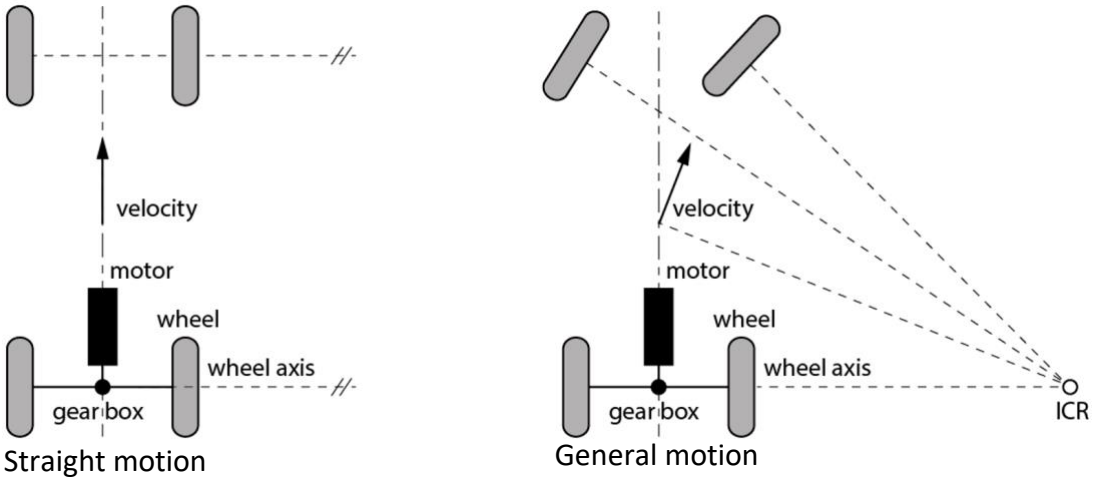
Car-like drive

A car-like configuration of a mobile robot is well-known from everyday life. A quite familiar design consists of two powered fixed wheels with coaxial wheel axes in the

rear and two actively steered front wheels. The rear wheels are commonly powered by a single motor coupled via a differential drive, allowing for different wheel speeds of inner and outer wheel in turns. Car-like configurations excel in the maximum velocities that can be reached while maintaining stable driving behavior.

There are two drive modes available, straight driving and performing turns with a curve radius larger than zero. It is neither possible to perform a pure rotation, nor to move sideways. This results in a lower maneuverability which makes this design rather unsuitable for tight and congested spaces.

Schematic of Straight Driving (Left), and a General Turn With Wheel Axes Intersecting at The Instantaneous Center of Rotation (Right)



Source: Florian Simroth (2023).

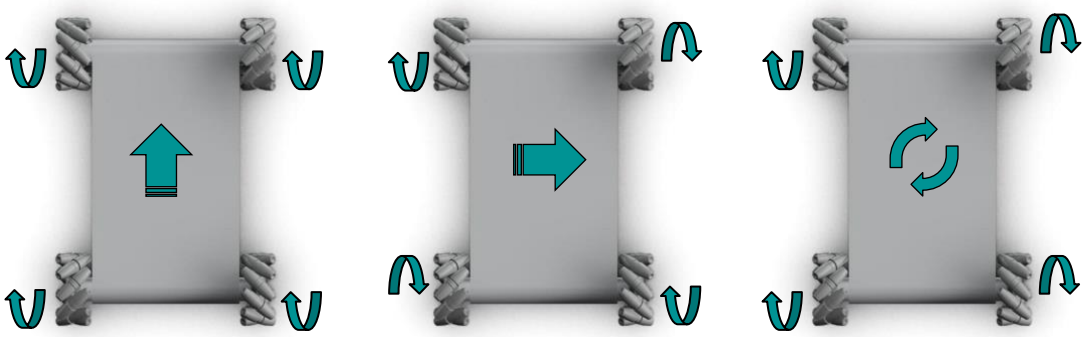
The kinematics of steering are often referred to as Ackermann steering, named after Rudolph Ackermann, who received the patent no. 4212 in 1808, even though the concept was already invented by Erasmus Darwin 50 years earlier (King-Hele, 2002). For a kinematically compliant motion, all wheel axles must intersect in a single point, the instantaneous center of rotation (ICR), for a general motion or must be parallel

for a purely straight motion as shown in [FIGURE]. To this end, the steering angles of the front wheels differ to meet this condition.

Omni-directional drive

The omni-directional drive mode allows a mobile platform to traverse into any direction from any point. This can be achieved with special wheel architectures, such as the Mecanum wheel, and even conventional wheels can be used to achieve this type of mobility as shown in Wada et al. (1996).

Basic Drive Modes Of A Platform With Four Mecanum Wheels



Straight forward motion Straight sidewards motion Pure rotation

Source: Florian Simroth (2023).

A realization of a mobile platform using four Mecanum wheels is shown in [FIGURE], which enables omnidirectional drive from any pose by different actuation of the wheels. Note, that there are two types of wheels mounted, namely two +45° and one with -45° angled rollers. All types of motion such as lateral translation, longitudinal translation, pure rotation, or a mixture of all can be introduced by certain combinations of wheel speeds and directions resulting in holonomic constraints. It is also worth mentioning, that it is also possible to introduce a pure rotation with less than four wheels, while the remaining wheels idle. The drive modes shown in [FIGURE] are just an excerpt of possible motion types as any other motion including turns with an ICR outside of the platform can be induced.

Another example to achieve omni-directional motion is to arrange three Swedish wheels with 90° angled rollers in a circular configuration. Rotating all wheels in the same direction will result in a purely rotational motion of the robot. Actuating two wheels in opposite direction will yield a translational motion, as the contradicting components of the frictional forces arising at the tires cancel each other out.

In general, the presented omni-directional robots with special wheel types require a plane surface to run on and therefore are well-suited for indoor tasks, such as intralogistics or present their advantages particularly in tight spaces. The downsides are vibration, lower efficiency due to increased friction in rollers, as well as increased maintenance and acquisition costs in consequence of the more demanding mechanical design.

Mobility Parameters of Wheeled Mobile Robots

There are some important properties of mobile platforms that give information on the kind of motion to be expected for a certain configuration. These parameters also give rise to whether some wheels need to be controlled in a coordinated way for the mobile platform to be able to move. A detailed derivation and mathematical description are provided by Siciliano et al. (2016) which the interested reader is encouraged to read and from which the following concepts are summarized hereafter.

The **degree of freedom** (DOF) of the rigid mobile platform corresponds to minimum number of independent coordinates necessary, to distinctly describe pose of the platform within the designated workspace. Assuming a planar worksp and that the mobile robot does not lift, the pose of a single mobile platform can described with two positional coordinates x and y as well as one rotatio coordinate φ. The degree of freedom, so to speak, applies to the position level.

Degree of freedom (DOF)

The minimum number of of independent coordinates to required to distinctly describe the configuration of a mobile robot.

Another parameter is the degree of mobility δ_m. It is equal to the degree of freedom of the robot within the workspace, reduced by the number of independent constraints:

$$\delta_m = DOF\ of\ unconstrained\ robot - \#\ of\ independent\ constriants$$

Depending on the type of wheel attached, the motion of the platform is constrained due to additional nonslip conditions introduced by the wheel. That said, adding a fixed wheel to the platform adds one constraint, as movements lateral to the wheel are blocked. A passive caster wheel on the other side does not add any constraints, as it does not limit any direction of motion. The question, whether two constraints are dependent is that easy to answer. Theoretically, adding three fixed wheels randomly distributed to a platform would introduce three constraints, resulting in a degree of mobility of zero, i.e., the platform would not be able to move. Though, if the wheels are aligned in a way, that all wheel axes intersect in one point or are parallel, the robot would be able to perform a pure rotational or pure translational motion respectively since one of the introduced constraints becomes redundant. The degree of mobility can be thought of as the degree of freedom on velocity level, in the sense of independent velocity coordinates.

The degree of steerability δ_s is equal to the number of independently controllable steering wheels, whereby independent means that the steering wheels can be rotated to any orientation without inhibiting a motion direction. For example, a car has to steering wheels but their orientation must be coordinated according to the Ackermann kinematics to ensure that wheel axes intersect at a common ICR.

In fact, if only practically useful robots are of concern, i.e., omitting those with no or only one motion direction, wheeled robots can be categorized into five distinct groups depending on their degree of mobility and steerability:

Five Categories Of Wheeled Mobile Robots - Each Column Represents One Group

δ_m	3	2	2	1	1
δ_s	0	0	1	1	2

Source: Florian Simroth (2023), based on Siciliano et al. (2016, p. 581).

From this table, some interesting conclusions can be drawn. For instance, only robots within the first group are holonomic, as they can move into any direction at any point. Omni-directional robots from group (1,2) can move into any direction as well but are non-holonomic as they need to re-orient steering wheels first to do so.

Self-Check Questions

5. Which type of motion are car-like robots related to: Holonomic or Non-holonomic?

Non-holonomic.

6. Complete the following sentence:

   The degree of freedom of a non-holonomic robot is larger than the degree of mobility.

1.4 Aerial Mobile Robots

This section is concerned with aerial mobile robots in the sense of unmanned aerial systems (UAS). In contrast to the previously mentioned mobile robots, aerial robots are capable of flying or hovering in the air. More commonly known than UAS is the term **unmanned aerial vehicle** (UAV), which only refers to the actual flying machine itself. UAS also enclose human machine interfaces or other systems required for operation that don't necessarily need to be part of the UAV and could also be positioned on ground.

Unmanned Aerial Vehicle (UAV)
UAV is the umbrella term for aerial mobile robots that only includes the flying robot itself.

Types of Aerial Robots

One of many ways to classify aerial robots can be done based on the method they use to generate lift and maintain flight. The most common components to create uplift include fixed wing, rotary wing, flapping wing, and lighter-than-air types. Fixed wing aerial robots, such as airplanes, generate lift through the movement of air over their wings directly fixed to their fuselage. For rotary wing aerial robots, such as helicopters, wings are not directly attached to the fuselage – instead they generate lift through the rotation of their blades. Flapping wing aerial robots, called ornithopters, mimic, for instance, the flight of birds or insects, and generate lift through the flapping of their wings. Lighter-than-air aerial robots use the principle of buoyancy and generate lift through the use of a gas that is less dense than the surrounding air, for which blimps are an example.

Autonomy of Aerial Robots

Unmanned aircraft systems come in different autonomy levels, including manual, semi-autonomous, and autonomous flight mode (Siciliano, 2016, p. 659). Manual flight requires direct human control and input, while semi-autonomous flight mode allows the UAS to make some decisions on its own while still requiring human oversight. Autonomous mode allows the UAS to make decisions and complete tasks without human input or oversight. This being said, even though the terms "drone" and "aerial robot" are often used interchangeably, one can discriminate some subtle differences between the two:

A drone is typically controlled remotely by a human operator but can still incorporate a certain level of perception of its local surrounding (Siciliano, 2016, p. 662). To that end it can be equipped with visual sensors to avoid collision with objects, the operator for instance, might have overseen or which were not considered in the general flight plan. Drones, such as the Skydio 2 or Phantom 4, are already an inherent part of the consumer market, used for video- and photography. Yet, drones are also used for military purposes, and, as such, the MQ-9 Reaper is an example for

a drone used for reconnaissance and surveillance, which can also be equipped with missiles.

An aerial robot, on the other hand, is equipped with variety of sensors allowing for a better perception of the environment. Coupled with advanced algorithms for control and robot vision, high levels of autonomy are achievable (Siciliano, 2016, p. 662). They are designed to complete complex tasks without human intervention. These can serve as autonomous air taxis, as the concept of the EHang 184 shows, or for the inspection of power lines.

The level of autonomy required will depend on the specific application and mission of the UAS.

Areas of Application for Aerial Robots

The range of feasible applications has expanded over the last years, with one reason certainly being the technology advances in lithium batteries with a better energy to weight ratio. Areas of application for unmanned aircraft systems include for example:

- **Reconnaissance** involves the gathering of information about an area or target. For these purposes often cameras and other sensors are being used.
- **Surveillance** is the act of continuously monitoring and observing an area or target.
- **Search and rescue** utilize UAS to get through to hard-to-reach areas for example in disaster regions, and to support emergency workers to get a faster overview of the scene to accurately plan rescue missions.
- **Transportation** generally does not only refer to the transportation of goods but also to the transportation of people via UAS.
- **Payload delivery** is concerned with the delivery of goods or materials to a specific location for which a trial was initiated for example through the Amazon Prime Air program.

Overall, aerial mobile robots have a wide range of applications and are used in many different fields and industries and are used in military, civilian, and commercial domains.

Self-Check Questions

7. What is the difference between a UAS and a UAV?

The unmanned aerial system encloses the unmanned aerial vehicle and supplementary components such as human-machine interfaces or ground stations.

8. Complete the following sentence:
   Aerial robots can be classified into *fixed wing*, *rotary wing*, *flapping wing*, and *lighter-than-air types* based on their method of generating lift.

Summary

Locomotion is the act of moving or transporting something from one place to another. Mobile robots realize different types of locomotion through legs, tracks, wheels, or means to fly.

Legged mobile robots can exhibit static gait which is a type of locomotion that maintains static equilibrium during at any time, or dynamic gait. Also, passive and active gait can be distinguished, whereby passive gait is inherently imposed on the walking mechanics of a robot and as such does not require any actuators to perform the motion. As a simple mathematical model for gait, the linear inverted pendulum model can be used. Legged mobile robots can be put in comparison using indicators such as the duty factor or specific resistance.

Wheeled mobile robots are equipped with conventional wheels or special wheels such as Swedish wheels to perform smooth motions. Depending on the choice of wheels and wheel arrangement, differential, synchronous, car-like, or omni-directional drive modes can be established. Based on the mobility parameters of each

configuration, statements on the expected motion behavior including whether the robot is governed by holonomic or non-holonomic constraints can be derived.

Aerial robots, or more generally unmanned aerial systems, have the ability to fly through the air. Different means for flying are fixed wings, rotating blades, flapping wings, or lighter-than-air designs. UAS can be either manually controlled, semi-autonomous, that is robots equipped with logic to maneuver in their immediate surrounding for obstacle avoidance, and fully autonomous. Typical areas of application are reconnaissance, surveillance, search and rescue, transportation, and payload delivery.

Unit 2 – Kinematics

Study Goals

On completion of this unit, you will be able to …

… define kinematic constraints of a wheeled mobile robot.

… decide which configuration best suits the desired tasks.

… identify the workspace of a given robot configuration.

… apply knowledge to derive kinematics for certain types of robotic platforms.

Unit 2 – Kinematics

Introduction

In kinematics, the motion of robots is analyzed under the absence of any forces. It is one subarea of general mechanics next to dynamics as shown in [FIGURE] which is covered in another unit.

Segmentation of Mechanics Into Dynamics and Kinematics

Source: Florian Simroth (2023).

The motion analysis aims at the positions, velocities, and accelerations of a mobile robot and its components, that are achievable in compliance with the present constraints. The constraints are induced by joints or wheel-ground contact assuming non-slip and pure rolling conditions.

It is essential to be able to predict the motions of a robot with respect to actuated inputs to assure that the robot is capable to accomplish its tasks and to design corresponding controllers for smooth motion.

This chapter covers the derivation of constraints and resulting equations of motion, to describe the kinematics of mobile robots. To reduce the complexity to a reasonable level, only wheeled mobile robots are of concern, as the most present form of mobile robots.

2.1 Basics

In order to give meaning to a location in physical space, a space-fixed coordinate frame is introduced. A Cartesian coordinate frame $\mathcal{K}$ is represented by an origin $\mathcal{O}$ and three axes $\mathbf{e}_x$, $\mathbf{e}_y$, $\mathbf{e}_z$, that are perpendicular to one another.

Any position in space can then be described by a vector $\mathbf{r}$ composed of a triplet of numbers, each of which representing the components along these three axes.

$$\mathbf{r} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = x \cdot \underbrace{\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}}_{\mathbf{e}_x} + y \cdot \underbrace{\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}}_{\mathbf{e}_y} + z \cdot \underbrace{\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}}_{\mathbf{e}_z}$$

In this course, mobile robots are of concern which are composed of rigid bodies only. That is, the shape and dimension of each body is considered to be constant over time and independent of the applied load. Therefore, if a body-fixed coordinate frame is attached at one location of a rigid body, the location of any mass particle of the body can be described with respect to the body-fixed coordinate frame. If the position and orientation, namely the pose, of the body-fixed coordinate frame is known with respect to a space-fixed coordinate frame rigidly attached to the surrounding environment, the pose of the whole body is defined (Siciliano et al., 2016, p. 12).

The relative spatial orientation of two coordinate frames $\mathcal{K}_i$ and $\mathcal{K}_j$ can be described by a 3×3 rotation matrix ${}^{i}\mathbf{R}_j$. The columns of the rotation matrix ${}^{i}\mathbf{R}_j$ can be interpreted as the unit vectors of the axes of $\mathcal{K}_j$ decomposed in coordinates of $\mathcal{K}_i$.

$${}^{i}\mathbf{R}_j = \left({}^{i}\mathbf{e}_{x_j} \, {}^{i}\mathbf{e}_{y_j} \, {}^{i}\mathbf{e}_{z_j} \right)$$

The nine elements of the rotation matrix are not independent. The rotation matrix fulfills the requirement, that all column vectors are of unit length and are mutually orthogonal from which six constraints result. Additionally, the determinant equals

$+1$ which can be achieved by rearranging the column vectors accordingly. One helpful property that results from the orthogonality statement is that the inverse of a rotation matrix is equal to its transpose, i.e., ${}^{j}\mathbf{R}_i = \left({}^{i}\mathbf{R}_j\right)^{-1} = \left({}^{i}\mathbf{R}_j\right)^{T}$. From the orthogonality and unit length requirements, six constraints emerge which have to be fulfilled by the nine elements, and only three independent parameters are necessary to describe relative orientation of two coordinate frames.

This becomes even clearer if one recalls that any three-dimensional rotation can be expressed by subsequent multiplication of three elementary rotations, about non-coinciding axes, whereby the order of multiplication is non-commutative. Each elementary rotation is a rotation by an angle about one of the coordinate axes and can be expressed as follows. Once the relative orientation of two different coordinate frames is known, vectors can be transformed: A vector ${}^{j}\mathbf{r}$ which is decomposed in coordinate frame $\mathcal{K}_j$ can be expressed in coordinates of frame $\mathcal{K}_i$ and vice versa by the following relation:

$$ {}^{i}\mathbf{r} = {}^{i}\mathbf{R}_j \cdot {}^{j}\mathbf{r} \qquad {}^{j}\mathbf{r} = \left({}^{i}\mathbf{R}_j\right)^{T} \cdot {}^{i}\mathbf{r} = {}^{j}\mathbf{R}_i \cdot {}^{i}\mathbf{r} $$

This change of frame decomposition is referred to as the passive interpretation - an active interpretation would indicate, that the frame of decomposition remains the same, but the vector is actively rotated by the rotation matrix.

In conclusion, there are six independent parameters necessary, to describe the absolute position and orientation of a rigid body in three-dimensional space, three for the position and three for the orientation. One way of how such a rigid body displacement can be expressed is by means of homogeneous coordinates. A general displacement can then be described by a single 4×4 transformation matrix $\mathbf{T}$ of the following form.

$$ {}^{i}\mathbf{T}_j = \begin{pmatrix} {}^{i}\mathbf{R}_j & {}^{i}\mathbf{r}_j \\ 0 & 1 \end{pmatrix} \quad \text{and} \quad {}^{j}\mathbf{T}_i = \left({}^{i}\mathbf{T}_j\right)^{-1} = \begin{pmatrix} \left({}^{i}\mathbf{R}_j\right)^{T} & -\left({}^{i}\mathbf{R}_j\right)^{T} {}^{i}\mathbf{r}_j \\ 0 & 1 \end{pmatrix} $$

Multiple transformations can then be easily be concatenated. If the transformations ${}^{i}\mathbf{T}_{j}$ and ${}^{j}\mathbf{T}_{k}$ between the coordinate frames $\mathcal{K}_i$ and $\mathcal{K}_j$ as well as $\mathcal{K}_j$ and $\mathcal{K}_k$ are known, the transformation ${}^{i}\mathbf{T}_{k}$ between $\mathcal{K}_i$ and $\mathcal{K}_k$ can be determined as ${}^{i}\mathbf{T}_{k} = {}^{i}\mathbf{T}_{j} \cdot {}^{j}\mathbf{T}_{k}$, whereby the order of general transformations is non-commutative as mentioned before. The rotational part within ${}^{i}\mathbf{T}_{k}$ describes the rotation from $\mathcal{K}_i$ to $\mathcal{K}_k$, while the translational part represents the displacement of the origin of $\mathcal{K}_k$ with respect to $\mathcal{K}_i$ in coordinates of $\mathcal{K}_i$.

Self-Check Questions

9. How many independent parameters are necessary to define the pose in 3D space?

6

10. What the columns of the rotation matrix represent?

Each column corresponds to a unit vector corresponding to the coordinate axes of the target frame decomposed in the current frame.

2.2 Kinematic Models and Constraints

Usually, a mobile robot is composed of a main platform to which wheels, legs, rotors, or manipulators are attached. While manipulators attached to mobile robots are discussed in a separate chapter, this section focuses on wheeled mobile robots (WMR), as one of the most common types of mobile robots. To reduce complexity, only planar embeddings of a wheeled mobile robots are of concern. Throughout this section, all components are assumed to be rigid, i.e., maintain their dimensions at any time. Only flat and solid ground is of concern on which the robot platform moves around in a planar motion such that roll and pitch angles as well as the height are constant over time. The wheels described hereafter are free of skidding and wheel planes are perpendicular to the ground plane. Even though suspension systems are not considered, it is assumed, that for an overconstrained mobile robot platform in

the sense of ground contact points, such as one with more than three wheels, all wheels are equally in contact with the ground plane.

The **configuration** of a mobile robot is a set of variables that allow to define the locations of all points and components of the robot. For conventional, stationary robots this is usually the set of all joint variables. For mobile robots, in addition, the pose $\boldsymbol{\xi}$ is required which in some sense can be seen as a general free joint. Of the n generalized coordinates $\mathbf{q} = [q_1, q_2, \dots, q_n]^T$ that describe the robot's configuration with respect to a reference system, only a subset of m coordinates may be independent due to the presence of constraints. The generalized velocities $\dot{\mathbf{q}} = [\dot{q}_1, \dot{q}_2, \dots, \dot{q}_n]^T$ are the time derivatives of $\mathbf{q}$. For example, typical generalized coordinates may include the robot's pose as well as steering angles. Next, some fundamental terminology is presented, before the actual constraints introduced by wheels are derived.

Configuration

The set of variables necessary to describe the poses of all components of a mechanism

Geometrical and Kinematic Constraints

Geometric constraints form limits to the configurations the robot can access. These limits solely result from constraints depending on configuration parameters but not on their derivatives. These constraints are of the form $g(\mathbf{q}, t) = 0$. Kinematic constraints operate on the velocity level by setting up restrictions to feasible velocities taking into account the current configuration and can be written as $g(\mathbf{q}, \dot{\mathbf{q}}, t) = 0$. They do not limit the achievable configuration such as the set of robot poses itself but rather the potential paths on how to get there.

Holonomic and Non-Holonomic Constraints

While geometric constraints are also holonomic, this is only the case for kinematic constraints if these can be integrated, such that they can also be expressed as $g(q, t) = 0$. If this is not the case, they are non-holonomic. A typical example for a

robot with non-holonomic constraints is a car-like assembly. Even though, any pose in the plane can be achieved eventually, the non-holonomic constraints arising at the fixed wheels do not allow direct lateral movement. Instead, a lateral displacement can be reached by maneuvering back and forth. Therefore, non-holonomic constraints do not restrict the possible configurations, but rather the paths to get there.

Forward and Inverse Kinematics

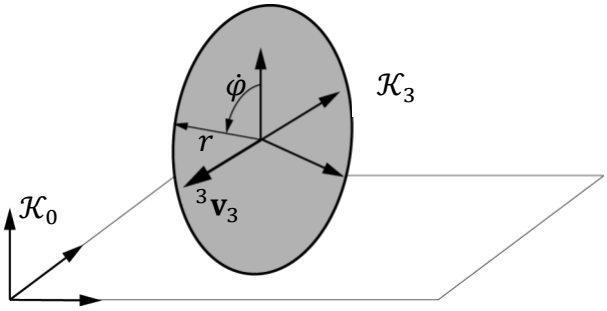
With forward kinematics, the configuration or pose of a robot is derived, based on the known control variables such as wheel velocities and steering angles. In other words, it is a mapping from the configuration space (joint variables) to the physical space with pose and velocity information of the robot. On the contrary, inverse kinematics map a desired configuration or pose to the control/joint variables to determine the values required to establish that pose.

Constraints Arising at Standard Wheels

For an idealized standard wheel shown in [FIGURE], two constraints arise:

1. **Pure rolling**: The velocity at the center of the wheel can directly be derived from its angular velocity $\dot{\varphi}$ and radius r, as slippage is assumed to be zero.
2. **Non-(lateral)-slip**: The velocity component of the wheel center normal to the wheel plane is assumed to be zero.

Wheel Model With Pure Rolling and Non-Slip Condition

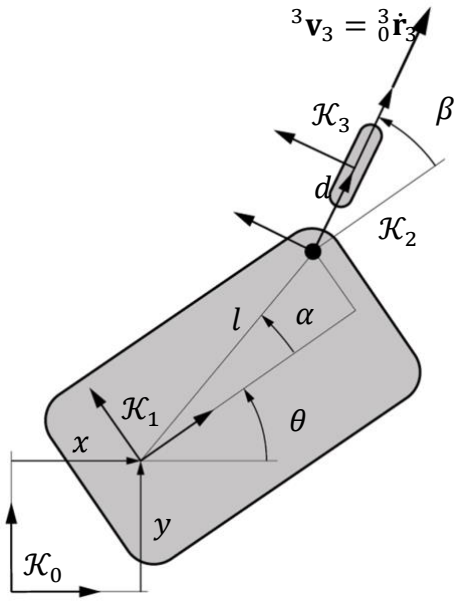


Source: Florian Simroth (2023).

These two conditions and the assumed non-lift condition can be expressed in a local "floating" coordinate frame $\mathcal{K}_3$ pinned to the wheel center with y- and z-axis being normal to the wheel and ground plane respectively. The velocity ${}^3\mathbf{v}_3$ of the center of the wheel with respect to a world-fixed frame $\mathcal{K}_0$ decomposed in $\mathcal{K}_3$ then only contains one component within the wheel plane:

Castor Wheel Attached to a Mobile Robot Platform

$$ {}^3_0\mathbf{v}_3 = \begin{pmatrix} \dot{\varphi} \cdot r \\ 0 \\ 0 \end{pmatrix} $$



Source: Florian Simroth (2023).

To derive the constraints emerging at a castor wheel as shown in [FIGURE] the same velocity is now expressed in terms of the robot's generalized coordinates. To this end, first, the absolute position vector ${}_0^3\mathbf{r}_0$ of the wheel's center is derived based on the robot's pose $\boldsymbol{\xi} = (x, y, \theta)$ and the steering angle β) by simply concatenating the transformation matrices

$$ {}^0\mathbf{T}_3 = \underbrace{\begin{pmatrix} \cos(\theta) & -\sin(\theta) & x \\ \sin(\theta) & \cos(\theta) & y \\ 0 & 0 & 1 \end{pmatrix}}_{{}^0\breve{\mathbf{T}}_1} \cdot \underbrace{\begin{pmatrix} \cos(\beta) & -\sin(\beta) & l\cos(\alpha) \\ \sin(\beta) & \cos(\beta) & l\sin(\alpha) \\ 0 & 0 & 1 \end{pmatrix}}_{{}^1\breve{\mathbf{T}}_2} \cdot \underbrace{\begin{pmatrix} 1 & 0 & d \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{{}^2\breve{\mathbf{T}}_3} $$

and extracting the position vector of the wheel center

$$ {}_0^0\mathbf{r}_3 = \begin{pmatrix} x + l\cos(\alpha + \theta) + d\cos(\beta + \theta) \\ y + l\sin(\alpha + \theta) + d\sin(\beta + \theta) \end{pmatrix} $$

Differentiating this vector yields the velocity ${}_0^0\dot{\mathbf{r}}_3$ of the wheel frame $\mathcal{K}_3$ in coordinates of the world-fixed frame $\mathcal{K}_0$

$$ {}_0^0\dot{\mathbf{r}}_3 = \begin{pmatrix} \dot{x} - l\dot{\theta}\sin(\alpha + \theta) - d\left(\dot{\beta} + \dot{\theta}\right)\sin(\beta + \theta) \\ \dot{y} + l\cos(\alpha + \theta)\dot{\theta} + d\cos(\beta + \theta)\left(\dot{\beta} + \dot{\theta}\right) \end{pmatrix} $$

To ensure that the pure rolling and non-slip conditions are fulfilled, this velocity must be equal to the previously derived wheel velocity ${}_0^3\mathbf{v}_3$. To do so, ${}_0^0\dot{\mathbf{r}}_3$ must also be decomposed in the wheel frame which can be accomplished by pre-multiplication with $({}^0\mathbf{R}_3)^T$ which yields

$$ {}_0^3\dot{\mathbf{r}}_3 = \begin{pmatrix} \cos(\beta + \theta) \cdot \dot{x} + \sin(\beta + \theta) \cdot \dot{y} - l\sin(\alpha - \beta) \cdot \dot{\theta} \\ -\sin(\beta + \theta) \cdot \dot{x} + \cos(\beta + \theta) \cdot \dot{y} + \left(d + l\cos(\alpha - \beta)\right) \cdot \dot{\theta} + d\dot{\beta} \end{pmatrix} $$

After setting ${}_3^3\dot{\mathbf{r}}_0$ equal to ${}_0^3\mathbf{v}_3$, the first of the two resulting equations corresponds to the pure rolling condition

$$ \cos(\beta + \theta) \cdot \dot{x} + \sin(\beta + \theta) \cdot \dot{y} - l\sin(\alpha - \beta) \cdot \dot{\theta} - r \cdot \dot{\varphi} = 0 $$

and the second equation to the non-slip condition

$$-\sin(\beta+\theta)\cdot\dot{x}+\cos(\beta+\theta)\cdot\dot{y}+\big(d+l\cos(\alpha-\beta)\big)\cdot\dot{\theta}+d\dot{\beta}=0$$

After rearranging these conditions, the constraint equations for castor wheels can finally be written in matrix form with notation adopted from Siciliano et al. (2016, p. 580):

$$\underbrace{\begin{pmatrix}\cos(\beta) & \sin(\beta) & -l\sin(\alpha-\beta)\\ -\sin(\beta) & \cos(\beta) & d+l\cos(\alpha-\beta)\end{pmatrix}}_{\mathbf{J}_{1c}(\beta_c)}\cdot{}^{\mathcal{R}}\mathbf{R}_{\mathcal{W}}(\theta)\cdot\dot{\boldsymbol{\xi}}-\underbrace{\begin{pmatrix}r\\ d\end{pmatrix}}_{\mathbf{J}_{2c}}\cdot\dot{\boldsymbol{\varphi}}=0$$

$$\underbrace{\begin{pmatrix}-\sin(\beta) & \cos(\beta) & d+l\cos(\alpha-\beta)\end{pmatrix}}_{\mathbf{C}_{1c}(\beta_c)}\cdot{}^{\mathcal{R}}\mathbf{R}_{\mathcal{W}}(\theta)\cdot\dot{\boldsymbol{\xi}}+\underbrace{(d)}_{\mathbf{C}_{2c}}\cdot\dot{\boldsymbol{\beta}}=0$$

In a more general case of multiple castor wheels, the matrices $\mathbf{J}_{1c},\mathbf{J}_{2c},\mathbf{C}_{1c}$, and $\mathbf{C}_{2c}$ contain one row for each wheel with $\mathbf{J}_{2c}$ and $\mathbf{C}_{2c}$ being diagonal matrices with the wheel radii and distances d for each wheel populating the diagonal respectively. The velocity vector $\dot{\boldsymbol{\xi}}$ is provided in absolute world coordinates (previously $\mathcal{K}_0$) which are transformed into robot coordinates (previously $\mathcal{K}_3$) with ${}^{\mathcal{R}}\mathbf{R}_{\mathcal{W}}$ as below.

$${}^{\mathcal{R}}\mathbf{R}_{\mathcal{W}}(\theta)={}^{\mathcal{W}}\mathbf{R}_{\mathcal{R}}(\theta)^{T}=\begin{pmatrix}\cos(\theta) & \sin(\theta) & 0\\ -\sin(\theta) & \cos(\theta) & 0\\ 0 & 0 & 1\end{pmatrix}$$

For conventional steering wheels and fixed wheels, the offset d usually is 0. A change with respect to the constraint equations of the castor wheel is the resulting simplification of the non-slip condition:

$$\underbrace{\begin{pmatrix}\cos(\beta) & \sin(\beta) & -l\sin(\alpha-\beta)\\ -\sin(\beta) & \cos(\beta) & d+l\cos(\alpha-\beta)\end{pmatrix}}_{\mathbf{J}_{1f},\mathbf{J}_{1s}(\beta_s)}\cdot{}^{\mathcal{R}}\mathbf{R}_{\mathcal{W}}(\theta)\cdot\dot{\boldsymbol{\xi}}-\underbrace{\begin{pmatrix}r\\ 0\end{pmatrix}}_{\mathbf{J}_{2f},\mathbf{J}_{2s}(\beta_s)}\cdot\dot{\boldsymbol{\varphi}}=0$$

$$\underbrace{\begin{pmatrix}-\sin(\beta) & \cos(\beta) & d+l\cos(\alpha-\beta)\end{pmatrix}}_{\mathbf{C}_{1f},\mathbf{C}_{1s}(\beta_s)}\cdot{}^{\mathcal{R}}\mathbf{R}_{\mathcal{W}}(\theta)\cdot\dot{\boldsymbol{\xi}}=0$$

As the indications for time dependency were dropped for brevity, it is important to point out that the angles β_c and β_s depend on time which is not the case for the fixed wheels.

For Mecanum or Swedish wheels, which are equipped with rollers at the circumference of the wheels that form an angle γ with respect to the wheel plane, there exists only a pure rolling constraint. For the meaningless case of $\gamma = 90°$, the constraint basically degenerates into the non-slip constraint of a fixed wheel, as the wheel can spin without generating any propulsion, but still only allows movement along the wheel plane. The equations for the Mecanum wheel can be reviewed in Siciliano et al. (2016, p. 580) under consideration of the difference in the chosen coordinate frames.

Self-Check Questions

1. Please complete the following sentence.

The *non-slip* condition restricts wheel motion perpendicular to the wheel plane.

2. Which of the following can be stated about constraints?

☐ *Geometric constraints are a restriction to the robot's achievable poses*

☐ Non-Holonomic constraints can be formulated based on pose parameters only

☐ Kinematic constraints do not limit the achievable velocity directions of the robot

☐ Non-Holonomic constraints are time derivatives of holonomic constraints

2.3 Mobile Robot Maneuverability

It is important to understand the general forms of motion of a wheeled mobile robot based on the configuration of conventional and omni-directional wheels. Depending on the setup, the steering angles of multiple wheels as well as the wheel velocities must be carefully coordinated in order to generate the desired motion. This can be formalized as the maneuverability of a wheeled mobile robot.

The degree of maneuverability δ_M is a measure to indicate how many independent motions, the robot can conduct in its workspace. It is limited by the non-slip constraints introduced by the conventional wheels, excluding castor wheels as will be shown. The maneuverability formally can be defined as

$$\underbrace{\delta_M}_{maneuverability} = \underbrace{\delta_m}_{\text{mobility}} + \underbrace{\delta_s}_{\text{steerability}}$$

The degree of mobility δ_m is a measure for the degrees of freedom that can directly be "actuated" through setting the wheel velocities. If all degrees of freedom of the robot can directly be controlled by adjusting the wheel velocities, such that the degrees of freedom of the robot equals its degree of mobility δ_m, the robot is governed by holonomic constraints only.

The degree of steerability δ_s on the other hand provides a measure for the number of independently steerable wheels. Indeed, δ_s does not necessarily equal the number of steerable wheels itself, as sometimes the steering angles are purposefully coordinated in a specific way to allow proper motion of the robot implying that the steering angles are not independent.

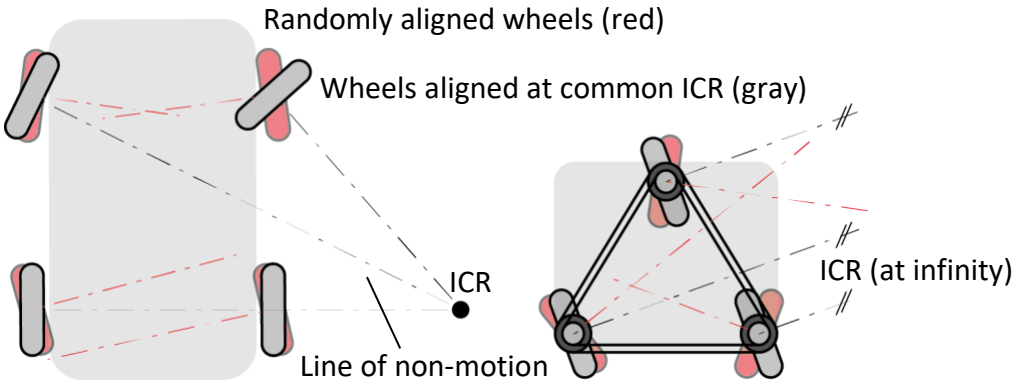
To gain an intuitive understanding, the problem is first discussed geometrically, before the definitions will be provided by analytical analysis of the constraint equations.

Geometric Approach

First, it shall be recalled, that during a translational displacement, every point is displaced by the same distance into the same direction. On the contrary, for a rotational displacement in a planar embedding there exists one point and in three dimensional space there exists one axis that is fixed in space under that displacement. Following the analogy of an arc segment of a circle which approaches a straight line with increasing circle radius, a translational displacement can be thought of as a rotation about a fixed point (in 2D) at infinity. With this in mind, any instantaneous

motion of a wheeled mobile robot at a certain point of time can be seen as a rotation about a fixed point. This point is the instantaneous center of rotation (ICR).

