ITMO UNIVERSITY

Learning Book

International summer school of Control Systems and Robotics

Saint Petersburg
2019

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

УНИВЕРСИТЕТ ИТМО

О.И. Борисов, А.А. Ведяков, С.М. Власов, К.А. Зименко, А.А. Капитонов, С.А. Колюбин, А.Ю. Краснов, А.А. Маргун, М.М. Синетова, О.В. Слита

INTERNATIONAL SUMMER SCHOOL OF CONTROL SYSTEMS AND ROBOTICS. LEARNING BOOK. PART 3 / МЕЖДУНАРОДНАЯ ЛЕТНЯЯ ШКОЛА ПО СИСТЕМАМ УПРАВЛЕНИЯ РОБОТОТЕХНИКЕ. УЧЕБНОЕ ПОСОБИЕ. ЧАСТЬ 3

УЧЕБНОЕ ПОСОБИЕ

РЕКОМЕНДОВАНО К ИСПОЛЬЗОВАНИЮ В УНИВЕРСИТЕТЕ ИТМО
по направлению подготовки 15.04.06
в качестве учебного пособия для реализации основных профессиональных образовательных программ высшего образования бакалавриата

УНИВЕРСИТЕТ ИТМО

Санкт-Петербург
2019

О.И. Борисов, А.А. Ведяков, С.М. Власов, К.А. Зименко, А.А. Капитонов, С.А. Колюбин, А.Ю. Краснов, А.А. Маргун, М.М. Синетова, О.В. Слита, International summer school of Control Systems and Robotics. Learning book. Part 3 / Международная летняя школа по системам управления робототехнике. Учебное пособие. Часть 3 – СПб: Университет ИТМО, 2019. – 250 с.

Рецензент:

Николаев Николай Анатольевич, кандидат технических наук, доцент (квалификационная категория "ординарный доцент") факультета систем управления и робототехники, Университета ИТМО.

The textbook contains theoretical material for studying Control Systems and Robotics. The order of topics follows the structure of the lectures given at ITMO University, Faculty of Control Systems and Robotics. The modern control approaches and digital control systems in robotics are considered. The textbook is intended to foreign students majoring in specialization 15.04.06 Mechatronics and Robotics.

Учебное пособие содержит теоретический материал для изучения систем управления и робототехники. Темы в пособии отражают структуру лекций, читаемых в Университете ИТМО на факультете Систем управления и робототехники. В учебном пособии рассматриваются современные подходы к управлению, а также цифровые системы управления в робототехнике. Учебное пособие предназначено для иностранных студентов, обучающихся по направлению подготовки 15.04.06 Мехатроника и робототехника.

УНИВЕРСИТЕТ ИТМО

Университет ИТМО – ведущий вуз России в области информационных и фотонных технологий, один из немногих российских вузов, получивших в 2009 году статус национального исследовательского университета. С 2013 года Университет ИТМО – участник программы повышения конкурентоспособности российских университетов среди ведущих мировых научно-образовательных центров, известной как проект «5 в 100». Цель Университета ИТМО – становление исследовательского университета мирового уровня, предпринимательского по типу, ориентированного на интернационализацию всех направлений деятельности.

Table of Contents

The modern theory of control systems
Mathematical basics of the systems theory

Mathematical Basics of the Systems Theory

Olga Slita

Matrices

- What is a matrix?

A mathematical object of the following view:

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{bmatrix}$$

Dimension of matrix A

$$\dim \mathrm{A} = (\mathrm{n} \times \mathrm{m})$$

Vectors

What is a vector?

A mathematical object of the following view:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

Dimension of vector x

$$\dim x = (1 \times \mathrm{n})$$

Matrix Operations

- Addition $A + B = B + A$,
  $\dim \mathrm{A} = \dim B = (\mathrm{n} \times \mathrm{m})$
- Multiplication $A \cdot B = C$,
  $\dim \mathrm{A} = (\mathrm{n} \times \mathrm{m}), \dim B = (\mathrm{m} \times \mathrm{r}),$
  $\dim C = (\mathrm{n} \times \mathrm{r}), A \cdot B \neq B \cdot A$
- Inversion $A \cdot A^{-1} = I$,
  $\dim A = \dim A^{-1} = \dim I = (\mathrm{n} \times \mathrm{n})$

$I = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$ is identity matrix.

Matrix Operations

- Transposition

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{bmatrix}$$

$$A^T = \begin{bmatrix} a_{11} & a_{21} & \dots & a_{n1} \\ a_{12} & a_{22} & \dots & a_{n2} \\ \dots & \dots & \dots & \dots \\ a_{1m} & a_{2m} & \dots & a_{nm} \end{bmatrix}$$

Matrix Characteristics

Consider $(\mathrm{n} \times \mathrm{n})$ matrix A

- **Determinant** of A $\det(A)$ is calculated according to the following relation (decomposition by column or row)

$$\det(A) = \sum_{i=1}^{n} a_{ij} A_{ij} = \sum_{k=1}^{n} a_{jk} A_{jk},$$

$$(j = 1,2, \dots, n)$$

A_{ij}, A_{jk} are algebraic adjuncts of entries a_{ij} and a_{jk}

Matrix Characteristics

- **Rank** $rank(A)$ is the greatest number of linearly independent columns (or rows) of A
- **Trace** $tr(A)$ of matrix A is sum of its diagonal entries

$$tr(A) = \sum_{i=1}^{n} a_{ii} \qquad (1)$$

Why do we use matrices and vectors?

We use matrices and vectors to write down equations in a compact form.

Consider a system of linear equations:

$$\begin{cases} 2x_1 + 3x_2 + 5x_3 = 1 \\ 3x_1 + 4x_3 = 2 \\ x_2 - 6x_3 = 0 \end{cases}$$

We can rewrite it using matrices and vectors as

$$Ax = B,$$

where $A = \begin{bmatrix} 2 & 3 & 5 \\ 3 & 0 & 4 \\ 0 & 1 & -6 \end{bmatrix}, x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}, B = \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix}.$

Eigenvalues and eigenvectors

Consider the following equality

$$A\xi = \lambda\xi \qquad (2)$$

where A is a square matrix, λ is its **eigenvalue** and ξ is its **eigenvector**.

Let us rewrite (2) as

$$(\lambda I - A)\xi = 0. \qquad (3)$$

Eigenvalues λ are the roots of the polynomial

$$det(\lambda I - A) = 0 \qquad (4)$$

$D(\lambda) = det(\lambda I - A)$ is called **characteristic polynomial** of matrix A.

Functions of Matrices

Consider a polynomial

$$f(\alpha) = a_0 + a_1\boldsymbol{\alpha} + a_2\boldsymbol{\alpha}^2 \ldots + a_p\boldsymbol{\alpha}^p, \qquad (5)$$

$\boldsymbol{\alpha}$ is scalar variable, a_i are coefficients.

Replace $\boldsymbol{\alpha}$ with matrix A $(\mathrm{n} \times \mathrm{n})$.

Then we have a function of matrix

$$f(A) = a_0 + a_1 A + a_2\boldsymbol{A}^2 \ldots + a_p\boldsymbol{A}^p. \qquad (6)$$

Caley-Hamilton Theorem

Any square matrix A $(\mathrm{n} \times \mathrm{n})$ with characteristic polynomial

$$D(\lambda) = det(\lambda I - A) = \lambda^n + a_{n-1}\lambda^{n-1} + \cdots + a_0$$

satisfies its own characteristic equation, i.e.

$$D(A) = A^n + a_{n-1}A^{n-1} + \cdots + a_0 I = 0$$

Examples of Functions of Matrices

1. $e^{\alpha} = 1 + \alpha + \frac{1}{2!}\alpha^2 + \frac{1}{3!}\alpha^3 + \cdots = \sum_{i=0}^{\infty}\frac{1}{i!}\alpha^i$

$$e^{A} = I + A + \frac{1}{2!}A^2 + \frac{1}{3!}A^3 + \cdots = \sum_{i=0}^{\infty}\frac{1}{i!}A^i$$

2. $\cos\alpha = 1 + \frac{1}{2!}\alpha^2 + \frac{1}{4!}\alpha^4 + \frac{1}{6!}\alpha^6 + \cdots$

$$\cos A = 1 + \frac{1}{2!}A^2 + \frac{1}{4!}A^4 + \frac{1}{6!}A^6 + \cdots$$

Matrix exponent

$$e^{At} = I + At + \frac{1}{2!}A^2t^2 + \frac{1}{3!}A^3t^3 + \cdots = \sum_{i=0}^{\infty}\frac{1}{i!}A^it^i$$

For matrices of simple structure (λ_i: $\lambda_i \neq \lambda_j, i \neq j$, $Im(\lambda_i) = 0, i = 1 \ldots n$) **matrix exponent** is calculated as $e^{At} = Me^{\Lambda t}M^{-1} =$

$$= M\begin{bmatrix} e^{-\lambda_1 t} & 0 & \cdots & 0 \\ 0 & e^{-\lambda_2 t} & \cdots & 0 \\ \cdots & \cdots & \ddots & \cdots \\ 0 & 0 & \cdots & e^{-\lambda_2 t} \end{bmatrix} M^{-1},$$

where $\Lambda = \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \lambda_n \end{bmatrix}$, $M = [\xi_1 \quad \cdots \quad \xi_n]$.

Why do we need matrices and vectors in the systems theory?

In the systems theory we usually describe plants as systems of differential equations:

$$\begin{cases} \dot{x}_1 = a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n + \\ \quad + b_{11}u_1 + \cdots b_{1r}u_r \\ \dot{x}_2 = a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n + \\ \quad + b_{21}u_1 + \cdots b_{2r}u_r \\ \quad \vdots \\ \dot{x}_n = a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n + \\ \quad + b_{n1}u_1 + \cdots b_{nr}u_r \\ y_1 = c_{11}x_1 + \cdots + c_{1n}x_n \\ \quad \vdots \\ y_l = c_{l1}x_1 + \cdots + c_{ln}x_n \end{cases} \quad (7)$$

If we use matrices and vectors, system (7) can be rewritten in a compact form

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases} \quad (8)$$

where $x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$ is state vector, $y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_l \end{bmatrix}$ is output vector, $u = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_r \end{bmatrix}$ is control; matrix A is state matrix, $\dim \mathrm{A} = (\mathrm{n} \times \mathrm{n})$, B is control matrix and C is output matrix.

Example

Consider direct current motor.

EMF balance equation:

$$U_y = E + IR + L\frac{dI}{dt} \quad (E1)$$

where Uy is supply voltage, $E{=}ke\omega$ is back EMF, R and L are rotor resistance and inductance, I is rotor current, ω is rotor speed.

Example

Equation of rotor motion

$$J_\Sigma \frac{d\omega}{dt} = M_m - M_d \qquad \text{(E2)}$$

where J_Σ is moment of inertia of all rotating parts, M_m is engine torque, k_m is torque coefficient (constructive constant), M_d is moment of all disturbances.

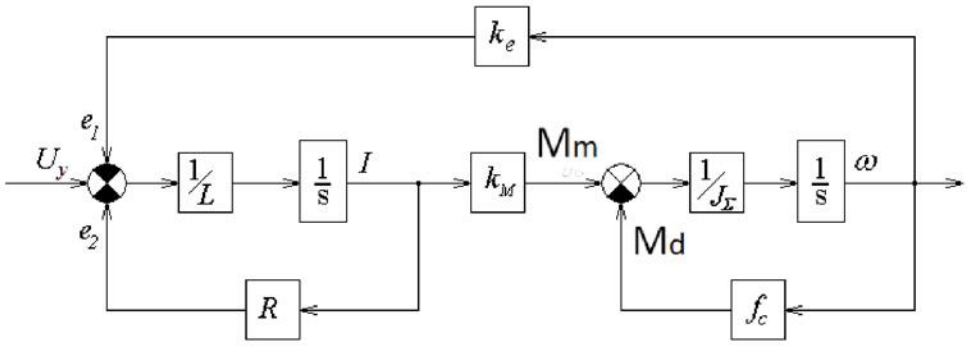
Example

Then motor model can be written as

$$\begin{cases} \dot{I} = \frac{1}{L}(U_y - e_1 - e_2), \\ \dot{\omega} = \frac{1}{J_\Sigma}(M_{дв} - M_c), \end{cases} \qquad \text{(E3)}$$

$e_1 = k_e\omega$, $e_2 = RI$, $M_m = k_M I$.

Example



Structure chart of (E3)

Example

Let us choose state variables as $x_1 = I$, $x_2 = \omega$ and output as x_1. Denote $u = U_y$.

Then we have

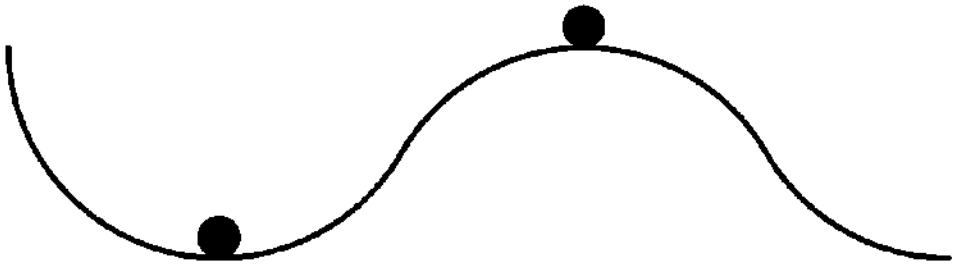
$$\begin{cases} \dot{x}_1 = -\frac{f_c}{J_\Sigma} x_1 + k_M \cdot \frac{1}{J_\Sigma} x_2 \\ \dot{x}_2 = \frac{1}{L}(u - k_e x_1 - R x_2) \\ y = x_1 \end{cases} \quad \text{(E4)}$$

Example

We can now use (E4) to obtain matrices A,B, and C:

$$A = \begin{bmatrix} -\frac{f_C}{J_\Sigma} & \frac{k_M}{J_\Sigma} \\ -\frac{k_e}{L} & -\frac{R}{L} \end{bmatrix}, B = \begin{bmatrix} 0 \\ 1/L \end{bmatrix}, C = [1 \quad 0].$$

Stability



Stability is an ability of a system to return to equilibrium position when external disturbances stope influencing it.

Let us use system (8)

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases}$$

Solution of system (8) is

$$\begin{cases} x(t) = e^{At}x(0) + \int_0^t e^{A(t-\tau)}Bu(\tau)d\tau \\ y(t) = Cx(t) \end{cases} \quad (9)$$

$e^{At}x(0)$ is free component of the solution,

$\int_0^t e^{A(t-\tau)}Bu(\tau)d\tau$ is forced component of the solution.

Thus, stability is determined by free behavior $x_f = e^{At}x(0)$.

Stability

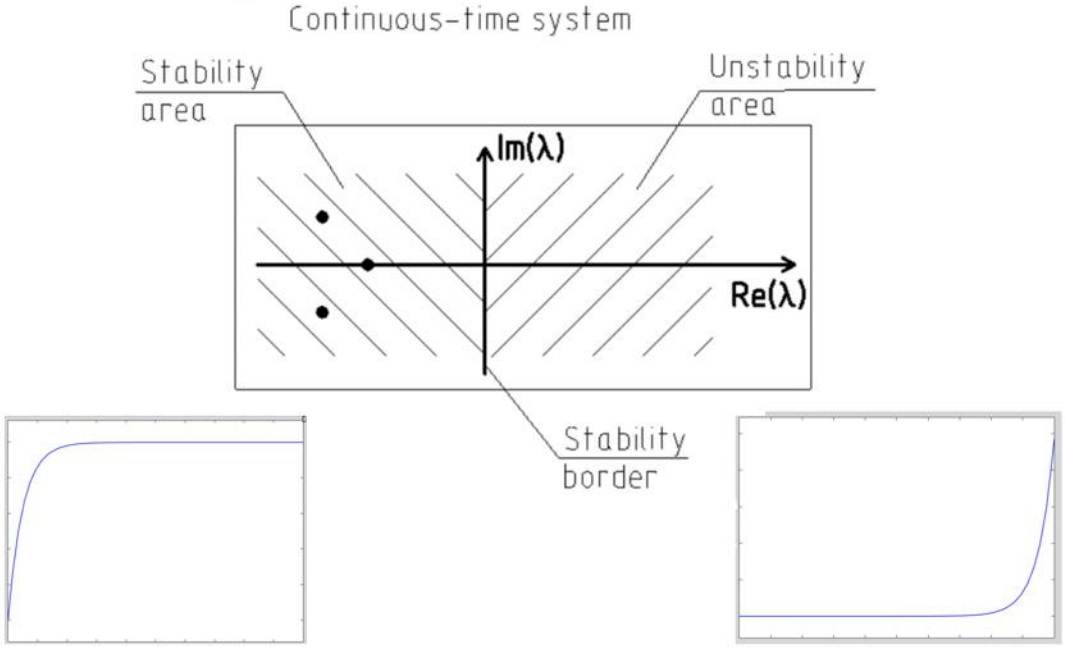
Free behavior

$$x_f(t) = Me^{\Lambda t}M^{-1}x(0) =$$

$$= [\xi_1 \quad \dots \quad \xi_n] \begin{bmatrix} e^{\lambda_1} & \dots & 0 \\ \dots & \dots & \dots \\ 0 & \dots & e^{\lambda_n} \end{bmatrix} M^{-1}x(0)$$

$\Rightarrow$ Behavior x_f is defined by eigenvalues of A.

Stability



Controllability

- Controllability is a property of a system
- A system is controllable if it can be forced to move from an arbitrary initial state $x(0)$ to an arbitrary final state $x(t_f)$ within a finite time using a properly chosen control $u(t)$
- Controllability criterion: controllability matrix

$$W_c = [B \vdots AB \vdots \cdots \vdots A^{n-1}B]$$

is a matrix of full rank, i.e. $rank(W_c) = n$.

Observability

- Observability is a property of a system
- A system is observable if its state $x(t)$ can be calculated within a time interval $t_0 \leq t \leq t_f$ on the base of observation of its output $y(t)$ and (possibly) control $u(t)$.
- Observability criterion: observability matrix

$W_o = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix}$ is a matrix of full rank,

i.e. $rank(W_o) = n$.

Automatic control theory
Stability types and Lyapunov equations

Automatic Control Theory.
Stability types and Lyapunov Equations

Madina Sinetova

Stability

- *Stability* is the system ability to return to initial position after stopping action to system external disturbances.

Stable system

Unstable system

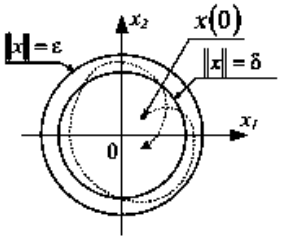
Segway

2

Stability types

1. Lyapunov stability.

- Guarantees bounded of all trajectories, but not guarantees convergence to some steady value.



where x_1, x_2 are state coordinates, ε, δ – some small numbers; as norm of x_1 and x_2 can be used quadratic norm, for example; $x(0)$ – initial position of trajectory.

- The equilibrium $x = 0$ is *Lyapunov stable* if for any small number $\varepsilon > 0$, exists small number $\delta(\varepsilon) > 0$, that for all trajectories starting from the initial conditions $\|x(0)\| \leq \delta(\varepsilon)$ for any time $\forall t \geq 0$ following inequality is satisfied: $\|x(t)\| \leq \varepsilon$.

3

Root stability criterion

Given continuous system:

$$\dot{x} = Fx, x \in R^n, F - n \times n.$$

Characteristic polynomial of the given system:

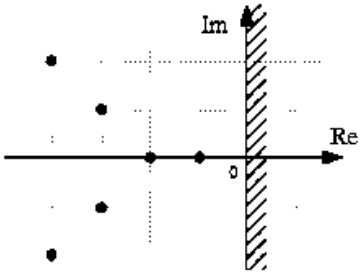
$$\det(F - sI) = s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0 = 0,$$

where $s_i, i = \overline{1,n}$ – roots of the polynomial, I – identity matrix.

If all roots have *negative real parts* $\mathrm{Re}(s_i) < 0, i = \overline{1,n}$, then the system is stable.

Im – imaginary axis (*stability border),*

Re – real axis.



4

Root stability criterion

In discrete case instead of function derivative is used value on the next discrete step:

$$\dot{f}(t) \sim f(m+1),$$

where m – number of discrete interval, $t = mT$ – continuous time, T – value of discrete interval.

Consider the discrete system:

$$x(m+1) = F_d x(m), x \in R^n, F_d - n \times n.$$

Characteristic polynomial of the given system:

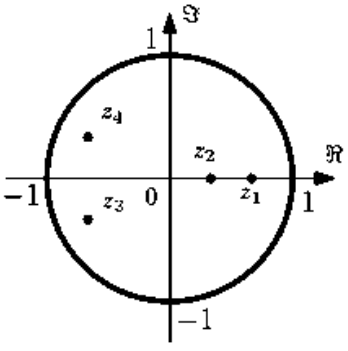
$$\det[F_d - zI] = z^n + \alpha_{n-1} z^{n-1} + \cdots + \alpha_1 z + \alpha_0 = 0,$$

where z – is a delay, I – is an identity matrix, and $z_i, i = \overline{1,n}$ – roots of the polynomial.

5

Root stability criterion

If all absolute values of roots less than one $|z_i| < 1, i = \overline{1,n}$, then the system is *stable*.



- The unit circle is a *stability border*.
- If one or more than one absolute values of roots more than $|z_i| > 1$, the system is *unstable*.

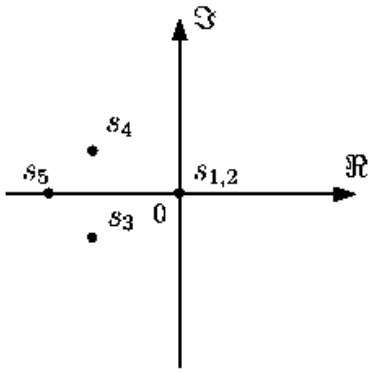
6

Stability borders

1.1. Stability border of neutral type.

Dynamic **continuous** system is on the border of neutral type if one or two roots of characteristic polynomial are equal to zero and rest roots have negative real parts:

$$s_{1,2} = 0, \mathrm{Re}(s_i) < 0, i = \overline{3,n}.$$



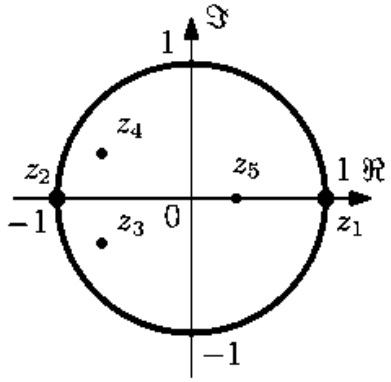
7

Stability borders

1.1. Stability border of neutral type.

Dynamic **discrete** system is on the border of neutral type if one or two roots of characteristic polynomial are equal to one and rest roots are in the unit circle:

$$|z_{1,2}| = 1, z_i < 1, i = \overline{3,n}.$$



8

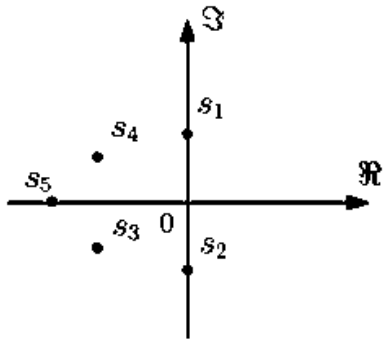
Stability borders

1.2. Stability border of oscillatory type.

The dynamic **continuous** system is on the border of oscillatory type if the characteristic polynomial has pair of purely imaginary roots and rest roots have negative real parts:

$$s_{1,2} = \pm j\omega, \omega > 0, \mathrm{Re}(s_i) < 0, i = \overline{3,n},$$

where j – imaginary unit.



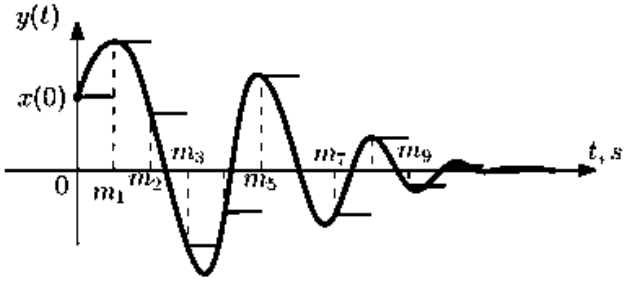
9

Stability types

2. Asymptotic stability

The equilibrium $x = 0$ is *asymptotically stable* if the equilibrium is Lyapunov stable and for any motion trajectories $x(t)$ from the arbitrary initial conditions $x(0)$ the condition $\lim_{t\to\infty} \|x(t)\| = 0$ is satisfied.

In discrete case trajectories are $x(m)$ and condition is $\lim_{m\to\infty} \|x(m)\| = 0$.

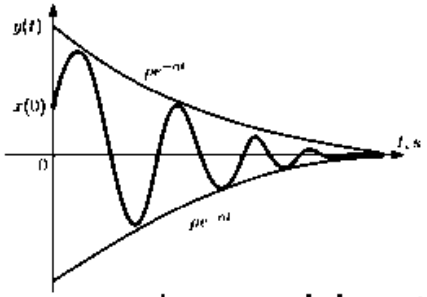


10

Stability types

3. Exponential stability

The equilibrium $x = 0$ is *exponential stable* if for any motion trajectories $x(t)$ from the arbitrary initial conditions $x(0)$ exists positive number $\alpha > 0$ that for any time $\forall t \geq 0$ inequality: $\|x(t)\| \leq \rho e^{-\alpha t}\|x(0)\|;\ \rho \geq 1$ is satisfied.



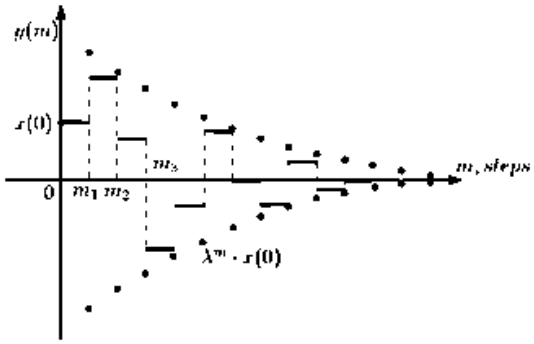
Constant α is the convergence degree and characterizes convergence velocity to equilibrium.

11

Stability types

3. Exponential stability

In discrete case: $\|x(m)\| \leq \rho\lambda^m\|x(0)\|;\ \rho \geq 1, \lambda < 1.$



Number λ characterizes *convergence velocity*. The smaller λ the faster convergence.

12

Stability types

4. Qualitative exponential stability

The equilibrium $x = 0$ is qualitative exponential stable if for any motion trajectories $x(t)$ from the arbitrary initial conditions exists numbers $\alpha > 0, r > 0, \rho \geq 1$ that for any time $\forall t \geq 0$ the following inequality:

$$\|x(t) - e^{-\alpha t}x(0)\| \leq \rho\left(e^{-(\alpha+r)t} - e^{-\alpha t}\right)\|x(0)\|$$

is satisfied.

In discrete case:

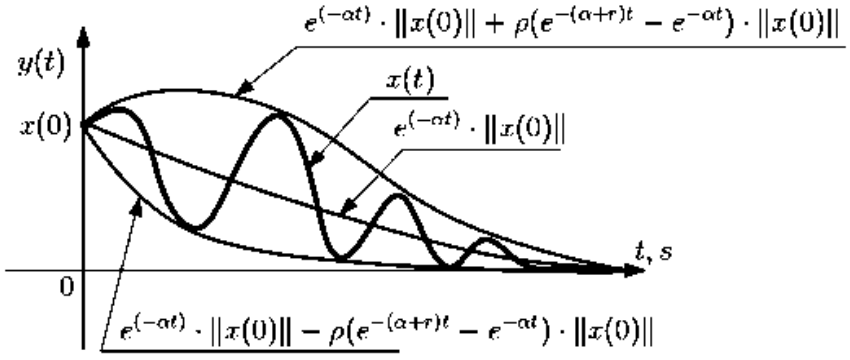
$$\|x(m) - \alpha^m x(0)\| \leq \rho\left((\alpha + r)^m - \alpha^m\right)\|x(0)\|,$$

where $0 \leq \alpha < 1 - r$.

13

Stability types

4. Qualitative exponential stability



Parameter α characterizes velocity convergence to equilibrium.
Parameter r characterizes trajectory average deviation.

14

Lyapunov functions

Lyapunov functions $V(x)$ have properties:

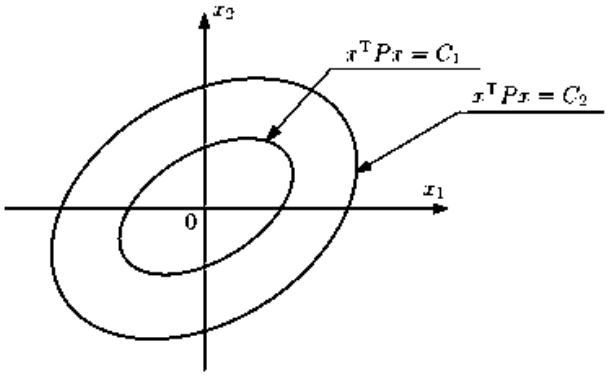
1. Lyapunov function $V(x)$ must be positive definite: for any $\forall x \in R^n$ Lyapunov function $V(x)$ is positive definite and $V(x) = 0$ in case x is null-vector.
2. Lyapunov functions must increases (decreases) uniformly with uniform increasing (decreasing) of x-vector norm.
3. The surfaces of constant level $V(x) = C$, where C – is a constant, must cover the origin of coordinates or equilibrium.

15

Lyapunov functions

Quadratic forms: $V(x) = x^T P x$,

P – $n \times n$ positive definite symmetric square matrix.



$C_2 > C_1$,

$P = I$ – identity matrix,

$$x^T P x = \begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = C,$$

$$x_1^2 + x_2^2 = C = \|x\|^2 = \left(\sqrt{x_1^2 + x_2^2}\right)^2.$$

16

Lyapunov Theorem

The equilibrium $x = 0$ is asymptotically stable if exists Lyapunov function $V(x)$ such that for any motion trajectories $x(t)$ starting from the arbitrary initial conditions for any time $\forall t \geq 0$ the derivative of the function is negative: $\frac{dV(x(t))}{dt} < 0$.

$$\frac{dV(x(t))}{dt} = \frac{\partial V(x)}{\partial x}\frac{\partial x}{\partial t} = \frac{\partial V(x)}{\partial x}\dot{x}.$$

$x - n$-dimension state vector:

$$\frac{\partial V(x)}{\partial x} = \left[\frac{\partial V(x)}{\partial x_1} \quad \dots \quad \frac{\partial V(x)}{\partial x_n}\right] = grad^T V(x).$$

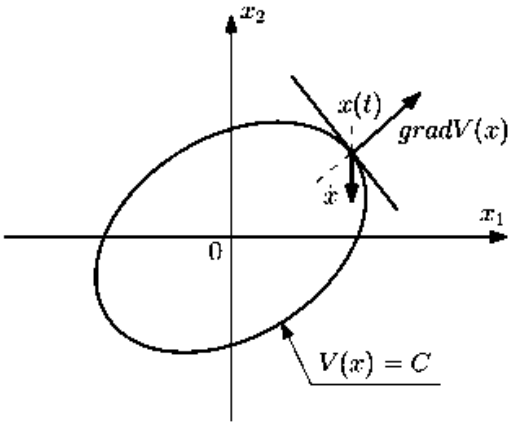
So:

$$\frac{dV(x(t))}{dt} = grad^T x \cdot \dot{x}.$$

17

Lyapunov Theorem

Geometric interpretation:



$V(x) = C$ – surface of constant level.

18

Example

Given:

$$\dot{x} = -x^3.$$

Lyapunov function:

$$V(x) = x^2.$$

Investigation:

$$\frac{dV(x(t))}{dt} = 2x \cdot \dot{x} = 2x \cdot (-x^3) = -2x^4 < 0.$$

Lyapunov function derivative is negative anytime, so, the given system is asymptotically stable.

19

Lyapunov inequalities

- For asymptotic stability:

$$\dot{V}(x(t)) < 0.$$

- For exponential stability:

$$\dot{V}(x(t)) \leq -2\alpha V(x(t)), \alpha > 0.$$

- For qualitative exponential stability:

$$V(\dot{x}(t) + (r + \alpha)x(t)) \leq r^2 V(x(t)).$$

20

Rayleigh ratio

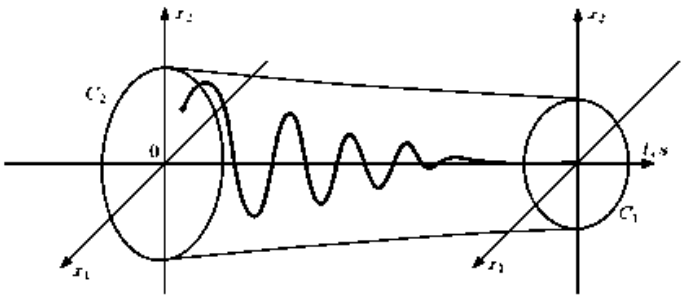
Let's consider inequality:

$$C_1^2\|x\|^2 \le x^T P x \le C_2^2\|x\|^2,$$

where $\dot{x} = Fx, x \in R^n, F - n\times n, V(x) = x^T P x, P - n\times n$ positive definite symmetric square matrix.

Omitting intermediate calculations obtain ratio:

$$\|x(t)\| \le \frac{C_2}{C_1} e^{-\alpha t}\|x(0)\|, \rho = \frac{C_2}{C_1} \ge 1.$$



21

Lyapunov equations

From Lyapunov inequalities follows Lyapunov equations:

- For asymptotic stability:

$$F^T P + PF = -Q.$$

- For exponential stability:

$$F^T P + PF + 2\alpha F = -Q.$$

- For qualitative exponential stability:

$$(F + (r+\alpha)I)^T P(F + (r+\alpha)I) - r^2 P = -Q.$$

F –state matrix of closed system, P, Q – positive definite symmetric square matrices of the same dimension.

For investigation stability it's required to choose matrix Q, solve Lyapunov equation with respect to matrix P and check it for positive definition.

22

Example

Given:

$$\dot{x} = Fx, \begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & -2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}.$$

Is system stable?

Choose:

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, P = \begin{bmatrix} p_1 & p_3 \\ p_3 & p_2 \end{bmatrix}.$$

And calculate:

$$F^T P = \begin{bmatrix} -p_3 & -p_2 \\ p_2 - 2p_3 & p_3 - 2p_2 \end{bmatrix}, PF = \begin{bmatrix} -p_3 & p_1 - 2p_3 \\ -p_2 & p_3 - 2p_2 \end{bmatrix},$$

23

Example

$$\begin{cases} -p_3 - p_3 = -1 \\ -p_2 + p_1 - 2p_3 = 0 \\ p_2 - 2p_3 - p_2 = 0 \\ p_3 - 2p_2 + p_3 - 2p_2 = -1 \end{cases} \Rightarrow P = \begin{bmatrix} 1{,}5 & 0{,}5 \\ 0{,}5 & 0{,}5 \end{bmatrix}.$$

$$\det(P - \lambda I) = \lambda^2 - 2\lambda + 0{,}5 = 0 \Rightarrow \lambda_1 = 0{,}29, \lambda_2 = 1{,}7.$$

All eigenvalues λ_i of matrix P are more than zero, so, system is asymptotically stable.

24

Qualitative exponential stability

Automatic Control Theory.
Qualitative exponential stability for discrete and continuous linear systems

Madina Sinetova

Qualitative exponential stability

Consider linear discrete system:

$$x(m+1) = F\big(x(m)\big).$$

Equilibrium $x = 0$ is *exponential stable*, if exists numbers $\rho > 0$, $\alpha > 0$, $d_x(\alpha) > 0$ that for all initial values of $\|x(0)\| \le d_x(\alpha)$ for any number of discreetness interval $m > 0$ the following inequality is satisfied:

$$\|x(m)\| \le \rho \cdot e^{-\alpha m} \cdot \|x(0)\|.$$

Introduce notation: $\lambda = e^{-\alpha}$, where $0 < \lambda < 1$, so:

$$\|x(m)\| \le \rho \cdot \lambda^m \cdot \|x(0)\|.$$

All trajectories $x(m)$ of exponential stable system are in «estimated tube» bounded by surfaces:

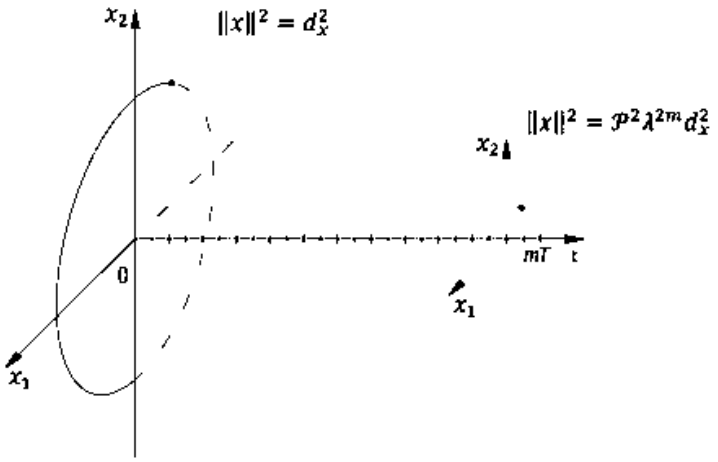
$$\|x(m)\|^2 = (\rho \cdot \lambda^m \cdot \|x(0)\|)^2.$$

2

Qualitative exponential stability

Surfaces consist of circles with radiuses:

$$\rho \cdot \lambda^m \cdot \|x(0)\|.$$



3

Local term

Equilibrium $x = 0$ is *qualitative exponential stable*, if system is exponential stable with parameters α $(\lambda = e^{-\alpha})$, $d_x(\alpha)$ and additionally exists positive number $0 < \lambda_0 < 1 + \lambda$ that for any number of discreetness interval $m > 0$ the following inequality is satisfied:

$$\|x(m) - x(0)\| \leq \lambda_0\rho \sum_{i=0}^{m-1} \lambda^i \|x(0)\| = \lambda_0\rho \frac{1-\lambda^m}{1-\lambda} \|x(0)\|.$$

This condition constrains state vector current values deviation from initial conditions $x(0)$.

System have very qualitative parameters under condition: $\lambda_0 < 1$. In this case inequality is strongest.

4

Qualitative exponential stability

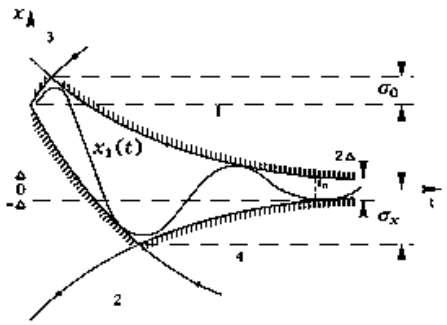
In case of non-zero initial conditions $x(0)$ trajectories are bounded by surfaces:

$$\|x(m) - x(0)\|^2 = \left[\lambda_0 \rho \frac{1-\lambda^m}{1-\lambda} d_x\right]^2.$$

with circles' radiuses:

$$\lambda_0 \rho \frac{1-\lambda^m}{1-\lambda} d_x.$$

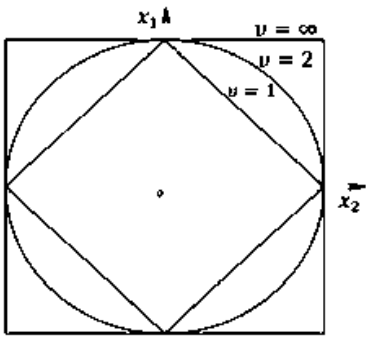
«Estimated tube» cross-section:



$\sigma_{0x}^* = \frac{(\rho-1)\lambda_0}{\lambda+\lambda_0-1}$ - first ejection;

$\sigma_x^* = \frac{\lambda-\rho\lambda_0-1}{1-\lambda+\lambda_0}$ - overcontrol.

Vector norms



Arbitrary vector norm:

$$\|x\| = \left[\sum_{i=1}^{n} |x_i|^v\right]^{\frac{1}{v}},$$

v is an integer and $v = 1, 2, \ldots$, and x_i – *i*-th component of state vector x.

If $v = 2$ the norm is *Euclidean*, if $v = 1$ the norm is *absolute*.

In case of two state components constant level surfaces are:

$$\|x\|^v = 1 \ (x \in R^2).$$

Remark. For deterministic processes (not for stochastic) from the convergence by some norm follows convergence by any norm.

6

Lyapunov function

Consider convex positively homogeneous Lyapunov function $V(x)$ from a class of K^v such:

$$C_1^v\|x\|^v \le V(x) \le C_2^v\|x\|^v.$$

Using quadratic forms:

$$V(x) = x^T P x,$$

where P – symmetric positive definite square matrix from a class of K^2, values C_1^2 and C_2^2 are minimum and maximum eigenvalues of matrix P respectively.

7

Sufficient conditions

For system $x(m+1) = F\big(x(m)\big)$ sufficiently existing number $0 < \lambda < 1$ that for any number of discreetness interval $m > 0$ the inequality is satisfied:

$$V\big(x(m+1)\big) \le \lambda^v V\big(x(m)\big),$$

and existing number $1 - \lambda < \lambda_0 < 1 + \lambda$, that for the system the inequality is satisfied:

$$V\big(x(m+1) - x(m)\big) \le \lambda_0^v V\big(x(m)\big).$$

From these conditions follows two consequences.

8

Consequences

Consequence 1. For qualitative exponential stability are sufficiently existing numbers $0 < \lambda < 1$ and $1 - \lambda < \lambda_0 < 1 + \lambda$ that for any number of discreetness interval $m > 0$ the following inequality is satisfied:

$$V\big(x(m+1) - (r+\alpha)x(m)\big) \le r^v V\big(x(m)\big).$$

Consequence 2. For qualitative exponential stability are sufficiently existing numbers $0 < r < 1$ and $0 < \alpha < 1 - 2r$ that for any number of discreetness interval $m > 0$ the following inequality is satisfied:

$$V\left(x(m+1) - \frac{\lambda - \lambda_0 + 1}{2}x(m)\right) \le \left(\frac{\lambda + \lambda_0 - 1}{2}\right)^v V\big(x(m)\big).$$

$\lambda = 2r + \alpha, \lambda_0 = 1 - \alpha.$

9

Geometric interpretation

Exponential stability.

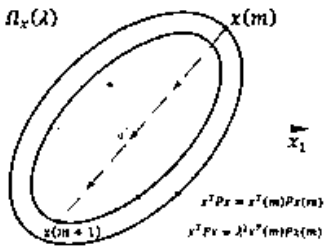
In case of quadratic form:

$$V\big(x(m+1)\big) \le \lambda^2 V\big(x(m)\big).$$

The each next value of state vector $x(m+1)$ must belongs to area:

$$\Omega_x(\lambda) = \{x: x^T P x \le \lambda^2 x^T(m) P x(m)\}$$

if the previous value of state vector was on a surface $x^T P x = x^T(m) P x(m)$.



10

Geometric interpretation

Consequence 1.

1. System must be exponential stable.

2. For Lyapunov functions from a class of K^2 the inequality which should be satisfied takes form:

$$\big(x(m+1)-x(m)\big)^T P\big(x(m+1)-x(m)\big) \le \lambda_0^2 x^T(m) P x(m).$$

And the each next value of state vector $x(m+1)$ with fixed $x(m)$ must belongs to area:

$$\Omega_x(\lambda_0) = \Big\{ x: \big(x-x(m)\big)^T P\big(x-x(m)\big) \le \lambda_0^2 x^T(m) P x(m) \Big\}.$$

3. The each fixed arbitrary value $x(m)$ the next value of state vector $x(m+1)$ belongs to area:

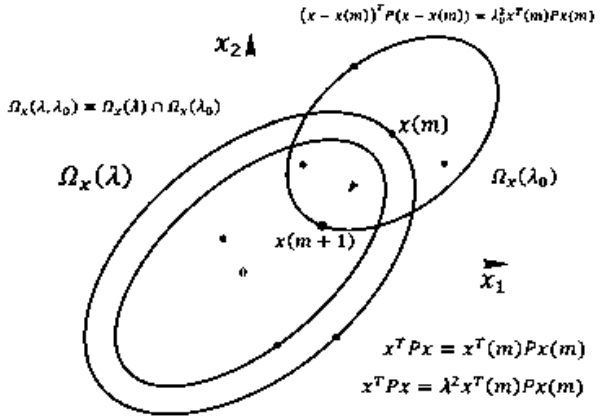
$$\Omega_x(\lambda, \lambda_0) = \Omega_x(\lambda) \cap \Omega_x(\lambda_0).$$

11

Geometric interpretation

Consequence 1.

As a result, qualitative exponential stability distinguish from all values of state vector $\Omega_x(\lambda)$ some part $\Omega_x(\lambda, \lambda_0)$ and it localizes system motion trajectories behavior.



12

Geometric interpretation

Consequence 2.

1. System must be exponential stable.

2. For Lyapunov functions from a class of K^2 the inequality which should be satisfied takes form:

$$\big(x(m+1)-(r+\alpha)x(m)\big)^T P\big(x(m+1)-(r+\alpha)x(m)\big) \le r^2 x^T(m)Px(m),$$

And the each next value of state vector $x(m+1)$ with fixed $x(m)$ must belongs to area:

$$\Omega_x(r,\alpha)=\Big\{x:\big(x-(r+\alpha)x(m)\big)^T P\big(x-(r+\alpha)x(m)\big) < r^2 x^T(m)Px(m)\Big\}.$$

3. The each fixed arbitrary value $x(m)$ the next value of state vector $x(m+1)$ belongs to area:

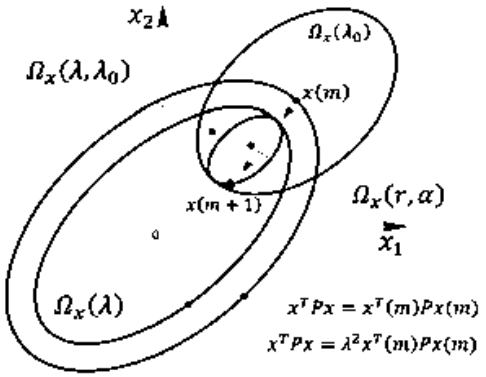
$$\Omega_x(r,\alpha) \subset \Omega_x(\lambda,\lambda_0).$$

13

Geometric interpretation

Consequence 2.

In case $\lambda = 2r+\alpha$, $\lambda_0 = 1-\alpha$ the area belongs to intersection areas $\Omega_x(\lambda_0)$ and $\Omega_x(\lambda)$: $\Omega_x(r,\alpha) \subset \Omega_x(\lambda,\lambda_0)$.



14

Summary

Consider system:

$$x(m+1) = F \cdot \big(x(m)\big),$$

$F - n \times n$ matrix of closed system.

Lyapunov inequality:

$$(F - (r+\alpha)I)^T P (F - (r+\alpha)I) \le r^2 P,$$

where $\lambda = 2r + \alpha$, $\lambda_0 = 1 - \alpha$.

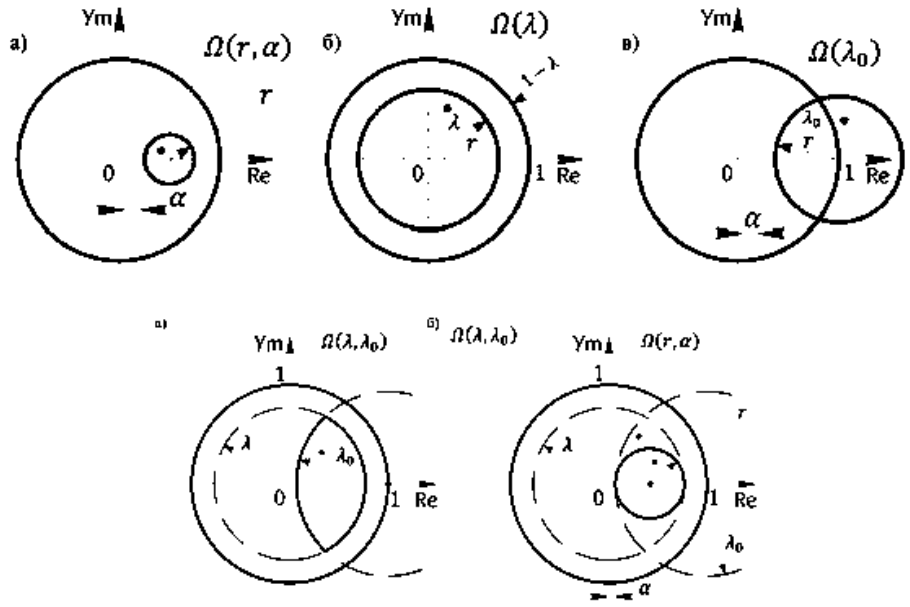
Lyapunov equation:

$$(F - (r+\alpha)I)^T P (F - (r+\alpha)I) - r^2 P = -Q,$$

where $P > 0$ – positive definite symmetric square matrix, $Q \ge 0$ – positive semi-definite symmetric square matrix.

15

Roots distribution



16

Continuous case

Consider continuous linear system:

$$\dot{x} = F(x),$$

x – state vector.

Local conditions:

1. There is a number $\alpha > 0$ such for any time $t > 0$ following inequality is satisfied:

$$\dot{V}\big(x(t)\big) \le -2\alpha V\big(x(t)\big),$$

2. There is a number $\lambda_0 \ge \alpha$ such for any time $t > 0$ following inequality is satisfied:

$$V\big(\dot{x}(t)\big) \le \lambda_0^2 V\big(x(t)\big),$$

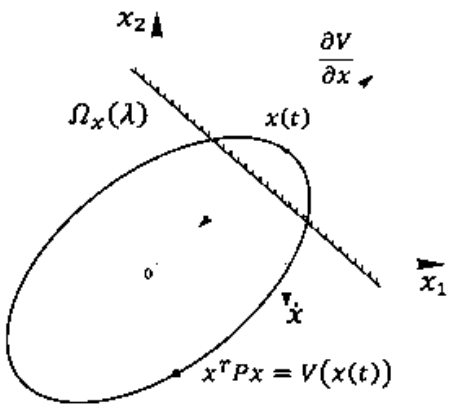
where $V\big(x(t)\big)$ – Lyapunov function from a class K^2 of quadratic forms.

17

Geometric interpretations

1st condition:

$$\frac{\partial V(x)}{\partial x}\dot{x} \le -2\alpha V(x).$$



18

Geometric interpretations

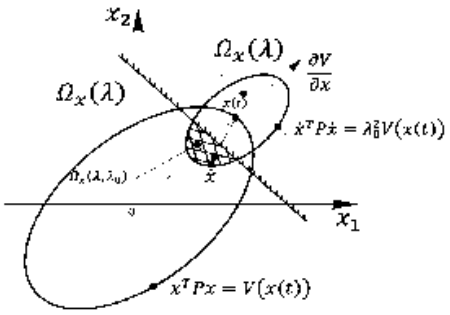
2nd condition:

All possible values of state vector must belong to area $\Omega_x(\lambda_0)$ bounded by surface:

$$\dot{x}^T P \dot{x} = \lambda_0^2 V\big(x(t)\big).$$

As a result, combine both conditions:

Values of state vector must belong to area $\Omega_x(\alpha, \lambda_0) = \Omega_x(\alpha) \cap \Omega_x(\lambda_0)$.



19

Geometric interpretations

As one condition.

System is qualitative exponential stable if exists numbers $\alpha > 0$ and $r > 0$ such for any time $t > 0$ following inequality is satisfied:

$$V\big(\dot{x}(t) + (r+\alpha)x(t)\big) \le r^2 V\big(x(t)\big),$$

where $\lambda = \alpha, \lambda_0 = 2r + \alpha$.

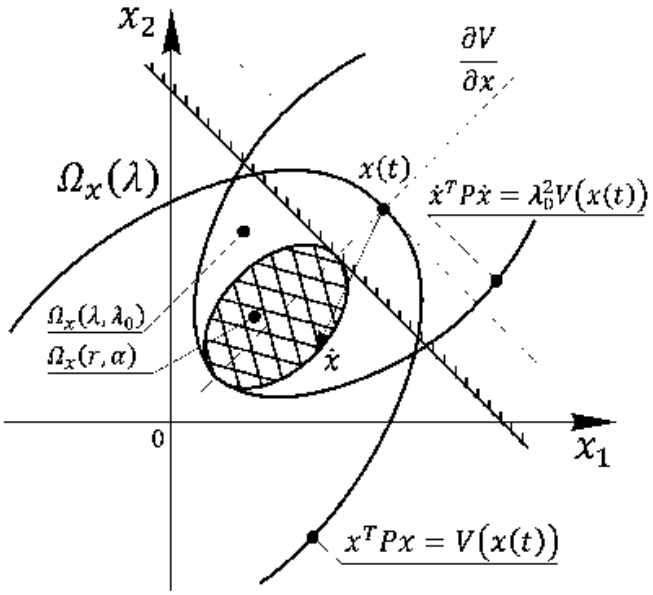
For any time $t > 0$ and for any state vector $x(t)$, velocity vector $\dot{x}(t)$ must belongs to area $\Omega_x(r, \alpha)$ bounded by surface:

$$\big(\dot{x} + (r+\alpha)x(t)\big)^T P\big(\dot{x} + (r+\alpha)x(t)\big) = r^2 V\big(x(t)\big),$$

where area $\Omega_x(r, \alpha) \subset \Omega_x(\alpha, \lambda_0)$.

20

Geometric interpretations



21

Continuous case

Linear system:

$$\dot{x} = F(x),$$

x – state vector.

Sufficient condition:

Existing such numbers (r, α): $\lambda = \alpha, \lambda_0 = 2r + \alpha$

that Lyapunov equation:

$$(F + (r + \alpha)I)^T P(F + (r + \alpha)I) - r^2 P = -Q,$$

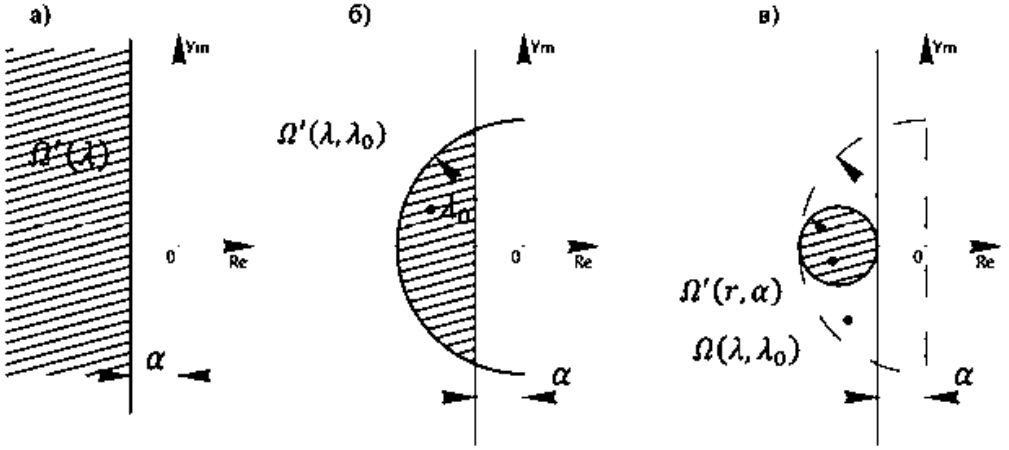
and Lyapunov inequality:

$$(F + (r + \alpha)I)^T P(F + (r + \alpha)I) \leq r^2 P$$

are satisfied.

