ITMO UNIVERSITY

Learning Book

International summer school
of Control Systems and Robotics

Saint Petersburg
2017

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ

УНИВЕРСИТЕТ ИТМО

**О.И. Борисов, А.А. Ведяков, Д.Н. Герасимов,
В.С. Громов, С.А. Колюбин, А.Ю. Краснов,
А.А. Пыркин, С.А. Чепинский, С.В. Шаветов,
А.А Жиленков**

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

УЧЕБНОЕ ПОСОБИЕ

РЕКОМЕНДОВАНО К ИСПОЛЬЗОВАНИЮ В УНИВЕРСИТЕТЕ ИТМО
по направлению подготовки 15.04.06, 27.04.03, 27.04.04
в качестве учебного пособия для реализации основных профессиональных
образовательных программ высшего образования магистратуры

УНИВЕРСИТЕТ ИТМО

**Санкт-Петербург
2017**

О.И. Борисов, А.А. Ведяков, Д.Н. Герасимов, В.С. Громов, Н.А. Дударенко, К.А. Зименко, А.А. Капитонов, С.А. Колюбин, А.Ю. Краснов, А.А. Пыркин, С.В. Шаветов, А.А. Жиленков, International summer school of Control Systems and Robotics. Learning book / Международная летняя школа по системам управления робототехнике. Учебное пособие – СПб: Университет ИТМО, 2017. – 330 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 specializations 27.04.03 System Analysis and Control, 27.04.04 Control in Technical Systems and 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 направлениям подготовки 27.04.03 Системный анализ и управление, 27.04.04 Управление в технических системах и 15.04.06 Мехатроника и робототехника.

Table of Contents

Basics of the control theory

Let start course of the modern theory of control systems from the basics of classic control theory. All physical processes are described by differential equations. Let's consider typical simple structural automatic control scheme, that is shown in the Fig. 1:

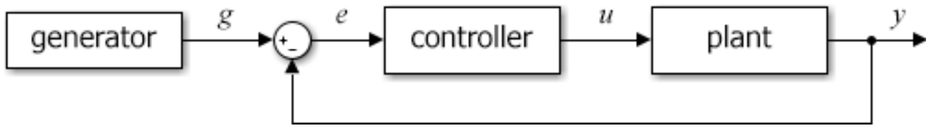


Fig. 1 – Structural scheme of automatic control system.

The Plant is the Control Object and it's a physical device, for example DC motor, electrical circuit, combustion engine, etc. Plant's behavior is described by some differential equations. Task of the Controller according to Error signal $e(t) = g(t) - y(t)$, which is equal to difference between reference signal $g(t)$ from generator and output $y(t)$ of the Plant, generate control signal $u(t)$ leading Plant to desired state and behavior. Output $y(t)$ in real object can be, for example, velocity or rotation angle of motor shaft.

Linear dynamic Plant (control system) can be described in two forms: Input-State-Output or Input-Output.

1. In a first form, typical equations system is:

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

where $x \in R^n$ is a state vector (it can consists of voltages in nodes of electrical circuit, for example); $u \in R^k$ is a system input; $y \in R^l$ is a system output, A – matrix of coefficients, called «state matrix» and described current state $x(t)$, dimension is $n \times n$; B – matrix of coefficients, called «control matrix» and described control $u(t)$ to each state, dimension is $n \times k$ (in case of system with single input is $n \times 1$); C – matrix of coefficients, called «output matrix» and described output $y(t)$ of the Plant, dimension is $l \times n$ (in case of system with single output is $1 \times n$). In continuous time, all variable x, y, u depend on time.

Let's introduce algebraic variable s and calculate *characteristic equation* of the Plant as:

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

where I – identity matrix, $s_i, i = \overline{1, n}$ are roots of the system, $a_j, j = \overline{0, n-1}$ are polynomial coefficients.

Characteristic equation corresponds to differential equations of the system and roots s_i are determined elementary behavior of the Plant.

