Robot Control Basics
Mobile robot kinematics

- Differential drive mobile robot
- Two wheels, with diameter $r$, point $P$ centered
- Between two wheels is the origin of the robot frame
- Each wheel is a distance $l$ from the center
Some terminology

• Effector (legs, arms, wheels, fingers)
• Actuator - enables effector to execute motion (electric, hydraulic)

• Degree of freedom DOF - number of parameters describing the pose/configuration of the robot
• Rigid body 6 DOF, mobile robot 3 DOF

• Simplest case one actuator controls one DOF → all degrees of freedom are controllable

• We have derived kinematics equations of the robot

\[
\begin{align*}
\dot{x} &= v \cos \theta \\
\dot{y} &= v \sin \theta \\
\dot{\theta} &= \omega
\end{align*}
\]
Some terminology

• Alternative derivation (optional) is in terms of wheel constraints section (3.2.3 - 3.4.2)
• Example
  • sliding constraint – each wheel can only roll in the plane of the wheel
  • steering constraint – steerable wheels can be steered
• degree of maneuverability – number of degrees of freedom robot can directly control $\delta_M$
• Car-like mobile robot – 3-DOF, two control inputs
  $$\delta_M = \delta_m + \delta_s$$
• Differential drive robot
  $$\delta_M = 2 + 0 = 2$$
  $$\dot{x} = v \cos \theta$$
  $$\dot{y} = v \sin \theta$$
  $$\dot{\theta} = \omega$$
Some terminology

- Degrees of Freedom (DOF)
- Differential number of Degrees of freedom (DOF in the velocity space) - DDOF
- DDOF is always equal to $\delta_m$ degree of mobility

- Car-like mobile robot - 3-DOF, two control inputs, two differential degrees of freedom

- If DOF = DDOF robot is holonomic, otherwise it is non-holonomic

- Differential drive robot - non-holonomic
- Omnidirectional drive - holonomic
Connection between DOF and actuators/effectors

• If there is an actuator for each DOF then each DOF is controllable.
• If not all DOF are directly controllable the control problems are much harder (see later).

The number of controllable DOF determines how hard the control problem will be.
holonomic robots # of DOF is the same as # of controllable DOF’s
nonholonomic robot # of DOF is bigger then # of controllable DOF’s
redundant robot # of controllable DOF is larger then # of total DOF’s

e.g. Human Arm 6 DOF’s – position and orientation of the Fingertip in 3D space – 7 actuators – 3 shoulder, 1 elbow, 3 wrist
Five Basic Types of Three-Wheel Configurations

Omnidirectional
\[ \delta_M = 3 \]
\[ \delta_m = 3 \]
\[ \delta_s = 0 \]

Differential
\[ \delta_M = 2 \]
\[ \delta_m = 2 \]
\[ \delta_s = 0 \]

Omni-Steer
\[ \delta_M = 3 \]
\[ \delta_m = 2 \]
\[ \delta_s = 1 \]

Tricycle
\[ \delta_M = 2 \]
\[ \delta_m = 1 \]
\[ \delta_s = 1 \]

Two-Steer
\[ \delta_M = 3 \]
\[ \delta_m = 1 \]
\[ \delta_s = 2 \]
Previously

• Kinematics models of kinematic chain, arm and mobile robot
• Relationship between the position of the end-effector and joint angles (manipulator), pose of the mobile robot and angular and linear velocities
• Control and Planning - How to do the right thing?

1. Open loop control
2. Feedback control
3. Potential field based methods (feedback control)
Paths and Trajectories

• In general – control problem – need to generate set of control commands to accomplish the task
• In an open loop setting there are two components

1. Geometric Path Generation
2. Trajectory generation (time indexed path)
3. Trajectory tracking

• Example omni-directional robot – can control all degrees of freedom independently
Trajectories

- Smooth 1D trajectories - scalar functions of time
- Polynomials of higher orders
- Piecewise linear segments and polynomial blends
- Blackboard

- Interpolation of orientation in 3D
- Rigid Body Pose and Motion
- Varying coordinate frames
Example trajectory generation

- Given end points in either work space or joint angle space
- Generate joint angle trajectory between start and end position
- Specify start and end position (or additional constraints – spatial (obstacles) or temporal (time of completion))
- How to generate velocities and accelerations to follow the trajectory
- Example blackboard: use cubic polynomials to generate the trajectory
Trajectory generation