22

Roots distribution



23

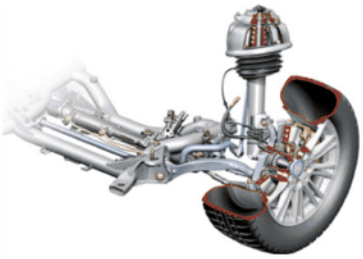
Identification theory

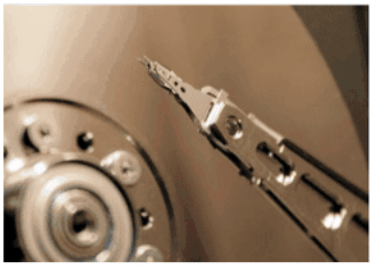
Identification Theory

Alexey Vedyakov

Application

- Measuring systems
- Disturbance compensation systems:
  - hard drives
  - ship
  - active suspension vehicle





First global convergent estimators

- Hsu L., Ortega R., Damm G. A globally convergent frequency estimator // IEEE Transactions on Automatic Control. 1999.
- G. Obregon-Pulido, B. Castillo-Toledo, A. A. Loukianov, "Globally convergent estimator for n-frequencies," IEEE Trans. Autom. Control, vol. 47, no. 5, pp. 857-863, May 2002.
- A. Bobtsov, A. Lyamin, D. Romasheva, "Algorithm of parameter's identification of polyharmonic function," in Proc. 15th IFAC World Congress on Automatic Control, Barcelona, Spain, Jul. 2002.
- X. Xia, "Global frequency estimation using adaptive identifiers," IEEE Trans. Autom. Control, vol. 47, no. 7, pp. 1188-1193, Jul. 2002.
- R. Marino, P. Tomei, "Global estimation of unknown frequencies," IEEE Trans. Autom. Control, vol. 47, no. 8, pp. 1324-1328, Aug. 2002.

Frequency estimation

Consider the measurable signal

$$y(t) = A\sin(\omega t + \phi), \tag{1}$$

where A is the amplitude, ω is the frequency, ϕ is the phase.

The goal is to obtain the frequency estimate $\hat{\omega}(t)$ such that

$$\lim_{t\to\infty} |\omega - \hat{\omega}(t)| = 0.$$

Sinusoidal signal generator

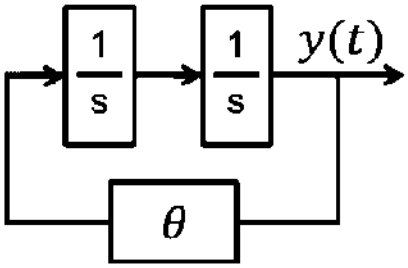
Consider derivatives of the signal (1)

$$\dot{y}(t) = \omega A \cos(\omega t + \phi),$$
$$\ddot{y}(t) = -\omega^2 A \sin(\omega t + \phi). \quad (2)$$

Using (1) and (2) we can obtain linear regression model

$$\ddot{y}(t) = \theta y(t), \quad (3)$$

where $\theta = -\omega^2$.



Gradient method

Consider the cost criterion

$$J(\theta) = \frac{1}{2}(\ddot{y}(t) - \hat{\ddot{y}}(t))^2 = \frac{1}{2}(\theta y(t) - \hat{\theta}(t) y(t))^2 = \frac{1}{2} e^2(t),$$

which we minimize with respect to $\hat{\theta}(t)$ using the gradient method

$$\dot{\hat{\theta}}(t) = -\gamma \nabla J(\hat{\theta}),$$

where $\gamma > 0$. In our case,

$$\nabla J(\hat{\theta}) = \frac{dJ}{d\theta} = -ey(t) = -y(t)\left(\ddot{y}(t) - \hat{\theta}(t) y(t)\right).$$

Finally,

$$\dot{\hat{\theta}}(t) = \gamma y(t)\left(\ddot{y}(t) - \hat{\theta}(t) y(t)\right).$$

Biased sinusoidal signal

Consider the measurable signal

$$y(t) = A_0 + A\sin(\omega t + \phi), \tag{4}$$

where A_0 is the constant bias. Consider derivatives of the signal (4)

$$\dot{y}(t) = \omega A\cos(\omega t + \phi), \tag{5}$$
$$\ddot{y}(t) = -\omega^2 A\sin(\omega t + \phi).$$
$$\dddot{y}(t) = -\omega^3 A\cos(\omega t + \phi). \tag{6}$$

Using (5) and (6) we can obtain linear regression model

$$\dddot{y}(t) = \theta\dot{y}(t).$$

The adaptive law

$$\dot{\hat{\theta}}(t) = \gamma\dot{y}(t)\left(\dddot{y}(t) - \hat{\theta}(t)\dot{y}(t)\right). \tag{7}$$

Modified version

Let us consider additional variable

$$\chi(t) = \hat{\theta}(t) - \gamma\dot{y}(t)\ddot{y}(t), \quad \text{then} \tag{8}$$
$$\hat{\theta}(t) = \chi(t) + \gamma\dot{y}(t)\ddot{y}(t). \tag{9}$$

Differentiating equation (9) we obtain

$$\dot{\hat{\theta}}(t) = \dot{\chi}(t) + \gamma\ddot{y}^2(t) + \gamma\dot{y}(t)\dddot{y}(t). \tag{10}$$

On the other hand, from (7) we have

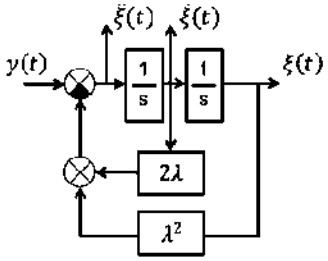
$$\dot{\hat{\theta}}(t) = \gamma\dot{y}(t)\left(\dddot{y}(t) - \hat{\theta}(t)\dot{y}(t)\right) = \gamma\dot{y}(t)\dddot{y}(t) - \gamma\hat{\theta}(t)\dot{y}^2(t). \tag{11}$$

Combining (10) and (11) gives

$$\hat{\theta}(t) = \chi(t) + \gamma\dot{y}(t)\ddot{y}(t),$$
$$\dot{\chi}(t) = -\gamma\hat{\theta}(t)\dot{y}^2(t) - \gamma\ddot{y}^2(t).$$

Without measuring derivatives

Let us consider linear filter



The signals ξ, $\dot{\xi}(t)$, $\ddot{\xi}(t)$ are measurable. Moreover,

$$\xi(t) = B_0 + B_1 \sin(\omega t + \psi) + \epsilon(t), \tag{12}$$

where $c(t)$ is exponentially decaying term. In this case,

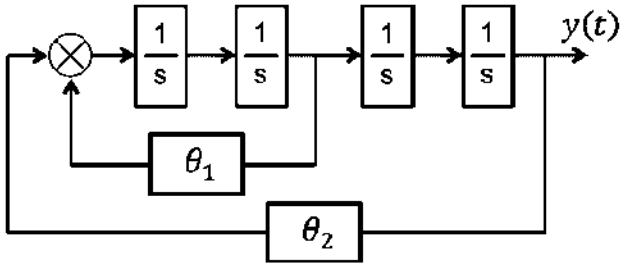
$$\hat{\theta}(t) = \chi(t) + \gamma\dot{\xi}(t)\ddot{\xi}(t),$$
$$\dot{\chi}(t) = -\gamma\hat{\theta}(t)\dot{\xi}^2(t) - \gamma\ddot{\xi}^2(t).$$

Multi-Sinusoidal signal

Consider the measurable signal

$$y(t) = A_0 + \sum_{i=1}^{k} A_i \sin(\omega_i t + \phi_i). \tag{13}$$

Signal generator for $k = 2$



where $\theta_1 = -(\omega_1^2 + \omega_2^2)$, $\theta_2 = -\omega_1^2\omega_2^2$.

Multi-Sinusoidal signal

The signal (13) can be generated by the following differential equation

$$p(p^2-\theta_1)(p^2-\theta_2)\dots(p^2-\theta_k)y(t)=0, \tag{14}$$

where $p=d/dt$ is the differentiation operator, $\theta_i=-\omega_i^2$, are constant parameters, $i=\overline{1,k}$. Equation (14) can be represented as

$$p^{2k}y(t)=\bar{\theta}_1 p^{2k-2}y(t)+\dots+\bar{\theta}_k p y(t), \tag{15}$$

where $\bar{\theta}_i$ can be calculated by the following system

$$\begin{cases} \bar{\theta}_1=\theta_1+\theta_2+\dots+\theta_k, \\ \bar{\theta}_2=-\theta_1\theta_2-\theta_1\theta_3-\dots-\theta_{k-1}\theta_k, \\ \vdots \\ \bar{\theta}_k=(-1)^{k+1}\theta_1\theta_2\dots\theta_k. \end{cases}$$

General linear filter

Introduce the linear filter

$$\xi(s)=F(s)y(s), \quad F(s)=\frac{\lambda_0^{2k}}{\gamma(s)}, \tag{16}$$

where $\lambda_0>0$, $\gamma(s)=s^{2k}+\gamma_{2k-1}s^{2k-1}+\dots+\gamma_1 s+\gamma_0$ is a Hurwitz polynomial.

Multiplying (15) by $\frac{\lambda_0^{2k}}{\gamma(s)}$ with (16) we obtain

$$s^{2k+1}\xi(s)=\bar{\theta}_1 s^{2k-1}\xi(s)+\dots+\bar{\theta}_k s\xi(s).$$

Regression model

After the inverse Laplace transformation for the filter (16) and the input signal $y(t)$ we get the relation

$$\xi^{(2k+1)}(t) = \Omega^T(t)\bar{\Theta} + \varepsilon(t),$$

where $\Omega(t)$ is a regressor of functions $\xi^{(j)}(t)$

$$\Omega^T(t) = \begin{bmatrix} \xi^{(2k-1)}(t) & \dots & \xi^{(3)}(t) & \xi^{(1)}(t) \end{bmatrix},$$

$\bar{\Theta}$ is a vector of unknown parameters depending on frequencies

$$\bar{\Theta}^T = \begin{bmatrix} \bar{\theta}_1 & \dots & \bar{\theta}_{k-1} & \bar{\theta}_k \end{bmatrix}.$$

Adaptive Frequency Estimation

The update law

$$\hat{\omega}_i = \sqrt{\left|\hat{\theta}_i\right|}, \tag{17}$$

where estimates θ_i calculated using $\hat{\bar{\theta}}_i$ that are elements of a vector $\hat{\bar{\Theta}}$:

$$\hat{\bar{\Theta}} = \Upsilon(t) + K\Omega(t)\xi^{(2k)}(t), \tag{18}$$

$$\dot{\Upsilon}(t) = -K\Omega(t)\Omega^T(t)\hat{\bar{\Theta}}(t) - K\dot{\Omega}(t)\xi^{(2k)}(t). \tag{19}$$

where $K = \mathrm{diag}\{k_i > 0, i = \overline{1,k}\}$, guarantees that the estimation error $\tilde{\omega}_i = \omega_i - \hat{\omega}_i$ exponentially converges to zero:

$$|\tilde{\omega}_i(t)| \le \rho_1 e^{-\beta_1 t}, \quad \rho_1, \beta_1 > 0, \quad \forall t \ge 0. \tag{20}$$

Harmonics observer

For the variable $\xi(t)$ we have

$$\xi(t) = \xi_0 + \xi_1(t) + \xi_2(t) + \cdots + \xi_k(t). \tag{21}$$

After differentiation (21) $2k$ times, we obtain two systems of k linear equations:

$$\begin{cases} \xi^{(1)}(t) = \dot{\xi}_1(t) + \dot{\xi}_2(t) + \cdots + \dot{\xi}_k(t), \\ \xi^{(3)}(t) = \theta_1\dot{\xi}_1(t) + \theta_2\dot{\xi}_2(t) + \cdots + \theta_k\dot{\xi}_k(t), \\ \vdots \\ \xi^{(2k-1)}(t) = \theta_1^{k-1}\dot{\xi}_1(t) + \cdots + \theta_k^{k-1}\dot{\xi}_k(t), \end{cases}$$

and

$$\begin{cases} \xi^{(2)}(t) = \theta_1\xi_1(t) + \theta_2\xi_2(t) + \cdots + \theta_k\xi_k(t), \\ \xi^{(4)}(t) = \theta_1^2\xi_1(t) + \theta_2^2\xi_2(t) + \cdots + \theta_k^2\xi_k(t), \\ \vdots \\ \xi^{(2k)}(t) = \theta_1^k\xi_1(t) + \theta_2^k\xi_2(t) + \cdots + \theta_k^k\xi_k(t). \end{cases} \tag{22}$$

Harmonics observer

From (21) and (22) we get the realizable estimation scheme for variables ξ_0 and $\xi_i(t)$

$$\begin{bmatrix} \hat{\xi}_1(t) \\ \hat{\xi}_2(t) \\ \vdots \\ \hat{\xi}_k(t) \end{bmatrix} = \begin{bmatrix} \hat{\theta}_1 & \cdots & \hat{\theta}_k \\ \hat{\theta}_1^2 & \cdots & \hat{\theta}_k^2 \\ \vdots & \ddots & \vdots \\ \hat{\theta}_1^k & \cdots & \hat{\theta}_k^k \end{bmatrix}^{-1} \begin{bmatrix} \xi^{(2)}(t) \\ \xi^{(4)}(t) \\ \vdots \\ \xi^{(2k)}(t) \end{bmatrix},$$

and

$$\hat{\xi}_0 = \xi(t) \quad \sum_{i-1}^{k} \hat{\xi}_i(t).$$

Parameters estimation

The estimates of the amplitudes and phases

$$\hat{A}_i(t) = \frac{\hat{\gamma}_{\xi i}(t)}{\hat{L}_{\xi i}(t)}, \quad \hat{\phi}_i(t) = \left(-\hat{\varphi}_{\xi i}(t) + \hat{\phi}_{\xi i}(t)\right) \bmod 2\pi,$$

where

$$\hat{\gamma}_{\xi i}(t) = \sqrt{\hat{\xi}_i^2(t) + \left(\frac{\dot{\hat{\xi}}_i(t)}{\hat{\omega}_i(t)}\right)^2},$$

$$\hat{\phi}_{\xi i}(t) = \left(\operatorname{sign}\left(\hat{\xi}_i(t)\right)\arccos\left(\frac{\dot{\hat{\xi}}_i(t)}{\hat{\gamma}_{\xi i}(t)\hat{\omega}_i(t)}\right) - \hat{\omega}_i(t)t\right) \bmod 2\pi,$$

$\hat{L}_{\xi i}(t)$ and $\hat{\varphi}_{\xi i}(t)$ can be obtained from filter frequency response

$$\hat{L}_{\xi i}(t) = |F(j\omega)|_{\omega=\hat{\omega}_i}, \quad \hat{\varphi}_{\xi i}(t) = \arg F(j\omega)|_{\omega=\hat{\omega}_i}.$$

Dynamic Regressor Extension and Mixing

Consider the regression model

$$\psi(t) = \theta^T \varphi(t), \tag{23}$$

where $\psi(t) \in \mathbb{R}$ is the regressand, $\theta \in \mathbb{R}^n$ is the constant vector of unknown parameters, $\varphi(t) \in \mathbb{R}^n$ is the regressor.

Consider two linear operators

- The stable LTI filter. For example, we can choose exponentially stable LTI filters
$$H_l(p) = \frac{\lambda_l}{p + \lambda_l}, \tag{24}$$
where $p = \frac{d}{dt}$, $\lambda_l \in \mathbb{R}_+$, $l = \overline{1,n}$.
- The delay operator
$$[H_l(\cdot)](t) = (\cdot)(t - d_l), \tag{25}$$
where $d_l > 0$ is a delay.

Let us choose delay operator and define the filtered signals

$$\phi_{f,l}(t) = \phi(t - d_l), \tag{26}$$
$$\psi_{f,l}(t) = \psi(t - d_l). \tag{27}$$

Combine (26)–(27) and signals $\phi(t)$, $\psi(t)$ as follows

$$\Phi_e(t) = \begin{bmatrix} \phi^\top(t) \\ \phi_{f,1}^\top(t) \\ \vdots \\ \phi_{f,n-1}^\top(t) \end{bmatrix}, \quad \Psi_e(t) = \begin{bmatrix} \psi(t) \\ \psi_{f,1}(t) \\ \vdots \\ \psi_{f,n-1}(t) \end{bmatrix}, \tag{28}$$

where $\Phi(t) \in \mathbb{R}^{n \times n}$, $\Psi(t) \in \mathbb{R}^{n \times 1}$.

Defining

$$\zeta(t) = \det\{\Phi(t)\}, \tag{29}$$
$$\xi(t) = \operatorname{adj}\{\Phi(t)\}\Psi(t), \tag{30}$$

where $\det\{\Phi(t)\}$ is the determinant and $\operatorname{adj}\{\Phi(t)\}$ is the adjugate of matrix $\Phi(t)$, we obtain a set of n equations of the form

$$\xi_l(t) = \zeta(t)\theta_l, \quad l = \overline{1,n}. \tag{31}$$

In the obtained first order regression models (31) we can identify parameters θ_l separately.

The standard gradient method can be used for identification of the obtained models with scalar regressor and parameter

$$\dot{\hat{\theta}}_l(t) = \gamma_d \zeta(t) \left(\xi_l(t) - \zeta(t) \hat{\theta}_l(t) \right), \tag{32}$$

where $\gamma_d \in \mathbb{R}_+$.

From (31) and (32) we can write

$$\dot{\tilde{\theta}}_l(t) = -\gamma_d \zeta^2(t) \tilde{\theta}_l(t). \tag{33}$$

Solving this differential equation we obtain

$$\tilde{\theta}_l(t) = \tilde{\theta}_l(0) \exp\left(-\gamma_d \int_0^t \zeta^2(\tau) d\tau \right). \tag{34}$$

If $\zeta(t)$ is bounded and not square-integrable function, *i.e.*

$$\zeta(t) \notin \mathcal{L}^2 \leftrightarrow \int_0^\infty \zeta^2(\tau) d\tau = \infty, \tag{35}$$

then (32) provides convergence of the estimation error to zero, *i.e.*

$$\lim_{t \to \infty} \left\| \theta_l - \hat{\theta}_l(t) \right\| = 0. \tag{36}$$

For exponential convergence, the following inequality should hold

$$\int_0^t \zeta^2(\tau) d\tau \geq Dt, \tag{37}$$

where $D \in \mathbb{R}_+$.

Nonlinear control systems

Nonlinear Control Systems

Zimenko Konstantin

Nonlinear versus linear systems

Linear systems

- Huge body of work in analysis and control of linear systems
- Most models currently available are linear (but most real systems are nonlinear...)

Idealization

Nonlinear systems

- Dynamics of linear systems are not rich enough to describe many commonly observed phenomena

Nonlinear systems can (sometime) be approximated by linear systems.
Nonlinear systems can (sometime) be "transformed" into linear systems.2

State-space model

State equation

$$\dot{x} = f(t, x, u)$$

Output equation

$$y = h(t, x, u)$$

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad u = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_m \end{bmatrix}, \quad f(t, x, u) = \begin{bmatrix} f_1(t, x, u) \\ f_2(t, x, u) \\ \vdots \\ f_n(t, x, u) \end{bmatrix}$$

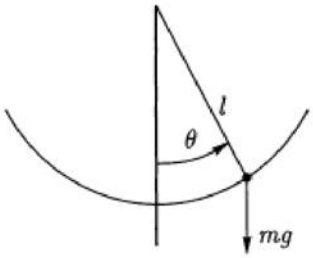
where $x \in \mathbb{R}^n$ is the state variable, $u \in \mathbb{R}^m$ is the input signal, and $y \in \mathbb{R}^q$ the output signal. The symbol $\dot{x} = \frac{dx}{dt}$ denotes the derivative of x with respect to time t.

3

Nonlinear systems: Example

Pendulum equation (equation of motion in the tangential direction)

$$ml\ddot{\theta} = -mg\sin\theta - kl\dot{\theta}.$$



State equations ($x_1 = \theta, x_2 = \dot{\theta}$)

$$\dot{x}_1 = x_2,$$
$$\dot{x}_2 = -\frac{g}{l}\sin x_1 - \frac{k}{m}x_2$$

Equilibrium points $(n\pi;0)$, $n = 0, \pm 1, \pm 2, \ldots$

4

Nonlinear systems: Example

State equations (frictional resistance is neglected)

$$\dot{x}_1 = x_2,$$
$$\dot{x}_2 = -\frac{g}{l}\sin x_1$$

Equilibrium points $(n\pi;0)$, $n = 0, \pm 1, \pm 2, \ldots$

State equations (with friction and applied torque)

$$\dot{x}_1 = x_2,$$
$$\dot{x}_2 = -\frac{g}{l}\sin x_1 - \frac{k}{m}x_2 + \frac{1}{ml^2}T$$

where T is the torque.

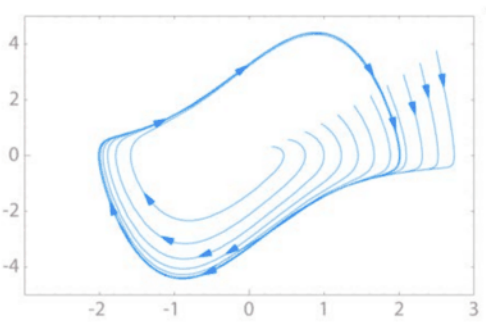
Equilibrium points $(\arcsin(T/mgl);0)$

5

Nonlinear systems: Example

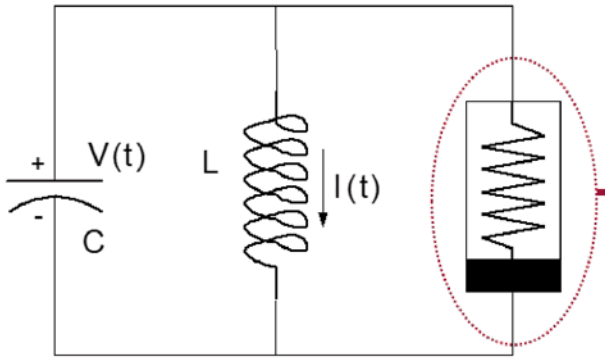
Robust oscillation

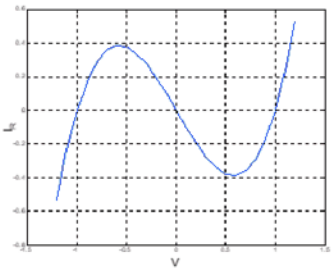
$$\dot{x}_1 = x_2,$$
$$\dot{x}_2 = -x_1 + \epsilon(1 - x_1^2)x_2$$



Van der Pol oscillator

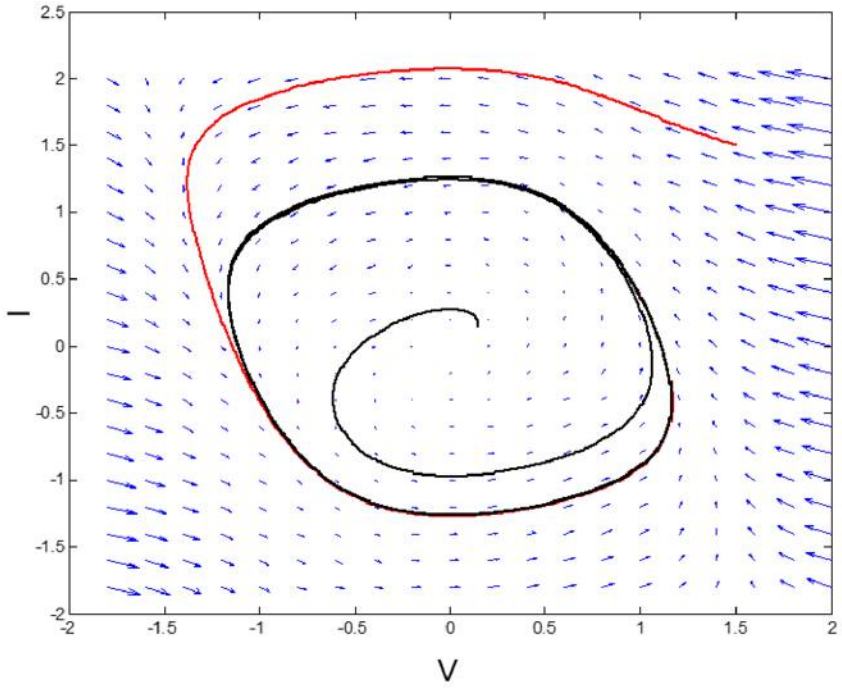
$$L\dot{I} = V$$
$$C\dot{V} = -I - I_R(V)$$





Nonlinear systems: Example

Van der Pol oscillator: phase portrait



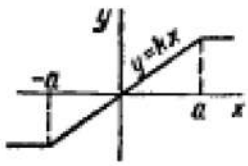
7

Nonlinear phenomena

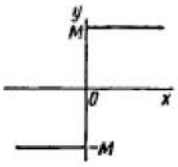
- Finite escape time
  (the state of unstable linear system goes to infinity as $t \to \infty$)
- Nonasimptotic stability (e.g. finite-time stability)
  (linear systems – infinite time of convergence)
- Multiple isolated equilibria
  (linear systems – only one isolated equilibrium point)
- Limit cycles
  (linear systems – system oscilates iff there is a pair of eigenvalues on the imaginary axis, which is a nonrobust condition)
- Subharmonic, harmonic, or almost-periodic oscillations
  (stable linear system under periodic input produces an output of the same frequency)
- Chaos
  (More complicated steady-state behavior)
- Multiple modes of behavior

8

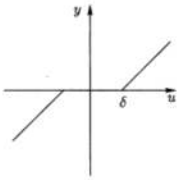
Common nonlinearities



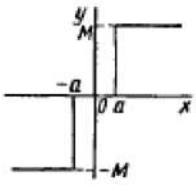
Saturation



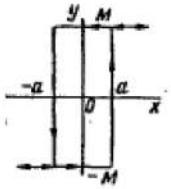
Relay



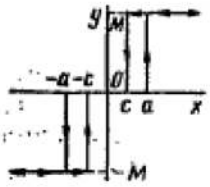
Dead zone



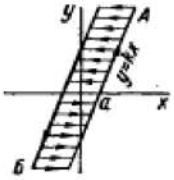
Relay with dead zone



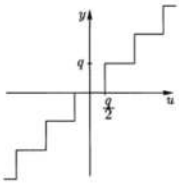
Relay with hysteresis



Three-position relay



Backslash



Quantization

9

Qualitative behavior of linear systems

Linear second order system

$$\dot{x} = Ax,\ x \in \mathbb{R}^2, A \in \mathbb{R}^{2\times 2}$$

Apply a similarity transformation M to A:

$$M^{-1}AM = J,\ M \in \mathbb{R}^{2\times 2}$$

where J is the real *Jordan form* of A, which depending on the eigenvalues of A may take one of the three forms

$$\begin{bmatrix}\lambda_1 & 0\\ 0 & \lambda_2\end{bmatrix}, \quad \begin{bmatrix}\lambda & k\\ 0 & \lambda\end{bmatrix}, \quad \begin{bmatrix}\alpha & -\beta\\ \beta & \alpha\end{bmatrix}$$

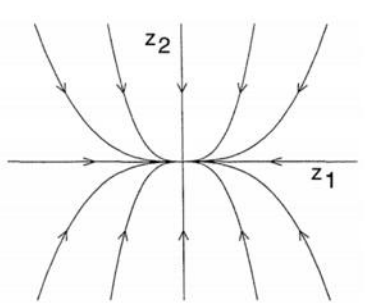
with k being either 0 or 1.

Present a change of coordinates

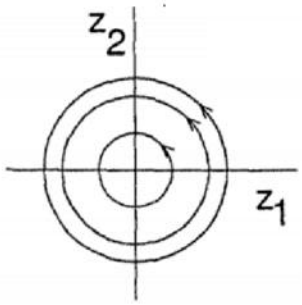
$$z = M^{-1}x$$
$$\dot{z} = M^{-1}\dot{x}$$

10

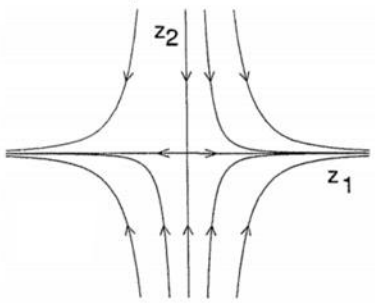
Qualitative behavior of linear systems



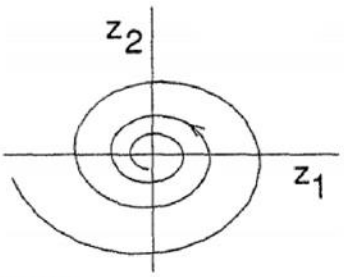
Stable node ($\lambda_{1,2} < 0$)



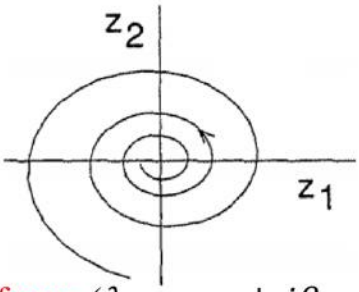
Center ($\lambda_{1,2} = \pm j\beta$)



Saddle point ($\lambda_2 < 0 < \lambda_1$)



Stable focus ($\lambda_{1,2} = a \pm j, a < 0$)

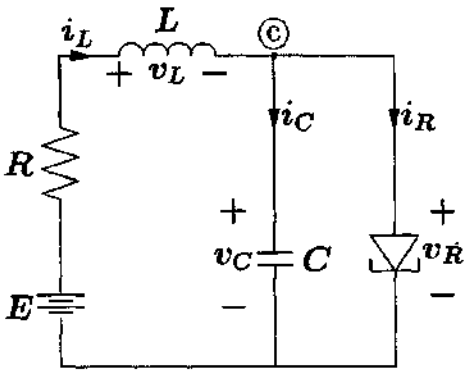


Stable focus ($\lambda_{1,2} = a \pm j\beta, a > 0$)

11

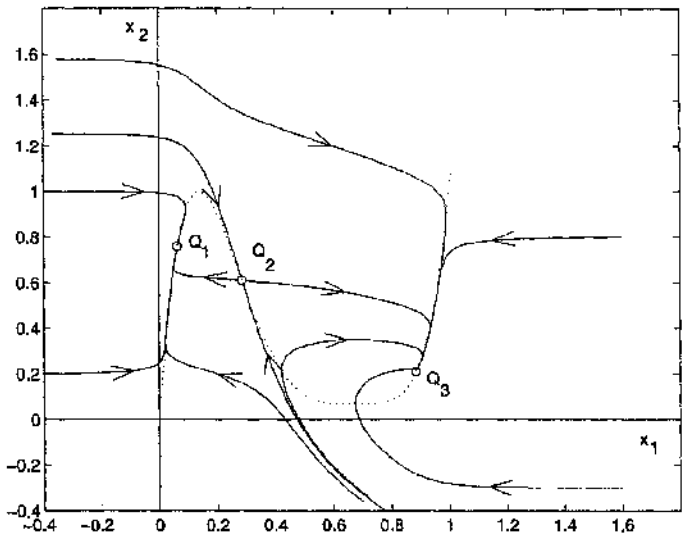
Multiple equilibrium points

Tunnel-diode circuit



$$\dot{x}_1 = \frac{1}{C}[-h(x_1) - x_2]$$

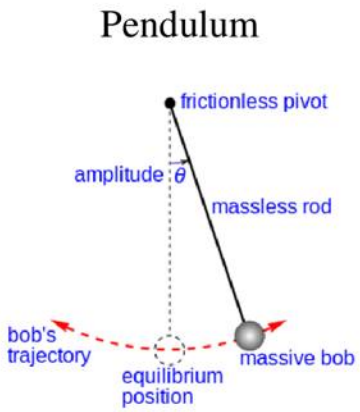
$$\dot{x}_2 = \frac{1}{L}[-x_1 - Rx_2 + u]$$

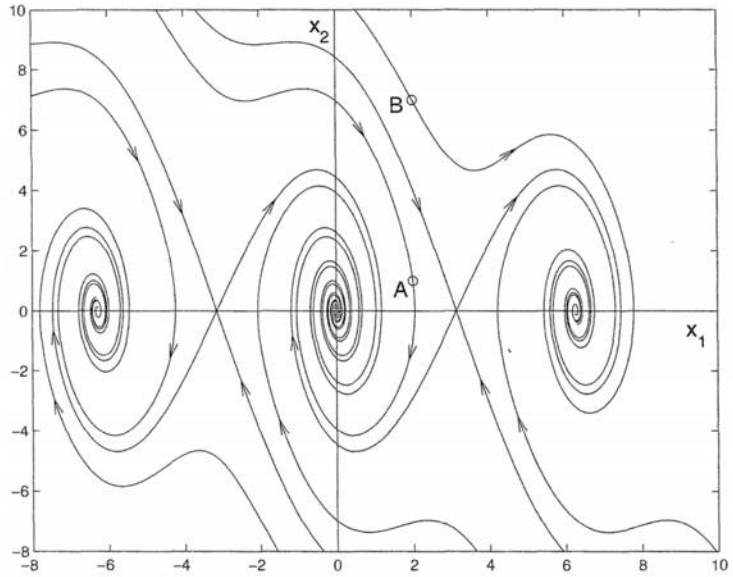


Phase portrait

12

Multiple equilibrium points





Phase portrait of the pendulum equation

13

Qualitative behavior near equilibrium

Consider autonomous system

$$\dot{x}_1 = f_1(x_1, x_2),$$
$$\dot{x}_2 = f_2(x_1, x_2)$$

where $f_1(x_1, x_2), f_2(x_1, x_2)$ are continuously differeniable.

Let $p = (p_1, p_2)$ is the equilibrium point. Expanding the right-hand side into its Taylor series about the point p, obtain

$$\dot{x}_1 = f_1(p_1, p_2) + a_{11}(x_1 - p_1) + a_{12}(x_2 - p_2) + HOT,$$
$$\dot{x}_2 = f_2(p_1, p_2) + a_{21}(x_1 - p_1) + a_{22}(x_2 - p_2) + HOT,$$

where HOT denotes high order terms and

$$a_{11} = \left.\frac{\partial f_1(x_1, x_2)}{\partial x_1}\right|_{x_1=p_1, x_2=p_2}, \quad a_{12} = \left.\frac{\partial f_1(x_1, x_2)}{\partial x_2}\right|_{x_1=p_1, x_2=p_2}$$
$$a_{21} = \left.\frac{\partial f_2(x_1, x_2)}{\partial x_1}\right|_{x_1=p_1, x_2=p_2}, \quad a_{22} = \left.\frac{\partial f_2(x_1, x_2)}{\partial x_2}\right|_{x_1=p_1, x_2=p_2}$$

Since $p = (p_1, p_2)$ is an equilibrium point

$$f_1(p_1, p_2) = f_2(p_1, p_2) = 0$$

14

Qualitative behavior near equilibrium

Define

$$y_1 = x_1 - p_1, \quad \text{и} \quad y_2 = x_2 - p_2$$

and rewrite the state equation as

$$\dot{y}_1 = \dot{x}_1 = a_{11}y_1 + a_{12}y_2 + HOT$$
$$\dot{y}_2 = \dot{x}_2 = a_{21}y_1 + a_{22}y_2 + HOT$$

HOT is negligible in a small neighborhood of equilibrium point:

$$\dot{y}_1 = \dot{x}_1 = a_{11}y_1 + a_{12}y_2$$
$$\dot{y}_2 = \dot{x}_2 = a_{21}y_1 + a_{22}y_2$$

Rewriting in a vector form, obtain

$$\dot{y} = Ay$$

where

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = \left[\begin{matrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{matrix} \right]\Bigg|_{x=p} = \frac{\partial f}{\partial x}\Big|_{x=p}$$

15

Example 1

Pendulum equation

$$\dot{x}_1 = x_2,$$
$$\dot{x}_2 = -10\sin x_1 - x_2.$$

Equilibrium points (0;0) и (π;0)

Jacobian

$$\frac{\partial f}{\partial x} = \begin{bmatrix} 0 & 1 \\ -10\cos x_1 & -1 \end{bmatrix}$$

Jacobian evaluated at the equilibrium point

$$A_1 = \begin{bmatrix} 0 & 1 \\ -10 & -1 \end{bmatrix}, \qquad \lambda_{1,2} = -0.5 \pm j3/12$$

$$A_2 = \begin{bmatrix} 0 & 1 \\ 10 & -1 \end{bmatrix}, \qquad \lambda_{1,2} = -3.7, 2.7$$

Equilibrium point (0;0) is a stable focus, equilibrium point (π;0) is a saddle point

16

Example 2

Consider the system

$$\dot{x}_1 = -x_2 - \mu x_1(x_1^2 + x_2^2)$$
$$\dot{x}_2 = x_1 - \mu x_2(x_1^2 + x_2^2)$$

Jacobian at (0;0) has eigenvalues $\pm j$.

Transition to polar coordinates:

$$x_1 = r\cos\theta \text{ и } x_2 = r\sin\theta$$

The system in polar coordinates

$$\dot{r} = -\mu r^3 \text{ и } \dot{\theta} = 1$$

For $\mu > 0$ the equilibrium point (0;0) is a stable focus, for $\mu < 0$ is a unstable focus.

17

Lyapunov function

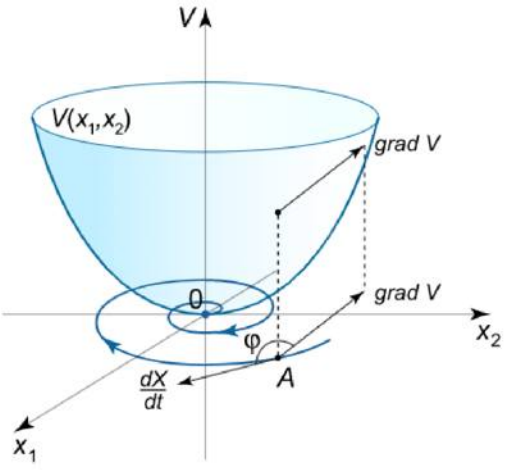
Consider the system

$$\dot{x} = f(x),$$

where $f: D \to R^n$ is locally Lipschitz

Let $p = 0$ is equilibrium point and $D \subset R^n$ is an open set, which contains p. Let $V : D \to R$ is a continuously differentiable function such, that

$$V(0) = 0 \text{ and } V(x) > 0 \text{ for } D \setminus \{0\}$$



If $\dot{V}(x) \le 0$ for $x \in D$, then $p = 0$ is stable.

If $\dot{V}(x) < 0$ for $x \in D \setminus \{0\}$, then $p = 0$ is asymptotically stable.

18

Lyapunov function

Consider the system

$$\dot{x} = f(x), f(0) = 0.$$

Expanding the right-hand side into its Taylor series

$$\dot{x} = f(0) + \left.\frac{\partial f}{\partial x}\right|_{x=0} x + g(x) = Ax + g(x),$$

where

$$A = \left.\frac{\partial f}{\partial x}\right|_{x=0}.$$

Choose the candidate Lyapunov function in the form

$$V(x) = x^T P x, \quad P > 0$$

Then

$$\dot{V}(x) = \dot{x}^T P x + x^T P \dot{x} = \left[x^T A^T + g^T(x)\right] P x + x^T P [Ax + g(x)] = x^T \left(A^T P + PA\right) x + 2x^T P g(x) =$$
$$= -x^T Q x + 2x^T P g(x),$$

where $Q>0$ such, that

$$A^T P + PA = -Q \qquad \text{Lyapunov equation}$$

19

Lyapunov function

Let

$$|g(x)| < \gamma |x|,$$

where $\gamma > 0$.

Since

$$x^T Q x \geq \lambda_{\min}(Q) x^T x = \lambda_{\min}(Q) |x|^2,$$

where $\lambda_{\min}(Q)$ is the smallest eigenvalue of the matrix Q, then

$$\dot{V}(x) \leq -\lambda_{\min}(Q) |x|^2 + 2\gamma \|P\| |x|^2 = -\left[\lambda_{\min}(Q) - 2\gamma \|P\|\right] x^2.$$

Lyapunov function derivative is negative if

$$\lambda_{\min}(Q) - 2\gamma \|P\| > 0 \Leftrightarrow \gamma < \frac{\lambda_{\min}(Q)}{2\|P\|}.$$

20

Example

Consider the system

$$\dot{x} = ax^3$$

Linearization:

$$A = \left.\frac{\partial f}{\partial x}\right|_{x=0} = 3ax^2\big|_{x=0} = 0.$$

Choose the Lyapunov function

$$V(x) = x^2.$$

Then

$$\dot{V}(x) = 2ax^4.$$

The equilibrium point is:

1) stable, if $a = 0$;
2) asymptotically stable, if $a < 0$;
3) unstable, if $a > 0$.

21

Stabilization: steady-state control

Consider the system

$$\dot{x} = f(x, u)$$

with desired equilibrium point $x = x_{\text{ss}}$

Steady-State Problem: Find steady-state control u_{ss} s.t.

$$0 = f(x_{\text{ss}}, u_{\text{ss}})$$

$$x_\delta = x - x_{\text{ss}}, \qquad u_\delta = u - u_{\text{ss}}$$

$$\dot{x}_\delta = f(x_{\text{ss}} + x_\delta, u_{\text{ss}} + u_\delta) \stackrel{\text{def}}{=} f_\delta(x_\delta, u_\delta)$$

$$f_\delta(0, 0) = 0$$

$$u_\delta = \gamma(x_\delta) \quad \Rightarrow \quad u = u_{\text{ss}} + \gamma(x - x_{\text{ss}})$$

22

State feedback stabilization

Nonlinear system

$$\dot{x} = f(x,u) \qquad [f(0,0) = 0]$$
$$u = \gamma(x) \qquad [\gamma(0) = 0]$$

Problem: stabilize the system at the origin

$$\dot{x} = f(x, \gamma(x))$$

where f and γ are locally Lipschitz functions

23

Stabilization: linearization approach

$$\dot{x} = Ax + Bu$$

$$A = \frac{\partial f}{\partial x}(x,u)\Big|_{x=0,u=0}; \quad B = \frac{\partial f}{\partial u}(x,u)\Big|_{x=0,u=0}$$

Closed-loop system:

$$\dot{x} = f(x, -Kx)$$

$$\dot{x} = \left[\frac{\partial f}{\partial x}(x,-Kx) + \frac{\partial f}{\partial u}(x,-Kx)\,(-K)\right]_{x=0} x$$
$$= (A - BK)x$$

$(A - BK)$ is Hurwitz $\Rightarrow$ the origin is an exponentially stable equilibrium point

24

Example: pendulum equation

$$\ddot{\theta} = -a \sin\theta - b\dot{\theta} + cT$$

Stabilize the pendulum at $\theta = \delta$

$$0 = -a \sin\delta + cT_{\text{ss}}$$

$$x_1 = \theta - \delta, \quad x_2 = \dot{\theta}, \quad u = T - T_{\text{ss}}$$

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -a[\sin(x_1 + \delta) - \sin\delta] - bx_2 + cu \end{aligned}$$

$$A = \left[\begin{array}{cc} 0 & 1 \\ -a\cos(x_1 + \delta) & -b \end{array} \right]_{x_1=0} = \left[\begin{array}{cc} 0 & 1 \\ -a\cos\delta & -b \end{array} \right]$$

25

Example: pendulum equation

$$A = \left[\begin{array}{cc} 0 & 1 \\ -a\cos\delta & -b \end{array} \right]; \quad B = \left[\begin{array}{c} 0 \\ c \end{array} \right]$$

$$K = \left[\begin{array}{cc} k_1 & k_2 \end{array} \right]$$

$$A - BK = \left[\begin{array}{cc} 0 & 1 \\ -(a\cos\delta + ck_1) & -(b + ck_2) \end{array} \right]$$

$$k_1 > -\frac{a\cos\delta}{c}, \quad k_2 > -\frac{b}{c}$$

$$T = \frac{a\sin\delta}{c} - Kx = \frac{a\sin\delta}{c} - k_1(\theta - \delta) - k_2\dot{\theta}$$

26

Feedback linearization

Consider the nonlinear system

$$\dot{x} = f(x) + G(x)u$$

$$f(0) = 0, \quad x \in R^n, \quad u \in R^m$$

Suppose there is a change of variables $z = T(x)$, defined for all $x \in D \subset R^n$, that transforms the system into the controller form

$$\dot{z} = Az + B\gamma(x)[u - \alpha(x)]$$

where (A, B) is controllable and $\gamma(x)$ is nonsingular for all $x \in D$

$$u = \alpha(x) + \gamma^{-1}(x)v \quad \Rightarrow \quad \dot{z} = Az + Bv$$

27

Feedback linearization

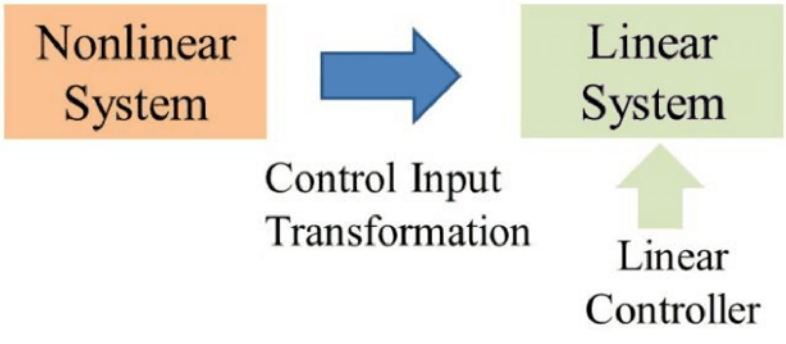
$$v = -Kz$$

Design K such that $(A - BK)$ is Hurwitz

$$u = \alpha(x) - \gamma^{-1}(x)KT(x)$$

Closed-loop system in the x-coordinates:

$$\dot{x} = f(x) + G(x)\left[\alpha(x) - \gamma^{-1}(x)KT(x)\right]$$



28

Feedback linearization

Closed-loop system:

$$u = \hat{\alpha}(x) - \hat{\gamma}^{-1}(x)K\hat{T}(x)$$

$$\dot{z} = (A - BK)z + B\delta(z)$$

$$\delta = \gamma[\hat{\alpha} - \alpha + \gamma^{-1}KT - \hat{\gamma}^{-1}K\hat{T}]$$

where $\hat{\alpha}, \hat{\gamma}, \hat{T}$ are nominal models of α, γ and T.

$V(z) = z^T Pz, \quad P(A - BK) + (A - BK)^T P = -I$

If $\|\delta(z)\| \leq k\|z\|$ for all z, where

$$0 \leq k < \frac{1}{2\|PB\|}$$

then the origin is globally exponentially stable

29

Example: pendulum equation

$$\ddot{\theta} = -a\sin\theta - b\dot{\theta} + cT$$

$$x_1 = \theta - \delta, \quad x_2 = \dot{\theta}, \quad u = T - T_{ss} = T - \frac{a}{c}\sin\delta$$

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -a[\sin(x_1 + \delta) - \sin\delta] - bx_2 + cu \end{aligned}$$

$$u = \frac{1}{c}\{a[\sin(x_1 + \delta) - \sin\delta] - k_1x_1 - k_2x_2\}$$

$$A - BK = \begin{bmatrix} 0 & 1 \\ -k_1 & -(k_2 + b) \end{bmatrix}$$ is Hurwitz

30

Example: pendulum equation

$$T = u + \frac{a}{c}\sin\delta = \frac{1}{c}\left[a\sin(x_1+\delta) - k_1x_1 - k_2x_2\right]$$

Let $\hat{a}$ and $\hat{c}$ be nominal models of a and c

$$T = \frac{1}{\hat{c}}\left[\hat{a}\sin(x_1+\delta) - k_1x_1 - k_2x_2\right]$$

$$\dot{x} = (A - BK)x + B\delta(x)$$

$$\delta(x) = \left(\frac{\hat{a}c - a\hat{c}}{\hat{c}}\right)\sin(x_1+\delta_1) - \left(\frac{c-\hat{c}}{\hat{c}}\right)(k_1x_1 + k_2x_2)$$

31

Example: pendulum equation

$$\delta(x) = \left(\frac{\hat{a}c - a\hat{c}}{\hat{c}}\right)\sin(x_1+\delta_1) - \left(\frac{c-\hat{c}}{\hat{c}}\right)(k_1x_1 + k_2x_2)$$

$$|\delta(x)| \le k\|x\| + \varepsilon$$

$$k = \left|\frac{\hat{a}c - a\hat{c}}{\hat{c}}\right| + \left|\frac{c-\hat{c}}{\hat{c}}\right|\sqrt{k_1^2+k_2^2}, \quad \varepsilon = \left|\frac{\hat{a}c - a\hat{c}}{\hat{c}}\right||\sin\delta_1|$$

$$P = \begin{bmatrix} p_{11} & p_{12} \\ p_{12} & p_{22} \end{bmatrix}, \quad PB = \begin{bmatrix} p_{12} \\ p_{22} \end{bmatrix}$$

$$k < \frac{1}{2\sqrt{p_{12}^2 + p_{22}^2}}$$

$$\sin\delta_1 = 0 \;\Rightarrow\; \varepsilon = 0$$

32

Backstepping

$$\begin{aligned}\dot{\eta} &= f(\eta) + g(\eta)\xi \\ \dot{\xi} &= u, \quad \eta \in R^n,\ \xi,\ u \in R\end{aligned}$$

Stabilize the origin using state feedback

View ξ as "virtual" control input to

$$\dot{\eta} = f(\eta) + g(\eta)\xi$$

Suppose there is $\xi = \phi(\eta)$ that stabilizes the origin of

$$\dot{\eta} = f(\eta) + g(\eta)\phi(\eta)$$

$$\frac{\partial V}{\partial \eta}[f(\eta) + g(\eta)\phi(\eta)] \le -W(\eta), \quad \forall\, \eta \in D$$

33

Backstepping

$$z = \xi - \phi(\eta)$$

$$\begin{aligned}\dot{\eta} &= [f(\eta) + g(\eta)\phi(\eta)] + g(\eta)z \\ \dot{z} &= u - \frac{\partial \phi}{\partial \eta}[f(\eta) + g(\eta)\xi]\end{aligned}$$

$$u = \frac{\partial \phi}{\partial \eta}[f(\eta) + g(\eta)\xi] + v$$

$$\begin{aligned}\dot{\eta} &= [f(\eta) + g(\eta)\phi(\eta)] + g(\eta)z \\ \dot{z} &= v\end{aligned}$$

34

Backstepping

$$V_c(\eta, \xi) = V(\eta) + \tfrac{1}{2} z^2$$

$$\begin{aligned} \dot{V}_c &= \frac{\partial V}{\partial \eta}[f(\eta) + g(\eta)\phi(\eta)] + \frac{\partial V}{\partial \eta} g(\eta) z + z v \\ &\leq -W(\eta) + \frac{\partial V}{\partial \eta} g(\eta) z + z v \end{aligned}$$

$$v = -\frac{\partial V}{\partial \eta} g(\eta) - kz, \quad k > 0$$

$$\dot{V}_c \leq -W(\eta) - kz^2$$

35

Backstepping

$$\begin{aligned} \dot{x} &= f_0(x) + g_0(x) z_1 \\ \dot{z}_1 &= f_1(x, z_1) + g_1(x, z_1) z_2 \\ \dot{z}_2 &= f_2(x, z_1, z_2) + g_2(x, z_1, z_2) z_3 \\ &\vdots \\ \dot{z}_{k-1} &= f_{k-1}(x, z_1, \ldots, z_{k-1}) + g_{k-1}(x, z_1, \ldots, z_{k-1}) z_k \\ \dot{z}_k &= f_k(x, z_1, \ldots, z_k) + g_k(x, z_1, \ldots, z_k) u \end{aligned}$$

$$g_i(x, z_1, \ldots, z_i) \neq 0 \quad \text{for } 1 \leq i \leq k$$

36

Example

$$\dot{x}_1 = x_1^2 - x_1^3 + x_2, \qquad \dot{x}_2 = u$$

$$\dot{x}_1 = x_1^2 - x_1^3 + x_2$$

$$x_2 = \phi(x_1) = -x_1^2 - x_1 \;\Rightarrow\; \dot{x}_1 = -x_1 - x_1^3$$

$$V(x_1) = \tfrac{1}{2}x_1^2 \;\Rightarrow\; \dot{V} = -x_1^2 - x_1^4, \quad \forall\, x_1 \in R$$

$$z_2 = x_2 - \phi(x_1) = x_2 + x_1 + x_1^2$$

$$\begin{aligned} \dot{x}_1 &= -x_1 - x_1^3 + z_2 \\ \dot{z}_2 &= u + (1 + 2x_1)(-x_1 - x_1^3 + z_2) \end{aligned}$$

37

Example

$$V_c(x) = \tfrac{1}{2}x_1^2 + \tfrac{1}{2}z_2^2$$

$$\begin{aligned} \dot{V}_c &= x_1(-x_1 - x_1^3 + z_2) \\ &\quad + z_2[u + (1 + 2x_1)(-x_1 - x_1^3 + z_2)] \end{aligned}$$

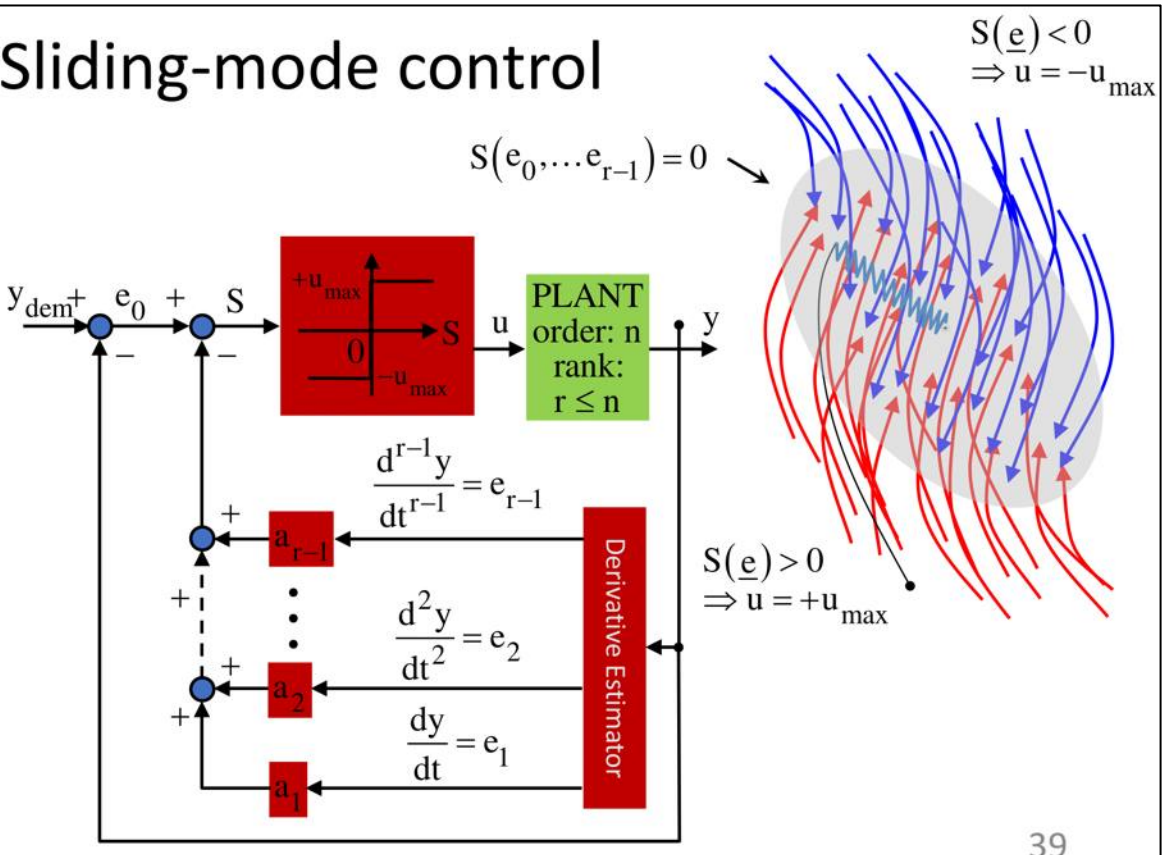
$$\begin{aligned} \dot{V}_c &= -x_1^2 - x_1^4 \\ &\quad + z_2[x_1 + (1 + 2x_1)(-x_1 - x_1^3 + z_2) + u] \end{aligned}$$

$$u = -x_1 - (1 + 2x_1)(-x_1 - x_1^3 + z_2) - z_2$$

$$\dot{V}_c = -x_1^2 - x_1^4 - z_2^2$$

$$\dot{V}_c = -x_1^2 - x_1^4 - (x_2 + x_1 + x_1^2)^2$$

Sliding-mode control



39

Sliding-mode control

$$\dot{x}_1 = x_2 \qquad \dot{x}_2 = h(x) + g(x)u, \quad g(x) \geq g_0 > 0$$

Sliding Manifold (Surface):

$$s = a_1 x_1 + x_2 = 0$$

$$s(t) \equiv 0 \;\Rightarrow\; \dot{x}_1 = -a_1 x_1$$

$$a_1 > 0 \;\Rightarrow\; \lim_{t \to \infty} x_1(t) = 0$$

40

Sliding-mode control

$$\dot{s} = a_1\dot{x}_1 + \dot{x}_2 = a_1 x_2 + h(x) + g(x)u$$

Suppose

$$\left|\frac{a_1 x_2 + h(x)}{g(x)}\right| \le \varrho(x)$$

$$V = \tfrac{1}{2}s^2$$

$$\dot{V} = s\dot{s} = s[a_1 x_2 + h(x)] + g(x)su \le g(x)|s|\varrho(x) + g(x)su$$

$$\beta(x) \ge \varrho(x) + \beta_0, \quad \beta_0 > 0$$

$$s > 0, \quad u = -\beta(x)$$

$$\dot{V} \le g(x)|s|\varrho(x) - g(x)\beta(x)|s|$$

$$\dot{V} \le g(x)|s|\varrho(x) - g(x)(\varrho(x) + \beta_0)|s| = -g(x)\beta_0|s|$$

41

Sliding-mode control

$$s < 0, \quad u = \beta(x)$$

$$\dot{V} \le g(x)|s|\varrho(x) + g(x)su = g(x)|s|\varrho(x) - g(x)\beta(x)|s|$$

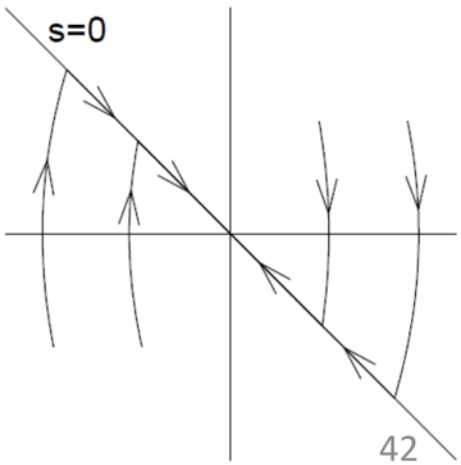
$$\dot{V} \le g(x)|s|\varrho(x) - g(x)(\varrho(x) + \beta_0)|s| = -g(x)\beta_0|s|$$

$$\operatorname{sgn}(s) = \begin{cases} 1, & s > 0 \\ -1, & s < 0 \end{cases}$$

$$u = -\beta(x)\operatorname{sgn}(s)$$

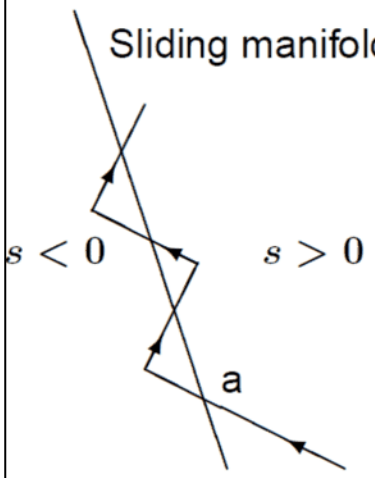
$$\dot{V} \le -g(x)\beta_0|s| \le -g_0\beta_0|s|$$

$$\dot{V} \le -g_0\beta_0\sqrt{2V}$$



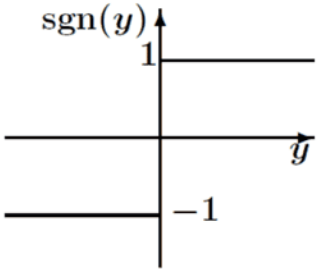
42

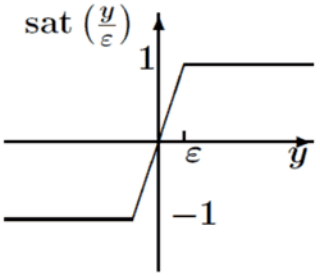
Sliding-mode control: chattering



$$u = -\beta(x)\ \mathrm{sat}\left(\frac{s}{\varepsilon}\right)$$

$$\mathrm{sat}(y) = \begin{cases} y, & \text{if } |y| \le 1 \\ \mathrm{sgn}(y), & \text{if } |y| > 1 \end{cases}$$





43

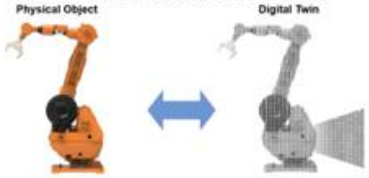
Digital twins

Cyber-physical systems Digital twins

Alexey Margun
alexeimargun@gmail.com

Digital twins

Digital Twin is a software analogue of a physical device that simulates internal processes, technical characteristics and behavior of a real object under the influence of the environment.