2. Now, let's consider the second form: Input-Output.

In a previous case system is described by n-differential equations of first order. In this case we will describe Plant's behavior using one differential equation of n-order:

$$\frac{d^n y}{dt^n} + a_{n-1}\frac{d^{n-1} y}{dt^{n-1}} + \dots + a_1 \frac{dy}{dt} + a_0 y = b_m \frac{d^m u}{dt^m} + b_{m-1}\frac{d^{m-1} u}{dt^{m-1}} + \dots + b_1 \frac{du}{dt} + b_0.$$

In the case instead of three matrices A, B, C we have two sets of coefficients: $a_i, i = \overline{0, n-1}$ and $b_j, j = \overline{0, m}$. These sets are characterized system behavior.

Equation $m \le n$ is a condition of *physical feasibility*. What is a *derivative*? It's a limit of the ratio function increment to argument increment with the argument increment tends to zero:

$$f'(x) = \lim_{\Delta x \to 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}.$$

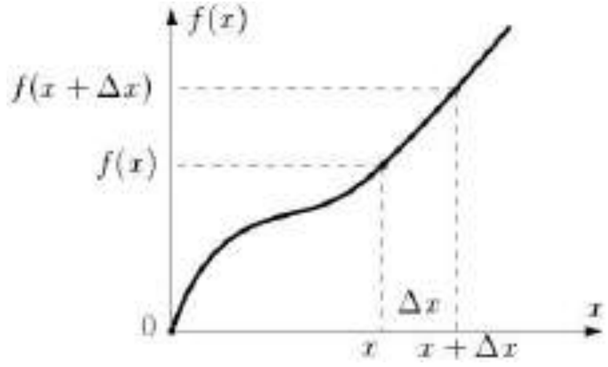


Fig. 2 – Derivative in the point x, mathematically it's correct.

But in case of physical systems axis x is an axis of time t, so, in a moment t we can't know value of $f(t+\Delta t)$, because the moment $(t+\Delta t)$ in the future. And due to this fact in a control theory we have limitation $m \leq n$.

Let's introduce algebraic variable s as a differentiation operator $s=\frac{d}{dt}$, then our n-order differential equation will take the form:

$$s^n y + a_{n-1}s^{n-1}y + \cdots + a_1 sy + a_0 y = b_m s^m u + b_{m-1}s^{m-1}u + \cdots + b_1 su + b_0 u.$$

Variables y and u put beyond the bracket:

$$y\underbrace{(s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0)}_{\text{characteristic equation}} = u(b_m s^m + b_{m-1}s^{m-1} + \cdots + b_1 s + b_0).$$

Let's divide this equation to u and to characteristic equation:

$$\frac{y}{u} = \frac{b_m s^m + b_{m-1}s^{m-1} + \cdots + b_1 s + b_0}{s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0} = W(s).$$

$W(s)$ is a *Transfer function* and it's a ratio output to input with zero initial conditions.

Roots of the Transfer function denominator are called *poles* and described system's *free* motion.

Roots of the Transfer function numerator are called *zeros* and described system's *forced* motion.

Example:

$$\frac{d^2y(t)}{dt^2} + 2\frac{dy(t)}{dt} + y(t) = \frac{du(t)}{dt} + u(t),$$

$W(s) = \frac{y(s)}{u(s)} - ?$

$W(s) = \frac{s+1}{s^2+2s+1}$,

$s+1=0 \Rightarrow s=-1$ – zero;

$s^2+2s+1=0 \Rightarrow s_{1,2}=-1$ – two poles (second order pole).

Let's consider conversion from input-state-output to input-output form:

$$W(s) = C(A-sI)^{-1}B.$$

In case of several inputs and/or several outputs we will obtain *Transfer matrix* $l \times k$ dimension, consists of several transfer functions $W_{i,j}(s)$ linking i-th output and j-th input. In a simple case with single input and single output (SISO-system) we will obtain only one transfer function.

Ok, we considered two forms describing dynamic systems. Now, let's consider behavior of output variable y. Typical transient is shown in the Fig. 3.