Car-Like Robot (a) and Synchronous Drive Robot With Indicated ICR (b)



Source: Florian Simroth (2023).

To determine the location of the ICR for a wheeled mobile robot with conventional wheels, the non-slip condition can be indicated by a line through the contact point of a wheel perpendicular to the wheel plane. These lines of "non-motion" are indicated by dot-dashed lines for each wheel of two different mobile robots in [FIGURE]. For the left example of a car-like robot, the wheels shaded in red indicate arbitrary orientations and the lines intersect at several points. This would imply a rotation about multiple centers simultaneously and violates the rigid body assumption as it would tear the robot apart. Indeed, a smooth defined motion for the car-like robot is only possible, if all lines intersect in a common point as indicated in black. As a result, assuming fixed wheels at the rear and steering wheels at the front, the relative angle of the steering wheels is subjected to the additional condition that the non-motion lines have to intersect on the non-motion line of the rear wheels. Since now the two steering wheels are not driven independently, this effectively reduces the degree of steerability to one.

As shown for the synchronous drive robot on the right, all lines are parallel and can be thought of to intersect at infinity yielding a purely translational motion. Clearly, if even one wheel would be misaligned, no smooth motion would be possible. Furthermore, as the velocity proportionally increases with the distance to the ICR, also the velocities of each wheel must be adapted.

The next section provides more insights on how the degrees of steerability, and mobility can be derived analytically from the constraints.

Analytical Approach

While the geometric approach provides general insight on how the constraints of different wheels influence the mobility of the robot, this information can be formalized using the constraint equations written in matrix form. For a more detailed derivation, the reader is referred to Campion et al. (1996) from which the following thoughts and properties are extracted.

First, the constraint equations resulting from the pure rolling conditions

$$\underbrace{\begin{pmatrix} \mathbf{J}_{1f} \\ \mathbf{J}_{1s}(\boldsymbol{\beta}_s) \\ \mathbf{J}_{1c}(\boldsymbol{\beta}_c) \\ \mathbf{J}_{1SW} \end{pmatrix}}_{\mathbf{J}_1(\overset{\smile}{\boldsymbol{\beta}_s},\boldsymbol{\beta}_c)} \cdot {}^{\mathcal{R}}\mathbf{R}_{\mathcal{W}}(\theta) \cdot \dot{\boldsymbol{\xi}} + \mathbf{J}_2 \cdot \dot{\boldsymbol{\varphi}} = 0$$

and from the non-slip constraints

$$\begin{aligned} \mathbf{C}_{1c}(\boldsymbol{\beta}_c) \cdot {}^{\mathcal{R}}\mathbf{R}_{\mathcal{W}}(\theta) \cdot \quad \dot{\boldsymbol{\xi}} \quad + \mathbf{C}_{2c} \cdot \dot{\boldsymbol{\beta}}_c \quad &= 0 \\ \mathbf{C}_{1f} \quad \cdot {}^{\mathcal{R}}\mathbf{R}_{\mathcal{W}}(\theta) \cdot \quad \dot{\boldsymbol{\xi}} \quad &= 0 \\ \mathbf{C}_{1s}(\boldsymbol{\beta}_s) \cdot {}^{\mathcal{R}}\mathbf{R}_{\mathcal{W}}(\theta) \cdot \quad \dot{\boldsymbol{\xi}} \quad &= 0 \end{aligned}$$

are gathered in matrix form. The indices c, f, s, SW indicate passive castor, fixed, steering, and Swedish wheels, respectively. The vector $\boldsymbol{\beta}$ gathers the time dependent steering angles for steering and castor wheels as well as the fixed angle for fixed wheels. $\dot{\boldsymbol{\xi}}$ stands for the robot velocity in the world-fixed coordinate frame, while $\dot{\boldsymbol{\varphi}}$

contains the angular wheel velocities. Each wheel generates one row of $\mathbf{J}_1$, $\mathbf{J}_2$, $\mathbf{C}_1$, and castor wheels also in $\mathbf{C}_{2c}$. $\mathbf{J}_1$ and $\mathbf{C}_1$ are composed of three columns corresponding to the three velocity parameters in $\dot{\boldsymbol{\xi}}$. $\mathbf{J}_2$ is a squared matrix containing the radii of each wheel and $\mathbf{C}_{2c}$ is a diagonal matrix containing the distances d of the castor wheels. For Swedish wheels, the equivalent radius $r_{\mathrm{SW}} = \cos(\gamma) \cdot r$ with $\gamma \neq \pi/2$ is inserted, taking account for the only a part of φ is effectively transformed into wheel center velocity.

Based on the equations for the castor wheels, it can be shown that passive castor wheels actually do not restrain the robot mobility in any way: There always exist values for $\dot{\boldsymbol{\beta}}_c$ and $\dot{\boldsymbol{\varphi}}$ multiplied with the non-singular matrices $\mathbf{C}_{2c}$ and $\mathbf{J}_2$ respectively, that comply with the velocities $\dot{\boldsymbol{\xi}}$, such that the conditions can always be fulfilled.

The degree of steerability directly corresponds to the rank of the constraint matrix for the steering wheels

$$\delta_s = \mathrm{rank}(\mathbf{C}_{1s}(\boldsymbol{\beta}_s))$$

Recall, that the rank reduces once multiple rows are linearly dependent. This happens, for example, if the steering angles are actuated in a way that they always fulfill the condition of a common ICR. For the degree of steerability follows $0 \leq \delta_s \leq 2$, ranging from none to multiple steering wheels steered in a coordinated motion. The rank of the combined constraint matrix for fixed and steering wheels gives formal insight on the degree of mobility

$$\delta_m = 3 - \mathrm{rank}\begin{pmatrix} \mathbf{C}_{1f} \\ \mathbf{C}_{1s}(\boldsymbol{\beta}_s) \end{pmatrix}$$

The maximum rank of $\mathbf{C}_{1f}$ with its three columns is three. This case would result in a non-mobile robot. A rank of 2 for $\mathbf{C}_{1f}$ would indicate that the fixed wheels intersect at a common ICR, such that the robot would only be able to perform a pure rotational motion. Therefore, $\mathrm{rank}(\mathbf{C}_{1f}) \leq 1$ for useful mobile robots which means that there

are either no fixed wheels or one or more fixed wheels on the same axle. Furthermore, if the steering wheels are not on a common axle with the fixed wheels, there are no rows in $\mathbf{C}_{1f}$ that are linearly dependent on rows in $\mathbf{C}_{1s}$ and the ranks of both matrices can be summed up independently.

Besides being able to determine the degree of maneuverability $\delta_M = \delta_m + \delta_s$ in an analytical manner, usable wheeled mobile robots, i.e. ones that are able to move around in planar space, can be categorized into five groups. These groups are identified by the possible combinations of the two parameters δ_m and δ_s and examples for each group are provided in [TABLE].

Five Groups of Usable, Wheeled Mobile Robots

Maneuverability δ_M	Mobility δ_m	Steerability δ_s	Example
3	3	0	Omni-directional platform with Swedish wheels or three castor wheels
3	2	1	Platform with 1 steering and multiple castor wheels
3	1	2	Platform with 2 steering and one castor wheel
2	2	0	Differential drive bot with two fixed wheels on same axis and one passive castor wheel
2	1	1	Synchronous drive robot with three simultaneously steered and actuated wheels

Source: Florian Simroth (2023).

Self-Check Questions

3. Why is the case $\delta_m + \delta_s = 1$ usually not considered as a "useful" robot? Which motion does it describe?
   Limited motions due to fixed ICR.
4. How is the location of the ICR in space different with respect to time for a pure rotation and a general motion?
   For a pure rotation, the location of the ICR is fixed while it changes over time for a general motion.

2.4 Mobile Robot Workspace

After having classified mobile robots into five classes, it is worth considering, how these classes relate to the robot's workspace. The workspace of the mobile robot can be seen as the full set of poses the robot can access given the specific constraints of its wheel configuration. For a wheeled mobile robot, the workspace usually is a planar subspace of the three dimensional physical space with a degree of freedom (DOF) of three. This is not always the case, as for the synchronous drive robot, the degree of freedom is just two, since the pose cannot be actuated. Next, the dimension of the robot workspace, i.e., the robot's degrees of freedom "on position level", shall be related to the degree of mobility and maneuverability.

The following analogy shall be highlighted: the degree of freedom is to the robot on position level, what the degree of mobility is to the robot on velocity level, i.e., the number of independently achievable velocities. The degree of mobility is often also referred to as the differential degree of freedom (DDOF) (Siegwart and Nourbakhsh, 2004, p. 74). "A robot is holonomic if and only if DDOF = DOF" (Siegwart and Nourbakhsh, 2004, p. 77). In that sense, if a mobile robot with DOF = 3 also has DDOF = 3, there are only holonomic constraints present and the robot can be referred to as a holonomic robot. This case can only appear in the absence of fixed and steering wheels.

As mentioned before, the degree of maneuverability can be less than the DOF of the robot itself. Yet, the robot can reach all poses within its workspace, though, through the presence of non-holonomic constraints, it cannot reach all poses directly but normally must maneuver. One case of particular interest is a maneuverability of $\delta_M = 3$ for a mobile robot with DOF=3. Such a robot can be referred to as omni-directional, as it can drive in any direction. Yet, there are subtle differences between a holonomic robot with $\delta_M = \delta_m + \delta_s = 3 + 0 = 3$ and a robot with $\delta_M = \delta_m + \delta_s = 2 + 1 = 3$ even though both have a maneuverability of 3. For an explanation the concepts of path and trajectory will be introduced.

Path

Time independent continuous sequence of poses that a robot passes.

The **path** of a mobile robot is the continuous sequence of poses the mobile robot passes through. The **trajectory** on the other hand is the path of the mobile robot with respect to time and therefore takes into account at which time instant which point of the path is reached. It can now be stated that both of the aforementioned omni-directional robots are capable to travel along the same path, yet, the latter one may not follow the holonomic on any trajectory as it needs to reorient its steering wheels to first adjust the ICR accordingly. In contrast to the holonomic robot, it cannot directly perform a motion laterally to its steering wheels. Therefore, from the analysis of the degree of mobility and maneuverability, important insights on how the robot is able to operate within its workspace can be drawn.

Trajectory

Continuous sequence of poses a robot passes which also considers the timings of the robot.

Self-Check Questions

5. Please complete the following sentence.

The *trajectory* is a continuous sequence of poses with respect to time.

6. A robot is considered holonomic if and only if:

☐ It is omni-directional

☐ Its maneuverability is equal to its degree of freedom

☐ *Its mobility equals its degree of freedom*

☐ Differential degree of freedom is equal to its maneuverability.

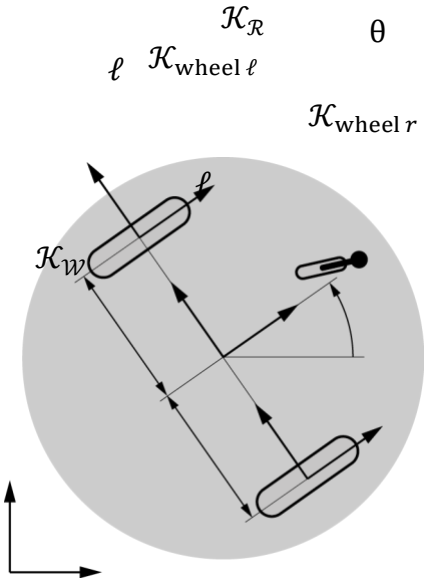
2.5 Applications

In this section, the full set of constraint equations for a wheeled mobile robot will be derived based on the constraints arising at standard wheels. These equations will be analyzed for maneuverability and mobility analysis and afterwards a forward kinematics simulation will be performed.

Two-Wheeled Differential Drive Robot

IN [FIGURE] the sketch of a simple differential drive robot with two wheels is shown. Often, these robots are equipped with an additional passive castor wheel for stability. As it does not contribute to any additional constraint equations on the robot's pose, though, it will be neglected in regards to the further analysis.

Differential Drive Robot With Two Fixed And One Castor Wheel



Source: Florian Simroth (2023).

The two pure rolling conditions can be derived from the previously presented equations, where the indices ℓ and r indicate the left and right wheel as seen from the robot's body-fixed coordinate system $\mathcal{K}_{\mathcal{R}}$:

$$\underbrace{\begin{pmatrix} 1 & 0 & -\ell \\ 1 & 0 & \ell \end{pmatrix}}_{\mathbf{J_1}} \cdot {}^{\mathcal{R}}\mathbf{R}_{\mathcal{W}}(\theta) \cdot \dot{\boldsymbol{\xi}} + \underbrace{\begin{pmatrix} -r & 0 \\ 0 & -r \end{pmatrix}}_{\mathbf{J_2}} \cdot \begin{pmatrix} \dot{\varphi}_\ell \\ \dot{\varphi}_r \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

The two non-slip constraints are similarly derived.

$$\underbrace{\begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix}}_{\mathbf{C_1}} \cdot {}^{\mathcal{R}}\mathbf{R}_{\mathcal{W}}(\theta) \cdot \dot{\boldsymbol{\xi}} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Next, the degree of maneuverability and mobility are derived, while the degree of steerability is clearly 0, due to the absence of any steering wheels.

$$\delta_M = \delta_m + \delta_s = \left(3 - \text{rank}\begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix}\right) + 0 = 2 + 0 = 2$$

The two identical rows in $\mathbf{C_1}$ stem from the fact, that both fixed wheels share a common axis, which reduces the rank to one. As the mobility is less than the DOF of the robot, the robot is classified as non-holonomic.

Forward Kinematics

As a last step the forward kinematics shall be simulated for two different motion types: 1) circular motion and 2) a spiral motion. The mobility analysis leads to the conclusion, that only two of the three components in $\dot{\boldsymbol{\xi}}$ are independent, as the degree of mobility was determined to be $\delta_m = 2$. In fact, only the two wheel speeds $\dot{\varphi}_\ell$ and $\dot{\varphi}_r$ can be controlled independently which need to be mapped to the three components of $\dot{\boldsymbol{\xi}}$. The respective mapping matrix can be derived by geometric reasoning using for instance the ICR. The ICR can easily be determined in a local robot frame as a function of the wheel speeds with its position restricted to some point on the common wheel axis. The resulting relationship can be used to derive $\dot{\theta}$ from the

wheel speeds. Here, these equations shall be established by reformulating the pure rolling constraint equations using the pseudo-inverse $\hat{\mathbf{J}}_1^{-1}$ of $\mathbf{J}_1$ according to Tzafestas (2013, p.34):

$$\hat{\mathbf{J}}_1^{-1} = \mathbf{J}_1^T \cdot (\mathbf{J}_1 \cdot \mathbf{J}_1^T)^{-1} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ 0 & 0 \\ -\frac{1}{2\ell} & \frac{1}{2\ell} \end{pmatrix}$$

The pure rolling constraint equation is then rearranged for $\dot{\boldsymbol{\xi}}$ as a function of the wheel speeds:

$$\begin{aligned} \dot{\boldsymbol{\xi}} = & \quad -{}^{\mathcal{R}}\mathbf{R}_{\mathcal{W}}(\theta)^T \cdot \hat{\mathbf{J}}_1^{-1} \cdot \mathbf{J}_2 \cdot \begin{pmatrix} \dot{\varphi}_\ell \\ \dot{\varphi}_r \end{pmatrix} \\ = & \begin{pmatrix} -\frac{1}{2} r\cos(\theta) & -\frac{1}{2} r\cos(\theta) \\ -\frac{1}{2} r\sin(\theta) & -\frac{1}{2} r\sin(\theta) \\ \frac{r}{2\ell} & -\frac{r}{2\ell} \end{pmatrix} \cdot \begin{pmatrix} \dot{\varphi}_\ell \\ \dot{\varphi}_r \end{pmatrix} \end{aligned}$$

The three resulting equations, now with explicitly indicated time dependency,

$$\begin{pmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \end{pmatrix} = \begin{pmatrix} -\frac{1}{2} r\cos(\theta)(\dot{\varphi}_r + \dot{\varphi}_\ell) \\ -\frac{1}{2} r\sin(\theta)(\dot{\varphi}_r + \dot{\varphi}_\ell) \\ \frac{r(\dot{\varphi}_\ell - \dot{\varphi}_r)}{2\ell} \end{pmatrix}$$

can then be analytically integrated for simple input functions for the wheel speeds. First, the case of straight motion shall be analyzed by assuming constant wheel velocities $\dot{\varphi}_\ell(t) = \dot{\varphi}_r(t) = c$:

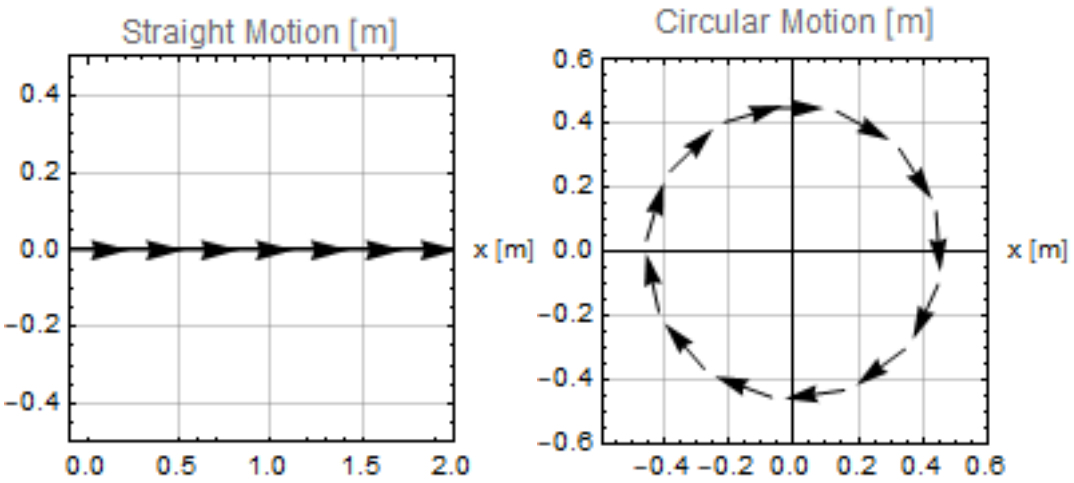
$$\begin{pmatrix} x(t) \\ y(t) \\ \theta(t) \end{pmatrix} = \begin{pmatrix} -cr\int \cos\big(\theta(t)\big)\, dt \\ -cr\int \sin\big(\theta(t)\big)\, dt \\ 0 \end{pmatrix} = \begin{pmatrix} -crt \\ 0 \\ 0 \end{pmatrix}$$

The result for $\theta(t)$ was inserted into the other two equations to obtain the robot's displacements over time with all initial values set to zero. Similarly, a circular motion with $\dot{\varphi}_\ell(t) = c$ and $\dot{\varphi}_r(t) = 2c$ yields:

$$\begin{pmatrix} x(t) \\ y(t) \\ \theta(t) \end{pmatrix} = \begin{pmatrix} -\frac{3}{2}cr\int \cos\big(\theta(t)\big)\,dt \\ -\frac{3}{2}cr\int \sin\big(\theta(t)\big)\,dt \\ -\frac{crt}{2\ell} \end{pmatrix} = \begin{pmatrix} -3\ell\sin\left(\frac{crt}{2\ell}\right) \\ -3\ell\cos\left(\frac{crt}{2\ell}\right) \\ -\frac{crt}{2\ell} \end{pmatrix}$$

When simulating these equations of motions with $\ell = 0.15\text{m}, r = 0.05\text{m}, c = 15(1/\text{s})$, one can come up with the following trajectories for the straight and circular motion respectively. In general though, the integration is not as obvious as in the case of these straight forward functions.

Forward Kinematics Of Robot Pose For Given Wheel Velocities



Source: Florian Simroth (2023).

Self-Check Questions

7. What kind of motion would result if the wheel velocity of one wheel would increase over time?

A spiral-like motion.

8. How do the wheel velocities need to be set to generate a pure rotation about the left wheel?

$\dot{\varphi}_\ell(t) = 0$ *and* $\dot{\varphi}_r(t) = c$

Summary

The pose of a robot is described by its position and orientation for which three parameters in a plane and six parameters is space are required. With the use of rotation matrices, the coordinates of a vector can be transformed from one coordinate frame to another, such that, for instance, the robot velocity can be expressed in absolute coordinates in a world-fixed frame or in a local body-fixed robot frame. Homogeneous coordinates allow a uniform description of combined rotations and translations, whereby multiple transformations can be concatenated by matrix multiplication.

The contacts of wheels with the ground plane introduce constraints on the robot's possible motions. At ideal conventional wheels a pure rolling and a non-slip constraint arises. The pure rolling conditions ensures that the wheel does not spin while the non-slip constraint implies that the motion vector at the wheel center is within the wheel plane. Fixed and steering wheels introduce non-holonomic constraints, which limit the achievable velocity directions of a robot but do not directly act on the accessible poses. In forward kinematics the robot pose is estimated in physical space based on the configuration parameters of the robot while inverse kinematics do the opposite which is usually not uniquely defined, as multiple (sequences) of configurations can yield the same pose.

The maneuverability is a measure for the way the robot is able to move in its environment. It is the sum of the mobility and steerability, whereby the mobility is equal to the differential degree of freedom and indicates the number of independent velocities of a robot, while the steerability corresponds to the number of independently steerable wheels. With the mobility and steerability parameters,

wheeled mobile robots can be classified into five different groups with different motion dependent properties.

Unit 3 – Dynamics

Study Goals

On completion of this unit, you will be able to …

…define the dynamic model of a wheeled mobile robot.

… identify an independent set of suitable generalized velocities.

… apply the dynamic model to simulate a mobile robot via forwards dynamics.

… identify mass and inertia properties.

Unit 3 – Dynamics

Introduction

In contrast to kinematics which was solely concerning motion constraints and how a mechanism can actually move, in dynamics, the motion of robots is analyzed under the presence of forces. The forces can originate from external sources or result from velocities and constraints within the mechanism. In addition to geometric properties of the robot used for kinematics, now also the mass distribution and forces like friction are inputs to the model. Dynamics can be subdivided as shown in [FIGURE] into statics, which analyzes the internal and external forces at static equilibrium as well as kinetics, which inspects the robot in motion.

Segmentation of Dynamics into Statics and Kinetics

Source: Florian Simroth (2023).

The dynamic analysis builds upon the kinematics of the constrained motion of a mechanism and derives internal and external resulting forces acting on the components and joints. Similar to kinematics, forward and inverse dynamics can be discerned. Forward dynamics determines the accelerations, based on the configuration, velocities, as well as forces and moments acting on a robot which can then be used to predict future states for simulation. Inverse dynamics on the other

hand calculate the forces and torques present within joints or bodies if the robot performs a predefined motion. This becomes handy for trajectory planning to determine the actuation forces necessary to realize a certain trajectory as well as control.

In this chapter, first, the basics of Lagrange equations are reviewed to model the dynamics of wheeled mobile robots. The equations are then applied to the main platform of wheeled mobile robots. The dynamics and kinematics of serial chains are covered in the unit of manipulators.

3.1 Basics

Several approaches can be used to model the dynamics of multi-body systems in general. Two widely used methods are the Newton-Euler approach as well as the Lagrange formalism.

The Newton-Euler method builds upon the equations based on Newton's laws to describe the relation of translational forces and accelerations of rigid bodies in a mechanism and the Euler equations to model the forces and torques resulting from rotational movement. Particularly for open chains, such as common manipulators, there are efficient algorithms available. For example, the inverse dynamics problem can be solved efficiently by the recursive Newton-Euler algorithms presented in Luh et al. (1980) and for forwards dynamics, methods such as the articulated-body algorithm by Featherstone (1983) can be used that both are of $O(n)$ complexity with n being the robot's degrees of freedom or number of bodies in the latter case (Siciliano & Kathib, 2016, p. 54).

For establishing the dynamic equations of motion using the Lagrange method, the kinetic and the potential energy of the mechanism of interest need to be determined. This is usually done in terms of generalized coordinates. The core is the Lagrange function $\mathcal{L}$ which is defined as the difference between the kinetic energy $\mathcal{T}$ and the potential energy $\mathcal{U}$ of the system.

$$\mathcal{L} = \mathcal{T} - \mathcal{U}$$

The kinetic energy can be computed as

$$\mathcal{T} = \frac{1}{2} \cdot \dot{\mathbf{q}}^T \cdot \mathbf{M}\dot{\mathbf{q}}$$

where $\dot{\mathbf{q}}$ are the generalized velocities and $\mathbf{M}$ is the generalized mass matrix. The potential energy $\mathcal{U}$ refers to the energy due to elevation with respect to a defined absolute reference point in the presence of gravity as well as other potentials, for instance, due to springs. The Lagrange formalism allows for a compact description of systems with a low number of degrees for freedom and, furthermore, it also provides insight on some important properties regarding control algorithms.

In the end, both approaches result in equations that will describe the dynamics for a mechanism equally. This section will use the Lagrange method for which some concepts will be closer illuminated hereafter.

Generalized Coordinates and Forces

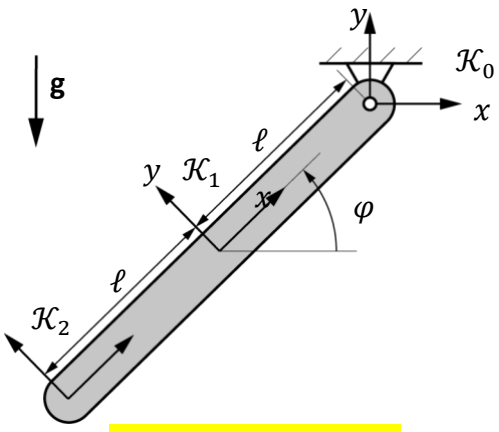
While coordinates in the physical robot workspace are quite familiar, generalized coordinates $\mathbf{q}$, velocities $\dot{\mathbf{q}}$, and accelerations $\ddot{\mathbf{q}}$, are less intuitive to imagine. This is mainly because it is hard to visualize the configuration space with more than three dimensions which is easily exceeded for mobile robots consisting of multiple components. If the absolute position of a mass particle is to be described, three coordinates are sufficient, while for a rigid body at least three more coordinates are required to describe its orientation in space. This could be three Euler angles or, for instance, four parameters if quaternions are used which are subject to one additional condition. Obviously, different sets of parameters are suitable to describe the configuration of a robot consisting of one or more rigid bodies, thus the term “generalized.”

If the number of generalized coordinates used to describe the configuration of the robot is equal to its degrees of freedom, the set is minimal and the parameters are sometimes referred to as **minimal coordinates**. If there are more generalized coordinates used than there are degrees of freedom, the generalized coordinates are subject to constraints, as not all coordinates are freely choosable. The set of generalized coordinates can then be divided into a set of independent and a set of the dependent coordinates. By solving the constraint equations for the dependent coordinates, the dependent coordinates can be expressed as functions of the independent ones, yet, this is often not an easy task (Shabana, 2005, p. 97).

Minimal coordinates
Set of generalized coordinates with number of coordinates corresponding to the degree of freedom of the mechanism

The example of a single pendulum shown in [FIGURE] shall illustrate how different sets of generalized coordinates can be used. Obviously, the pendulum has one degree of freedom and if the rotation angle φ is chosen as the generalized coordinate $\mathrm{q} = \varphi$, the configuration is completely defined. This represents a choice of generalized coordinates, which is also minimal.

Geometric Properties of a Single Pendulum



Source: Florian Simroth (2023).

Likewise, one could choose the x- and y- position of the center of gravity as well as the orientation φ as generalized coordinates $\mathbf{q} = (x, y, \varphi)^{\mathrm{T}}$. Though, as the motion of the center of gravity is constrained to a circle about the pivot, these coordinates

cannot be chosen freely, but instead need to fulfill the additional constraints $x = -\ell \cdot \cos(\varphi)$, $y = -\ell \cdot \sin(\varphi)$. Note, that one could also decide to express y and φ as functions of x this though, would leave some ambiguity as there are two solutions for each x value (except for $x = +/-\ell$). In conclusion, it is possible to reduce or eliminate additional constraints by a corresponding choice of generalized coordinates.

In direct analogy to the generalized coordinates, the generalized forces $\mathbf{Q}$ are a set of forces and torques acting "along" the generalized coordinates. Depending on whether the corresponding generalized coordinate refers to a translation or rotation, the generalized force can represent a force or a torque. If the configuration of the pendulum is described with the generalized coordinate $\mathrm{q} = \varphi$ and the external force $\mathbf{Q}_{\boldsymbol{g}}$ due to gravity shall be modeled, from geometry, the following term can be derived:

$$\mathbf{Q}_{\boldsymbol{g}} = (\boldsymbol{g} \cdot \mathrm{m} \cdot \ell \cdot \cos(\varphi))$$

The term describes the moment of force about the pivot exerted by the gravitational force $\boldsymbol{g} \cdot \mathrm{m}$ acting on the center of gravity. Alternatively, if the generalized coordinates $\mathbf{q} = (x, y, \varphi)^{\mathrm{T}}$ are chosen, the generalized force vector would have one component for each coordinate such that:

$$\mathbf{Q}_{\boldsymbol{g}} = \begin{pmatrix} 0 \\ \boldsymbol{g} \cdot \mathrm{m} \\ 0 \end{pmatrix}$$

In the latter case, the forces that hold the pendulum at its pivot result from the constraint equations which will add additional reaction forces.

Generalized mass matrix
Matrix storing the mass and inertia properties of a mechanism with entries depending on the choice of generalized coordinates.

Generalized Mass Matrix

The mass matrix contains the properties of a rigid body that describe its "resistance" with respect to the resulting acceleration once exposed to external forces and torques. These properties do not only depend on the total mass

of the rigid body itself, but also on the distribution of the mass. If observed in a body-fixed coordinate system, the mass matrix is constant, while from the perspective of a space-fixed coordinate frame, the mass matrix depends on the orientation of the rigid body. If the pose of a rigid body is described by a set of generalized coordinates $\mathbf{q} = (x, y, z, \gamma, \beta, \alpha)^T$ composed of three position parameters and the three ZYX-Euler angles, the generalized mass matrix in its general form and its sub components can be written as shown by Shabana (2005, p. 146):

$$\mathbf{M} = \begin{pmatrix} \mathbf{M}_T & \mathbf{M}_{TR} \\ \text{sym.} & \mathbf{M}_R \end{pmatrix}$$

Here, $\mathbf{M}_T$ can be understood as the translational inertia component of the body with a total mass of m which is derived from the volume integral and the density ρ:

$$\mathbf{M}_T = \int_{\mathsf{V}} \rho \cdot \mathbf{I} \, d\mathsf{V} = \begin{pmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & m \end{pmatrix}$$

$\mathbf{M}_R$ corresponds to the rotational component of the mass matrix:

$$\mathbf{M}_R = \int_{\mathsf{V}} \rho \cdot \mathbf{B}^T \cdot {}^i\tilde{\mathbf{r}}^T \cdot {}^i\tilde{\mathbf{r}} \cdot \mathbf{B} \, d\mathsf{V} = \mathbf{B}^T \cdot \boldsymbol{\Theta} \cdot \mathbf{B}$$

The matrix $\mathbf{B}$ maps the generalized orientation velocities $(\dot{\gamma},\ \dot{\beta},\ \dot{\alpha})$ to the angular velocity ${}^i\boldsymbol{\omega}$ in body-fixed coordinates $\boldsymbol{\omega} = \mathbf{B} \cdot (\dot{\gamma},\ \dot{\beta},\ \dot{\alpha})^T$, which can be derived from the Poisson equation $\tilde{\boldsymbol{\omega}} = {}^0\mathbf{R}_i^T \cdot {}^0\dot{\mathbf{R}}_i$. The tilde indicates the skew symmetric matrix form of a vector:

$$\mathbf{B} = \begin{pmatrix} -\sin(\beta) & 0 & 1 \\ \cos(\beta)\sin(\alpha(t)) & \cos(\alpha(t)) & 0 \\ \cos(\beta)\cos(\alpha(t)) & -\sin(\alpha(t)) & 0 \end{pmatrix}$$

$\boldsymbol{\Theta}$ is the spatial moment of inertia tensor of the body with respect to $\mathcal{K}_i$ resulting from the volume integral over the vectors ${}^i\tilde{\mathbf{r}}$ to all mass particles in body-fixed coordinates.

The matrix $\mathbf{M}_{TR}$ represents the coupling between translational and rotational inertia components and can be determined in the following way:

$$\mathbf{M}_{TR} = -\int_{\mathrm{V}} \rho \cdot {}^{0}\mathbf{R}_i \cdot {}^{i}\tilde{\mathbf{r}} \cdot \mathbf{B}\, d\mathrm{V} = -{}^{0}\mathbf{R}_i \cdot \left(\int_V \rho \cdot {}^{i}\tilde{\boldsymbol{r}}\, dV \right) \cdot \mathbf{B}$$

where $\mathbf{R}$ is the rotation matrix from space-fixed $\mathcal{K}_0$ to body-fixed $\mathcal{K}_i$ coordinates according to the generalized rotations.

$${}^{0}\mathbf{R}_i = \mathrm{Rot}\,[\,z, \gamma\,] \cdot \mathrm{Rot}\,[\,y, \beta\,] \cdot \mathrm{Rot}\,[\,x, \alpha\,]$$

It is important to highlight that the equations simplify for a smart choice of $\mathcal{K}_i$. If $\mathcal{K}_i$

- is located at the center of mass, $\mathbf{M}_{TR} = \mathbf{0}$
- is oriented along the principal axes, $\boldsymbol{\Theta}$ is a diagonal matrix with the principal moments of inertia on the diagonal

In general, the generalized mass matrix maps the generalized accelerations $\ddot{\mathbf{q}}$ to the generalized forces and torques similar as the physical mass matrix maps the accelerations in task space to the forces and torques acting in the physical space.

Self-Check Questions

11. For which choice of generalized coordinates is the generalized mass matrix of the form described above valid?

In general, the described mass matrix is valid for an unconstrained rigid body in three-dimensional space with a configuration described by its three position coordinates in a Cartesian reference frame and using ZYX-Euler angle parameterization for its orientation. If other generalized coordinates are used, due to a different order of rotation angles or a subset resulting from elimination of Lagrange multipliers, the matrix may look quite different.

12. For the single pendulum, would the y position of the center of mass be a suitable generalized coordinate?

The y-position could be used as a generalized coordinate, though it is not the best choice due to non-unique solutions.

3.2 Dynamic Modeling

To derive a general description of the dynamics for a mobile robot, the kinematic relationships have to be defined first. Based on these relations, the reaction of the system to external forces and torques can generally be described by the **dynamic equations of motions in canonical form** as shown in Lynch (2017, p.271):

$$\begin{aligned} &\text{Inverse Dynamics} && \mathbf{Q} &&= \mathbf{M}(\mathbf{q}_{\text{min}}) \cdot \ddot{\mathbf{q}}_{\text{min}} + \mathbf{V}(\mathbf{q}_{\text{min}}, \dot{\mathbf{q}}_{\text{min}}) \\ &\text{Forward Dynamics} && \ddot{\mathbf{q}}_{\text{min}} &&= \mathbf{M}^{-1}(\mathbf{q}_{\text{min}}) \cdot \left(\mathbf{Q} - \mathbf{V}(\mathbf{q}_{\text{min}}, \dot{\mathbf{q}}_{\text{min}})\right) \end{aligned}$$

Here, $\mathbf{Q}$ is the vector of generalized forces, $\mathbf{M}$ is the generalized mass matrix, and $\mathbf{q}_{\text{min}}$ and its derivative are the minimal coordinates and their derivatives, while $\mathbf{V}(\mathbf{q}_{\text{min}}, \dot{\mathbf{q}}_{\text{min}})$ gathers, for instance, dynamical terms resulting from centrifugal and Coriolis forces. The indicator “min” was used to point out, that the shown equations are only valid for an independent set of generalized coordinates, i.e., one that is not subject to further constraints and therefore corresponds to the number of degrees for freedom. Forward dynamics derive the generalized accelerations based on the configuration, velocities, and for given generalized (actuation) forces, whereby a non-singular mass matrix is presumed. Inverse dynamics derive the necessary generalized forces and torques for a given trajectory described by the configuration coordinates, velocities, accelerations, and system properties.

In order to derive a state-space formulation suitable for integration, which also considers typical forces and constraints for wheeled mobile robots, the Lagrangian equations are used as a basis. The following derivation is guided by the elaboration of Klančar et al. (2017). A general form of the Lagrange equations can be given as

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{q}_i}\right) - \frac{\partial \mathcal{L}}{\partial q_i} + \frac{\partial \mathcal{P}}{\partial \dot{q}_i} + \mathbf{Q}_{\text{Dist}} = \mathbf{Q} - \sum_{j=1}^{m} \lambda_j \, a_{ij}$$

where the Lagrange function $\mathcal{L} = \mathcal{T} - \mathcal{U}$ is the difference of kinetic $\mathcal{T}$ and potential $\mathcal{U}$ energy, while $\mathcal{P}$ accounts for power losses due to friction or damping, $\mathbf{Q}_{\text{Dist}}$ to model system disturbances, $\mathbf{Q}$ as the vector of generalized forces, the Lagrange multiplier λ_j, and the coefficient a_{ij} corresponding to the generalized coordinate q_i stemming from the constraint j resulting in $i = 1,2, \dots, n$ equations with n being the number of generalized coordinates and m being the number of constraints. Concentrating on wheeled mobile robots, it is assumed that the robot moves in a plane, i.e., $\partial \mathcal{U} / \partial \dot{\mathbf{q}} = 0$, neglecting friction and damping $\mathcal{P} = 0$ as well as system disturbances $\mathbf{Q}_{\text{Dist}} = 0$, the formula can be further simplified.