- Online copy of a real technical system (digital shadow)
- Offline modeling of technical systems

Digital twins

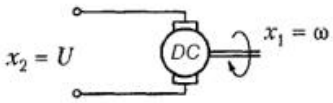
Problems:

- Unknown parameters
- Parameters changing
- Absence of sensors
- External noises and disturbances

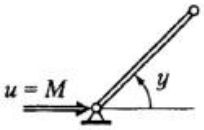
Possible solutions:

- Identification of unknown parameters
- Observers instead of sensors

Input-output model



$$T\dot{x}_1(t) + x_1(t) = kx_2(t)$$
$$x_1(t) = k\left(1 - e^{-t/T}\right)x_2$$



$$J\ddot{y}(t) = u(t)$$

Laplace transformation ($p = \frac{d}{dt}$): $\dot{x}_1 = px_1, \int x = \frac{x}{p}$.

Plant model:

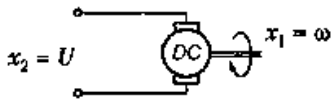
$$a(p)y(t) = b(p)u(t),$$

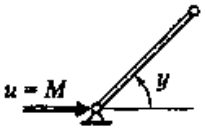
$a(p) = a_o p^n + a_1 p^{n-1} + \cdots + a_{n-1}p + a_n$ is a characteristic polynomial

$b(p) = b_o p^m + b_1 p^{m-1} + \cdots + b_{m-1}p + b_m$.

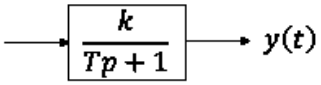
Input-output model

$y(t) = W(p)u(t), W(p) = \frac{b(p)}{a(p)}$ is a transfer function

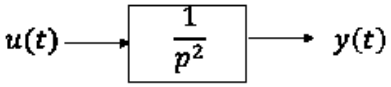




$$y(t) = \frac{k}{Tp+1}u(t)$$



$$y(t) = \frac{1}{p^2}u(t)$$



State space model

All linear differential equations ca be written as

$$\begin{aligned}
\dot{x}_1 &= a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n + b_1u,\\
\dot{x}_2 &= a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n + b_2u,\\
&\cdots\\
\dot{x}_n &= a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n + b_nu,\\
y(t) &= c_1x_1(t) + c_2x_2(t) + \cdots + c_nx_n(t).
\end{aligned}$$

In matrix representation:

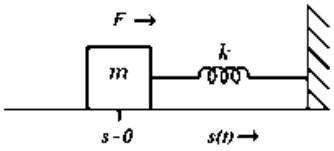
$$\dot{x} = Ax + Bu,$$
$$y = Cx$$

$$x = \begin{vmatrix} x_1 \\ \cdots \\ x_n \end{vmatrix}, A = \begin{vmatrix} a_{11} & \cdots & a_{1n} \\ \cdots & \cdots & \cdots \\ a_{n1} & \cdots & a_{nn} \end{vmatrix}, \mathrm{B} = \begin{vmatrix} b_1 \\ \cdots \\ b_n \end{vmatrix}, C = [c_1 \quad \cdots \quad c_n]$$

State space model. Example

Dynamic equations:

$$\dot{s} = v$$
$$m\dot{v} = F - ks - hv$$



State space model:

$$\dot{x} = Ax + Bu,$$
$$y = Cx$$

Let us choose state vector as $x = \begin{vmatrix} s \\ v \end{vmatrix}$.

$$A = \begin{vmatrix} 0 & 1 \\ -k/m & -h/m \end{vmatrix}, B = \begin{vmatrix} 0 \\ 1/m \end{vmatrix}, C = |1 \quad 0|.$$

$$x(t) = x_{free} + x_{forced} = e^{At}x_0 + \int_0^t e^{A(t-\tau)}Bu(\tau)d\tau$$

State space model. Change of coordinates

Consider new state vector:

$$x^* = Px$$

P is a transformation matrix, $\det P \neq 0$.

Inverse transformation:

$$x = P^{-1}x^*$$

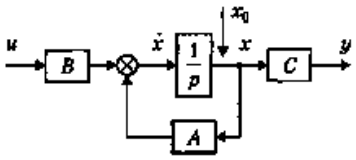
Model in new coordinates:

$$\begin{matrix} \dot{x}^* = A^*x^* + B^*u \\ y^* = C^*x^* \end{matrix}, A^* = PAP^{-1}, B^* = PB, C^* = CP^{-1}$$

Characteristic polynomial and poles of the system don't changes.

State space model

Modeling scheme



Transformation to input-output form:

$$W(p) = C(pI - A)^{-1}B,$$

$$I = \begin{vmatrix} 1 & 0 & 0 \\ 0 & \dots & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

Characteristic polynomial:

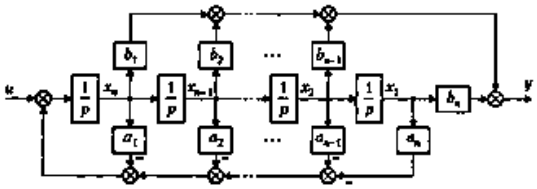
$$\det(pI - A) = 0$$

State space model

Transformation to state space model:

$$W(p) = \frac{b_1 p^{n-1} + \dots + b_{n-1}p + b_n}{p^n + a_1 p^{n-1} + \dots + b_{n-1}p + b_n}$$

Canonical controlled form:



$$\dot{x}_1 = x_2$$
$$\dot{x}_2 = x_3$$
$$\dots$$
$$\dot{x}_n = -a_n x_1 - a_{n-1} x_2 - \dots - a_1 x_n + u$$
$$y = b_n x_1 + b_{n-1} x_2 + \dots + b_1$$

$$A^* = \begin{vmatrix} 0 & | & I \\ a^T & & \end{vmatrix} = \begin{vmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 \\ -a_n & -a_{n-1} & \dots & -a_1 & \end{vmatrix}, B^* = \begin{vmatrix} 0 \\ 0 \\ \dots \\ 0 \\ 1 \end{vmatrix}, C^{*T} = \begin{vmatrix} b_n \\ b_{n-1} \\ \dots \\ b_2 \\ b_1 \end{vmatrix}$$

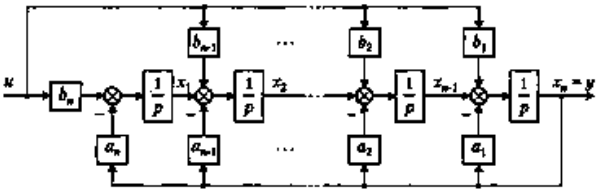
Transformation matrix: $P = U^* U^{-1}$, U, U^* **are controllability matrices of canonical and original model**

$$U = [B|\, AB|\, \dots\, |A^{n-1}B]$$

State space model

Transformation to state space model:

$$W(p) = \frac{b_1 p^{n-1} + \cdots + b_{n-1} p + b_n}{p^n + a_1 p^{n-1} + \cdots + b_{n-1} p + b_n}$$



Canonical observed form:

$$\dot{x}_1 = -a_n x_n + b_n u$$
$$\dot{x}_2 = x_1 - a_{n-1} x_n + b_n u$$
$$\dot{x}_n = x_{n-1} - a_1 x_n + b_1 u$$
$$y = x_n$$

$$A^* = \left|\begin{matrix} 0^T \\ \cdots \\ I \end{matrix}\right| a \Bigg| = \left|\begin{matrix} 0 & 0 & \ldots & 0 & -a_n \\ 1 & 0 & \ldots & 0 & -a_{n-1} \\ & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & \ldots & 0 & -a_2 \\ 0 & 0 & \ldots & 1 & -a_1 \end{matrix}\right|, B^* = \left|\begin{matrix} b_n \\ b_{n-1} \\ \ldots \\ b_2 \\ b_1 \end{matrix}\right|, C^{*T} = \left|\begin{matrix} 0 \\ 0 \\ \ldots \\ 0 \\ 1 \end{matrix}\right|$$

Transformation matrix: $P = (Q^*)^{-1} Q$, Q, Q^* are observability matrices of canonical and original model

$$Q^T = [C|\, CA|\, \ldots\, |CA^{n-1}]$$

Identification. Scalar example

Identification is a set of methods for constructing mathematical models of a dynamic systems from observational data.

Consider plant:

$$y(t) = \theta^* u(t)$$

$u(t)$ is a scalar input,
$y(t)$ is a scalar output,
θ^* is an unknown parameter.
The obvious solution :

$$\theta = \frac{y(t)}{u(t)}$$

Identification. Scalar example

Consider plant:

$$y(t) = \theta^* u(t)$$

$u(t)$ is a scalar input,
$y(t)$ is a scalar output,
θ^* is an unknown parameter.
The obvious solution :

$$\theta = \frac{y(t)}{u(t)}$$

Doesn't work if $u = 0$.
Hardly calculated if $u \to 0$.
High influence of noises.

Online estimation

Let θ is an estimate of θ^*.

Parallel model:

$$\hat{y}(t) = \theta u(t)$$

Error:

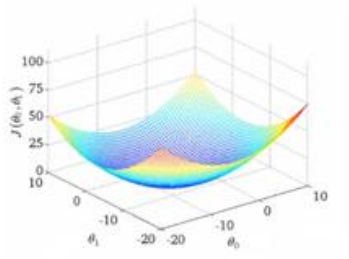
$$e = y - \hat{y} = y - \theta u$$

Consider functional:

$$J(\theta) = \frac{e^2}{2} = \frac{(y - \theta u)^2}{2}$$

Goal: minimize $J(\theta)$

Online estimation



Let denote:

$\nabla f(x) = \begin{bmatrix} \frac{\partial f(x)}{\partial x_1} \\ \cdots \\ \frac{\partial f(x)}{\partial x_n} \end{bmatrix}$ is a gradient of $f(x)$.

Lemma. If $J \in C^1$ and is convex on R^n than θ^* is a global minimum if

$$\nabla J(\theta^*) = 0$$

Therefore, we need to solve equation $\nabla J(\theta^*) = 0$ with respect to the θ^*

Gradient search. Discrete

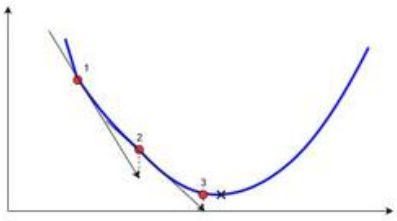
The search for the minimum is in the direction of reducing the function $d_k = -\nabla J(\theta_k)$

Identification algorithm:

$$\theta_{k+1} = \theta_k + \lambda_k d_k = \theta_k - \lambda_k \nabla J(\theta_k),$$
$$k = 0, 1, 2, \ldots$$

λ_k is a step size

θ_k is an estimate of θ on k-th step.



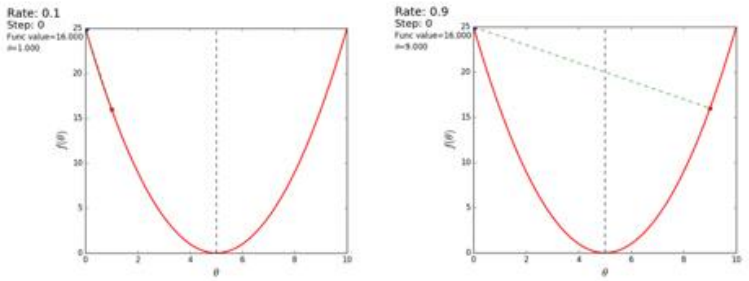
Gradient search. Discrete

Example:

$$y = (\theta - 5)^2$$

$$\frac{\partial y}{\partial \theta} = 2(\theta - 5)$$

Initial value $\theta = 0$.



Gradient search. Continuous

Rewrite algorithm as:

$$\frac{\theta_{k+1} - \theta_k}{\lambda_k} = -\nabla J(\theta)$$

If step is infinite small: $\lim_{\lambda_k \to 0} \frac{\theta_{k+1} - \theta_k}{\lambda_k} = \dot{\theta}$

Algorithm takes the form:

$$\dot{\theta} = -\gamma \nabla J(\theta)$$

$\gamma > 0$ is a coefficient that regulates convergence speed

For scalar case

$$\dot{\theta} = -\gamma \nabla J(\theta) = \gamma(y - \theta u)u = \gamma e u, \theta(0) = \theta_0$$

Gradient search. Continuous

Consider estimation error:

$$\tilde{\theta} = \theta^* - \theta$$

Error transient:

$$\tilde{\theta}(t) = e^{-\gamma \int_0^t u^2(\tau)d\tau}\tilde{\theta}(0)$$

If $u = 0$ or $u = e^{-t}$, $\tilde{\theta}(t)$ will not converges to zero.

If $u^2 = \frac{1}{1+t}$, $\tilde{\theta}(t)$ asymptotically converges to zero.

$\tilde{\theta}(t)$ exponentially converges to zero if persistent excitation condition holds:

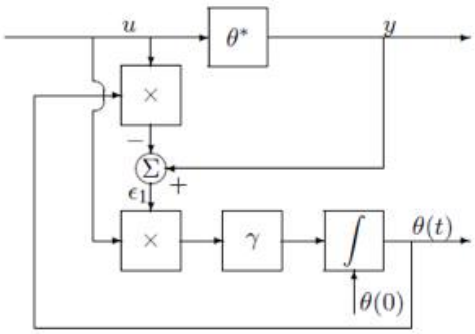
$$\int_t^{t+T_0} u^2(\tau)d\tau \geq \alpha_0 T_0, \forall t \geq 0$$

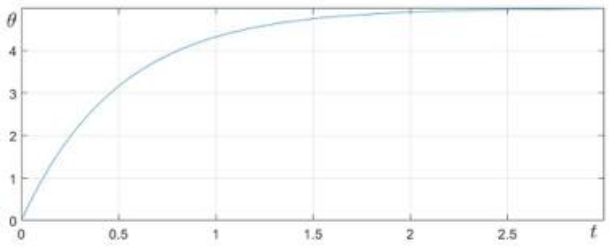
where $\alpha_0, T_0 > 0$

Gradient search. Example

Consider the system $y = \theta u, \theta = 5$

Identification algorithm: $\dot{\theta} = -\gamma \nabla J(\theta) = \gamma(y - \theta u)u = \gamma e u$
$\theta(0) = \theta_0, \gamma = 2$





Gradient search. Normalization

For system $y(t) = \theta^* u(t)$ with unbounded y and u problem

$$\min_\theta J = \min_\theta \frac{(y - \theta u)^2}{2}$$

can become hard for computing.

Solution is a normalization

$$\bar{y}(t) = \theta^* \bar{u}(t),$$

$$\bar{y}(t) = \frac{y}{m}, \bar{u}(t) = \frac{u}{m}, m^2 = 1 + u^2$$

Gradient search:

$$\dot{\theta} = \gamma e \bar{u}, \gamma > 0.$$

In origin coordinates:

$$\dot{\theta} = \frac{\gamma e u}{m^2}$$

Gradient search. Two unknown

Consider system

$$\dot{x} = -ax + bu, x(0) = x_0,$$

$$\dot{x} = \theta^T \phi, \theta = [a \quad b]^T, \phi = [-x \quad u]$$

where $a > 0$ and b are unknown constants to be identified.

Parallel model:

$$\dot{\hat{x}} = -\hat{a}\hat{x} + \hat{b}\hat{u}, \hat{x}(0) = \hat{x}_0$$

Error:

$$e = x - \hat{x}$$

Functional:

$$J(\theta) = \frac{e^2}{2}$$

$$\dot{\theta} = \gamma \nabla J(\theta), \dot{\hat{a}} = -\gamma_1 e x, \dot{\hat{b}} = \gamma_2 e u$$

Gradient search. Two unknown

If $\dot{x}$ is unmeasured.

Rewrite system:

$$\dot{x} = -a_m x + (a_m - a)x + bu \text{ или } x = \frac{1}{p+a_m}[(a_m - a)x + bu]$$

$a_m > 0$ is chosen by developer.

$$x = \theta^{*T}\phi,$$

$$\theta^* = [b, a_m - a]^T, \phi = \left[\frac{1}{p + a_m}u, \frac{1}{p + a_m}x\right]^T$$

Error:

$$e = x - \hat{x}$$

Serial-parallel model:

$$\dot{\hat{x}} = -a_m\hat{x} + (a_m - \hat{a})x + \hat{b}u \text{ или } \hat{x} = \frac{1}{p+a_m}\left[(a_m - \hat{a})x + \hat{b}u\right]$$

$$J = \frac{e^2}{2}$$

$$\theta = [a \quad b]^T$$

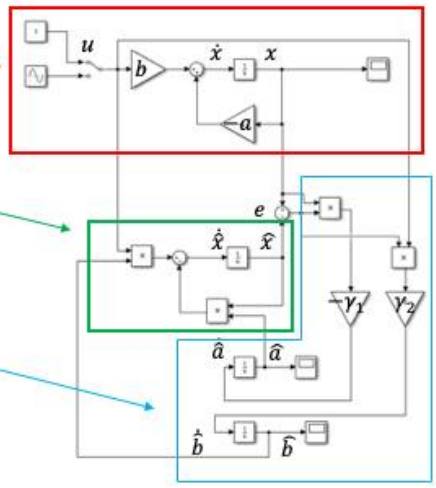
$$\dot{\theta} = \gamma\nabla J(\theta), \dot{\hat{a}} = -\gamma_1 ex, \dot{\hat{b}} = \gamma_2 eu$$

Gradient search. Two unknown

System: $\dot{x} = -ax + bu.$

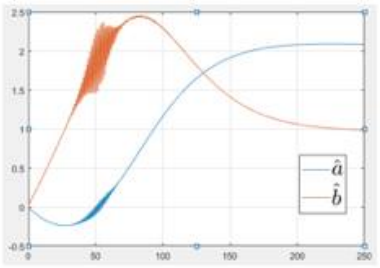
Parallel model: $\dot{\hat{x}} = -a\hat{x} + \hat{b}u.$

Identification algorithm:
$\dot{\hat{a}} = -\gamma_1 ex,$
$\dot{\hat{b}} = \gamma_2 eu$

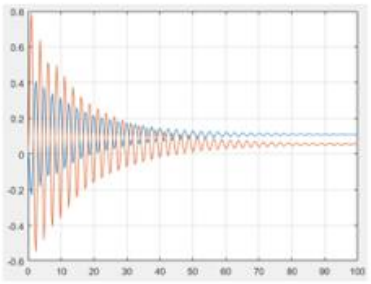


Gradient search. Two unknown

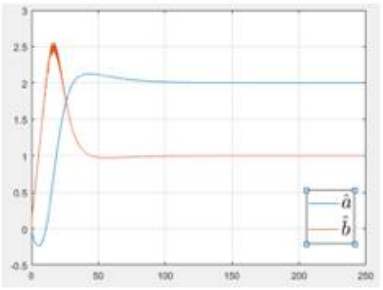
$u = \sin 5t$
$\gamma_{1,2} = 1$



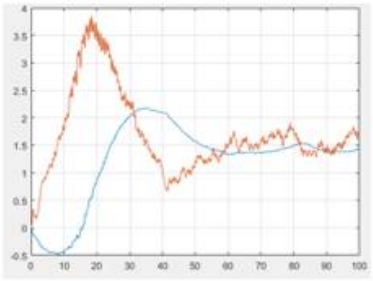
$u = 1(t)$
$\gamma_{1,2} = 5$



$\gamma_{1,2} = 5$



Белый шум



Gradient search. Linear dynamic system

Linear dynamic system:

$$\dot{x} = A \quad x + B \quad u$$
$$A \quad \in R^{nxn}, B \quad \in R^{n}, x \in R^{n}$$

Error:

$$e = x - \hat{x}$$

Functional:

$$J = \frac{e^T e}{2}$$

Parallel model:

$$\dot{\hat{x}} = \hat{A} \quad \hat{x} + \hat{B} \quad u, \hat{x} \in R^{n}$$
$$\dot{\hat{A}} = \gamma_1 e x^T, \dot{\hat{B}} = \gamma_2 e u^T$$

or serial-parallel model

$$\dot{x} = A_m x + (A \quad - A_m)x + B \quad u, A_m \in R^{nxn}$$
$$\dot{\hat{x}} = A_m \hat{x} + (\hat{A} \quad - A_m)\hat{x} + \hat{B} \quad u$$
$$\dot{\hat{A}} = \gamma_1 e x^T, \dot{\hat{B}} = \gamma_2 e u^T$$

System parametrization

Consider plant:

$$y^{(n)} + a_{n-1}y^{(n-1)} + \cdots + a_0 y = b_{n-1}u^{(n-1)} + b_{n-2}u^{(n-2)} + \cdots + b_0 u$$

Rewrite all parameters as vector

$$\theta^* = [b_{n-1}, b_{n-2}, \dots, b_0, a_{n-1}, a_{n-2}, \dots, a_0]^T$$

Rewrite input/output signals and their derivatives:

$$Y = \left[u^{(n-1)}, u^{(n-2)}, \dots, u, -y^{(n-1)}, -y^{(n-2)}, \dots, -y\right]^T =$$
$$= [\alpha_{n-1}^T(p)u, -\alpha_{n-1}^T(p)y]^T, \alpha_i(p) = \left[p^i, p^{i-1}, \dots, 1\right]^T$$

Therefore, we can rewrite system equation:

$$y^{(n)} = \theta^{*T} Y$$

System parametrization

If derivatives $y^{(n)} = \theta^{*T} Y$ are unmeasured

Apply stable filter $\frac{1}{\Lambda(p)}$ for both parts of equation, $\Lambda(p)$ is a Hurwitz polynomial:

$$z = \theta^{*T}\phi,$$

$$z = \frac{p^n}{\Lambda(p)}y, \phi = \left[\frac{\alpha_{n-1}^T(p)}{\Lambda(p)}u, -\frac{\alpha_{n-1}^T(p)}{\Lambda(p)}y\right]$$

$$\Lambda(p) = p^n + \lambda_{n-1}p^{n-1} + \cdots + \lambda_0$$

All signals of filtered model are measured.

System parametrization

Consider $\Lambda(p)$ as $\Lambda(p) = p^n + \lambda^T \alpha_{n-1}(p), \lambda = [\lambda^{n-1}, \dots, \lambda_0]^T$
In this case:

$$z = \frac{p^n}{\Lambda(p)} y = \frac{\Lambda(p) - \lambda^T \alpha_{n-1}(p)}{\Lambda(p)} y = y - \lambda^T \frac{\alpha_{n-1}(p)}{\Lambda(p)} y$$

$$y = z + \lambda^T \frac{\alpha_{n-1}(p)}{\Lambda(p)} y$$

$$z = \theta^{*T}\phi = \theta_1^{*T}\phi_1 + \theta_2^{*T}\phi_2, \theta_1^{*T} = [b_{n-1}, \dots, b_0], \theta_2^{*T} = [a_{n-1}, \dots, a_0],$$

$$\phi_1 = \frac{\alpha_{n-1}(p)}{\Lambda(p)} u, \phi_2 = -\frac{\alpha_{n-1}(p)}{\Lambda(p)} y$$

$$y = \theta_1^{*T}\phi_1 + \theta_2^{*T}\phi_2 - \lambda^T \phi_2$$

$$y = \theta_\lambda^{*T}\phi, \qquad \theta_\lambda^{*T} = [\theta_1^{*T}, \theta_2^{*T}, -\lambda^T]$$

State observers

Observer is an algorithm that allows to estimate the unmeasurable variables of the state vector.
Consider linear dynamic model:

$$\dot{x} = Ax + Bu,$$
$$y = C^T x$$

Parameters of system are known. Vector x is unmeasured.
If x_0 is known, than algorithm

$$\dot{\hat{x}} = A\hat{x} + Bu, \hat{x}(0) = x_0$$

provide $\hat{x}(t) = x(t) \forall t \geq 0$.

State observers

If x_0 is unknown and matrix A is stable we can use observer:

$$\dot{\hat{x}} = A\hat{x} + Bu, \hat{x}(0) = \hat{x}_0$$

Consider observation error:

$$\tilde{x} = x - \hat{x}$$

Its dynamics satisfy equation:

$$\dot{\tilde{x}} = A\tilde{x}, \tilde{x}(0) = x(0) - \hat{x}(0)$$

Solution of error dynamic equation:

$$\tilde{x}(t) = e^{At}\tilde{x}(0)$$

Because of A is stable $\tilde{x}$ exponentially converges to zero

Luenberger observer

If x_0 is unknown and matrix A is unstable or we need increase speed of convergence:

$$\dot{\hat{x}} = A\hat{x} + Bu + K(y - \hat{y}), \hat{x}(0) = \hat{x}_0,$$
$$\hat{y} = C^T\hat{x},$$

where K is chosen by developer.

Dynamics of estimation error:

$$\dot{\tilde{x}} = (A - KC^T)\tilde{x}, \tilde{x}(0) = x(0) - \hat{x}(0)$$

So;ution of error dynamics equation:

$$\tilde{x}(t) = e^{(A-KC^T)t}\tilde{x}(0)$$

By tuning K we ensure the stability of the error model and adjust its transient (overshoot, transient time, etc.)

Luenberger observer. Example

System:

$$\dot{x} = \begin{bmatrix} -4 & 1 \\ -4 & 0 \end{bmatrix} x + \begin{bmatrix} 1 \\ 3 \end{bmatrix} u, x(0) = \begin{bmatrix} 14 \\ 0{,}5 \end{bmatrix}$$
$$y = [1 \quad 0]x.$$

Luenberger observer

$$\dot{\hat{x}} = \begin{bmatrix} -4 & 1 \\ -4 & 0 \end{bmatrix} \hat{x} + \begin{bmatrix} 1 \\ 3 \end{bmatrix} u + \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} (y - \hat{y}),$$
$$\hat{y} = [1 \quad 0]\hat{x}.$$

Luenberger observer. Example

Denote $A_0 = A - KC^T = \begin{bmatrix} -4 & 1 \\ -4 & 0 \end{bmatrix} - \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} [1 \quad 0] = \begin{bmatrix} -4 - k_1 & 1 \\ -4 - k_2 & 0 \end{bmatrix}$.

Let we need speed of convergence faster than e^{-5t}.

In this case real part of A_0 eigenvalues should be less than -5.
Let $\lambda_1 = -6, \lambda_2 = -8$.

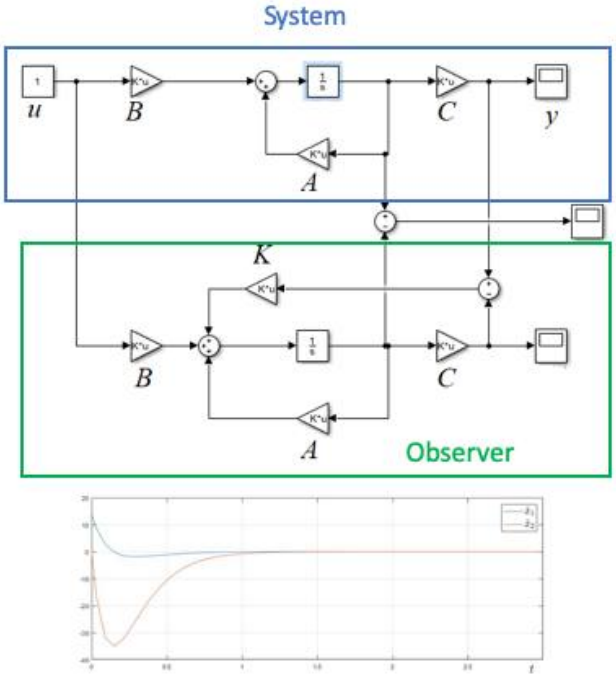
Therefore:

$$\det(pI - A_0) = p^2 + (4 + k_1)p + 4 + k_2 = (p + 6)(p + 8)$$

We can find:

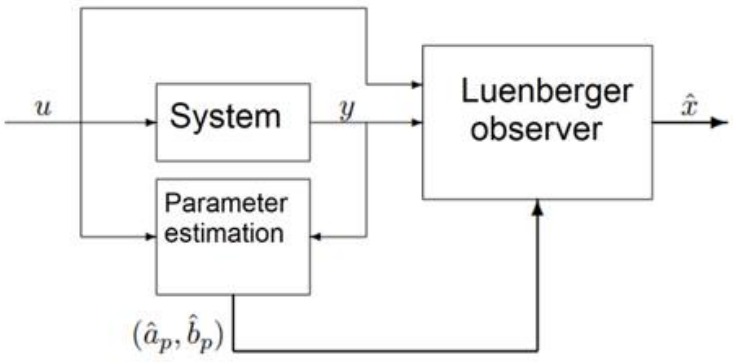
$$k_1 = 10, k_2 = 44$$

Luenberger observer. Example



Adaptive Luenberger observer

State vector is unmeasured.
System parameters are unknown.
Solution: simultaneously use the observer and the parameter estimation algorithm.



Adaptive Luenberger observer

In state space form we need to estimate $n^2 + 2n$ parameters.
In input output form we need to estimate $n + m + 1 \leq 2n$ parameters.
Obtain transfer function:

$$C^T(pI - A)^{-1}B = \frac{b_{n-1}p^{n-1} + \ldots + b_1 p + b_0}{p^n + a_{n-1}p^{n-1} + \cdots + a_0}$$

Rewrite system in canonical observable form:

$$\dot{x}_\alpha = \begin{bmatrix} & \vdots & I_{n-1} \\ -a_p & \vdots & \cdots \\ & \vdots & 0 \end{bmatrix} x_\alpha + b_p u, y = [1 \ 0 \ \ldots 0] x_\alpha$$

$$a_p = [a_{n-1}, a_{n-2}, \ldots, a_0]^T, b_p = [b_{n-1}, b_{n-2}, \ldots, b_0]^T$$

Adaptive Luenberger observer

Observer:

$$\dot{\hat{x}} = \hat{A}\hat{x} + \hat{b}_p u + K(y - \hat{y}), \hat{x}(0) = \hat{x}_0,$$
$$\hat{y} = [1 \ 0 \ldots 0]\hat{x},$$
$$\hat{A} = \begin{bmatrix} & \vdots & I_{n-1} \\ -\hat{a}_p & \vdots & \cdots \\ & \vdots & 0 \end{bmatrix}, K = a^* - \hat{a}_p$$

a^* is chosen such that

$$A^* = \begin{bmatrix} & \vdots & I_{n-1} \\ -a^* & \vdots & \cdots \\ & \vdots & 0 \end{bmatrix}$$

is stable, i.e. roots of $\det(pI - A^*) = 0$ have negative real part.

Digital twins

It is necessary for development of digital twin:

- Build mathematical model of system
- Estimate unknown parameters with identification algorithm
- Build observer for state vector estimation
- Run obtained model in real time with the same input signal as a real system

Modeling of systems and complexes

Modeling and control of robotic systems

Kinematics of industrial robots

Kinematics of Industrial Robots

Dr. Oleg Borisov

Basic Concepts and Definitions: Joints and Generalized Coordinates

> **Kinematic Chain**
> The *kinematic chain* is used to describe the geometry of the robot manipulator. Ot represents a graphic representation of the sequence of manipulator links connected by joints.

There are two elementary types of 1-DOF joints

- revolte (joint coordinate is angular)
- prismatic (joint coordinat is linear)

Both joint coordinates are so-called *generalized coordinates*

$$q_i = \begin{cases} \theta_i, & \text{if the link } i \text{ is revolute,} \\ d_i, & \text{if the link } i \text{ is prisnatic.} \end{cases} \tag{1}$$

> **Configuration**
> A set of all the generalized coordinates of the manipulator, which uniquely determines it in the space, is called *configuration.*

Basic Concepts and Definitions: FK and IK

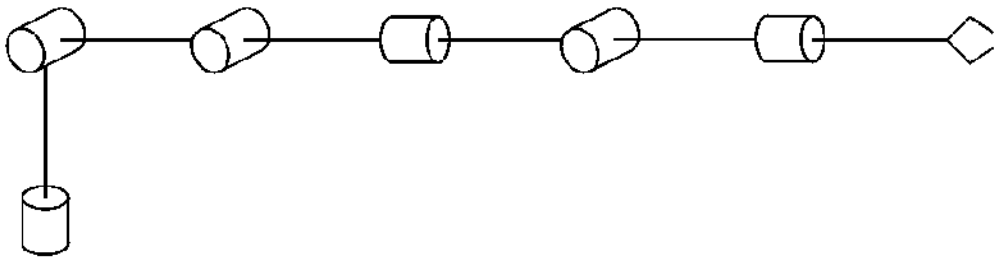
There are two fundamental tasks of the kinematics analysis

- forward kinematics
- inverse kinematics

Forward kinematics
The *forward kinematics (FK)* is to calculate the coordinates of the tool frame (its position and orientation) given the configuration of the robot.

Inverse kinematics
The *inverse kinematics (IK)* is to calculate the configuration of the robot given the coordinates of the tool frame (its position and orientation).

Forward Kinematics: Algorithm



Kinematic chain of 6-DOF robot

1. Assigning frames to the links.
2. Determining Denavit-Hartenberg parameters
3. Forming homogeneous transformation matrices
4. Parametrization of rotation matrix

Forward Kinematics: Assigning Frames

Choice of z_i-axes

Choose the axis z_i so that it coincides with the axis of rotation or translational motion of the subsequent joint $i+1$ depending on its type. This means that the relative location of adjacent links (coordinate systems) will be determined precisely by the variable around (or along) this axis.

Choice of x_i-axes

Choose the axis x_i, $i = \{1, 2 \ldots, n-1\}$ so that the following two conditions are satisfied.

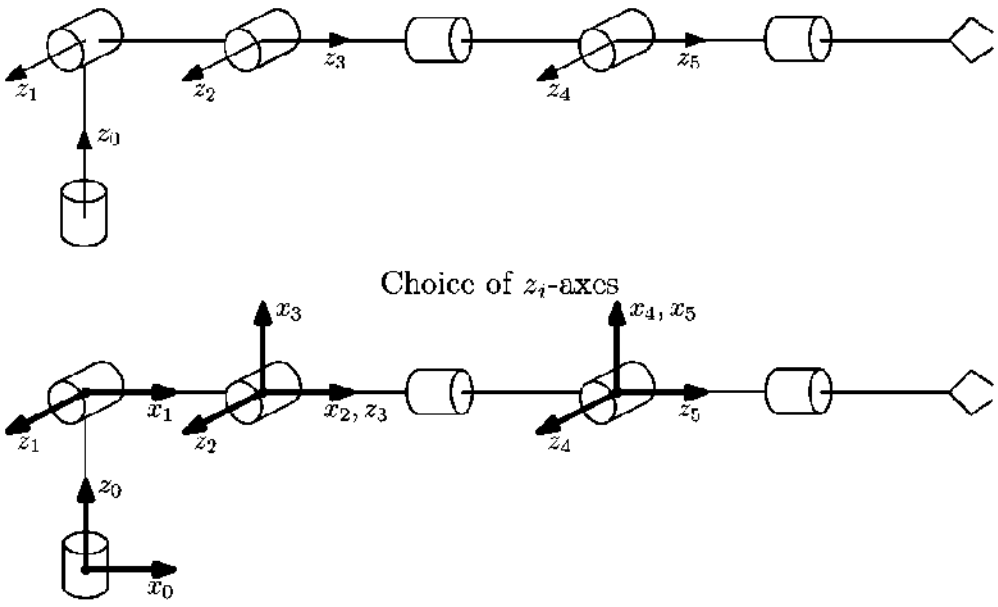
- The axis x_i is perpendicular to the axis z_{i-1}.
- The axis x_i intersects the axis z_{i-1}.

Choice of y_i-axes

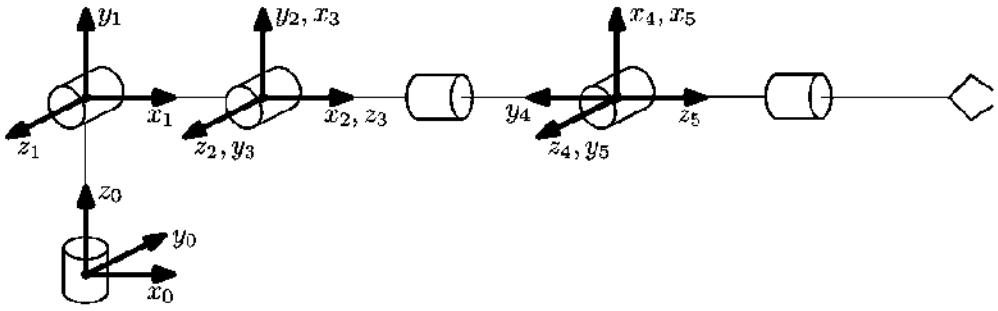
Choose the axis y_i so that the frame given by the unit vectors $\vec{x}_i$, $\vec{y}_i$, $\vec{z}_i$ is right-handed, i.e. in the direction given by the vector product:

$$\vec{y}_i = \vec{z}_i \times \vec{x}_i. \tag{2}$$

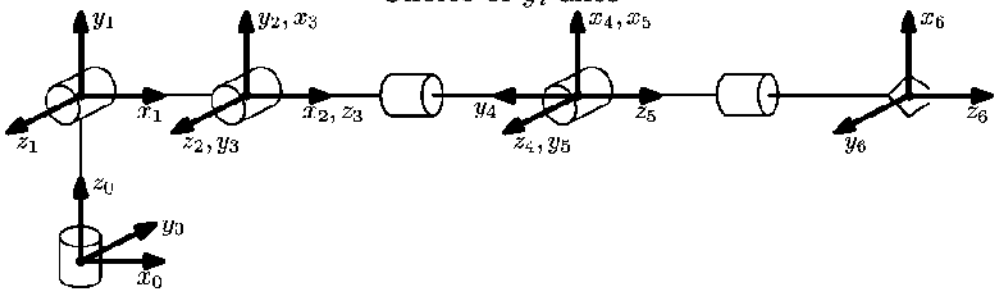
Forward Kinematics: Assigning Frames



Forward Kinematics: Assigning Frames



Choice of y_i-axes



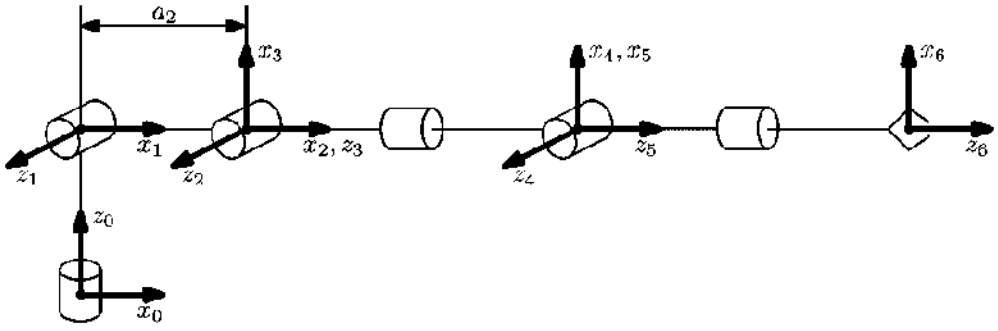
Choice of n-axes

Forward Kinematics: DH parameters

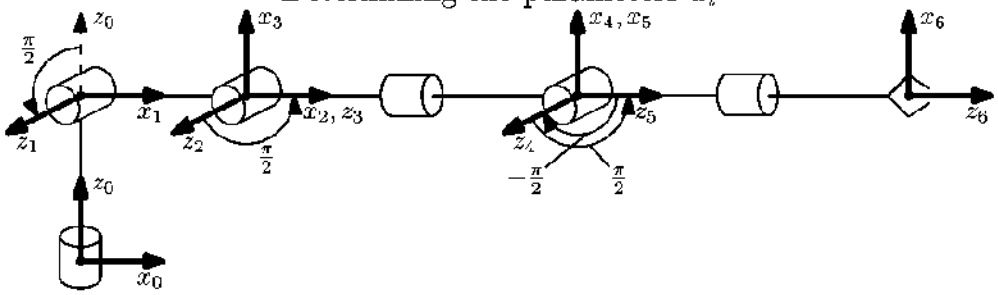
The Denavit-Hartenberg convention allows to reduce the number of coordinates that uniquely determine the body (its frame) in the space, from six to four, known as the *Denavite-Harteberg parameters* listed below.

- a_i is the distance along the axis x_i from z_{i-1} to z_i
- α_i is the angle around the axis x_i from z_{i-1} to z_i
- d_i is the distance along the axis z_{i-1} from x_{i-1} to x_i
- θ_i is the angle around the axis z_{i-1} from x_{i-1} to x_i

Forward Kinematics: DH parameters

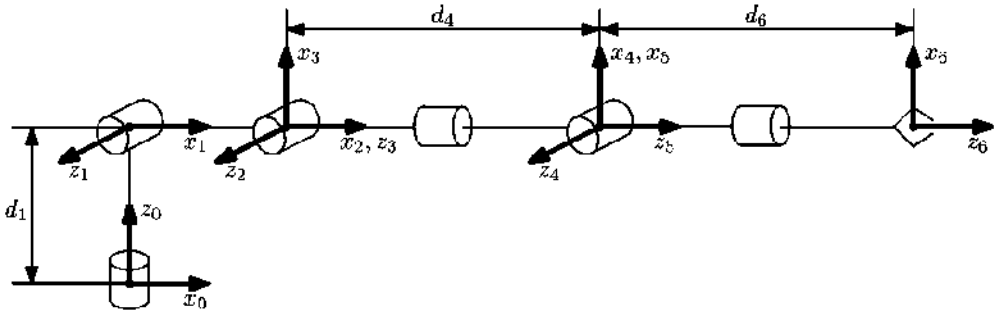


Determining the parameters a_i

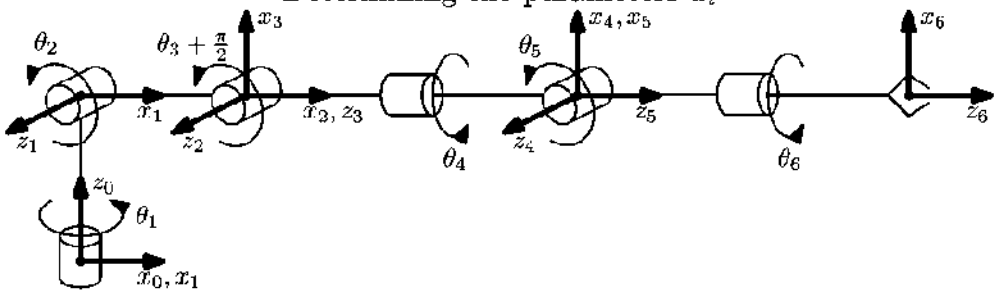


Determining the parameters α_i

Forward Kinematics: DH parameters



Determining the parameters d_i



Determining the parameters θ_i

Forward Kinematics: DH parameters

Link, i	a_i	α_i	d_i	θ_i
1	0	$\frac{\pi}{2}$	d_1	θ_1
2	a_2	0	0	θ_2
3	0	$\frac{\pi}{2}$	0	$\theta_3 + \frac{\pi}{2}$
4	0	$-\frac{\pi}{2}$	d_4	θ_4
5	0	$\frac{\pi}{2}$	0	θ_5
6	0	0	d_6	θ_6

DH parameters of the 6-DOF robot

Forward Kinematics: HT Matrix

Consider to sets of coordinates k^0 and k^n of the same point in the space expressed with respect to two frames $o_0x_0y_0z_0$ and $o_nx_ny_nz_n$, respectively:

$$k^0 = T_n^0 k^n, \tag{3}$$

where T_n^0 is the transformation carrying information about relative position and orientation of one frame with respect to another one.

Homogeneous Transformation Matrixe

The matrix T_n^0 defining the relation between frames $o_0x_0y_0z_0$ and $o_nx_ny_nz_n$ is called *a homogeneous transformation (HT) matrix* and has the form

$$T_n^0 = \begin{bmatrix} n_x & s_x & a_x & p_x \\ n_y & s_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} n_n^0 & s_n^0 & a_n^0 & p_n^0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} R_n^0 & p_n^0 \\ 0 & 1 \end{bmatrix}, \tag{4}$$

where the vectors n_n^0, s_n^0 and a_n^0 express directions of x_n, y_n and z_n with respect to $o_0x_0y_0z_0$, $R_n^0 \in \mathcal{SO}(3)$ is the rotation matrix of the frame $o_nx_ny_nz_n$ with respect to $o_0x_0y_0z_0$, $p_n^0 \in \mathbb{R}^3$ is the vector of linear displacement of the origin of $o_nx_ny_nz_n$ with respect to $o_0x_0y_0z_0$.

Forward Kinematics: Properties of HT Matrix

1. The rotation by zero angle is determined by the identity matrix

$$R_{\beta=0} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = I. \tag{5}$$

2. Rotation in the negative direction is determined by

$$R_{-\beta} = R_{\beta}^{-1} = R_{\beta}^{T}. \tag{6}$$

3. There are three basic rotation matrices around x, y and z axes given as

$$R_{x,\beta} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\beta & -\sin\beta \\ 0 & \sin\beta & \cos\beta \end{bmatrix}, \; R_{y,\beta} = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}, \; R_{z,\beta} = \begin{bmatrix} \cos\beta & -\sin\beta & 0 \\ \sin\beta & \cos\beta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

where β is some angle.

Forward Kinematics: Properties of HT Matrix

4. Serial rotations around several *current* axes are determined by multiplying on the right. For example, the transformation parametrized by Euler angles ϕ, θ and ψ is given as

$$\begin{aligned} R_{zyz} &= R_{z,\phi} R_{y,\theta} R_{z,\psi} = \\ &= \begin{bmatrix} c_\phi & -s_\phi & 0 \\ s_\phi & c_\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix} \begin{bmatrix} c_\psi & -s_\psi & 0 \\ s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} = \\ &= \begin{bmatrix} c_\phi c_\theta c_\psi - s_\phi s_\psi & -c_\phi c_\theta s_\psi - s_\phi c_\psi & c_\phi s_\theta \\ s_\phi c_\theta c_\psi + c_\phi s_\psi & -s_\phi c_\theta s_\psi + c_\phi c_\psi & s_\phi s_\theta \\ -s_\theta c_\psi & s_\theta s_\psi & c_\theta \end{bmatrix}, \end{aligned} \tag{7}$$

where $c_\beta \equiv \cos\beta$, $s_\beta \equiv \sin\beta$, $\beta = \{\phi, \theta, \psi\}$.

Forward Kinematics: Properties of HT Matrix

Using the DH convention form the homogeneous transformation matrices for each link as follows

$$T_i = T_{z,\theta_i} T_{z,d_i} T_{x,a_i} T_{x,\alpha_i} = \begin{bmatrix} R_{z,\theta_i} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} I & p_{d_i} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} I & p_{a_i} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R_{x,\alpha_i} & 0 \\ 0 & 1 \end{bmatrix} =$$

$$= \begin{bmatrix} c_{\theta_i} & -s_{\theta_i} c_{\alpha_i} & s_{\theta_i} s_{\alpha_i} & a_i c_{\theta_i} \\ s_{\theta_i} & c_{\theta_i} c_{\alpha_i} & -c_{\theta_i} s_{\alpha_i} & a_i s_{\theta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{8}$$

where i is the link number, R_{z,θ_i} and R_{x,α_i} are the basic rotation matrices, p_{d_i} and p_{a_i} are vectors with nonzero components $p_z = d_i$ and $p_x = a_i$

$$R_{z,\theta_i} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 \\ \sin\theta_i & \cos\theta_i & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad R_{x,\alpha_i} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha_i & -\sin\alpha_i \\ 0 & \sin\alpha_i & \cos\alpha_i \end{bmatrix},$$

$$p_{d_i} = \begin{bmatrix} 0 \\ 0 \\ d_i \end{bmatrix}, \qquad p_{a_i} = \begin{bmatrix} a_i \\ 0 \\ 0 \end{bmatrix}. \tag{9}$$

Forward Kinematics: Parametrization of Rotation Matrices

There different ways to parametrize rotation matrices

- Euler angles
- Roll-Pitch-Yaw angles
- Axis-Angle Representation

All of them are intended to reduce amount of parameters from 9 to 3.

Forward Kinematics: Euler Angles

The matrix of ZYZ-transformation is given as

$$R_n^0(q) = \begin{bmatrix} r_{11}(q) & r_{12}(q) & r_{13}(q) \\ r_{21}(q) & r_{22}(q) & r_{23}(q) \\ r_{31}(q) & r_{32}(q) & r_{33}(q) \end{bmatrix} = \begin{bmatrix} c_\phi c_\theta c_\psi - s_\phi s_\psi & -c_\phi c_\theta s_\psi - s_\phi c_\psi & c_\phi s_\theta \\ s_\phi c_\theta c_\psi + c_\phi s_\psi & -s_\phi c_\theta s_\psi + c_\phi c_\psi & s_\phi s_\theta \\ -s_\theta c_\psi & s_\theta s_\psi & c_\theta \end{bmatrix}. \quad (10)$$

Consider three cased depending on the entry $r_{33}(q)$.