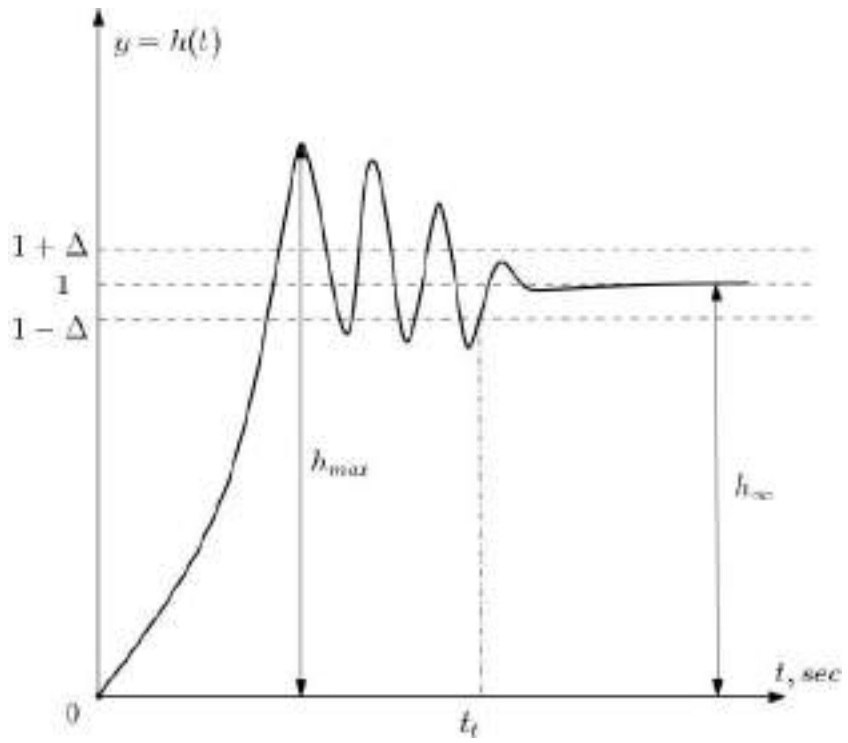


Fig. 3 – Typical transient in automatic control system.

t_t – transient time: $\forall t > t_t$: $|h(t) - 1| < \Delta, \Delta > 0$;

$h(t)$ – transient function;

$\Delta = 0.05$ (5% deviation from steady value h_∞);

in general case $h_\infty = \lim\limits_{t\to\infty} h(t)$, but in case of constant reference signal $g(t) = const = g$: $h_\infty = W(s)|_{s=0} \cdot g$.

We can calculate other important parameter – *overcontrol* (is noted as δ):

$$\delta = \left|\frac{h_{max} - h_\infty}{h_\infty}\right| \cdot 100\%.$$

For typical systems $\delta = 0 \cdots 30\%$.

Ok, now let's try to create a model of a simple abstract Plant.

Example:

Let $A = \begin{bmatrix} 1 & -1 \\ 2 & 0 \end{bmatrix}$; $B = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$; $C = [3 \quad -2]$.

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx \end{cases} \Rightarrow \begin{cases} \dot{x}_1 = x_1 - x_2 + u \\ \dot{x}_2 = 2x_1 + 2u \\ y = 3x_1 - 2x_2 \end{cases}.$$

For modelling we will use integrators, because due to condition $m \leq n$ we can't use differentiators:

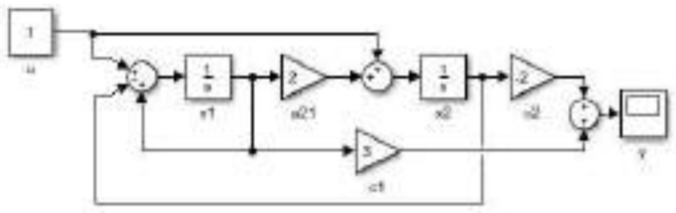


Fig. 4 – Scheme of system modelling.

Let's find out a transfer function:

$W(s) = C(A - sI)^{-1}B;$

$A - sI = \begin{bmatrix} 1 & -1 \\ 2 & 0 \end{bmatrix} - \begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix} = \begin{bmatrix} 1-s & 1 \\ -2 & 1-s \end{bmatrix} = A^*;$

$A^{*-1} = \frac{1}{\det A^*}(\mathrm{adj}A^*)^{\mathrm{T}};$

$\mathrm{adj}A^*$ – *adjoint* (or allied or interconnected) matrix;

elements of adjoint matrix are *algebraic complements* $a_{ij}^* = (-1)^{i+j} \cdot M_{ij}$,

where i – row number, j – column number, M_{ij} – minors of A^*.