Next, it can be shown, that after performing the differentiation steps, the Lagrange equations can be represented in a matrix form. But before introducing the matrix form, it is worth having a closer look on the holonomic and non-holonomic constraints. Let $\mathbf{C}(\mathbf{q}) = 0$ be the set of m_h holonomic constraints and $\mathbf{J}_q(\mathbf{q}) = \partial \mathbf{C} / \partial \mathbf{q}$ its time derivative, such that $\mathbf{J}_q(\mathbf{q})\dot{\mathbf{q}} = 0$. Furthermore, assume m_n non-holonomic constraints of the form $\mathbf{A}(\mathbf{q})\dot{\mathbf{q}} = 0$ and $m = m_h + m_n$. The dynamic motion equations under assumption of the previously mentioned simplifications can be written as shown in Yun (1995, p. 2691):

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{V}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{Q} + \underbrace{\mathbf{J}_q^T(\mathbf{q})\boldsymbol{\lambda}_\mathbf{h}}_{\text{holonomic term}} + \underbrace{\mathbf{A}^T(\mathbf{q})\boldsymbol{\lambda}_n}_{\text{non-holonomic term}} \qquad (1)$$

Elimination of Holonomic Constraints

There are multiple ways of treating these constraints which in Shabana (2005, p.120) are termed embedding technique and augmented formulation. The latter one refers to working with a set of redundant generalized coordinates which requires the consideration of constraint equations as an additional mean to ensure that the dynamic model is congruent with the kinematic robot model. Therefore, in addition to the set of differential equations, the algebraic constraint equations need to be solved. The embedding technique rather eliminates the constraint forces and as a

consequence thereof, the Lagrangian multipliers are eliminated as well by expressing the dependent generalized coordinates by means of the independent (minimal) generalized coordinates. This is possible due to the implicit function theorem, though the process may lead to singularities, and therefore in general, it is preferred to directly use a set of well-defined minimal coordinates instead of reducing the redundant coordinates according to Siciliano (2010, p. 470). Yet, for the purpose of better understanding, the embedding technique will be presented to eliminate the Langrangian multipliers from the set of equations.

First, the holonomic constraints are embedded by partitioning the generalized coordinates into a set of dependent $\mathbf{q}_d$ and independent $\mathbf{q}_i$ coordinates such that $\mathbf{q} = (\mathbf{q}_d^T\ \mathbf{q}_i^T)^T$. The constraint Jacobian can then be resorted to $\mathbf{J}_q = \left(\mathbf{J}_{q_d}^T \mathbf{J}_{q_i}^T\right)^T$ with $\mathbf{J}_{q_d} = \partial \mathbf{C}(\mathbf{q}) / \partial \mathbf{q}_d$ and $\mathbf{J}_{q_i} = \partial \mathbf{C}(\mathbf{q}) / \partial \mathbf{q}_i$. By means of virtual displacements $\delta\mathbf{q}$ the following relation can be derived following Shabana (2005, p.121):

$$
\begin{aligned}
\mathbf{J}_q \delta \mathbf{q} &= \mathbf{J}_{q_d} \delta \mathbf{q}_d + \mathbf{J}_{q_i} \delta \mathbf{q}_i = 0 \\
\delta \mathbf{q}_d &= -\mathbf{J}_{q_d}^{-1} \mathbf{J}_{q_i} \cdot \delta \mathbf{q}_i \\
\delta \mathbf{q} &= \underbrace{\begin{pmatrix} -\mathbf{J}_{q_d}^{-1} \cdot \mathbf{J}_{q_i} \\ \mathbf{I} \end{pmatrix}}_{\mathbf{P}} \cdot \delta \mathbf{q}_i
\end{aligned}
$$

With $\mathbf{P}$, the mass matrix and generalized forces can then be projected to the minimal set of coordinates:

$$
\begin{aligned}
\mathbf{M}_{\text{min}} &= \mathbf{P}^T \mathbf{M} \mathbf{P} \\
\mathbf{Q}_{\text{min}} &= \mathbf{P}^T \mathbf{Q}
\end{aligned}
$$

Example

For the previously introduced pendulum, the redundant generalized coordinates $\mathbf{q} = (x, y, \varphi)^T$ referring to the center of mass are chosen. The following mass matrix results:

$$\mathbf{M} = \begin{pmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & \Theta \end{pmatrix}$$

The generalized coordinates are partitioned such that $\mathbf{q}_i = (\varphi)$ is the independent and $\mathbf{q}_d = (x, y)^T$ are the dependent coordinates. The constraints and resulting constraint Jacobians are given as

$$\mathbf{C} = \begin{pmatrix} \cos(\varphi)\ell + x \\ \sin(\varphi)\ell + y \end{pmatrix} \quad \mathbf{J}_{q_d} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \mathbf{J}_{q_i} = \begin{pmatrix} -\sin(\varphi)\ell \\ \cos(\varphi)\ell \end{pmatrix}$$

The dependent coordinates can then be expressed in terms of the independent coordinate as follows:

$$\mathbf{P} = \begin{pmatrix} -\mathbf{J}_{q_d}^{-1} \cdot \mathbf{J}_{q_i} \\ \mathbf{I} \end{pmatrix} = \begin{pmatrix} -\sin(\varphi)\ell \\ \cos(\varphi)\ell \\ 1 \end{pmatrix}$$

and the mass matrix for the minimal coordinates reduces to

$$\begin{aligned} \mathbf{M}_{\text{min}} &= (-\sin(\varphi)\ell \ \cos(\varphi)\ell \ 1) \begin{pmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & \Theta \end{pmatrix} \begin{pmatrix} -\sin(\varphi)\ell \\ \cos(\varphi)\ell \\ 1 \end{pmatrix} \\ &= \ell^2 m\Theta \end{aligned}$$

Note how the mass matrix with respect to minimal coordinates incorporates Steiner's theorem, as the pendulum's motion is a forced rotation about the revolute joint due to the now embedded holonomic constraints and not about the center of mass. The generalized force vector due to gravity

$$\mathbf{Q} = \begin{pmatrix} 0 \\ \boldsymbol{g} \cdot \mathrm{m} \\ 0 \end{pmatrix}$$

expressed in terms of minimal coordinates

$$\mathbf{Q}_{\text{min}} = (-\sin(\varphi)\ell \ \cos(\varphi)\ell \ 1) \begin{pmatrix} 0 \\ \boldsymbol{g} \cdot \mathrm{m} \\ 0 \end{pmatrix} = \boldsymbol{g} \cdot \mathrm{m} \cdot \cos(\varphi)\ell$$

is equal to the previously derived force vector from [SECTION 3.1.1].

State-Space Formulation of the Dynamic Equations of Motion

In the next step, the non-holonomic constraints shall be eliminated, and a state-space formulation shall be derived, by adapting the derivation of Klančar et al. (2017, pp. 49). Non-holonomic constraints result in dependencies along the generalized velocities, which can be likewise reduced by introducing a set of minimal (independent) pseudo velocities $\boldsymbol{\nu}$. Through kinematic relations, a matrix $\mathbf{S}(\mathbf{q})$ can derived, which maps the pseudo velocities to the generalized velocities for a certain configuration:

$$\dot{\mathbf{q}} = \mathbf{S}\boldsymbol{\nu}$$

After differentiating this relation with respect to time

$$\ddot{\mathbf{q}} = \dot{\mathbf{S}}\boldsymbol{\nu} + \mathbf{S}\dot{\boldsymbol{\nu}}$$

the generalized velocities $\dot{\mathbf{q}}$ and accelerations $\ddot{\mathbf{q}}$ can now be replaced by the pseudo velocity and acceleration terms within the dynamic motion equations (1). Assuming eliminated holonomic constraints and dropping the notion of dependencies on $\mathbf{q}$, $\dot{\mathbf{q}}$ yields

$$\mathbf{M}\dot{\mathbf{S}}\boldsymbol{\nu} + \mathbf{M}\mathbf{S}\dot{\boldsymbol{\nu}} + \mathbf{V} = \mathbf{Q} + \mathbf{A}^T\boldsymbol{\lambda}_n$$

By pre-multiplication with $\mathbf{S}^T$, the Lagrange multipliers can be removed since $\mathbf{S}^T\mathbf{A}^T = 0$:

$$\underbrace{\mathbf{S}^T\mathbf{M}\mathbf{S}}_{\tilde{\mathbf{M}}}\dot{\boldsymbol{\nu}} + \underbrace{\mathbf{S}^T\mathbf{M}\dot{\mathbf{S}}}_{\tilde{\mathbf{V}}}\boldsymbol{\nu} + \underbrace{\mathbf{S}^T\mathbf{V}}_{\tilde{\mathbf{E}}\mathbf{u}} = \mathbf{S}^T\mathbf{Q} + \underbrace{\mathbf{S}^T\mathbf{A}^T}_{=0}\boldsymbol{\lambda}_n \quad (2)$$

Next, the generalized forces are of concern. While the generalized force vector $\mathbf{Q}$ corresponds to the generalized coordinates, the directly actuated inputs $\mathbf{u}$ may or may not differ from the generalized coordinates. For instance, if motors impose torques on the wheels for propulsion, the generalized forces $\mathbf{Q}$ can be expressed in

terms of the actuated inputs $\mathbf{u}$. The input vector $\mathbf{u}$ is hereafter mapped with matrix $\mathbf{E}$ from the actuation space to the generalized forces $\mathbf{Q}$ in the configuration space via

$$\begin{aligned} \mathbf{Q} &= \mathbf{E}\mathbf{u} \quad | \cdot \mathbf{S}^T \\ \mathbf{S}^T\mathbf{Q} &= \mathbf{S}^T\mathbf{E}\mathbf{u} = \tilde{\mathbf{E}}\mathbf{u} \end{aligned}$$

such that the equation (2) and its rearrangement for $\dot{\boldsymbol{\nu}}$ can be written as

$$\begin{aligned} \tilde{\mathbf{M}}\dot{\boldsymbol{\nu}} + \tilde{\mathbf{V}} &= \tilde{\mathbf{E}}\mathbf{u} \\ \dot{\boldsymbol{\nu}} &= \tilde{\mathbf{M}}^{-1}\left(\tilde{\mathbf{E}}\mathbf{u} - \tilde{\mathbf{V}}\right) \end{aligned}$$

From the previous equations, finally the state space form with $\mathbf{x} = (\mathbf{q}^T\boldsymbol{\nu}^T)^T$ as the state vector can be derived:

$$\dot{\mathbf{x}} = \begin{pmatrix} \dot{\mathbf{q}} \\ \dot{\boldsymbol{\nu}} \end{pmatrix} = \begin{pmatrix} \mathbf{S}\boldsymbol{\nu} \\ -\tilde{\mathbf{M}}^{-1}\tilde{\mathbf{V}} \end{pmatrix} + \begin{pmatrix} 0 \\ \tilde{\mathbf{M}}^{-1}\tilde{\mathbf{E}} \end{pmatrix}\mathbf{u}$$

The application of these equation to some common wheeled robots will be demonstrated in the next section.

Self-Check Questions

9. In which two ways can the Lagrange multipliers be dealt with in the dynamic equations of motion?

Using the augmented formulation, the reaction forces are calculated determined by additionally solving the constraint equations, while the embedding technique eliminates the Lagrange multipliers by projection to a minimal set of generalized position and velocity coordinates.

10. What is the difference between the pseudo and the generalized velocities?

Generalized velocities are the time derivatives of the chosen generalized coordinates. The pseudo velocities can be a different set of generalized velocities practically containing only independent entries.

3.3 Examples

In the sequel, the dynamic equations for a wheeled mobile robot are derived based on the Lagrangian formulas. For a simple example, the forward dynamics shall be simulated for a short period of time, to demonstrate how the components of the formula work together.

Two-Wheeled Differential Drive Robot

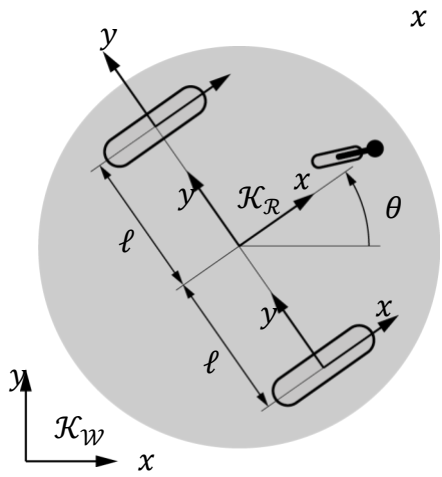
The dynamic motion model of the differential drive robot shown in [FIGURE] is derived based on the kinematic model that was already developed in the last unit. For the derivation, it is assumed that the wheels are mass-less, and the robot's reference frame coincides with the center of mass. Furthermore, the robot is moving parallel to the ground plane, such that gravitational forces will be neglected as well, and disturbances are assumed not to be present. The configuration of the robot will be described by the generalized coordinates $\mathbf{q}$ that are only containing the parameters of the robot pose $\boldsymbol{\xi} = (x, y, \theta)^{\mathsf{T}}$.

To this end, the individual matrices within the dynamic equation

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{V}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{Q} + \mathbf{A}^{T}(\mathbf{q})\boldsymbol{\lambda}_{n}$$

shall be determined using the Lagrange formula.

Geometric Properties of the Differential Drive Robot



Source: Florian Simroth (2023).

The two constraint equations arising from the non-slip condition were redundant, such that only one independent constraint equation remains:

$$\begin{aligned}(0 \quad 1 \quad 0) \cdot {}^{\mathcal{R}}\mathbf{R}_{\mathcal{W}}(\theta) \cdot \dot{\boldsymbol{\xi}} &= (0) \\ -\sin(\theta)\dot{x} + \cos(\theta)\dot{y} &= 0\end{aligned}$$

Due to this non-holonomic constraint, only two of the three generalized velocities are independent. Similar to the embedding technique, one could try to express one dependent velocity in terms of the independent velocities, i.e., when choosing $\dot{x}$ and $\dot{\varphi}$ as the independent velocities, from the equation above follows $\dot{y} = \tan(\varphi)\dot{x}$ and one could choose the pseudo velocities $\boldsymbol{\nu} = (\dot{x}, \dot{\varphi})$ such that the kinematic model $\dot{\mathbf{q}} = \mathbf{S}\boldsymbol{\nu}$ would yield

$$\underbrace{\begin{pmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{pmatrix}}_{\dot{\mathbf{q}}} = \underbrace{\begin{pmatrix}1 & 0\\ \tan(\theta) & 0\\ 0 & 1\end{pmatrix}}_{\mathbf{S}} \underbrace{\begin{pmatrix}\dot{x}\\ \dot{\theta}\end{pmatrix}}_{\boldsymbol{\nu}}$$

Clearly, this would introduce a singularity for $\theta = \pi/2$ which leads to unfavorable numerics. Another choice are the angular wheel velocities, through which the robot is driven. Yet, for example for path planning, it is favorable to choose the independent pseudo velocities as $\boldsymbol{\nu} = (v, \omega)$, the velocity v in local x-direction as well as the angular velocity ω of the robot. In fact, this kinematic model applies to several

wheeled mobile robots such that it can be reused. Therefore, the following kinematic model for the differential robot will be used:

$$\underbrace{\begin{pmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{pmatrix}}_{\dot{\mathbf{q}}} = \underbrace{\begin{pmatrix}\cos(\theta) & 0\\ \sin(\theta) & 0\\ 0 & 1\end{pmatrix}}_{\mathbf{S}} \underbrace{\begin{pmatrix}v\\ \omega\end{pmatrix}}_{\boldsymbol{\nu}}$$

For the derivation of the dynamic model, the Lagrange formula will be determined. Since the center of mass is assumed to coincide with the robot's coordinate system, the mass matrix only has entries on its diagonal.

$$\mathbf{M} = \begin{pmatrix} m & 0 & 0\\ 0 & m & 0\\ 0 & 0 & \Theta \end{pmatrix}$$

Due to the planar movement of the robot, the potential energy is constant and with its datum set to the robot's coordinate system it is zero and only the kinetic energy remains to be determined:

$$\begin{aligned}\mathcal{L} &= \mathcal{T} - \mathcal{U}\\ \mathcal{L} &= \frac{1}{2}\dot{\mathbf{q}}^T\mathbf{M}\dot{\mathbf{q}} - 0 = \frac{1}{2}(\dot{x}\ \dot{y}\ \dot{\varphi})\begin{pmatrix} m & 0 & 0\\ 0 & m & 0\\ 0 & 0 & \Theta \end{pmatrix}\begin{pmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{pmatrix}\\ &= \frac{1}{2}\left(\dot{x}^2 m + \dot{y}^2 m + \dot{\theta}^2\Theta\right)\end{aligned}$$

The derivatives of the Lagrange function are:

$$\frac{d}{dt}\begin{pmatrix}\partial\mathcal{L}/\,\partial\dot{x}\\ \partial\mathcal{L}/\,\partial\dot{y}\\ \partial\mathcal{L}/\,\partial\dot{\theta}\end{pmatrix} = \frac{d}{dt}\begin{pmatrix}\dot{x}m\\ \dot{x}m\\ \dot{\theta}\Theta\end{pmatrix} = \begin{pmatrix}\ddot{x}m\\ \ddot{y}m\\ \ddot{\theta}\Theta\end{pmatrix} \quad \text{and} \quad \begin{pmatrix}\partial\mathcal{L}/\,\partial x\\ \partial\mathcal{L}/\,\partial y\\ \partial\mathcal{L}/\,\partial\theta\end{pmatrix} = 0$$

From the derivatives it follows, that there are no dynamic components present depending on the velocity, such that

$$\mathbf{V} = 0$$

The generalized forces acting on the robot are due to the torques generated at the drive wheels. One can now either derive a term for the generalized forces generated by input torques on the wheels by hand, or recall the dual character from velocity and force and derive the generalized forces from the pure rolling constraints.

Recognizing that the power P as the product of the generalized velocities and the forces $P = \dot{\mathbf{q}}^T\mathbf{Q}$ is equal to the wheel velocities $\dot{\boldsymbol{\varphi}} = (\dot{\varphi}_\ell\ \dot{\varphi}_r)^T$ times wheel torques $\mathbf{u} = (\tau_\ell\ \tau_r)^T$ yielding $P = (\dot{\varphi}_\ell\ \dot{\varphi}_r)^T\mathbf{u}$, one can express the generalized forces $\mathbf{Q}$ in terms of the input wheel torques $\mathbf{u}$

$$\begin{aligned}\dot{\mathbf{q}}^T\mathbf{Q} &= \dot{\boldsymbol{\varphi}}^T\mathbf{u} \quad \textbf{with} \quad \dot{\boldsymbol{\varphi}} = \mathbf{J}\dot{\mathbf{q}}\\ \dot{\mathbf{q}}^T\mathbf{Q} &= (\mathbf{J}\dot{\mathbf{q}})^T\mathbf{u} = \dot{\mathbf{q}}^T\mathbf{J}^T\mathbf{u}\\ \mathbf{Q} &= \underbrace{\mathbf{J}^T}_{\mathbf{E}}\mathbf{u}\end{aligned}$$

Where the Jacobian $\mathbf{J}$ stems from the pure rolling constraints:

$$\begin{aligned}\begin{pmatrix}0\\0\end{pmatrix} &= \begin{pmatrix}1 & 0 & -\ell\\ 1 & 0 & \ell\end{pmatrix}\cdot {}^{\mathcal{R}}\mathbf{R}_{\mathcal{W}}(\theta)\cdot\dot{\boldsymbol{\xi}} - \begin{pmatrix}r & 0\\ 0 & r\end{pmatrix}\cdot\begin{pmatrix}\dot{\varphi}_\ell\\ \dot{\varphi}_r\end{pmatrix}\\ \begin{pmatrix}\dot{\varphi}_\ell\\ \dot{\varphi}_r\end{pmatrix} &= \frac{1}{r}\underbrace{\begin{pmatrix}\cos(\theta) & \sin(\theta) & -\ell\\ \cos(\theta) & \sin(\theta) & \ell\end{pmatrix}}_{\mathbf{E}^T}\begin{pmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{pmatrix}\end{aligned}$$

At this point, all matrices describing the dynamic model of the differential drive robot are known and the state space form can be derived using $\mathbf{S}$:

$$\mathbf{S} = \begin{pmatrix}\cos(\theta) & 0\\ \sin(\theta) & 0\\ 0 & 1\end{pmatrix}$$

The model matrices and their projections to the pseudo velocities are hereafter summed up:

$$\begin{aligned}
\mathbf{M} &= \begin{pmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & \Theta \end{pmatrix} & \widetilde{\mathbf{M}} &= \mathbf{S}^T\mathbf{M}\mathbf{S} = \begin{pmatrix} m & 0 \\ 0 & \Theta \end{pmatrix} \\
\mathbf{V} &= 0 & \widetilde{\mathbf{V}} &= \mathbf{S}^T\mathbf{M}\dot{\mathbf{S}}\boldsymbol{\nu} + \mathbf{S}^T\mathbf{V} = 0 \\
\mathbf{E} &= \frac{1}{r}\begin{pmatrix} \cos(\theta) & \cos(\theta) \\ \sin(\theta) & \sin(\theta) \\ -\ell & \ell \end{pmatrix} & \widetilde{\mathbf{E}} &= \mathbf{S}^T\mathbf{E} = \frac{1}{r}\begin{pmatrix} 1 & 1 \\ -\ell & \ell \end{pmatrix}
\end{aligned}$$

Next, the remaining terms of the state space model are established:

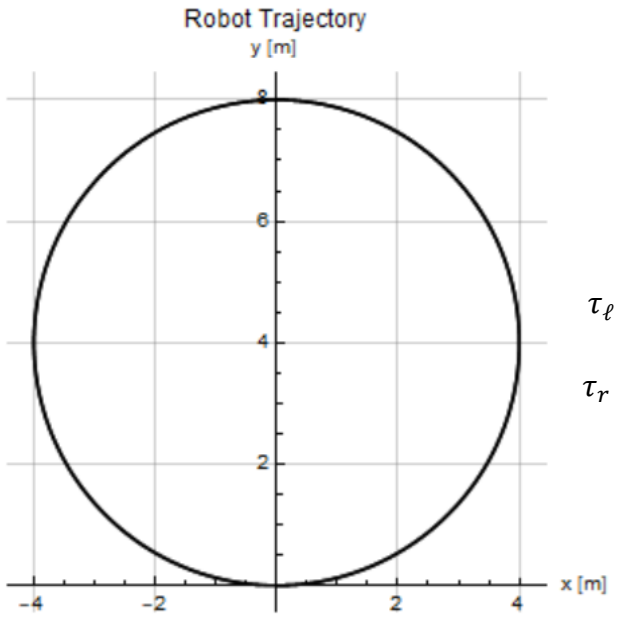
$$\begin{aligned}
\dot{\mathbf{q}} &= \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix}, \ \boldsymbol{\nu} = \begin{pmatrix} v \\ \omega \end{pmatrix} \\
\mathbf{S}\boldsymbol{\nu} &= \begin{pmatrix} v\cos(\theta) \\ v\sin(\theta) \\ \omega \end{pmatrix} \\
-\widetilde{\mathbf{M}}^{-1}\widetilde{\mathbf{V}} &= \begin{pmatrix} 0 \\ 0 \end{pmatrix} \\
\widetilde{\mathbf{M}}^{-1}\widetilde{\mathbf{E}} &= \begin{pmatrix} \frac{1}{mr} & \frac{1}{mr} \\ -\frac{\ell}{\Theta r} & \frac{\ell}{\Theta r} \end{pmatrix}
\end{aligned}$$

With these terms, finally the dynamics in state space form can be assembled:

$$\dot{\mathbf{x}} = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ v \\ \omega \end{pmatrix} = \begin{pmatrix} v\cos(\theta) \\ v\sin(\theta) \\ \omega \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{1}{mr} & \frac{1}{mr} \\ -\frac{\ell}{\Theta r} & \frac{\ell}{\Theta r} \end{pmatrix} \begin{pmatrix} \tau_\ell \\ \tau_r \end{pmatrix}$$

These equations describe the dynamic model of the two-wheeled differential drive robot and can be integrated for given input torques to receive the forward dynamics. With the following numerical values, $m = 5\text{kg}$, $\ell = 0.25\text{m}$, $\Theta = 1\text{kg m}^2$, $r = 0.05\text{m}$, and constant input torques $\tau_\ell = 0.1\text{Nm}$ $\tau_r = 0.15\text{Nm}$ and a simulation time of 10 seconds, the circular trajectory shown in [FIGURE] is obtained.

Trajectory of Differential Drive Robot For Constant, Non-Equal Torque Inputs



$\tau_\ell = 0.1\text{Nm}$

$\tau_r = 0.15\text{Nm}$

Source: Florian Simroth (2023).

Self-Check Questions

11. How can the kinematic model $\dot{\mathbf{q}} = \mathbf{S}\boldsymbol{\nu}$ for the derivation of the dynamic model be established for the two-wheeled differential robot?

    Either by manual analysis of the model or by exploiting the constraints of the non-slip constraint.

12. How can the **E** matrix be derived for the differential robot?

    Either by manual analysis of the model or by using the pure rolling constraints of the differential robot.

Summary

The dynamic analysis of a mobile robot considers its kinematic model, i.e., the motions that the robot is capable to perform under the given kinematic constraints, as well as forces and torques acting on the robot. These forces emerge from the robot's motion itself, such as the Coriolis or centrifugal forces, arise from damping or friction, or act upon the robot due to gravity or actuators. In forward dynamics, the resulting motion of the robot is predicted based on the external forces. In inverse dynamics, the required forces are derived for the robot to follow a predefined trajectory.

In addition to the geometric properties, also the mass distribution takes an important role which is gathered in a generalized mass matrix whose form depends on the choice of generalized coordinates. In particular, the mass properties as well as generalized forces are expressed corresponding to the generalized coordinates.

To derive the dynamic equations of motion, two methods are prominently used: The Newton-Euler and the Lagrange method. Here, the Lagrange method is used to derive the dynamics in terms of generalized forces, which is suitable for systems with only a few degrees of freedom. For the Lagrange method it is necessary to calculate the kinetic and potential energy of the system. If constraints are present, the reaction forces are determined in form of Lagrange multipliers. By choosing a minimal set of position and velocity coordinates, and expressing the generalized coordinates in these terms, the Lagrange multipliers can be eliminated.

In the end, the terms of the Lagrange function are gathered in matrix form, from which a state space model is derived. Through integration of this model, the motion of a mobile robot can be simulated based on input forces and torques.

Unit 4 – Perception

Study Goals

On completion of this unit, you will be able to …

…describe different physical principles used for sensors.

… decide which sensor best suits the needs of a particular mobile robot.

… identify erroneous sensor outputs induced by the measurement method.

… understand how mobile robots perceive their environment.

Unit 4 – Perception

Introduction

Humans perceive their environment through their five senses of sight, touch, hearing, taste, and smell with which they are able to perform even the most complex tasks. Similarly, a mobile robot needs to navigate through its environment, prevent collisions, and avoid dangerous situations, and has to be able to interact with its environment to perform the tasks it is designed for. For this reason, mobile robots require sensors as an interface between their physical surroundings and the electronics/software. These sensors make use of a variety of physical principles in order to "translate" physical properties into electronic signals which can be processed by the algorithms.

This chapter provides an overview of

- Common sensors used for mobile robot perception
- Properties of sensors found on datasheets
- Physical principles used to perceive the environment

4.1 Sensors for Mobile Robots

After defining the tasks for a mobile robot, an appropriate set of sensors needs to be selected to enable the robot to maneuver and accomplish its purpose. The choice of sensors does not only significantly impact costs, but also decides whether the robot can accurately and reliably succeed in performing its tasks. Hereafter, some properties of sensors are discussed.

Ground truth

The ground truth resembles the exact information/value in the absence of any measurement errors.

Accuracy

The accuracy provides a measure of how close the measured value is to the **ground truth** value. To determine the accuracy of a device, a reference is required,

established, for instance, by another measurement device that is at least one order more exact than the device being tested.

Precision

The precision is an indicator of the extent of the spread of repeated measurements of the same physical property under the same environmental conditions. That is, if a certain distance is measured multiple times with an ultrasonic sensor, the variance or standard deviation of measured values represents the precision.

Resolution

The smallest offset of a physical quantity that can be measured is considered the resolution of a sensor. For instance, typical rulers used in school have a scale with tick marks which are spaced at equal distances of one millimeter. The smallest differentiable distance that can be read is one millimeter, thus, the resolution.

Range

The range is defined by the difference of the minimum and maximum physical quantity that can be measured by the device.

Frequency

Digital sensors, in particular, provide discrete values that are sampled at a certain rate. The higher this frequency, the more measurement values can be provided in the same time period.

Environmental working conditions

Datasheets provide information on the conditions under which a valid sensor output can be expected. Limiting conditions often include a temperature range, pressure/shock resistance, and more.

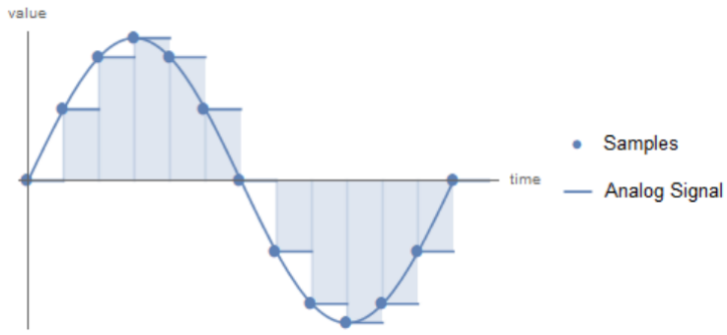
Types of sensors

There is a variety of sensor types available which can be categorized in different manners.

Classification by type of output signal

Regarding the output signal, sensors can be classified into digital and analog sensors. Analog signals, for instance, in the form of a varying voltage, contain a continuous stream of information. These signals are prone to interference from other sources which may alter the signal itself on its way from the sensor to the receiving end. At the receiving end, so called A/D (Analog to Digital) converters are able to translate analog into digital signals by sampling the analog into discrete values at a certain frequency (see [FIG]).

An Analog Signal Is Discretized by Sampling the Analog Signal at a Certain Frequency



Source: Florian Simroth (2023).

Digital signals, on the other hand, contain discrete values, provided with a certain frequency. These signals can be transmitted either in a serial or parallel fashion. For serial transmission, values are sent in the form of a sequential bit stream which,

according to specific protocols (such as I2C, or Inter-Integrated-Circuit), are then re-assembled to values at the receiving end.

A parallel transmission utilizes multiple lines in parallel. All lines synchronously transmit bits which can be directly translated into values. In this manner, more data can usually be sent in a certain time period, though the range of values that can be represented are generally dependent on the number of parallel lines.

Classification as Active/Passive

Depending on how the sensor interacts with its environment, the sensor can be regarded as active or passive. Passive sensors, such as a camera, gather information without changing or interfering with the environment. These sensors usually depend on external environmental conditions – for the case of a camera, the scene needs to be illuminated for the camera to be able to perceive the surroundings. Active sensors emit a signal and observe how this signal interacts with the environment. In the case of a Lidar, laser beams are emitted and the time difference between the emission and reception of an echo will reveal the distance to an obstacle.

Classification By Type of Physical Principle

Sensors make use of many different physical principles to observe certain effects. A sensor includes a **transducer** unit which "translates" a physical property into an electrical signal that afterwards is processed into a reading of the desired sensor output. Based on the underlying physical properties used by the transducer unit, it is possible to categorize sensors into the following:

Transducer

A transducer translates a physical property into an electronic signal.

- Optical
- Mechanical
- Magnetic
- Inductive
- Acoustical
- Electromagnetic

Self-Check Questions

13. Please complete the following sentence.

With an *A/D converter*, analog signals can be converted into digital signals.

14. What is the difference between active and passive sensors?

Active sensors emit a signal and observe the interaction pattern with the environment while passive sensors directly observe environmental conditions.

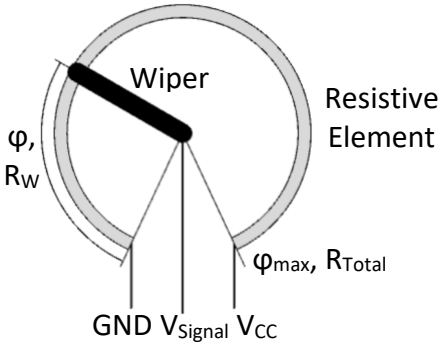
4.2 Position and Velocity Sensors

In this section, position and velocity sensors are of concern, which provide feedback on the robot's configuration, i.e., the joint positions connecting the rigid links between the mobile platform and the end effector, as well as the position and orientation of the mobile platform itself. In particular, by tracking the robot's wheel revolutions over time, statements about the pose can be derived with respect to its start point. Commonly used position and velocity sensors are potentiometers, resolvers, encoders (mechanical (brush encoders), optical, magnetic, electromagnetic induction), and tachometers.

Potentiometer

A potentiometer makes use of the fact that the electrical resistance of a conductor increases with its length. It is set up with three terminals, of which two (here labeled GND for ground and V_{CC} for reference voltage) are connected to the ends of a resistive element, while the third one, termed wiper, slides along the resistive element as depicted in [FIG].

Schematic of a Potentiometer With an Output Resistance Depending On the Position of a Wiper



Source: Florian Simroth (2023).

Depending on the angle by which the wiper is displaced, the resistance between the wiper and respective terminal changes. Using all three terminals, the potentiometer can be operated as a voltage divider (see [EQ1]), such that the signal voltage can be measured and converted to the corresponding angle, as shown in [EQ2]. Note that the formula only holds for potentiometers with a linear resistive element, which needs to be adapted accordingly for potentiometers with a logarithmic resistive characteristic.

$$V_{signal} = \frac{R_W}{R_{total}} \cdot V_{CC} = \frac{\varphi}{\varphi_{max}} \cdot V_{CC}$$

$$\varphi = \frac{V_{signal}}{V_{CC}} \cdot \varphi_{max}$$

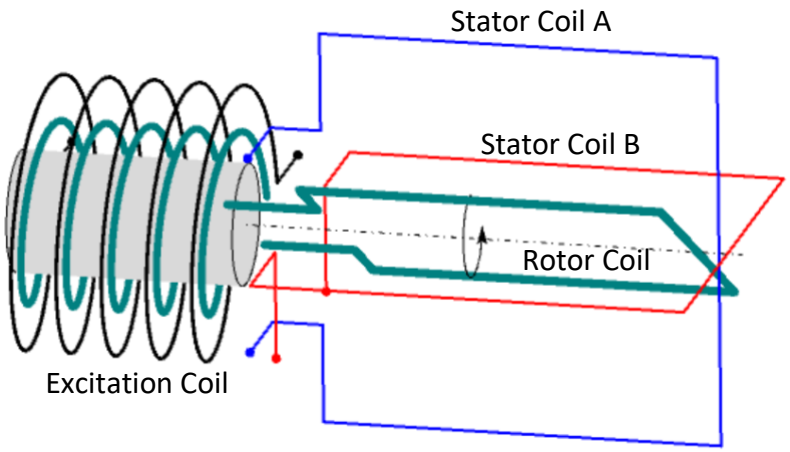
It is worth mentioning that both angular and linear versions of potentiometers for measuring angles and linear displacements exist. As the resistance is a continuous function of the angle or distance, the potentiometer is able to directly measure the absolute position. Servos, for example, usually use integrated potentiometers to actuate the output lever to the intended angle. However, potentiometers are usually restricted to a maximum angle as the length of the resistive element is limited.

Resolvers

Resolvers use electromagnetic induction to measure the position and velocity of a rotating shaft. An alternating current is applied to a coaxial excitation coil fixed to the

non-rotating stator which, in turn, induces a current to the coaxial counterpart fixed to the rotating shaft. Through this current, an alternating electromagnetic field builds up in the rotor coil (see [FIG]).

Schematic of a Resolver Measuring the Angle of a Rotor Shaft



Source: Florian Simroth (2023).

This alternating field then induces currents within the secondary coils of the stator which are mounted with a 90° offset to one another. If the rotor coil is in alignment with one of the secondary stator coils, the alternating current is only induced in the aligned coil, while no current is induced in the perpendicular secondary coil. If the rotor coil continues to rotate, the magnitudes of currents shift from one secondary coil to the other. From the relative magnitudes of the alternating current signals, the rotation angle can then be determined.

Since there is no mechanical contact between stator and rotor, resolvers operate in a wear-free manner. They work reliably and are robust to external influences. On the other hand, they are relatively bulky and expensive. Furthermore, if the signal is designated for a mobile robot, it will most likely be required in digital form. That makes an additional circuit and processing logic necessary to extract rotation angles.

Encoders

Encoders are usually used as axis or motor sensors and, in contrast to potentiometers, are not limited by a maximum rotation angle. In general, there exist both linear and angular encoders, with angular encoders the prevailing type for mobile robots. Independent of the underlying physical principle, two types of encoders are differentiated:

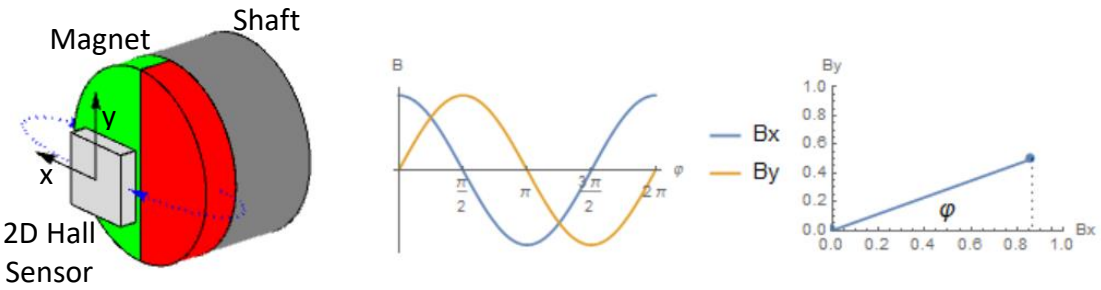
1. **Absolute encoders** directly measure the current angle or distance and can return valid information instantly after powering on.
2. **Incremental encoders** have no information about the current angle or distance after start-up, but, rather, count angle or distance increments during motion. Often, a reference indicator is used to determine the zero-position and, once this mark has been detected, the absolute angle can be derived by accumulating the number of increments. This implies that the motion has to be continuously monitored since missing one increment would lead to an incorrect absolute angle or distance reading.