First Case

If $r_{33}(q) \neq \pm 1$ then $\sin\theta(q) \neq 0$. Use the Pythagorean trigonometric identity

$$\sin^2\theta(q) + \cos^2\theta(q) = 1, \quad (11)$$

$$\sin(\theta(q)) = \pm\sqrt{1 - \cos^2\theta(q)} = \pm\sqrt{1 - r_{33}(q)}\ , \quad (12)$$

from which it follows that $\theta(q)$ can be calculated as

$$\theta(q) = \operatorname{atan2}\left(\pm\sqrt{1 - r_{33}^2(q)}, r_{33}(q)\right). \quad (13)$$

Note that the remaining expressions to calculate $\phi(q)$ and $\psi(q)$ depend on the choice of the sign in front of the root in (13)

$$\phi(q) = \operatorname{atan2}\left(\pm r_{23}(q), \pm r_{13}(q)\right), \quad (14)$$

$$\psi(q) = \operatorname{atan2}\left(\pm r_{32}(q), \mp r_{31}(q)\right). \quad (15)$$

Second Case

If $r_{33}(q) = 1$ then $\cos\theta(q) = 1$, $\sin\theta(q) = 0$, from which $\theta(q) = 0$ and as a result

$$\begin{aligned} R_n^0(q) &= \begin{bmatrix} c_\phi c_\psi - s_\phi s_\psi & -c_\phi s_\psi - s_\phi c_\psi & 0 \\ s_\phi c_\psi + c_\phi s_\psi & s_\phi s_\psi + c_\phi c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} = \\ &= \begin{bmatrix} c_{\phi+\psi} & -s_{\phi+\psi} & 0 \\ s_{\phi+\psi} & c_{\phi+\psi} & 0 \\ 0 & 0 & 1 \end{bmatrix} = \\ &= \begin{bmatrix} r_{11}(q) & r_{12}(q) & 0 \\ r_{21}(q) & r_{22}(q) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \end{aligned} \tag{16}$$

This case leads to uncertainty, since only the sum $\phi(q) + \psi(q)$ can be computed

$$\phi(q) + \psi(q) = \operatorname{atan2}\left(r_{21}(q), r_{11}(q)\right). \tag{17}$$

Third Case

If $r_{33}(q) = -1$ then $\cos\theta(q) = -1$, $\sin\theta(q) = 0$, from which $\theta(q) = \pi$, as a result

$$\begin{aligned} R_n^0(q) &= \begin{bmatrix} -c_\phi c_\psi - s_\phi s_\psi & c_\phi s_\psi - s_\phi c_\psi & 0 \\ -s_\phi c_\psi + c_\phi s_\psi & s_\phi s_\psi + c_\phi c_\psi & 0 \\ 0 & 0 & -1 \end{bmatrix} = \\ &= \begin{bmatrix} -c_{\phi-\psi} & -s_{\phi-\psi} & 0 \\ s_{\phi-\psi} & c_{\phi-\psi} & 0 \\ 0 & 0 & -1 \end{bmatrix} = \\ &= \begin{bmatrix} r_{11}(q) & r_{12}(q) & 0 \\ r_{21}(q) & r_{22}(q) & 0 \\ 0 & 0 & -1 \end{bmatrix}. \end{aligned} \tag{18}$$

This case leads to uncertainty, since only the difference $\phi(q) \quad \psi(q)$ can be computed

$$\phi(q) - \psi(q) = \operatorname{atan2}\left(-r_{12}(q), -r_{11}(q)\right). \tag{19}$$

Inverse Kinematics

Initial data for IK are

- three linear coordinates (components of the vector p_n^0)
- three angular coordinates (e.g. Euler angles ϕ, ϕ and ψ)
- DH parameters

The geometric (analytical) method of solving IK is to find explicit expressions using the apparatus of trigonometric functions, taking into account the kinematic scheme of the manipulator.

Consider kinematic decoupling approach applied to standard 6-DOF robot with spherical wrist. It is comprised of two subtasks

- position IK (to compute q_1, q_2 and q_3)
- orientation IK (to compute q_4, q_5 and q_6)

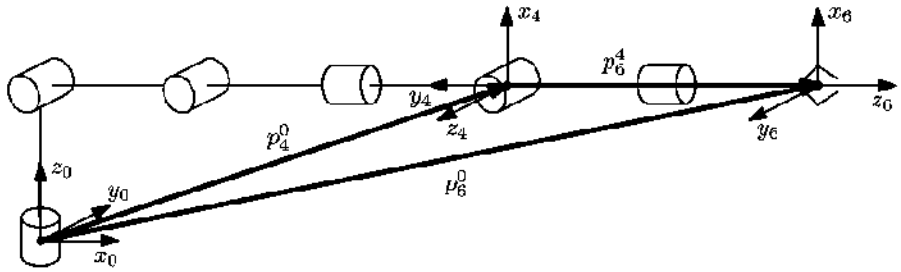
Inverse Kinematics: Position IK

> **Spherical wrist**
> A spherical wrist is a kinematic scheme of the last three rotational joints such that their axes of rotation intersect at the same point.

The subtask is

- to determine relations between the given point of the end-effector and the point of three axes intersection
- to derive expressions for q_1, q_2 and q_3 given the point of three axes intersection

Inverse Kinematics: Position IK



Kinematic decoupling

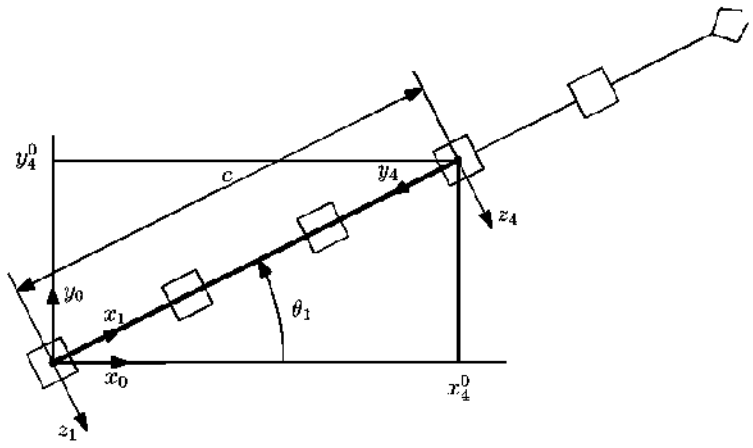
Using the sum of the vectors

$$p_6^0 = p_4^0 + d_6 R_6^0 \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \tag{20}$$

express coordinates of the point as

$$p_4^0 = p_6^0 - d_6 R_6^0 \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} x_4^0 \\ y_4^0 \\ z_4^0 \end{bmatrix}. \tag{21}$$

Inverse Kinematics: Position IK



Vector c

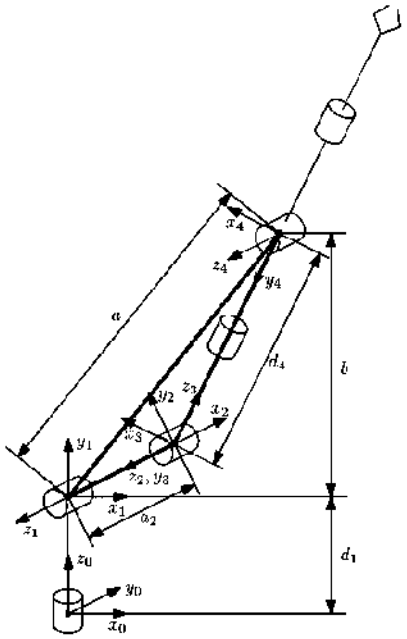
The first generalized coordinate can be computed as

$$\theta_1 = \text{atan2}(y_4^0, x_4^0) \tag{22}$$

or

$$\theta_1 = \text{atan2}(y_4^0, x_4^0) + \pi. \tag{23}$$

Inverse Kinematics: Position IK



Vectors a and b

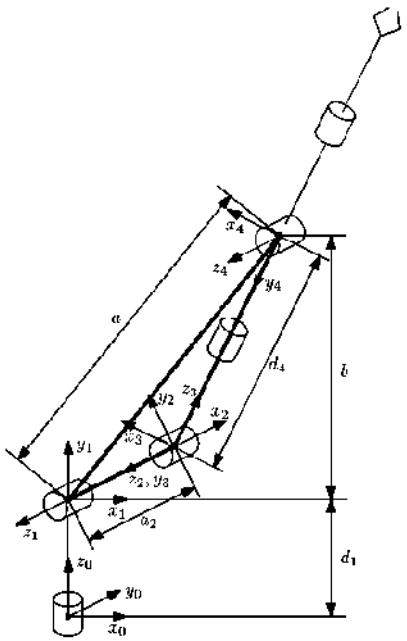
Use the following notations

$$a = \sqrt{\left(x_4^1\right)^2 + \left(y_4^1\right)^2 + \left(z_4^1\right)^2}, \quad (24)$$

$$b = \left(z_4^0 - d_1\right), \quad (25)$$

$$c = \sqrt{\left(x_4^0\right)^2 + \left(y_4^0\right)^2}. \quad (26)$$

Inverse Kinematics: Position IK



Vectors a and b

Using the Pythagorean theorem write

$$a^2 = b^2 + c^2. \quad (27)$$

Using the law of cosines write

$$a^2 = a_2^2 + d_4^2 - 2a_2 d_4 \cos(\pi - \theta_3) = \\ = a_2^2 + d_4^2 + 2a_2 d_4 \cos\theta_3. \quad (28)$$

Combining the both expressions write

$$b^2 + c^2 = a_2^2 + d_4^2 + 2a_2 d_4 \cos\theta_3, \quad (29)$$

from which express $\cos\theta_3$

$$\cos\theta_3 = \frac{b^2 + c^2 - a_2^2 - d_4^2}{2a_2 d_4}. \quad (30)$$

As a result the generalized coordinate θ_3 can be computed as

$$\theta_3 = \operatorname{atan2}\left(\pm\sqrt{1 - \cos^2\theta_3}, \cos\theta_3\right). \quad (31)$$

Inverse Kinematics: Position IK

Consider difference between to angles

- angle α formed by a and c
- angle β formed by a and a_2

Express the generalized coordinate θ_2 as

$$\theta_2 = \alpha - \beta. \tag{32}$$

Taking into account trigonometric expressions

$$\tan\alpha = \frac{b}{c}, \tag{33}$$

$$\tan\beta = \frac{d_4 \sin\theta_3}{a_2 + d_4 \cos\theta_3}, \tag{34}$$

rewrite (32) as

$$\theta_2 = \operatorname{atan2}(b, c) - \operatorname{atan2}(d_4 \sin\theta_3, a_2 + d_4 \cos\theta_3). \tag{35}$$

Inverse Kinematics: Orientation IK

Express the rotation matrix R_6^0 as

$$R_6^0 = R_3^0 R_6^3, \tag{36}$$

where R_6^0 is given, R_3^0 can be calculated solving FK. Express R_6^3 as

$$R_6^3 = \left(R_3^0\right)^{-1} R_6^0 = \left(R_3^0\right)^T R_6^0. \tag{37}$$

Consider ZYZ-transformation given by the Euler angles as

$$R_6^3 = R_{zyz} = R_{z,\phi} R_{y,\theta} R_{z,\psi} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}. \tag{38}$$

The remaining three generalized coordinates can be computed as

$$\theta_4 = \phi = \operatorname{atan2}\left(\pm r_{23}, \pm r_{13}\right), \tag{39}$$

$$\theta_5 = \theta = \operatorname{atan2}\left(\pm\sqrt{1 - r_{33}^2}, r_{33}\right), \tag{40}$$

$$\theta_6 = \psi = \operatorname{atan2}\left(\pm r_{32}, \mp r_{31}\right). \tag{41}$$

Inverse Kinematics: Summary

1. Solve forward kinematics
2. Calculate the coordinates of the intersection between the rotation axes given the coordinated of the tool
3. Solve position IK and get θ_1, θ_2 and θ_3
4. Calculate R_3^0 from forward kinematics
5. Calculate matrix R_6^3
6. Solve orientation IK and get θ_4, θ_5 and θ_6 as the Euler angles forming the matrix R_6^3

Dynamics of industrial robots

Dynamics of Industrial Robots

Dr. Oleg Borisov

Dynamical Model of Revolute Joint: Two Components

The electrical component of the model describes a circuit with the inductance, resistance and motor as

$$L\dot{i}(t) + Ri(t) = u(t) - K_\varepsilon \omega(t) = u(t) - K_\varepsilon \dot{\theta}(t), \tag{1}$$

where L, R, $i(t)$, $u(t)$ are the inductance, resistance, current and voltage of the armature, respectively, K_ε is the back emf constant, $\omega(t)$, $\theta(t)$ are the angular velocity and position of the rotor, respectively.

The mechanical component of the model describes a gear train with the gear ratio j connected with the motor as

$$J\ddot{\theta}(t) + K_f \dot{\theta}(t) = K_\mu i(t) - \mu_l(t), \tag{2}$$

where J is the sum of the actuator and gear moments of inertia, K_f is the friction constant, K_μ is the torque constant, $\mu_l(t) = \frac{1}{j}\mu_l(t)$, $\mu_l(t)$ is the load torque, j is the gear ratio.

Dynamical Model of Revolute Joint: Transfer Functions

Apply the Laplace transform and rewrite the model (1) and (2) as

$$(Ls + R)I(s) = U(s) - K_\varepsilon s\Theta(s), \tag{3}$$
$$(Js + K_f)s\Theta(s) = K_\mu I(s) - M_l(s). \tag{4}$$

Taking into account (3) and (4) let us write the transfer function from the input $U(s)$ to the output $\Theta(s)$ with $M_l(s) = 0$

$$\frac{\Theta(s)}{U(s)} = \frac{K_\mu}{s((Ls + R)(Js + K_f) + K_\varepsilon K_\mu)}. \tag{5}$$

The transfer function from $M_l(s)$ to $\Theta(s)$ with $U(s) = 0$ is

$$\frac{\Theta(s)}{M_l(s)} = -\frac{Ls + R}{s((Ls + R)(Js + K_f) + K_\varepsilon K_\mu)}. \tag{6}$$

Dynamical Model of Revolute Joint: Simplification

Now divide numerator and denominator of the transfer functions (5) and (6) by R

$$\frac{\Theta(s)}{U(s)} = \frac{\frac{K_\mu}{R}}{s\left(\left(\frac{L}{R}s+1\right)(Js+K_f)+\frac{K_\varepsilon K_\mu}{R}\right)}, \tag{7}$$

$$\frac{\Theta(s)}{M_l(s)} = -\frac{\frac{L}{R}s+1}{s\left(\left(\frac{L}{R}s+1\right)(Js+K_f)+\frac{K_\varepsilon K_\mu}{R}\right)}. \tag{8}$$

Since the time constant of the electrical component is reasonably much smaller than the time constant of the mechanical one

$$\frac{L}{R} << \frac{J}{K_f}, \tag{9}$$

rewrite transfer functions (7) and (8)

$$\frac{\Theta(s)}{U(s)} \approx \frac{\frac{K_\mu}{R}}{s\left(Js+K_f+\frac{K_\varepsilon K_\mu}{R}\right)}, \frac{\Theta(s)}{M_l(s)} \approx \frac{-1}{s\left(Js+K_f+\frac{K_\varepsilon K_\mu}{R}\right)}. \tag{10}$$

Dynamical Model of Revolute Joint: The Resultant Model

Define new notations for these transfer functions

$$\frac{\Theta(s)}{M_u(s)} \approx \frac{1}{s(Js+K)}, \tag{11}$$

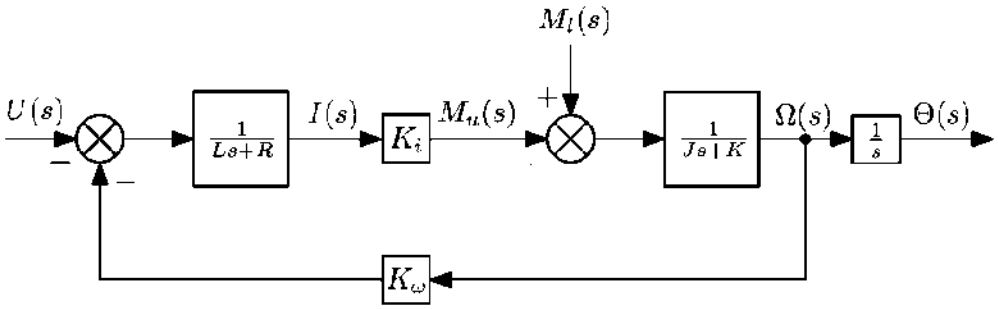
$$\frac{\Theta(s)}{M_l(s)} \approx \frac{1}{s(Js+K)}, \tag{12}$$

where $M_u(s) = \frac{K_\mu}{R}U(s)$, $K = K_f + \frac{K_\varepsilon K_\mu}{R}$.

Combining transfer functions (11) and (12) we get

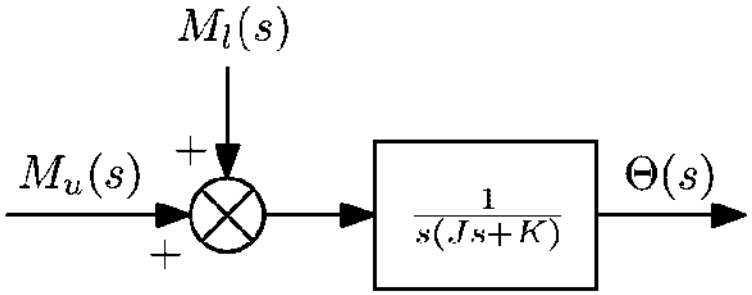
$$\Theta(s) = \frac{1}{s(Js+K)}\left(M_u(s) - M_l(s)\right) = P(s)\left(M_u(s) - M_l(s)\right). \tag{13}$$

Dynamical Model of Revolute Joint: Initial Scheme



Initial scheme of the revolute joint model

Dynamical Model of Revolute Joint: Simplified Scheme



Simplified scheme of the revolute joint model

Dynamical Model of the Robot: Euler-Lagrange Equation

Dynamics of mechanical systems can be described by the *Euler-Lagrange equation* as

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = \mu_i, \tag{14}$$

where L is the Lagrangian, q_i, $\dot{q}_i$ are the generalized coordinates and velocities, μ_i are the generalized torques applied to the joints.

The Langragian L can be computed as

$$L = K - P, \tag{15}$$

where K and L are the full kinetic and potential energies of the system, respectively.

Dynamical Model of the Robot: Kinetic Energy

The kinetic energy of the link is comprised of the linear and angular components

$$K_i = \frac{1}{2} m_i |v_i|^2 + \frac{1}{2} \omega_i^T I_i^0 \omega_i, \tag{16}$$

where m_i is the mass of the link, v_i is the linear velocity of the center of mass, ω_i is the angular velocity of the frame assigned with the link, I_i^0 is the inertia tensor with respect to the base frame.

Express the linear and angular velocities using the Jacobian matrix

$$v_i = J_{v_i}(q)\dot{q}, \tag{17}$$

$$\omega_i = J_{\omega_i}(q)\dot{q}. \tag{18}$$

Dynamical Model of the Robot: Kinetic Energy

Express the inertia tensor as

$$I_i^0 = R_i I R_i^T, \tag{19}$$

where R_i is the rotation matrix between the base and link frames, I is the inertia tensor with respect to the link frame given as $I = \begin{bmatrix} i_{11} & i_{12} & i_{13} \\ i_{21} & i_{22} & i_{23} \\ i_{31} & i_{32} & i_{33} \end{bmatrix}$, where the elements are defined as

$$i_{11} = \iiint (y^2 + z^2)\rho(x,y,z)dxdydz, \quad i_{12} = i_{21} = -\iiint xy\rho(x,y,z)dxdydz,$$
$$i_{12} = \iiint (x^2 + z^2)\rho(x,y,z)dxdydz, \quad i_{13} = i_{31} = -\iiint xz\rho(x,y,z)dxdydz,$$
$$i_{13} = \iiint (x^2 + y^2)\rho(x,y,z)dxdydz, \quad i_{23} = i_{32} = -\iiint yz\rho(x,y,z)dxdydz,$$

where $\rho(x,y,z)$ is the funcition of mass density.

Rewrite the kinetic energy as

$$K_i = \frac{1}{2} m_i \dot{q}^T J_{v_i}^T J_{v_i} \dot{q} + \frac{1}{2} \dot{q}^T J_{\omega_i}^T R_i I R_i^T J_{\omega_i} \dot{q}. \tag{20}$$

Dynamical Model of the Robot: Full Energy

The full kinetic energy of the robot can be computed as

$$K = \frac{1}{2} \dot{q}^T \sum_{i=1}^{n} \left(m_i J_{v_i}^T J_{v_i} + J_{\omega_i}^T R_i I R_i^T J_{\omega_i} \right) \dot{q} = \frac{1}{2} \dot{q}^T \Lambda(q) \dot{q}. \tag{21}$$

The potential energy of the each link is computed as

$$P_i = m_i g^T p_i, \tag{22}$$

whre m_i is the mass of the link, g is the vector defining the direction of the gravitation with respect to the base frame, p_i is the radius-vector to the center of mass of the link expressed with respect to the base frame.

The full potential energy of the robot can be computed as

$$P = \sum_{i=1}^{n} m_i g^T p_i. \tag{23}$$

Dynamical Model of the Robot: Model of Multilink System

Substitute the kinetic and potential energies to the Lengrangian

$$L = \frac{1}{2}\dot{q}^T \Lambda(q)\dot{q} - \sum_{i=1}^{n} m_i g^T p_i. \tag{24}$$

Substitute the Langrangian to the Euler-Langrange equation

$$\Lambda(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \mu, \tag{25}$$

where $\Lambda(q) \in \mathbb{R}^{n\times n}$ is the symmetrical matrix of inertia, $C(q,\dot{q}) \in \mathbb{R}^{n\times 1}$ is the matrix of Coriolis forces, $G(q) \in \mathbb{R}^{n\times 1}$ is the vector of gravitational forces.

Dynamical Model of the Robot: Actuator Dynamics Revised

Write the dynamical model of the actuator dynamics as follows

$$J_i\ddot{\theta}_i(t) + F_i\dot{\theta}_i(t) = K_i\frac{u_i(t)}{r_i} - \mu_i(t), \tag{26}$$

where $F_i = K_f$, $K_i = K_\mu$, $r_i = R$, $\frac{u_i(t)}{r_i} = i(t)$, $i = \{1, 2, \ldots n\}$ is the number of the link.

Take into account gear box

$$q_i = \frac{\theta_i}{j_i}. \tag{27}$$

Rewrite the actuator dynamics as

$$j_i^2 J_i\ddot{q}_i(t) + j_i^2 F_i\dot{q}_i(t) = j_i K_i\frac{u_i(t)}{r_i} - \bar{\mu}_i(t), \tag{28}$$

where $\mu_i = \mu_l$ for the link i.

Dynamical Model of the Robot: Actuator Dynamics Augmentation

Add the actuator dynamical model to the model of the mechanical system and get

$$\Gamma(q)\ddot{q} + C(q,\dot{q})\dot{q} + F\dot{q} + G(q) = u, \tag{29}$$

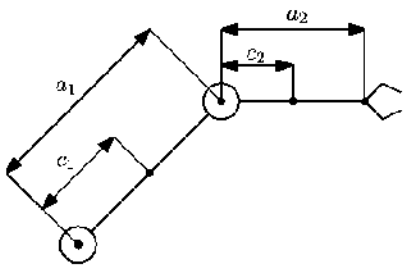
where the matrix $\Gamma(q)$ is of the form

$$\Gamma(q) = \Lambda(q) + J = \Lambda(q) + \begin{bmatrix} j_1^2 J_1 & 0 & \dots & 0 \\ 0 & j_2^2 J_2 & \dots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & 0 & j_n^2 J_n \end{bmatrix}, \tag{30}$$

where the friction vector and vector of control inputs are given respectively as

$$\begin{bmatrix} j_1^2 F_1 \\ j_2^2 F_2 \\ \vdots \\ j_n^2 F_n \end{bmatrix}, \quad \begin{bmatrix} j_1 K_1 \frac{u_1(t)}{R_1}(t) \\ j_2 K_2 \frac{u_2(t)}{R_2}(t) \\ \vdots \\ j_n K_n \frac{u_n(t)}{R_n}(t) \end{bmatrix}. \tag{31}$$

Example of Two-Link Planar Manipulator: Jacobian matrices



Kinematic chain of two-link robot

Write relations between linear end-effector velocities and generalized ones using the notion of the Jacobian matrix as follows

$$v_1 = J_{v,1}\dot{q}, \quad v_2 = J_{v,2}\dot{q}, \tag{32}$$

where

$$J_{v,1} - \begin{bmatrix} -a_1 \sin q_1 & 0 \\ c_1 \cos q_1 & 0 \\ 0 & 0 \end{bmatrix}, J_{v,2} - \begin{bmatrix} -a_1 \sin q_1 - c_2 \sin(q_1+q_2) & -c_2 \sin(q_1+q_2) \\ a_1 \cos q_1 + c_2 \cos(q_1+q_2) & c_2 \cos(q_1+q_2) \\ 0 & 0 \end{bmatrix}.$$

Example of Two-Link Planar Manipulator: Kinetic Energy

The kinetic energy is comprised of translational and rotational components. Let us address them separately. The translational component caused by the linear velocity can be computed as

$$K_{tr} = \frac{m_1 v_1^T v_1}{2} + \frac{m_2 v_2^T v_2}{2} = \underbrace{0.5\dot{q}^T m_1 J_{v,1}^T J_{v,1}\dot{q}}_{\text{1st link}} + \underbrace{0.5\dot{q}^T m_2 J_{v,2}^T J_{v,2}\dot{q}}_{\text{2nd link}} \tag{33}$$

The rotational component caused by the angular velocity can be computed as

$$K_{rt} = \underbrace{0.5\dot{q}^T I_1 \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \dot{q}}_{\text{1st link}} + \underbrace{0.5\dot{q}^T I_2 \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \dot{q}}_{\text{2nd link}} \tag{34}$$

Example of Two-Link Planar Manipulator: Inertia Matrix

The inertia matrix $\Lambda(q)$ becomes of the form

$$\Lambda(q) = m_1 J_{v,1}^T J_{v,1} + m_2 J_{v,2}^T J_{v,2} + \begin{bmatrix} I_1 + I_2 & I_2 \\ I_2 & I_2 \end{bmatrix} = \begin{bmatrix} \lambda_{11} & \lambda_{12} \\ \lambda_{21} & \lambda_{22} \end{bmatrix}$$

$$= \begin{bmatrix} m_1 c_1^2 + m_2(a_1^2 + c_2^2 + 2a_1 c_2^2 + 2a_1 c_2 \cos q_2) + I_1 + I_2 & m_2(c_2^2 + a_1 c_2 \cos q_2) + I_2 \\ m_2(c_2^2 + a_1 c_2 \cos q_2) + I_2 & m_2 c_2^2 + I_2 \end{bmatrix}$$

Example of Two-Link Planar Manipulator: Matrix of Coriolis Forces

Each element of the matrix of Coriolis forces $C(q)$ can be calculated using the equation

$$c_{kj} = \sum_{i=1}^{n} 0.5 \left(\frac{\partial \lambda_{kj}}{\partial q_i} + \frac{\partial \lambda_{ki}}{\partial q_j} - \frac{\partial \lambda_{ij}}{\partial q_k} \right) \dot{q}_i \tag{35}$$

The matrix of Coriolis forces $C(q)$ becomes of the form

$$C(q) = \begin{bmatrix} -m_2 a_1 c_2 \sin q_2 \dot{q}_2 & -m_2 a_1 c_2 \sin q_2 (\dot{q}_2 + \dot{q}_1) \\ m_2 a_1 c_2 \sin q_2 \dot{q}_1 & 0 \end{bmatrix} \tag{36}$$

Example of Two-Link Planar Manipulator: Vector of Gravitational Forces

Each element of the vector of gravitational forces $G(q)$ can be calculated using the equation

$$g_i = \frac{\partial P}{\partial q_i} \tag{37}$$

The vector of gravitational forces $G(q)$ becomes of the form

$$G(q) = \begin{bmatrix} (m_1 c_1 + m_2 a_1) g \cos q_1 + m_2 c_2 g \cos(q_1 + q_2) \\ m_2 c_2 \cos(q_1 + q_2) \end{bmatrix} \tag{38}$$

Example of Two-Link Planar Manipulator: Resultant Model

The resultant model of the two-link robot is

$$\lambda_{11}\ddot{q}_1 + \lambda_{12}\ddot{q}_2 + c_{11}\dot{q}_1 + c_{12}\dot{q}_2 + (m_1 c_1 + m_2 a_1) g \cos q_1 + m_2 c_2 g \cos(q_1 + q_2) = \mu_1$$
$$\lambda_{21}\ddot{q}_1 + \lambda_{22}\ddot{q}_2 + c_{21}\dot{q}_1 + m_2 c_2 \cos(q_1 + q_2) = \mu_2$$

Summary

- Dynamical models of industrial robots allow to describe and take into account (designing a control law) physical processes specific to them
- The simplified model of the revolute joint can be represented by the transfer function of the relative degree 2
- The dynamical model of the industrial robot can be derived using the Euler-Lagrange approach

Motion planning for industrial robots

Motion Planning for Industrial Robots

Dr. Oleg Borisov

Basic Concepts and Definitions: Configuration Space

Configuration

A configuration q is a set of all intermediate generalized coordinates (joint variables).

Configuration space

Configirations space $\mathcal{Q}$ is a set of all possible configurations q

$$\mathcal{Q} = \{q\}. \tag{1}$$

Basic Concepts and Definitions: Workspace

Workspace

Workspace $\mathcal{W}$ is a set of points, which belong to the robot itself and the reachable environment including all the obstacles

$$\mathcal{R}(q) \subset \mathcal{W}, \quad \mathcal{O} \subset \mathcal{W}, \tag{2}$$

where $\mathcal{R}(q)$ is space occupied by the robot and $\mathcal{O}$ is space occupied by the obstacles.

In case of a planar manipulator which movements are constrained by the plane

$$\mathcal{W} \subset \mathbb{R}^2, \tag{3}$$

its workspace has two-dimensional.

In case of a spatial manipulator, which is able to move along three orthogonal axes

$$\mathcal{W} \subset \mathbb{R}^3, \tag{4}$$

its workspace is three-dimensional.

Basic Concepts and Definitions: Collision-Free Space

Collision-Free Space

Space corresponding to collision of the robot with some obstacle is defined as follows

$$\mathcal{Q}_\times = \{q \in \mathcal{Q} | \mathcal{R}(q) \cap \mathcal{O} \neq 0\}, \tag{5}$$

from which collision-free space can be expressed as

$$\mathcal{Q}_0 = \mathcal{Q} \setminus \mathcal{Q}_\times. \tag{6}$$

Basic Concepts and Definitions: Path and Trajectory

Path Planning

Path planning is a process of searching a cosecutive set of configurations within collision-free space connecting the initial configuration with the given final one.

Trajectory Planning

Trajectory planning is a process of time parametrization of the path, i.e. computation of reference functions of time for generalized coordinates, velocities and accelerations.

Path Planning: Exact Cell Decomposition Approach

Exact Cell Decomposition

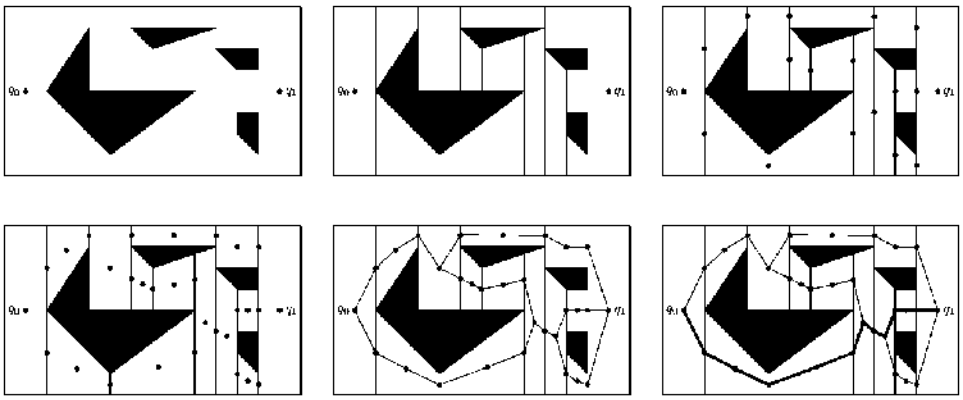
The idea of *exact cell decomposition* is to divide whole free configuration space on triangle or trapezoid cells and to construct a graph. Its nodes are represented by centers of the cells and its links are common sides between adjacent cells.

In case of exact cell decomposition there are two types of cells

- white cells correspond to the collision-free space
- black cells correspond to the collision space

Then given initial and final configurations, search of consecutive transition from one white cell to another one is carrying out to connect these two configurations and avoid all the black cells.

Path Planning: Exact Cell Decomposition Approach



Steps of Exact Cell Decomposition Approach

Path Planning: Approximate Cell Decomposition Approach

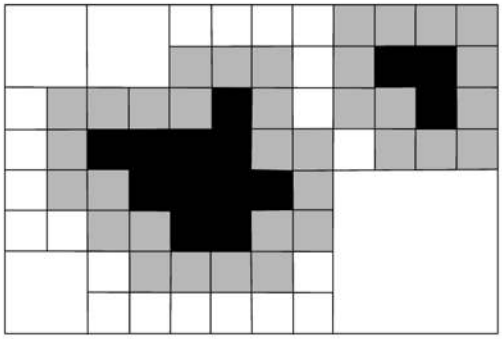
> **Approximate Cell Decomposition**
> Difference of the *approximate cell decomposition* with respect to its "exact" version is that instead of the whole configuration space its subset is divided on cells. So, the remaining space could include also slight parts of collision-free space, which is caused by complex shape of the collision space.

In case of approximate cell decomposition there are two types of cells

- white cells correspond to the collision-free space
- black cells correspond to the collision space
- gray cells correspond to the both spaces

While searching a path could pass both white and gray cells. If it touches grays cells, additional cell decomposition should be carried out until the path connecting initial and final configurations goes through white cells only.

Path Planning: Approximate Cell Decomposition Approach



Approximate Cell Decomposition

Path Planning: Potential Field Approach

Potential Field Approach

he robot is considered as a material point moving in a configuration space under influence of a potential field function $P(q)$. It has attraction component $P_a(q)$ assigned with the final configuration and repulsive component $P_r(q)$ assigned with the collision space

$$P(q) = P_a(q) + P_r(q). \tag{7}$$

Path Planning: Potential Field Approach

Set the global minimum of the function $P(q)$ as the attraction component $P_a(q)$

$$P_a(q) = \frac{1}{2} k_a ||q - q_d||^2, \tag{8}$$

where q, q_d are the current and desired configurations, respectively, k_a is the scaling factor.

The repulsive component $P_r(q)$ ensures singularity of the function $P(q)$ when the material point is approaching the collision space

$$P_r(q) = \begin{cases} \frac{1}{2} k_r \left(\frac{1}{\delta(q)} - \frac{1}{\delta_0} \right)^2 & \text{if} \quad \delta(q) \leq \delta_0, \\ 0 & \text{if} \quad \delta(q) > \delta_0, \end{cases} \tag{9}$$

where k_r is the scaling factor, $\delta(q)$ is the shortest distance from the current configuration to the collision space, δ_0 is the minimum value.

Path Planning: Potential Field Approach

The gradient descent algorithm can be used to plan a path

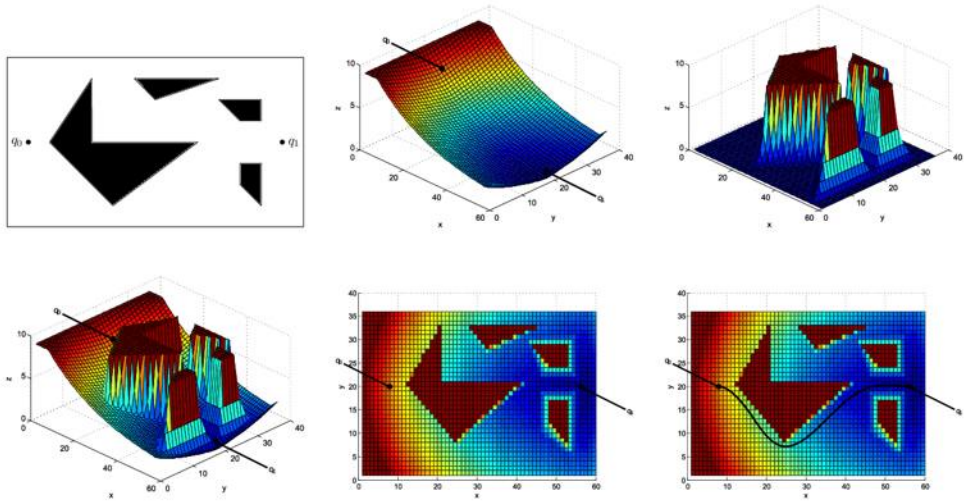
$$q_{j+1} = q_j - \gamma_j \nabla P(q_j), \tag{10}$$

where $\nabla P(q) = \left[\frac{\partial P}{\partial q_1} \quad \frac{\partial P}{\partial q_2} \ldots \quad \frac{\partial P}{\partial q_n} \right]^T$, γ_j is a iterative step, which can be either fixed, fractioned, or calculated in the direction of the fastest descent as

$$\gamma_j = \text{argmin}_j \, P(q_j - \gamma \nabla P(q_j)). \tag{11}$$

The main disadvantage of the potential field approach is possibility to stuack at the local minimum instead of the global one. So called random motion approach is used to avoid this issue.

Path Planning: Potential Field Approach



Steps of Potential Field Approach

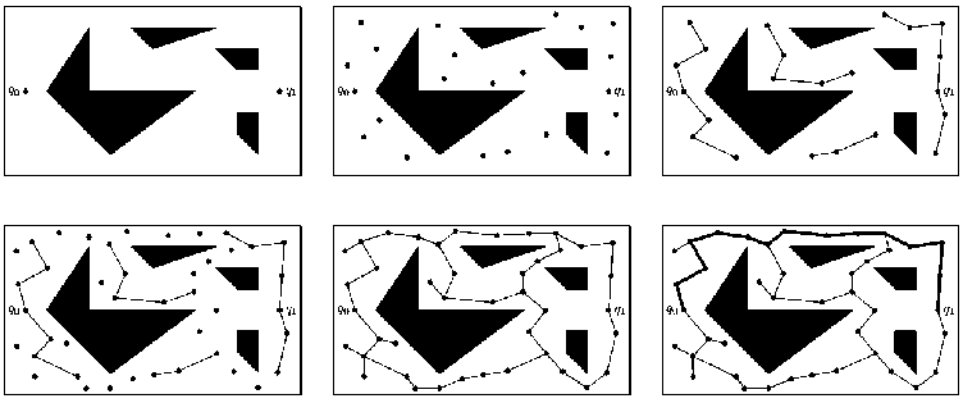
Path Planning: Probabilistic Roadmap Approach

Probabilistic Roadmap Approach

Probabilistic roadmap approach is useful for fast path generation. It is based on the usage of random samples from the configuration space.

1. Several nodes (samples) are chosen randomly from the configuration space. Each node is assigned with a particular configuration.
2. Adjacent nodes are being connected between each other within the specified norm in the configuration space.
3. The fisrt two steps are repeated to cover sufficiently large area between the initial and final configurations.
4. A cosequtive set of samples are chosen to connect the initial and final configurations.

Path Planning: Probabilistic Roadmap Approach



Steps of Probabilistic Roadmap Approach

Trajectory Planning: Spline Functions Approach

Spline Functions Approach

The idea of this approach is to interpolate generalized coordinates, velocities and accelerations between the reference points using the polynomials of the form

$$q_i(t) = a_{l,i}t^l + a_{l-1,i}t^{l-1} + \cdots + a_{2,i}t^2 + a_{1,i}t + a_{0,i}, \quad (12)$$

$$\dot{q}_i(t) = la_{l,i}t^{l-1} + (l-1)a_{l-1,i}t^{l-2} + \cdots + 2a_{2,i}t + a_{1,i}, \quad (13)$$

$$\ddot{q}_i(t) = l(l-1)a_{l,i}t^{l-2} + (l-1)(l-2)a_{l-1,i}t^{l-3} + \cdots + 2a_{2,i}, \quad (14)$$

where the degree l and coefficients a_{ji}, $j = \{1, 2, \ldots, l\}$ are calculated depending on the constraints and continuity requirements on the trajectory.

1. Divide the whole trakectory on several elementary subtrajectories.
2. Compute relative time functions τ_i for each subtrjactory.
3. Apply constraints and continuity requirements on the trajectory.
4. Determine the highst polynomial degree for each subtrajectory.
5. Solve matrix equation to compute coefficients of all the polynomials.

Trajectory Planning: Single Subtrajectory Case

Only initial and final configurations are given. No intermediate requirements.

Consider the following constraints for each link of the robot

$$q_i(t_0) = \vartheta_0, \quad \dot{q}_i(t_0) = v_0, \quad \ddot{q}_i(t_0) = \alpha_0, \tag{15}$$

$$q_i(t_1) = \vartheta_1, \quad \dot{q}_i(t_1) = v_1, \quad \ddot{q}_i(t_1) = \alpha_1. \tag{16}$$

Choose the polynomial to interpolate intermediate values of the generalized coordinates

$$\vartheta(t) = a_5t^5 + a_4t^4 + a_3t^3 + a_2t^2 + a_1t + a_0. \tag{17}$$

Calculate the first and second derivatives of this polynomial to interpolate values of generalized velocities and accelerations

$$\dot{\vartheta}(t) = v(t) = 5a_5t^4 + 4a_4t^3 + 3a_3t^2 + 2a_2t + a_1, \tag{18}$$

$$\ddot{\vartheta}(t) = \alpha(t) = 20a_5t^3 + 12a_4t^2 + 6a_3t + 2a_2. \tag{19}$$

Trajectory Planning: Single Subtrajectory Case

Write the system of equations taking into account the imposed constraits and continuity requirements as follows

$$\begin{cases} \vartheta_0 = a_5t_0^5 + a_4t_0^4 + a_3t_0^3 + a_2t_0^2 + a_1t_0 + a_0, \\ v_0 = 5a_5t_0^4 + 4a_4t_0^3 + 3a_3t_0^2 + 2a_2t_0 + a_1, \\ \alpha_0 = 20a_5t_0^3 + 12a_4t_0^2 + 6a_3t_0 + 2a_2, \\ \vartheta_1 = a_5t_1^5 + a_4t_1^4 + a_3t_1^3 + a_2t_1^2 + a_1t_1 + a_0, \\ v_1 = 5a_5t_1^4 + 4a_4t_1^3 + 3a_3t_1^2 + 2a_2t_1 + a_1, \\ \alpha_1 = 20a_5t_1^3 + 12a_4t_1^2 + 6a_3t_1 + 2a_2. \end{cases} \tag{20}$$

Rewrite this system in matrix form as

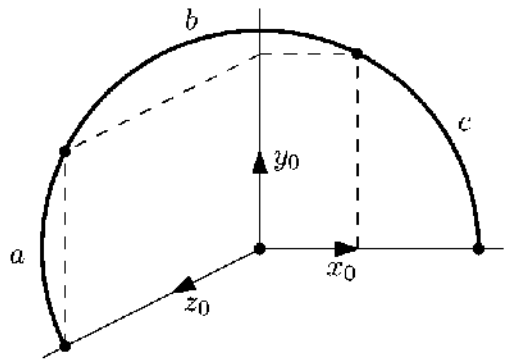
$$\underbrace{\begin{bmatrix} \vartheta_0 \\ v_0 \\ \alpha_0 \\ \vartheta_1 \\ v_1 \\ \alpha_1 \end{bmatrix}}_{\varrho} = \underbrace{\begin{bmatrix} t_0^5 & t_0^4 & t_0^3 & t_0^2 & t_0 & 1 \\ 5t_0^4 & 4t_0^3 & 3t_0^2 & 2t_0 & 1 & 0 \\ 20t_0^3 & 12t_0^2 & 6t_0 & 2 & 0 & 0 \\ t_1^5 & t_1^4 & t_1^3 & t_1^2 & t_1 & 1 \\ 5t_1^4 & 4t_1^3 & 3t_1^2 & 2t_1 & 1 & 0 \\ 20t_1^3 & 12t_1^2 & 6t_1 & 2 & 0 & 0 \end{bmatrix}}_{T} \underbrace{\begin{bmatrix} a_5 \\ a_4 \\ a_3 \\ a_2 \\ a_1 \\ a_0 \end{bmatrix}}_{\varsigma}, \tag{21}$$

from which the vector of unknown coefficients can be easily expressed as

Trajectory Planning: Multiple Subtrajectory Case

Consider a trajectory comprised of three subtrajectories as follows

- Leaving (a)
- Transition (b)
- Approach (c)



Trajectory comprised of three segments

Trajectory Planning: Multiple Subtrajectory Case

Consider the following constraints for each link of the robot

$$q_i(t_0) = \vartheta_0, \quad q_i(t_1) = \vartheta_1, \quad q_i(t_2) = \vartheta_2, \quad q_i(t_3) = \vartheta_3, \tag{23}$$

$$\dot{q}_i(t_0) = v_0, \quad \ddot{q}_i(t_0) = \alpha_0, \quad \dot{q}_i(t_3) = v_3, \quad \ddot{q}_i(t_3) = \alpha_3. \tag{24}$$

Use the relative time functions for each subtrajectory

$$\tau_a = \frac{t - t_0}{t_1 - t_0}, \qquad \tau_b = \frac{t - t_1}{t_2 - t_1}, \qquad \tau_c = \frac{t - t_2}{t_3 - t_2}, \tag{25}$$

where t_0, t_1, t_2, t_3 are the given time moments of passing all the reference configurations.

Impose continuity requirements to get a smooth trajectory

$$v_a(1) = v_b(0), \quad \alpha_a(1) = \alpha_b(0), \quad v_b(1) = v_c(0), \quad \alpha_b(1) = \alpha_c(0). \tag{26}$$

Taking into account the relative time functions rewrite constraints on the trajectory

$$\vartheta_a(0) = \vartheta_0, \quad \vartheta_b(0) = \vartheta_1, \quad \vartheta_c(0) = \vartheta_2, \tag{27}$$

$$\vartheta_a(1) = \vartheta_1, \quad \vartheta_b(1) = \vartheta_2, \quad \vartheta_c(1) = \vartheta_3, \tag{28}$$

and continuity requirementsas follows

Trajectory Planning: Multiple Subtrajectory Case

Choose the polynomial to interpolate intermediate values of the generalized coordinates within each subtrajectory

$$\vartheta_a(\tau_a) = a_4\tau_a^4 + a_3\tau_a^3 + a_2\tau_a^2 + a_1\tau_a + a_0, \quad (30)$$

$$\vartheta_b(\tau_b) = b_3\tau_b^3 + b_2\tau_b^2 + b_1\tau_b + b_0, \quad (31)$$

$$\vartheta_c(\tau_c) = c_4\tau_c^4 + c_3\tau_c^3 + c_2\tau_c^2 + c_1\tau_c + c_0, \quad (32)$$

Calculate the first derivative of these polynomials to interpolate values of generalized velocities

$$\dot{\vartheta}_a(\tau_a) = v_a(\tau_a) = 4a_4\tau_a^3 + 3a_3\tau_a^2 + 2a_2\tau_a + a_1, \quad (33)$$

$$\dot{\vartheta}_b(\tau_b) = v_b(\tau_b) = 3b_3\tau_b^2 + 2b_2\tau_b + b_1, \quad (34)$$

$$\dot{\vartheta}_c(\tau_c) = v_c(\tau_c) = 4c_4\tau_c^3 + 3c_3\tau_c^2 + 2c_2\tau_c + c_1, \quad (35)$$

Calculate the second derivative of these polynomials to interpolate values of generalized accelerations

$$\ddot{\vartheta}_a(\tau_a) = \alpha_a(\tau_a) = 12a_4\tau_a^2 + 6a_3\tau_a + 2a_2, \quad (36)$$

$$\ddot{\vartheta}_b(\tau_b) = \alpha_b(\tau_b) = 6b_3\tau_b + 2b_2, \quad (37)$$

$$\ddot{\vartheta}_c(\tau_c) = \alpha_c(\tau_c) = 12c_4\tau_c^2 + 6c_3\tau_c + 2c_2, \quad (38)$$

Trajectory Planning: Multiple Subtrajectory Case

Write the system of equations taking into account the imposed constraits and continuity requirements as follows

$$\begin{cases} \vartheta_0 = a_0, \\ v_0 = a_1, \\ \alpha_0 = 2a_2, \\ \vartheta_1 = a_4 + a_3 + a_2 + a_1 + a_0, \\ \vartheta_1 = b_0, \\ 0 = 4a_4 + 3a_3 + 2a_2 + a_1 - b_1, \\ 0 = 12a_4 + 6a_3 + 2a_2 - 2b_2, \\ \vartheta_2 = b_3 + b_2 + b_1 + b_0, \\ \vartheta_2 = c_0, \\ 0 = 3b_3 + 2b_2 + b_1 - c_1, \\ 0 = 6b_3 + 2b_2 - 2c_2, \\ \vartheta_3 = c_4 + c_3 + c_2 + c_1 + c_0, \\ v_3 = 4c_4 + 3c_3 + 2c_2 + c_1, \\ \alpha_3 = 12c_4 + 6c_3 + 2c_2. \end{cases} \quad (39)$$

Trajectory Planning: Multiple Subtrajectory Case

Rewrite this system in matrix form as

$$\underbrace{\begin{bmatrix}\vartheta_0\\ v_0\\ \alpha_0\\ \vartheta_1\\ \vartheta_1\\ 0\\ 0\\ \vartheta_2\\ \vartheta_2\\ 0\\ 0\\ \vartheta_3\\ v_3\\ \alpha_3\end{bmatrix}}_{\varrho} = \underbrace{\begin{bmatrix}0&0&0&0&1&0&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&2&0&0&0&0&0&0&0&0&0&0&0\\ 1&1&1&1&1&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&1&0&0&0&0&0\\ 4&3&2&1&0&0&0&-1&0&0&0&0&0&0\\ 12&6&2&0&0&0&-2&0&0&0&0&0&0&0\\ 0&0&0&0&0&1&1&1&1&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&1\\ 0&0&0&0&0&3&2&1&0&0&0&0&-1&0\\ 0&0&0&0&0&6&2&0&0&0&0&-2&0&0\\ 0&0&0&0&0&0&0&0&0&1&1&1&1&1\\ 0&0&0&0&0&0&0&0&0&4&3&2&1&0\\ 0&0&0&0&0&0&0&0&0&12&6&2&0&0\end{bmatrix}}_{T} \underbrace{\begin{bmatrix}a_4\\ a_3\\ a_2\\ a_1\\ a_0\\ b_3\\ b_2\\ b_1\\ b_0\\ c_4\\ c_3\\ c_2\\ c_1\\ c_0\end{bmatrix}}_{\varsigma}, \tag{40}$$

from which the vector of unknown coefficients can be easily expressed as

$$\varsigma = T^{-1}\varrho. \tag{41}$$

Arc Approximation Algorithm of Spatial Movements

> **Research Objective**
> This study focuses on spatial motion planning algorithms, which allows to characterize sophisticated reference paths in 3D space and simplify the way how they can be given. The key point used in this study is approximation of a sequence of points by a sequence of arcs within a specified δ-region.

In industry such algorithm can be applied for such tasks as surface finishing, engraving and welding. The last operation represents the main interest of this research.

Problem

Mitsubishi RV-3SDB

Objective

The purpose is automated code generating to move the end-effector along some counters specified by the input bitmap image or 3D model.

After extracting coordinates of initial points sequence they already can be programmed using trivial point-to-point motion, but it might lead to some issues.

- significant input data (robot controller overload)
- decrease of the motion velocity (reconfiguration at each reference point)

Arc Approximation Algorithm

Basic Idea

This approach is based on the feasibility of the standard software to move the end-effector along an arc, specified with only three points. This basic motion provided by the internal software is more natural then complex combinations of multiple linear point-to-point movements. As a result, the robot reconfigures only three times at the reference points forming this arc. Such solution allows to reduce the code size and increase the velocity.

Planar Planning

Consider three reference points

$$p_1 = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}, \quad p_2 = \begin{bmatrix} x_2 \\ y_2 \end{bmatrix}, \quad p_3 = \begin{bmatrix} x_3 \\ y_3 \end{bmatrix}. \tag{42}$$

All intermediate points between p_1, p_2 and p_3 should belong to a corresponding arc within some δ_{arc}-region.

Consider two lines p_1–p_2 and p_2–p_3. In order to find coordinates of the arc center $c = \begin{bmatrix} x_c \\ y_c \end{bmatrix}$ consider three cases.

Case 1

If $x_2 = x_3$ and $x_1 \neq x_2$ then

$$y_c = \frac{y_2 + y_3}{2}, \quad x_c = -k_1 \frac{y_c - (y_1 + y_2)}{2} + \frac{x_1 + x_2}{2},$$

where k_1 is the slope of the line $(x_1; y_1)$–$(x_2; y_2)$ given by $k_1 = \frac{y_2 - y_1}{x_2 - x_1}$.

Case 2

If $x_1 = x_2$ and $x_2 \neq x_3$ then

$$y_c = \frac{y_1 + y_2}{2}, \quad x_c = -k_2 \frac{y_c - (y_2 + y_3)}{2} + \frac{x_2 + x_3}{2},$$

where k_2 is the slope of the line $(x_2; y_2)$–$(x_3; y_3)$ given by $k_2 = \frac{y_3 - y_2}{x_3 - x_2}$.

Case 3

If all x-coordinates are distinct, then

$$x_c = \frac{k_1 k_2 (y_1 - y_3) + k_2 (x_1 + x_2) - k_1 (x_2 + x_3)}{2(k_2 - k_1)}, \tag{43}$$

$$y_c = -\frac{x_c - \frac{x_1 + x_2}{2}}{k_1} + \frac{y_1 + y_2}{2}, \tag{44}$$

where k_1 and k_2 are given above.

Calculate distance from a forth point $p_4 = \begin{bmatrix} x_4 \\ y_4 \end{bmatrix}$ to the arc formed by p_1, p_2 and p_3 as follows

$$d_{arc} = \left| \sqrt{(x_c - x_4)^2 + (y_c - y_4)^2} - r \right|, \tag{45}$$

where $r = \sqrt{(x_1 - x_c)^2 + (y_1 - y_c)^2}$ is the radius of the arc.

As a result we get a sequence of arcs each specified by three consecutive points. Such point list can be used together with the operator MVR P1 P2 P3, which allows to move along an arc specified by three reference points.

Spatial Planning

Consider three points that do not lie on the same line. Coordinates of vectors specified in the Cartesian space are defined as

$$p_1^0 = \begin{bmatrix} x_1^0 \\ y_1^0 \\ z_1^0 \end{bmatrix}, \quad p_2^0 = \begin{bmatrix} x_2^0 \\ y_2^0 \\ z_2^0 \end{bmatrix}, \quad p_3^0 = \begin{bmatrix} x_3^0 \\ y_3^0 \\ z_3^0 \end{bmatrix}. \tag{46}$$

Consider two coordinate systems denoted as $x_0 y_0 z_0 o_0$ and $x_1 y_1 z_1 o_1$. Derive a normal to the plane $x_1 y_1 o_1$ through a cross product

$$n = \begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix} = \left(p_2^0 - p_1^0\right) \times \left(p_3^0 - p_1^0\right). \tag{47}$$

Then calculate a unit vector

$$z = \begin{bmatrix} z_x \\ z_y \\ z_z \end{bmatrix} = \frac{n}{\sqrt{n_x^2 + n_y^2 + n_z^2}}. \tag{48}$$

Compute the rotational transformation as $R_1^0 = R_{z,\alpha} R_{y,\beta}$, where the angles α and β can be calculated as follows

$$\alpha = \text{atan2}\,2\left(\frac{z_y}{\sqrt{z_x^2+z_y^2}}, \frac{z_x}{\sqrt{z_x^2+z_y^2}}\right), \quad \beta = \text{atan2}\,2\left(\sqrt{z_x^2+z_y^2}, z_z\right).$$

Substitute α and β into the rotation matrices around $z-$ and y-axes

$$R_1^0 = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}. \tag{49}$$

Calculate coordinates of the reference points with respect to the local coordinate system using the rotation matrix

$$p_1^1 = R_0^1 p_1^0, \quad p_2^1 = R_0^1 p_2^0, \quad p_3^1 = R_0^1 p_3^0, \tag{50}$$

where $R_0^1 = [R_1^0]^T$.