$\det A^* = (1-s)(-s) - (2)(-1) = -s + s^2 + 2 = s^2 - s + 2;$

$A^{*-1} = \frac{1}{s^2-s+2}\begin{bmatrix} -s & -2 \\ 1 & 1-s \end{bmatrix}^{\mathrm{T}} = \frac{1}{s^2-s+2}\begin{bmatrix} -s & 1 \\ -2 & 1-s \end{bmatrix};$

$CA^{*-1} = [3 \quad -2] \cdot \frac{1}{s^2-s+2} \cdot \begin{bmatrix} -s & 1 \\ -2 & 1-s \end{bmatrix} = \frac{1}{s^2-s+2} \cdot [-3s+4 \quad 3-2+2s];$

$W(s) = \frac{1}{s^2-s+2} \cdot [-3s+4 \quad 1+2s] \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \frac{1}{s^2-s+2}(-3s+4+2+4s) = \frac{s+6}{s^2-s+2}.$

Now, let's try to create a model from the transfer function:

$W(s) = \frac{Y(s)}{U(s)} = \frac{s+6}{s^2-s+2}, s = \frac{d}{dt};$

$s^2y - sy + 2y = su + 6u;$

Let's divide equation to s a power of differential equation order:

$s^2y - sy + 2y = su + 6u \mid : s^2;$

$y - \frac{1}{s}y + 2\frac{1}{s^2}y = \frac{1}{s}u + 6\frac{1}{s^2}u;$

Now, let's leave only y on the left:

$y = \frac{1}{s}u + \frac{1}{s}y + 6\frac{1}{s^2}u - 2\frac{1}{s^2}y;$

Rewrite equation:

$y = \frac{1}{s}(u + y) + \frac{1}{s^2}(6u - 2y);$

And create a model:

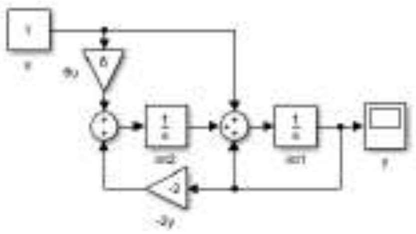


Fig. 5 – Scheme of system modelling.

Structural transformations

Let's consider typical transformation for structural schemes, how to simplify some of them.

1. Consecutively k-connected elements.

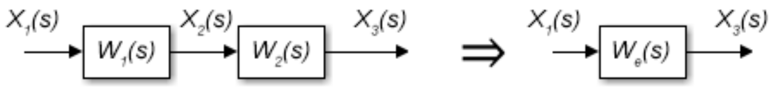


Fig. 6 – Scheme of consecutively connected elements.

$W_e(s) = \frac{X_3(s)}{X_1(s)}$;

from the scheme is visible, that:

$\begin{cases} X_2 = W_1 X_1, \\ X_3 = W_2 X_2 = W_2 W_1 X_1 \end{cases} \Rightarrow W_e(s) = \frac{X_3(s)}{X_1(s)} = W_2(s) W_1(s);$

$W_e(s) = \prod_{i=1}^{k} W_i\,(s).$

2. Parallel k-connected elements.

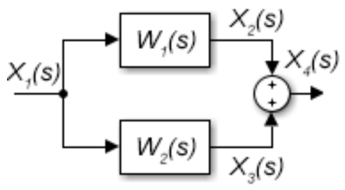


Fig. 7 – Scheme of parallel connected elements.

Using the similar logic obtain: $W_e(s) = \sum_{i=1}^{k} W_i\,(s)$.

3. Elements with a feedback.

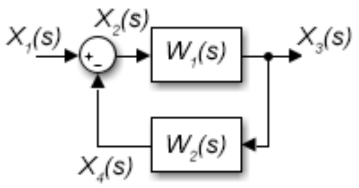


Fig. 8 – Scheme of elements with a feedback.