Encoders can take advantage of a great variety of physical principles, each of them implying specific advantages and disadvantages. The commonly used principles are presented below.

Magnetic encoders

Magnetic encoders usually use “hall sensors” to detect changes in a rotating magnetic field. For typical “on-axis” design variants, a permanent magnet is attached to the end of an axis and the rotation of the magnetic field is measured. A single hall sensor only measures the ”magnetic flux density” along one axis, such that, for each reading (excluding the reversal points), there are two possible shaft angles. To encode a full 360° rotation, two hall sensors arranged at a 90° angle are required; they are usually integrated in a single housing.

Schematic of a Magnetic Encoder Composed of a Permanent Magnet and a Hall Sensor



Source: Florian Simroth (2023).

If the shaft rotates continuously as shown in [FIG], the magnitudes of the magnetic flux densities B_x and B_y in the respective hall sensors form sine waves with a 90° phase shift. From these signals, the corresponding shaft angle φ can be determined via [EQ], either with an external microcontroller or with the algorithms implemented on an integrated circuit.

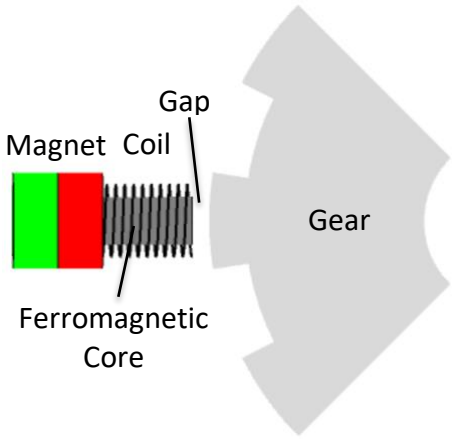
$$\varphi = \arctan2(B_y, B_x)$$

As the angle is directly derived from the two signals, the magnetic encoder can be regarded an absolute encoder. Due to the design and physical principle, they are robust and can be deployed in a variety of environments, even in the presence of dust or liquids. As there is no physical contact, these sensors are wear-free and can be built in a lightweight and small manner if the logic is integrated in a single chip. Next to the described “on-axis” design, there also are versions where the magnet is not mounted at the tip of the axis but, instead, is radially attached, leading to an “off-axis” design.

Inductive encoders

Inductive encoders use induction to measure the proximity of ferrous objects. This makes them suitable to measure, for instance, the angular velocity of gears and are present in most anti-lock braking systems (ABS). There are basically two components: 1) a variable reluctance sensor and 2) the electronics that derive information, such as angular velocity or position information. The variable reluctance sensor contains a coil enclosing a ferrite core with an attached permanent magnet which generates a magnetic field.

Schematic of an Inductive Encoder Sensing Variations in the Magnetic Flux Density



Source: Florian Simroth (2023).

If arranged according to [FIG], the air gap between the ferrite core and the gear oscillates with each passing gear tooth. This influences the magnetic flux density and, in turn, induces an alternating current whose frequency directly corresponds to the angular velocity. The magnitude of the current relates to the size of the gap. With the additional electronics, the signal can be further processed to return an impulse for each passing gear, forming an incremental encoder. In addition, by marking one tooth, for instance, by removing or attaching a magnet, an absolute signal can be

provided by counting the number of teeth with respect to the index. The downside of this method is that, at first, the index tooth must be detected by some motion before an absolute signal can be provided and, furthermore, continuous counting of impulses must be ensured. Besides, inductive encoders are robust against external impacts, work in rough environments with dust and liquids, and are free of wear.

Capacitive encoders

Capacitive encoders basically determine position by altering the capacity of capacitors by inserting a moving dielectric medium. The capacity of a capacitor can be determined by [EQ], with the permittivity ε_0 in vacuum, the permittivity ε_r of the dielectric medium relative to vacuum, A the area, and d the distance between the capacitor plates.

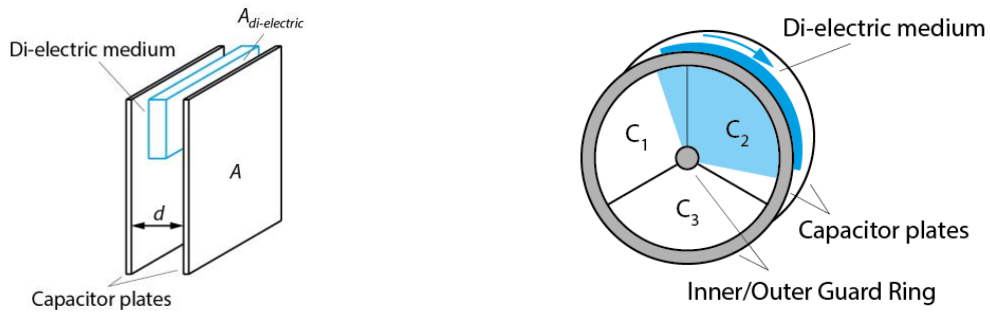
$$C = \varepsilon_0 \varepsilon_R \frac{A}{d}$$

A capacitor with a partially inserted dielectric medium, as shown in [FIG a)], can be treated like two parallel capacitors, one with and one without the dielectric medium, whose capacities are determined independently with corresponding areas according to [EQ].

$$C = \varepsilon_0 \frac{A - A_{dielectric}}{d} + \varepsilon_0 \varepsilon_R \frac{A_{dielectric}}{d}$$

A simplified version of a capacitive encoder is shown in [FIG b)], with three capacitors plate pairs and a di-electric medium whose size fits one of the capacitor plates and which is attached to the rotating shaft. If the shaft rotates, the capacitors are successively partially “filled” with the di-electric medium. Now the capacity of each capacitor can be measured and the coverage fraction derived. With the known arrangement of the three capacitors, the absolute angle of the shaft can be obtained. More details on this simplified capacitive encoder can be found in Das et al. (2018).

Plate Capacitor With Dielectric Medium (a) and Schematic of a Simplified Capacitive Encoder (b)



Source: Florian Simroth (2023).

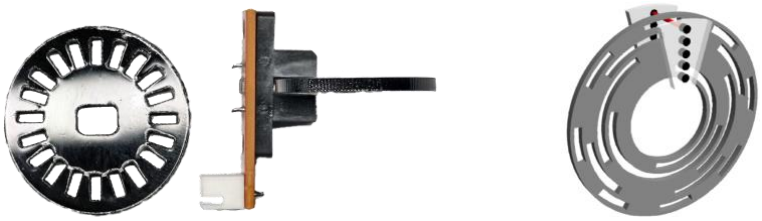
Usually, though, capacitive encoders are set up in a more complicated fashion, where a sinusoidal pattern is etched onto a rotor disc which modulates a high frequency signal applied to the capacitors in a specific way, enabling the derivation of a rotation angle. More details can be found in Zheng et al. (2014). In general, high resolutions can be reached with capacitive encoders which are also wear-free. On the other hand, they are susceptible to electromagnetic fields or temperature fluctuations, which impact the capacities and, therefore, need to be compensated. In addition, protection anodes (guard rings) are required to avoid static charge.

Optical encoders

Optical encoders consist of three main components: 1) a light-emitting diode, 2) a translucent or reflective encoder disc, and 3) a photodiode. The underlying physical principle is similar to a light barrier. The encoder disc is equipped with translucent and opaque segments which segmentally interrupt a light beam. This pattern can then be used to determine the incremental or absolute rotation angle or translational displacement, whereby two different designs come into play. The incremental encoder disc design is quite simple, consisting of evenly distributed gaps creating light impulses at the photodiode which can be measured and then transformed into a digital signal (see [FIG a)]). The resolution corresponds to the number of gaps

distributed along the encoder disc (or strip). With a second emitter receiver pair mounted with an offset of half a gap size, the rotation direction can also be determined, depending on which emitter receiver pair first spots the gap. Since only incremental steps are provided, a homing cycle, which brings the system into a known reference state is required from which the steps can be accumulated to derive an absolute reading.

Low-Cost Optical Incremental Encoder (a) and Schematic of an Absolute Encoder (b)



Source: Florian Simroth (2023).

An absolute angle encoder disc comprises multiple tracks, each with a different pattern of opaque and translucent segments (see [FIG b)]). From the combination of the binary state of all tracks, an absolute position can also be derived directly after start-up. In its simplest from, each track can be thought of as a single digit of a binary number with two states depending on whether the light beam passes or is blocked by the current track segment. A disc with four tracks (4-bits) could encode $2^4 = 16$ different angles. Clearly, for a decent resolution, a corresponding high number of tracks is required. Commonly, the tracks are encoded using Gray code, named after its patent holder Frank Gray, which is less error prone when reading consecutive angle values.

Advantages of optical encoders are the high accuracy and low costs, while, on the other hand, the whole set up is influenced by (and vulnerable to) its surroundings. As such, dust may disturb the optics or shocks and vibrations may lead to small displacements of the encoder disc or photo elements yielding erroneous values.

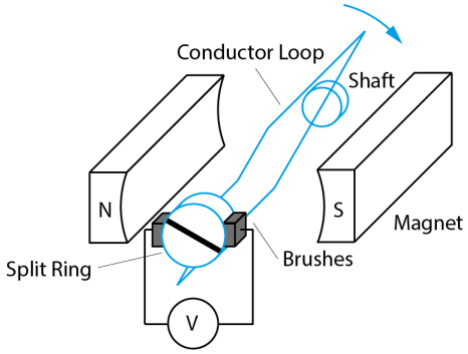
Mechanical Brush Encoders

Bruch encoders use a similar encoder scheme as optical encoders regarding distinct tracks on which absolute or incremental distances or angles are encoded. Instead of a disc with opaque and translucent segments on tracks already described within the last section, there are conducting and non-conducting segments. On the fixed counterpart, there are sliding contacts that close an electric circle once an conductive segment is passed. The resulting downsides of mechanical encoders are wear and corrosion of the contacts, limiting the lifetime and environmental impacts due to dust and liquids. Furthermore, the speed is limited and sensors may not be used in environments with explosive gases due to potential spark formation.

Electrical Tacho Generator

Tacho generators are used for velocity measurement and provide an output voltage proportional to the angular velocity. There are two variants: 1) a direct current (DC) and 2) an alternating current (AC). The principle of the direct current type is shown in [FIG] with a single conductor loop rotating in the magnetic field of a fixed permanent magnet. As the magnetic flux through the area enclosed by the conductor loop changes with the rotation angle, a measurable voltage is induced. Through a split ring and brushes, the alternating voltage is commutated and the resulting polarity indicates the direction of rotation. In real applications, there is not just one conductor loop but multiple coils and a corresponding number of split ring segments. A disadvantage of the DC tacho generator are ripples in the output voltage induced by the commutator, which particularly exacerbate measurements at low speeds, and mechanical wear.

DC Tacho Generator Returning the Angular Velocity



Source: Florian Simroth (2023).

AC tacho generators, on the other hand, do not use the commutator and, therefore, are ripple- and wear-free. The components of the stator and rotor are switched, such that a permanent magnet is now attached to the rotating shaft, while the coils are on the stationary part. The rotating magnetic field now induces a voltage in the stator coils which is rectified by a simple rectifier circuit. The angular velocity is then proportional to frequency and magnitude.

As the name indicates, tacho generators do not need an external power supply but rather "generate" a measurable voltage. Contrarily, for very slow motions, not enough power is generated to achieve an accurate measurement, which is one of the reasons resolvers evolved.

Self-Check Questions

15. What is the difference between absolute and incremental encoders?

Absolute encoders provide the current position directly after startup. Incremental encoders only provide increments with respect to the initial position or require a "homing" cycle to set a reference.

16. Mark the correct statements

☐ Mechanical sensors are free of wear

☐ Optical encoders should be preferred in dusty environments

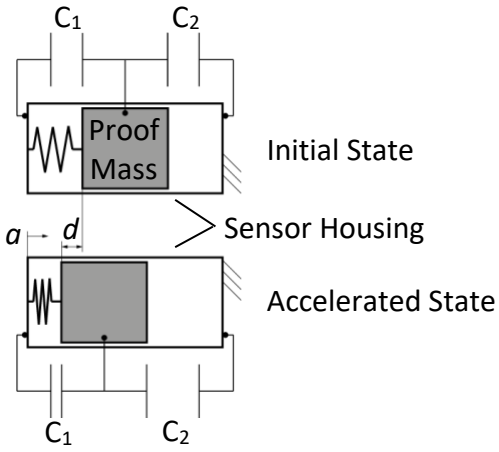
☐ Optical encoders are known to be resistant to vibration

☐ *Resolvers are robust against shocks and vibration*

4.3 Accelerometers

Accelerometers measure the acceleration which a body is subjected to. In its most basic form, an accelerometer measures the displacement of a proof mass, which is hinged to the frame whose acceleration shall be measured with a spring (see [FIG]).

Schematics of an Accelerometer



Source: Florian Simroth (2023).

According to Newton's first law, the probe mass would remain in its state of rest unless an external force is applied. If the outer frame is accelerated with acceleration a, a force $F = m \cdot a$ is required to accelerate the proof mass m equally. This force is exerted to the proof mass by compressing the spring with stiffness k by distance d to re-establish equilibrium according to Hooke's law $F = k \cdot d$. From this equilibrium condition, the acceleration of the frame can be derived as in [EQ].

$$a = \frac{k \cdot d}{m}$$

Thus, for known stiffness and probe mass, the only thing remaining for the accelerometer is to measure the displacement for which various methods are available. Nowadays, accelerometers are realized as micro-electro-mechanical systems (MEMS) on a chip. Widely used capacitive accelerometers determine the

displacement of a proof mass by measuring the capacity change it induces by altering the gap of the plates of a capacitor. Two of the capacitor electrodes are fixed to the frame, while a third electrode located in between the fixed electrodes is attached to the movable proof mass. In theory it would be sufficient to measure the capacity between the proof mass electrode and one of the fixed electrodes, though, through the differential set up, a linear voltage output can be achieved by a synchronous demodulation technique. Often, also two or three sensitive axes are integrated into one chip which allows measuring accelerations in three-dimensional space.

Self-Check Questions

17. Which principle do accelerometers usually rely on?

The inertia of a probe mass allows the sensor to indirectly measure acceleration.

18. What does MEMS stand for?

Micro-electro-mechanical system

4.4 Inertial Measurement Unit

For many tasks, it is necessary for a mobile robot to precisely determine its ego-motion and pose, i.e., its own movement, position, and orientation with respect to the environment. Often, for wheeled robots, the odometry acquired by encoders attached to the wheels gives a first indication on the robot's position and velocity with respect to a starting point, but usually suffers from drift and accumulated errors. Also, for other types of robots such legged or airborne robots, this kind of odometry is not available at all.

In both cases, inertial-measurement-unit (IMU) provide valuable additional information, such as angular velocities or accelerations. Besides, in the robotics domain, they are also deployed in aviation, automotive, or consumer electronics. IMUs are comprised of accelerometers and gyroscopes, measuring accelerations and angular velocities along one or more axes. For mobile robots, mainly miniaturized

MEMS, as shown in [FIGURE] are put into use, but even today smaller airplanes, for example, make use of bulky, physical IMUs with rotating gyroscopes.

MEMS of a 6 DOF IMU (Three Sensitive Axes for Accelerometer and Gyroscope Respectively)

Source: Florian Simroth (2023).

The operating principle of accelerometers was already discussed in [SECTION ACCELEROMETERS]. It is worth mentioning, that the measured accelerations are often adduced to determine the inclination (pitch and roll angle) with respect to ground. To this end, the components of the gravitational force vector are measured along the sensitive axes of the accelerometer, from which the tilt can be derived by simple geometry. This can, of course, only be done if no other accelerations are present or appropriate filtering is applied.

The angular velocity is measured using the Coriolis force exerted on an oscillating proof mass embedded into a MEMS. If the proof mass that oscillates in one direction is being rotated about an axis perpendicular to the oscillation axis, a Coriolis force is exerted along the axis normal to the previous ones. Like for the accelerometers, this force can be measured in form of a displacement of the proof mass and change of connected capacitor capacities.

Some IMUs also include a magnetometer to measure the current heading based on the earth's magnetic field. As the magnetic field is influenced by ferromagnetic materials present in the environment, the accuracy is limited. Through (multiple)

integration and fusion of all three sensor outputs, it would be possible to recover the trajectory of a robot purely based on an IMU. But since the sensors suffer from noise and drift, for long term observations, the IMU data needs to be complemented by additional sensors, depending on the use case.

4.5 Distance Sensors

For obstacle avoidance, mapping, and localization it is essential for mobile robots to sense the distance to their surroundings. With increasing computational capabilities, vision-based systems using structured light, stereo cameras or even a single camera (structure from motion) play an increasing role and will be covered in the next section. Here, sensors which are capable to directly measure distance to other objects will be covered.

Sonar Sensors

Similarly, to bats, sonar sensors emit impulses of ultrasonic sound and measure the **time-of-flight (TOF)** until the echo is received. The distance d is then the direct result of the measured time t between emitting and receiving the impulse multiplied by the velocity of sound v_{Sound} and dividing the result by two to take into account that the sound has to travel back and forth to the object (see [EQ]).

Time-of-flight

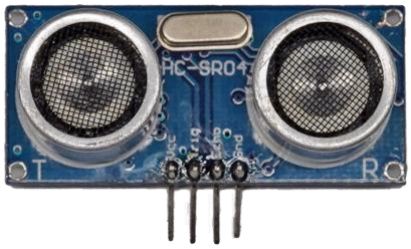
Time passed between the emission of a signal and the reception of its echo.

$$d = \frac{v_{Sound} \cdot t}{2}$$

Even though the principle is simple and sensors as shown in [FIG] are low-cost, there are some caveats. One of them is the speed of sound itself. Not only does it depend on temperature of air, due to the relatively low velocity of $v_{Sound,20°C} = 343\ \frac{m}{s}$ it takes a certain amount of time to get a result for objects further away. This makes a single sonar sensor unsuitable for a 360° swipe with plenty measurement points to be taken as it is common for laser scanners (see next section). Therefore, for mobile robot applications based on sonar sensors typically several sensors are mounted for

full 360° coverage. Typical frequencies for sonar sensors are 40 kHz and above, and are inconceivable for the human ear which has a perception threshold of roughly 20 Hz - 20 kHz.

Low-Cost Ultrasonic Sensor With One Emitter and One Receiver



Source: Florian Simroth (2023).

As the energy of the impulse cannot be focused as sharply as for instance lasers, the resolution of objects is relatively low and the range of typical sonar sensors is within a few meters. Also, for smooth surfaces and shallow incident angles, the sound impulse might be deflected with no echo thrown back to the receiver, making the object “invisible” to the sensor. On the other hand, in the presence of many objects or a complex shaped environment, repeatedly reflected echoes may lead to erroneous sensor readings. An advantage over optical sensors though is its insensitivity to dust, fog, or light.

Infrared Distance Sensors

Infrared (IR) distance sensors use an infrared LED to emit a light pulse, a Position Sensitive Device (PSD) as a receiving unit, and a triangulation method to determine the distance to an object. A PSD is capable to determine the position and intensity at which a light beam impinges upon its surface. In theory, the distance could be determined by emitting an infrared pulse and then measuring the intensity of the echo – as the intensity gradually declines with increasing distance, the distance to an object could be estimated. Even though this principle is used for simple infrared

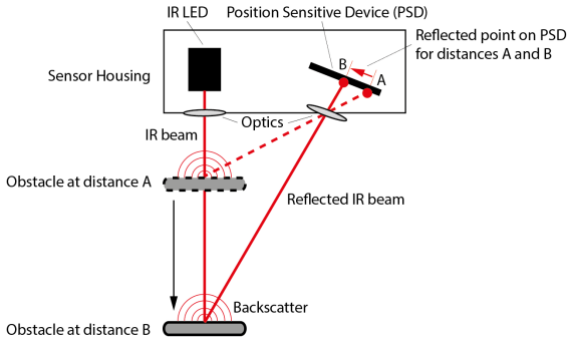
proximity sensors, it lacks in accuracy and is impacted by the reflectivity of the respective surface. Infrared distance sensors as shown in [FIG1] therefore use triangulation to overcome this shortcoming.

Low-Cost Infrared Distance Sensor With Visible Emitting and Receiving Optics

Source: Florian Simroth (2023).

[FIG2] shows how an infrared beam emitted by an infrared LED is backscattered by an obstacle positioned in two different distances A and B. From the backscatter, a beam of light enters the aperture of the receiving end and forms a light point on the PSD whose relative position can be measured. For different distances A and B, two different light spots are formed on the PSD. With known sensor dimensions and optics properties, the angle of the reflected beam can be determined as a function of the light point's position on the PSD and consequently the distance of the obstacle can be derived.

Schematic of the Triangulation Principle Used Within an Infrared Distance Sensor



Source: Florian Simroth (2023).

Infrared distance sensors can overcome some of the downsides of sonar sensors in terms of resolution, accuracy, and are available at low costs. Though, ranges are not much higher than one meter for the low cost versions and the non-linear voltage output of the sensor requires more effort for deducing the actual distance (Benet et al., 2002).

Lidars

Lidars (Light Detection And Ranging Sensors) are sensors that emit laser pulses and determine the distance to an object by the elapsed time of flight. As the impulses travel with the speed of light, the environment can be sampled with a large number of measurement points. Older lidars and commonly industrial lidars make use of rotating mirrors which deflects the laser impulses to cover up to 360° field of view (FOV). While 2D lidars as shown in [FIG] sample a planar slice of the environment, 3D sensors add additional layers of laser beams and receivers such that a three dimensional point cloud of the environment is gathered. Depending on the sensor, these point clouds are dense enough to not only build up maps but also identify and classify objects.

Low-Priced 360° Planar Lidar Usable for Simple Mobile Robot Applications, Such as Vacuum Cleaners

Source: Florian Simroth (2023).

Commonly, wavelengths invisible to the human eye within the near infrared range such as 885nm or 905nm are used – particularly for these wavelengths cheap silicon chips are available. There also exist lidars with short wave infrared frequencies like 1550nm, which allow for more power output and resulting ranges without endangering the human eye as these wavelengths are already absorbed within the vitreous body of the eye. Yet, other materials such as IndiumGalliumArsenide (InGaAs) are required which are more expensive to produce. Lidars currently under development aim for ranges of 200 meters and more.

Next to lidars with rotating mirrors, particularly for the automotive industry, solid state lidars are without any moving parts getting into focus to better withstand shocks and vibration present in moving vehicles. Some of these principles are listed below:

- Flash or sequential flash lidars illuminate the scene with either a single flash or sequentially and use an array of SPADs (Single Photon Avalanche Diodes) mounted behind optics to receive the light and determine distances for each SPAD with the time-of-flight method
- Optical phased array lidars scan the environment by a laser beam whose direction is controlled by an interference pattern that is based on phase-shifting the light from each laser emitter within an array.
- MEMS lidars utilize micro-mechanical mirrors that deflect a laser beam to systematically sample the environment. As there are still movable parts involved, they are often not considered to be true solid state lidars. The micro-mechanical mirrors have the advantage, that different scan patterns can be performed, allowing for varying resolutions in different parts of the field of view.

Areas of application for lidars within the industrial domain are for example autonomous ground vehicles (AGVs) used in intra-logistics, autonomous vehicles in

the automotive domain, counting of people in crowded areas, or even new and experimental areas such as for ships or autonomous baggage carts at airports.

Disadvantages of lidars are requirements regarding laser eye safety which limit maximum power output and correspondingly range and costs. Even though resolutions are steadily increasing, they are still far from what cameras can achieve. In contrast to radar sensors, lidars are susceptible to visual interference such as fog, rain or sun light. On the other hand, as lidars are active sensors, they work without the necessity of external light sources. Next to a 3D dimensional image of the surrounding, many types are also able to gather information about the intensity of certain objects, enabling the detection of lane markings in the automotive domain or other reflective beacons in the industrial domain. So-called Frequency Modulated Continuous Wave (FMCW) lidars are able to additionally return information on the speed of objects based on the Doppler-effect, but are costly due to the complexity of the underlying technology.

Self-Check Questions

1 Given the velocity of sound v_{Sound} and the time t passed between emission of a sound impulse and reception of its echo – how is the distance to the obstacle calculated?

$$d = \frac{v_{Sound} \cdot t}{2}$$

2 How is the distance with an infrared sensor being determined?

Via triangulation

4.6 Vision Sensors

Cameras enable robots to perceive their environment in a way similar to the human eye. Main components are the optics in form of lenses that collimate the light and the actual sensor which transforms the photons into an electric signal to be processed afterwards. Sometimes these two components are complemented by an additional device for illumination the scene.

Photoresistor

Photoresistors change their resistance when exposed to a light source. In particular, the resistance decreases with an increase of light intensity which can be measured. As photoresistors are susceptible to temperature variations, show latency in signal output, and are less light-sensitive then for instance photodiodes, their area of application in mobile robots is rather limited. An exemplary use case could be a line following robot which only needs to distinguish between bright and dark areas on the ground.

Photodiode

A photodiode is a semiconductor which capable to generate a current once subjected to light with a working principle similar to solar cells. The current increases with higher light intensity and with additional filters, the photodiodes can be modified to be only sensible to certain wave lengths.

Complementary Metal-Oxide Semiconductors (CMOS)

CMOS sensors refer to integrated arrays of pixels, where each pixel consists of a photodetector such as a photodiode coupled with one or more transistors for amplification. They are mainly used for imaging sensors within cameras and are cost-efficient due to the combination of sensing and amplification elements within the same integrated circuit.

Example of a CMOS Sensor Built Into a Consumer Camera

Source: Florian Simroth (2023).

Charge Coupled Device (CCD)

CCD imaging sensors are composed of an array of light-sensitive capacitors forming the pixels which build up a charge over a period of light exposure. The charge is then transported to an output amplifier were each pixel is evaluated at a time while the capacitor is discharged. Overexposure of one pixel can lead to a spill-over of charge to neighboring elements resulting in so-called “blooming” – white areas around bright points in an image. Nowadays CCDs are mostly superseded by CMOS sensors, which are cheaper, consume less power, and are less sensitive to blooming.

Dynamic Vision Sensor (DVS)

Dynamic vision sensors are a relatively new field, which capture not the full scene such as cameras equipped with CCD or CMOS sensors, but instead each pixel returns an event signal if the perceived brightness changes above a certain threshold. The signal may also contain the polarity, i.e., whether the brightness has increased or decreased. After the event, the pixel brightness reference is reset to the value where the event was triggered. The pixel itself consists of a photoreceptor and a differentiator which sends a signal to a shared bus once the difference of current brightness and current reference exceeds a threshold – in that sense all pixels can send events asynchronously.

These kinds of sensors are particularly useful to capture fast moving objects and do not suffer from motion blur as conventional cameras.

Self-Check Questions

3 Please complete the following sentence.

The *CMOS* imaging sensor has pretty much superseded *CCD* imaging sensors in today's consumer cameras.

4 What is the key difference between a CMOS and a DVS sensor?

The pixels of DVS sensors only respond to change in light intensity.

4.7 Robot Vision and Image Processing

For some more sophisticated tasks, a mobile robot may need more information about its environment than it can purely acquire with the sensors mentioned before. In fact, it may need an interpretation of what its surrounding is comprised of to derive further actions of motion or manipulation. To accomplish this, the robot needs a model to relate the sensor data to the 3D surrounding and give meaning to it in the sense of the identification of objects, obstacles, and so forth. This whole process is encapsulated in the domain of robot vision, which according to Muruganand et. al. (2020, p. 371-372), is a branch of computer vision, with the extension of physically interacting with the environment and therefore includes the robot's kinematics and additional calibration requirements. The main steps are:

- Image acquisition: The robot records images or sequences of images of its environments using vision sensors such as cameras
- Preprocessing: Prior to further processing, usually the image data needs to be preprocessed to reduce noise, or apply other adjustments such as contrast, brightness, or scaling
- Segmentation: Division of the image into parts relevant for further processing and parts to be discarded
- Feature extraction: Extraction of features which are special points of interest within an image such as corners or edges which will later be used for subsequent tasks

- Classification: Association of the content of an image to certain classes.
- Object tracking: Detection of Objects within an image and tracking within a sequence of images which requires recognizing the same object along several images.
- Action: Deduction of proper operations for the robot to fulfill its task like grasping an object or moving to a certain location.

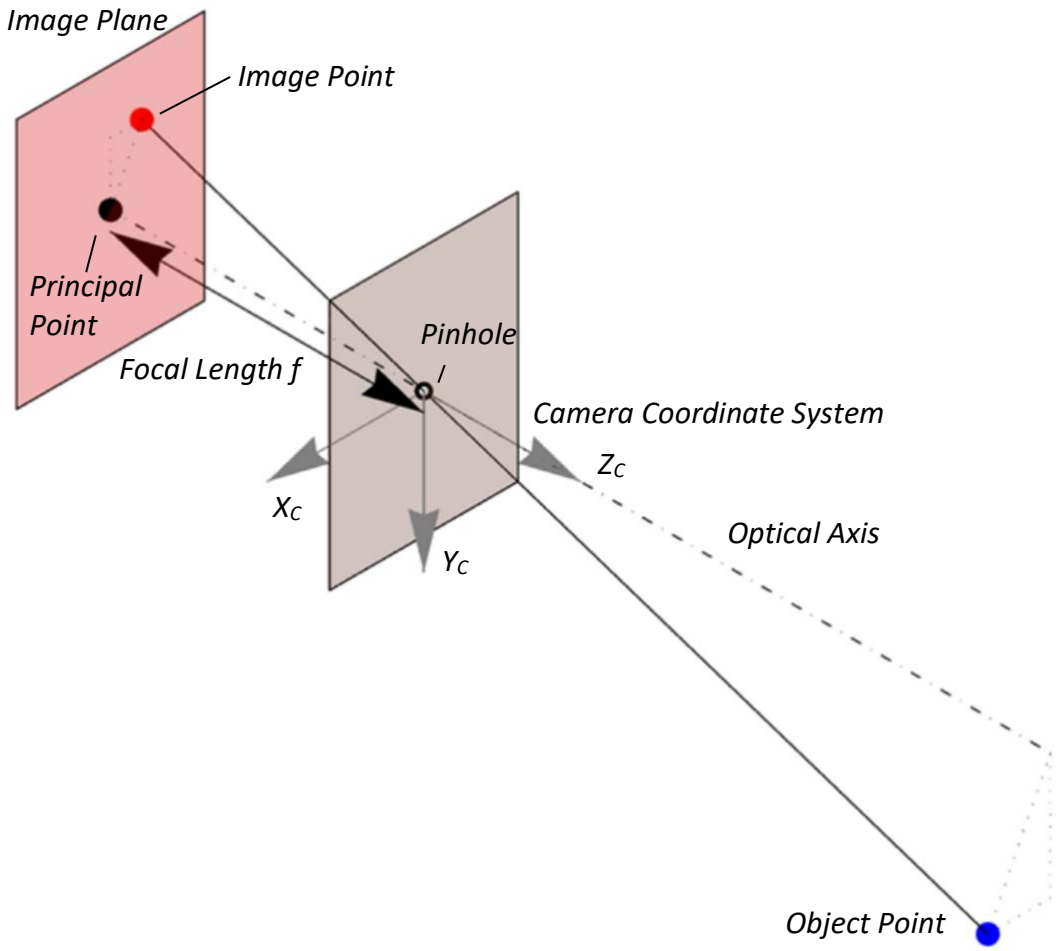
Image Acquisition

The first step in robot vision is the acquisition of the image itself. From the illuminated scene around the robot, light is reflected by objects depending on the properties of the objects' materials. This light is captured by cameras and eventually transformed into digital images.

Perspective camera and camera model

A perspective camera is composed of an aperture, a lens system, and an imaging sensor such as the previously described CMOS sensor and has a working principle similar to the human eye. By adjusting the shutter speed and aperture size, the amount of light hitting the sensor is varied, yielding brighter or darker images while the lens system allows for changing the focus. The image itself is a two-dimensional projection of the three-dimensional environment. In order to understand how an image point is related to its origin within the three-dimensional physical world, the pinhole model can be used to represent this relationship. In a pinhole camera, the lens system is replaced by a small hole, resulting that the whole scenery is in focus. The components of an ideal pinhole camera are shown in [FIGURE].

Components and Parameters of an Ideal Pinhole Camera

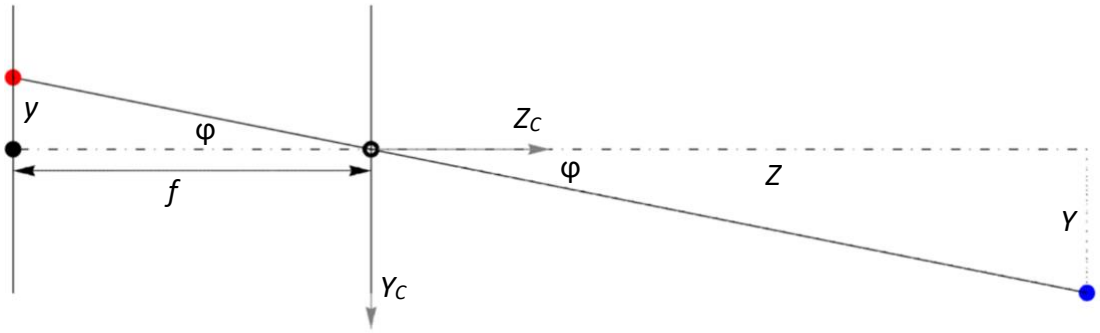


Source: Florian Simroth (2023).

The figure shows, how a light ray is reflected by an object point (blue), passes through the pinhole of the camera, and forms the image point where it hits the image plane. The focal length represents the distance between the pinhole and the image plane and directly relates how a large an object appears on the image plane. The optical axis is perpendicular to the image plane passing through the pinhole intersecting the image plane at the principle point. The camera coordinate system is typically oriented with its x-axis to the right, the y-axis pointing downwards, and the z-axis pointing outwards of the camera. Choosing the pinhole as the origin simplifies the projection equations described below.

In the following the relationship between an object point and its projected image point are derived as shown in [FIGURE].

Cross-Section of a Pinhole Camera With Object Point (Blue) and its Projection (Red) to the Image Plane



Source: Florian Simroth (2023).

The Y-component of a three-dimensional object point (X, Y, Z) can be mapped to the y-component of the two-dimensional point on the image, by equating the tangent of the angle φ for the object point

$$\tan\varphi = \frac{Y}{Z}$$

and the tangent of the angle φ for the image point

$$\tan\varphi = -\frac{y}{f}$$

which yields the projection

$$y = -f \cdot \frac{Y}{Z}$$

And respectively for the x-component

$$x = -f \cdot \frac{X}{Z}$$

With these equations it is possible to relate three-dimensional object points to their two-dimensional image representatives.

Camera Calibration

Even though the geometric dimensions and properties of a certain camera are defined by its design, there exist defects such as misalignments or dimensions deviating from the design due to manufacturing inaccuracies. Typically, radial and tangential distortions are present, with radial distortions being the dominant ones. Radial distortions describe the displacement of image points with respect to their ideal position depending on the radial distance to the principal point. By means of camera calibration, the geometric properties within the camera matrix as well as the distortions can be determined whereby distortions within the image can be compensated.

There exist many different approaches for calibration for which often specific calibration patterns are used whose composition and dimensions are exactly known. During the calibration now two problems need to be solved: 1) the external orientation of the camera with respect to the calibration pattern needs to be estimated and 2) the intrinsic camera geometry needs to be determined. The common approaches solve both problems mostly simultaneously in an iterative manner. A well-calibrated camera is a prerequisite for the upcoming processing steps.

Omnidirectional cameras

There exist a variety of other camera systems used by mobile robots to gather information about their environment. To generate a 360° surround view which simultaneously provides information on objects around a mobile robot, outputs of multiple cameras, evenly distributed around the mobile robot, can be stitched together. In fact, even with a single camera, a 360° view can be created: To this end, the camera view axis is vertically aligned, and a parabolic mirror is placed directly

above, such that the light from all around the robot is reflected into the camera lens. The result is a highly distorted image, which can be processed with the knowledge of a corresponding camera model.

Stereo Cameras

Stereo cameras allow for depth perception of the environment like the proprioception of humans. To this end, two cameras are placed side by side, and simultaneously record images. As both images are taken from slightly different perspectives, a particular point in space is projected to different positions on the image planes within each camera. If the correspondences of the object point on both images is identified, this difference can be used in algorithms to estimate the depth distribution of the objects within the scene. An exemplary stereo camera from the consumer market is shown in [FIGURE].

Fujifilm FinePix Real 3D W3 Stereo Camera Developed for the Consumer Market

Source: Florian Simroth (2023).

Time of Flight Camera

Time of flight cameras emit a pulse of light and measure the time difference before the back scatter hits the imaging sensor. Depending on the distance to the objects within the scene, the reflections reach each pixel of the imaging sensor at different times from which a depth image can be created.

Preprocessing

The preprocessing step is often necessary to normalize the images in the sense of brightness or contrast. Also, smaller defects can be removed using appropriate filter algorithms.

Segmentation

Often, only certain parts of the image need to be analyzed or are relevant for the designated task. The segmentation step is concerned with separating areas within the field of view. For instance, autonomous vehicles often need to distinguish the road from sidewalks or moving objects from the static environment. The subspaces of the image, i.e., the segments, can then be addressed by more specific algorithms in a more efficient way than if the full image was processed.

Feature Extraction

Features are certain notable points or areas within the image, whereby it depends on the utilized algorithm of what will be used as a feature. This can be, for example, corners, edges, or colors. Also, the number of detected features and whether the features are robust against rotation or scaling, i.e., whether the same feature points would be identified after the image was rotated or scaled, depend on the chosen algorithm. Feature points are used, for example, for panoramic pictures: multiple images are stitched together and are transformed by detecting common feature points within the overlapping area of two consecutive pictures taken during the panoramic shot. Other uses are the classification of images or objects as well as tracking for instance a vehicle recorded in a video by associating the vehicle's feature points in consecutive images.