Denote coordinates as follows

$$p_1^1 = \begin{bmatrix} x_1^1 \\ y_1^1 \\ z_1^1 \end{bmatrix}, \quad p_2^1 = \begin{bmatrix} x_2^1 \\ y_2^1 \\ z_2^1 \end{bmatrix}, \quad p_3^1 = \begin{bmatrix} x_3^1 \\ y_3^1 \\ z_3^1 \end{bmatrix}. \tag{51}$$

In order to find coordinates of the arc center $c^1 = \begin{bmatrix} x_c^1 \\ y_c^1 \\ z_c^1 \end{bmatrix}$ consider three cases.

Case 1

If $x_2^1 = x_3^1$ and $x_1^1 \neq x_2^1$ then

$$y_c^1 = \frac{y_2^1 + y_3^1}{2}, \quad x_c^1 = -k_1 \frac{y_c^1 - (y_1^1 + y_2^1)}{2} + \frac{x_1^1 + x_2^1}{2},$$

where k_1 is the slope of the line $(x_1^1; y_1^1)$–$(x_2^1; y_2^1)$ given by $k_1 = \frac{y_2^1 - y_1^1}{x_2^1 - x_1^1}$.

Case 2

If $x_1^1 = x_2^1$ and $x_2^1 \neq x_3^1$ then

$$y_c^1 = \frac{y_1^1 + y_2^1}{2}, \quad x_c^1 = -k_2 \frac{y_c^1 - (y_2^1 + y_3^1)}{2} + \frac{x_2^1 + x_3^1}{2},$$

where k_2 is the slope of the line $(x_2^1; y_2^1)$–$(x_3^1; y_3^1)$ given by $k_2 = \frac{y_3^1 - y_2^1}{x_3^1 - x_2^1}$.

Case 3

If all x-coordinates are distinct, then

$$x_c^1 = \frac{k_1 k_2 (y_1^1 - y_3^1) + k_2 (x_1^1 + x_2^1) - k_1 (x_2^1 + x_3^1)}{2(k_2 - k_1)}, \tag{52}$$

$$y_c^1 = -\frac{x_c^1 - \frac{x_1^1 + x_2^1}{2}}{k_1} + \frac{y_1^1 + y_2^1}{2}, \tag{53}$$

where k_1 and k_2 are given above.

The third z-coordinate can be derived trivially as

$$z_c^1 = z_\perp^1 = z_2^1 = z_3^1. \tag{54}$$

Express coordinates of the center with respect to the base coordinate system

$$c^0 = \begin{bmatrix} x_c^0 \\ y_c^0 \\ z_c^0 \end{bmatrix} = R_\perp^0 c^1. \tag{55}$$

The equation of a plane is given as

$$n_x x + n_y y + n_z z + n_0 = 0, \tag{56}$$

where $n_0 = -(n_x x_3^0 + n_y y_3^0 + n_z z_3^0)$.

Distances from a forth point $p_4 = \begin{bmatrix} x_4^0 \\ y_4^0 \\ z_4^0 \end{bmatrix}$ respectively to the plane d_{plane} and to the arc formed by p_1, p_2 and p_3 can be computed as

$$d_{plane} = \frac{|n_x x_4^0 + n_y y_4^0 + n_z z_4^0 + n_0|}{\sqrt{n_x^2 + n_y^2 + n_z^2}}, \tag{57}$$

$$d_{arc} = \left| \sqrt{(x_c^0 - x_4^0)^2 + (y_c^0 - y_4^0)^2 + (z_c^0 - z_4^0)^2} - r \right|, \tag{58}$$

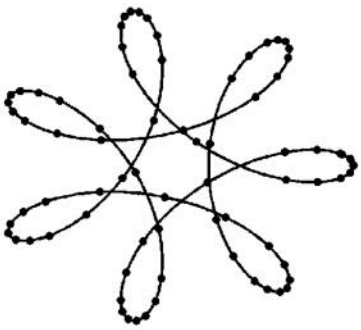
where $r = \sqrt{(x_1^0 - x_c^0)^2 + (y_1^0 - y_c^0)^2 + (z_1^0 - z_c^0)^2}$ is the radius of the arc.

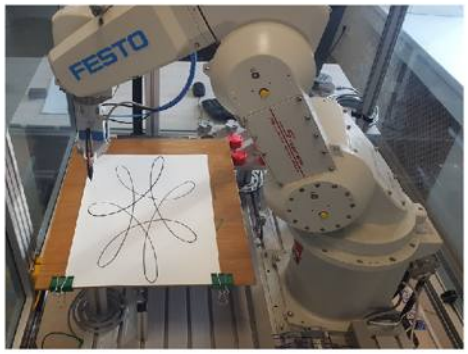
Then all points should be processed and checked on belonging them to a particular plane and arc within the specified δ_{plane}- and δ_{arc}-regions, respectively. As a result of this procedure, a sequence of three-points-sets each specifying a particular arc should be obtained.

Experimental Approval

Experimental Approval

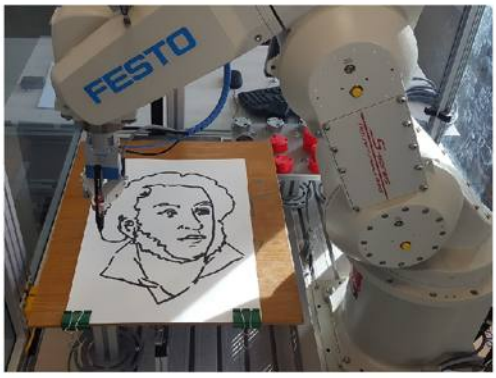
Experimental Results: Planar Planning





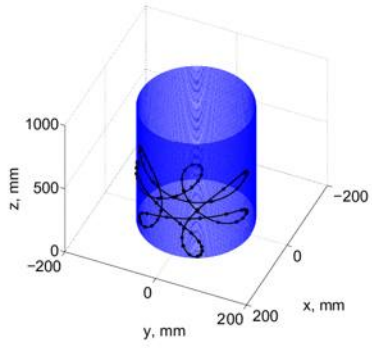
A hypotrochoid drawn by the robot on a flat surface

Experimental Results: Planar Planning



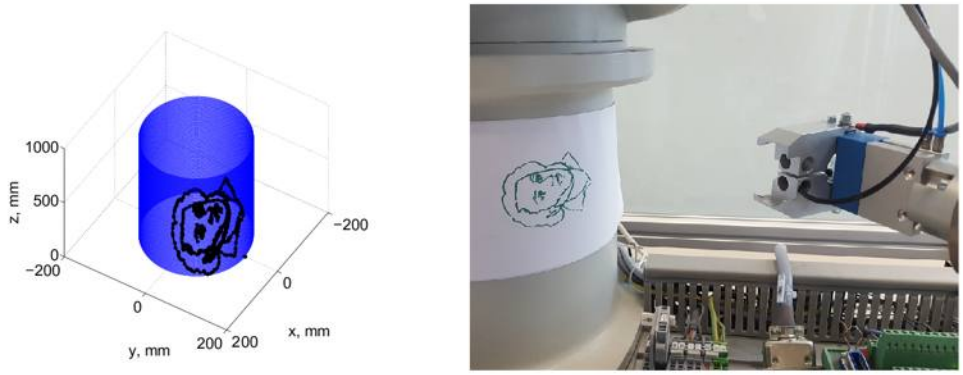
A portrait of Alexander Pushkin drawn by the robot on a flat surface

Experimental Results: Spatial Planning



A hypotrochoid drawn by the robot on the a curved (cylindrical) surface

Experimental Results: Spatial Planning



A portrait of Alexander Pushkin drawn by the robot on a curved (cylindrical) surface

Summary

- Reference motion can be programmed manually using a teach pendant or automatically using some path planning algorithm
- Once a path is generated, its intermediate positions, velocities and accelerations should be interpolated
- Advanced algorithms for spatial movement planning can be designed for industrial applications
- The next step is control design to make the robot to track the reference trajectory

Control Design for Industrial Robots

Dr. Oleg Borisov

PD Controller

Consider the control plant specified by the transfer function. We introduce a proportional-differential (PD) controller with a transfer function

$$R(s) = k_p + k_d s. \tag{1}$$

We calculate the transfer function of a closed-loop system

$$\begin{aligned} W(s) &= \frac{R(s)P(s)}{1 + R(s)P(s)} = \frac{\frac{k_p - k_d s}{Js^2 + Ks}}{1 + \frac{k_p + k_d s}{Js^2 + Ks}} = \\ &= \frac{k_p + k_d s}{Js^2 + (K + k_d)s + k_p}. \end{aligned} \tag{2}$$

PD Controller scheme

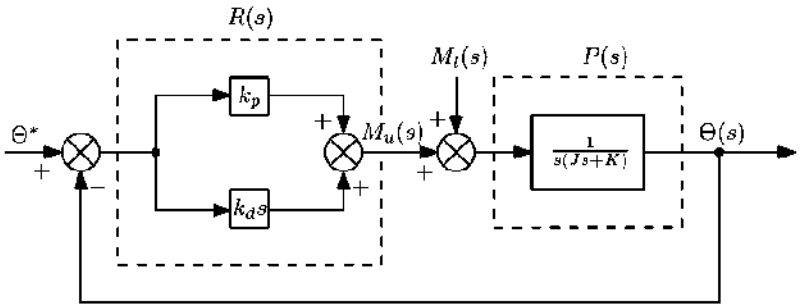


Figure 1: Simulation scheme of a closed-loop system with PD controller

Further, with known parameters of the object J & K, based on the roots of the characteristic polynomial of the transfer function $Js^2 + (K + k_d)s + k_p$, it is possible to calculate such coefficients of the PD controller k_p & k_d to ensure the required quality indicators of the closed system.

PID Controller

Consider the control plant given by the transfer function. We introduce a proportional-integral-differential (PID) controller with a transfer function

$$R(s) = k_p + k_i\frac{1}{s} + k_d s. \tag{3}$$

With structural transformations we express the output variable

$$\Theta(s) - \frac{R(s)P(s)}{1 + R(s)P(s)}\Theta^*(s) + \frac{P(s)}{1 + R(s)P(s)}M_l(s). \tag{4}$$

We calculate the transfer function of a closed-loop system

$$\begin{aligned} W(s) &= \frac{R(s)P(s)}{1 + R(s)P(s)} = \frac{\frac{k_d s^2 + k_p s + k_i}{Js^3 + Ks^2}}{1 + \frac{k_d s^2 + k_p s + k_i}{Js^3 + Ks^2}} = \\ &= \frac{k_d s^2 + k_p s + k_i}{Js^3 + (K + k_d)s^2 + k_p s + k_i}. \end{aligned} \tag{5}$$

PID Controller scheme

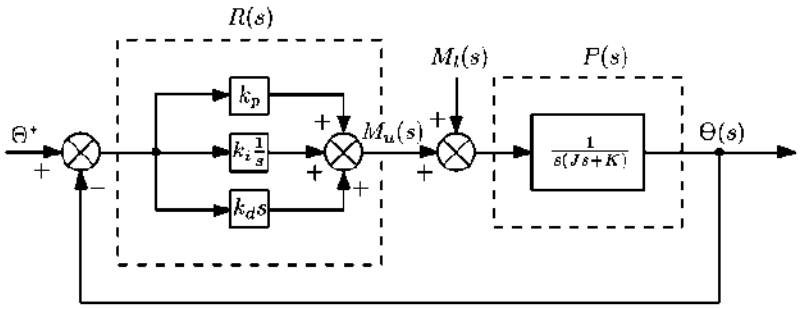


Figure 2: Simulation scheme of a closed-loop system with a PID controller

Further, with known parameters of the object J & K, based on the roots of the characteristic polynomial of the transfer function $Js^3 + (K + k_d)s^2 + k_p s + k_i$, it is possible to calculate such coefficients of the PID regulator k_p, k_i & k_d in order to ensure the required quality indicators of the closed system.

Robust control

Let us write down the consequtive compensator in the form of a transfer function

$$R(s) = k\gamma_0\sigma^{\rho-1}\frac{\alpha(s)}{\gamma(s)}, \tag{6}$$

where ρ — relative degree of the plant, k & $\sigma > k$ — tuning parameters of the controller, $\alpha(s)$ an arbitrary Hurwitz polynomial of degree $\rho - 1$, $\gamma(s)$ — Hurwitz polynomial of the form

$$\gamma(s) = s^{\rho-1} + \sigma\gamma_{\rho-2}s^{\rho-2} + \cdots + \sigma^{\rho-2}\gamma_1 s + \sigma^{\rho-1}\gamma_0. \tag{7}$$

Robust control in closed-loop system

Consider the control object specified by the transfer function. Its relative degree is $\rho = 2$, so, chosen $\alpha(s) = s + 1$ & $\gamma_0 = 1$, rewrite the regulator (6) like

$$R(s) = \frac{k\sigma s + k\sigma}{s + \sigma}. \tag{8}$$

Transfer function of a closed-loop system is

$$\begin{aligned} W(s) &= \frac{R(s)P(s)}{1 + R(s)P(s)} = \frac{\frac{k\sigma s + k\sigma}{(s+\sigma)(Js^2 + Ks)}}{1 + \frac{k\sigma s + k\sigma}{(s+\sigma)(Js^2 + Ks)}} = \\ &= \frac{k\sigma s + k\sigma}{(s+\sigma)(Js^2 + Ks) + k\sigma s + k\sigma}. \end{aligned} \tag{9}$$

Robust control scheme

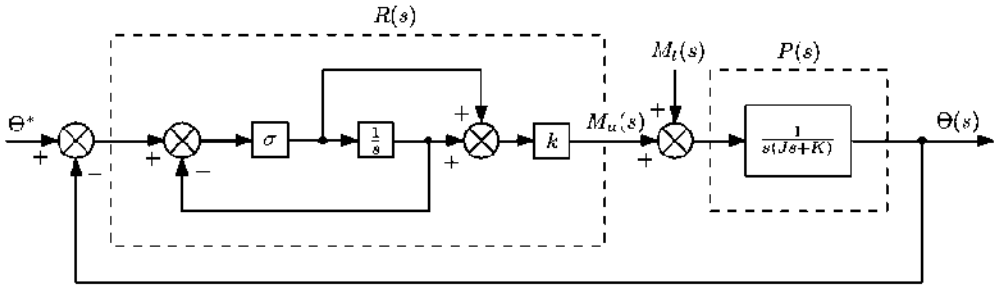


Figure 3: Simulation scheme of a closed-loop system with a consecutive compensator

The characteristic polynomial of the transfer function (9) contains unknown parameters of the plat, but due to the robustness of the regulator (8), for sufficiently large coefficients k & σ exponential stability is attained.

Robust control extended

Adding the integral component we rewrite the regulator (6)

$$R(s) = k\gamma_0\sigma^{\rho-1}\frac{\beta(s)}{s\gamma(s)}, \tag{10}$$

where $\beta(s)$ — Hurwitz polynomial of degree ρ.

Having chosen $\beta(s) = s^2 + s + 1$ & $\gamma_0 = 1$ rewrite the regulator (10) like

$$R(s) = \frac{k\sigma s^2 + k\sigma s + k\sigma}{s^2 \mid \sigma s}. \tag{11}$$

Transfer function of a closed-loop system

$$\begin{aligned} W(s) &= \frac{R(s)P(s)}{1 + R(s)P(s)} = \frac{\frac{k\sigma s^2+k\sigma s+k\sigma}{(s^2+\sigma s)(Js^2+Ks)}}{1 + \frac{k\sigma s^2+k\sigma s+k\sigma}{(s^2+\sigma s)(Js^2-Ks)}} = \\ &= \frac{k\sigma s^2 + k\sigma s + k\sigma}{(s^2 \mid \sigma s)(Js^2 \mid Ks) \mid k\sigma s^2 \mid k\sigma s \mid k\sigma} \end{aligned} \tag{12}$$

Robust control extended scheme

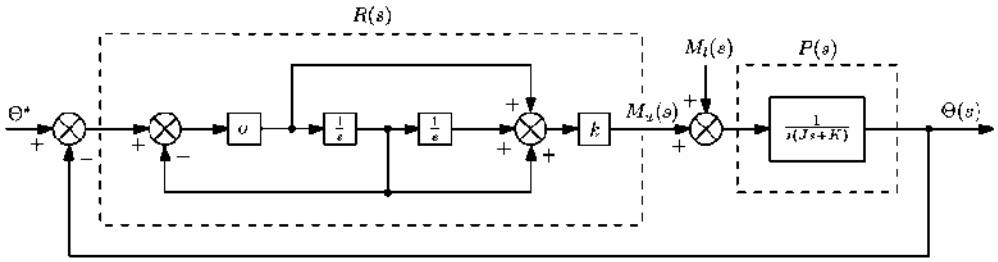


Figure 4: Simulation scheme for a closed-loop system with a consecutive compensator with integral loop

The increased order of astaticism of a system with a transfer function (12) makes it possible to compensate the effect of gravitational forces.

Anti-Windup Control

Saturated input

$$\hat{u}(t) = \operatorname{sat} u(t) = \begin{cases} u_t, & \text{if} \quad u(t) \geq u_t, \\ u(t), & \text{if} \quad u_b < u(t) < u_t, \\ u_b, & \text{if} \quad u(t) \leq u_b; \end{cases} \tag{13}$$

where u_t & u_b — upper and lower limits of the input signal.

Let us write down the control law of the PID controller (3) like

$$u(t) = k_p \tilde{q}(t) + k_i \frac{\tilde{q}(t)}{p} + k_d p \tilde{q}(t), \tag{14}$$

where $p = \frac{d}{dt}$ — differentiation operator, $\tilde{q}(t) = q^* - q(t)$ — error.

Following the antiwindup correction method we add to (14) an additional contour

$$u(t) = k_p \tilde{q}(t) + k_i \frac{\tilde{q}(t) + k_u \tilde{u}(t)}{p} + k_d p \tilde{q}(t), \tag{15}$$

where $k_u > 0$ — gain, $\tilde{u}(t) = \hat{u}(t) - u(t)$ — difference signal between saturated and source control.

Anti-Windup Control scheme

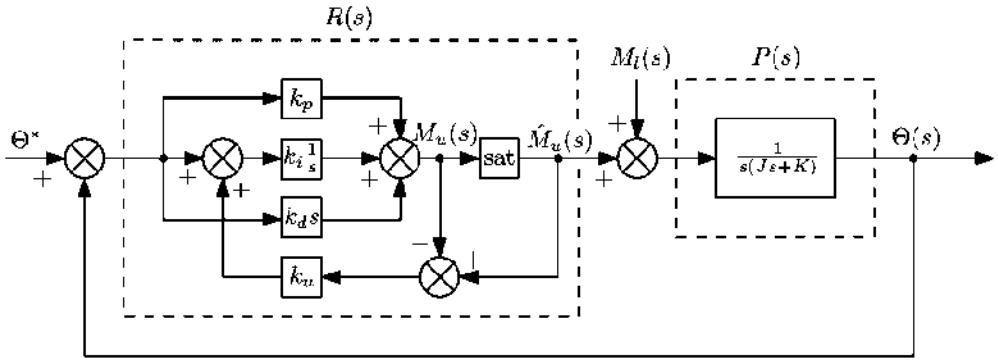


Figure 5: Simulation scheme of a closed-loop system with a PID controller and anti-windup correction

The control law (15) helps to avoid the effect of integral saturation in conditions of limited input.

Anti-Windup Robust Control

Let us write down the control law of a consequtive compensator with an integrated circuit (10) like

$$u(t) = k\frac{\beta(p)}{p}\hat{\tilde{q}}(t), \tag{16}$$

$$\dot{\xi}(t) = \sigma(\Gamma\xi(t) + d\gamma_0\tilde{q}(t)), \tag{17}$$

$$\hat{\tilde{q}}(t) = h^T\xi(t), \tag{18}$$

where $\hat{\tilde{q}}(t)$ — error signal estimation $\tilde{q}(t)$, matrices and vectors Γ, d, h in form

$$\Gamma = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -\gamma_0 & -\gamma_1 & -\gamma_2 & \dots & -\gamma_{\rho-1} \end{bmatrix}, \; d = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}, \; h = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{19}$$

Anti-Windup Robust Control

Transform the control law (16), with integrator

$$u(t) = k\frac{\beta(p)}{p}\hat{\tilde{q}}(t) = k\left(\bar{\beta}(p) + \frac{\beta_0}{p}\right)\hat{\tilde{q}}(t) = k\bar{\beta}(p)\hat{\tilde{q}}(t) + k\frac{\beta_0}{p}\hat{\tilde{q}}(t), \tag{20}$$

where $\bar{\beta}(p) = \frac{\beta(p)-\beta_0}{p}$.

Following the anti-windup correction method we add to the (20) an additional contour

$$u(t) = k\bar{\beta}(p)\hat{\tilde{q}}(t) + k\frac{\beta_0}{p}\left(\hat{\tilde{q}}(t) + k_u\breve{u}(t)\right), \tag{21}$$

where $k_u > 0$ — gain, $\breve{u}(t) = \hat{u}(t) - u(t)$ — difference signal between saturated and source control.

Having chosen $\beta(p) = p^2 + p + 1$ & $\gamma_0 = 1$, rewrite the regulator (21) like

$$u(t) = kp\hat{\tilde{q}}(t) + k\dot{\hat{\tilde{q}}}(t) + k\frac{1}{p}(\hat{\tilde{q}}(t) + k_u\breve{u}(t)). \tag{22}$$

Anti-Windup Robust Control

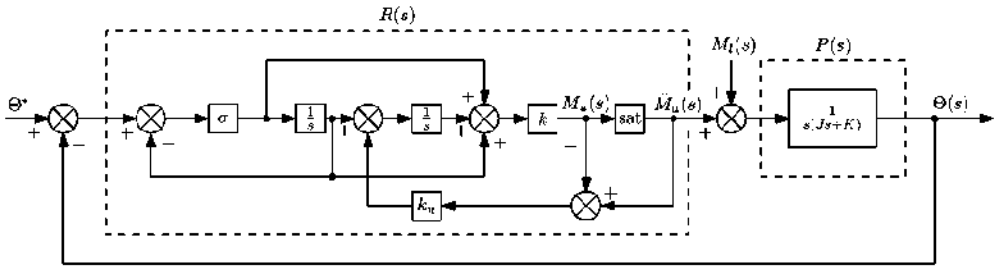


Figure 6: Simulation scheme of a closed-loop system with a consecutive compensator and anti-windup correction

The regulator (22) allows to solve the stabilization problem with the increased order of astaticism in comparison with the regulator (6) and with compensation of the integral saturation effect by means of anti-windup.

Tracking control

Let's express the output signal $\Theta(s)$:

$$\Theta(s) = \frac{R(s)P(s) + F(s)P(s)}{1 + R(s)P(s)}\Theta^*(s) + \frac{P(s)}{1 + R(s)P(s)}M_l(s). \qquad (23)$$

We choose the transfer function of direct coupling in the form:

$$F(s) = \frac{1}{P(s)}, \qquad (24)$$

then the expression (23) takes the form:

$$\Theta(s) = \Theta^*(s) + + \frac{P(s)}{1 + R(s)P(s)}M_l(s). \qquad (25)$$

Tracking control scheme

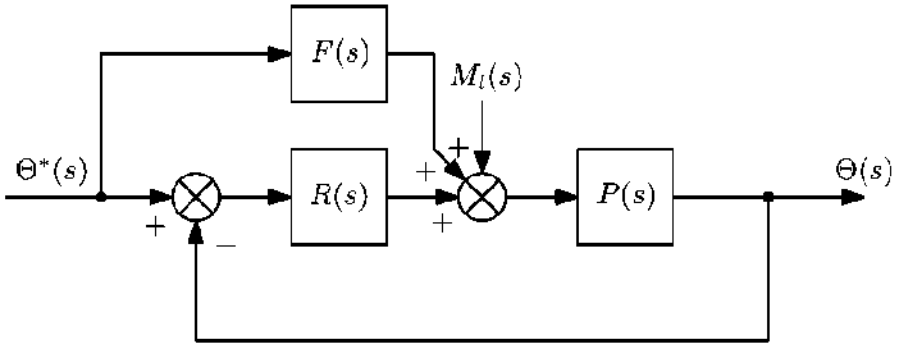


Figure 7: Simulation scheme for closed-loop tracking system

Direct link allows the system to monitor any given trajectory, provided that the system is completely stable. The steady-state error in this case will be due only to the influence of an external perturbation $M_l(s)$.

Multivariable control

Consider the dynamic model of a robotic system

$$\Gamma(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = u. \tag{26}$$

Stabilization of desired q^* will be performed with PD controller.

First, for simplicity, we neglect the effect of gravity, assuming that $G(q) = 0$. In view of this model (26) looks like

$$\Gamma(q)\ddot{q} + C(q,\dot{q})\dot{q} = u. \tag{27}$$

We choose the vector of control actions u like

$$u = K_p(q^* - q(t)) - K_d\dot{q}(t) = K_p\tilde{q}(t) + K_d\dot{\tilde{q}}(t), \tag{28}$$

where $\tilde{q}(t) = q^* - q(t)$ — error between the specified configuration and the current one, K_p & K_d looks like

$$K_p = \begin{bmatrix} k_{p,1} & 0 & \dots & 0 \\ 0 & k_{p,2} & \dots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & k_{p,n} \end{bmatrix}, \; K_d = \begin{bmatrix} k_{d,1} & 0 & \dots & 0 \\ 0 & k_{d,2} & \dots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & k_{d,n} \end{bmatrix}. \tag{29}$$

Multivariable control

Substituting the control law (28) to the plant (27) we obtain a model of a closed system

$$\Gamma(q)\ddot{q} + C(q,\dot{q})\dot{q} = K_p\tilde{q}(t) + K_d\dot{\tilde{q}}(t). \tag{30}$$

To analyze the stability of a closed-loop system (30) we consider the candidatc Lyapunov function in quadratic form

$$V(t) = \frac{1}{2}\tilde{q}^T K_p\tilde{q} + \frac{1}{2}\dot{\tilde{q}}^T\Gamma\dot{\tilde{q}}. \tag{31}$$

Taking the time derivative of (31) we get

$$\dot{V}(t) = -\dot{\tilde{q}}^T K_d\dot{\tilde{q}} \leq 0, \tag{32}$$

this together with the Lassalle theorem shows the asymptotic stability of a closed system (30).

Multivariable control

When $\dot{V} = 0$ from (32) we can conclude that the generalized velocities and accelerations are zero $\dot{q}(t) = 0$ & $\ddot{q}(t) = 0$. Taking this into account we rewrite the equation of a closed system for $t \to \infty$

$$0 = K_p\tilde{q}(t), \tag{33}$$

from which it follows that $\tilde{q}(t) = q^* - q(t) = 0$ with $t \to \infty$.

The influence of gravity $G(q) \neq 0$ leads to the appearance of a steady error. The PD controller in this case does not provide asymptotic stability. The equation (33) looks like

$$G(q) = K_p\tilde{q}(t). \tag{34}$$

To eliminate the established error we supplement the law of control

$$u - K_p\tilde{q}(t) + K_d\dot{\tilde{q}}(t) + G(q), \tag{35}$$

which makes it possible to provide asymptotic stability with the influence of gravity.

Dynamic of robotic systems

Dynamics of Robotic Systems
Euler-Lagrange Method and Special Cases

Sergey Kolyubin

Outline

- Motivation
- Energy-based Approach - Euler-Lagrange Method
  - Energy calculation
  - Motion equation
- Special Cases
  - Drive dynamics
  - Flexible joints modeling
- Motion Equation in Operational Space

Why Do We Need to Know Dynamics?

- simulation
- defining dynamic constraints
- mechanical design optimization
- trajectory planners and controllers synthesis

Tasks

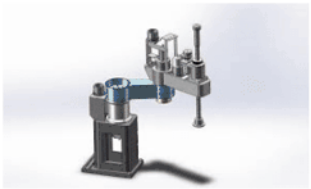
- Forward Dynamics: given desired trajectory (coordinates, velocities, acceleration) find generalized forces/torques
- Inverse Dynamics: given generalized forces/torques, find generated motion (trajectory)

Tasks

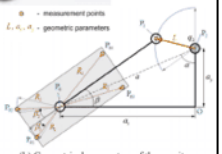
- Forward Dynamics: given desired trajectory (coordinates, velocities, acceleration) find generalized forces/torques
- Inverse Dynamics: given generalized forces/torques, find generated motion (trajectory)

Practical tasks

- f/t calculation - find external (control design) and internal (find reaction forces in kinematic pairs)
- performance indicators - find possible cycle time given dynamic constraints
- (serial) manipulators balancing - unload drives in statics
- (parallel) manipulators dynamic balancing - minimize distortions during the motion by placing counter-weights



(a) gravity compensator of KUKA KR-270 TM robot



(b) Geometrical parameters of the gravity compensator and measurement point locations

Tasks

- Forward Dynamics: given desired trajectory (coordinates, velocities, acceleration) find generalized forces/torques
- Inverse Dynamics: given generalized forces/torques, find generated motion (trajectory)

Theoretical sub-tasks

- trajectories calculation
- motion stability analysis
- calculating time response
- identifying critical motion modes

Methods Comparison

- E-L - kinetic and potential energy
  - multibody dynamics as a whole
  - exclude reaction forces between links
  - symbolic form
  - better for analysis
- N-E - forces/torques balance
  - separate equation for each body
  - explicit relations for reaction forces
  - numeric recursion form
  - better for synthesis and real-time applications

By excluding reaction forces and substituting these relations we can derive E-L equations from N-E equations

E-L General Framework

1. select generalized coordinates $q_1, q_2, \dots q_n$
2. derive relations for kinetic $\mathcal{K}$ and potential $\mathcal{P}$ energy as functions of generalized coordinates and its derivatives
3. calculate system Lagrangian $\mathcal{L}$
4. derive motion equation

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_k} - \frac{\partial L}{\partial q_k} = \tau_k,\ k = 1,2,\dots n \qquad (1)$$

τ_k is a generalized force/torque

Full Kinetic Energy

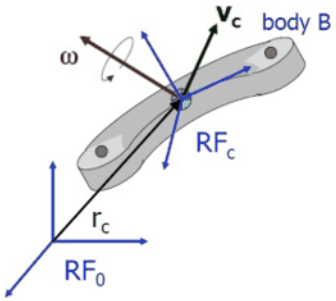


Figure 1: ©DeLuca

Konig theorem
Full energy consist of an energy assoc. with body CoM motion and relative body motion around it CoM

$$\mathcal{K} = \frac{1}{2} m |v|^2 + \frac{1}{2} \omega^T \mathcal{I} \omega$$

where m is a body mass, v and ω are linear and rotational velocities vectors, $\mathcal{I}$ is an inertia tensor

All values in the same CF

Formula for rotational velocity

$$\omega \leftarrow S(\omega) = \dot{R}(t) R^T(t),$$

where R is a rotation matrix from body frame to inertial frame

Kinetic Energy of n-links Robot

Sum of kinetic energy of linear and rotational motions

$$\mathcal{K} = \tfrac{1}{2} m |v_c|^2 + \tfrac{1}{2} \omega^T \mathcal{I} \omega$$

CoM velocities

- $v_c = \dot{r}_c$ and ω are functions of generalized coordinates q and velocities $\dot{q}$

Kinetic Energy of n-links Robot

Sum of kinetic energy of linear and rotational motions

$$\mathcal{K} = \tfrac{1}{2} m |v_c|^2 + \tfrac{1}{2} \omega^T \mathcal{I} \omega$$

CoM velocities

- $v_c = \dot{r}_c$ and ω are functions of generalized coordinates q and velocities $\dot{q}$

Relations can be computed via Jacobian assoc. with links CoMs

$$v_{c,i} = J_{v_i}(q)\dot{q}, \quad \omega_i = J_{\omega_i}(q)\dot{q}$$

Kinetic Energy of n-links Robot

Sum of kinetic energy of linear and rotational motions

$$\mathcal{K} = \tfrac{1}{2} m |v_c|^2 + \tfrac{1}{2} \omega^T \mathcal{I} \omega$$

CoM velocities

- $v_c = \dot{r}_c$ and ω are functions of generalized coordinates q and velocities $\dot{q}$

Relations can be computed via Jacobian assoc. with links CoMs

$$v_{c,i} = J_{v_i}(q)\dot{q}, \quad \omega_i = J_{\omega_i}(q)\dot{q}$$

Robot kinetic energy

$$\mathcal{K} = \frac{1}{2}\dot{q}^T \left[\sum_{i=1}^{n} m_i J_{v_i}(q)^T J_{v_i}(q) + J_{\omega_i}(q)^T R_i(q) I R_i(q)^T J_{\omega_i}(q) \right] \dot{q}$$

Recurrent Velocities Formulas

- rotation (angular) velocities

$$\omega_i = \left(R_i^{i-1}(q_i)\right)^T \cdot \omega_{i-1} + (1-\sigma_i)\dot{q}_i z_{i-1}] = \left(R_i^{i-1}(q_i)\right)^T \omega_i^{i-1}. \quad (2)$$

where $R_i^{i-1}(q_i)$ is a rotation matrix for neighbor CFs $O_{i-1}x_{i-1}y_{i-1}z_{i-1}$ and $O_i x_i y_i z_i$,

$$\sigma_i = \begin{cases} 0, \textit{for rotational joint}, \\ 1, \textit{for prismatic joint}, \end{cases}$$

$z_{i\ 1} = [0\,0\,1]^T$ is a vector of z axis coord. if D-H convention used, ω_i^{i-1} is a rotational velocity of i-th link with respect to CF $O_{i-1}x_{i-1}y_{i-1}z_{i-1}$

- linear velocities

$$v_{c,i} = v_i + \omega_i \times r_{c,i}, \quad (3)$$

where

$$v_i = \left(R_i^{i-1}(q_i)\right)^T \left[v_{i-1} + \sigma_i \dot{q}_i z_{i-1} + \omega_i^{i-1} \times r_{i-1,i}^{i-1}\right] \quad (4)$$

denotes linear velocity of a CF origin O_i, $r_{c,i}$ is a CoM vector for i-th link with respect to O_i, $r_{i-1,i}^{i\ 1}$ are coordinates of radius-vectors from O_{i-1} to O_i with respect to CF $O_{i-1}x_{i-1}y_{i-1}z_{i-1}$.

Potential Energy of n-links Robot

Potential energy of i-th link

$$\mathcal{P}_i = m_i g^T r_{c,i}$$

where $r_{c,i}$ is a CoM coordinates vector

$$\begin{vmatrix} r_{c,i} \\ 1 \end{vmatrix} = {}^0H_1(q_1)\,{}^1H_2(q_2)\cdots{}^{i-1}H_i(q_i) \begin{vmatrix} {}^i r_{c,i} \\ 1 \end{vmatrix} \quad (5)$$

Robot potential energy

$$\mathcal{P} = \sum_{i=1}^{n} \mathcal{P}_i = \sum_{i=1}^{n} m_i g^T r_{c,i}$$

For a serial kinematic chain

$$\mathcal{P} = \sum_{i=1}^{n} \mathcal{P}_i$$

$$\mathcal{P}_i = \mathcal{P}_i(q_j, j \le i)$$

Motion Equation

- Kinetic energy

$$\begin{aligned}\mathcal{K} &= \frac{1}{2}\dot{q}^T\left[\sum_{i=1}^{n} m_i J_{v_i}(q)^T J_{v_i}(q) + J_{\omega_i}(q)^T R_i(q) I R_i(q)^T J_{\omega_i}(q)\right]\dot{q} \\ &= \frac{1}{2}\dot{q}^T M(q)\dot{q} = \frac{1}{2}\sum_{k,j} m_{kj}(q)\dot{q}_k\dot{q}_j\end{aligned}$$

- for conservative generalized forces $\psi_k = -\frac{\partial \mathcal{P}}{\partial q_k} + \tau_k$
- system Lagrangian $\mathcal{L} = \mathcal{K} - \mathcal{P}$

$$\frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{q}_k} - \frac{\partial \mathcal{L}}{\partial q_k} = \tau_k$$

$$\frac{d}{dt}\frac{\partial(\mathcal{K}-\mathcal{P})}{\partial \dot{q}_k} - \frac{\partial(\mathcal{K}-\mathcal{P})}{\partial q_k} = \tau_k$$

$$\frac{d}{dt}\frac{\partial \mathcal{K}}{\partial \dot{q}_k} - \frac{\partial(\mathcal{K}-\mathcal{P})}{\partial q_k} = \tau_k$$

Motion Equation (contd.)

Equation structure

$$\frac{d}{dt}\frac{\partial \mathcal{K}}{\partial \dot{q}_k} - \frac{\partial(\mathcal{K}-\mathcal{P})}{\partial q_k} = \tau_k, \qquad k = 1, \ldots, n$$

1st term

$$\frac{\partial \mathcal{K}}{\partial \dot{q}_k} = \frac{\partial}{\partial \dot{q}_k}\left[\frac{1}{2}\dot{q}^T M(q)\dot{q}\right] = \sum_{j=1}^{n} m_{kj}\dot{q}_j$$

and

$$\begin{aligned}\frac{d}{dt}\frac{\partial \mathcal{K}}{\partial \dot{q}_k} = \frac{d}{dt}\left[\sum_{j=1}^{n} m_{kj}\dot{q}_j\right] &= \sum_{j=1}^{n} m_{kj}\ddot{q}_j + \sum_{j=1}^{n}\frac{d}{dt}\left[m_{kj}(q)\right]\dot{q}_j \\ &= \sum_{j=1}^{n} m_{kj}\ddot{q}_j + \frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n}\left(\frac{\partial m_{kj}}{\partial q_i} + \frac{\partial m_{ki}}{\partial q_j}\right)\dot{q}_i\dot{q}_j\end{aligned}$$

Motion Equation (contd.)

2nd term

$$\frac{\partial(\mathcal{K}-\mathcal{P})}{\partial q_k} = \frac{\partial}{\partial q_k}\left[\frac{1}{2}\dot{q}M(q)\dot{q}-\mathcal{P}\right] = \frac{1}{2}\dot{q}\left[\frac{\partial}{\partial q_k}M(q)\right]\dot{q}-\frac{\partial}{\partial q_k}\mathcal{P}$$
$$= \frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n}\frac{\partial m_{ij}}{\partial q_k}\dot{q}_i\dot{q}_j-\frac{\partial}{\partial q_k}\mathcal{P}$$

Motion Equation (contd.)

2nd term

$$\frac{\partial(\mathcal{K}-\mathcal{P})}{\partial q_k} = \frac{\partial}{\partial q_k}\left[\frac{1}{2}\dot{q}M(q)\dot{q}-\mathcal{P}\right] = \frac{1}{2}\dot{q}\left[\frac{\partial}{\partial q_k}M(q)\right]\dot{q}-\frac{\partial}{\partial q_k}\mathcal{P}$$
$$= \frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n}\frac{\partial m_{ij}}{\partial q_k}\dot{q}_i\dot{q}_j-\frac{\partial}{\partial q_k}\mathcal{P}$$

Resulting relations

$$\sum_{j=1}^{n}m_{kj}\ddot{q}_j+\frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n}\left(\frac{\partial m_{kj}}{\partial q_i}+\frac{\partial m_{ki}}{\partial q_j}\right)\dot{q}_i\dot{q}_j$$
$$-\frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n}\frac{\partial m_{ij}}{\partial q_k}\dot{q}_i\dot{q}_j+\frac{\partial}{\partial q_k}\mathcal{P}=\tau_k$$

Motion Equation (contd.)

2nd term

$$\begin{aligned}\frac{\partial(\mathcal{K}-\mathcal{P})}{\partial q_k} &= \frac{\partial}{\partial q_k}\left[\frac{1}{2}\dot{q}M(q)\dot{q}-\mathcal{P}\right] = \frac{1}{2}\dot{q}\left[\frac{\partial}{\partial q_k}M(q)\right]\dot{q}-\frac{\partial}{\partial q_k}\mathcal{P}\\ &= \frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n}\frac{\partial m_{ij}}{\partial q_k}\dot{q}_i\dot{q}_j-\frac{\partial}{\partial q_k}\mathcal{P}\end{aligned}$$

Resulting relations

$$\sum_{j=1}^{n} m_{kj}(q)\ddot{q}_j+\sum_{j=1}^{n}\sum_{i=1}^{n} c_{ijk}(q)\dot{q}_i\dot{q}_j+g_k(q)=\tau_k\,,\quad k=1,\dots,n$$

where $c_{ijk}=c_{jik}$ is a Christoffel symbol and

$$c_{ijk}(q)=\frac{1}{2}\left(\frac{\partial m_{kj}}{\partial q_i}+\frac{\partial m_{ki}}{\partial q_j}-\frac{\partial m_{ij}}{\partial q_k}\right),\quad g_k(q)=\frac{\partial}{\partial q_k}\mathcal{P}$$

is a potential energy gradient

Motion Equation (contd.)

2nd term

$$\begin{aligned}\frac{\partial(\mathcal{K}-\mathcal{P})}{\partial q_k} &= \frac{\partial}{\partial q_k}\left[\frac{1}{2}\dot{q}M(q)\dot{q}-\mathcal{P}\right] = \frac{1}{2}\dot{q}\left[\frac{\partial}{\partial q_k}M(q)\right]\dot{q}-\frac{\partial}{\partial q_k}\mathcal{P}\\ &= \frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n}\frac{\partial m_{ij}}{\partial q_k}\dot{q}_i\dot{q}_j-\frac{\partial}{\partial q_k}\mathcal{P}\end{aligned}$$

Resulting relations

$$\sum_{j=1}^{n} m_{kj}(q)\ddot{q}_j+\sum_{j=1}^{n}\sum_{i=1}^{n} c_{ijk}(q)\dot{q}_i\dot{q}_j+g_k(q)=\tau_k\,,\quad k=1,\dots,n$$

in a vectorial form

$$M(q)\ddot{q}+C(q,\dot{q})\dot{q}+G(q)=\tau$$

with Coriolis and centrifugal forces $C(q)$, $c_{kj}=\sum_{i=1}^{n} c_{ijk}(q)\dot{q}_i$.

Accounting for Gear and Motor Dynamics

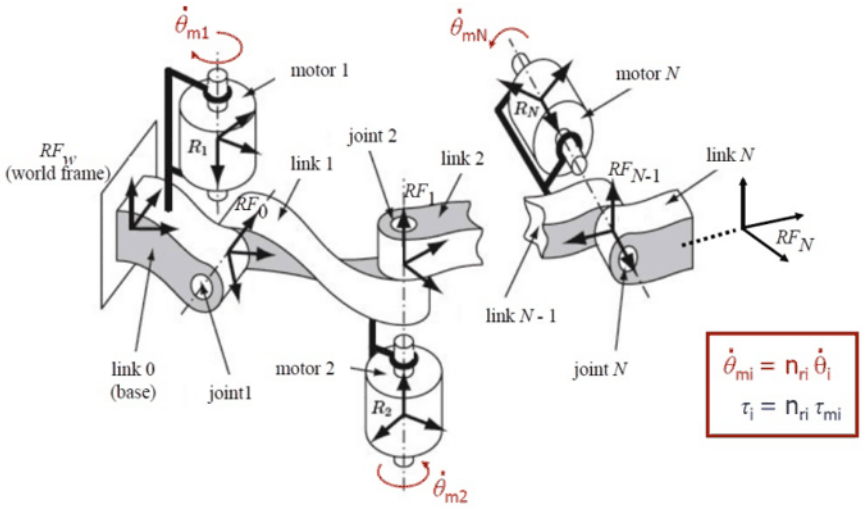
Assumptions

- drive is fixed to the link preceding the link it is moving
- motor and joint axis are coinciding

General considerations

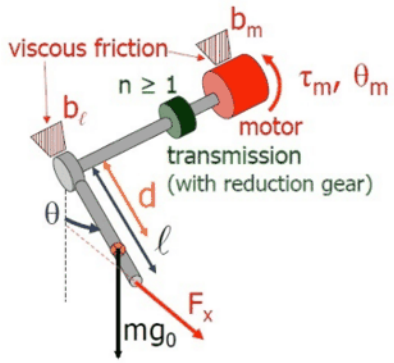
- drive mass should be added to link mass
- drive rotor inertia should be taking into account when computing total kinetic energy
- gear ration should be taken into account when computing velocities and forces

Motor Placement



©DeLuca

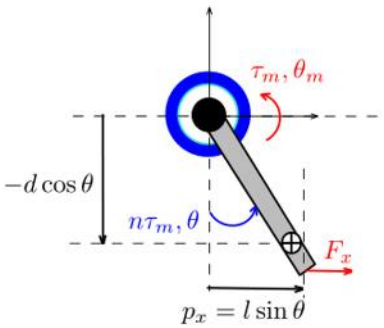
Pendulum with Gear



©DeLuca

I_l – link moment of inertia w.r.t. its CoM
m – link mass
d – distance from axis of rotation to link CoM
$\dot{\theta}$ – link rotation velocity (after gear)
$\dot{\theta}_m = n\dot{\theta}$ – motor rotation velocity (before gear)
n – gear ratio
I_m – drive moment of inertia w.r.t its axis of rotation

Kinetic Energy



Pendulum kinetic energy

$$\mathcal{K}_l = \frac{1}{2}\left(I_l + md^2\right)\dot{\theta}^2,$$

Drive kinetic energy

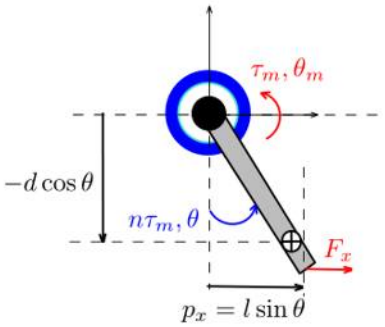
$$\mathcal{K}_m = \frac{1}{2} I_m \dot{\theta}_m^2,$$

Total kinetic energy

$$\mathcal{K} = \mathcal{K}_l + \mathcal{K}_m = \frac{1}{2} I \dot{\theta}^2,$$

where $I = I_l + md^2 + n^2 I_m$ is a total moment of inertia w.r.t. axis of rotation

Potential Energy and Lagrangian



Total potential energy

$$\mathcal{P} = \mathcal{P}_0 - mg_0 d\cos\theta.$$

System Lagrangian

$$\mathcal{L} = \frac{1}{2} I\dot{\theta}^2 + mg_0 d\cos\theta - \mathcal{P}_0.$$

Motion Equation

From the link side

$$I\ddot{\theta} + mg_0 d\sin\theta = \tau.$$

From the motor side

$$\frac{I}{n^2}\ddot{\theta}_m + \frac{m}{n} g_0 d\sin\frac{\theta_m}{n} = \tau_m - \left(\frac{k_{fl}}{n^2} + k_{fm}\right)\dot{\theta}_m + \frac{l}{n}\cos\frac{\theta_m}{n} F_x.$$

Friction Forces

General considerations

- is a dissipative force
- localized in joints
- static model captures major influence for relatively fast motion

$$\tau - n\tau_m - k_{fl}\dot{\theta} - nk_{fm}\dot{\theta}_m + \dot{p}_x F_x - n\tau_m - (k_{fl} + n^2 k_{fm})\dot{\theta} + l\cos\theta F_x,$$

where τ_m is drive torque before gear, k_{fm} and k_{fl} are viscous friction coefficients

- dynamic models are more accurate, but usually hard to identify

Flexible-Joints Robots

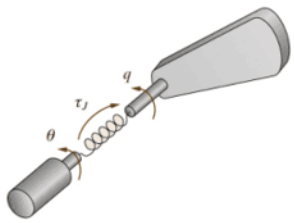


Figure 2: Flexible joint sketch

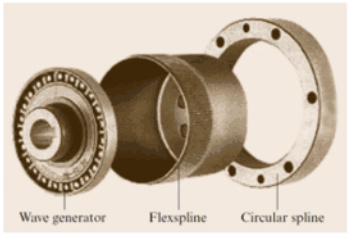


Figure 3: Harmonic drive

Flexible joints
Motor (input) and link (output) are connected by a flexible (deformable) element

- long shaft
- harmonic drive gearbox
- belts

Useful flexibility

1. physically (VSA, SEA)
2. on a software level

for

- safe pHRI
- explosive motions

Modeling Flexible Joints

Assumptions

1. flexibility is localized in joint
2. small deformations for linear spring model
3. symmetric drive shafts with CoM on the axis of rotation
4. drive is located before the link it is actuating

Modeling Flexible Joints

- introduce $2n$ generalized coordinates $q \in R^n$ for links and $\theta \in R^n$ for drives ($\theta_i = \theta_{mi}/r_i$, r_i is a gear ratio)
- add drive kinetic energy

$$\mathcal{K}_{mi} = \frac{1}{2}\mathcal{I}_m\dot{\theta}_{mi}^2 = \frac{1}{2}\mathcal{I}_m r_i^2\dot{\theta}_i^2$$

$$\mathcal{K}_m = \sum_{i=1}^{n}\mathcal{K}_{mi} = \frac{1}{2}\dot{\theta}^\top M_m\dot{\theta}$$

  M_m is a diagonal drive inertia matrix

- add potential energy of a deformed spring

$$\mathcal{P}_{ei} = \frac{1}{2}K_i(q_i - \theta_i)^2$$

$$\mathcal{P}_e = \sum_{i=1}^{n}\mathcal{P}_{ei} = \frac{1}{2}(q-\theta)^T K(q-\theta)$$

  K is a matrix of joint stiffness coefficients

Modeling Flexible Joints

Motion equation

$$M(q)\ddot{q} - c(q,\dot{q}) + G(q) + K(q-\theta) = 0,$$
$$M_m\ddot{\theta} + K(\theta - q) = \tau$$

Operational Space Formulation

- Configuration space

$$M(q)\ddot{q} + c(q,\dot{q}) + g(q) = \tau$$

- Operational space

$$\Lambda\ddot{x}_e + \mu + \rho = F_e$$

Operational Space Formulation

- Configuration space

$$M(q)\ddot{q} + c(q,\dot{q}) + g(q) = \tau$$

- Operational space

$$\Lambda\ddot{x}_e + \mu + \rho = F_e$$

- projecting joint forces/torques to end-effector forces

$$\tau = J_e^T F_e$$

- kinematic relations

$$\dot{x}_e = J_e\dot{q} \Rightarrow \ddot{x}_e = J_e\ddot{q}_e + \dot{J}_e\dot{q}_e$$

$$\ddot{x}_e = J_e M^{-1}\left(J_e^T F_e - (c(q,\dot{q}) + g(q))\right) + \dot{J}_e\dot{q}_e \Rightarrow$$

$$\ddot{x}_e + J_e M^{-1}(c(q,\dot{q}) + g(q)) - \dot{J}_e\dot{q}_e = J_e M^{-1} J_e^T F_e$$

- operational-space model

$$\Lambda = \left(J_e M^{-1} J_e^T\right)^{-1} \qquad \mu = \Lambda J_e M^{-1} c(q,\dot{q}) - \Lambda\dot{J}_e\dot{q}_e \qquad \rho = \Lambda J_e M^{-1} g(q)$$

Trajectory control algorithms

Trajectory control algorithms based on stabilisation of sets

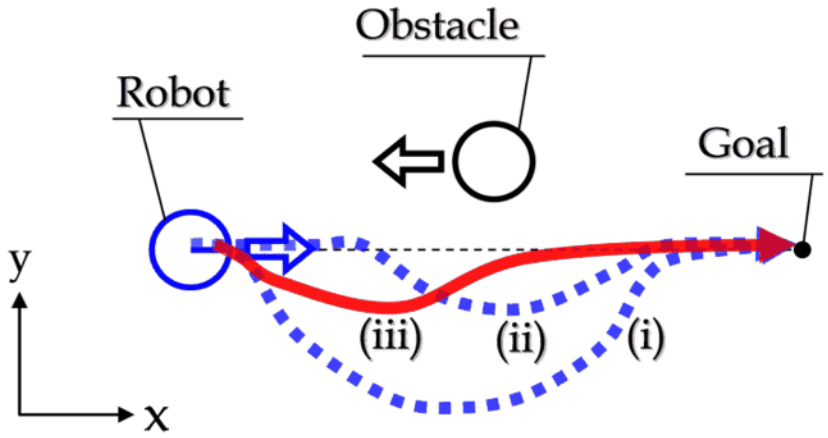
Aleksandr Y. Krasnov

Examples of problems

Autonomous vehicles control

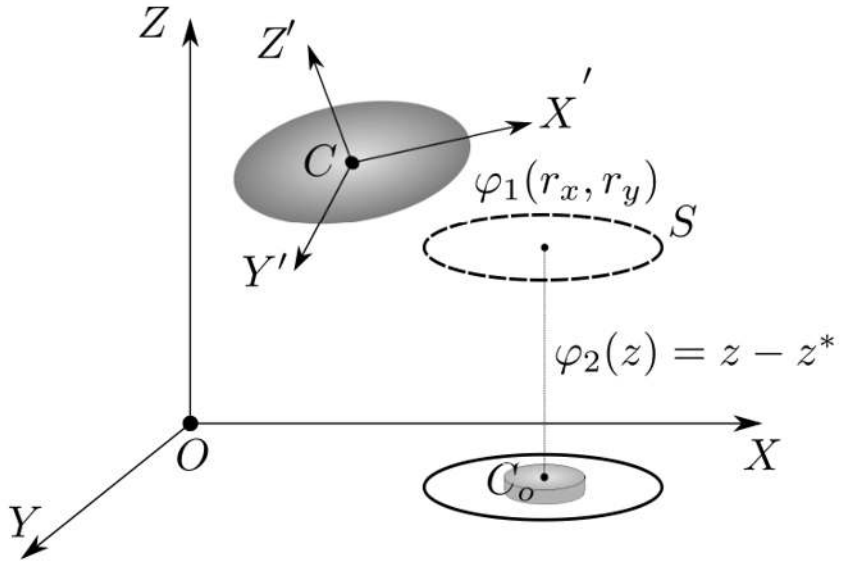
Examples of problems

Collision avoidance:



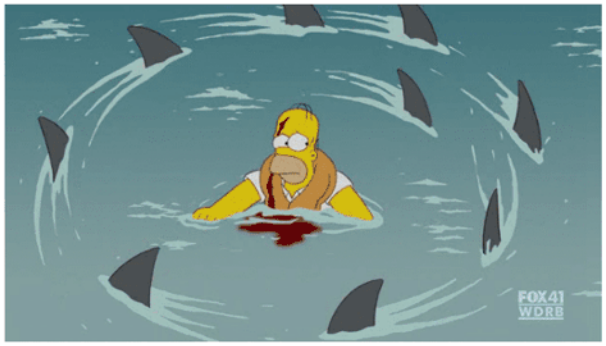
Examples of problems

Following the moving target:



Examples of problems

Control of underwater biomimetic robots (Robotic Fish)



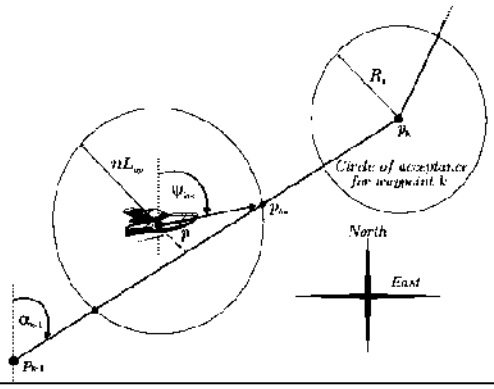
Problem: robot can not stop after reaching a goal.
Possible solution: continue motion along closed curve around the the goal

Possible solutions: Tracking approaches

- Virtual Target Tracking:
  - Backstepping based

    Aguiar, A.P.; Hespanha, J.P., Kokotovic, P.V., "Path-following for nonminimum phase systems removes performance limitations," Automatic Control, IEEE Transactions on , vol.50, no.2, pp.234,239, Feb. 2005
  - LOS(Line-of-Sight) methods

    M. Breivik and T.I. Fossen Principles of Guidance Based Path Following in 2D and 3D Proceedings of the IEEE Conference on Decision and Control, Seville, Spain,2005,pp. 627-634



Possible solutions: Set stabilization approaches

- Sliding mode

  Ashrafiuon, H., Muske, K. R., McNinch, L. C., and Soltan, R., "Sliding Model Tracking Control of Surface Vessels," IEEE Transactions on Industrial Electronics SS on Sliding Mode Control in Industrial Applications, 2008

- Passification

  M. El-Hawwary, M. Maggiore, Case Studies on Passivity-Based Stabilization of Closed Sets, International Journal of Control, 2011

- Feedback linearization:
  - Methods of transversal linearization

    Nielsen, C.; Fulford, C.; Maggiore, M., "Path following using transverse feedback linearization: Application to a maglev positioning system," American Control Conference, 2009
  - Vector Field Path Following

    Nelson, D.R.; Barber, D.B.; McLain, T.W.; Beard, R.W., "Vector field path following for small unmanned air vehicles," American Control Conference, 2005
  - Coordination control by Iliya V. Miroshnik

Description of the desired trajectory

Three ways to describe straight line:

- Explicit description

$$y = \frac{ax - c}{b}$$

- Implicit description

$$ax + by + c = 0$$

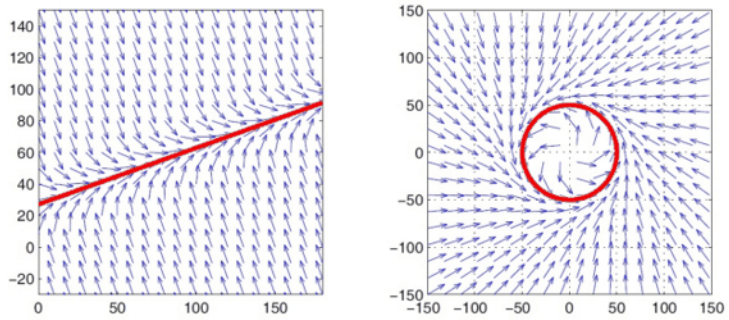
- Parametric description

$$x = x_0 + ft$$

$$y = y_0 + gt$$

Main ideas of methods based on the stabilization of sets

- Implicit representation of curve
- Dependence on the current position in the space
- Invariance of desired path(an attractor in the output space)
- Potentially higher accuracy of motion



Sujit, P.B.; Saripalli, S.; Sousa, J.B., "An evaluation of UAV path following algorithms," Control Conference (ECC), 2013

Nelson, D.R.; Barber, D.B.; McLain, T.W.; Beard, R.W., "Vector field path following for small unmanned air

Formal statement of control problem

- Geometric sub-task:

$$dist\,(p - f(p_d)) \to 0,$$

where $f(p_d)$ - a desired path of motion, p_d - spatial coordinates of space, p - current position of a plant.