• Alternatives linear paths with parabolic blends (splines) – the via points are not actually reached
• Previous example – paths computed in joint space - the path in the workspace depends on the kinematics of the manipulator
• Another scenario – compute paths in the workspace - specify at each instance of time the pose (R,T) the end effector robot should be at - interpolate between poses
• Problems with workspace and singularities make sure all intermediate points are reachable the joint rates are attainable
Path / Trajectory Considerations: Omnidirectional Drive

Move for 1s with constant speed along X, change orientation Counterclockwise by 90 degrees in 1s, move for 1s with constant Speed along Y – wheels can move and roll – omni-directional drive
Path / Trajectory Considerations: Two-Steer

Move for 1s with constant speed along X, rotate steered wheels by -50/50 degrees; change orientation counterclockwise by 90 degrees in 1s, move for 1s with constant speed along Y
Beyond Basic kinematics

• So far we have considered only trajectories in time and space – no velocities
• When handling more dynamic scenarios velocities become important, we need to design trajectory profiles which can be nicely followed

• Two main control approaches
  Open Loop Control
  Feedback Control
Motion Control: Open Loop Control

- Trajectory (path) divided in motion segments of clearly defined shape:
  - straight lines and segments of a circle.
- Control problem:
  - pre-compute a smooth trajectory based on line and circle segments
- Disadvantages:
  - It is not at all an easy task to pre-compute a feasible trajectory
  - limitations and constraints of the robots velocities and accelerations
  - does not adapt or correct the trajectory if dynamical changes of the environment occur.
  - The resulting trajectories are usually not smooth

© R. Siegwart, I. Nourbakhsh
Motion Control: Open Loop

- Problem: given initial and final configuration of the robot, compute the path as a sequence of predefined motion segments – segments of straight line and circle (Dubins car, Reeds-Shephard car)
Feedback control

- More suitable alternative
- Use state feedback controller
- At each instance of time compute a control law
- Given the current error between current and desired position

\[ \Delta y \]
Kinematic Position Control

The kinematic of a differential drive mobile robot described in the initial frame \( \{ x_I, y_I, \theta \} \) is given by,

\[
\begin{bmatrix}
\dot{x} \\
\dot{y} \\
\dot{\theta}
\end{bmatrix} = \begin{bmatrix}
\cos \theta & 0 \\
\sin \theta & 0 \\
0 & 1
\end{bmatrix} \begin{bmatrix}
v \\
\omega
\end{bmatrix}
\]

relating the linear velocities in the direction of the \( x_I \) and \( y_I \) of the initial frame.

Let \( \alpha \) denote the angle between the \( x_R \) axis of the robots reference frame and the vector connecting the center of the axle of the wheels with the final position.
The objective of a kinematic controller is to follow a trajectory described by its position and/or velocity profiles as function of time.

Motion control is not straightforward because mobile robots are non-holonomic systems.

However, it has been studied by various research groups and some adequate solutions for (kinematic) motion control of a mobile robot system are available.

Most controllers are not considering the dynamics of the system.
Motion Control - Steering to a point

• **Consider our robot**

\[
\begin{align*}
\dot{x} &= v \cos \theta \\
\dot{y} &= v \sin \theta \\
\dot{\theta} &= \omega
\end{align*}
\]

• **To steer the robot to desired position** \([x_g, y_g]\)

\[
v = K_v \sqrt{(x - x_g)^2 + (y - y_g)^2}
\]

\[
\omega = \tan^{-1} \frac{y_g - y}{x_g - x}
\]
Motion Control: Steering to a pose

- Set intermediate positions lying on the requested path.
- Given a goal how to compute the control commands for linear and angular velocities to reach the desired configuration.

Problem statement
- Given arbitrary position and orientation of the robot $[x, y, \theta]$ how to reach desired goal orientation and position $[x_g, y_g, \theta_g]$. 