Classification

The classification step consists in associating the images (or parts within) to certain types to better understand the content of an image. For the classification step commonly algorithms such as artificial neural networks, support vector machines, or

decision trees are being used. As an example, more and more vehicles use image classification to get information about the current speed limit by classifying image segments as corresponding road signs.

Object detection/tracking

For many tasks it is required to detect and keep track of certain objects, such as faces within a video stream or other traffic participants whose future paths may be estimated by tracking and extrapolating their movements. The previously identified Features can help to identify the same object in two consecutively recorded images. The task is first to detect an object within an image, which often includes defining a bounding box around that object and second to recognize the same object within different images.

Action

This action step is very specific to robot vision, as physical action will be triggered. This could be a change of the driving direction, to bypass an object, or to move a robotic arm to grab an object. A set of actions or desired behavior is usually predefined from which the appropriate one needs to be chosen.

Self-Check Questions

5 If the focal length of a pinhole camera is increased, the projected image size of an object *increases*.

6 Which processing step is particularly internalized in robot vision, and generally distinguishes it from computer vision?

The action step, as the robot needs to interact with its physical environment.

4.8 Global Positioning System

The Global Navigation Satellite System (GNSS) is widely used nowadays with applications far beyond the well-known navigation systems for cars. For mobile robots operating in open space such as airborne drones or outdoor mobile robots,

GNSS is very helpful in the process of localization, mapping, and enhancement of ego-motion data.

Even though everyday speech mostly refers to “GPS” for the Global Positioning System (GPS), the list of meanwhile available systems is long (see Joubert et al., 2020 and Jin et al., 2022):

- Global Positioning System (GPS), globally available – USA
- Galileo, globally available – European Union
- GLObal Navigation Satellite System (GLONASS), globally available – Russia
- BeiDou Navigation Satellite System (BDS), globally available – China
- Indian Regional Navigation Satellite System (IRNSS/NavIC), locally availbale – India
- Quasi-Zenith Satellite System (QZSS), locally available – Japan
- Regional South Korean Positioning System (KPS), locally available – Korea

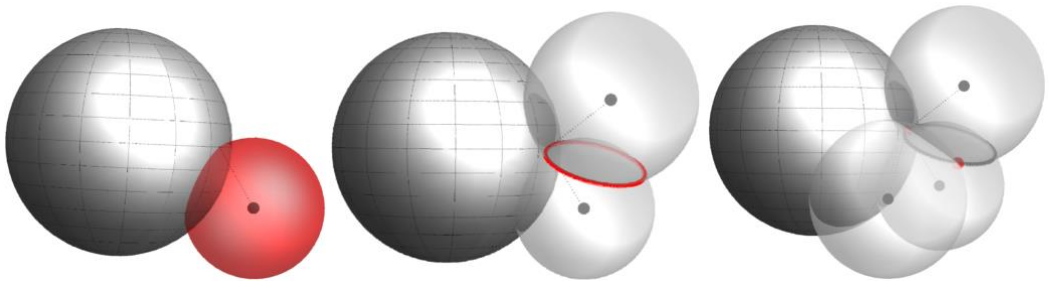
There are two fundamental methodologies for localization: Absolute positioning solely relies on a single receiver and the signals received by multiple satellites, while for differential positioning, the receiver’s position is determined with the support of additional terrestrial stations with exactly known coordinates (Jin et al., 2022).

The satellites are equipped with atomic clocks to ensure time synchronicity of all system’s satellites. For positioning purposes, each satellite continuously transmits information of its trajectory to derive the exact position with respect to an earth-fixed coordinate system for any point of time as well as the satellite clock time. The satellite’s trajectory information and time is regularly updated via ground stations to account for course and clock deviations. A receiver unit can then determine the time-of-flight of the satellite’s signal through comparison of the satellite’s and receiver unit’s time stamp from which the distance to the satellite results. As this would require a precise synchronized clock, the time information is derived with the help of

an additional satellite. Hence, the receiver needs contact to at least four satellites to estimate its position – three for positioning and one to derive the timing information.

Once the distances to at least three satellites with known absolute positions are derived, the coordinates of the receiver can be calculated by trilateration (see [FIG]): With the information of one satellite, the receiver's position is constrained to any location on the surface of a sphere centered on the satellite with a radius equal to the estimated distance. The information of a second satellite further restricts the receiver's position to the intersection of two spherical surfaces resulting in a circle. The intersection of this circle with another spherical surface derived from the third satellite has two points as a solution. Of these, one can be discarded by simple reasoning, considering that the receiver will be placed somewhere on the earth's surface.

Potential Positions in Red With Line of Sight to One, Two, and Three Satellites Respectively



Source: Florian Simroth (2023).

For full coverage meaning, that at least four satellites are visible for any point on earth at any time, GNSS systems usually comprise 24 satellites that continuously broadcast their signals (Joubert et al., 2020, p.2). The number of receivers on the other hand is unlimited, as each receiver can perform the processing independently. For the so-called Real-Time Kinematic (RTK) method, accuracies within the centimeter range are achievable, while for the Precise Point Positioning (PPP) method, the accuracy is about a few decimeters (Joubert et al., 2020, p.3) As walls of

buildings dampen the satellite's signal, GNSS signals best perform outdoors and are not suitable for precise indoor mobile robot navigation.

Self-Check Questions

7 Complete the following sentence.

The maximum number of GPS receivers that can be used with the GPS system is *unlimited*.

8 If the distances of a receiver towards two satellites are known, the space of possible locations of the receiver is...

- ☐ *a circle*
- ☐ the surface of a sphere
- ☐ the volume of a sphere
- ☐ a point

Summary

Mobile robots gain knowledge about their environment with sensors. Sensors are usually composed of a transducer unit which translates a physical property into some sort of electrical signal and a processing unit or circuit that converts the signal into the desired information. The information such as a distance to the next obstacle or angular velocity of the wheels can then be used within the algorithms that determine the behavior of the robot.

Sensors make use of a variety of physical principles such as induction, magnetism, acoustics, or optics that are often used indirectly to determine a certain quantity such as the angular position of a rotating shaft. Important properties of sensors for engineers to pay attention to when choosing an appropriate sensor are the accuracy, precision, frequency, resolution, and working conditions.

Unit 5 – Manipulators

Study Goals

On completion of this unit, you will be able to …

… derive the kinematic and dynamic equations of motion of manipulators.

… describe issues arising with different types of manipulators and configurations.

… analyze the manipulability of manipulators within the workspace

… understand the contributions of mobile platform and manipulator in a combined mobile manipulator.

Unit 5 – Manipulators

Introduction

In the last units, the fundamentals of mobile robots were covered and the kinematics and dynamics of a mobile platform were derived. Even though, mobile platforms by themselves are already capable to fulfill certain tasks such as transportation in intralogistics, the versatility of mobile robots can greatly be enhanced by the addition of manipulators. Manipulators mounted on a mobile platform and therefore referred to as mobile manipulators, greatly increase the variety of tasks a mobile robot can perform as it allows various forms of interaction with the environment. In contrast to fixed manipulators, the workspace of mobile manipulators is greatly increased.

This unit aims at understanding differences in modeling manipulators and mobile platforms and how the equations of motion can be combined to eventually assemble a mobile manipulator. Some concepts and properties that are particularly inherent to manipulators will be covered based on fixed manipulators at first.

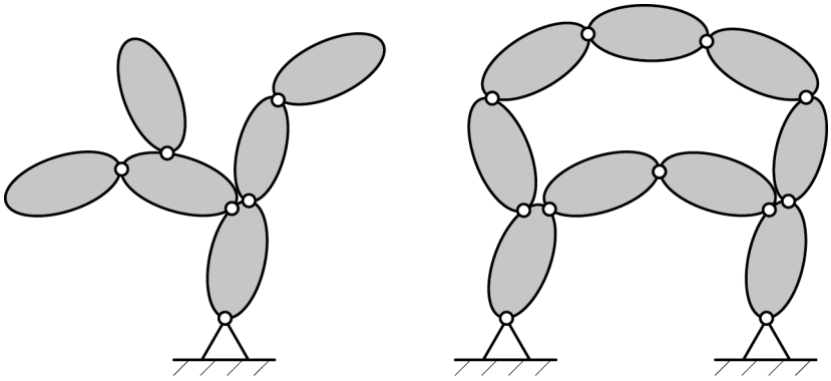
Once the concepts are understood, the kinematic and dynamic equations of motion of mobile platforms and manipulators will be joined to create an overall model of the mobile manipulator. To illustrate the theory, several examples of common manipulator designs will be analyzed in regards to the equations of motion and their properties.

5.1 Basics

A manipulator forms an interface between the robot and its environment. A manipulator consists of rigid bodies commonly termed links which are connected by joints. A series of links and joints is often referred to as a kinematic chain. Usually, revolute or prismatic joints are used for manipulators for which industrialized actuators exist. The end effector of a manipulator, such as a gripper or a tool, is the component which is in direct contact with the environment or the object to be "manipulated". On the other hand, the component of the manipulator by which it is rigidly connected to the world is called base frame. In general, there exist also manipulators with more than one end effector which though only play a minor role and will not be further elucidated.

One important distinction of different types of manipulators that can be made are serial manipulators and parallel manipulators. Serial manipulators are characterized by the fact, that there exists only a single kinematic chain between any link including the end effector and the base frame. In parallel manipulators, on the contrary, there exist multiple paths, such that one can find kinematic chains that form closed loops. Therefore, serial and parallel manipulators are also referred to as open and closed kinematic chains. An example for an open and closed kinematic chain is shown in [FIGURE].

Open Kinematic Chain (Serial Manipulator) With Only One Path From Base to (Each) Tip (Left) and a Closed Kinematic Chain (Parallel Manipulator) With Kinematic Loops (Right)



Source: Florian Simroth (2023).

Parallel manipulators have the advantage, that conceptually they can bear heavier loads since the load distributes to multiple kinematic chains along the manipulator. Similar to springs put in parallel, parallel manipulators have a higher stiffness and can usually yield higher accuracy. On the other hand, more space is required for the multiple kinematic chains, thus, the workspace is rather limited and can be seen as the common subspace of all parallel kinematic chains. In addition, each closed kinematic chain imposes additional constraints as the joint variables are not independent but underlie additional constraints. Solving these closure conditions explicitly is difficult such that these are often solved numerically. The scope of this lecture therefor will be restricted to serial manipulators.

For serial manipulators, the relative displacement between two connected links allowed by the intermediate joint are described by joint variables or coordinates. The configuration of a manipulator is then given by the set of its joint variables. As any joint can be articulated independently (in contrast to parallel manipulators), the set of joint variables also constitutes the minimal set of generalized coordinates. Consequently, the degree of freedom of a serial manipulator is equal to the number of joint variables that determine its configuration.

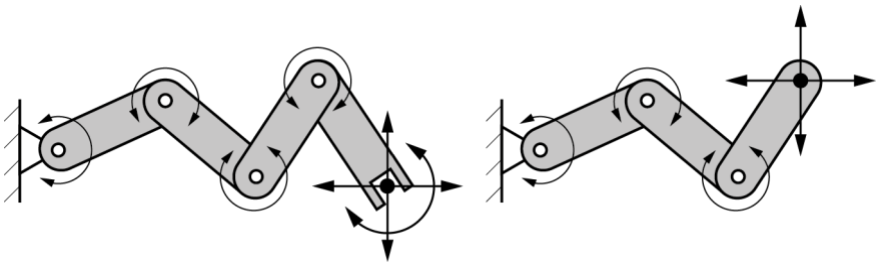
In the context of serial manipulators, the pose and motion of the end effector with respect to the base frame is usually the main feature of interest. The accuracy of a manipulator is an indicator for how large the deviation between the set point pose

and the actual pose of the end effector is. Furthermore, the repeatability is used as a measure of how well a manipulator can approach the same pose multiple times, that is the variance of the actual poses the end effector acquires when driven to the same target pose.

For a fixed serial manipulator, the workspace embodies all reachable poses of the end effector in physical space and is often also referred to as operational space (Siciliano & Kathib, 2016, p. 38). The configuration space is then the n dimensional space spanned by n independent joint variables. The task space highly relies on the "task" assigned to the manipulator, such that the same manipulator may have different task spaces. For instance, an assembly task may require the full six spatial degrees of freedom to exactly position and orient certain parts. If, on the other hand, a pick-and-place task is assigned to the same manipulator, that only includes lifting, planar positioning, and orientation along one axis, the task space only has four degrees of freedom.

If the degree of freedom of the configuration space is larger than the workspace, the manipulator has some degree of redundancy. An example is shown in [FIGURE] of a planar manipulator which has four revolute joints with parallel axes and thus a configuration space with four degrees of freedom. The workspace on the other hand only has three degrees of freedom with two position coordinates and one orientation coordinate. Sometimes it is reasonable to build manipulators with more degrees of freedom than actually required as this increases their versatility. This becomes handy, for example, for surgery robots that often have to reach certain areas behind obstacles such as tissue. From a kinematic point of view though, this introduces some issues that need to be handled, as there exist an infinite number of configurations to achieve a particular end effector pose. In that sense, a manipulator mounted on a mobile platform almost always has redundant degrees of freedom, as the degrees of freedom of the platform add to the degrees of freedom of the manipulator.

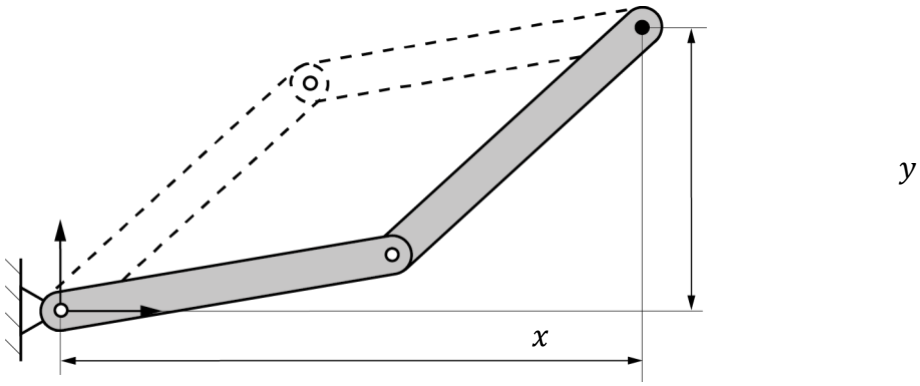
Intrinsically Redundant Manipulator With Four Degrees of Freedom in Planar (3 DOF) Embedding of the End Effector (Left); Task-Related Redundant Manipulator With Three Degrees of Freedom, of Which Only Two (Translations) Are Required for the Task (Right)



Source: Florian Simroth (2023).

As already defined in the last units, the **forward kinematics**, now in the context of manipulators, are concerned with determining the end effector pose within the operational space in terms of the joint variables in the configuration (or joint) space. **Inverse kinematics** then derives the the values for the joint variables for a predefined end effector pose. While the forward kinematics are straightforward and yield only a single solution, the same is not true for the inverse kinematics as the simple example in [FIGURE] shall demonstrate. Already for the simple two degree of freedom manipulator, there are multiple solutions of which one has to choose. In fact, for a general six degree of freedom manipulator, there are 16 valid possible solutions (Siciliano, 2010, p.91).

Two Solutions for the Inverse Kinematics of a 2 DOF Manipulator: Lefty and Righty Elbow Configurations



Source: Florian Simroth (2023).

Self-Check Questions

13. Why is it sometimes beneficial to build redundant manipulators?

To increase the dexterity throughout the workspace or for obstacle avoidance.

14. Please complete the following sentence.

For *serial* manipulators, there exists only one chain of links between base and end effector.

5.2 Modeling

For the kinematics and dynamics of a manipulator, the derivations from the previous chapters on wheeled mobile robots will be applied. For this reason, two coordinate frames of interest will be introduced: The base frame $\mathcal{K}_{\mathcal{B}}$, which will be attached to the base of the manipulator, and the end effector frame $\mathcal{K}_{\mathcal{E}}$ that represents the position and orientation of the end effector. The next sections will be devoted to derive the kinematics and the dynamics for fixed manipulators. Once these relations are established, the equations for mobile platforms will be united with the equations of the fixed manipulators to effectively model mobile manipulators.

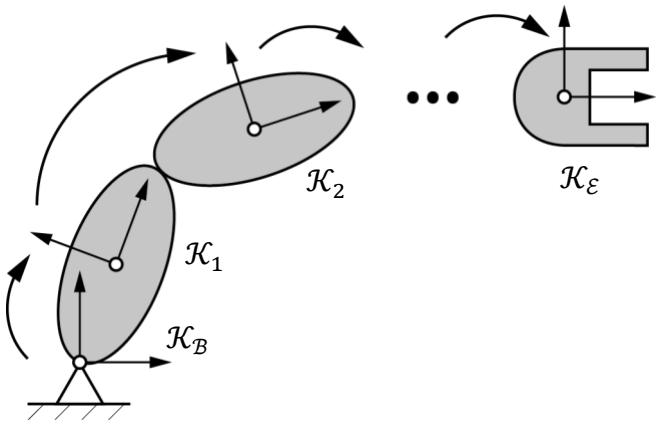
Forward Kinematics of a Fixed Manipulator

The direct kinematics regarding the **pose** of a serial manipulator as shown in [FIGURE] can be derived by a sequence of transformations

$$^{\mathcal{B}}\mathbf{T}_{\mathcal{E}} = {}^{\mathcal{B}}\mathbf{T}_1 {}^{1}\mathbf{T}_2 \ldots {}^{n-1}\mathbf{T}_{\mathcal{E}}$$

describing the relative transformation from the base frame $\mathcal{K}_{\mathcal{B}}$ to the end effector frame $\mathcal{K}_{\mathcal{E}}$ in homogeneous coordinates.

Coordinate Frames Attached to the Links of a Serial Manipulator



Source: Florian Simroth (2023).

To this end, a coordinate frame is attached to each link and the relative transformation between links $i-1$ and i is described by $^{i-1}\mathbf{T}_i$. Assuming, that the kinematic chain only consists of revolute and prismatic joints, the transformation $^{i-1}\mathbf{T}_i$ is a function of a single joint variable q_i. The base link will be denoted by 0, and the end effector link by n for n mobile links and n joints.

Even though homogeneous coordinates provide a neat way in regards to expressing transformations by simple matrix multiplications, the number of elements used (nine for the orientation and three for the position) is highly redundant since only six generalized coordinates are required to specify the pose in three dimensional space. Thus, if the pose of the end effector is to be expressed by the generalized coordinates $\boldsymbol{\xi}_{\mathcal{E}} = (\mathbf{p}_{\mathcal{E}}^T,\ \boldsymbol{\phi}_{\mathcal{E}}^T)^T$ with $\mathbf{p}_{\mathcal{E}} = (x,\ y,\ z)^T$ denoting the end effector position and $\boldsymbol{\phi}_{\mathcal{E}} =$

$(\gamma, \beta, \alpha)^T$ expressing the orientation using ZYX Euler angles, the relation can be written as

$$\boldsymbol{\xi}_{\mathcal{E}} = \mathbf{f}(\mathbf{q})$$

where $\mathbf{f}(\mathbf{q})$ normally is a non-linear function mapping the joint coordinates $\mathbf{q}$ to the end effector coordinates $\boldsymbol{\xi}_{\mathcal{E}}$ in operational space. While it is straightforward, to extract the position directly from the last column of the transformation matrix ${}^{\mathcal{B}}\mathbf{T}_{\mathcal{E}}$, the orientation parameters $\boldsymbol{\phi}_{\mathcal{E}}$ need to be calculated from the rotation matrices.

The velocity of the end effector frame can be represented in several ways, such as the physical velocity $\mathbf{v}_{\mathcal{E}} = (\dot{\mathbf{p}}_{\mathcal{E}}^T, \boldsymbol{\omega}_{\mathcal{E}}^T)^T$ with the linear velocity $\dot{\mathbf{p}}_{\mathcal{E}}$ and the angular velocity $\boldsymbol{\omega}_{\mathcal{E}}$ of the end effector or the derivative of the generalized pose coordinates $\dot{\boldsymbol{\xi}}_{\mathcal{E}} = \left(\dot{\mathbf{p}}_{\mathcal{E}}^T,\ \dot{\boldsymbol{\phi}}_{\mathcal{E}}^T\right)^T$. If the orientation of the end effector is described by ZYX Euler angles as introduced above, and recalling from the previous units that

$$\boldsymbol{\omega} = \underbrace{\begin{pmatrix} -\sin(\beta) & 0 & 1 \\ \cos(\beta)\sin(\alpha) & \cos(\alpha) & 0 \\ \cos(\beta)\cos(\alpha) & -\sin(\alpha) & 0 \end{pmatrix}}_{\mathbf{B}_{\text{ZYX}}} \cdot \begin{pmatrix} \dot{\gamma} \\ \dot{\beta} \\ \dot{\alpha} \end{pmatrix}$$

both end effector velocity descriptions can be related by

$$\mathbf{v}_{\mathcal{E}} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_{\text{ZYX}} \end{pmatrix} \cdot \dot{\boldsymbol{\xi}}_{\mathcal{E}} \tag{1}$$

The velocity relationship between the joint velocities $\dot{\mathbf{q}}$ in the configuration space and the physical velocity $\mathbf{v}_{\mathcal{E}}$ of the end effector in the operational space can be established using the Jacobian matrix $\mathbf{J}_{\mathbf{v}_{\mathcal{E}}}$:

$$\mathbf{v}_{\mathcal{E}} = \underbrace{\begin{pmatrix} \mathbf{J}_{\text{T}} \\ \mathbf{J}_{\text{R}} \end{pmatrix}}_{\mathbf{J}_{\mathbf{v}_{\mathcal{E}}}} \dot{\mathbf{q}}$$

The sub Jacobians $\mathbf{J}_T$ and $\mathbf{J}_R$ refer to the translational and the rotational part respectively. For the **translational** part, the component ${}^{\mathcal{B}}_{\mathcal{B}}\mathbf{r}_{\mathcal{E}}$ of the transformation from base to the end effector

$$ {}^{\mathcal{B}}\mathbf{T}_{\mathcal{E}} = \begin{pmatrix} {}^{\mathcal{B}}\mathbf{R}_{\mathcal{E}} & {}^{\mathcal{B}}_{\mathcal{B}}\mathbf{r}_{\mathcal{E}} \\ \mathbf{0}^T & 1 \end{pmatrix} $$

can be differentiated with respect to time and the coefficients of the generalized velocities $\dot{\mathbf{q}}$ are gathered in the matrix $\mathbf{J}_T$:

$$ \dot{\mathbf{p}}_{\mathcal{E}} = \frac{d}{dt}{}^{\mathcal{B}}_{\mathcal{B}}\mathbf{r}_{\mathcal{E}} = \frac{\partial {}^{\mathcal{B}}_{\mathcal{B}}\dot{\mathbf{r}}_{\mathcal{E}}}{\partial \mathbf{q}}\dot{\mathbf{q}} = \mathbf{J}_T \cdot \dot{\mathbf{q}} $$

For the rotational part, the angular velocities can be extracted using the Poisson equation after rearranging and sorting the coefficients of $\dot{\mathbf{q}}$:

$$ \tilde{\boldsymbol{\omega}} = \begin{pmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{pmatrix} = \dot{\mathbf{R}}\mathbf{R}^T \quad \boldsymbol{\omega} = \mathbf{J}_R \cdot \dot{\mathbf{q}} $$

The combined Jacobian $\mathbf{J}_{\mathbf{v}_{\mathcal{E}}}$ is also referred to as geometrical Jacobian, while there exists another type, the analytical Jacobian $\mathbf{J}_{\dot{\boldsymbol{\xi}}_{\mathcal{E}}}$ which maps the joint velocities $\dot{\mathbf{q}}$ of the manipulator to the end effector velocities expressed by generalized coordinates $\dot{\boldsymbol{\xi}}_{\mathcal{E}}$ (Siciliano, 2010, p. 104):

$$ \dot{\boldsymbol{\xi}}_{\mathcal{E}} = \frac{\partial \mathbf{f}(\mathbf{q})}{\partial \mathbf{q}}\dot{\mathbf{q}} = \mathbf{J}_{\dot{\boldsymbol{\xi}}_{\mathcal{E}}}\dot{\mathbf{q}} \tag{2} $$

The analytical Jacobian can be derived directly by taking the derivative of the non-linear function $\mathbf{f}(\mathbf{q})$ with respect to $\mathbf{q}$, though the function $\mathbf{f}(\mathbf{q})$ is often not directly available. On the other hand, it is possible to derive the analytical Jacobian from the geometric one and vice versa. Using the previously established relationship between $\mathbf{v}_{\mathcal{E}}$ and $\dot{\boldsymbol{\xi}}_{\mathcal{E}}$, the geometric Jacobian can be derived from the analytic Jacobian by multiplying (2) with the transformation matrix in (1)

$$\underbrace{\begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} \end{pmatrix} \cdot \dot{\boldsymbol{\xi}}_{\mathcal{E}}}_{\breve{\mathbf{v}}_{\mathcal{E}}} = \underbrace{\begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} \end{pmatrix} \mathbf{J}_{\dot{\boldsymbol{\xi}}_{\mathcal{E}}}}_{\breve{\mathbf{J}}_{\mathbf{v}_{\mathcal{E}}}} \dot{\mathbf{q}}$$

$$\mathbf{J}_{\mathbf{v}_{\mathcal{E}}} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} \end{pmatrix} \mathbf{J}_{\dot{\boldsymbol{\xi}}_{\mathcal{E}}}$$

and vice versa via the inverse

$$\mathbf{J}_{\dot{\boldsymbol{\xi}}_{\mathcal{E}}} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{B}^{-1} \end{pmatrix} \mathbf{J}_{\mathbf{v}_{\mathcal{E}}}$$

Note that with a determinant of $-\cos(\beta)$, the inverse is undefined for $\beta = 90°$ since the third rotation axis matches the first one and not all angular velocities can be mapped, which is termed a **representation singularity** as it just arises for the specific choice of generalized coordinates (Siciliano, 2010, p. 130).

The geometric Jacobian maps the joint variables to real physical linear and angular velocities represented in the base frame which is useful for many further analyses. The analytical Jacobian on the other hand allows the integration of the orientation parameters and eases the process to provide a trajectory for inverse kinematics. Depending on choice of representation of $\dot{\boldsymbol{\xi}}_{\mathcal{E}}$ and the simplicity of the problem at hand, both Jacobians can also be equal.

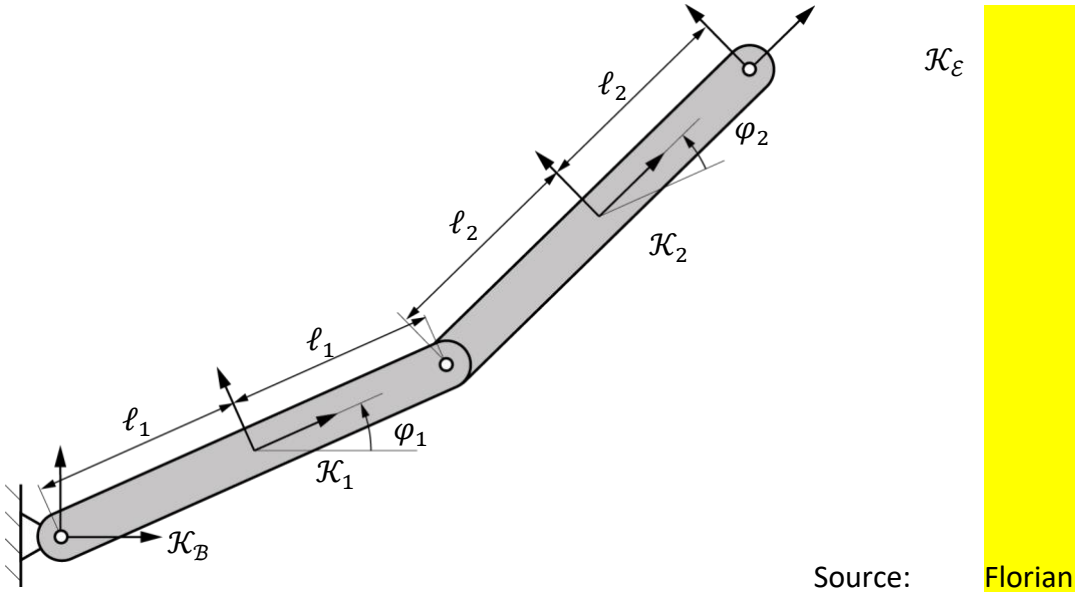
The previously presented concepts are a very shortened general summary on the forward kinematics of manipulators. Usually, the coordinate frames are aligned in a manner according to Denavit & Hartenberg (1955) or modified versions which requires only four parameters for the transformations between two successive coordinate frames, but expects a predefined methodology for placing the coordinate frames. By this formal description of manipulator frames, the geometric Jacobian can be derived in a recursive manner, expressing the Jacobian column by column by means of the preceding links. A detailed derivation and more background information on the geometric and analytical Jacobian can be found in Siciliano (2010).

Example

For the planar 2-DOF manipulator shown in [FIGURE], the position and orientation of the end effector can be described in homogeneous coordinates:

$$
\begin{aligned}
{}^{\mathcal{B}}\mathbf{T}_{\mathcal{E}}(\mathbf{q}) &= {}^{\mathcal{B}}\mathbf{T}_1(\mathbf{q})\,{}^{1}\mathbf{T}_2(\mathbf{q})\,{}^{2}\mathbf{T}_{\mathcal{E}} \\
&= \begin{pmatrix}
\cos(\varphi_1+\varphi_2) & -\sin(\varphi_1+\varphi_2) & 0 & 2\left(\ell_1\cos(\varphi_1)+\ell_2\cos(\varphi_1+\varphi_2)\right) \\
\sin(\varphi_1+\varphi_2) & \cos(\varphi_1+\varphi_2) & 0 & 2\left(\ell_1\sin(\varphi_1)+\ell_2\sin(\varphi_1+\varphi_2)\right) \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{pmatrix}
\end{aligned}
$$

Planar Manipulator With Two Degrees of Freedom



Source: Florian Simroth (2023).

The velocity of the end effector can be described with the vector $\mathbf{v}_{\mathcal{E}} = \left(v_x, v_y, v_z, \omega_x, \omega_y, \omega_z\right)^T$ and the components of the geometrical Jacobian can be derived as follows. The linear velocities directly follow from the time derivatives of the position vector in ${}^{\mathcal{B}}\mathbf{T}_{\mathcal{E}}$:

$$\begin{aligned}\dot{\mathbf{p}}_{\mathcal{E}} &= \begin{pmatrix} -2\dot{\varphi}_1\ell_1\sin(\varphi_1) - 2(\dot{\varphi}_1+\dot{\varphi}_2)\ell_2\sin(\varphi_1+\varphi_2) \\ 2\big(\dot{\varphi}_1\ell_1\cos(\varphi_1) + (\dot{\varphi}_1+\dot{\varphi}_2)\ell_2\cos(\varphi_1+\varphi_2)\big) \\ 0 \end{pmatrix} \\ &= \underbrace{\begin{pmatrix} -2\big(\ell_1\sin(\varphi_1)+\ell_2\sin(\varphi_1+\varphi_2)\big) & -2\ell_2\sin(\varphi_1+\varphi_2) \\ 2\big(\ell_1\cos(\varphi_1)+\ell_2\cos(\varphi_1+\varphi_2)\big) & 2\ell_2\cos(\varphi_1+\varphi_2) \\ 0 & 0 \end{pmatrix}}_{\breve{\mathbf{J}}_T} \begin{pmatrix} \dot{\varphi}_1 \\ \dot{\varphi}_2 \end{pmatrix}\end{aligned} \tag{3}$$

The angular velocities can be derived from the corresponding components of the resulting rotational matrix $\dot{\mathbf{R}}\mathbf{R}^T$ in ${}^{\mathcal{B}}\mathbf{T}_{\mathcal{E}}$:

$$\dot{\mathbf{R}}\mathbf{R}^T = \begin{pmatrix} 0 & -\dot{\varphi}_1-\dot{\varphi}_2 & 0 \\ & & \breve{\omega}_y \\ \dot{\varphi}_1+\dot{\varphi}_2 & 0 & 0 \\ \breve{\omega}_z & & \\ 0 & 0 & 0 \\ & \breve{\omega}_x & \end{pmatrix}$$

$$\boldsymbol{\omega} = \begin{pmatrix} 0 \\ 0 \\ \dot{\varphi}_1+\dot{\varphi}_2 \end{pmatrix} = \underbrace{\begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 1 \end{pmatrix}}_{\breve{\mathbf{J}}_R} \begin{pmatrix} \dot{\varphi}_1 \\ \dot{\varphi}_2 \end{pmatrix}$$

The geometric Jacobian then yields:

$$\mathbf{J}_{\mathbf{v}_{\mathcal{E}}} = \begin{pmatrix} \mathbf{J}_{\mathrm{T}} \\ \mathbf{J}_{\mathrm{R}} \end{pmatrix}$$

The analytical Jacobian will hereafter be derived both, from the geometric Jacobian and directly from the function $\mathbf{f}(\mathbf{q})$. Specifying the end effector pose with the coordinates $\boldsymbol{\xi}_{\mathcal{E}} = (x,\ y,\ z,\ \gamma,\ \beta,\ \alpha)^T$ using ZYX Euler angles, the function $\mathbf{f}(\mathbf{q})$ can easily be extracted from ${}^{\mathcal{B}}\mathbf{T}_{\mathcal{E}}(\mathbf{q})$:

$$\underbrace{\begin{pmatrix} x \\ y \\ z \\ \gamma \\ \beta \\ \alpha \end{pmatrix}}_{\breve{\boldsymbol{\xi}}_{\mathcal{E}}} = \underbrace{\begin{pmatrix} 2\left(\ell_1 \cos(\varphi_1) + \ell_2 \cos(\varphi_1 + \varphi_2)\right) \\ 2\left(\ell_1 \sin(\varphi_1) + \ell_2 \sin(\varphi_1 + \varphi_2)\right) \\ 0 \\ \varphi_1 + \varphi_2 \\ 0 \\ 0 \end{pmatrix}}_{\breve{\mathbf{f}}(\mathbf{q})} \tag{4}$$

By performing $\partial \mathbf{f}(\mathbf{q}) / \partial \mathbf{q}$, the analytical Jacobian can directly be derived, where the translational part is equal to the translational part of the geometrical Jacobian and the rotational part is straightforward.

$$\mathbf{J}_{\dot{\boldsymbol{\xi}}_{\mathcal{E}}} = \begin{pmatrix} \mathbf{J}_{\mathrm{T}} \\ \mathbf{J}_{\mathrm{R}_{\dot{\boldsymbol{\xi}}}} \end{pmatrix} \quad \text{with} \quad \mathbf{J}_{\mathrm{R}_{\dot{\boldsymbol{\xi}}}} = \begin{pmatrix} 1 & 1 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}$$

The same result is achieved when deriving the analytical from the geometrical Jacobian:

$$\mathbf{B}^{-1} = \begin{pmatrix} 0 & \sin(\alpha)/\cos(\beta) & \cos(\alpha)/\cos(\beta) \\ 0 & \cos(\alpha) & -\sin(\alpha) \\ 1 & \sin(\alpha)\tan(\beta) & \cos(\alpha)\tan(\beta) \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

$$\mathbf{J}_{\dot{\boldsymbol{\xi}}_{\mathcal{E}}} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{B}^{-1} \end{pmatrix} \begin{pmatrix} \mathbf{J}_{\mathrm{T}} \\ \mathbf{J}_{\mathrm{R}} \end{pmatrix} = \begin{pmatrix} \mathbf{J}_{T} & \\ & \mathbf{J}_{\mathrm{R},\dot{\boldsymbol{\xi}}} \end{pmatrix} \quad \text{and} \quad \mathbf{J}_{\mathrm{R},\dot{\boldsymbol{\xi}}} = \begin{pmatrix} 1 & 1 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}$$

For the planar case, the matrix $\mathbf{B}$ simplifies due to $\alpha = 0$ and $\beta = 0$ and for the presented simple manipulator only swaps the locations of the entries. In the general case, each component of the angular velocity is a term originating from multiple orientation coordinates in $\boldsymbol{\xi}_{\mathcal{E}}$. Here, the the yaw angle rate $\dot{\gamma}$ equals the angular velocity component ω_z.

Inverse Kinematics

For the case of inverse kinematics on **position** level, a function $\bar{\mathbf{f}}$ is sought, which maps the end effector pose $\boldsymbol{\xi}_{\mathcal{E}}$ to the joint coordinates $\mathbf{q}$:

$$\mathbf{q} = \bar{\mathbf{f}}(\boldsymbol{\xi}_{\mathcal{E}})$$

Yet, while it is always possible to determine the forward kinematics for given joint coordinates, this is not always the case for the inverse kinematics. For the simple manipulator in figure 3 it was already shown, that the inverse kinematics problem is not uniquely defined, since there were two solutions for the joint coordinates to achieve a single position of the end effector. Furthermore, if $\boldsymbol{\xi}_{\mathcal{E}}$ was chosen outside the reachable workspace or even within the reachable workspace, but with a non-achievable orientation, no solution exists at all. On the other hand, for a redundant manipulator, there exist infinite many solutions due to additional degrees of freedom within the manipulator that do not affect the end effector. At last, closed-form solutions exist only for manipulators with special designs. These closed-form solutions can be achieved by deriving the equations from the geometry of the manipulator or by algebraic transformations of the forward kinematics equations. In the general case, numerical methods such as the Newton-Raphson method can be used to determine the inverse kinematics, which though, usually only determine a single solution close to the provided initial guess, whereas by the analytical approach, all solutions can be derived. An example for the numerical inverse kinematics of a 2 DOF manipulator can be found in Lynch & Park (2017, pp.231).