- Kinematic sub-task - maintenance of desired velocity of motion along the path:

$$\lim_{t\to\infty} \Delta V = \lim_{t\to\infty} (V - V^*) \to 0,$$

where V - current velocity of motion, V^* - desired velocity.

Motion on the plane

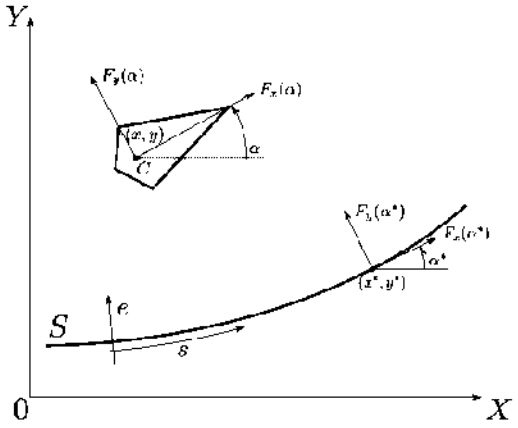
Dynamic model of robot motion:

$$\dot{v}_x = v_y\omega + \frac{1}{m}F_x, \qquad (1)$$

$$\dot{v}_y = -v_x\omega + \frac{1}{m}F_y, \qquad (2)$$

$$\dot{\omega} = \frac{1}{J}M_c, \qquad (3)$$

where v_x and v_y are linear velocities,
F_x and F_y are control forces,
ω is the angular velocity,
m is the mass of the plant,
J is the moment of inertia,
M_c is the control torque.



Motion on the plane

Relation of linear velocities in the fixed and absolute frames:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = T^T(\alpha) \begin{bmatrix} v_x \\ v_y \end{bmatrix}, \qquad (4)$$

where $T^T(\alpha) = \begin{bmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{bmatrix}$ is the rotational matrix of C-fixed frame.

Linear accelerations in absolute frame

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix} = \frac{1}{m}T^T(\alpha) \begin{bmatrix} F_x \\ F_y \end{bmatrix}. \qquad (5)$$

Motion on the plane

The desired path is an implicitly described smooth segment of curve S:

$$\varphi(x, y) = 0, \tag{6}$$

and relevant local coordinate s (path length) is defined as

$$s = \psi(x, y) \tag{7}$$

Selection of functions (6) and (7) is mostly limited by regularity condition implying that Jacobian matrix

$$\Upsilon(x, y) = \begin{bmatrix} \frac{\partial \psi}{\partial x} & \frac{\partial \psi}{\partial y} \\ \frac{\partial \varphi}{\partial x} & \frac{\partial \varphi}{\partial y} \end{bmatrix} \tag{8}$$

is not degenerate for any (x, y) belonging to curve S, i.e.
$det\Upsilon(x, y) \neq 0$

For regular geometrical objects there exists normalized description with orthogonal Jacobian matrix:

$$\Upsilon(x, y) = T(\alpha^*(s)) = \begin{bmatrix} \cos\alpha^*(s) & \sin\alpha^*(s) \\ -\sin\alpha^*(s) & \cos\alpha^*(s) \end{bmatrix} \in SO(2)$$

where $T(\alpha^*(s))$ is the rotational matrix of moving Frenet frame, $\alpha^*(s)$ is s-dependent target angle determining the current orientation of Frenet frame.
Frenet matrix satisfies to the differential equation

$$\dot{T^*}(\alpha^*) = \dot{s}\xi(s)ET^*(\alpha^*), \tag{9}$$

where $E = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$ and $\xi(s)$ is the path curvature.

From (9) also follows

$$\dot{\alpha^*} - \dot{s}\xi(s). \tag{10}$$

Angular orientation

Robot angular orientation with respect to curve S is defined as

$$\alpha = \alpha^*(s) + \Delta\alpha, \tag{11}$$

where $\Delta\alpha = const$ is the desired robot orientation with respect to the path.

In matrix notation, (11) takes the form

$$T(\alpha) - T(\Delta\alpha)T(\alpha^*). \tag{12}$$

Introducing errors and problem statement

Violation of condition (6) is characterised by orthogonal deviation

$$\epsilon = \varphi(x, y). \tag{13}$$

Violation of condition (10) is characterised by angular deviation

$$\delta - \alpha - \alpha^* + \Delta\alpha. \tag{14}$$

Therefore, the path following control problem consists in determination of inputs F_x, F_y and M in closed loop, which provides:

- stabilization of robot motion with respect to curve S;
- stabilization of robot angular orientation with respect to curve S;
- maintenance of the desired longitudinal motion by asymptotic zeroing of velocity error

$$\Delta V_s = V_s^* - \dot{s}. \tag{15}$$

Coordinate transformation

Perform the transformation of the system model (1)-(3) to the task-based form with outputs s, e and δ. To do so, differentiate (7), (13) and (14) with respect to time:

$$\begin{bmatrix} \dot{s} \\ \dot{e} \end{bmatrix} - \Upsilon(x, y) \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} - T(\alpha^*) \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix}, \tag{16}$$

$$\dot{\delta} = -\xi(s)\dot{s} + \omega. \tag{17}$$

Find the inverse transformation:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = T^T(\alpha^*) \begin{bmatrix} \dot{s} \\ \dot{e} \end{bmatrix},$$

$$\omega = \dot{\delta} \mid \xi(s)\dot{s}.$$

Control design

Once more differentiate (16) and (17) with account for (5), (9) and (12):

$$\begin{bmatrix} \ddot{s} \\ \ddot{e} \end{bmatrix} + \xi(s)\dot{s}E^T \begin{bmatrix} \dot{s} \\ \dot{e} \end{bmatrix} = \frac{1}{m} T^T(\Delta\alpha) \begin{bmatrix} F_x \\ F_y \end{bmatrix}, \tag{18}$$

$$\ddot{\delta} + \xi(s)\ddot{s} + \dot{\xi}(s)\dot{s} = \frac{1}{J} M. \tag{19}$$

Now consider virtual task-based controls:

$$\begin{bmatrix} u_s \\ u_e \end{bmatrix} = \frac{1}{m} T^T(\Delta\alpha) \begin{bmatrix} F_x \\ F_y \end{bmatrix} \tag{20}$$

$$u_\delta = \frac{1}{J} M - \xi(s) u_s \tag{21}$$

Control design

Substitute (20) and (21) to (18) and (19):

$$\begin{bmatrix} \ddot{s} \\ \ddot{e} \end{bmatrix} + \xi(s)\dot{s}E^T \begin{bmatrix} \dot{s} \\ \dot{e} \end{bmatrix} = \begin{bmatrix} u_s \\ u_e \end{bmatrix}, \tag{22}$$

$$\ddot{\delta} + \dot{\xi}(s)\dot{s} + \xi^2(s)\dot{s}\dot{e} = u_\delta. \tag{23}$$

Rewrite equations (22) and (23) with account for (15) for determining the velocity error dynamics:

$$\Delta\dot{V} + \xi(s)\dot{s}\dot{e} = -u_s,$$

$$\ddot{e} + \xi(s)\dot{s}^2 = u_e;$$

$$\ddot{\delta} + \dot{\xi}(s)\dot{s} + \xi^2(s)\dot{s}\dot{e} = u_\delta.$$

Control design

Now select the controllers:

$$u_s = -\xi(s)\dot{s}\dot{e} + k_s\Delta V, \tag{24}$$

$$u_e = \xi(s)\dot{s}^2 \quad k_{e1}\dot{e} \quad k_{e2}e, \tag{25}$$

$$u_\delta = \dot{\xi}(s)\dot{s} \mid \xi^2(s)\dot{s}\dot{e} \quad k_{\delta 1}\dot{\delta} \quad k_{\delta 2}\delta, \tag{26}$$

where k_s, k_{e1}, k_{e2}, $k_{\delta 1}$, $k_{\delta 2}$ are positive constants.

Finaly we determine actual control actions F_x, F_y and M and obtain

$$\begin{bmatrix} F_x \\ F_y \end{bmatrix} = mT(\Delta\alpha) \begin{bmatrix} -\xi(s)\dot{s}\dot{e} + k_s\Delta V \\ \xi(s)\dot{s}^2 - k_{e1}\dot{e} - k_{e2}e \end{bmatrix}, \tag{27}$$

$$M = J(\xi(s)u_s + \dot{\xi}(s)\dot{s} + \xi^2(s)\dot{s}\dot{e} - k_{\delta 1}\dot{\delta} - k_{\delta 2}\delta). \tag{28}$$

Example. Straight line segment

The normalized equation of the straight line

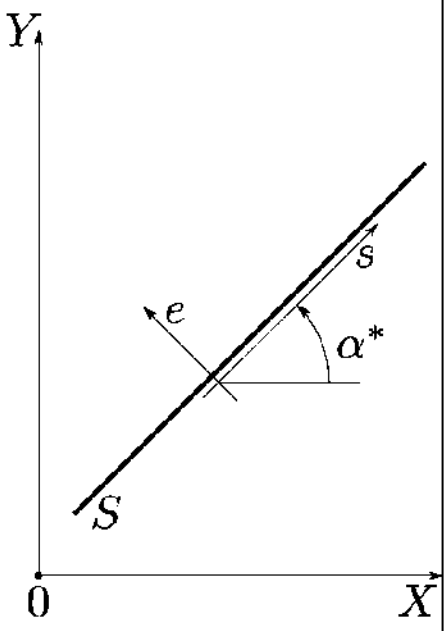
$$\varphi(q) = -\sin\alpha^* x + \cos\alpha^* y + \varphi_0 = 0,$$

$$\psi(q) = \cos\alpha^* x + \sin\alpha^* y + \psi_0,$$

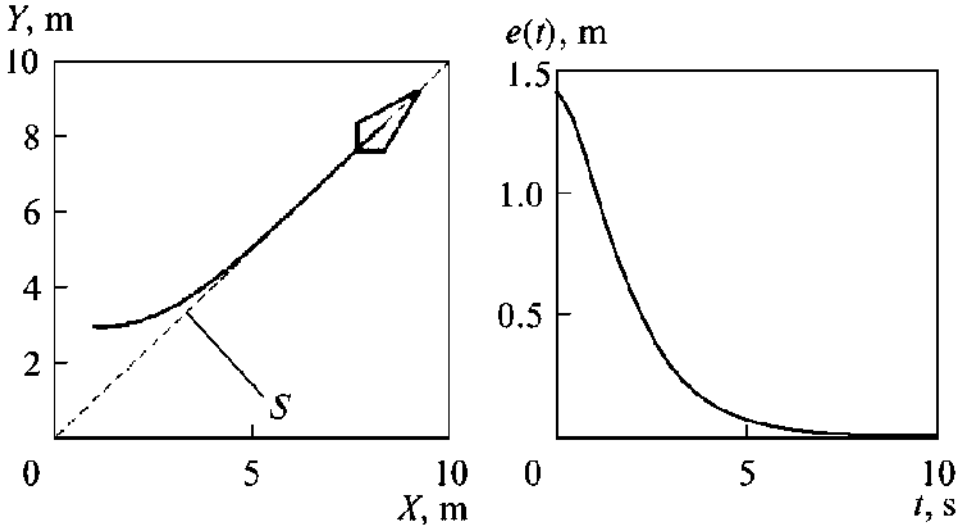
where α^* is the line inclination,
$\varphi_0 = const$, $\psi_0 = const$.
Orthogonal Jacobian matrix takes the form

$$\Upsilon(q) = \begin{bmatrix} \cos\alpha^* & \sin\alpha^* \\ -\sin\alpha^* & \cos\alpha^* \end{bmatrix} \in SO(2).$$

Obviously, the path curvature is zero.



Example. Simulation of the motion along the straight line.



Example. An arc of a circle

The normalized equation of the arc of the circle

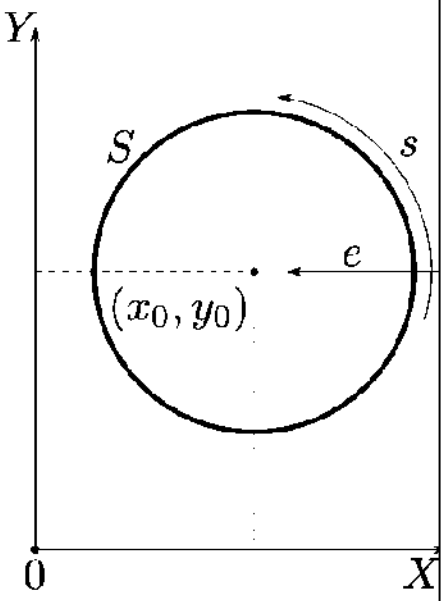
$$\varphi(q) = \frac{1}{2R}(R^2-(x-x_0)^2-(y-y_0)^2) = 0,$$

$$\psi(q) = R\arctan\frac{(y-y_0)}{(x-x_0)}.$$

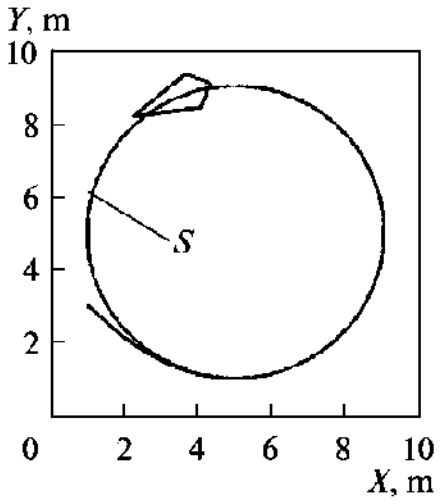
Orthogonal Jacobian matrix takes the form

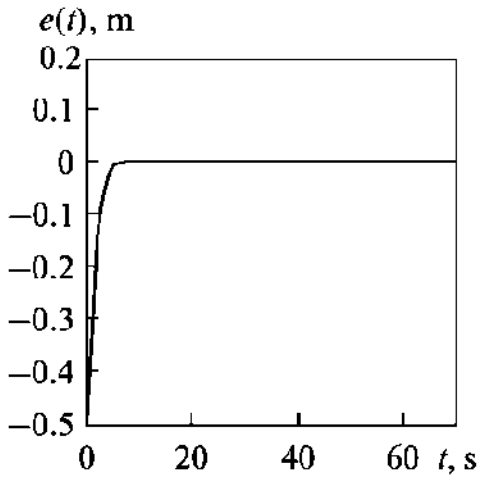
$$\Upsilon(q) = \frac{\perp}{R}\begin{bmatrix} -(y-y_0) & (x-x_0) \\ -(x-x_0) & -(y-y_0) \end{bmatrix} \in SO(2).$$

The path curvature is $\xi(s) = \frac{1}{R}$



Example. Simulation of the motion along the arc of a circle.





Dynamic model

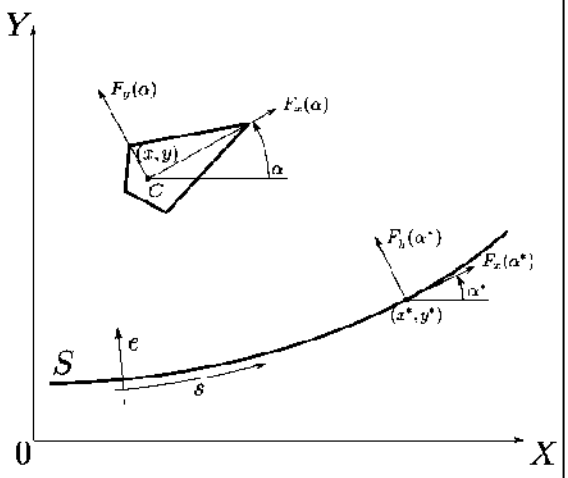
$$m\ddot{q} = F, \tag{29}$$

$$\dot{q} = R_O^I(\alpha)v, \tag{30}$$

$$R_O^I(\alpha) = \begin{bmatrix} cos\alpha & sin\alpha \\ -sin\alpha & cos\alpha \end{bmatrix}, \tag{31}$$

$$\dot{\alpha} = \omega, \tag{32}$$

where $q = \begin{bmatrix} x & y \end{bmatrix}^T \in R^2$ is vector of the Cartesian coordinates,
$F = \begin{bmatrix} F_x & F_y \end{bmatrix}^T \in R^2$ is vector of the control forces,
α is the orientation angle,
ω is the angular velocity.



Motion on the plane

The desired path is an implicitly described smooth segment of curve S:

$$\varphi(q) = 0, \tag{33}$$

and relevant local coordinate s (path length) is defined as

$$s = \psi(q) \tag{34}$$

Selection of functions (33) and (34) is mostly limited by regularity condition implying that Jacobian matrix

$$\Upsilon(q) = \begin{bmatrix} \frac{\partial\psi(q)}{\partial x} & \frac{\partial\psi(q)}{\partial y} \\ \frac{\partial\varphi(q)}{\partial x} & \frac{\partial\varphi(q)}{\partial y} \end{bmatrix} \tag{35}$$

is not degenerate for any (x, y) belonging to curve S, i.e.
$det\Upsilon(x, y) \neq 0$

Control design

Design of the velocity (inner) loop. Consider Lyapunov Function:

$$V_1 = \frac{1}{2}(\dot{q} - \bar{v})^T(\dot{q} - \bar{v}), \tag{36}$$

where $\bar{v}$ - a vector of desired velocities.

Find the derivation of the Lyapunov function V_1:

$$\dot{V}_1 = (\dot{q} - \bar{v})^T(\ddot{q} - \dot{\bar{v}}) = (\dot{q} - \bar{v})^T(\frac{F}{m} - \dot{\bar{v}}). \tag{37}$$

Control design

Define the control signal as:

$$\frac{1}{m}F = \dot{\bar{v}} - k_q(\dot{q} - \bar{v}), \tag{38}$$

where k_q is a positive constant. Then the derivation of the Lyapunov function V_1 is

$$\dot{V}_1 = -k_q(\dot{q} - \bar{v})^T(\dot{q} - \bar{v}) \leq 0, \tag{39}$$

which means asymptotic stability of the point $\dot{q} - v = 0$.

Now we can rewrite original system in reduced form:

$$\dot{q} = \bar{v}.$$

Let's construct the control $\bar{v}$ in th following form:

$$\bar{v} = u_e + u_s,$$

where u_e is the term, which provides stabilization with respect to the desired path and u_s provides desired velocity along the path.

Reduced system

Perform the transformation of the system model (29)-(32) to the task-based form with outputs s and e_1, using Jacobian matrix (35):

$$\begin{bmatrix} \dot{s} \\ \dot{e}_1 \end{bmatrix} = \Upsilon(q)\dot{q} = \Upsilon(q)R_I^O(\alpha)v. \tag{40}$$

We can choose the control signal u_s in the form

$$u_s = R_O^I \Upsilon^{-1}(r) \begin{bmatrix} V^* \\ 0 \end{bmatrix} \tag{41}$$

Now design stabilization control u_e. Consider Lyapunov Function:

$$V_2 = \frac{k_e}{2}\varphi^2(q), \tag{42}$$

Reduced system

Find the derivation of the Lyapunov function V_2.

$$\dot{V}_2 = k_e\varphi(q)\nabla\varphi(q) = (k_e\varphi(q)\nabla\varphi(q))^\top u_e +$$
$$+(k_e\varphi(q)\nabla\varphi(q))^\top\Upsilon^{-1}q)\begin{bmatrix}V^*\\0\end{bmatrix} = (k_e\varphi(q)\nabla\varphi(q))^\top u_e.$$

As you can see, the second half of the expression is identically zero due to orthogonality. Now select u_e as

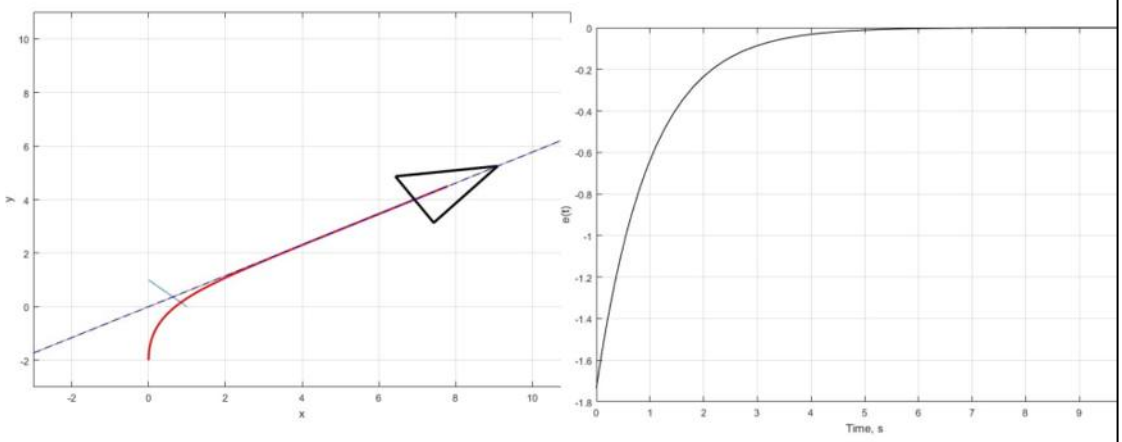
$$u_e = -k_e\varphi(q)\frac{\partial}{\partial q}\varphi(q), \tag{43}$$

where k_e is positive constant.
Then the derivation of the Lyapunov function V_2 is

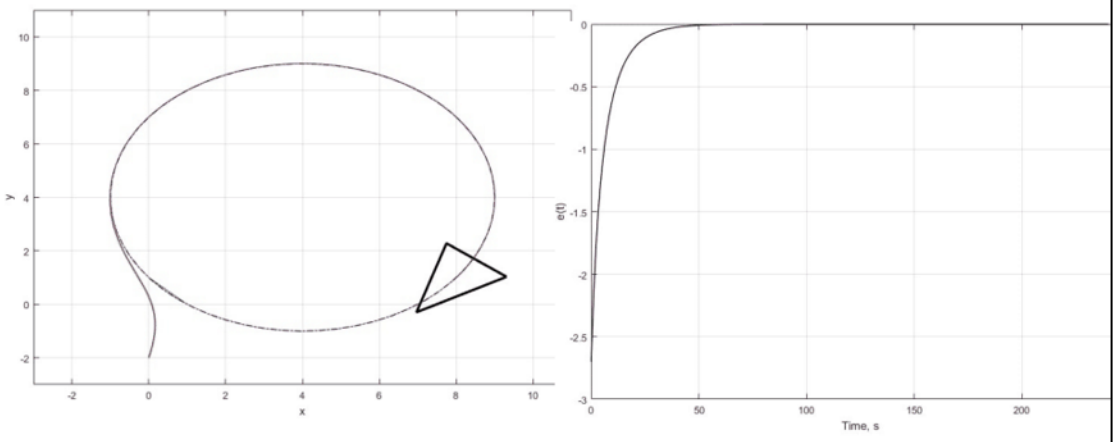
$$\dot{V}_2 = -u_e^2 \leq 0$$

It proves the asymptotic stability of the initial system at the point $e(q) = 0$.

Example. Simulation of the motion along the straight line.



Example. Simulation of the motion along the arc of a circle.



2D moving frame

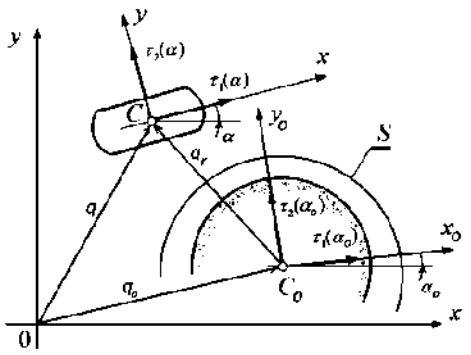
Dynamic model of the plant:

$$\begin{bmatrix} \dot{q} \\ \dot{\alpha} \end{bmatrix} = R^T(\alpha) \begin{bmatrix} v \\ \omega \end{bmatrix}, \qquad (44)$$

$$A \begin{bmatrix} \dot{v} \\ \dot{\omega} \end{bmatrix} = R^T(\alpha) \begin{bmatrix} F \\ M \end{bmatrix}, \qquad (45)$$

$$R(\alpha) = \begin{bmatrix} T(\alpha) & 0 \\ 0 & 1 \end{bmatrix},$$

$$A = \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & J \end{bmatrix},$$



External moving object

Dynamic model of the external moving object:

$$\dot{q}_o = v_o, \tag{46}$$

$$\dot{T}(\alpha_o) = \omega_o E T(\alpha_o), \tag{47}$$

Desired trajectory in relative coordinates

$$\varphi(q_r) = 0, \tag{48}$$

Local coordinate

$$s = \psi(q_r), \tag{49}$$

Relative coordinates

Position, velocity and acceleration of the plant in moving frame:

$$q_r = T(\alpha_o)(q - q_o), \tag{50}$$

$$\alpha_r = \alpha - \alpha_o. \tag{51}$$

$$\dot{q}_r = \omega_o E q_r + T(\alpha_o)\left(\dot{q} - \dot{q}_o\right), \tag{52}$$

$$\dot{\alpha}_r = \omega - \omega_o, \tag{53}$$

$$\begin{aligned} \ddot{q}_r &= (\omega_o)^2 q_r \mid 2\omega_o E T'(\alpha_o)\left(\dot{q} \quad \dot{q}_o\right) \mid \\ &+ \frac{1}{m} T'(\alpha_o) T^T(\alpha) F_z, \end{aligned} \tag{54}$$

$$\ddot{\alpha}_r = \frac{1}{J} M. \tag{55}$$

Task-oriented coordinates

Consider orthogonal deviation

$$e(q_r) = \varphi(q_r), \tag{56}$$

and local coordinate s

$$s = \psi(q_r) \tag{57}$$

Choosing of functions (56) and (57) based on regularity condition which implies that Jacoby matrix

$$\Upsilon(q_r) = \begin{bmatrix} \partial\psi/\partial q_r \\ \partial\varphi/\partial q_r \end{bmatrix} \tag{58}$$

is nondegenerate for all q_r, belongs to curve S, i.e. $det\Upsilon(q_r) \neq 0$.

Trajectory control synthesis

Imply the transformation of model (44)-(45) to the task-oriented coordinates:

$$\begin{bmatrix} \dot{s} \\ \dot{e} \end{bmatrix} = T(\alpha_r^*)\left(T^T(\alpha_r)v_z + \omega_o E q_r - T(\alpha_o)v_o\right), \tag{59}$$

$$\dot{\delta} = -\dot{s}\xi(s) + \omega - \omega_o. \tag{60}$$

Choose local controllers as

$$u_s = k_s\Delta V - \dot{s}\xi(s)\dot{e} - 2\omega_o\dot{e}, \tag{61}$$

$$u_e = k_{e1}e + k_{e2}\dot{e} + \dot{s}^2\xi(s)\dot{e} + 2\omega_o\dot{s}, \tag{62}$$

$$u_\delta = k_{\delta 1}\delta + k_{\delta 2}\dot{\delta} + \frac{\partial\xi}{\partial s}\dot{s} + \ddot{s}\xi(s). \tag{63}$$

Final control laws

$$F = mT(\alpha_r)T^T(\alpha_r^*)\left(\begin{bmatrix} u_s \\ u_e \end{bmatrix} - (\omega_o)^2 T(\alpha^*)q_r\right), \tag{64}$$

$$M = Ju_\delta. \tag{65}$$

Collision avoidance strategies

Some collision avoidance strategies

- Bypass
- Detour

Equidistant border around the obstacle

$$\varphi^*(q) = x^2 + y^2 - R^2 = 0, \tag{66}$$

Trajectory control in presence of moving external object.

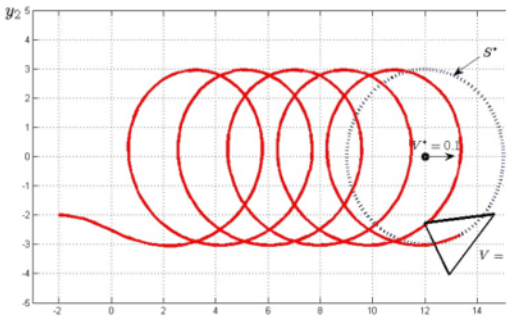


Figure 1: The results of modeling the motion relative to a moving external object.

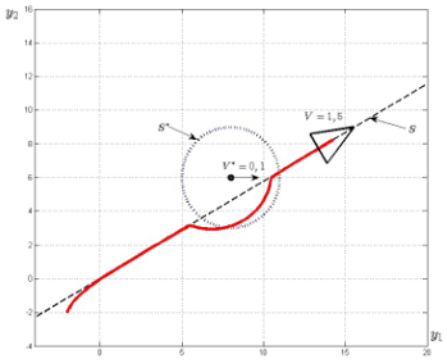


Figure 2: The results of modeling the detour of a moving obstacle.

Spatial motion

Dynamic model:

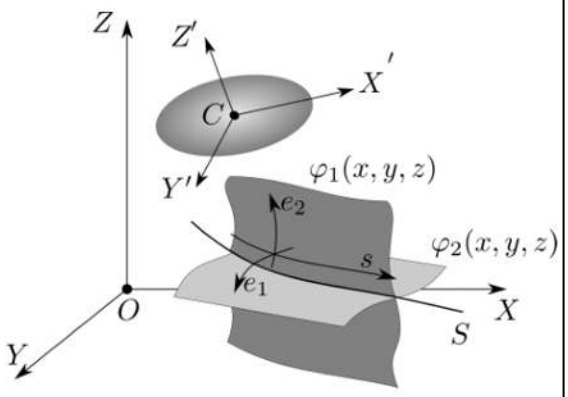
$$\dot{q} = v, \tag{67}$$

$$\dot{v} = \frac{1}{m} F_c, \tag{68}$$

$$\dot{R}(\alpha) = S(\omega) R(\alpha), \tag{69}$$

$$J\dot{\omega} + \omega \times J\omega = M_c, \tag{70}$$

$$S(\omega) = \begin{bmatrix} 0 & \omega_3 & -\omega_2 \\ -\omega_3 & 0 & \omega_1 \\ \omega_2 & -\omega_1 & 0 \end{bmatrix}.$$



Rotational matrix

The rotation matrix $R(\alpha)$ can be represented through Euler angles as

$$R(\alpha) = R_3(\psi) R_2(\theta) R_1(\phi), \tag{71}$$

where

$$R_1(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}$$

$$R_2(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

$$R_3(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Desired path

The desired path S describes as an intersection of two implicit surfaces:

$$\varphi_1(x, y, z) = 0 \cap \varphi_2(x, y, z) = 0. \tag{72}$$

Tangential velocity along the curve S is defined as

$$\dot{s} = \frac{\nabla\varphi_1 \times \nabla\varphi_2}{\|\nabla\varphi_1 \times \nabla\varphi_2\|} v, \tag{73}$$

where $\times$ is the vector product and $\|\cdot\|$ is the vector norm.
Jacobian matrix:

$$\Upsilon(x, y, z) = \begin{bmatrix} \frac{\nabla\varphi_1 \times \nabla\varphi_2}{\|\nabla\varphi_1 \times \nabla\varphi_2\|} \\ \frac{\nabla\varphi_1}{\|\nabla\varphi_1\|} \\ \frac{\nabla\varphi_2}{\|\nabla\varphi_2\|} \end{bmatrix} \tag{74}$$

Introducing errors and problem statement

Violation of condition (72) is characterised by orthogonal deviations

$$e_1 = \varphi_1(x, y, z). \tag{75}$$

$$e_2 = \varphi_2(x, y, z). \tag{76}$$

Therefore, the path following control problem consists in determination of inputs $F_c = \begin{bmatrix} F_x & F_y & F_z \end{bmatrix}$ and M_c in closed loop, which provides:

- stabilization of robot motion with respect to curve S;
- maintenance of the desired longitudinal motion by asymptotic zeroing of velocity error

$$\Delta V_s = V_s^* - \dot{s}; \tag{77}$$

- stabilization of robot angular orientation with respect to curve S.

Translation motion control

Perform the transformation of the system model (67)-(70) to the task-based form with outputs s, e_1 and e_2. To do so, differentiate (73), (75) and (76) with respect to time:

$$\begin{bmatrix} \dot{s} \\ \dot{e_1} \\ \dot{e_2} \end{bmatrix} = \Upsilon(x, y, z)v. \tag{78}$$

Once more differentiate (78) with account for (68):

$$\begin{bmatrix} \ddot{s} \\ \ddot{e_1} \\ \ddot{e_2} \end{bmatrix} - \dot{\Upsilon}(x, y, z)v + \Upsilon(x, y, z)\frac{F_c}{m}. \tag{79}$$

Translation motion control

Consider the virtual (task-based) controls:

$$\dot{\Upsilon}(x, y, z)v + \Upsilon(x, y, z)\frac{F_c}{m} = \begin{bmatrix} u_s \\ u_{e1} \\ u_{e2} \end{bmatrix} \tag{80}$$

Substitute (80) to (79) and obtain

$$\begin{bmatrix} \ddot{s} \\ \ddot{e_1} \\ \ddot{e_2} \end{bmatrix} = \begin{bmatrix} u_s \\ u_{e1} \\ u_{e2} \end{bmatrix}. \tag{81}$$

Translation motion control

Now select the controllers:

$$u_s = K_s \Delta \dot{s}, \tag{82}$$

$$u_{e1} = -K_{1e1}\dot{e_1} - K_{2e1}e_1, \tag{83}$$

$$u_{e2} = -K_{1e2}\dot{e_2} - K_{2e2}e_2, \tag{84}$$

where K_s, K_{1e1}, K_{2e1}, K_{1e2}, K_{2e2} are positive constants.

Finaly we determine actual control action F_c and obtain

$$F_c = m\Upsilon(x,y,z)^{-1}(\begin{bmatrix} u_s \\ u_{e1} \\ u_{e2} \end{bmatrix} - \dot{\Upsilon}(x,y,z)v). \tag{85}$$

Rotation motion control

Introduce vector of angular errors $\delta - \begin{bmatrix} \delta_\phi & \delta_\theta & \delta_\psi \end{bmatrix}^T \in R^3$ and the angular deviation matrix

$$R(\delta) = R(\alpha)R^T(\alpha^*)R^T(\Delta), \tag{86}$$

where $R(\alpha^*) \in SO(3)$ is the matrix of angular orientation of the body-fixed frame along the curve S, $R(\Delta) \in SO(3)$ is the matrix of the desired angular orientation. Define the angular error function as

$$e_r = \frac{1}{2}(R(\delta) - R(\delta)^T)^\vee, \tag{87}$$

where $\vee$ it the transformation $SO(3) \to R^3$.

Rotation motion control

Define the angular speed error e_ω. Differentiate (86) with account for (69) and obtain the equation

$$\frac{d}{dt}R(\delta) = S(\dot{\delta})R(\delta) = e_\omega R(\delta), \tag{88}$$

$$\frac{d}{dt}R(\delta) = S(\omega)R(\delta) - R(\alpha)R^T(\alpha^*)S(\omega^*)R^T(\Delta), \tag{89}$$

Use the property of skew symmetric matrix $RS(\omega)R^T = S(R\omega)$ and obtain final expression

$$\frac{d}{dt}R(\delta) = (S(\omega) - S(R(\alpha)R^T(\alpha^*)\omega^*))R(\delta), \tag{90}$$

and

$$e_\omega = \omega - R(\alpha)R^T(\alpha^*)\omega^*. \tag{91}$$

Rotation motion control

Differentiating (91) with account for (69)

$$\dot{e_\omega} = \frac{1}{J}(M - \omega \times J\omega) + a_d, \tag{92}$$

where $a_d = -S(\omega)R(\alpha)R^T(\alpha^*)\omega^* + R(\alpha)R^T(\alpha^*)\dot{\omega}^*$. Resulting attitude controller has form

$$M_c = \omega \times J\omega - Ja_d - K_R e_r - K_\omega e_\omega. \tag{93}$$

where K_R, K_ω, are positive constants.

Numerical example

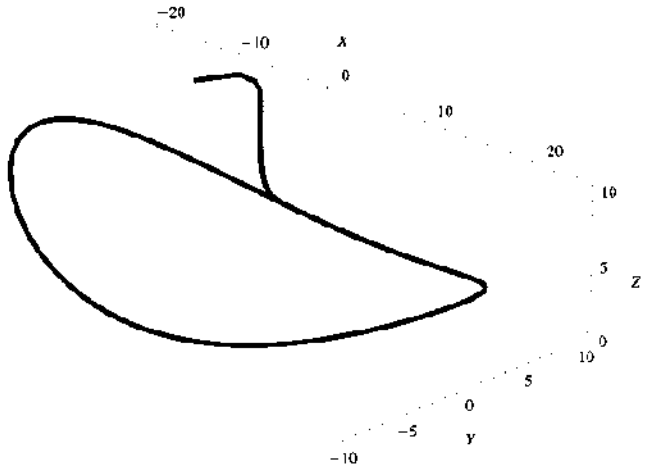
Consider the plant as a rigid body described by model (67)-(70) with $m = 1$, $J = 1$.

Initial position of the plant is $x_0 = \begin{bmatrix} -10 & 5 & 10 \end{bmatrix}^T$ and initial orientation is $\alpha_0 = \begin{bmatrix} 3 & 2 & 1 \end{bmatrix}^T$.

Parameters of the controller are $K_{1e1} = 1$, $K_{2e1} = 10$, $K_{1e2} = 1$, $K_{2e2} = 10$, $K_R = 20$, $K_\omega = 50$.

Desired speed along the path $\dot{s} = 1$.

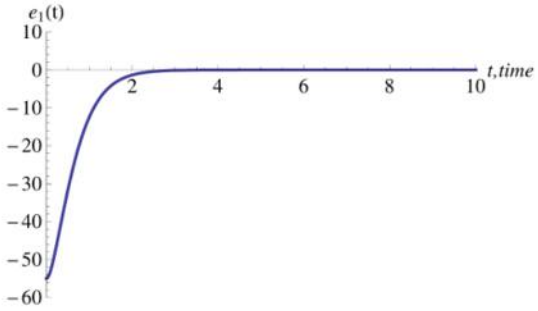
Numerical example

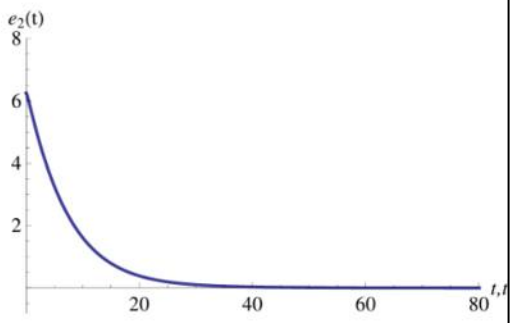


Motion along desired path:

$$\varphi_1(x, y, z) = 0.2x^2 + y^2 - R^2 = 0 \ \cap \ \varphi_2(x, y, z) = z + 0.05y^2 - 5 = 0$$

Numerical example

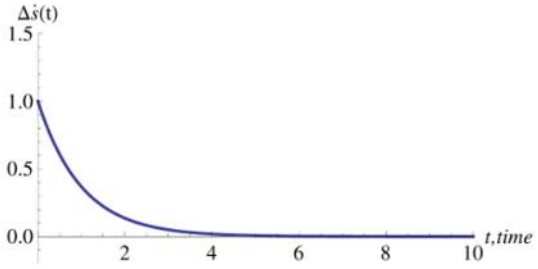


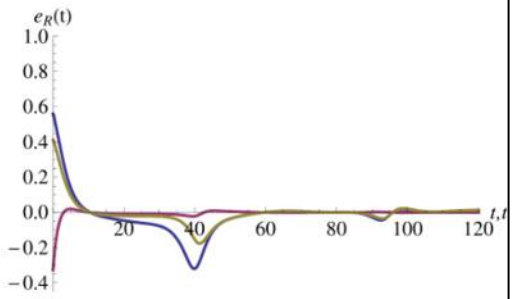


Position error $e_1 = \varphi_1(x, y, z)$

Position error $e_2 = \varphi_2(x, y, z)$

Numerical example





Speed error $\Delta V = \dot{s}^* - \dot{s}$

Angular error e_r

Moving frame description

Model of the plant motion:

$$\ddot{x}(t) = g - \frac{f(t)}{m}\vec{n}(t), \qquad (94)$$

$$\dot{R}(t) = R(t)S(\omega(t)), \qquad (95)$$

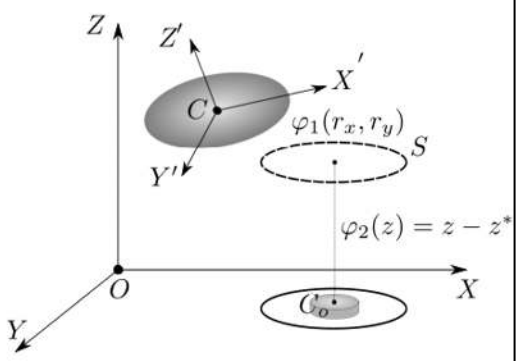
$$M_c(t) = J\dot{\omega}(t) + \omega(t) \times J\omega(t). \qquad (96)$$

Description of the moving frame:

$$\dot{x}_i = R_T(\alpha^*)v_i, \qquad (97)$$

$$\dot{\alpha}^* = \omega_i, \qquad (98)$$

$$\dot{R}_T(\alpha^*) = R_T S(\omega_i). \qquad (99)$$



Moving frame description

Relative position:

$$r = R_T^\top(\alpha^*)(x - x_i), \qquad (100)$$

Relative velocity:

$$\dot{r} = R_T^\top(\alpha^*)\dot{x} - S(\omega_i)r, \qquad (101)$$

Relative acceleration:

$$\ddot{r} = R_T^\top(\alpha^*)\ddot{x} - 2S(\omega_i)\dot{r} - S^2(\omega_i)r. \qquad (102)$$

Control design

Design of the velocity (inner) loop. Consider Lyapunov Function:

$$V_1 = \frac{1}{2}(\dot{r}_x - \bar{v})^T(\dot{r}_x - \bar{v}) + k_d \ln(2 - (R^\top \bar{n})^\top (R^\top R_T \bar{n}_d)), \qquad (103)$$

where $\bar{v}$ - a vector of desired velocities, $\bar{n}_d$ - a vector of desired orientation and k_d - a positive constant.

Find the derivation of the Lyapunov function V_1:

$$\dot{V}_1 = (\dot{r} - \bar{u})^\top (R_T^\top - \frac{f}{m} R_T^\top \bar{n} - 2S(\omega_i)\dot{r} - S^2(\omega_i)r - \dot{\bar{u}}) + \gamma^\top \left(\omega - \omega_i - \frac{S(R^\top R_T \bar{n}_d)}{|\bar{n}_d|} R^\top R_T \dot{\bar{n}}_d\right), \qquad (104)$$

where $\gamma^\top = \frac{k_d (R^\top \bar{n})^\top S^\top (R^\top R_T \bar{n}_d)}{(2 - (R^\top \bar{n})^\top (R^\top R_T \bar{n}_d))}$ and $|a|$ is the euclidean norm of vector a.

Control design

Define the substitution of variables if following form:

$$\delta = R_T^\top g - 2S(\omega_i)\dot{r} - S^2(\omega_i)r - \dot{\bar{u}}, \delta = \frac{f_d}{m}\bar{n}_d,$$

where $f_d = |\delta|$ and $\bar{n}_d = \frac{\delta}{|\delta|}$.

Select control signals f and $\omega = \omega_i$ in the form

$$f = f_d \cdot ((R_T^\top \bar{n})^\top \bar{n}_d) \quad k_v (\dot{r} \quad \bar{u})^\top R_T^- \bar{n}, \qquad (105)$$

$$\omega_d = \omega_i + \frac{S(R^\top R_T \bar{n}_d)}{|\bar{n}_d|} R^- R_T \dot{\bar{n}}_d) + \sigma - K_\gamma \gamma, \qquad (106)$$

where k_v, k_γ are positive constants and σ is

$$\sigma = \left(\frac{f_d(2 - (R^\top n)^\top (R^\top R_T n_d))}{m k_d} (\dot{r} - \bar{u})^\top S(R_T^\top \bar{n}) R_T^\top R\right). \qquad (107)$$

Then the derivation of the Lyapunov function V_1 is

$$\dot{V}_1 = -k_v((\dot{r} - \bar{u})^\top R_T^\top \bar{n})^2 - k_\gamma \gamma^\top \gamma \leq 0, \tag{108}$$

which means asymptotic stability of the point $\dot{r} - \bar{u} = 0$, $\bar{n} - \bar{n}_d$.
Now we can rewrite original system in reduced form:

$$\dot{r} = \bar{u}.$$

Let's construct the control $\bar{u}$ in th following form:

$$\bar{u} = u_e + u_s,$$

where u_e is the term, which provides stabilization with respect to the desired path and u_s provides desired velocity along the path.

Reduced system

Perform the transformation of the system model (100)-(102) to the task-based form with outputs s, e_1 and e_2, using Jacobian matrix (74):

$$\begin{bmatrix} \dot{s} \\ \dot{e}_1 \\ \dot{e}_2 \end{bmatrix} = \Upsilon(r)\dot{r}$$

We can choose the control signal u_s in the form

$$u_s = \Upsilon^{-1}(r) \begin{bmatrix} V_s^* \\ 0 \\ 0 \end{bmatrix} \tag{109}$$

Now design stabilization control u_e. Consider Lyapunov Function:

$$V_2 = \frac{k_1}{2}\varphi_1^2(r) + \frac{k_2}{2}\varphi_2^2(r), \tag{110}$$

Reduced system

Find the derivation of the Lyapunov function V_2.

$$\dot{V}_2 = (k_1\varphi_1(r)\nabla\varphi_1(r) + k_2\varphi_2(r)\nabla\varphi_2(r))^\top \dot{r} =$$
$$(k_1\varphi_1(r)\nabla\varphi_1(r) + k_2\varphi_2(r)\nabla\varphi_2(r))^\top u_s +$$
$$+(k_1\varphi_1(r)\nabla\varphi_1(r) + k_2\varphi_2(r)\nabla\varphi_2(r))^\top \Upsilon^{-1} r) \begin{bmatrix} V^* \\ 0 \\ 0 \end{bmatrix} =$$
$$(k_1\varphi_1(r)\nabla\varphi_1(r) + k_2\varphi_2(r)\nabla\varphi_2(r))^\top u_s.$$

As you can see, the second half of the expression is identically zero due to orthogonality. Now select u_e as

$$u_e = -(k_1\varphi_1(r)\nabla\varphi_1(r) + k_2\varphi_2(r)\nabla\varphi_2(r)), \qquad (111)$$

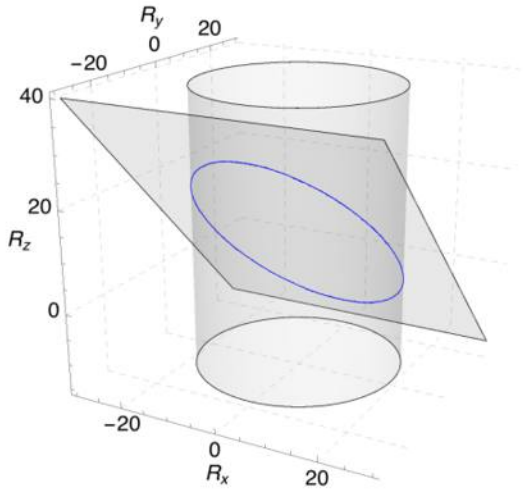
where k_1 and k_2 are positive constants.

Resulting control

Resulting control:

$$\begin{aligned}
M_c &= \omega \times J\omega + J\dot{\omega}_d + k_\omega J(\omega - \omega_d), \\
\omega_d &= \omega_T + \frac{S\left(R^\top R_T \bar{n}_d\right)}{|\bar{n}_d|} R^\top R_T \dot{\bar{n}}_d + \sigma - k_\gamma \gamma, \\
\sigma^\top &= \frac{f_d \cdot \left(2 - (R^\top \bar{n})^\top (R^\top R_T \bar{n}_d)\right)}{m k_d} (\dot{r} - u)^\top S(R_T^\top n) R_T^\top R \\
\gamma^\top &= \frac{k_d (R^\top \bar{n})^\top S^\top \left(R^\top R_T \bar{n}_d\right)}{\left(2 - (R^\top \bar{n})^\top (R^\top R_T \bar{n}_d)\right)}, \\
\delta &= R_T^\top g - 2S(\omega_T)\dot{r} - S^2(\omega_T) r - \dot{\bar{u}}, \\
f_d &= |\delta|, \quad \bar{n}_d = \frac{\delta}{|\delta|}, \\
f &= f_d \cdot ((R_T^\top \bar{n})^\top \bar{n}_d) - k_v (\dot{r} - \bar{u})^\top R_T^\top \bar{n}.
\end{aligned}$$

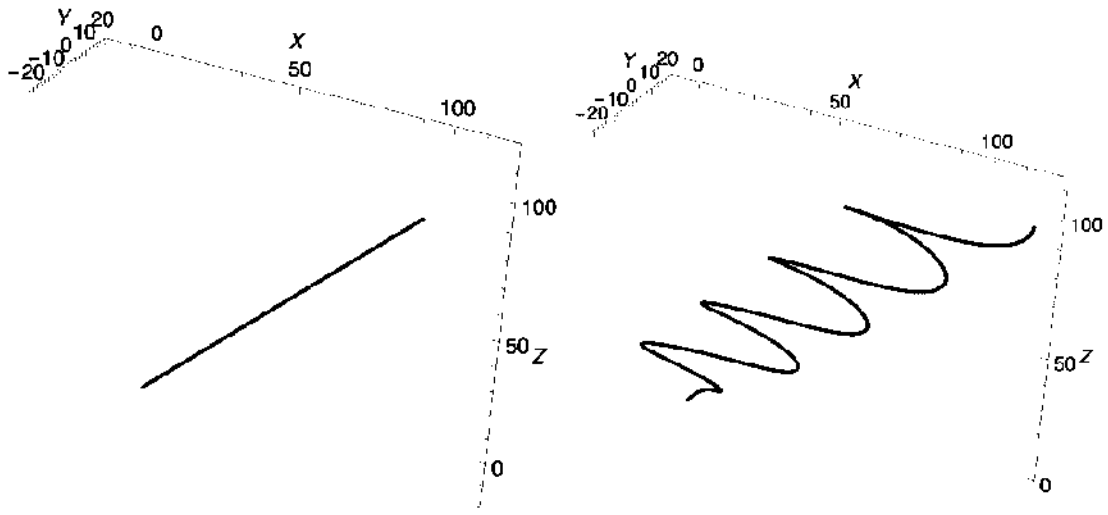
Example



$$\varphi_1(r) = r_x^2 + r_y^2 - 400 = 0 \cap \varphi_2(r) = r_z + r_y - 10 = 0$$

The desired speed along the given path $\dot{s}^* = 30$.