© R. Siegwart, I. Nourbakhsh
Motion Control: Feedback Control, Problem Statement

- Find a control matrix $K$, if exists

$$K = \begin{bmatrix}
k_{11} & k_{12} & k_{13} \\
k_{21} & k_{22} & k_{23}
\end{bmatrix}$$

- with $k_{ij} = k(t, e)$
- such that the control of $v(t)$ and $\omega(t)$

$$\begin{bmatrix}
v(t) \\
\omega(t)
\end{bmatrix} = K \cdot e = K \cdot \begin{bmatrix}
x \\
y \\
\theta
\end{bmatrix}$$

- drives the error $e$ to zero.

$$\lim_{t \to \infty} e(t) = 0$$
Motion Control:

Kinematic Position Control

- The kinematic of a differential drive mobile robot described in the initial frame \{x_I, y_I, \theta\} is given by,

\[
\begin{bmatrix}
\dot{x} \\
\dot{y} \\
\dot{\theta}
\end{bmatrix} =
\begin{bmatrix}
\cos \theta & 0 \\
\sin \theta & 0 \\
0 & 1
\end{bmatrix}
\begin{bmatrix}
v \\
\omega
\end{bmatrix}
\]

where \(v\) and \(\omega\) are the linear velocities in the direction of the \(x_I\) and \(y_I\) of the initial frame. Let \(\alpha\) denote the angle between the \(x_R\) axis of the robot's reference frame and the vector connecting the center of the axle of the wheels with the final position.
Kinematic Position Control: Coordinates Transformation

Coordinates transformation into polar coordinates with its origin at goal position:

\[ \rho = \sqrt{\Delta x^2 + \Delta y^2} \]

\[ \alpha = -\theta + \tan^{-1}(\Delta y, \Delta x) \]

\[ \beta = -\theta - \alpha \]

System description, in the new polar coordinates

\[
\begin{bmatrix}
\dot{\rho} \\
\dot{\alpha} \\
\dot{\beta}
\end{bmatrix}
= \begin{bmatrix}
-\cos \alpha & 0 \\
\sin \alpha & -1 \\
-\sin \alpha & 0
\end{bmatrix}
\begin{bmatrix}
\nu \\
\omega
\end{bmatrix}
\]

For \( \alpha \) from \( I_1 = \left( -\frac{\pi}{2}, \frac{\pi}{2} \right) \)

\[
\begin{bmatrix}
\dot{\rho} \\
\dot{\alpha} \\
\dot{\beta}
\end{bmatrix}
= \begin{bmatrix}
\frac{1}{\rho} \cos \alpha & 0 \\
-\frac{1}{\rho} \sin \alpha & -1 \\
\sin \alpha & 0
\end{bmatrix}
\begin{bmatrix}
\nu \\
\omega
\end{bmatrix}
\]

for \( I_2 = (-\pi, -\pi/2] \cup (\pi/2, \pi] \)
Kinematic Position Control: Remarks

- The coordinates transformation is not defined at \( x = y = 0 \); as in such a point the determinant of the Jacobian matrix of the transformation is not defined, i.e. it is unbounded.

- For \( \alpha \in I_1 \) the forward direction of the robot points toward the goal, for \( \alpha \in I_2 \) it is the backward direction.

- By properly defining the forward direction of the robot at its initial configuration, it is always possible to have \( \alpha \in I_1 \) at \( t=0 \). However this does not mean that \( \alpha \) remains in \( I_1 \) for all time \( t \).
Kinematic Position Control: The Control Law

• It can be shown, that with

\[ \nu = k_\rho \rho \quad \omega = k_\alpha \alpha + k_\beta \beta \]

the feedback controlled system

\[
\begin{bmatrix}
\dot{\rho} \\
\dot{\alpha} \\
\dot{\beta}
\end{bmatrix} =
\begin{bmatrix}
-k_\rho \rho \cos \alpha \\
k_\rho \sin \alpha - k_\alpha \alpha - k_\beta \beta \\
-k_\rho \sin \alpha
\end{bmatrix}
\]

• will drive the robot to \((\rho, \alpha, \beta) = (0, 0, 0)\)

• The control signal \(\nu\) has always constant sign,
  - the direction of movement is kept positive or negative during movement
  - parking maneuver is performed always in the most natural way and without ever inverting its motion.
• Question: How to select the constant parameters
to achieve that the error will go to zero

• Digression - review
Previously - Eigenvalues and Eigenvectors

For the previous example

\[ \lambda_1 = -1, x_1 = [1, 1]^T \quad \lambda_2 = -2, x_2 = [5, 2]^T \]

We will get special solutions to ODE \( \dot{u} = Au \)

\[ Au = e^{\lambda_1 t} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad u = e^{\lambda_2 t} \begin{bmatrix} 5 \\ 2 \end{bmatrix} \]

Their linear combination is also a solution (due to the linearity of \( \dot{u} = Au \))

\[ u = c_1 e^{\lambda_1 t} \begin{bmatrix} 1 \\ 1 \end{bmatrix} + c_2 e^{\lambda_1 t} \begin{bmatrix} 5 \\ 2 \end{bmatrix} \]