The inverse velocity kinematics can generally be solved, using the inverse of the Jacobian matrix, if the robot is non redundant and the configuration $\mathbf{q}$ is nonsingular:

$$\dot{\mathbf{q}} = \mathbf{J}^{-1}(\mathbf{q})\mathbf{v}_{\mathcal{E}}$$

In the case of a redundant manipulator, one can resort to the pseudo inverse

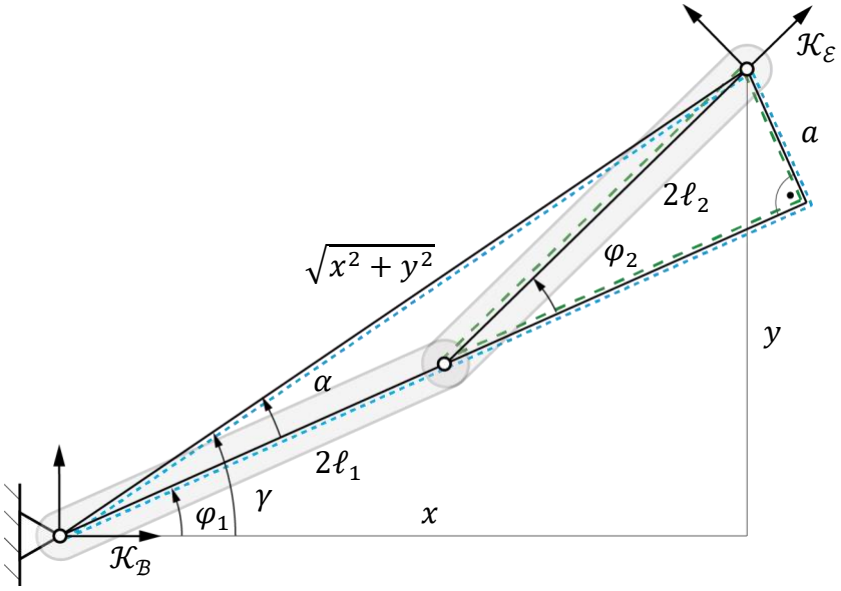
$$\dot{\mathbf{q}} = \mathbf{J}^{\dagger}(\mathbf{q})\mathbf{v}_{\mathcal{E}} \quad \text{with} \quad \mathbf{J}^{\dagger} = \mathbf{J}^{T}(\mathbf{J}\mathbf{J}^{T})^{-1}$$

which effectively minimizes the joint velocities. For a more elaborate derivation one may be referred to Siciliano (2010, pp. 123).

Example

Following up on the derivation of the forward kinematics, the inverse kinematics will be derived for the 2 DOF manipulator. Theoretically, equation (4) can be solved for φ_1 and φ_2 as functions of x and y. Even though, the simplicity of these two equations makes this approach very tempting, one may be encouraged to solve these equations using, for instance, symbolic processing software as can be accessed on https://www.wolframalpha.com/. The resulting equations turn out surprisingly cumbersome, which demonstrates, that the derivation of closed form solutions can be somewhat exhaustive.

Derivation of Inverse Kinematics for a Manipulator With 2 DOF



Source: Florian Simroth (2023).

Therefore, for the example at hand, a geometric approach based on FIGURE 6 is chosen to derive the inverse kinematics equations. The squared distance between the base and end effector frame corresponds to $x^2 + y^2$ which can likewise be expressed by the path along the links of the manipulator. This relation can be

expressed by squaring and adding both equations which after some simplification results in

$$4(2\ell_2\ell_1\cos(\varphi_2) + \ell_1^2 + \ell_2^2) - x^2 - y^2 = 0$$

from which the angle φ_2 can be derived as

$$\varphi_2 = \pm\arccos\left(\frac{x^2 + y^2 - 4\ell_1^2 - 4\ell_2^2}{8\ell_1\ell_2}\right)$$

Following Tzafestas (2014, p.392), the shared edge a of the two highlighted triangles can be determined for each triangle:

$$\begin{aligned} a &= \sin(\alpha)\sqrt{x^2 + y^2} \\ a &= \sin(\varphi_2)\ell_2 \end{aligned}$$

from which the angle α can be derived as

$$\alpha = \arcsin\left(\frac{\sin(\varphi_2)\ell_2}{\sqrt{x^2 + y^2}}\right)$$

The indicated angle γ can be extracted using the arctan2 function

$$\gamma = \text{arctan2}(y, x)$$

The delta between these two angles corresponds to the remaining angle φ_1:

$$\varphi_1 = \gamma - \alpha$$

Inspecting the manipulator, the following observations can be made regarding the number of solutions for certain cases:

- 2 solutions: For the general case, where the end effector is within the dexterous workspace, there exist two solutions as displayed in FIGURE 3
- 1 solution: If (x, y) is on the circle of the fully outstretched (/contracted) end effector articulation point, only the solution $\varphi_2 = 0$ (/ $\varphi_2 = \pi$) exists

- 0 solutions: If (x, y) is outside the workspace, the term $(x^2 + y^2 - 4\ell_1^2 - 4\ell_2^2)/(8\ell_1\ell_2) > 1$ and $\arccos x$ does not yield a solution
- ∞ solutions: For the special dimensions $\ell_1 = \ell_2$ any angle φ_1 is suitable to achieve $(x, y) = (0,0)$, for which end effector and base coincide

For the velocities, if the determinant of the 2x2 Jacobian matrix $\mathbf{J}_T$

$$\det(\mathbf{J}_T) = D = 4\ell_1\ell_2\sin(\varphi_2)$$

is unequal to zero, the inverse can be computed as

$$\mathbf{J}_T^{-1} = \frac{1}{D}\begin{pmatrix} 2\cos(\varphi_1 + \varphi_2)\ell_2 & 2\sin(\varphi_1 + \varphi_2)\ell_2 \\ -2(\cos(\varphi_1)\ell_1 + \cos(\varphi_1 + \varphi_2)\ell_2) & -2(\sin(\varphi_1)\ell_1 + \sin(\varphi_1 + \varphi_2)\ell_2) \end{pmatrix}$$

such that the joint velocities result from $\dot{\mathbf{q}} = \mathbf{J}_T^{-1}\mathbf{v}_{\mathcal{E}}$.

Singularities

Depending on the design of a manipulator, there may exist certain configurations, for which the manipulator loses some of its instantaneous degrees of freedom. These configurations are referred to as singularities and pose an issue in multiple ways (Siciliano, 2010, p.116):

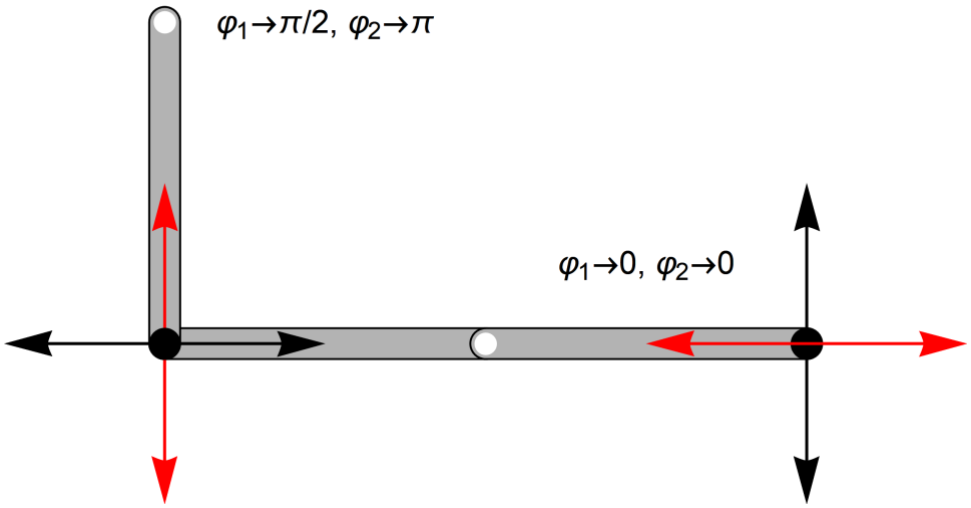
- The number of instantaneous "motion directions" is reduced
- There exists an infinite number of solutions for the inverse kinematics at that configuration
- When approaching a singular configuration, some of the joint velocities may become extremely large, with resulting numerical and physical problems

Singularity
Configuration at which the manipulator loses some of its instantaneous degrees of freedom.

According to Siciliano (2010, p.116), one can distinguish between boundary and internal **singularities**. Typical configurations prone to boundary singularities are at the limits of the workspace, were the manipulator is fully expanded or folded up. Internal singularities occur

within the boundaries of the workspace, for example, when multiple joint axes are aligned similar to a gimbal lock situation.

Two Singular Boundary Configurations (Extended and Retracted) of a 2 DOF Manipulator



Source: Florian Simroth (2023).

Due to the above-mentioned issues, it is necessary to carefully circumvent these configurations and close neighborhoods during path planning. Next to geometrical inspection, singularities can be identified mathematically by analyzing the Jacobian matrix. In fact, at a singular configuration, the Jacobian exhibits a reduced rank which will be demonstrated for two configurations of the 2 DOF manipulator shown in [FIGURE]. The Jacobians are derived for an extended $(\varphi_1 = 0,\ \varphi_2 = 0)$ and a retracted $(\varphi_1 = \pi/2,\ \varphi_2 = \pi)$ configuration:

$$\begin{pmatrix} v_x \\ v_y \end{pmatrix} = \mathbf{J}_{\mathbf{v}\varepsilon} \begin{pmatrix} \dot{\varphi}_1 \\ \dot{\varphi}_2 \end{pmatrix}$$

$\mathbf{J}_{\mathbf{v}_\varepsilon}$ maps the

$$\begin{aligned} \mathbf{J}_{\mathbf{v}\varepsilon}(\varphi_1 = 0,\ \varphi_2 = 0) &= \begin{pmatrix} 0 & 0 \\ 2(\ell_1 + \ell_2) & 2\ell_2 \end{pmatrix} \\ \mathbf{J}_{\mathbf{v}\varepsilon}(\varphi_1 = \pi/2,\ \varphi_2 = \pi) &= \begin{pmatrix} 2\ell_2 - 2\ell_1 & 2\ell_2 \\ 0 & 0 \end{pmatrix} \end{aligned}$$

In both cases, the rank of the Jacobian is reduced. For the first expression, v_x is always equal to 0, independently of the choice of $(\dot{\varphi}_1,\ \dot{\varphi}_2)^T$ and analogously $v_y = 0$ for the second case. This demonstrates the problematic for the inverse kinematics: When trying to determine the joint velocities $(\dot{\varphi}_1,\ \dot{\varphi}_2)^T$ necessary to achieve a certain end effector velocity v_x at the singular (horizontally) stretched out configuration, no solution can be obtained as that direction is blocked. On the other hand, when deriving the joint velocities for a desired v_y, an infinite amount of combinations of $(\dot{\varphi}_1,\ \dot{\varphi}_2)^T$ exist, that yield v_y.

Manipulability

An interesting aspect besides the pure workspace of a manipulator is, how well the end effector at a certain configuration is able to change its position and orientation in the operational space based on changes in its joint variables. As discussed previously, there may even exist singular configurations, where the robot loses the ability to move into certain directions. In this section, measures will be introduced to characterize the mobility of a configuration and shed light on how close the robot is to a singular configuration. An elaborate description can be found in Siciliano (2010, pp. 152) from which this section reviews the main aspects.

Starting with normalized input joint velocities expressed as $\dot{\mathbf{q}}^T\dot{\mathbf{q}} = 1$ which represent all points on a unit sphere (and its equivalent in n dimensions) in the configuration space, the question arises, how the output of the end effector velocities $\mathbf{v}_\mathcal{E}$ will look like. After solving $\mathbf{v}_\mathcal{E} = \mathbf{J}_{\mathbf{v}_\mathcal{E}}\dot{\mathbf{q}}$ for $\dot{\mathbf{q}}$ and inserting the result $\dot{\mathbf{q}} = \mathbf{J}_{\mathbf{v}_\mathcal{E}}^{-1}\mathbf{v}_\mathcal{E}$ into $\dot{\mathbf{q}}^T\dot{\mathbf{q}} = 1$ yields (Siciliano 2010, p.153)

$$\mathbf{v}_\mathcal{E}^T\left(\mathbf{J}_{\mathbf{v}_\mathcal{E}}\mathbf{J}_{\mathbf{v}_\mathcal{E}}^T\right)^{-1}\mathbf{v}_\mathcal{E} = 1 \tag{5}$$

This equation also holds for redundant manipulators with a task space dimension smaller than the manipulator degree of freedom. Equation (5) implicitly defines the manipulation ellipsoid for the end effector velocities.

The principal axes of the ellipsoid can be obtained using Eigenvector decomposition. The principal axes with the smallest and largest Eigenvalue indicate the direction of smallest and largest motion sensitivity, respectively. When approaching a singular configuration, the length of one axis vanishes, meaning that none of the normalized input joint velocities is capable to induce a motion along that direction and the instantaneous degree of freedom of the manipulator degenerates. Though, usually it is desired to operate a manipulator in configurations, where all axes of the ellipsoid are close to equal length, to achieve a uniform motion capability for which several indices are introduced.

Manipulability Measure
Number which characterizes the mobility of an endeffector at a certain configuration.

The **manipulability measure** w provides insights on how uniformly the end effector can move into all directions or how close the configuration is to a singularity. Its value is always larger or equal to zero $w >= 0$, while a value of zero indicates a singular configuration.

$$w = \sqrt{\det(\mathbf{J}\mathbf{J}^T)}$$

The condition number of a matrix emphasizes how big the effect of small errors of a vector multiplied by the matrix is on the resulting product. In terms of the manipulability ellipsoid, the minimum and maximum Eigenvalues λ_{min} and λ_{max} of $\mathbf{J}\mathbf{J}^T$ corresponding to the smallest and larges principal axes of the ellipsoid, can be used to derive the condition number. The closer the fraction of both Eigenvalues is to one, the more the shape resembles a sphere. Resultingly, values close to one are preferable for manipulator design (Lynch & Park, 2017, p. 199):

$$\mu = \frac{\lambda_{\text{max}}}{\lambda_{\text{min}}} \geq 1$$

Example

For the previously introduced 2 DOF planar manipulator, the manipulability ellipsoid for several configurations shall be derived. In addition, both, the manipulability

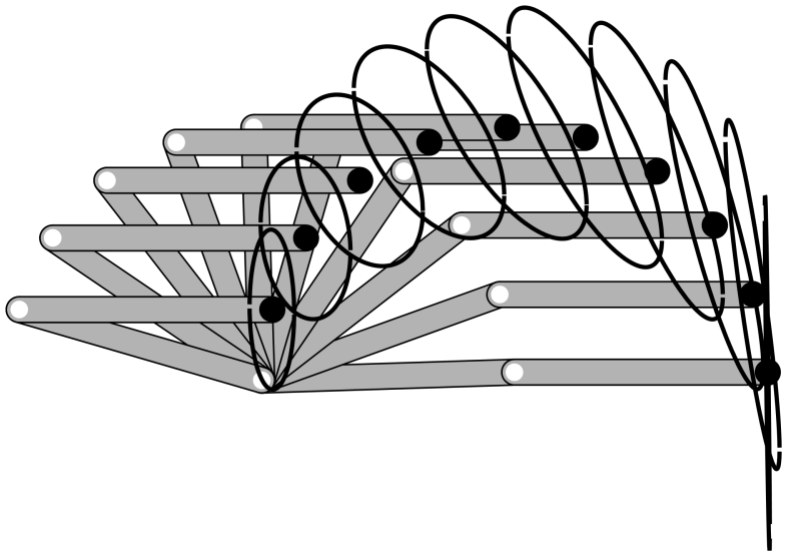
measure and the condition number shall be plotted for all configurations within the workspace.

First, equation (4) is applied to the 2 DOF planar manipulator considering the end effector velocity $\mathbf{v}_{\mathcal{E}} = \left(v_x, v_y\right)^T$:

$$\begin{aligned}\mathbf{v}_{\mathcal{E}}^T\left(\mathbf{J}_{\mathbf{v}_{\mathcal{E}}}\mathbf{J}_{\mathbf{v}_{\mathcal{E}}}^T\right)^{-1}\mathbf{v}_{\mathcal{E}} \quad &= \frac{1}{4\ell_1^2\ell_2^2\sin^2(\varphi_2)}(\ell_1^2\left(v_x\cos(\varphi_1)+v_y\sin(\varphi_1)\right)^2 \\ &+2\ell_2\ell_1\left(v_x\cos(\varphi_1)+v_y\sin(\varphi_1)\right) \\ &\cdot\left(v_x\cos(\varphi_1+\varphi_2)+v_y\sin(\varphi_1+\varphi_2)\right) \\ &+2\ell_2^2\left(v_x\cos(\varphi_1+\varphi_2)+v_y\sin(\varphi_1+\varphi_2)\right)^2) = 1\end{aligned}$$

This equation contains v_x and v_y as unknowns for a given configuration of φ_1 and φ_2. Therefore, by inserting values for v_x and solving for v_y, the results can be plotted as shown in [FIGURE]. Here, a sequence of configurations is displayed and the manipulability ellipsoid is displayed for each of the configurations by evaluation of the equation above. It becomes clear, that for the stretched configuration, the end effector velocity towards the base vanishes and the ellipse thins out. Along the sequence, the shape of the ellipse approaches a sphere (circle), before it thins out again at the second singularity, where the end effector coincides (for $\ell_1 = \ell_2$) with the base of the manipulator.

Manipulability Ellipsoids for a Sequence of Configurations of the 2 DOF Manipulator



Source: Florian Simroth (2023).

The manipulability measure w corresponds to the homogeneity of the manipulability ellipsoid. For the given manipulator, equation (4) yields:

$$w = \sqrt{\det(\mathbf{J}\mathbf{J}^T)} = 4\sqrt{\ell_1^2 \ell_2^2 \sin^2(\varphi_2)}$$

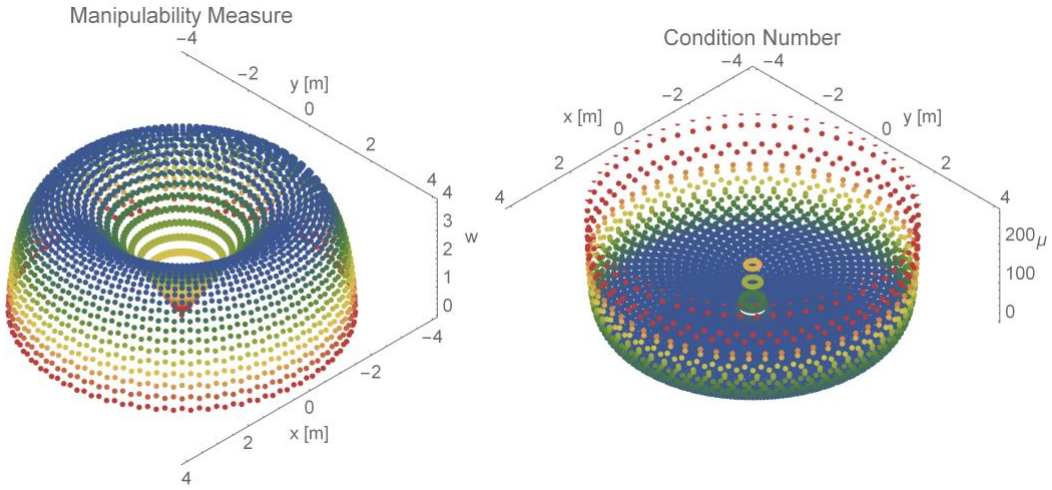
It might seem surprising, that the formula indeed only depends on φ_2. Yet, once understanding the rotation symmetry of its workspace, it becomes obvious, that for a given φ_2, w must be the same for any φ_1. [FIGURE] (left) shows the manipulability measure for each of the reachable x, y positions of the end effector in the operational space. When comparing the ellipsoids to the manipulability measure, the correspondence of higher values for w in between the outstretched and folded configuration is visible, while at the singulariteis, $w = 0$.

Similarly, the condition number $\mu = \lambda_{\text{max}}/\lambda_{\text{min}}$ with the eigenvalues is inspected:

$$\begin{aligned} \lambda_{\text{min}} &= 4\ell_2\ell_1\cos(\varphi_2) - 2c + 2\ell_1^2 + 4\ell_2^2 \\ \lambda_{\text{max}} &= 2(2\ell_2\ell_1\cos(\varphi_2) + c + \ell_1^2 + 2\ell_2^2) \\ c &= \sqrt{(2\ell_2\ell_1\cos(\varphi_2) + \ell_1^2 + 2\ell_2^2)^2 - 4\ell_1^2\ell_2^2\sin^2(\varphi_2)} \end{aligned}$$

Usually the eigenvalues need to be determined numerically instead of analytically as shown above. For the manipulator, the ratio of the two principal axes is lowest, i.e., closest to 1, for configurations other than the singular ones as shown in [FIGURE] (right). Often, the inverse of the condition number is plotted, to avoid values approaching infinity close to singular configurations (see for instance Pashkevich et al. (2006)) resulting in a similar image to that of w.

Manipulability Measure (Left) and Condition Number (Right) for the Workspace of a 2 DOF Manipulator



Source: Florian Simroth (2023).

Modeling Mobile Manipulators

A manipulator mounted on top of a mobile platform overcomes the workspace limitations of the manipulator by itself and therefore allows for a flexible application in various use cases. In some cases, it is sufficient or even required, that the mobile platform transports the manipulator to the desired place, and remains inactive during the manipulation task, such that the previously introduced concepts for mobile platforms and fixed manipulators can directly be applied in a decoupled way. Yet, sometimes it is desirable, to control both, the mobile platform and the manipulator simultaneously for which a combined kinematic and dynamic model is required. The

derivations in this section are mainly reworked from Padois et al. (2007) who provide a great overall structured approach to modeling mobile manipulators.

Before the equations of motions for the mobile manipulator will be assembled, an appropriate notation shall be established. For this purpose, four coordinate frames are of importance:

- $\mathcal{K}_{\mathcal{W}}$: World-fixed frame as a reference for all motions
- $\mathcal{K}_{\mathcal{R}}$: Robot-fixed frame used to describe the robot's pose
- $\mathcal{K}_{\mathcal{B}}$: Robot-fixed frame at the root of the mobile manipulator
- $\mathcal{K}_{\mathcal{E}}$: End effector fixed frame used for the manipulation task

In order to distinguish components induced by the robot platform and by the manipulator, the index $\mathcal{R}$ will be used for items related to the mobile robot, while $\mathcal{M}$ indicates items regarding the manipulator. For the **mobile robot**, the following notation will be used:

- $\boldsymbol{\xi}_{\mathcal{R}}$, $\dot{\boldsymbol{\xi}}_{\mathcal{R}}$: Operational space coordinates and velocities of the mobile platform. For planar mobile robots, usually the position and orientation coordinates $\boldsymbol{\xi}_{\mathcal{R}} = (x_{\mathcal{R}},\ y_{\mathcal{R}},\ \theta_{\mathcal{R}})^T$ are sufficient to describe its pose in operational space.
- $\mathbf{q}_{\mathcal{R}}$, $\dot{\mathbf{q}}_{\mathcal{R}}$: Configuration space coordinates and velocities of the mobile platform. In general, $\mathbf{q}_{\mathcal{R}} = (\boldsymbol{\xi}_{\mathcal{R}}^T, \boldsymbol{\varphi}_{\mathcal{R}}^T, \boldsymbol{\beta}_{\mathcal{R}}^T)^T$, corresponding to the platform pose, wheel spin angles $\boldsymbol{\varphi}_{\mathcal{R}} = (\boldsymbol{\varphi}_{\mathrm{f}},\ \boldsymbol{\varphi}_{\mathrm{s}},\ \boldsymbol{\varphi}_{\mathrm{c}},\ \boldsymbol{\varphi}_{\mathrm{m}})^T$, and steering/attachment angles $\boldsymbol{\beta}_{\mathcal{R}} = (\boldsymbol{\beta}_{\mathrm{c}},\ \boldsymbol{\beta}_{\mathrm{s}})^T$ of the components relevant for the regarded robot design (c: off-centered (caster) wheel, s: centered (steering) wheel, f: fixed wheel, m: omnidirectional (Swedish/Mecanum) wheel).
- $\mathbf{u}_{\mathcal{R}}$: Control inputs of the mobile platform, usually composed of the independent (pseudo) velocities or actuated wheel spins.

Analogously, for the manipulator, the following terms are used:

- $\boldsymbol{\xi}_{\mathcal{M}}$, $\dot{\boldsymbol{\xi}}_{\mathcal{M}}$: Operational space coordinates and velocities of the end effector of the *fixed* manipulator with respect to its base. In the general spatial case, $\boldsymbol{\xi}_{\mathcal{M}} = (x_{\mathcal{M}},\, y_{\mathcal{M}},\, z_{\mathcal{M}},\, \gamma_{\mathcal{M}},\, \beta_{\mathcal{M}},\, \alpha_{\mathcal{M}})^T$ using Cartesian coordinates for the position and ZYX Euler angles for the orientation.
- $\mathbf{q}_{\mathcal{M}}$, $\dot{\mathbf{q}}_{\mathcal{M}}$: Configuration space coordinates and velocities of the manipulator such as the joint variables.
- $\mathbf{u}_{\mathcal{M}}$: Control inputs of the manipulator for which the joint velocities $\dot{\mathbf{q}}_{\mathcal{M}}$ usually are a suitable choice.

With this notation, the overall model of the mobile manipulator can be established. First, the direct kinematics of the end effector with respect to a world reference frame can be expressed as:

$$ {}^{\mathcal{W}}\mathbf{T}_{\mathcal{E}} = {}^{\mathcal{W}}\mathbf{T}_{\mathcal{R}}(\mathbf{q}_{\mathcal{R}}) \underbrace{{}^{\mathcal{R}}\mathbf{T}_{\mathcal{B}}}_{\text{const.}} {}^{\mathcal{B}}\mathbf{T}_{\mathcal{E}}(\mathbf{q}_{\mathcal{M}}) $$

where ${}^{\mathcal{R}}\mathbf{T}_{\mathcal{B}}$ is the constant transformation from the robot's reference coordinate frame $\mathcal{K}_{\mathcal{R}}$ to the base $\mathcal{K}_{\mathcal{B}}$ of the manipulator, while ${}^{\mathcal{W}}\mathbf{T}_{\mathcal{R}}$ and ${}^{\mathcal{B}}\mathbf{T}_{\mathcal{E}}$ refer to previously introduced robot and manipulator kinematics. When using the generalized operational space coordinates

$$ \boldsymbol{\xi} = (x,\, y,\, z,\, \gamma,\, \beta,\, \alpha)^T $$

specifying the position in Cartesian coordinates and the orientation in ZYX Euler angles of the end effector, the direct kinematics can also be expressed by the function $\mathbf{f}$ which maps the configuration coordinates of the combined mobile manipulator

$$ \mathbf{q} = (\mathbf{q}_{\mathcal{R}}^T,\, \mathbf{q}_{\mathcal{M}}^T)^T = (\boldsymbol{\xi}_{\mathcal{R}}^T, \boldsymbol{\varphi}_{\mathcal{R}}^T, \boldsymbol{\beta}_{\mathcal{R}}^T,\, \mathbf{q}_{\mathcal{M}}^T)^T $$

to operational space:

$$ \boldsymbol{\xi} = \mathbf{f}(\boldsymbol{\xi}_{\mathcal{R}},\, \mathbf{q}_{\mathcal{M}}) $$

Note that $\mathbf{f}$ does not directly depend on the wheel spin and steering angles. The components of $\boldsymbol{\xi}$ and $\mathbf{f}$ can be expressed in terms of the robot pose coordinates $\boldsymbol{\xi}_{\mathcal{R}}$ and the manipulator coordinates $\boldsymbol{\xi}_{\mathcal{M}}$ derived earlier in this section for the fixed manipulator. Following Bayle et al. (2003), the end effector coordinates $\boldsymbol{\xi}$ for a general 6 DOF manipulator mounted on a wheeled mobile robot moving in the plane whose pose is determined by $\boldsymbol{\xi}_{\mathcal{R}} = (x_{\mathcal{R}}, y_{\mathcal{R}}, \theta_{\mathcal{R}})^T$ can be derived as:

$$\begin{aligned} x &= x_{\mathcal{R}} + \cos(\theta_{\mathcal{R}})(a + x_{\mathcal{M}}) - \sin(\theta_{\mathcal{R}})(b + y_{\mathcal{M}}) \quad (6.1) \\ y &= y_{\mathcal{R}} + \sin(\theta_{\mathcal{R}})(a + x_{\mathcal{M}}) + \cos(\theta_{\mathcal{R}})(b + y_{\mathcal{M}}) \quad (6.2) \\ z &= c + z_{\mathcal{M}} \\ \gamma &= \theta_{\mathcal{R}} + \gamma_{\mathcal{M}} \\ \beta &= \beta_{\mathcal{M}} \\ \alpha &= \alpha_{\mathcal{M}} \end{aligned} \quad (6)$$

Here, $(a,\ b,\ c)$ describe the position offset of the manipulator base and the robot reference frame. Note that from all robot configuration parameters in $\mathbf{q}_{\mathcal{R}}$, only the pose of the platform $\boldsymbol{\xi}_{\mathcal{R}}$ is of relevance for the end effector pose $\boldsymbol{\xi}$. These equations assume that the robot coordinate frame and the base frame of the mounted manipulator are oriented equally, such that the end effector coordinates $\boldsymbol{\xi}_{\mathcal{M}}$ can be transformed from the robot coordinate frame to the world coordinate frame using $\theta_{\mathcal{R}}$. For the orientation coordinates, the z-rotations $\theta_{\mathcal{R}}$ and $\gamma_{\mathcal{M}}$ add up as the robot moves in the plane and rotates about the z-axis, while the other rotation parameters are based on the manipulator only.

The differentiation of the equations above can be split into two components, where $\mathbf{J}_{\boldsymbol{\xi}_{\mathcal{M}},\mathrm{T}}$ and $\mathbf{J}_{\boldsymbol{\xi}_{\mathcal{M}},\mathrm{R}}$ refer to the translational and rotational part of the manipulator Jacobian (Bayle et al., 2003):

$$
\begin{aligned}
\dot{\boldsymbol{\xi}} &= \frac{\partial \mathbf{f}}{\partial \boldsymbol{\xi}_{\mathcal{R}}} \dot{\boldsymbol{\xi}}_{\mathcal{R}} + \frac{\partial \mathbf{f}}{\partial \mathbf{q}_{\mathcal{M}}} \dot{\mathbf{q}}_{\mathcal{M}} = \mathbf{J}(\mathbf{q}) \dot{\mathbf{q}} \\
\frac{\partial \mathbf{f}}{\partial \boldsymbol{\xi}_{\mathcal{R}}} &= \begin{pmatrix} 1 & 0 & -\sin(\theta_{\mathcal{R}})(a + x_{\mathcal{M}}) + \cos(\theta_{\mathcal{R}})(b + y_{\mathcal{M}}) \\ 0 & 1 & \cos(\theta_{\mathcal{R}})(a + x_{\mathcal{M}}) - \sin(\theta_{\mathcal{R}})(b + y_{\mathcal{M}}) \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \\
\frac{\partial \mathbf{f}}{\partial \mathbf{q}_{\mathcal{M}}} &= \begin{pmatrix} \mathbf{R}_z(\theta_{\mathcal{R}}) \cdot \mathbf{J}_{\xi_{\mathcal{M}}, \mathrm{T}} \\ \mathbf{J}_{\xi_{\mathcal{M}}, \mathrm{R}} \end{pmatrix}
\end{aligned} \tag{7}
$$

Yet, the relations provided above do not fully model the mobile manipulator, since the operational velocities of the robotic platform are subject to the non-holonomic wheel constraints. Therefore, as was already shown in the unit on kinematics, a vector of independent *mobility* control inputs $\mathbf{u}_{\mathcal{R},m}$ holds inputs that can arbitrarily be chosen. With the matrix $\boldsymbol{\Sigma}$, the following relation

$$
\dot{\boldsymbol{\xi}}_{\mathcal{R}} = \mathbf{R}_z(\theta_{\mathcal{R}}) \cdot \boldsymbol{\Sigma}(\boldsymbol{\beta}_{\mathrm{s}}) \cdot \mathbf{u}_{\mathcal{R},m} \tag{8}
$$

maps the *mobility* control inputs to a set of platform velocities that always comply with the wheel constraints. The matrix $\boldsymbol{\Sigma}$ was derived in a way that

$$
\mathbf{C}_{1\mathrm{fs}}(\boldsymbol{\beta}_{\mathrm{s}}) \boldsymbol{\Sigma} = 0
$$

where $\mathbf{C}_{1\mathrm{fs}}(\boldsymbol{\beta}_{\mathrm{s}})$ originates from the no-sliding constraints of the fixed and steering wheels. Inserting these relations into (8), yields a relation between the independent velocity control inputs and the end effector generalized velocities

$$
\dot{\boldsymbol{\xi}} = \frac{\partial \mathbf{f}}{\partial \boldsymbol{\xi}_{\mathcal{R}}} \mathbf{R}_z(\theta_{\mathcal{R}}) \cdot \boldsymbol{\Sigma}(\boldsymbol{\beta}_{\mathrm{s}}) \cdot \mathbf{u}_{\mathcal{R},m} + \frac{\partial \mathbf{f}}{\partial \mathbf{q}_{\mathcal{M}}} \dot{\mathbf{q}}_{\mathcal{M}} = \bar{\mathbf{J}}(\mathbf{q}) \dot{\mathbf{q}} \tag{9}
$$

The kinematic model, that maps the control inputs to the full set of generalized coordinates in configuration space can be achieved, by recalling that the matrix $\mathbf{S}_{\mathcal{R}}$ maps the control inputs $\mathbf{u}_{\mathcal{R}}$ of the platform to its configuration variables $\dot{\mathbf{q}}_{\mathcal{R}}$

$$
\dot{\mathbf{q}}_{\mathcal{R}} = \mathbf{S}_{\mathcal{R}} \mathbf{u}_{\mathcal{R}}
$$

with the *mobility* $\mathbf{u}_{\mathcal{R},m}$ and *steerability* $\mathbf{u}_{\mathcal{R},s}$ control inputs $\mathbf{u}_{\mathcal{R}} = \left(\mathbf{u}_{\mathcal{R},m}^T,\ \mathbf{u}_{\mathcal{R},s}^T\right)^T$, while for the manipulator, the joint velocities directly correspond to the control inputs

$$\dot{\mathbf{q}}_{\mathcal{M}} = \mathbf{u}_{\mathcal{M}}$$

such that the direct kinematic model results:

$$\dot{\mathbf{q}} = \underbrace{\begin{pmatrix} \mathbf{S}_{\mathcal{R}} & 0 \\ 0 & \mathbf{I} \end{pmatrix}}_{\breve{\mathbf{S}}} \mathbf{u} \tag{10}$$

with $\mathbf{I}$ being the identity matrix, and $\mathbf{u}_{\mathcal{R}}$ being the combined control inputs $\mathbf{u}_{\mathcal{R}} = (\mathbf{u}_{\mathcal{R}}^T,\ \mathbf{u}_{\mathcal{M}}^T)^T$.

It shall be noted, that for the inverse kinematics of redundant mobile manipulators, a choice has to made in regards of how to handle the additional degrees of freedom. For this case, using the pseudo inverse minimizes the overall joint motions, while alternatively, a mass/inertia based weighting matrix can be used to penalize motion of the platform over motion of the manipulator. Other approaches make use of the (combined) manipulability measure, to keep the manipulator away from singular configurations.

The dynamics were previously derived including the Lagrange multipliers as

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{V}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{Q} + \mathbf{A}^T(\mathbf{q})\boldsymbol{\lambda}_n$$

After left multiplication with $\mathbf{S}^T$ and replacing instances of $\mathbf{q}$ and its derivations by the control inputs according to equation (10) and its derivative, the following equation is achieved

$$\underbrace{\mathbf{S}^T\mathbf{M}\mathbf{S}}_{\breve{\mathbf{M}}}\dot{\mathbf{u}} + \underbrace{\mathbf{S}^T\mathbf{M}\dot{\mathbf{S}}}_{\breve{\mathbf{V}}}\mathbf{u} + \mathbf{S}^T\mathbf{V} = \underbrace{\mathbf{S}^T\mathbf{Q}}_{\breve{\mathbf{Q}}} + \underbrace{\mathbf{S}^T\mathbf{A}^T}_{=0}\boldsymbol{\lambda}_n$$

in which dynamics already implicitly imply the constraints through the introduction of $\mathbf{u}$.

The generalized force vector $\mathbf{Q} = \mathbf{Q}_{\mathcal{M}} + \mathbf{Q}_g + \mathbf{Q}_d + \mathbf{Q}_{ext}$ is composed of forces $\mathbf{Q}_{\mathcal{M}}$ originating from the actuators, gravity forces $\mathbf{Q}_g = -\mathbf{g}(\mathbf{q})$, friction and other disturbances $\mathbf{Q}_d$, and external forces $\mathbf{Q}_{ext}$ for the interaction with objects.

The components of the mass matrix have the following structure (Padois et al. 2007, p.16):

$$\widetilde{\mathbf{M}}(\mathbf{q}) = \begin{pmatrix} \widetilde{\mathbf{M}}_{\mathcal{R}}(\mathbf{q}) + \widetilde{\mathbf{M}}_{\mathcal{MR}}(\mathbf{q}) & \widetilde{\mathbf{M}}_{\mathcal{RM}}(\mathbf{q})^T \\ \widetilde{\mathbf{M}}_{\mathcal{RM}}(\mathbf{q}) & \widetilde{\mathbf{M}}_{\mathcal{M}}(\mathbf{q}_{\mathcal{M}}) \end{pmatrix}$$

The matrices $\widetilde{\mathbf{M}}_{\mathcal{R}}$ and $\widetilde{\mathbf{M}}_{\mathcal{M}}$ are the mass matrices compliant with the constraints of the detached robot platform and the manipulator, respectively. Adding $\widetilde{\mathbf{M}}_{\mathcal{MR}}$ to $\widetilde{\mathbf{M}}_{\mathcal{R}}$ accounts for the additional mass of the manipulator mounted on the platform. The coupling between the platform and the manipulator is established by $\widetilde{\mathbf{M}}_{\mathcal{RM}}$ which represents the impact of the acceleration of the robot platform on the manipulator. In $\widetilde{\mathbf{V}}$ also Coriolis and centrifugal terms of the manipulator, the platform, and their coupling are contained which, though, will not be carried out further.