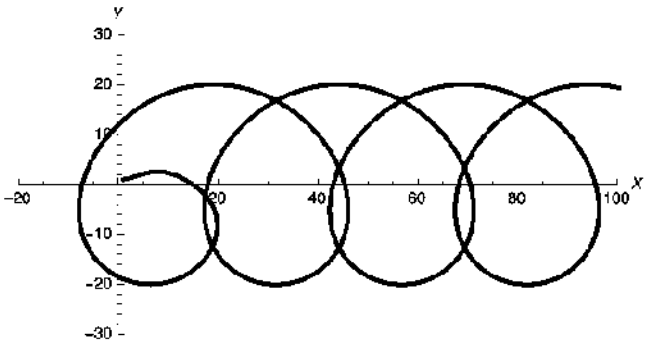
Example



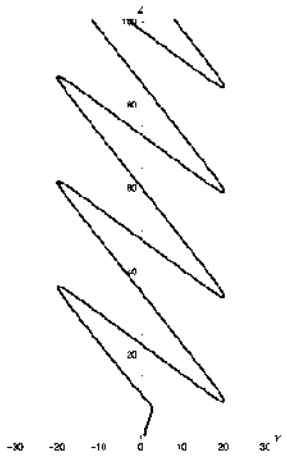
Moving frame spatial motion

Plant spatial motion

Example

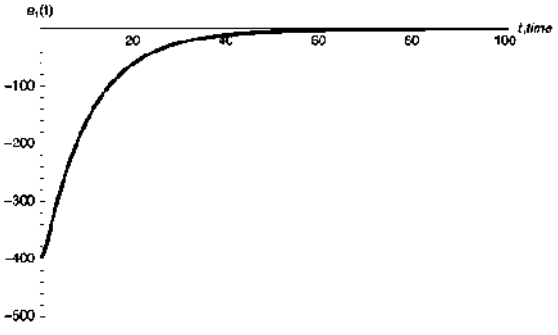


Projection of the plant motion on XY plane

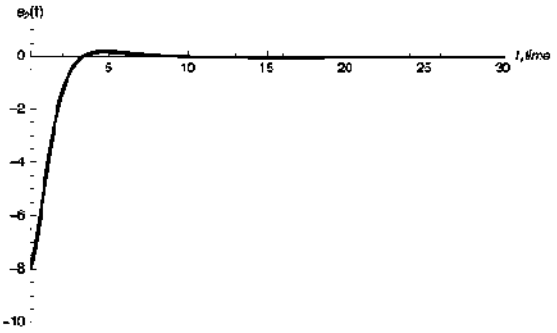


Projection of the plant motion on YZ plane

Example

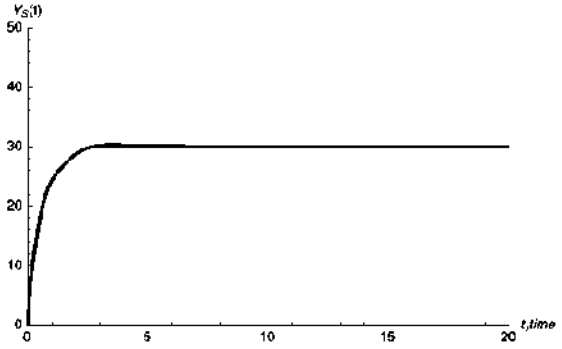


$e_1 = \varphi_1(r)$

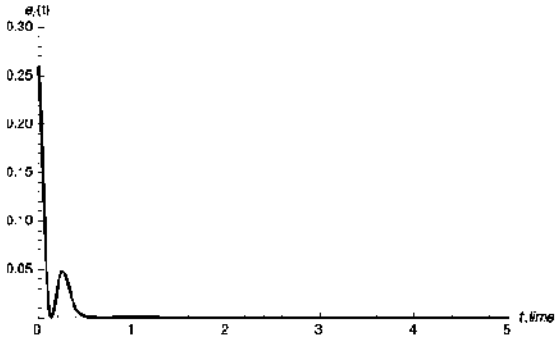


$e_2 = \varphi_2(r)$

Example

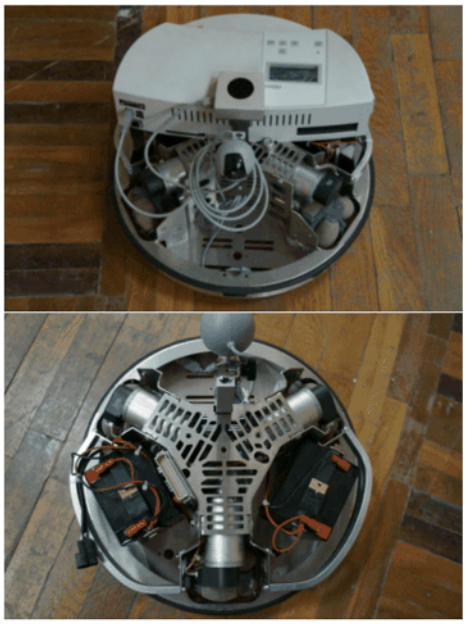


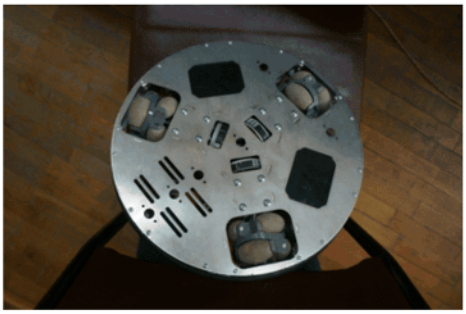
The velocity along the path
$V^*(t) = 30$



Angular error
$e_r = 1 - (R^\mathsf{T}\bar{n})^\mathsf{T}(R^\mathsf{T}R_T\bar{n}_d)$

Omnidirectional mobile robot "Robotino" by Festo Didactics

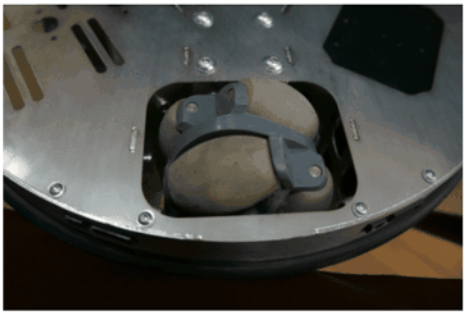




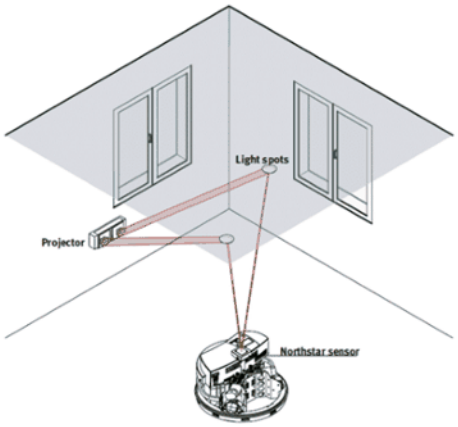
Geometric dimensions:

- Diameter: 370 mm
- Height: 210 mm
- Weight: 11 kg

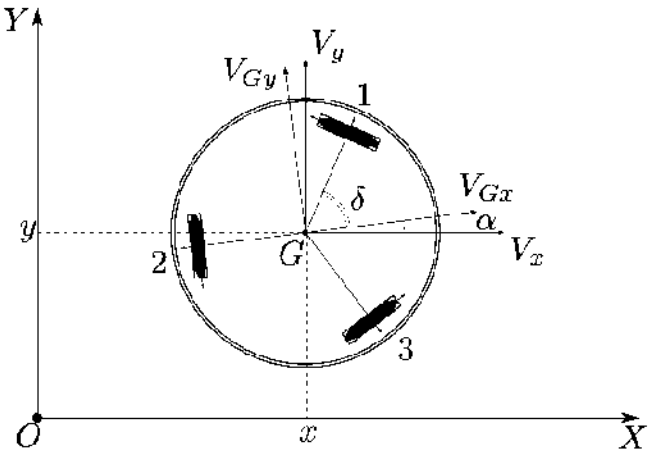
Omni wheels “Robotino”



Local Navigation “Northstar”

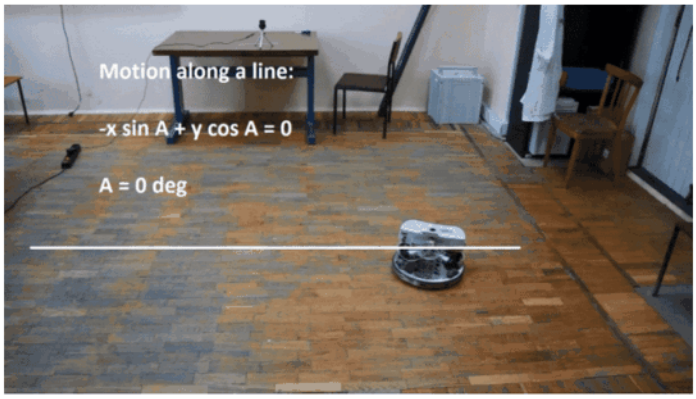


Mathematical model

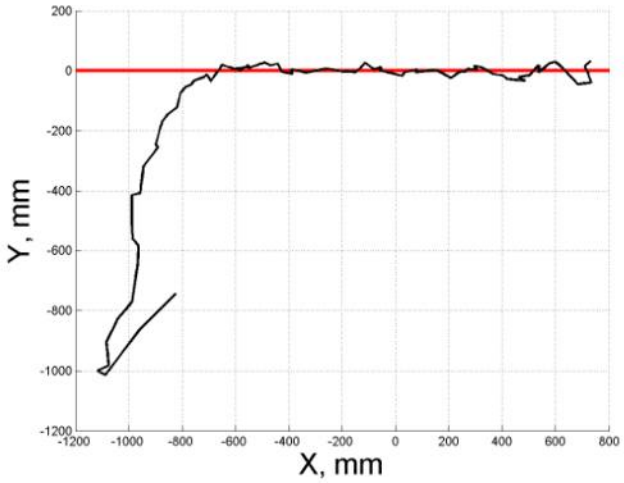


$$\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix} = \begin{bmatrix} -\sin\frac{\pi}{3} & \cos\frac{\pi}{3} & L \\ 0 & -1 & L \\ \sin\frac{\pi}{3} & \cos\frac{\pi}{3} & L \end{bmatrix} \begin{bmatrix} V_{Gx} \\ V_{Gy} \\ \dot{\alpha} \end{bmatrix}$$

Motion along a straight line

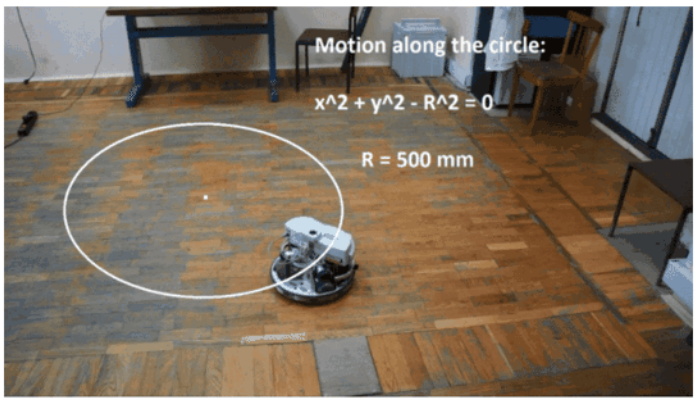


Motion along a straight line

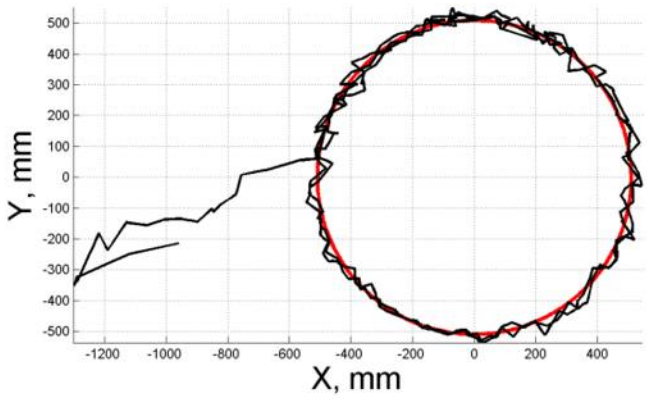


$$\varphi(x, y) = -\sin \alpha x + \cos \alpha y = 0,$$

Motion along the circle

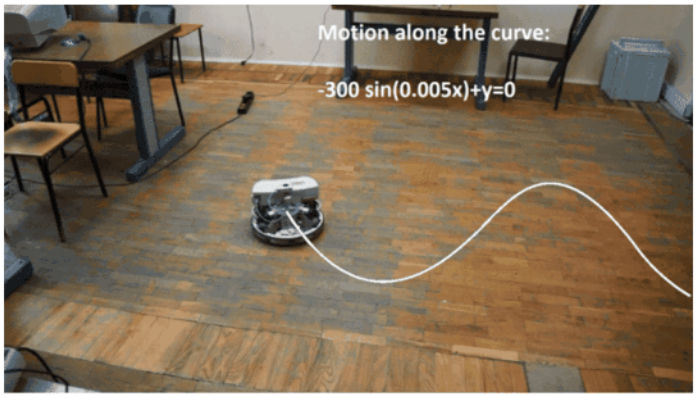


Motion along the circle

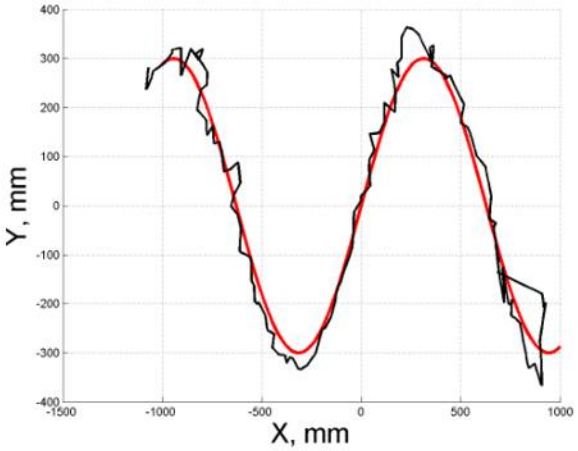


$$\varphi(x, y) = x^2 + y^2 - 2500 = 0$$

Motion along the sinusoid

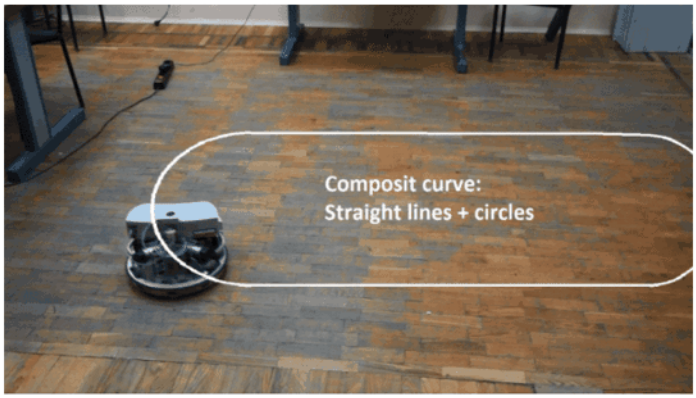


Motion along the sinusoid

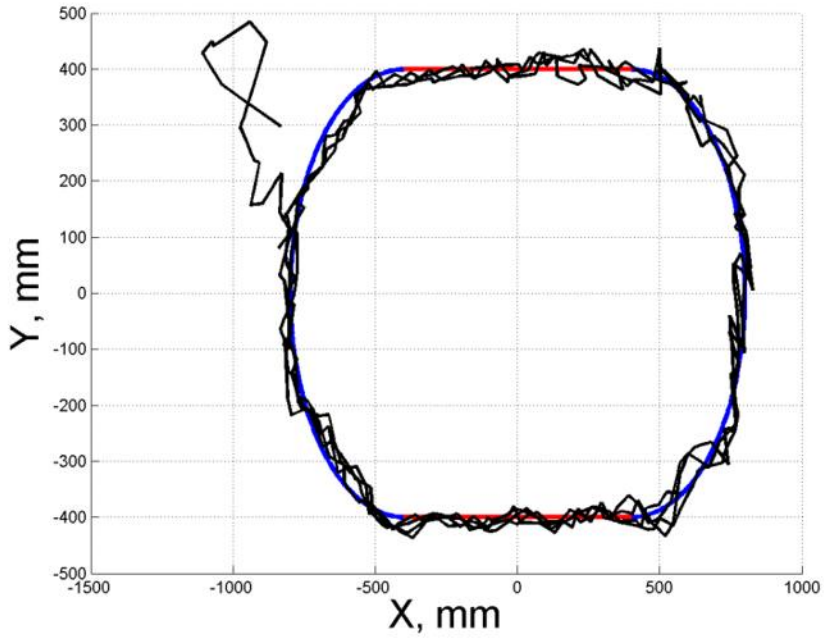


$$\varphi(x, y) = -300 \sin 0.005x + y = 0$$

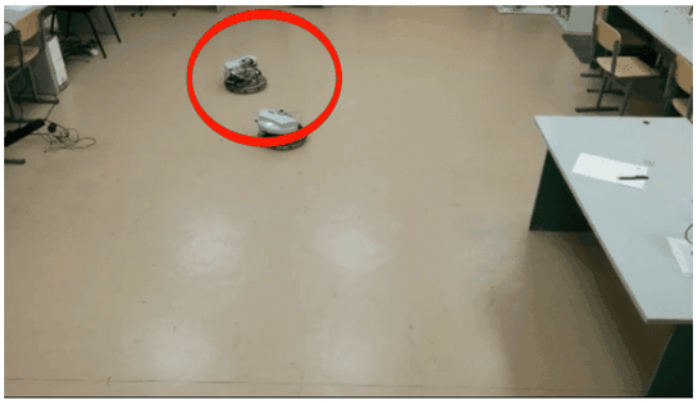
Motion along a complex curve



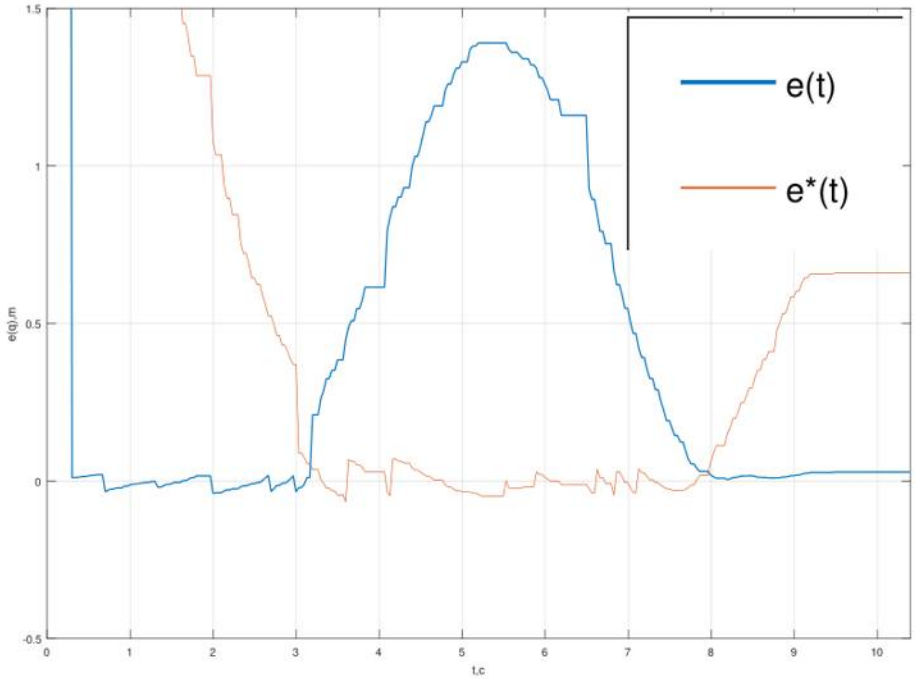
Motion along a complex curve



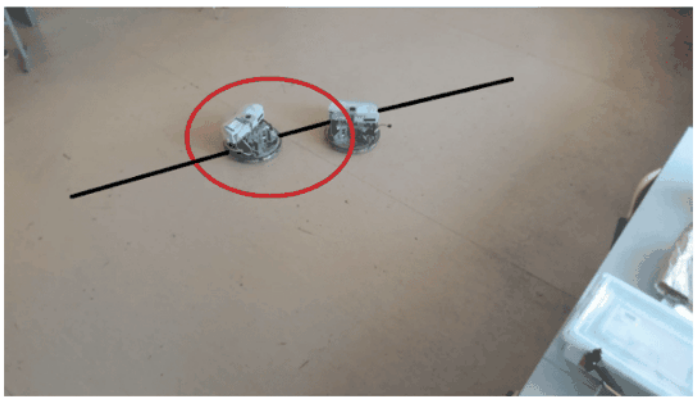
Avoiding moving obstacle



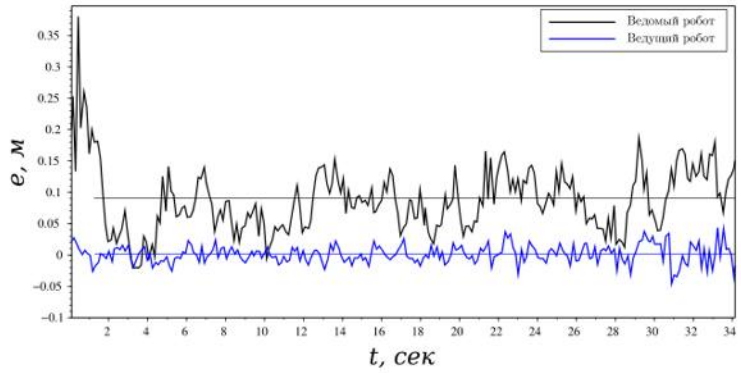
Avoiding moving obstacle



Coordinated motion of two robots



Coordinated motion of two robots



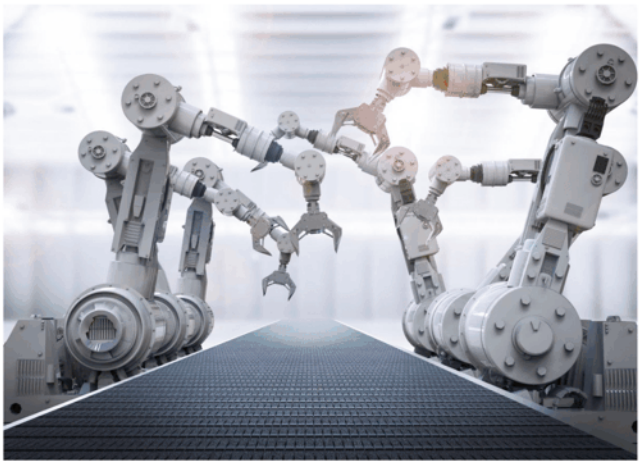
Digital control systems
Digital and microcontroller devices

Digital and Microcontroller Devices

Vlasov Sergei

Robots, what is it?

Robots, what is it?

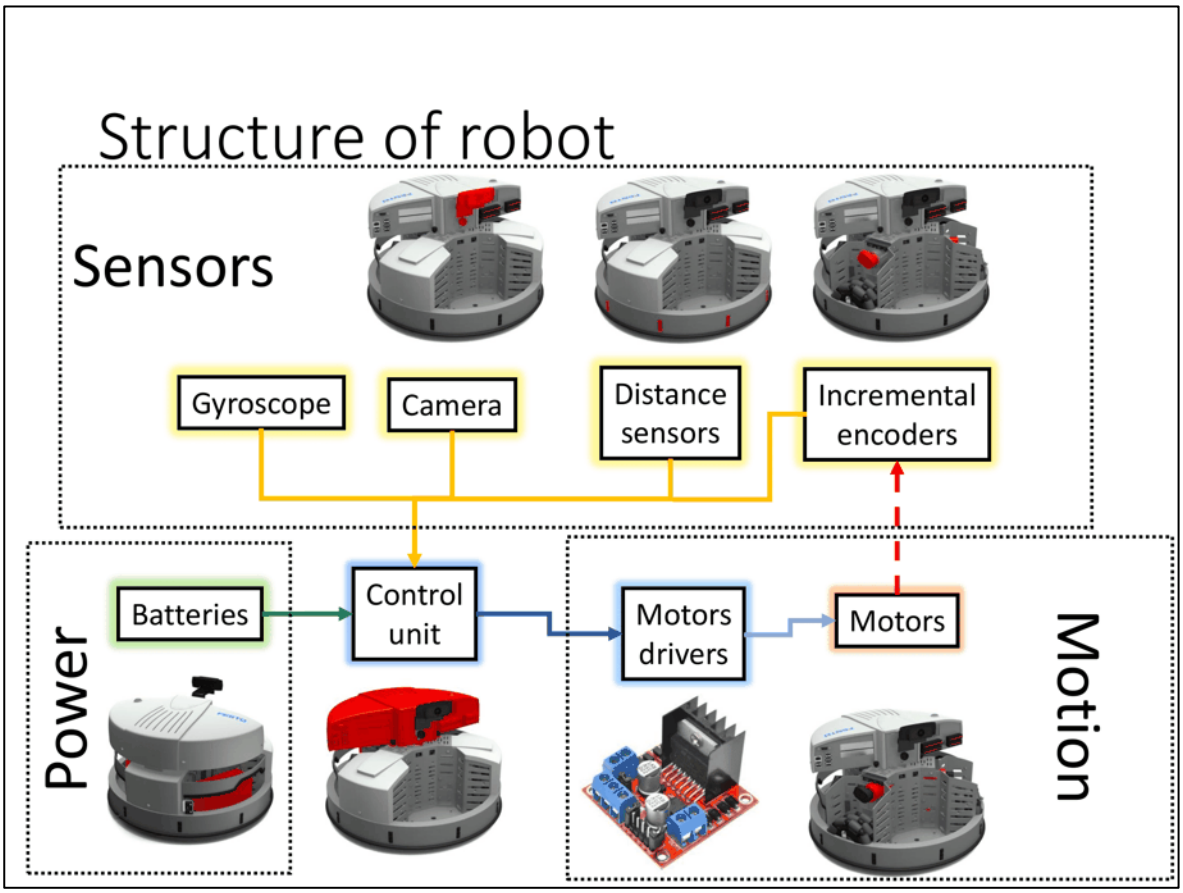


Structure of robot

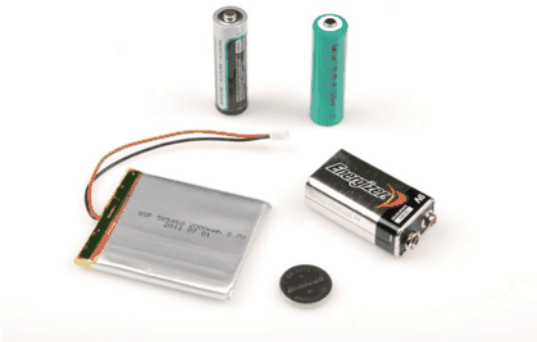
Hardware

Control	Drive systems	Sensors	Interfaces	Supply
→ Power switch	→ Omnidrive	→ Bumper	→ WLAN	→ Batteries
→ Control unit	→ Motors	→ Distance sensors	→ I/O-Interfaces	→ Power supply unit
→ Embedded PC	→ Incremental encoder	→ Gyroscope	→ Motor/encoder	→ Charging electronics
→ Microcontroller	→ Gear units	→ Camera	→ USB	→ Pedestal
→ Reset button	→ Wheels	→ Opto-electronic sensors	→ PCI Express	
		→ Inductive sensors	→ Ethernet	
			→ VGA	

Structure of robot



Power supply



- Primary Batteries
- Secondary Batteries
  - Lithium (Li-ion, Li-pol)
  - Nickel Cadmium (Ni-Cd)
  - Nickel-Metal Hydride (Ni-MH)
  - Lead-Acid

Schematic symbols

Single cell Multi-cell

Power supply

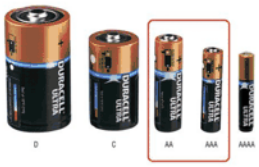
Terminology

- **Capacity** - Batteries have different ratings for the amount of power a given battery can store. When a battery is fully charged, the capacity is the amount of power it contains. Batteries of the same type will often be rated by the amount of current they can output over time. For example, there are 1000mAh (milli-Amp Hour) and 2000mAh batteries.
- **Nominal Cell Voltage** - The average voltage a cell outputs when charged. The nominal voltage of a battery depends on the chemical reaction behind it. A lead-acid car battery will output 12V. A lithium coin cell battery will output 3V.
- The key word here is "nominal", the actual measured voltage on a battery will decrease as it discharges. A fully charged LiPo battery will produce about 4.23V, while when discharged its voltage may be closer to 2.7V.
- **Shape** - Batteries come in many sizes and shapes. The term 'AA' references a specific shape and style of a cell. There are a large variety.

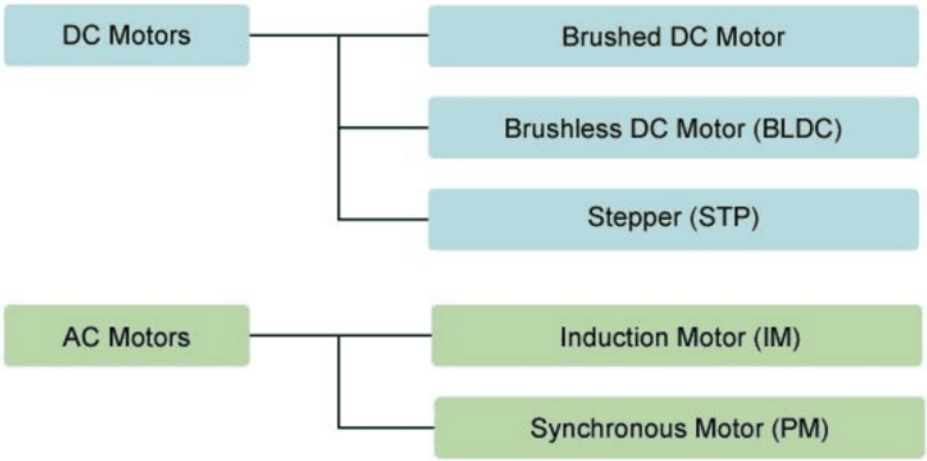
Power supply

Common batteries, their chemistry, and their nominal voltage

Battery Shape	Chemistry	Nominal Voltage	Rechargeable?
AA, AAA, C, and D	Alkaline or Zinc-carbon	1.5V	No
9V	Alkaline or Zinc-carbon	9V	No
Coin Cell	Lithium	3V	No
Silver Flat Pack	Lithium Polymer (LiPo)	3.7V	Yes
AA, AAA, C, D (Rechargeable)	NiMH or NiCd	1.2V	Yes
Car Battery	Six-cell lead acid	12.6V	Yes

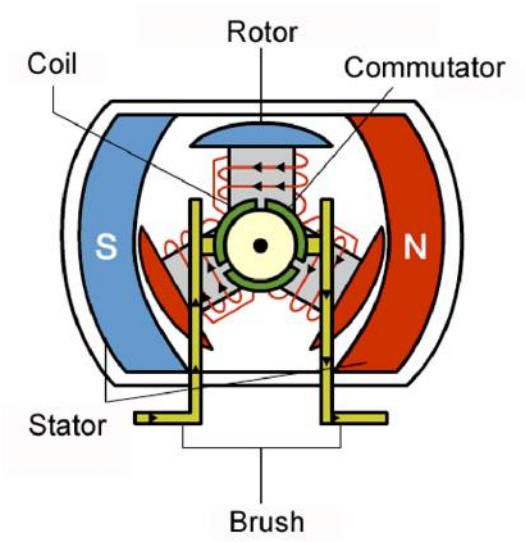


Motion – motors



Brushed DC electric motor

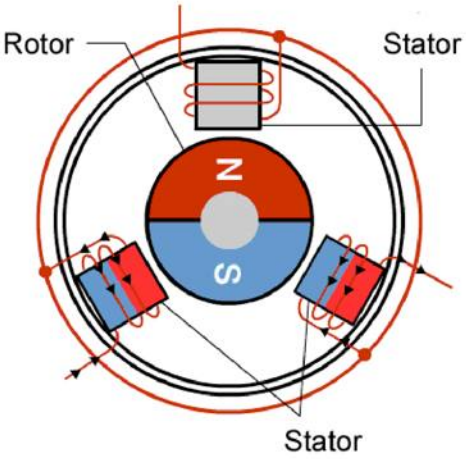
Fixed brushes supply electric energy to the rotating commutator. As the commutator rotates, it continually flips the direction of the current into the coils, reversing the coil polarities so that the coils maintain rightward rotation. The commutator rotates because it is attached to the rotor on which the coils are mounted.



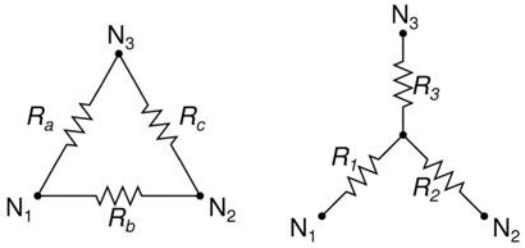
Schematic symbols

Brushless DC electric motor

Since the rotor is a permanent magnet, it needs no current, eliminating the need for brushes and commutator. Current to the fixed coils is controlled from the outside.

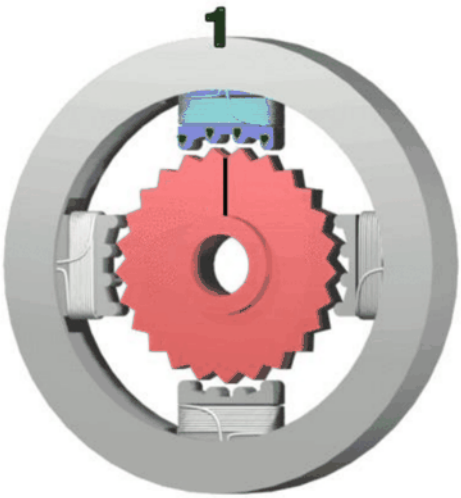


Schematic for delta and wye winding styles. (This image does not illustrate the motor's inductive and generator-like properties)

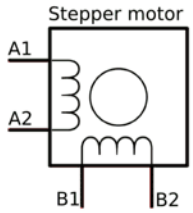


Stepper motor

A stepper motor, also known as step motor or stepping motor, is a brushless DC electric motor that divides a full rotation into a number of equal steps. The motor's position can then be commanded to move and hold at one of these steps without any position sensor for feedback (an open-loop controller), as long as the motor is carefully sized to the application in respect to torque and speed.

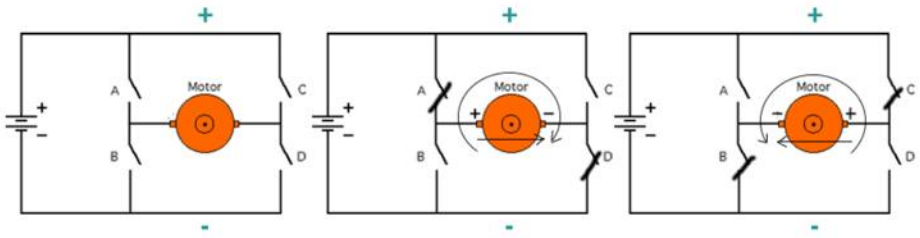


Schematic symbols

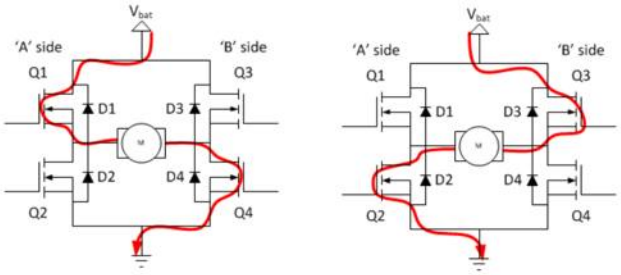


Controlling Brushed DC Motors

Rotation in different directions

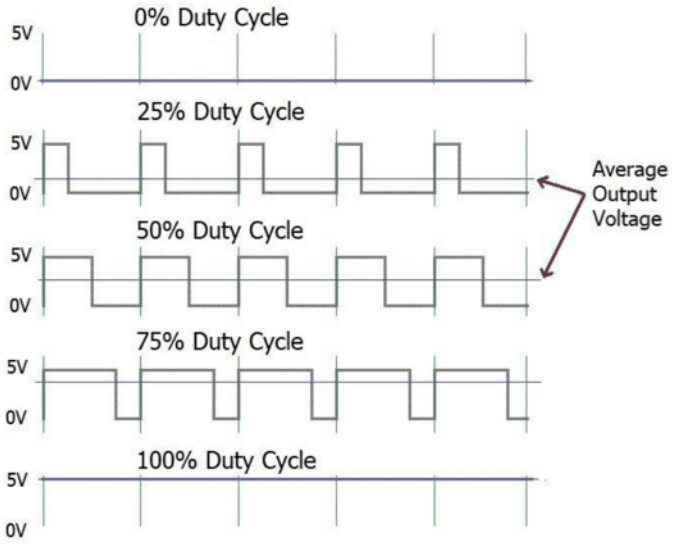


Schematic for microcontroller use

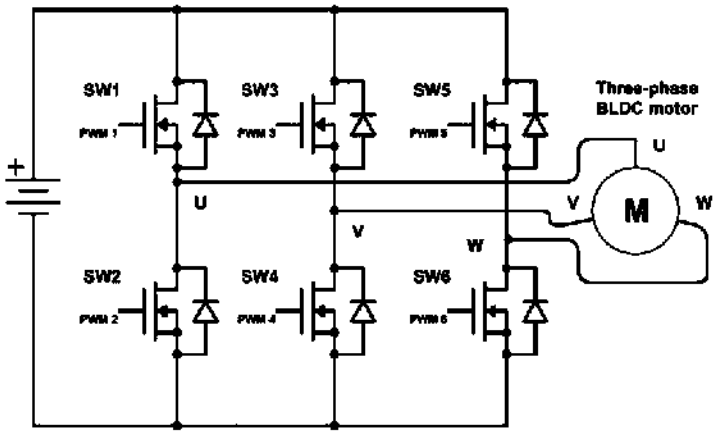


Controlling Brushed DC Motors

Control speed by PWM (Pulse-Wide Modulation)



Controlling Brushless DC Motors

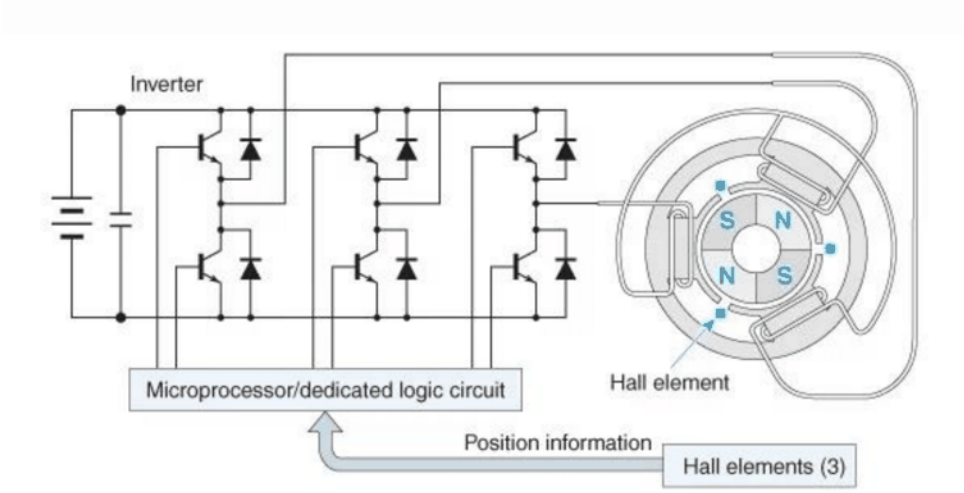


Controlling Brushless DC Motors

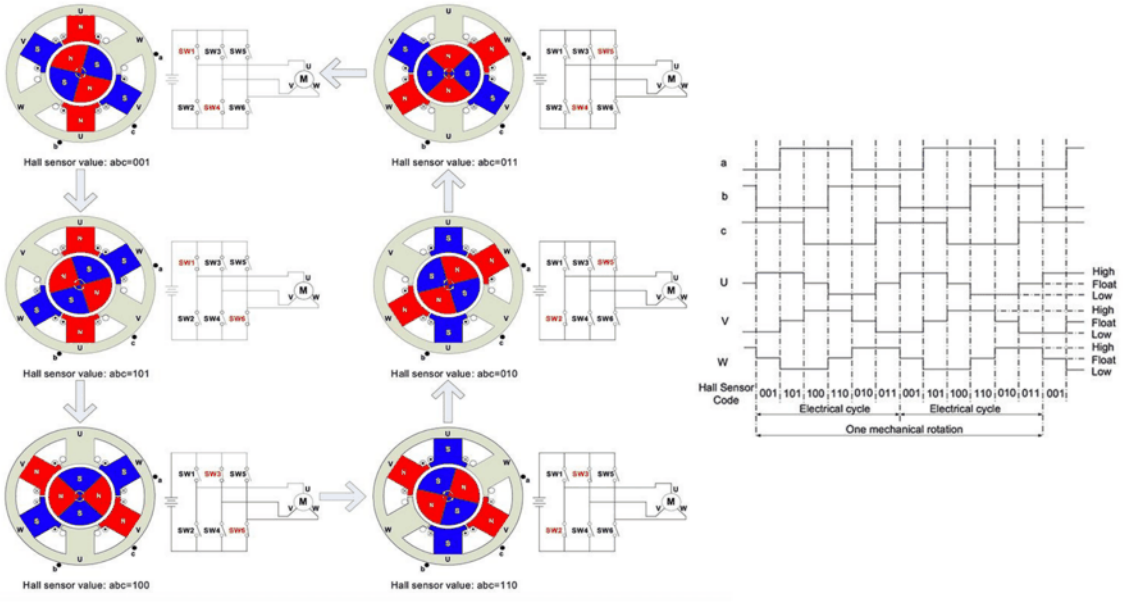
Sensored vs. sensorless

- Two technologies offer a solution for positional feedback. The first and most common uses three Hall-effect sensors embedded in the stator and arranged at equal intervals, typically 60° or 120°. A second, 'sensorless' control technology comes into its own for BLDC motors that require minimal electrical connections.

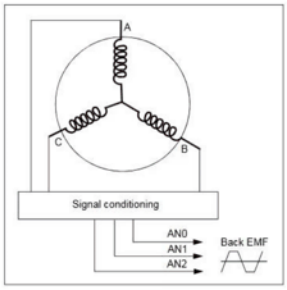
Controlling Brushless DC Motors with sensors

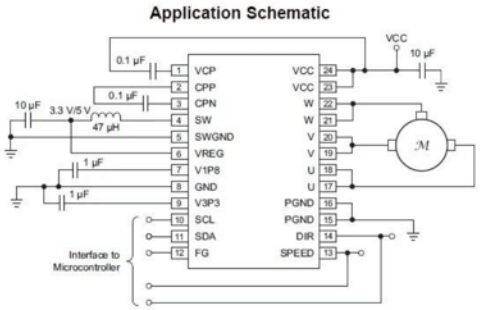


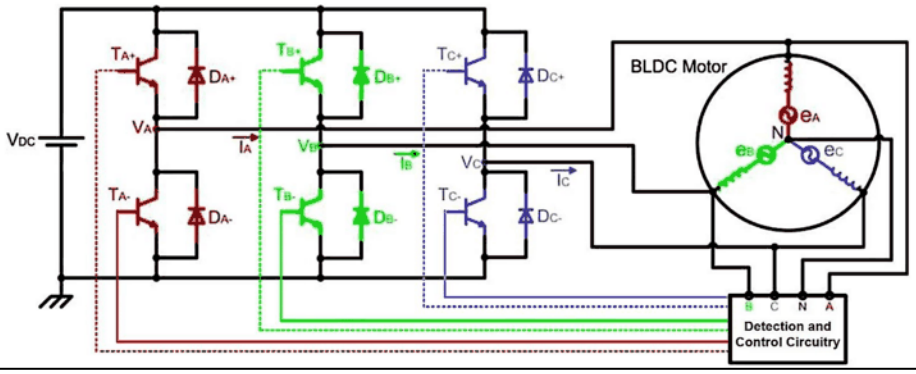
Controlling Brushless DC Motors with sensors



Controlling Brushless Sensorless DC Motors

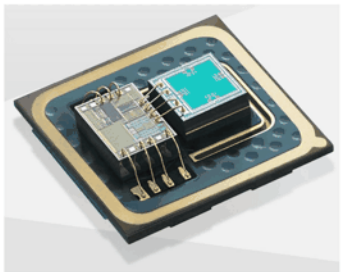






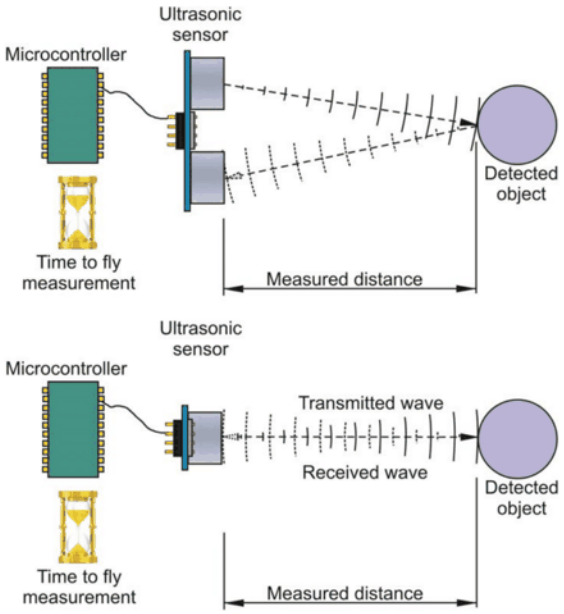
Sensors

- Distance
- Position
- Velocity
- Temperature



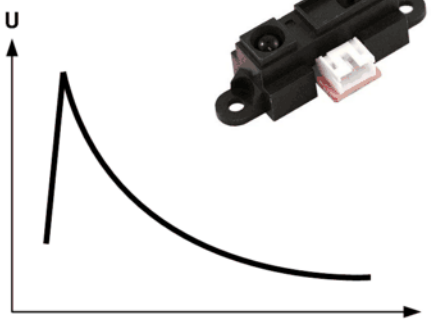
Distance sensors

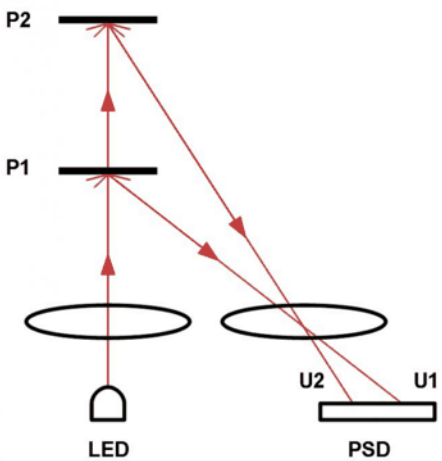
- Ultrasonic



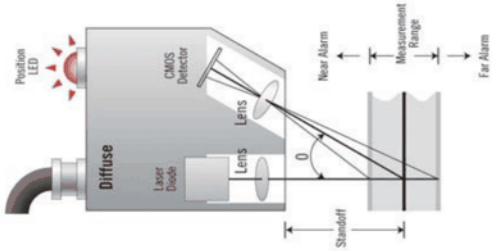
Distance sensors

- Infrared





- Laser

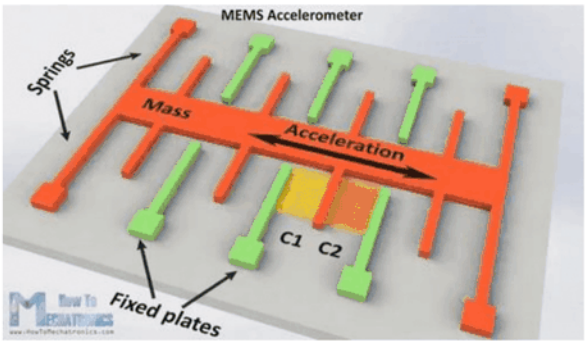


Position/Velocity sensors

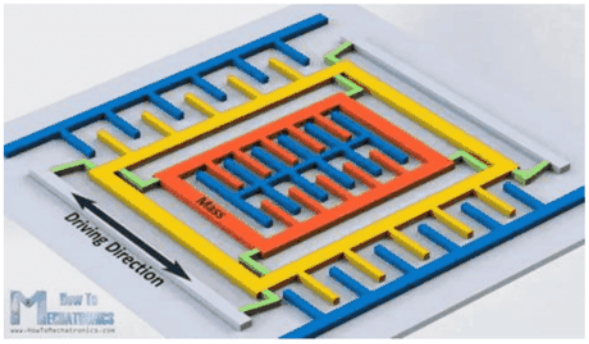
- MEMS (Micro Electro Mechanical Systems)
  - Accelerometer
  - Gyroscope
  - Magnetometer
- Encoder
- Potentiometer

MEMS sensors

- Accelerometer

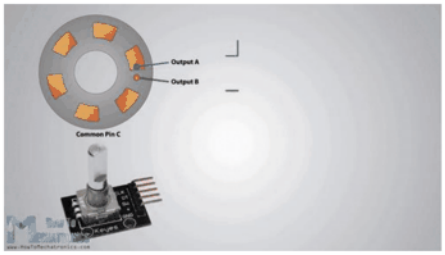


- Gyroscope

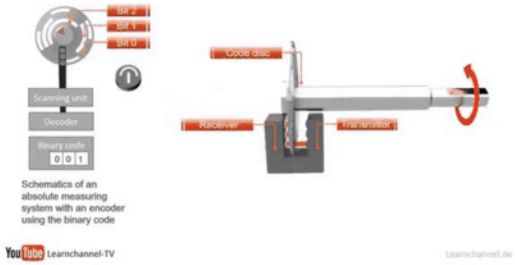


Encoders

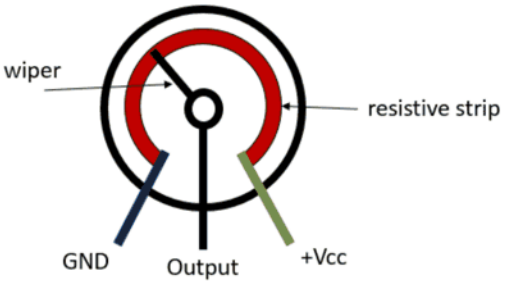
- Mechanical
- Optical

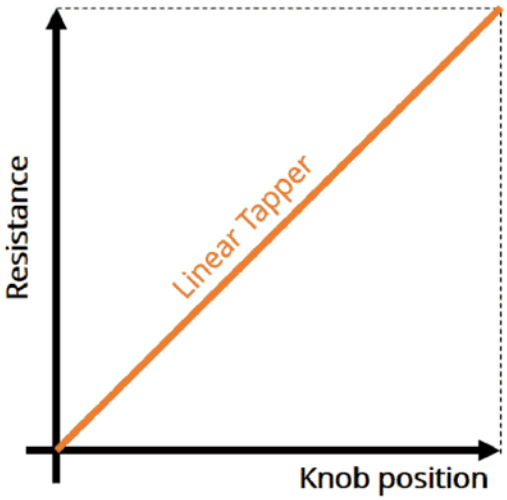


- Incremental
- Absolute



Potentiometer

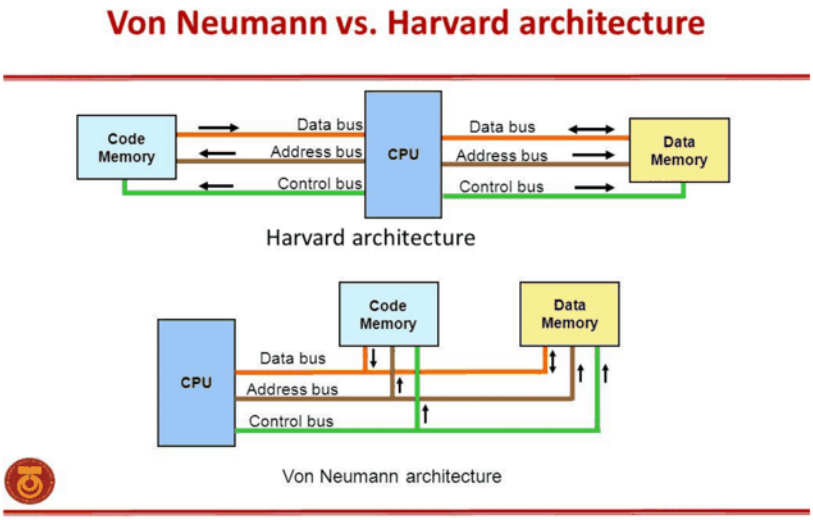




Microcontrollers

A micro-controller can be comparable to a little stand alone computer; it is an extremely powerful device, which is able of executing a series of pre-programmed tasks and interacting with extra hardware devices. Being packed in a tiny integrated circuit (IC) whose size and weight is regularly negligible, it is becoming the perfect controller for as robots or any machines required some type of intelligent automation. A single microcontroller can be enough to manage a small mobile robot, an automatic washer machine or a security system. Several microcontrollers contains a memory to store the program to be executed, and a lot of input/output lines that can be a used to act jointly with other devices, like reading the state of a sensor or controlling a motor.

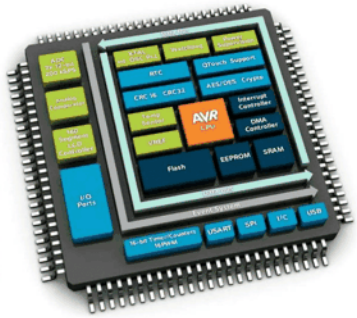
Microcontroller's architecture

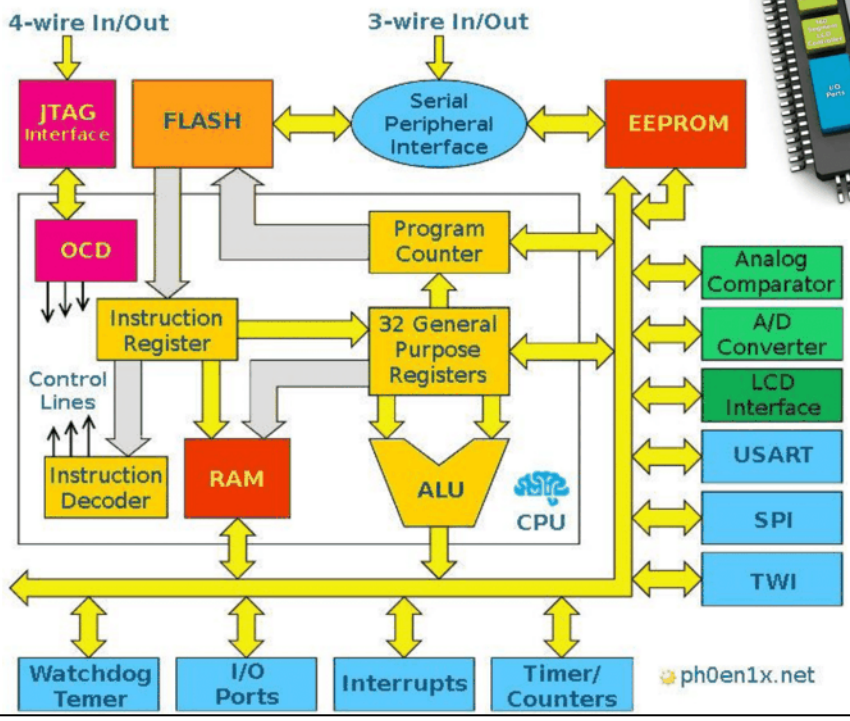


Microcontrollers

	8051	PIC	AVR	ARM
Bus width	8-bit for standard core	8/16/32-bit	8/32-bit	32-bit mostly also available in 64-bit
Communication Protocols	UART, USART,SPI,I2C	PIC, UART, USART, LIN, CAN, Ethernet, SPI, I2S	UART, USART, SPI, I2C, (special purpose AVR support CAN, USB, Ethernet)	UART, USART, LIN, I2C, SPI, CAN, USB, Ethernet, I2S, DSP, SAI (serial audio interface), IrDA
Speed	12 Clock/instruction cycle	4 Clock/instruction cycle	1 clock/ instruction cycle	1 clock/ instruction cycle
Memory	ROM, SRAM, FLASH	SRAM, FLASH	Flash, SRAM, EEPROM	Flash, SDRAM, EEPROM
ISA	CLSC	Some feature of RISC	RISC	RISC
Memory Architecture	Von Neumann architecture	Harvard architecture	Modified	Modified Harvard architecture
Power Consumption	Average	Low	Low	Low
Families	8051 variants	PIC16,PIC17, PIC18, PIC24, PIC32	Tiny, Atmega, Xmega, special purpose AVR	ARMv4,5,6,7 and series
Community	Vast	Very Good	Very Good	Vast
Manufacturer	NXP, Atmel, Silicon Labs, Dallas, Cyprus, Infineon, etc.	Microchip Average	Atmel	Apple, Nvidia, Qualcomm, Samsung Electronics, and TI etc.
Cost (as compared to features provide)	Very Low	Average	Average	Low
Other Feature	Known for its Standard	Cheap	Cheap, effective	High speed operation Vast
Popular Microcontrollers	AT89C51, P89v51, etc.	PIC18fXX8, PIC16f88X, PIC32MXX	Atmega8, 16, 32, Arduino Community	LPC2148, ARM Cortex-M0 to ARM Cortex-M7, etc.