In the context of diff. equations - special meaning
Any solution can be expressed as linear combination
Individual solutions correspond to modes
Eigenvalues of linear system

• Given linear system of differential equations

\[ \dot{x} = Ax \]

• For 2 dimensional system (A is 2 x 2), A has two eigenvalues

• Define \( \Delta = \lambda_1 \lambda_2 \) and \( \tau = \lambda_1 + \lambda_2 \)

• if \( \Delta < 0 \) saddle node

• if \( \Delta > 0 \) we have two cases
  1. \( \tau > 0 \) eigenvalues positive
  2. \( \tau < 0 \) eigenvalues negative: stable nodes of the system
Kinematic Position Control: Resulting Path

![Diagram showing robot trajectory in a 2D space with X and Y axes labeled in millimeters. Two similar diagrams are displayed, each showing a series of connected points indicating the path of a robot.]
• Digression
• Notes of system linearization
Linearization

• But our system is not linear, e.g. cannot be written in the form

\[
\dot{x} = Ax
\]
Some terminology

• We have derived kinematics equations of the robot

\[
\begin{align*}
\dot{x} &= v \cos \theta \\
\dot{y} &= v \sin \theta \\
\dot{\theta} &= \omega
\end{align*}
\]

• Non-linear differential equation \( \dot{x} = f(x,u) \)
• In our case

\[
\begin{align*}
\dot{x} &= f_1(x,y,\theta,v,\omega) \\
\dot{y} &= f_2(x,y,\theta,v,\omega) \\
\dot{\theta} &= f_3(x,y,\theta,v,\omega)
\end{align*}
\]
Jacobian Matrix

• Suppose you have two dim function

\[ f(x) = \begin{bmatrix} f_1(x) \\ f_2(x) \end{bmatrix} \]

• Gradient operator

\[ \nabla_x = \begin{bmatrix} \frac{\partial}{\partial x_1} \\ \frac{\partial}{\partial x_2} \\ \vdots \\ \frac{\partial}{\partial x_n} \end{bmatrix}^T \]

• Jacobian is defined as

\[ F_x = \begin{bmatrix} f_1(x) \\ f_2(x) \end{bmatrix} \cdot \begin{bmatrix} \frac{\partial}{\partial x_1} \\ \vdots \\ \frac{\partial}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \cdots & \frac{\partial f_2}{\partial x_n} \end{bmatrix} \]

• Linearization of a function

\[ F(x) = F(x_0) + J_F(x_0) dx \]
Linearization

• But our system is not linear, e.g. cannot be written in the form
  \[ \dot{x} = Ax \]

• Linearization of the system
  \[ \dot{x} = J_F(x_0)dx + F(x_0) \]

  \[ F(x) = F(x_0) + J_F(x_0)dx \]
Kinematic Position Control: Stability Issue

- Continuous linear time-invariant system is exponentially stable if and only of the system has eigenvalues (i.e. poles of input-to-output systems) with strictly negative real parts.

- Exponential Stability is a form of asymptotic stability.

- In practice the system will not “blow up” give unbounded output, when given an finite input and non-zero initial condition.
Kinematic Position Control: Stability Issue

- It can further be shown, that the closed loop control system is locally exponentially stable if

\[ k_\rho > 0 \ ; \ k_\beta < 0 \ ; \ k_\alpha - k_\rho > 0 \]

- Proof: linearize around equilibrium for small \( x \rightarrow \cos x = 1, \sin x = x \)

\[
\begin{bmatrix}
\dot{\rho} \\
\dot{\alpha} \\
\dot{\beta}
\end{bmatrix} =
\begin{bmatrix}
-k_\rho & 0 & 0 \\
0 & -(k_\alpha-k_\rho) & -k_\beta \\
0 & -k_\rho & 0
\end{bmatrix}
\begin{bmatrix}
\rho \\
\alpha \\
\beta
\end{bmatrix}
\]

\[ A =
\begin{bmatrix}
-k_\rho & 0 & 0 \\
0 & -(k_\alpha-k_\rho) & -k_\beta \\
0 & -k_\rho & 0
\end{bmatrix}
\]

- and the characteristic polynomial of the matrix \( A \) of all roots have negative real parts.

\[
(\lambda + k_\rho)(\lambda^2 + \lambda(k_\alpha - k_\rho) - k_\rho k_\beta)
\]