Self-Check Questions

1. Which of the following is true for the manipulability measure w?

☐ At singular configurations, w equals one.

☐ At singular configurations, w approaches infinity.

☐ negative values of w indicate end effector positions out of the workspace.

☐ *the maximum value of w is different for any manipulator design*.

2. Please complete the following sentence.

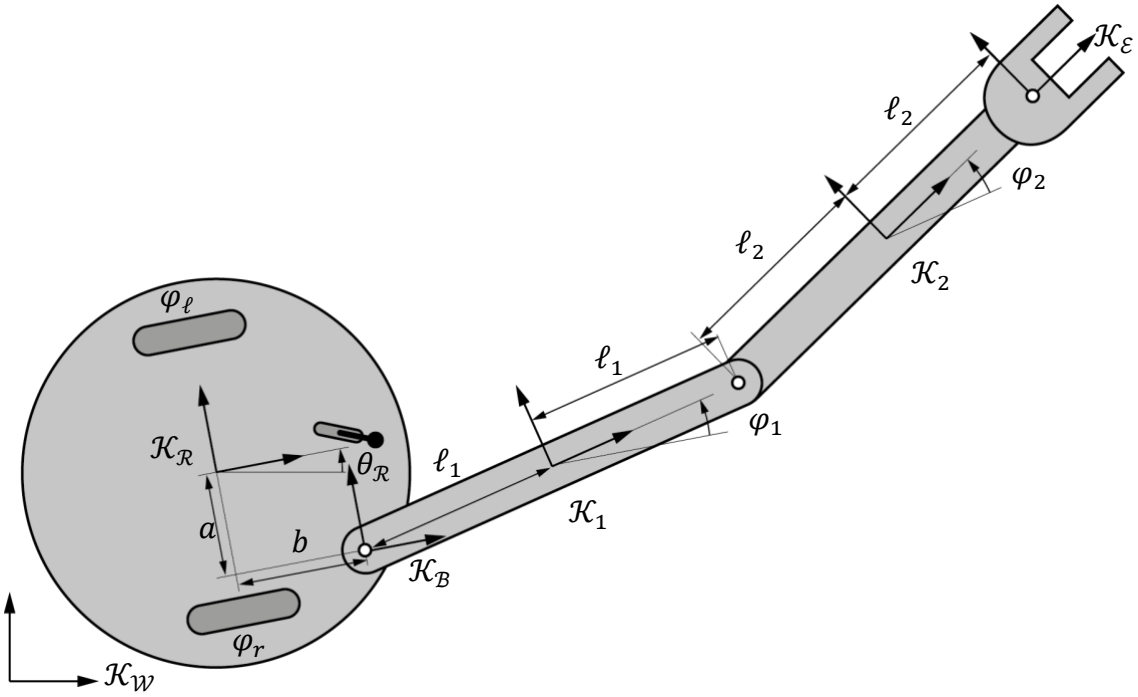
At a singular configuration, the instantaneous degree of freedom is *reduced*.

5.3 Examples

For the differential drive robot with a mounted planar 2 DOF manipulator depicted in FIGURE, the kinematics and dynamics shall be derived. Indeed, both components, the mobile platform as well as the manipulator have already been treated

independently throughout this lecture. The previous results shall be reused where possible to represent the combined system of the mobile manipulator.

Mobile Manipulator: Differential Drive With Mounted 2 DOF Manipulator



Source: Florian Simroth (2023).

Kinematics

The following generalized coordinates shall be used to specify the mobile manipulator (the caster wheel will be omitted):

$$\mathbf{q} = (\mathbf{q}_\mathcal{R}, \mathbf{q}_\mathcal{M}) = (\boldsymbol{\xi}_\mathcal{R}, \varphi_\ell, \varphi_r, \varphi_1, \varphi_2)^T$$

Furthermore, it is assumed, that only the x and y coordinate of the end effector are relevant for the robot's task such that

$$\boldsymbol{\xi} = (x, y)^T$$

The direct kinematics equation then follow by inserting x_M and y_M from (4)

$$\begin{aligned} x_M &= 2\big(\ell_1\cos(\varphi_1) + \ell_2\cos(\varphi_1+\varphi_2)\big) \\ y_M &= 2\big(\ell_1\sin(\varphi_1) + \ell_2\sin(\varphi_1+\varphi_2)\big) \end{aligned}$$

into equations (6.1) and (6.2):

$$\boldsymbol{\xi} = \begin{pmatrix} x_{\mathcal{R}} \\ y_{\mathcal{R}} \end{pmatrix} + \mathbf{R}_z(\theta_{\mathcal{R}}) \begin{pmatrix} a + x_M \\ b + y_M \end{pmatrix} \tag{11}$$

For the generalized velocity in (9)

$$\dot{\boldsymbol{\xi}} = \frac{\partial \mathbf{f}}{\partial \boldsymbol{\xi}_{\mathcal{R}}} \mathbf{R}_z(\theta_{\mathcal{R}}) \cdot \boldsymbol{\Sigma}(\boldsymbol{\beta}_s) \cdot \mathbf{u}_{\mathcal{R},m} + \frac{\partial \mathbf{f}}{\partial \mathbf{q}_{\mathcal{M}}} \dot{\mathbf{q}}_{\mathcal{M}} \tag{12}$$

the component $\partial \mathbf{f} / \partial \boldsymbol{\xi}_{\mathcal{R}}$ of the mobile robot was derived in **(7)** which after projection to the inputs $\mathbf{u}_{\mathcal{R},m} = (v,\, \omega)^T$ yields

$$\begin{aligned} \frac{\partial \mathbf{f}}{\partial \boldsymbol{\xi}_{\mathcal{R}}} \mathbf{R}_z(\theta_{\mathcal{R}}) \cdot \boldsymbol{\Sigma}(\boldsymbol{\beta}_s) &= \begin{pmatrix} 1 & 0 & -\sin(\theta_{\mathcal{R}})(a + x_{\mathcal{M}}) + \cos(\theta_{\mathcal{R}})(b + y_{\mathcal{M}}) \\ 0 & 1 & \cos(\theta_{\mathcal{R}})(a + x_{\mathcal{M}}) - \sin(\theta_{\mathcal{R}})(b + y_{\mathcal{M}}) \end{pmatrix} \\ &\cdot \begin{pmatrix} \cos(\theta_{\mathcal{R}}) & -\sin(\theta_{\mathcal{R}}) & 0 \\ \sin(\theta_{\mathcal{R}}) & \cos(\theta_{\mathcal{R}}) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \\ &= \begin{pmatrix} -\sin(\theta_{\mathcal{R}}) & -\sin(\theta_{\mathcal{R}})(a + x_{\mathcal{M}}) + \cos(\theta_{\mathcal{R}})(b + y_{\mathcal{M}}) \\ \cos(\theta_{\mathcal{R}}) & \cos(\theta_{\mathcal{R}})(a + x_{\mathcal{M}}) - \sin(\theta_{\mathcal{R}})(b + y_{\mathcal{M}}) \end{pmatrix} \end{aligned}$$

The component of the (fixed) manipulator was already stated in (3):

$$\frac{\partial \mathbf{f}}{\partial \mathbf{q}_{\mathcal{M}}} = \begin{pmatrix} -2\big(\ell_1\sin(\varphi_1) + \ell_2\sin(\varphi_1+\varphi_2)\big) & -2\ell_2\sin(\varphi_1+\varphi_2) \\ 2\big(\ell_1\cos(\varphi_1) + \ell_2\cos(\varphi_1+\varphi_2)\big) & 2\ell_2\cos(\varphi_1+\varphi_2) \end{pmatrix}$$

With (11) and (12) it is now possible to express the end effector position and velocity based on its current configuration. The combined control inputs of the robot and manipulator $\mathbf{u} = (v,\, \omega,\, \dot{\varphi}_1,\, \dot{\varphi}_2)^T$ can be mapped to the generalized velocities $\dot{\mathbf{q}}$ using the matrix $\mathbf{S}$:

$$\dot{\mathbf{q}} = \mathbf{S}\mathbf{u} = \begin{pmatrix} \mathbf{R}_z(\theta_{\mathcal{R}})\Sigma(\mathbf{q}) & \mathbf{0}_{3x2} \\ \mathbf{0}_{2x2} & \mathbf{I}_{2x2} \end{pmatrix} \mathbf{u} = \begin{pmatrix} -\sin(\theta_{\mathcal{R}}) & 0 & 0 & 0 \\ \cos(\theta_{\mathcal{R}}) & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} v \\ \omega \\ \dot{\varphi}_1 \\ \dot{\varphi}_2 \end{pmatrix}$$

Note that due to the absence of steering and (modelled) caster wheels, only the $\mathbf{R}_z(\theta_{\mathcal{R}})\Sigma(\mathbf{q})$ term within the $\mathbf{S}_{\mathcal{R}}$ matrix of the mobile robot remains.

Summary

Manipulators are an interface for a mobile robot to its environment which is composed of links connected by joints and an end effector, such as a gripper, which allows to interact with objects and fulfill the robot's task. For serial or open kinematic chain manipulators, there exists exactly one alternating sequence of joints and links, from the manipulator's base to the end effector, while multiple paths exist for parallel or closed kinematic chain manipulators.

The reachable workspace includes all points that the end effector can reach, whereas the dexterous workspace only includes the reachable points for which also the orientation mobility is preserved. The configuration space of a manipulator is spanned by all joint coordinates in the case of serial manipulators. The task space depends on the task to fulfill. For example, if the task for the robot is to write on a whiteboard, the task space may be composed of the translations only if the orientation of the pen is not required to be changed.

A manipulator is considered to be redundant with respect to its task, if it possesses more degrees of freedom than required to fulfill its task. A certain configuration is termed "singular", if the manipulator experiences a loss of some instantaneous degrees of freedom which can occur both, at the boundary in an outstretched configuration or internally due to special arrangement of joint axes, effectively prohibiting some motion directions. Both, redundancy and singularities require special treatment in inverse kinematics, as for a redundant manipulator, infinite solutions exist to achieve a certain end effector pose, while at singularities, no solution exist for certain motions.

The shape of the manipulability ellipsoid indicates how sensitive the end effector motion into certain directions is with respect to changes within the joint variables. A circular shape indicates that the manipulator can move equally into all directions, which is usually preferred. The manipulability measure or condition number provides a scalar value of how isotropically the end effector can move or on the contrary, how close a configuration is to a singularity.

When modelling mobile manipulators, large portions of the equations can be derived individually for the mobile platform and the manipulator. Though, when assembling manipulator and platform to a mobile manipulator, additional coupling terms arise, due to the interaction of motions within the manipulator and the platform. These interactions were demonstrated for a differential drive mobile robot with a planar 2 DOF manipulator mounted on top.

Unit 6 – Path, Motion, and Task Planning

Study Goals

On completion of this unit, you will be able to …

… understand different approaches for motion planning

… apply different planning algorithms to find a path between two points

… select appropriate environmental representation and pick suitable algorithm for a specific problem

… describe the fundamental elements and steps of task planning

Unit 6 – Path, Motion, and Task Planning

Introduction

Under planning, a variety of problems and tasks can be understood, depending on the context and areas of application. In the light of mobile robotics, path, motion, and task planning are of particular interest. While path and motion planning are concerned with the actual movement of the robot in between two locations or more precisely between two configurations, task planning operates on a more general level.

Problems in the domain of task planning can reach from process planning in factories, over solving puzzles to pick and place tasks. More relevant for mobile robotics, are, for instance, transportation of goods or people, stacking of objects, or assembly tasks. The general task planners start from an initial state which shall be transformed into the end state. To this end, some kind of representation or the task world needs to be found and the transformation has to be conducted based on a set of rules which are consecutively applied. Task planning often takes place in the domain of artificial intelligence and mostly needs to deal with discrete states.

Path and motion planning are concerned with the actual motion of the robot. The start and end state are two configurations for which a smooth transition needs to be found. A suitable representation of the workspace is required in which the motion takes place. This environment can be populated with obstacles, for collision avoidance is another aspect to be taken care of by the motion planning algorithms. With respect to nonholonomic mobile robots, not any geometric path found between a start and terminal configuration is actually feasible, such that additional constraints need to be considered during the search.

Unfortunately, there is no single algorithm that solves all of the problems above. Hence, it is only possible to present a selection of different algorithms and approaches in this unit. With these concepts it is already possible to solve many fundamental tasks and the presented ideas lay the foundation to further explore this fascinating area for more advanced topics.

6.1 Basics

Motion planning is complex problem, with a variety of existing approaches aiming at different sub problems. Before presenting a handful of concepts for motion planning, it is therefore necessary to shed some light to different classes of problems and terminology.

Geometrical path planning usually refers to finding a continuous sequence of configurations $\mathbf{q}(s)$ of a robot from an initial configuration $\mathbf{q}_{\text{start}} = \mathbf{q}(0)$ to a desired target configuration $\mathbf{q}_{\text{end}} = \mathbf{q}(1)$ where $s \in [0,1]$. In that sense, path planning can be distinguished from motion planning, as it does not take into account dynamics, time or constraints arising from the motion constraints or limits of the control inputs of the robot (Lynch, 2017, p. 355). A classic example often cited for path planning is the *piano mover's problem*. Essentially, this problem consists in finding a (geometrical) path for moving a bulky piano from one location to another for which tilting and rotations may be required to avoid collisions in narrow passages. For this problem it is assumed, that the piano can freely be moved and oriented in all directions in space.

For the piano mover's problem, the knowledge of the full environment prior to the planning is presumed, in order to find an appropriate path. Often, this is referred to as global planning, i.e., requiring a map within which the path planning takes place. For a mobile robot, this is not always available and the robot has to rely on sensor measurement's which capture the immediate environment in which then local planning is conducted to avoid collision with obstacles. Somewhat related to global

and local planning are the concepts of offline and online planning. Offline planning is performed before actually executing the motion (Choset et al., 2005, p. 2), thus requiring a sufficient amount of time for planning and knowledge of the surrounding. Online planning on the other hand is conducted if fast responses are required, i.e., to circumvent dynamic obstacles. In order to deliver quick results, often only partial environmental information by the robot's sensors is considered for motion planning.

Depending on the presence of obstacles that may change their location during the time of execution, dynamic environments and static environments can be distinguished (Klančar et al., 2017, p. 162). For dynamic environments with moving obstacles, such as for self driving cars within traffic, the velocity of objects needs to be taken into account, to anticipate the future locations for collision avoidance.

If, in particular, motion constraints of non-omnidirectional robots are of concern, often the term nonholonomic planning or planning with differential constraints is used to emphasize this aspect. In fact, this can be already be quite difficult, even if collision with objects are not of concern, as the example of parallel parking for a car-like robot illustrates.

In the broader sense, motion planning can be understood as the encapsulating term that takes into account any of the previously mentioned aspects to produce feasible trajectories to control the robot. To generalize the algorithms, motion planning is often conducted in the configuration space which will be further elaborated hereafter.

Motion Planning
Planning of robot motion considering kinematic constraints, dynamic constraints, and obstacles.

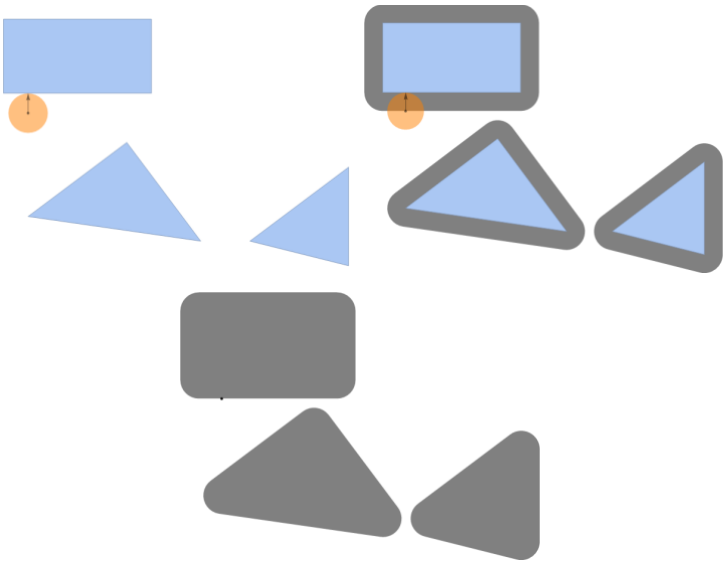
Configuration Space

The concept of the configuration space has already been addressed in preceding units. In particular, the configuration space was described as the set of all possible configurations, whereby each point refers to a single configuration. The dimension of the configuration space is equal to the number of degrees of freedom of the robot under consideration.

Up to this point, the configuration space was assumed to be free of any obstacles. Now, for the purpose of path planning, obstacles play an important role and shall be modeled within the configuration space. For this reason, the configuration space ($\mathcal{C}$-space) will be divided into the free space $\mathcal{C}_{\text{free}}$ and the obstacle space $\mathcal{C}_{\text{obs}}$. The free space is composed of all configurations which are collision free. The occupied space embodies configurations for which some parts of the robot would intersect with obstacles. The overall C-space is then the union $\mathcal{C} = \mathcal{C}_{\text{free}} \cup \mathcal{C}_{\text{obs}}$.

For illustration, a circular robot with radius r is considered. Its configuration can be described by $\mathbf{q} = (x, y)^T$. In its workspace, there exist multiple polygonal obstacles as shown in FIGURE1 (left). The space $\mathcal{C}_{\text{obs}}$ occupied by these obstacles within the configuration space can be derived by growing the obstacles by radius r and declaring the resulting configuration within the configuration space as occupied as shown in FIGURE1 (right). In this manner, the robot can be considered as a single point in the configuration space and a path from a start to an end configuration can be searched for within $\mathcal{C}_{\text{free}}$.

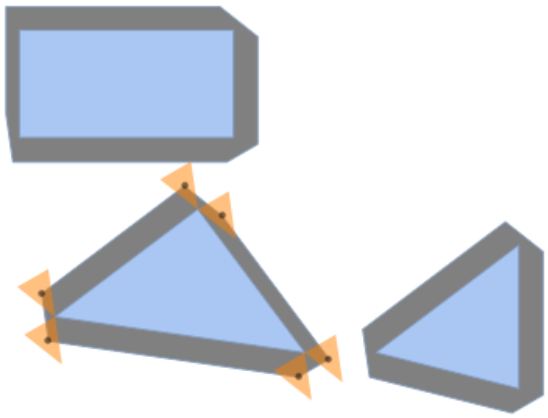
Workspace of Circular Robot (Orange) Which Is Inflated (Center) by the Robot Radius and Finally Yields Blue Obstacles



Source: Florian Simroth (2023).

In this simple example, the work space and the configuration space were of the same dimension, described by the same coordinates, which is usually not the case. If, for instance, the robot has a triangular shape, its configuration may be described by $\mathbf{q} = (x, y, \theta)^T$, such that the corresponding configuration space has three degrees of freedom. The free and obstacle space for one specific θ is shown in FIGURE2 which can be derived by the Minkowski sum of the robot and the obstacle, which effectively expands the obstacles as if the robot shape was shifted around the outer shape of the obstacle. If this process was repeated for all $\theta \in [0,2\pi]$, the three dimensional configuration space can be generated. Within this space, the path planning problem reduces again to searching for a simple path between two points. An elevation within the configuration space in this case is then equivalent to a rotation of the robot in the work space. For an algorithm on how to compute the $\mathcal{C}_{\text{obs}}$ for convex polygons, one may be referred to Lozano-Pérez (1990).

The Obstacle Space of a Triangular Robot Determined by zhe Minkowski Sum



Source: Florian Simroth (2023).

Within the configuration space, the *obstacle space* refers to the occupied configurations while the *free space* is composed of configurations that are eligible for suitable paths.

Self-Check Questions

15. Please mark the correct statement(s).

☐ For online planning, a global map environment needs to be known beforehand.

☐ Dynamic environments refer to the process of measuring the environment while the robot is moving.

☐ Nonholonomic planning aims at path planning with moving objects.

☐ *Online planning often relies on the robot's sensor measurements.*

16. Please complete the following sentence.

Within the configuration space, the *obstacle space* refers to the occupied configurations while the *free space* is composed of configurations that are eligible for suitable paths.

6.2 Path Planning

There exist a great variety of path planning algorithms, of which the most common concepts are presented in this section. To better understand the advantages and disadvantages of certain algorithms, some properties are introduced in accordance to Lynch (2017, p. 356):

- A path planning algorithm is considered complete, if it is capable to find a path if one exists or to detect that there is no path between the queried configurations.
    - Resolution completeness is a weakening statement, that implies, that the algorithm finds a path, if one exists at the resolution with which the space is discretized
    - Probabilistic completeness refers to sampling based algorithms which only model the configuration space by discrete samples. If for an infinite number of samples or infinite run time, the algorithm is guaranteed to find an existing path, it is referred to as probabilistically complete.
- If a path planning algorithm starts from scratch for each request, it is a single-query algorithm. On the other hand, if the algorithm builds up some sort of data structure, that can be reused for multiple path search requests, it is referred to as multi-query.
- The term anytime planner refers to algorithms which output the first feasible result and thereafter improves this solution with longer run time. At “any time” the currently found best solution will be provided.
- Different algorithms can be compared inn terms of their computational complexity which indicates how much time or memory an algorithm requires and how this changes when up scaling the search problem, for instance to a broader search space.

Other properties that refer to the outputted path, are the clearance to obstacles, the length of the path or shortest travel time, its feasibility regarding the motion constraints of the robot and its smoothness (Klančar et al., 2017, p. 163). For simplification, the problem of nonholonomic motion planning is neglected for now, and for the illustrations, a point shaped omnidirectional robot moving in the plane is assumed. This way, the configuration space can easily be associated with the workspace and can be illustrated in the plane. Once the configuration space has been modeled, it needs to be represented in some way usable for the path planning algorithms. The algorithms use quite different approaches to accomplish this, which will get clearer after the following sections.

Graph Search

A quite abstract representation for the topology of the configuration space is by means of a graph. A graph consists of so-called *vertices* $V = (v_1, v_2, \dots)$ which are connected by *edges* $E = (e_1, e_2, \dots)$, whereby an edge $e_k = (v_i, v_j)$ is just an unordered pair of vertices. Edges can be assigned a particular *weight*, indicating for example some sort of costs travelling between the connected vertices. A path within the graph is then composed of an alternating sequence of vertices and edges starting from a start vertex and terminating at an end vertex whereby every vertex is only passed once. A path whose start equals its end vertex is called a *cycle*. If paths between any pair of vertices of a graph exist, it is *connected*. A connected graph for which there exist exactly one path between any pair of vertices, or similarly is free of cycles, is a *tree*. The vertices of an unconnected graph can be partitioned into sets of connected vertices called *components*, whereby any pair of vertices of different components is unconnected.

Depending on the used methodology, vertices may have different meanings. As such, the vertices may indicated specific configurations and edges represent the motion a robot has to conduct to move from one configuration to another, or they may refer to whole (sub)areas of the free space whereby the edges indicate

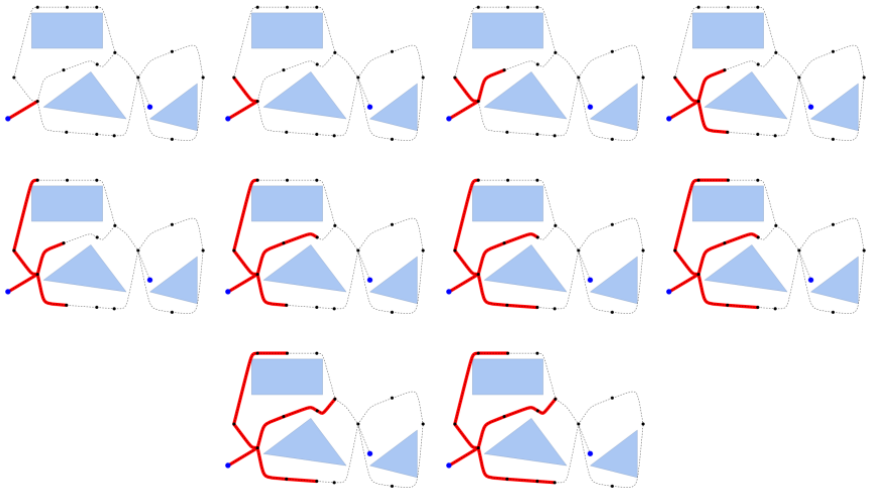
neighbored subareas in between which a robot can transition without collision. In either way, a graph can be some sort of abstraction which allows the application of well-studied graph search algorithms from the field of discrete mathematics. Three fundamental approaches are presented hereafter. For illustration purposes, a graph representation of the previously introduced environment is used whose derivation is not of interest at this point and which will be covered in one of the following sections.

Breadth First Search

The vertices and edges of the graph visible in FIGURE are indicated by dots and dashed lines respectively. The left blue dot indicates the start vertex and the right blue dot corresponds to the end vertex to which a path shall be found. Edges already visited by the algorithm are highlighted in red.

Beginning at the start vertex, a breadth first search will first register all directly connected vertices, or in other words, it will "expand" the start vertex. These vertices are then visited and checked for being the sought target vertex. If the target vertex was not found, the first visited vertex is expanded and all connected vertices are visited. Afterwards, the procedure is repeated for the second vertex and repeated likewise for all remaining vertices that are one edge away from the start vertex. FIGURE shows the first ten steps of the breadth first search. Once the target vertex has been discovered, the path can be recovered by tracking all its predecessors up to the start vertex. The resulting path is also the shortest path in terms of the number of edges.

Sequence of The First Ten Steps of a Breadth First Search

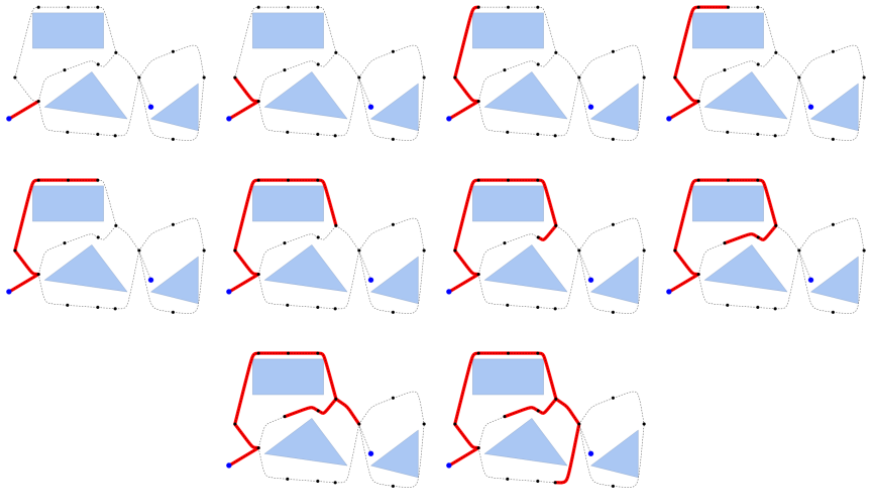


Source: Florian Simroth (2023).

Depth First Search

Starting out from the start vertex, a depth first search will visit one of the directly connected vertices first and then continue to the next directly connected vertex until either no further (unvisited) connected vertex is found or the target vertex is detected. In the former case, the procedure is recursively repeated for any unvisited vertices connected to the previous vertex. A sequence of visited vertices is shown in FIGURE. If the target vertex was found, the path is reconstructed as the list of predecessors. The found path is not necessarily also the shortest path.

Sequence of the First Ten Steps of a Depth First Search



Source: Florian Simroth (2023).

A^*-Search

In contrast to the previous search algorithms, the A^* search uses an additional heuristic such as the distance towards the target vertex and therefore belongs to a class called informed searches. The outputted path is the minimum cost path, where the costs are defined as the sum of edge weights within the path. For illustration purposes, it is assumed, that every vertex corresponds to a position within the planar configuration space, and the edge weights correspond to the distance, the robot needs to travel between the connected vertices. The A^* algorithm aims at visiting the most promising vertices first for finding a path to the target vertex. For this reason, for any newly discovered vertex, its Euclidean distance towards the target vertex is calculated, as an estimate for how close the vertex is to the target. This distance is added to the known path distance from the start vertex, determined by summing up the edge weights from that path. The resulting value represents an estimate for the path from the start vertex to the target vertex via the newly discovered vertex. At each iteration, the A^* algorithm picks from all discovered vertices the one with the lowest total path estimate until the target vertex is found. An exact description of the algorithm with its updating and tracking routines and data structures can be found for example in Lynch (2017, p. 365). Despite its simplicity, the A^* algorithm is quite

powerful and lead to many further developments. The algorithm is complete and the outputted path is guaranteed to be optimal, here in the sense of length.

Road Maps

A general road map points out how different cities are connected by roads, passable by a vehicle. In the context of path planning, the cities in conventional road maps represent certain sites in the configuration space and the roads are the traversable connections in between. Once such a map has been established, it can be used for multiple queries. Two methods to construct a road map are presented shortly, on is based on the visibility graph and the other one makes use of a so-called Voronoi diagram.

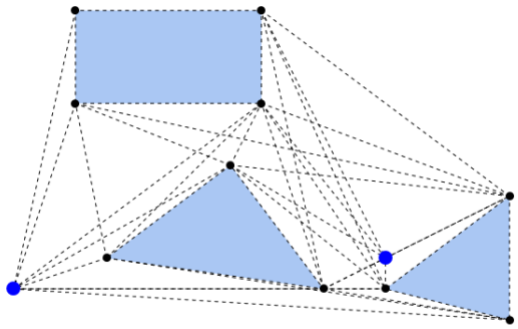
Visibility Graph

A visibility graph as shown in FIGURE is primarily related to a polygonal configuration space. Two vertices of the visibility graph are connected by an edge, if there exists a direct line of sight, i.e, a straight connection within the configuration space that is not blocked by any obstacle. The graph vertices are composed of the vertices of the polygons here marked in black as well as an additional start and an end vertex marked in blue which are likewise connected to all other vertices with a direct line of sight. A line sweep algorithm as shown in Choset et al. (2005, p. 114) can easily construct the visibility graph for polygonal environments. Afterwards, a standard graph search algorithm can be applied to find a path, which also may be augmented by distance measures to the end vertex for an informed search. With some adaptions, it is also possible to reduce the number of edges and also account for objects with curved shapes (Liu and Arimoto, 1992).

Even though, the shortest path can be found with this representation, due to the fact that edges of the polygons are also part of the graph and paths, there is no clearance between the robot and obstacles. This might not be in issue for example for path

finding in computer games, but can pose issues for real world problems due to the uncertainty in robot motion. Besides forcing additional clearance through inflation of the obstacle space, the next approach based on Voronoi diagrams ensures maximum clearance from obstacles.

Visibility Graph



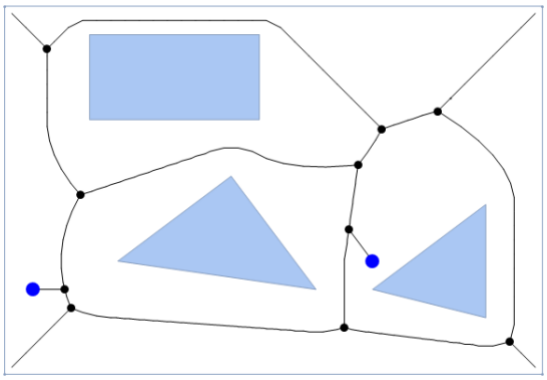
Source: Florian Simroth (2023).

Generalized Voronoi Diagram

While for the visibility road map, the graph edges where close to the obstacles, the roadmap resulting from the generalized Voronoi diagram keeps maximal clearance with respect to the obstacles (see FIGURE). Each obstacle forms a cell whereby the cell is successively expanded in the direction normal to the obstacle boundary. If this is conducted for all cells simultaneously, the set of points, where the cell boundaries meet, form the edges of the road map. The start and end configurations are connected to the road map following a straight line originating from the start/end point in a direction normal to the boundary of the closest obstacle. Once this line intersects with some segment of the Voronoi diagram, the cell boundaries and connection lines can be interpreted as a graph, open for general graph search algorithms. Again, by adding additional information such as distance measure to the end node, informed graph search algorithms can be used.

Due to the preservation of maximum clearance, this method is for instance useful for narrow indoor applications and uncertainty in robot motion, to avoid collision. On the other hand the resulting paths are not the shortest. Though, this procedure will always detect a path, if it exists and will recognize, if there is none. Once constructed, the road map can be used for multiple queries. A detailed description can be found in Choset et al. (2005, p. 123) or Siciliano et al. (2009, p. 532).

Generalized Voronoi Diagram

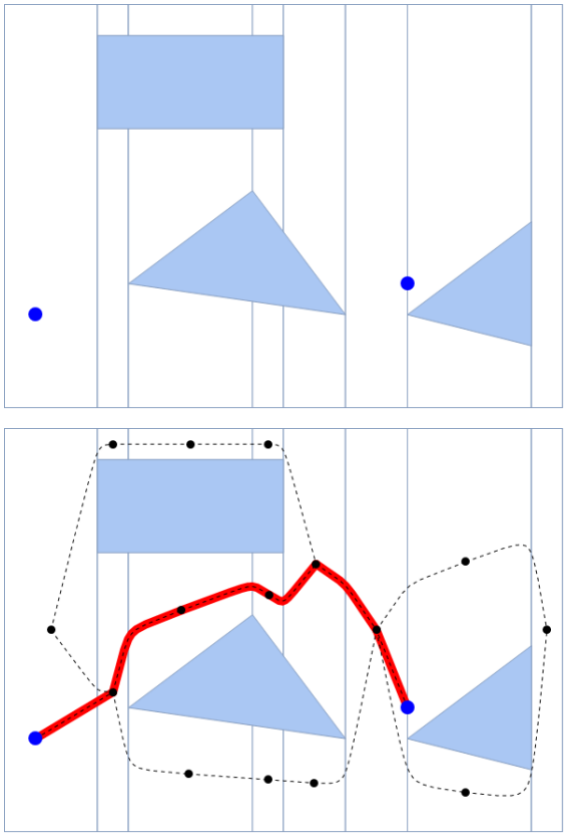


Source: Florian Simroth (2023).

Cell Decomposition

The idea of cell decomposition lies in the discretization of the configuration space into distinct cells which are either occupied by obstacles or allow collision free motion. The approaches for deriving the cells can either generate an *exact* representation of the configuration space or an *approximate* decomposition. In the latter case, there also exist cells which are only partially filled with an obstacle but still will be labeled as occupied.

Exact Cell Decomposition With Highlighted Path Connecting Start and End Point



Source: Florian Simroth (2023).

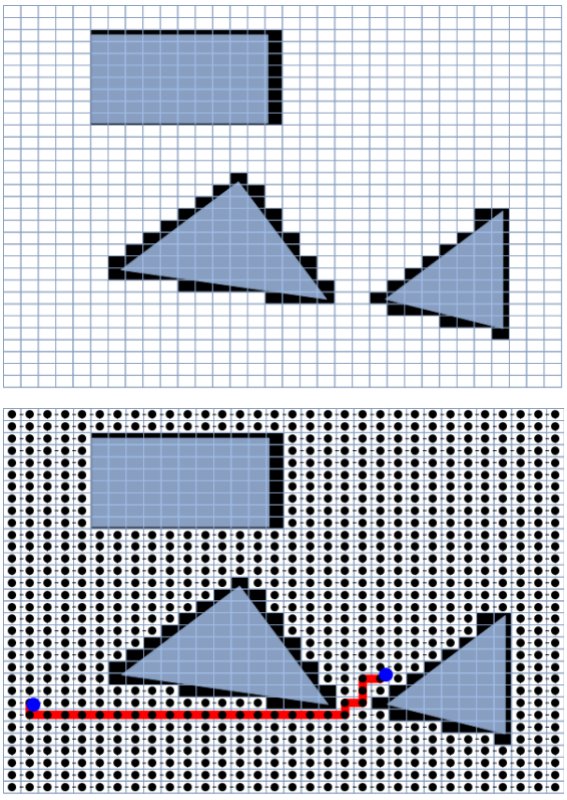
Exact Cell Decomposition

An example for exact cell decomposition is shown in FIGURE. The decomposition can be achieved by a simple line sweep algorithm that draws a vertical line originating at each obstacle vertex, expanding from there until intercepted by another obstacle boundary. The regions bounded by the vertical line segments and the obstacle boundaries form cells that are either fully occupied or free of obstacles. Within each free cell, collision-free robot motion is possible. Furthermore, a collision free path to an obstacle free neighboring cell can be found, passing through the shared common cell boundary. By representing each free cell with a graph vertex, a topological map of the free space can be established. Two vertices, whose corresponding free cells are direct neighbors, will be connected by an edge, thus forming a road map

indicating the connected free space. The start and end configuration points can be linked to the free cell they are contained in.

As the configuration space is exactly represented, the algorithm is complete in the sense that it is guaranteed to find a path, if one exists. Once established, the road map can be used for multiple queries - provided that the map is static and does not change between the queries.

Approximate Cell Decomposition With Equally Sized Cells (Left) and Path From Start To End (Right)



Source: Florian Simroth (2023).