AVR Architecture

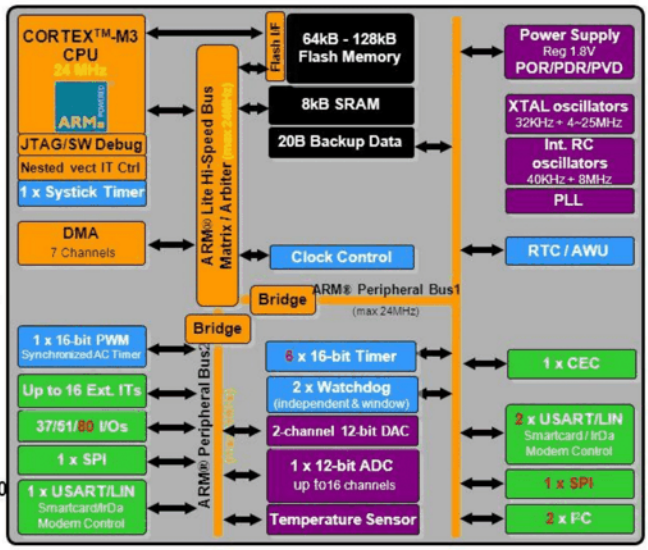




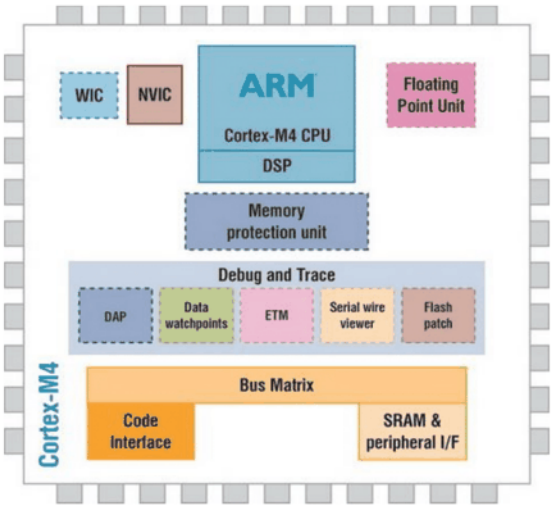
ARM – STM32 Architecture

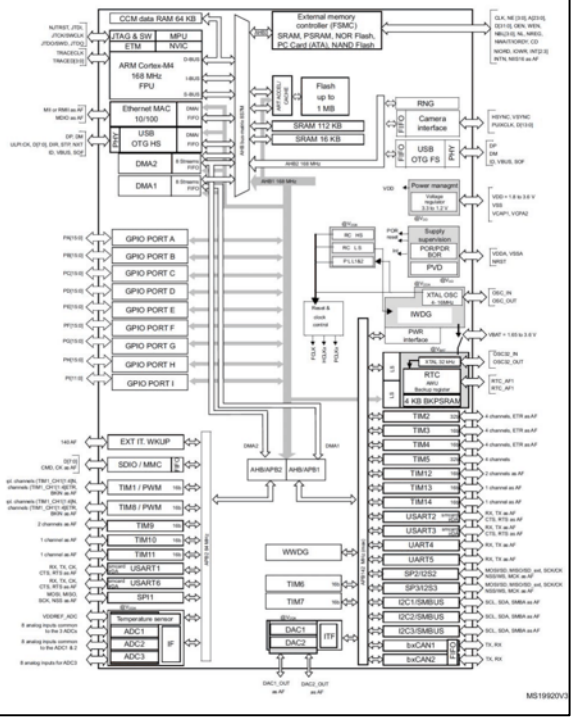
STM32 Value line 64K-128KBytes block diagram

- **Core and operating conditions**
  - ARM® Cortex™-M3 1.25 DMIPS/MHz up to 24 MHz
  - 2.0 V to 3.6 V range
  - -40 to +105 °C
- **Rich connectivity**
  - 8 communications peripherals
- **Advanced analog**
  - 12-bit1.2 µs conversion time ADC
  - Dual channel 12-bit DAC
- **Enhanced control**
  - 16-bit motor control timer
  - 6x 16-bit PWM timers
- **LQFP48, LQFP/BGA64, LQFP100**

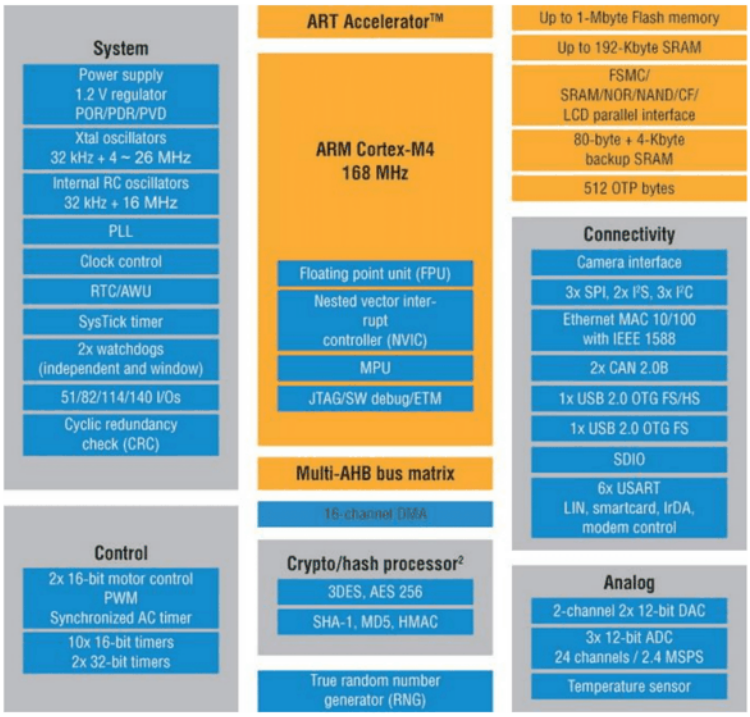


STM32F40xxx architecture

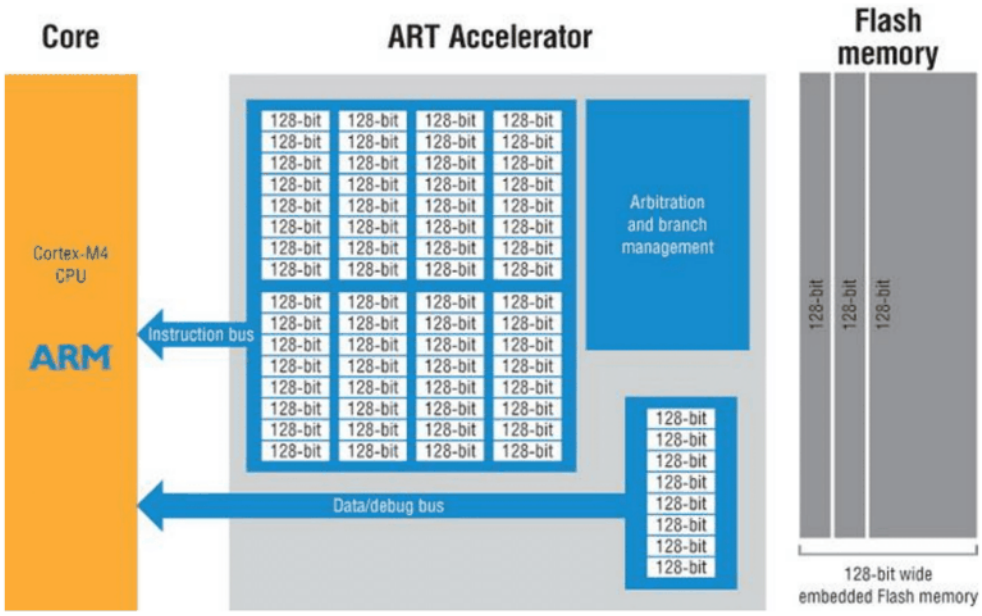




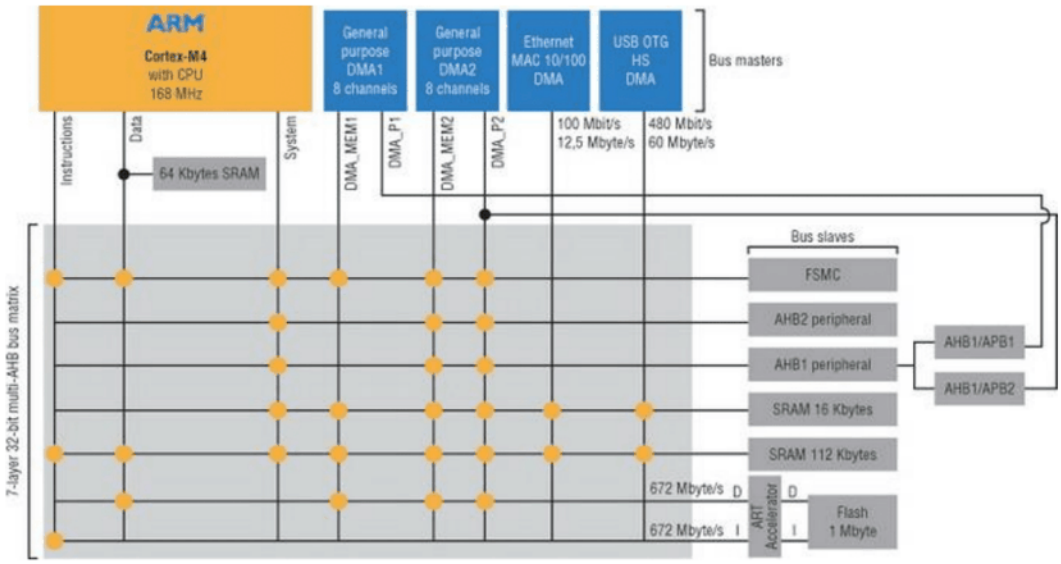
STM32F4xxx Structure



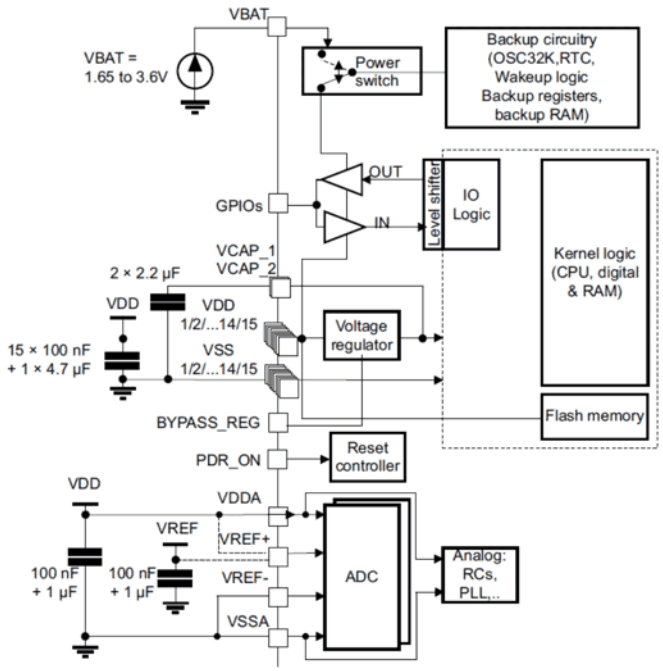
STM Adaptive Real Time Memory



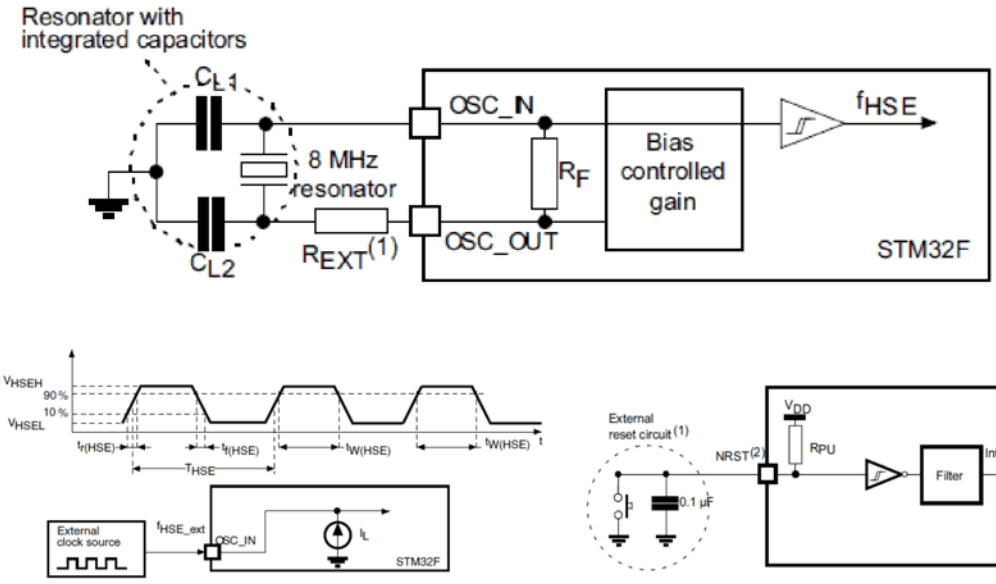
STM Bus matrix



Power supply scheme



External clocking and reset circuit



Actuators and mobile robots control
Mathematical model of DC motor

Control and modeling of mobile robots

Mathematical model of DC motor

Alexander A. Kapitonov

Constructon of DC motor

Figure 1. DC motor assembled.

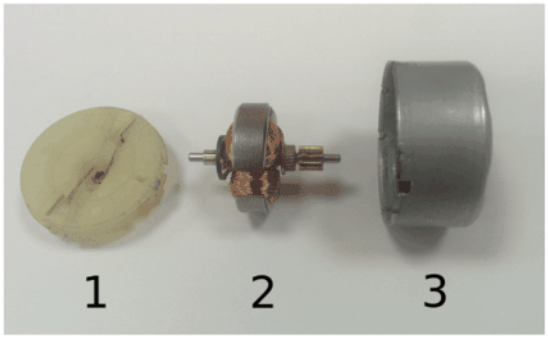


Figure 2. DC motor disassembled:
1 — cap, 2 — rotor, 3 — stator.

Constructon of DC motor

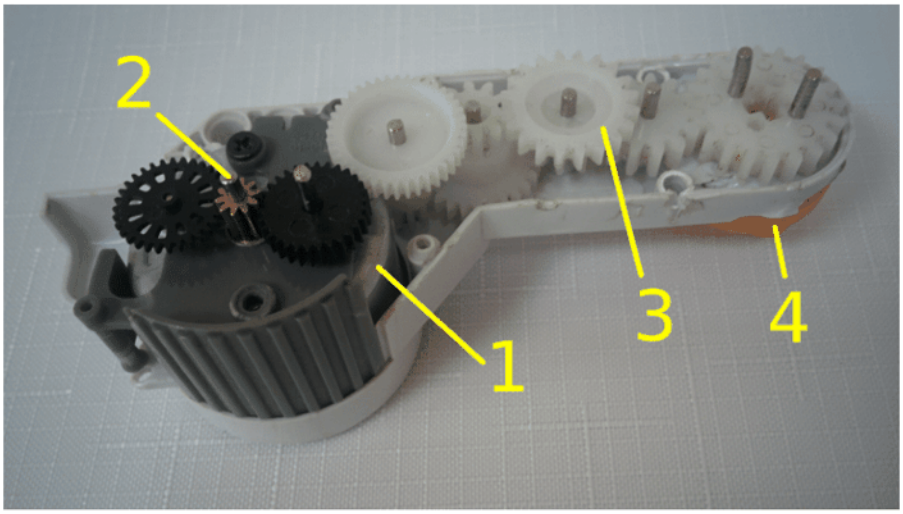


Figure 3. NXT motor disassembled: 1 — DC motor 2 — it's shaft, 3 — reducer, 4 — NXT motor external shaft; (chip with encoder isn't shown).

Mathematical model

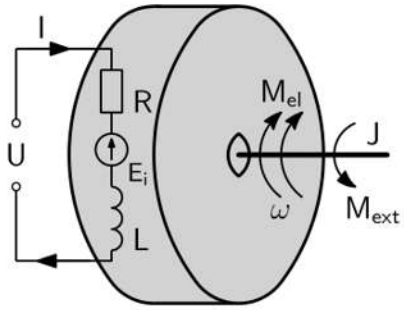


Figure 4. Physical scheme of DC motor.

$$\begin{cases} M_{el} - M_{ext} = J\dot{\omega}, \\ U = RI + E_i + L\dot{I}, \end{cases} \quad (1)$$

$$M_{el} = k_m I, \quad (2)$$

$$E_i = k_e \omega, \quad (3)$$

$$\begin{cases} k_m I - M_{ext} = J\dot{\omega}, \\ U = RI + k_e\omega + L\dot{I}, \end{cases} \quad (4)$$

where M_{el} — motor torque; M_{ext} — torque of external forces; J — total moment of inertia of the rotor and reducer's gears; ω — rotor speed; U — motor supply voltage; R, L — resistance and an inductance of rotor's wires; I — the current flowing through the latter; E_i — EMF which appeared in rotor's wires due to its rotation in magnetic field of stator's magnets; k_m, k_e — torque and back EMF motor constants.

Mathematical model

If $L \approx 0$ H, then

$$I = \frac{1}{R}U - \frac{k_e}{R}\omega, \tag{5}$$

therefore

$$k_m\left(\frac{1}{R}U - \frac{k_e}{R}\omega\right) - M_{ext} = J\dot{\omega}, \tag{6}$$

hence

$$\frac{JR}{k_m k_e}\dot{\omega} + \omega - \frac{1}{k_e}U - \frac{R}{k_m k_e}M_{ext}, \tag{7}$$

$$T_m\dot{\omega} + \omega = \frac{1}{k_e}U - \frac{T_m}{J}M_{ext}, \tag{8}$$

where $T_m = \dfrac{JR}{k_m k_e}$ is a motor mechanical constant.

Mathematical model

Also we can get differential equation which contains I, not ω:

1. differentiating (5):

$$\dot{\omega} = \frac{1}{k_e}\dot{U} - \frac{R}{ke}\dot{I} \tag{9}$$

2. putting (9) to the first equation from (4):

$$k_m I - M_{ext} = J\left(\frac{1}{k_e}\dot{U} - \frac{R}{ke}\dot{I}\right) \tag{10}$$

3. transforming (10):

$$\frac{JR}{k_m k_e}\dot{I} + I = \frac{J}{k_m k_e}\dot{U} + \frac{1}{k_m}M_{ext}, \tag{11}$$

$$T_m\dot{I} + I - \frac{T_m}{R}\dot{U} + \frac{1}{k_m}M_{ext}. \tag{12}$$

Mathematical model

For a situation when

$$\begin{cases} U = const, \\ M_{ext} = 0\ N \cdot m, \end{cases} \tag{13}$$

and

$$\begin{cases} \theta(0) = 0, \\ \omega(0) = 0\ s,^{1} \end{cases} \tag{14}$$

where θ — angle of rotor's rotation ($\dot{\theta} = \omega$), next expressions for $\omega(t)$ and $\theta(t)$ can be obtained from (8):

$$\omega(t) = \omega_{nls}\left(1 - \exp\left(\frac{t}{T_m}\right)\right), \tag{15}$$

$$\theta(t) = \omega_{nls}t - \omega_{nls}T_m + \omega_{nls}T_m \exp\left(\frac{t}{T_m}\right), \tag{16}$$

where $\omega_{nls} = U/k_e$ — no-load speed of the rotor.

Mathematical model

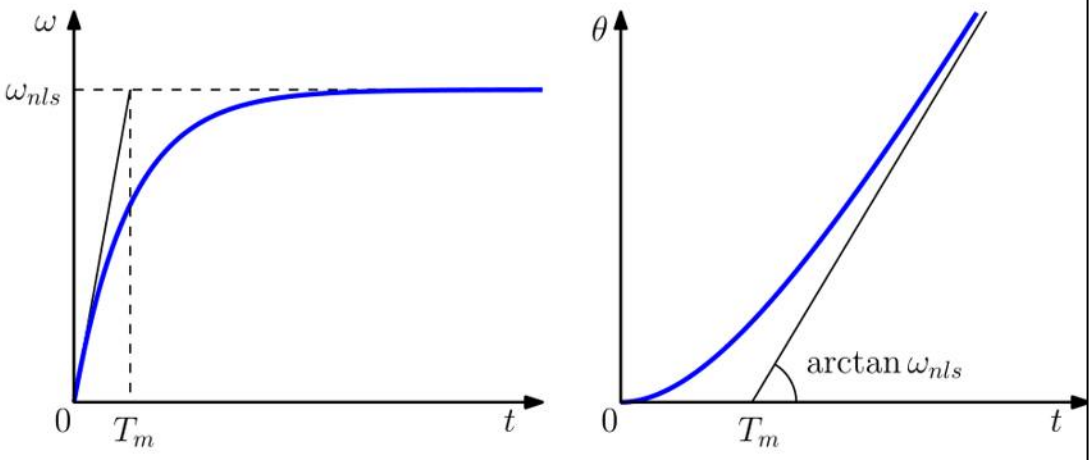


Figure 5. Graphs of $\omega(t)$ and $\theta(t)$ from (15) and (16) in case $\omega_{nls} > 0$.

Description of experiment

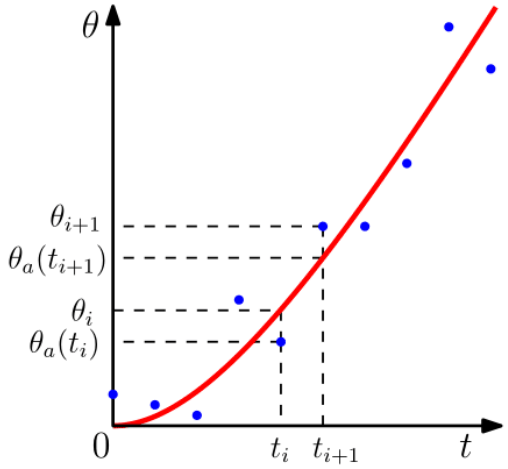


Figure 6. Approximation curve.

Least squares method:

find values for ω_{nls} and T_m such that the sum S:

$$S = \sum_{j=1}^{N} \left(\theta_a(t_j) - \theta_j\right)^2 \qquad (17)$$

would have a minimal possible value.

There
N — number of pairs (t_j, θ_j) which were recorded during the experiment,
$\theta_a(t_j)$ — value of (16) when $t = t_j$.

Modeling scheme of DC motor in Scilab

Control and modeling of mobile robots

Modeling scheme of DC motor in Scilab

Alexander A. Kapitonov

Scilab

Scilab is free and open source software for numerical computation providing a powerful computing environment for engineering and scientific applications.[1]

Figure 1. Scilab logo.

At this course we will use Scilab's:

- Xcos — hybrid dynamic systems modeler and simulator;
- some mathematical algorithms.

[1] Logo and some text on this slide were taken from www.scilab.org.

System modeling

Figure 2 demonstrates example of modeling scheme for device which is desribed by this system of equations:

$$\begin{cases} \dot{x} = Ax + Bu, \\ y = Cx + Du, \\ u = K(x_d - x) + Lx_g. \end{cases} \tag{1}$$

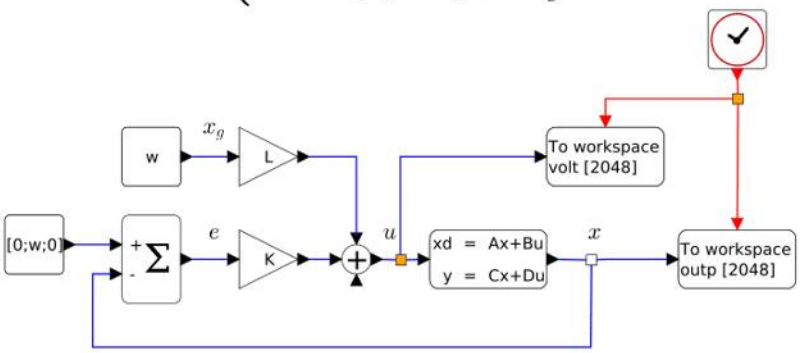
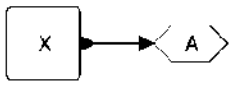


Figure 2. Example of modeling scheme.

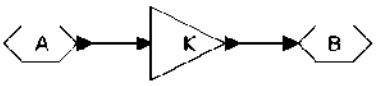
System modeling

Const value generator



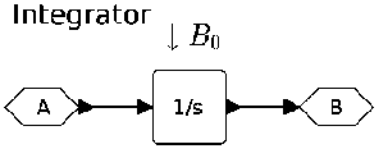
$A - X - const$

Proportional gain



$B - K \cdot A$

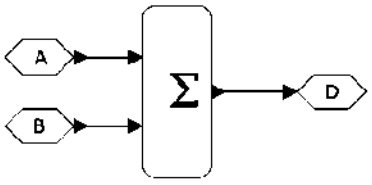
Integrator



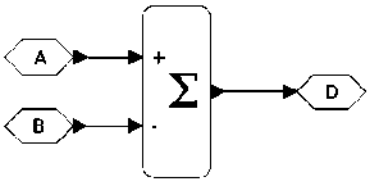
$$B = \int A\,dt + B_0$$

Figure 3. Some standard blocks.

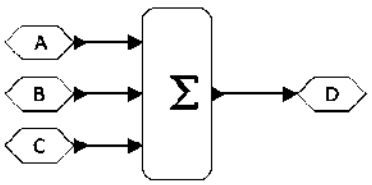
System modeling



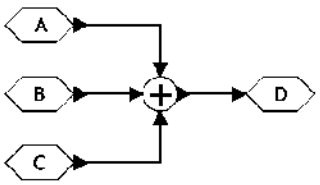
$D = A + B$



$D = A - B$



$D = A + B + C$



$D = A + B + C$

Figure 4. Summator block.

System modeling

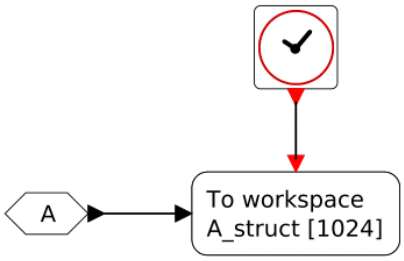


Figure 5. Some service blocks.

This subscheme saves values of A and appropriate moments of time into two matrices: `A_struct.values` and `A_struct.time`.

System modeling

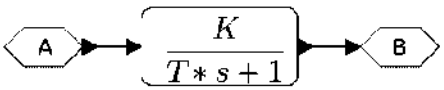


Figure 6. Transfer function block.

$$T \cdot \dot{B}(t) + B(t) = K \cdot A(t), \quad (2)$$

$$\mathcal{L}\{T \cdot \dot{B}(t) + B(t)\} = \mathcal{L}\{K \cdot A(t)\}, \quad (3)$$

$$T \cdot s \cdot B(s) - B(s) = K \cdot A(s), \quad (4)$$

$$\frac{B(s)}{A(s)} = \frac{K}{Ts+1}, \quad (5)$$

where $L\{\}$ — Laplace transform.

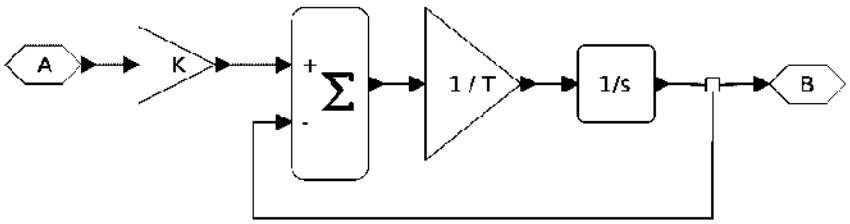


Figure 7. An equivalent sheme.

Modeling scheme of DC motor

Model of DC motor is described by two diffential equations:

$$\begin{cases} T_m \dot{\omega} + \omega = \dfrac{1}{k_e} U - \dfrac{T_m}{J} M_{ext}, \\ T_m \dot{I} + I = \dfrac{T_m}{R} \dot{U} + \dfrac{1}{k_m} M_{ext}, \end{cases} \tag{6}$$

therefore its modeling scheme is equal to one which is demonstrated by figure 8.

Modeling scheme of DC motor

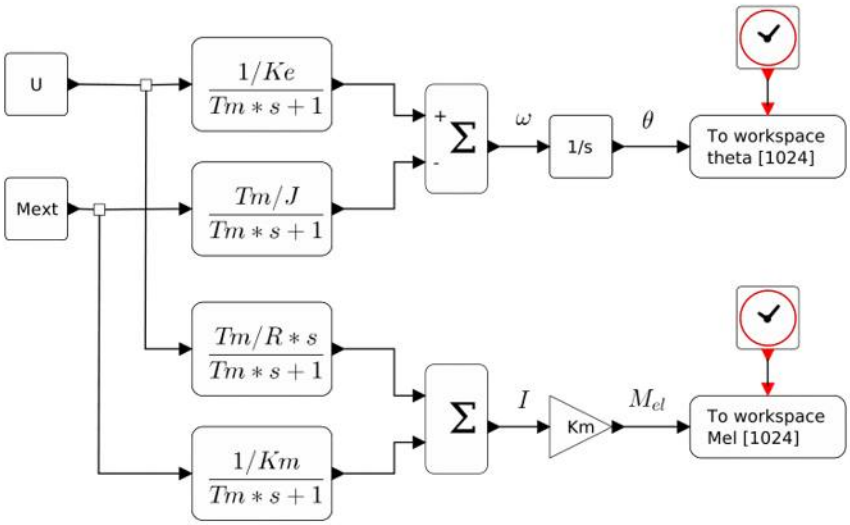


Figure 8. Modeling scheme of DC motor.

Control of DC motor using PID controller

Control and modeling of mobile robots

Control of DC motor using PID regulator

Alexander A. Kapitonov

Some basics of control theory

In a control theory all systems are considered as a single object or a "box" which has some number of input and output signals.

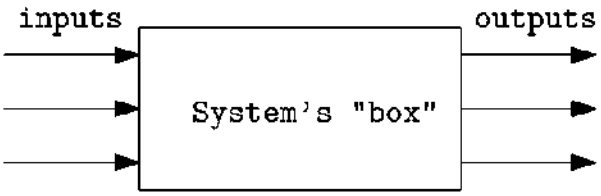


Figure 1. One of a possible representation of every system in a control theory.

input signals — some impacts which change system state

output signals — some physical quantities which describe system state

Some basics of control theory

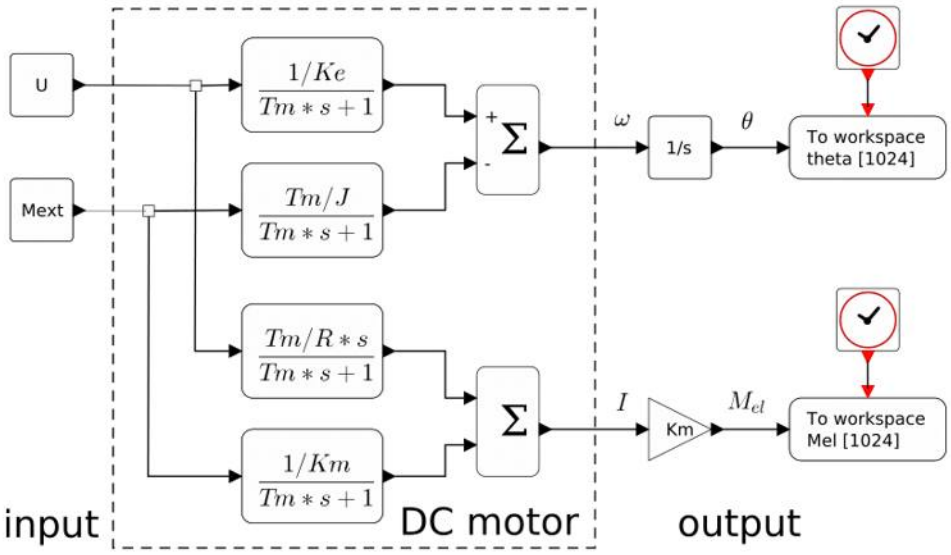


Figure 2. Structure of the model of a DC motor.

Some basics of control theory

Some important definitions:

Control is a process of changing in a desired way values of some output signals using some input signals.

Controller is a special device and/or algorithm which creates required input signals.

Methods of control:

- forward
- using feedback

Some basics of control theory

Forward control — a method of control when a controller doesn't use information about values of system's output signals.

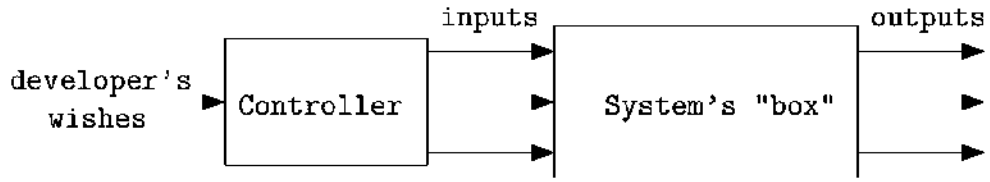


Figure 3. Scheme of forward control.

Some basics of control theory

Control with feedback — a method of control when a controller use information about values of system's output signals.

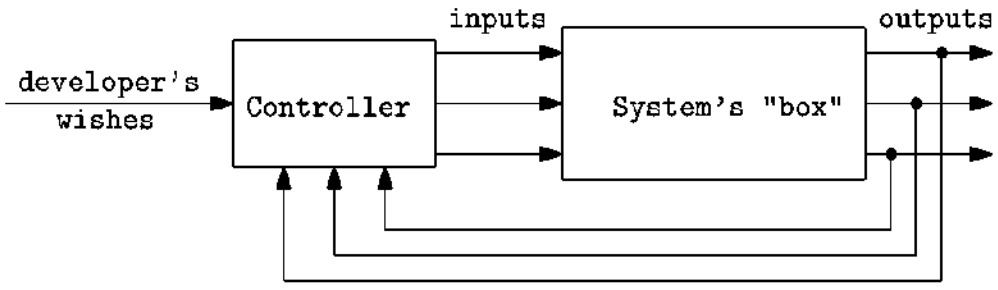


Figure 4. Scheme of control with feedback.

PID controller

PID controller is an algorithm of feedback control which calculate value for input signal in accordance to the formulas:

$$e(t) = x_d(t) - x(t), \tag{1}$$

$$u = K_p \cdot e + K_i \cdot \int e\,dt + K_d \dot{e}, \tag{2}$$

where x — controllable output signal; x_d — desired value of signal; e — error of control; u — used system's input signal; K_p, K_i, K_d — constant coefficients of PID controller.

PID controller

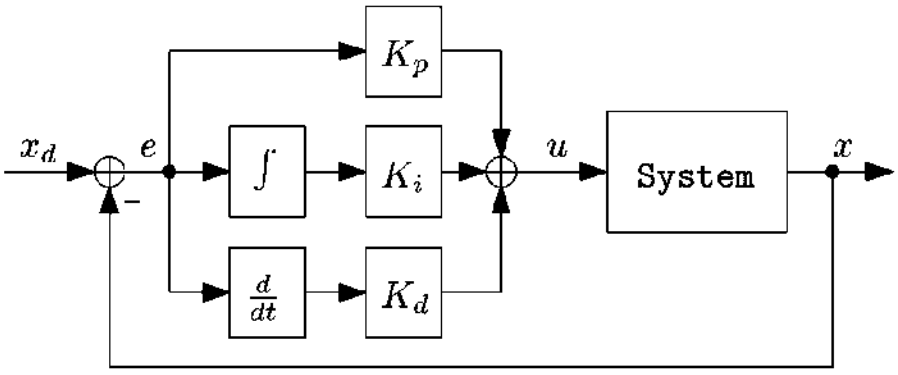


Figure 5. Scheme of PID controller sctructure.

PID controller

P controlller or a proportional piece of PID which is calculated as

$$u = K_p \cdot e, \tag{3}$$

does the main part of a controller's job;

I controlller or a piece of PID with integral which is calculated as

$$u = K_i \cdot \int e\,dt, \tag{4}$$

prevents errors (makes e is being equal to 0);

D controlller or a piece of PID with derivative which is calculated as

$$u = K_d \cdot \dot{e}, \tag{5}$$

dampens oscillations.

PID controller

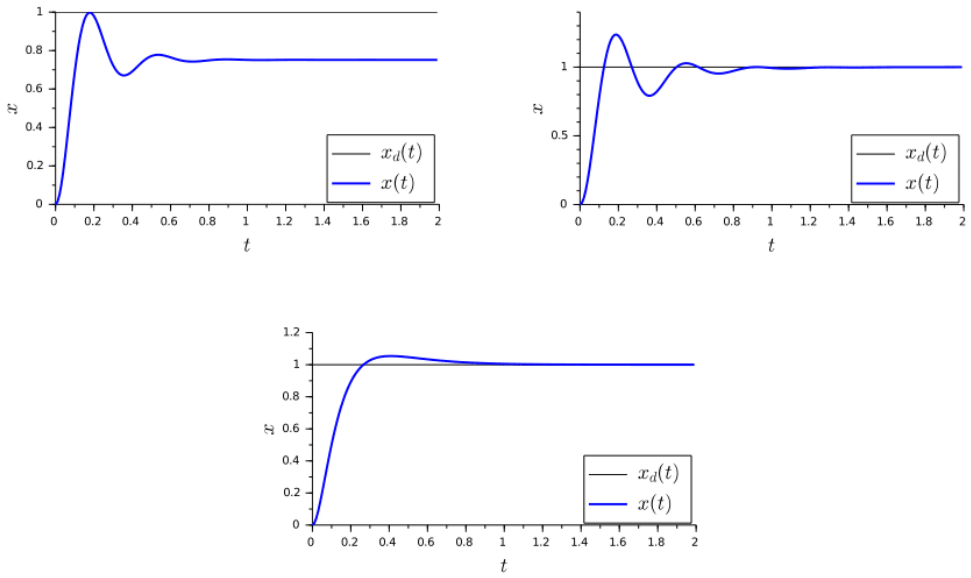


Figure 6. System with P, PI and PID controller respectively.

PID controller

Methods of tuning controller's coefficients:

- calculations using mathematical model of a controllable object;
- setting with according to one of a special algorithm;
- fully manual setting.

Ziegler–Nichols method

Algorithm of tuning values for coefficients of PID controller:

1. make K_i and K_d is being equal to 0;
2. increase value of K_p until $x(t)$ starts making undamped oscillations; remember this value of K_p as K_u and a period of the oscillations as T_u;
3. calculate coefficients of PID controller using these formulas:

$$K_p = 0.6K_u; \qquad K_i = \frac{2K_p}{T_u}, \qquad K_d = \frac{K_p T_u}{8}. \tag{6}$$

Ziegler–Nichols method

This method's strengths:

- it is quite simple.

This method's weaknesses:

- it doesn't work for all systems;
- it doesn't give the best value of coefficients.

Numerical methods

$$\int_{t_1}^{t_2} f\,dt = S_{1,2} \tag{7}$$

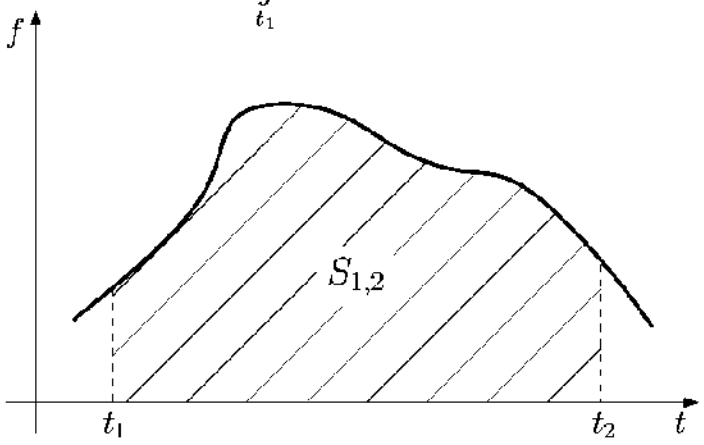


Figure 7. Geometry meaning of integrals.

Numerical methods

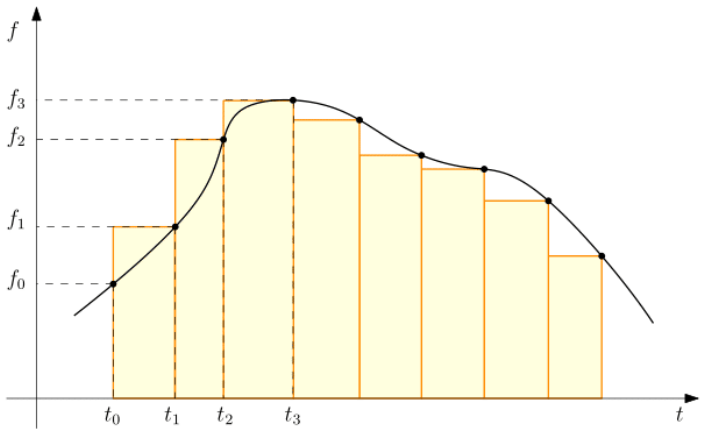


Figure 8. One of numerical methods for calculating value of integral.

$$\int_{t_m}^{t_n} f\,dt \approx \sum_{i=m+1}^{n} f_i(t_i - t_{i-1}), \quad m < n, \;\; m, n \in \mathbb{Z} \tag{8}$$

Numerical methods

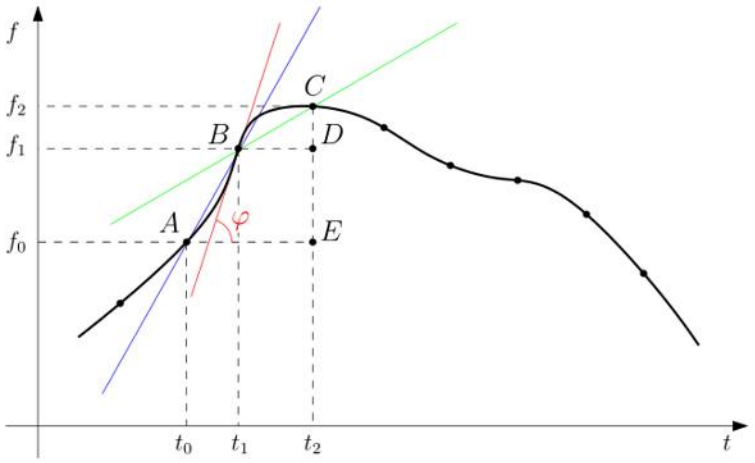


Figure 9. Numerical methods for derivative calculating.

$$f'(t_1) = \lim_{\Delta t \to 0} \frac{f(t_1 + \Delta t) - f(t_1)}{\Delta t} = \operatorname{tg}\varphi, \quad f'(t_1) \approx \frac{f_1 - f_0}{t_1 - t_0} = \operatorname{tg}\angle BAE$$

A controller for to-point motion for a mobile robot with differential drive type

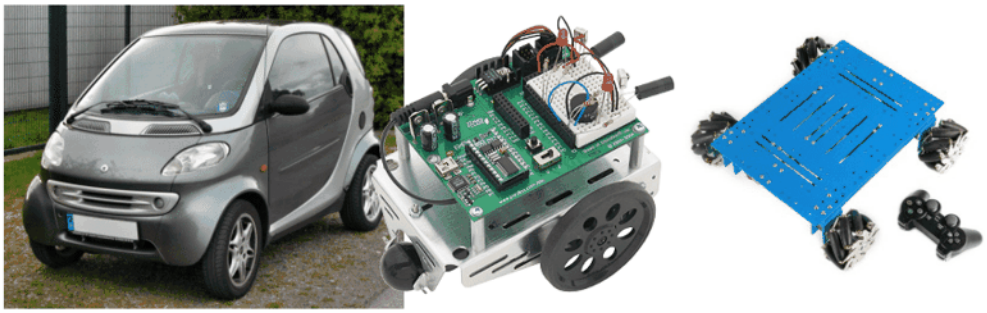
Control and modeling of mobile robots

A controller for to-point motion for a mobile robot with differential drive type

Alexander A. Kapitonov

Robots' drive types

Drive type	Controllable velocities
Car-like type	v_x, $\omega(v_x)$
Differential	v_x, ω
Omnidirectional	v_x, v_y, ω



a) b) c)

Figure 1. Examples of "robots" with different drive types: a—car-like, b—differential, c—omnidirectional

General view of the robot

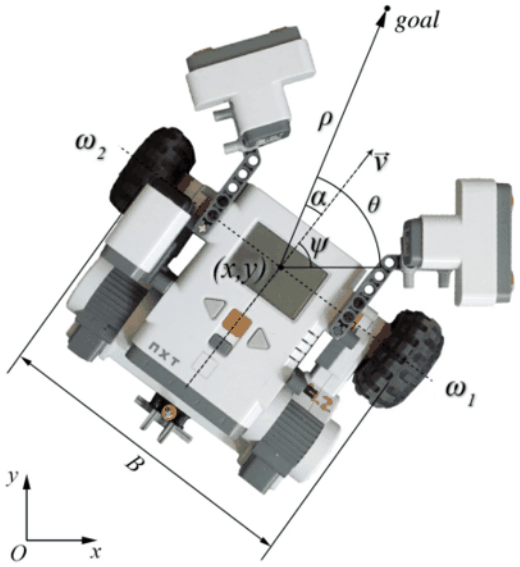


Figure 2. General view of a considered robot.

Structure of the control system

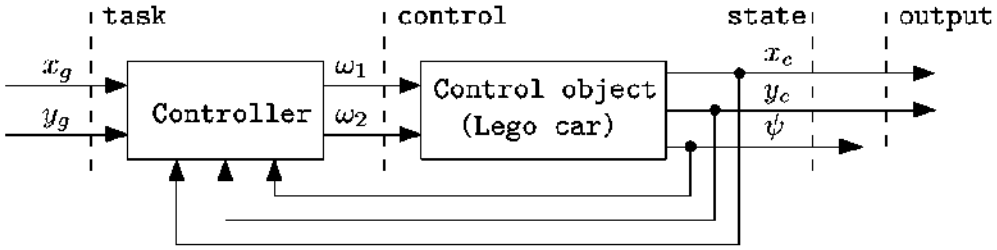


Figure 3. Structure of the control system.

x_g, y_g — coordinates of goal point;
ω_1, ω_2 — angular velocities of robot's wheels;
x_c, y_c, ψ coordinates and rotation angle of the robot.

Mathematical model of the robot

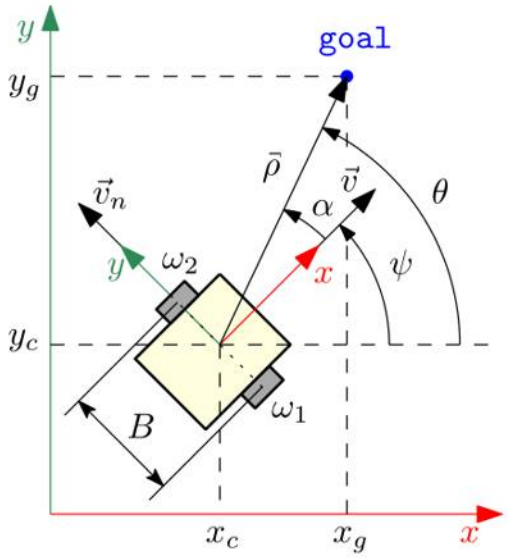


Figure 4. Useful drawing.

Kinematic model:

$$\begin{cases} \dot{x}_c = |\vec{v}| \cos\psi \\ \dot{y}_c = |\vec{v}| \sin\psi \\ \dot{\psi} = \omega \end{cases} \quad (1)$$

where

$$|\vec{v}| = R \cdot \frac{\omega_1 + \omega_2}{2}, \quad (2)$$

$$\omega = \frac{R}{B} \cdot (\omega_1 - \omega_2), \quad (3)$$

where R — wheel radius.

Mathematical model of the robot

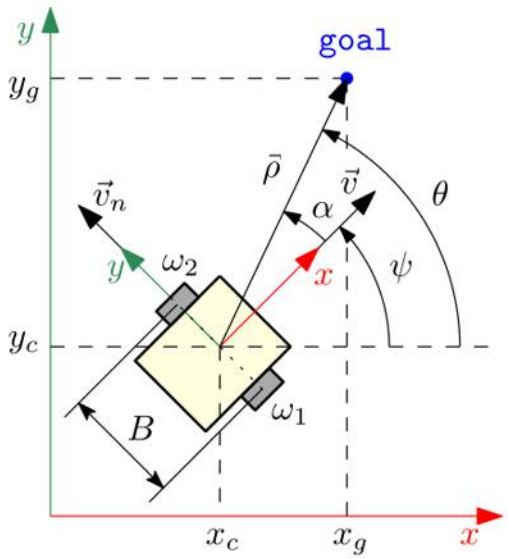


Figure 5. Useful drawing.

Some important variables:

$$\vec{\rho} = \begin{Bmatrix} x_g - x_c & y_g - y_c \end{Bmatrix}, \quad (4)$$

$$\theta = \arctan \frac{y_g - y_c}{x_g - x_c}, \quad (5)$$

$$\alpha = \theta - \psi, \quad (6)$$

$$|\vec{v}_n| = |\vec{v}|. \quad (7)$$

Mathematical model of the robot

$$|\vec{\rho}| - \sqrt{(x_g - x_c)^2 + (y_g - y_c)^2} \tag{8}$$

$$\frac{d|\vec{\rho}|}{dt} = \frac{1}{2\sqrt{(x_g - x_c)^2 + (y_g - y_c)^2}} \cdot \left((x_g - x_c)^2 + (y_g - y_c)^2\right)' =$$

$$= \frac{1}{2|\vec{\rho}|}(-2\dot{x}_c(x_g - x_c) - 2\dot{y}_c(y_g - y_c)) =$$

$$= -\frac{1}{|\vec{\rho}|} \cdot \{\dot{x}_c \;\; \dot{y}_c\} \cdot \{x_g - x_c \;\; y_g - y_c\} = -\frac{1}{|\vec{\rho}|} \cdot \vec{v} \cdot \vec{\rho} = -|\vec{v}| \cos\alpha \tag{9}$$

$$\dot{\alpha} = \dot{\theta} - \dot{\psi} \tag{10}$$

Mathematical model of the robot

$$\dot{\theta} = \left(\arctan\frac{y_g - y_c}{x_g - x_c}\right)' = \frac{1}{1 + \left(\dfrac{y_g - y_c}{x_g - x_c}\right)^2} \cdot \left(\frac{y_g - y_c}{x_g - x_c}\right)' =$$

$$= \frac{(x_g - x_c)^2}{(x_g - x_c)^2 + (y_g - y_c)^2} \cdot \frac{-\dot{y}_c(x_g - x_c) + \dot{x}_c(y_g - y_c)}{(x_g - x_c)^2} =$$

$$= \frac{\{-\dot{y}_c \;\; \dot{x}_c\} \cdot \{x_g - x_c \;\; y_g - y_c\}}{|\vec{\rho}|^2} = \frac{\vec{v}_n \cdot \vec{\rho}}{|\vec{\rho}|^2} =$$

$$= \frac{|\vec{v}_n|\cos(90° - \alpha)}{|\vec{\rho}|} = \frac{|\vec{v}|\sin\alpha}{|\vec{\rho}|} \tag{11}$$

$$\dot{\alpha} = \frac{|\vec{v}|\sin\alpha}{|\vec{\rho}|} - \omega \tag{12}$$

Mathematical model of the robot

Robot's mathematical model:

$$\begin{cases} \dfrac{d|\vec{\rho}|}{dt} = -|\vec{v}|\cos\alpha \\ \dfrac{d\alpha}{dt} = \dfrac{|\vec{v}|\sin\alpha}{|\vec{\rho}|} - \omega \end{cases} \quad \text{or} \quad \dot{x} = f(x), \text{ where } x = \begin{bmatrix} |\vec{\rho}| \\ \alpha \end{bmatrix} \tag{13}$$

Let's use for it this control law:

$$\begin{cases} |\vec{v}| = v_{max} \cdot \tanh|\vec{\rho}| \cdot \cos\alpha \\ \omega = K_\omega \alpha + v_{max} \cdot \dfrac{\tanh|\vec{\rho}|}{|\vec{\rho}|} \cdot \sin\alpha \cdot \cos\alpha \end{cases} \tag{14}$$

where v_{max} and K_ω are constant positive coefficients.

Mathematical model of the robot

Some theoretical information:

- Stability is an ability of a controlled system to run to particular state and stay in it.
- For checking system for stability Lyapunov functions are used.
- If time derivative of Lyapunov functions for considered system is always negative, the system is stable.

Mathematical model of the robot

Possible Lyapunov function for our system:

$$V(x) = \frac{1}{2}|\vec{\rho}|^2 + \frac{1}{2}\alpha^2 \tag{15}$$

Its derivative:

$$\frac{dV}{dt} = \frac{d|\vec{\rho}|}{dt} \cdot |\vec{\rho}| + \frac{d\alpha}{dt} \cdot \alpha = -|\vec{v}||\vec{\rho}| \cos\alpha + \alpha \left(\frac{|\vec{v}| \sin\alpha}{|\vec{\rho}|} - \omega \right) \tag{16}$$

or after using equations (14) for control law:

$$\frac{dV}{dt} = -v_{max} \cdot |\vec{\rho}| \cdot \tanh\ \vec{\rho} \cdot \cos^2\alpha - K_\omega \alpha^2 < 0. \tag{17}$$

Due to $\dot{V}$ is always negative the system is stable.

Mathematical model of the robot

Note that:

- angular speeds of robot's wheels can be found using these formulas:

$$\omega_1 = \frac{1}{R} \cdot (2|\vec{v}| + B\omega), \qquad \omega_2 = \frac{1}{R} \cdot (2|\vec{v}| - B\omega). \tag{18}$$

- in the steady state angular speeds of robot's motors are proportional to voltages which are applied to them; so we will make the latters are being proportional to values obtained from equations (18).

Sources for pictures

- slide 2:
  - https://en.wikipedia.org/wiki/Car
  - https://www.parallax.com/product/boe-bot-robot
  - http://www.makeblock.com/mecanum-wheel-robot-kit

ITMO UNIVERSITY
Faculty of Control systems and robotics

- per. Grivtsova, 14,
  Saint Petersburg, Russia, 190000

  +7 (812) 595-41-28
  csi.ifmo.ru/en/