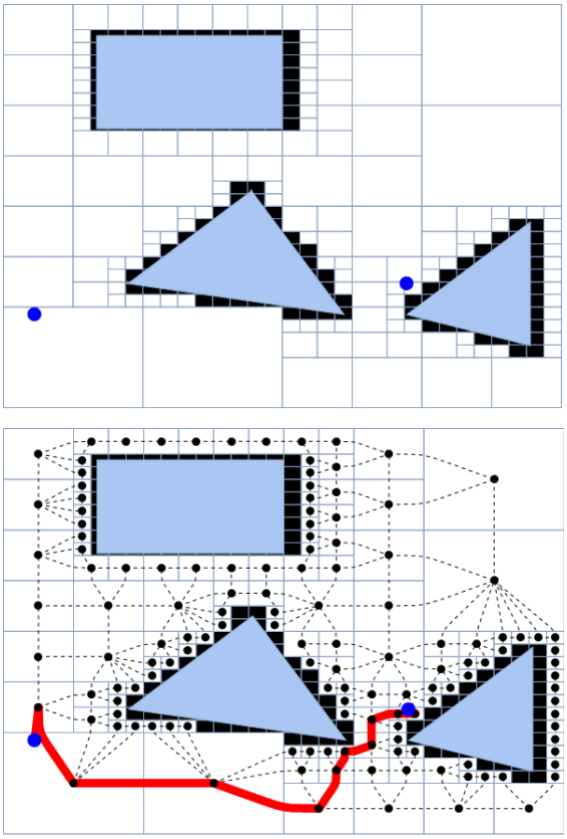
Approximate Cell Decomposition

For the approximate cell decomposition, there exist a variety of possibilities to discretize the configuration space. An example with rectangular cell shapes is shown

in FIGURE (left). Here, equally sized cells are used to rasterize the configuration space into a grid. Depending on the task, the cells can be defined to be four-connected, i.e., horizontal and vertical transition into the neighboring cells is allowed, or eight-connected which would also allow diagonal movement. Similar to the procedure of exact cell decomposition, the cells and connections can be understood as a graph which can be sought for a feasible path as shown in FIGURE (right). As the configuration space is only approximated, an existing path may not be found depending on the resolution. In general, completeness is approached by this method with increasing resolution, which though is only possible within certain limits.

Obviously, the number of cells can get large if a finer resolution is desired. This impacts not only memory consumption, but also computation times for the actual path finding algorithms. One possibility to improve these aspects is shown in FIGURE (left). An exemplary algorithm would split the workspace into four large equally sized cells and if a cell is (partially) occupied, the cell is repeatedly subdivided until either all cells are either fully occupied or free, or a maximum recursion limit is reached. This will iteratively increase the resolution at the borders of obstacles, while large areas of free space are kept undivided, thus decreasing the number of total cells. The path of the resulting graph is shown in FIGURE (right), whereby the varying resolution may induce deviations from the shortest path. Again, with increasing recursion limit and finer resolution, completeness is approached.

Approximate Hierarchical Cell Decomposition With Adaptive Cell Size (Left) and Path From Start To End (Right)



Source: Florian Simroth (2023).

Potential Field

The potential field method builds upon two overlaid force fields: 1) an attraction field, which pulls the robot towards its destination and 2) a repulsion field, which repels the robot from obstacles. Within the attraction field $U_{\text{Attraction}}$, each robot configuration $\mathbf{q}$ is assigned a value which increases the further point $\mathbf{q}$ is away from the target configuration $\mathbf{q}_{\text{Target}}$. The shape of the resulting attraction field may for example be linear or parabolic, depending on the chosen function. A parabolic attraction field can for instance be modeled as

$$U_{\text{Attraction}}(\mathbf{q}) = \frac{1}{2} k \left(\mathbf{q}_{\text{Target}} - \mathbf{q}\right)\left(\mathbf{q}_{\text{Target}} - \mathbf{q}\right)^{T}$$

whereby the positive constant k scales the field. The repulsion field shall avoid collision of the robot with obstacles and therefore should have non-zero values at

occupied configuration and quickly decay at the obstacle boundaries. This behavior can be expressed as

$$U_{\text{Repulsion}}(\mathbf{q}) = \begin{cases} \frac{1}{2}k\sum_i \left(\frac{1}{d(\mathbf{q},\mathcal{B}_i)} - \frac{1}{d_0}\right)^2 & \text{for } d(\mathbf{q},\mathcal{B}_i) \leq d_0 \\ 0 & \text{for} d(\mathbf{q},\mathcal{B}_i) > d_0 \end{cases}$$

where k is a positive scaling constant, $d(\mathbf{q},\mathcal{B}_i)$ refers to the distance of the configuration $\mathbf{q}$ to the ith obstacle $\mathcal{B}_i$, and d_0 represents the furthest distance, an obstacle influences the robot's path. The repulsion forces are added for obstacles.

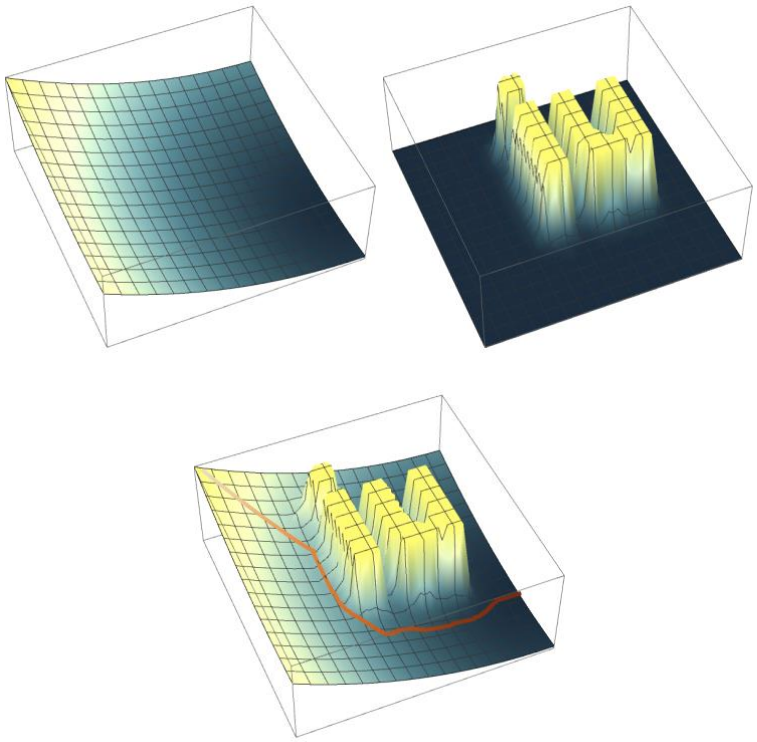
In a last step, both fields are simply added:

$$U = U_{\text{Attraction}}(\mathbf{q}) + U_{\text{Repulsion}}(\mathbf{q})$$

In FIGURE, the result of the attraction field (left), the repulsion field (center), and the resulting potential field (right) is illustrated. The robot path then simply follows the direction of the steepest descent which is calculated by the gradient of the potential field.

The gradient field can be derived in a fast manner, such that this method is often used online for local planning rather then for global path planning (Lynch, 2017, p.386). A particular issue with potential fields are local minima in which the robot can get stuck. Therefore, the potential field method can not be considered to be complete. Also, the found paths are not necessarily the shortest ones as the robot is deflected from the obstacles. The problem with local minima can be tackled by a best-first algorithm which gradually "fills" a local minimum until the robot can move on or by implementing random walks that lead the robot out of a local minimum (Siciliano et al. 2009, p.552).

Potential Field Method

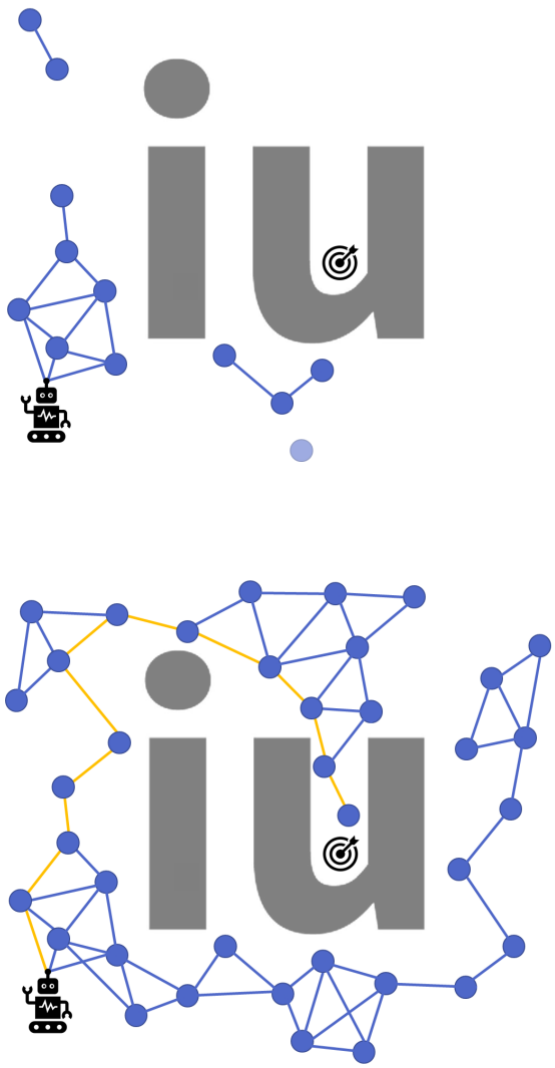


Source: Florian Simroth (2023).

Sampling Based

With increasing dimension of the configuration space, the derivation of the free and obstacle space gets more and more cumbersome, such that previous approaches are often infeasible to be used. In that case, probabilistic approaches often deliver better results. Instead of modeling the complete configuration space, distinct samples are taken for which it is easy to determine whether they are within the free space or imply a collision with an obstacle. With increasing sample number it is then possible to gain an adequate representation of the free space while samples within the obstacle are dropped.

Probabilistic Road Map (PRM) at Intermediate State (Left) and Final State With Highlighted Path (Right)



Source: Florian Simroth (2023).

The Probabilistic Roadmap (PRM) generates a road map of the configuration space by successively sampling random configurations within the configuration space, keeping only collision free samples. For each additional sample configuration, its local neighborhood is checked for already existing samples. For each sample within this neighborhood, a connection will be added to the roadmap, if a collision free (local) path between both configurations exist. The local path can be generated by a local planner, while the collision check can be conducted by checking a sufficient number

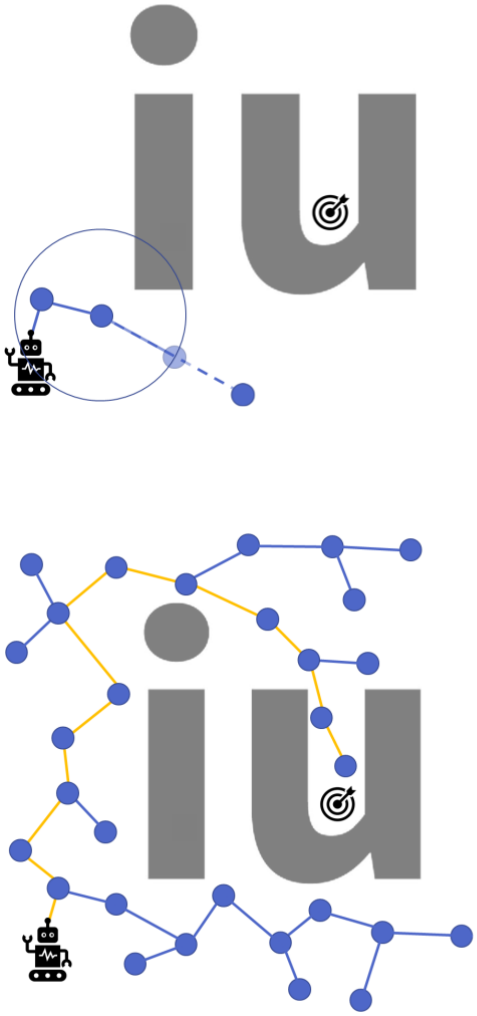
of intermediate configurations of the path for collision. In this manner, a road map of the free space is gained as shown in FIGURE. While initially, many samples remain unconnected and form separate *components* of the graph, with increasing sample number, the individual components will grow and eventually consolidate if there exists a path in between. The algorithm is stopped, if, for instance, a maximum number of points or repetitions has been performed.

The resulting graph now forms a roadmap of the free space which can be used for multiple search queries. For each search query between a start and an end configuration, both configurations are connected to the closest sampled configuration (presuming that those samples belong to the same component) by a local collision-free path. Afterwards, the roadmap can be searched for a feasible path between the start and end configuration by standard graph search algorithms. Usually, the resulting path requires some smoothing, as it was constructed of independent path elements. With the sample number approaching infinity, the algorithm can be considered to be probabilistically complete. While in terms of run time, this approach is superior to the previous methods, it shall be noted that narrow passages form a challenge as these require high sample density in order to be detected appropriately.

Unlike the PRM method, the Rapidly-Exploring Random Tree (RRT) approach builds a new graph for each query. In particular, a graph is "grown" starting from the start configuration by iteratively adding new configurations, whereby a tree-like graph is build. Similar to the PRM method, random samples taken within the configuration space, but these are not added to the emerging graph directly. Instead, the random configuration is "shifted" towards the closest vertex of the already established graph, until it is within a certain threshold distance as shown in FIGURE (left). Afterwards this "shifted" configuration is tested to be collision-free, and is connected to the closest configuration using a local planner again. If the "shifted" configuration and the generated path are free of collision, they are added to the graph. Once a new

vertex is found in proximity to the end configuration, the algorithm is finished and a path is immediately found by tracking the last node's predecessors back to the start configuration (FIGURE (right)). This process can be even further accelerated by a bi-directional approach, where an additional tree is expanded from the end configuration simultaneously, until both tree graphs are in proximity and can be merged.

Rapidly-Exploring Random Tree (RRT) at intermediate state (left) and final state with highlighted path (right)



Source: Florian Simroth (2023).

Both, the PRM and RRT are probabilistically complete. The RRT is even faster than the PRM approach, as only a subset of the configuration space is sampled. On the other hand, the roadmap produced by the PRM can be used for multiple queries, whereas a new graph needs to be build for every query when using the RRT method. Another major advantage of both resampling techniques with respect to the previously introduced approaches is, that the obstacles do not have to modeled in a potentially high dimensional configuration space.

Self-Check Questions

19. Mark the correct statements...

☐ Approximate cell decomposition is probabilistically complete

☐ Exact cell decomposition finds the shortest path

☐ RRT is a multi-query approach

☐ *Potential field method is suitable for online planning*

20. Which approach is most suitable for higher dimensional path planning?

☐ *PRM*

☐ Approximate cell decomposition

☐ Visibility graph

☐ Potential field

6.4 Motion Planning

To this point, only geometric path planning was addressed, and both, the timing component as well as nonholonomic constraints were neglected. Trajectory planning closes this gap, by planning feasible paths under consideration of the kinematic constraints and then applying a time scale. How this can be applied to the general motion planning problem for a wheeled mobile robot in an environment with obstacles is discussed using a unicycle as an example.

For the previously presented path planing methods, such as the RRT, the points in the configuration space where connected by straight lines, neglecting eventual kinematic constraints. In the following, two things shall be accomplished: 1) Provide a time scaling function for a geometric path and 2) derive a geometric path between two configurations, that fulfills the kinematic constraints.

Time scaling of a geometric path

To establish a relation between timing and a geometric path, let a geometric path be given by the continuous sequence of configurations $\mathbf{q}(s)$ with the scaled parameter s indicating a configuration on the path between path start indicated by subscript "s" and the path end "e". As a scaling factor, one possible choice is, for instance, the arc

length ℓ such that $s \in [0, \ell]$. Further, it is assumed that this geometric path shall be traversed by a robot within the time interval $t \in [t_s, t_e]$. One can now introduce an arbitrary time scaling function $s = s(t)$ that defines, at which time instance t the robot shall reach the configuration $\mathbf{q}(s(t))$. As an example, if s is normalized to $s \in [0,1]$ and the time interval is set to $t \in [0, t_{end}]$, the time scaling function $s(t) = t/t_e$ would result in a robot traversing the path within a time t_{end}. Other functions could be designed, such that the robot has 0 velocity at the beginning and end of the path and accelerates in between. The time derivative $\dot{\mathbf{q}}$ can also be expressed in terms of s such that (Siciliano, 2009, p. 489)

$$\dot{\mathbf{q}} = \frac{d\mathbf{q}}{dt} = \frac{d\mathbf{q}}{ds}\dot{s} = \mathbf{q}'\dot{s}$$

and the respective kinematic model yields

$$\dot{\mathbf{q}} = \mathbf{G}(\mathbf{q})\mathbf{u} \quad \Rightarrow \quad \mathbf{q}' = \mathbf{G}(\mathbf{q})\tilde{\mathbf{u}}(s)$$

where the time dependent and path dependent control input are related by

$$\begin{matrix} \mathbf{u}(t) & = & \tilde{\mathbf{u}}(s)\dot{s} \\ \begin{pmatrix} v(t) \\ \omega(t) \end{pmatrix} & & \begin{pmatrix} \tilde{v}(s)\dot{s}(t) \\ \tilde{\omega}(s)\dot{s}(t) \end{pmatrix} \end{matrix}$$

From the kinematic model of the unicycle follows the relationship between the velocities

$$\begin{pmatrix} x' \\ y' \\ \theta' \end{pmatrix} = \begin{pmatrix} \cos(\theta) & 0 \\ \sin(\theta) & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \tilde{v} \\ \tilde{\omega} \end{pmatrix} \quad \Rightarrow \quad \begin{matrix} x' & = \cos(\theta)\tilde{v} \\ y' & = \sin(\theta)\tilde{v} \\ \theta' & = \tilde{\omega} \end{matrix}$$

With these relations, it is possible to determine the time dependent aspects such as $\dot{\mathbf{q}}(t)$ and $\mathbf{u}(t)$ from a geometric path $\mathbf{q}(s)$ using a time scaling function $s(t)$.

Constraint consistent geometric path

Next, a geometric path consistent with the kinematic constraints between a start configurations $\mathbf{q}_\mathrm{s}$ and an end configuration $\mathbf{q}_\mathrm{e}$ of a unicycle robot with the generalized coordinates $\mathbf{q} = (x, y, \theta)$ shall be derived. One of many approaches to accomplish this, is the use of cubic polynomials. Based on the derivation of Siciliano (2009, p. 492), a planar path with coordinates $x(s)$ and $y(s)$ between $\mathbf{q}_\mathrm{s}$ and $\mathbf{q}_\mathrm{e}$ is expressed as a cubic interpolation polynomial

$$\begin{aligned} x(s) &= s^3 x_\mathrm{e} - (s-1)^3 x_\mathrm{s} + a_x s^2 (s-1) + b_x s (s-1)^2 \\ y(s) &= s^3 y_\mathrm{e} - (s-1)^3 y_\mathrm{s} + a_y s^2 (s-1) + b_y s (s-1)^2 \end{aligned}$$

with the derivative functions

$$\begin{aligned} x'(s) &= \frac{dx}{ds} = 3s^2 x_\mathrm{e} - 3(s-1)^2 x_\mathrm{s} + 3(a_x + b_x)s^2 - 2(a_x + 2b_x)s + b_x \\ y'(s) &= \frac{dy}{ds} = 3s^2 y_\mathrm{e} - 3(s-1)^2 y_\mathrm{s} + 3\big(a_y + b_y\big)s^2 - 2\big(a_y + 2b_y\big)s + b_y \end{aligned}$$

where s is the normalized arc length from 0 to 1 such that $x(0) = x_\mathrm{s}$, $x(s) = x_\mathrm{e}$, and analogously for y. In a next step, the parameters a_x, a_y, b_x, and b_y have to be defined such that the boundary constraints are fulfilled. Clearly, the path has to start in the direction of θ_s and finish in the direction of θ_e which can be accomplished by imposing constraints on the derivatives x' and y' at the start and end point:

$$\begin{aligned} x'(0) &= -3x_\mathrm{s} + b_x = k_\mathrm{s}\cos(\theta_\mathrm{s}) &\Leftrightarrow\quad b_x &= 3x_\mathrm{s} + k_\mathrm{s}\cos(\theta_\mathrm{s}) \\ y'(0) &= -3y_\mathrm{s} + b_y = k_\mathrm{s}\sin(\theta_\mathrm{s}) &\Leftrightarrow\quad b_y &= 3y_\mathrm{s} + k_\mathrm{s}\sin(\theta_\mathrm{s}) \\ x'(1) &= 3x_\mathrm{s} + a_x = k_\mathrm{e}\cos(\theta_\mathrm{e}) &\Leftrightarrow\quad a_x &= -3x_\mathrm{e} + k_\mathrm{e}\cos(\theta_\mathrm{e}) \\ y'(1) &= 3y_\mathrm{s} + a_y = k_\mathrm{e}\sin(\theta_\mathrm{e}) &\Leftrightarrow\quad a_y &= -3y_\mathrm{e} + k_\mathrm{e}\sin(\theta_\mathrm{e}) \end{aligned}$$

Here, k_s and k_e are scaling factors and can be thought of as the start and end velocity which can be freely chosen. With the path now being defined, the full configuration of the robot from start to end can be described as

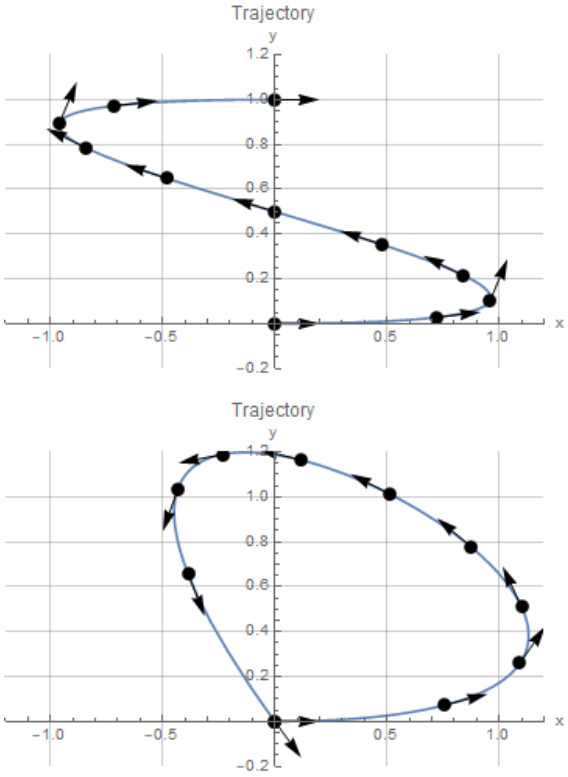
$$\mathbf{q}(s) = \begin{pmatrix} x(s) \\ y(s) \\ \underbrace{\arctan2\big(y'(s), x'(s)\big)}_{\breve{\theta}(s)} \end{pmatrix} \quad \text{and} \quad \mathbf{q}'(s) = \begin{pmatrix} x'(s) \\ y'(s) \\ \dfrac{y''(s)x'(s) - x''(s)y'(s)}{\big(x'(s)\big)^2 + \big(y'(s)\big)^2} \end{pmatrix}$$

Next, the required inputs $\mathbf{u}(s) = \big(v(s), \omega(s)\big)^T$ are determined to drive the robot on the path, automatically ensuring that all constraints are fulfilled. $\tilde{\mathbf{u}}$ can either be directly derived from [eq_velocities], or more formally by solving the kinematic model for $\mathbf{u}$ using the pseudo inverse:

$$\underbrace{\begin{pmatrix} \tilde{v} \\ \tilde{\omega} \end{pmatrix}}_{\breve{\tilde{\mathbf{u}}}} = \underbrace{\begin{pmatrix} \cos(\theta) & \sin(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{\breve{\mathbf{G}}^{\dagger}} \underbrace{\begin{pmatrix} x' \\ y' \\ \theta' \end{pmatrix}}_{\breve{\mathbf{q}}(s)} = \begin{pmatrix} x'\cos(\theta) + y'\cos(\theta) \\ \tilde{\omega} \end{pmatrix} = \begin{pmatrix} \sqrt{x'^2 + y'^2} \\ \dfrac{y''x' - x''y'}{(x'^2 + y'^2)^2} \end{pmatrix}$$

In a last step, a time scaling could be applied, under consideration of the maximum permissible control inputs v_{max} and ω_{max}. FIGURE shows the plots for two different end configurations. Note that by the usage of cubic splines, the trajectory neither incorporates cusps, i.e., reversing of the robot, nor in-place rotation. To achieve these kinds of motion, a chained form of the kinematics can be used as described by Siciliano (2009, p. 494).

Trajectory For Purely Translational Displacement of a Robot (Left), and Trajectory For Pure Rotation With Cubic Splines That Do Not Allow Motion Reversal Or In-Place Rotation (Right)



Source: Florian Simroth (2023).

Nonholonomic Motion Planning With Obstacles

At last, an example shall be given, of how motion planning with obstacles and nonholonomic constraints can be accomplished. There exist different strategies, to more or less directly integrate nonholonomic constraints into the path planning algorithms presented in the previous section. Here, the Rapid-Exploring Random Tree (RRT) approach is revisited, which was previously introduced with the following general steps:

- Randomly sample a configuration point in free configuration space
- Find the nearest existing configuration point of the tree graph
- Apply a local planner to determine a feasible path starting at the nearest existing configuration to a new configuration in direction of the random sample, but within a limited radius of the existing configuration
- Check new path for collisions and add to tree if collision free

Within the local planning step, for which previously only straight lines were used, now different possibilities exist, to incorporate the kinematic constraints. Lynch (2017, p. 382) lists two different approaches which are possible for robots with nonholonomic constraints:

- Discretization of controls by using for example maximum speed straight and maximum left and right curved motions. These discretized inputs are then integrated for a time duration Δt starting at the nearest configuration. Of the resulting configuration points, the one that is closest to the random sample is picked.
- By using so called Reeds-Shepp curves, i.e., paths generated by concatenation of minimum turning radius curves and straight line segments, the shortest path to the new configuration point between the nearest existing and the random sample configuration.

Besides these methods, the method using cubic splines as demonstrated before can also be used to connect the nearest configuration to a new configuration. Checking for collisions can be achieved by sampling the generated path and checking each sampled configuration for collisions in the configuration space.

To conclude this section, a few remarks shall be made on different wordings used in conjunction with motion planning. A term that emphasizes planning under kinematic constraints and limits on dynamics is kinodynamic planning, as characterized by Donald et al. (1993). LaValle (2006, p. 792) argues, that the term trajectory planning is nowadays used almost synonymously with kinodynamic planning, but mentions, that its original procedure consisted of first planning a (geometric) path in the free configuration space and afterwards applying a time scaling function as shown in this section. Sometimes, planning also takes place in the phase state space, which additionally takes velocity limits into account. Another interesting concept is the expansion of the configuration space to regions of inevitable collision (LaValle, 2006,

p. 796). That is, if the robot configuration also incorporates velocities, certain regions within the free configuration space may also not be entered: due to the speed and robot dynamics, the robot would inevitably collide with some obstacle in a next step.

Self-Check Questions

21. How can the shortest path of a differential robot be described?
As a sequence of pure rotation and straight motion

22. Which step or module in the RRT takes care of kinematic constraints?
Local Planner

6.5 Task Planning

The primary focus in the previous sections was on how the robot can move from one location to another, avoiding obstacles and respecting the constraints induced by the robot design itself. Task planning is aiming at a higher abstraction level of which one of the outputs are the instructions for the robot to conduct. For a self-driving car, the task can be to overtake another car for example, in service robotics it could be the delivery of packages of medicine to different rooms in a hospital, or in an industrial setting, some pick and place task to move goods within a warehouse. Task planning, in general, resides in the domain of artificial intelligence (AI) for which multi-purpose planners are developed, capable to plan tasks way beyond the context of mobile robotics. In that sense, this section will only introduce some very general concepts.

Similar to motion planning, task planning can be understood as changing a system from a start state to an end state. For task planning though, a state may be described in a far more general way, also being specified by discrete elements, such as a set of colors (red, green, blue). The task could be to change the color from red to green. While for motion planning, a heuristic such as the distance between initial and end configuration is often easily derived, this may be hard for qualitative predicates such as a color.

As a first initiative in this domain, the SHAKEY research project is often cited, from which not only the A^* algorithm emanated, but also a general framework to describe and solve tasks, the *Stanford Institute Problem Solver (STRIPS)* (Fikes and Nilsson, 1971). In this early version though, "the world is regarded as being in a static state and is transformable to another static state only by a single agent performing any of a given set of actions" (Fikes and Nilsson, 1993, p. 228). Yet, it lay a foundation for task planning and introduced ideas which are still being in use nowadays and are reproduced from Tzafestas (2014, p. 470):

- Within the state space, all relevant characteristics are included which contain both, continuous elements such as robot configuration or discrete properties such as colors or a certain ordering of objects. To specify the task, the description of an initial state as well as the end state is required.
- The state may be manipulated by applying an action. Possible actions and their impact on the state must be defined as well as preconditions for when these actions may be applicable.
- The sequence of actions that can be taken to change the system from an initial state to an end state may be described in a plan. The plan is derived by a planner which may be a human or a software program.

These three topics will be further elaborated in the next sections.

6.5.1 World Representation

In order to reuse existing problem solvers, it is desired to provide a uniform language to describe the world in which the problem is to be solved, as well as the actions that are available to reach a solution, and of course the problem statement itself. STRIPS provided such an environment, even though limited to many constraints and assumptions, such as a closed world, in which only a single actor operates and in the absence of uncertainty. A descriptive language, that supports and integrates multiple languages and concepts including STRIPS is the *Planning Domain Definition Language*

(PDDL) by Ghallab et al. (1998), which is widely used with newer published versions. To describe a world domain with a kitchen and an office as rooms, a coffee mug as an object, and a mobile robot which shall be able to transport the coffee from the kitchen to the office, can be described in PDDL as follows:

```
(define (domain Office)
  (:requirements :strips)
  (:types room robot object)
  (:predicates
    (at-robot ?r - robot ?room - room)
    (at-object ?o - object ?room - room)
    (in-room ?r - room)
    (free-hand ?r - robot)
  )
[...ACTIONS...]
)
```

The world name is set to "Office" and the domain requires the STRIPS planning system. There are three types of things in this domain, *robots*, *rooms*, and other *objects*. The predicates take parameters in form of variables noted by an "?" of the predefined types and evaluate either to true or false. The *at-robot*/*at-object* predicate takes a robot/object and a room as arguments and evaluate to true, if the robot/object provided as the argument is within the specified room. The *in-room* predicate takes a room as an argument and specifies the existence of the room in the domain, while the *free-hand* predicate checks, whether the provided robot has a free hand. *Actions* that are available within this domain will be defined in the next section.

With this world domain being specified, now a problem instance “FetchCoffee” can be created where the robot has to fetch a coffee mug from the kitchen and bring it to the office. Here’s an example of a PDDL problem description for this scenario:

```
(define (problem fetchCoffee)
  (:domain Office)
  (:objects
    coffee - object
    robot - robot
    office - room
    kitchen - room)
```

```
  (:init
    (in-room office)
    (in-room kitchen)
    (at-robot robot office)
    (at-object coffee kitchen)
    (free-hand robot)
  )
  (:goal (and (at-object coffee office)))
)
```

The problem is defined in the "Office" domain and has four "things": a robot, a coffee-mug, a kitchen, and an office with according types. The initial state is defined with the following predicates:

- (at-robot *office*): The robot is at the room *office*.
- (at-object *office*): The coffee mug is at the room *kitchen*.
- (in-kitchen *kitchen*): The room *kitchen* exists in the domain.
- (in-office *office*): The room *office* exists in the domain.
- (free-hand *robot*): The robot has nothing in his hand.

The goal state is defined as

- (at-object *office*), which means that the coffee mug should be at the office location.

No further details are defined on whether the coffee-mug shall be placed on a table or where the robot is shall be located in the end. What is still missing are the actions that are available in the domain, which will be covered in the next section.

Action Representation

Actions or operations have the ability to perform changes of the state space and are therefor necessary to achieve a certain goal. An action in STRIPS can be defined by the following three means (Siciliano and Kathib, 2016, p. 338):

- Precondition: Not every action can be performed for every state of the system. For example, an object on a table can only be grasped by a robot, if

there actually is an obstacle on the table. Thus, the existence of an object at that place forms a precondition for the grasping action.

- Add list: The set of (state) properties, that become true when the action has been performed
- Delete list: Some properties become false once the action has been conducted - these are defined in the delete list.

In PDDL, the add and delete list are combined in the *effects* of an action. For the Office example, three actions *move-to*, *pick-up*, and *put-down* are defined:

```
(:action move-to
    :parameters (?r - robot ?from - room ?to - room)
    :precondition (and (at-robot ?r ?from) (in-room ?to))
    :effect (and (not (at-robot ?r ?from)) (at-robot ?r ?to))
  )
  (:action pick-up
    :parameters (?r - robot ?o - object ?room - room)
    :precondition (and (at-robot ?r ?room) (at-object ?o ?room)
(free-hand ?r))
    :effect (and (not (at-object ?o ?room)) (not (free-hand ?r))
(at-robot ?r ?room))
  )
  (:action put-down
    :parameters (?r - robot ?o - object ?room - room)
    :precondition (and (at-robot ?r ?room) (not (free-hand ?r))
(not (at-object ?o ?room)))
    :effect (and (at-object ?o ?room) (free-hand ?r) (at-robot ?r
?room))
  )
```

The *move-to* action takes a robot, a source room, and a destination room as parameters, and moves the robot from the source room to the destination room if the preconditions are met, i.e, whether the robot is in the *from* room and if the *to* room exists in the domain. The *effect* is that the robot is not any more in the *from* room but in the *to* room. The *pick-up* action (and correspondingly the *put-down* action) receives a robot, an object, and a room as arguments, and has the robot pick up the object if both, the robot and the object are in the (same) room, and the robot's hand is free. The effect is that the object is not any more in the room, the robot's hand is not free any more, but the robot remains in the room.

Plan Representation

There exist multiple ways to represent a plan in task planning. Two commonly used examples are State-Transition Plans and Policies, discussed hereafter.

A quite intuitive way to represent a plan is by a sequence of actions to be conducted to change a system from an initial to a goal state. These State-Transition Plans assume, that the execution itself, such as the time taken to conduct the action, does not impact the outcome and therefore can be ignored. Additionally, the success of the action is presumed, as otherwise the preconditions of the next action might not be fulfilled. As such, state-transition plans often need to be complemented by additional layers for feedback control and a reactive plan (Siciliano and Kathib, 2016, p. 338). A planner for the "fetchCoffee" problem might come up with the following plan for the sequence of actions:

```
Step 1: move(robot, office, kitchen)
Step 2: pick-up(robot, coffee, kitchen)
Step 3: move(robot, kitchen, office)
Step 4: put-down(robot, coffee, office)
```

Alternatively, instead of lining up tasks, one can define specific actions to be conducted, once a certain robot and environment state is experienced. Different states are then associated with different actions, such that the robot behaviour underlies certain **policies**. In the planning process, it is determined, which actions in which states most likely bring the system closer to task fulfillment whereby the derivation of these policies is often accomplished by Markov decision processes. The main aim of policies is towards robustness in uncertain environments, while they are rather restricted to short term tasks (Siciliano and Kathib, 2016, p. 338). Note that this is a different approach than the one described by the "fetchCoffee" example.

Policies

Association between certain states and actions that, if conducted by the robot, will contribute to the fulfillment of a task with a certain probability.

6.5.4 Search Process

Once the domain, tasks, and objectives are defined, the planner may choose from a variety of search algorithms to develop a plan to fulfill the end. The planner may apply a forward search, i.e., start at the initial state, or correspondingly, a backward search when starting the search at the end state. Neglecting uncertainty and external interference, a very simple planning procedure could start at the initial state and then find all actions, whose precondition allow the application to that state. The resulting states are gather and the procedure is repeated for each state. In this manner, a graph can be derived similar to the breadth first search, with the states as nodes and the respective actions forming the edges. The search is stopped once the end state is reached - the path back to the initial state provides the plan of actions to be conducted to reach the end state. This may work, if the number of objects and actions is very small, but will not be feasible for most practical problems due to the large number of possible actions and variations. STRIPS, for example, first determines the differences between the initial and the end state and afterwards focuses only on actions, that affect particularly these states to reduce the difference between the start and the end state (Fikes and Nilsson, 1971, p. 11).

As mentioned earlier, one of the key challenges for planning in state space is to find a heuristic to assist the planning process. As such, the *Heuristic Search Planner* (HSP) by Bonet and Geffner (2001) or *Fast Forward* (FF) by Hoffmann (2001) automatically determine a heuristic, from which further planning algorithms such as the *Fast Downward Planning System* by Helmert (2006) and others evolved and continue to evolve.

Self-Check Questions

23. What are the elements that need to be defined for an action?
Preconditions and the effects (added and deleted) from the world state

24. What is another way to plan tasks besides state-transition plans?
Policies

Summary

Path planning can be considered a subarea of motion planning, mainly focused on finding a geometric path between two configurations, avoiding collisions with obstacles. A trajectory can be derived from a geometric path by time scaling such that velocity and acceleration limits are met.

A fundamental concept in path planning is the configuration space, whose dimension equals the degrees of freedom of the robot. It is the union of the free and obstacle space, which are respectively all collision-free configurations and configurations occupied by obstacles. Planning in the configuration space allows regarding robots of different shapes as a point in the configurations space and therefore simplifies the application of different search algorithms.

When searching for a suitable path, the configuration space is usually discretized using roadmaps, cell decomposition, or stochastically by random samples. Roadmaps represent the connected free space as a graph, suitable for multiple queries. Cell decomposition represent the free space either exactly or by approximation, but are usually limited to a low number of degrees of freedom. Rapid-Exploring Trees are commonly used also for complex problems with many degrees of freedom, but can only reach probabilistic completeness. Nonholonomic constraints can be incorporated either within the local planner or by discretizing the control inputs.

In task planning, commonly, the the world is described by a discrete state space and a set of possible actions which manipulate the state in a defined way. One variant of task planning consists in finding a sequence of actions that will consecutively change an initial state into a desired target state. This is possible by languages such as Planning Domain Definition Language, that can capture a broad variety of problem types and allow to reuse the same planning algorithms. Instead of action plans, task planning can also be accomplished by describing the actions a robot shall perform in

certain states, whereby the planner determines, which actions are most likely to reach a goal from specific states.

Appendix 1 – References

Alexander, R. M. (1984). The gaits of bipedal and quadrupedal animals. *The International Journal of Robotics Research*, *3*(2), 49-59.

Blickhan, R. (1989). The spring-mass model for running and hopping. *Journal of Biomechanics*, *22*(11-12), 1217–1227. https://doi.org/10.1016/0021-9290(89)90224-8

Collins, S. H., Wisse, M., & Ruina, A. (2001). A three-dimensional passive-dynamic walking robot with two legs and knees. The International Journal of Robotics Research, 20(7), 607-615.