$\begin{cases} X_2 = X_1 - X_4, \\ X_3 = W_1 X_2, \\ X_4 = W_2 X_3, \end{cases} \Rightarrow \begin{cases} X_3 = W_1(X_1 - X_4), \\ X_3 = W_1 X_1 - W_1 W_2 X_3, \\ (1 + W_1 W_2) X_3 = W_1 X_1. \end{cases}$

$W_e(s) = \frac{X_3}{X_1} = \frac{W_1}{1 + W_1 W_2}.$

Stability

Now, let's consider one of the most important property of dynamic systems: «stability». What is stability? *Stability* is the system ability to return to initial position after stopping action to system external disturbances. For example, let's see to the picture:

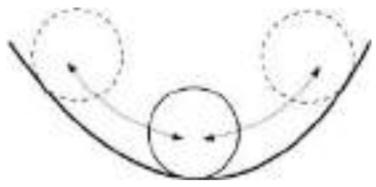


Fig. 9 – Stable system.

Is this system stable? Yes! If we move a ball to left or to right side and leave it there, the ball returns to stable bottom position.

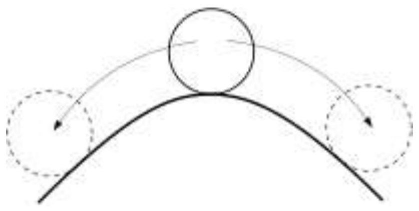


Fig. 10 – Unstable system.

And, is this system stable? No, system is unstable, but the current position is the equilibrium. In our life, this picture is like a Segway, two-wheeled balancing pendulum platform.

Fig. 11 – Segway.

In control theory identify several kinds of stability. We consider three of them. The first and the weakest kind of stability is Lyapunov stability.

1. Lyapunov stability

Lyapunov stability guarantees bounded of all trajectories, but not guarantees convergence to some steady value. Geometrically it can be shown in Fig. 12:

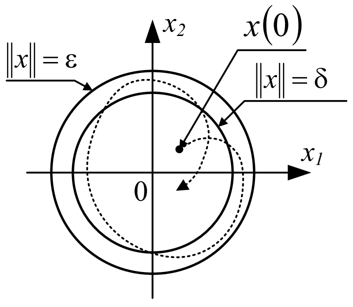


Fig. 12 – Lyapunov stability geometrical interpretation.

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: $\|x(t)\| \leq \varepsilon$.

Let's consider *Root stability criterion* for the system:

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

Characteristic polynomial of matrix A is $\det(A - sI) = s^n + a_{n-1}s^{n-1} + \dots + a_1 s + a_0 = 0$, where $s_i, i = \overline{1,n}$ – are roots of polynomial.

So, if all roots have negative real parts $\operatorname{Re} s_i < 0, i = \overline{1,n}$, then the system is stable. On the complex plane, they are:

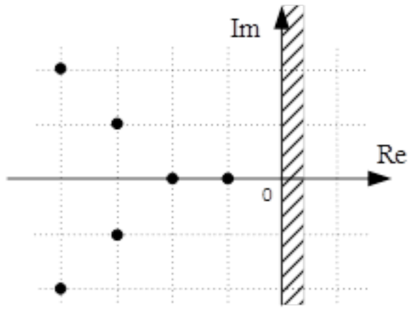


Fig. 13 – Roots distribution in a stable system.

where Im – imaginary axis, Re – real axis. Imaginary axis is a *stability border*.

If one or more than one root is more than zero system is *unstable*.

As sub kinds of Lyapunov stability we can distinguish two stability borders.

1.1. Stability border of neutral type.

Root stability criterion claims that the dynamic 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:

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

1.2. Stability border of oscillatory type.

Root stability claims that the dynamic 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.

$s_{1,2} = \pm j\omega, \omega > 0$ Re$s_i < 0, i = \overline{3, n}$, where j – imaginary unit.

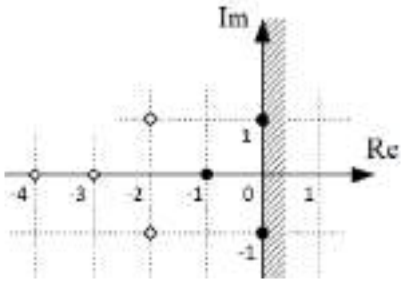


Fig. 14 – Stability border of oscillatory type.

So, all these sub kinds are Lyapunov stability. The next kind of stability is asymptotic stability.

2. Asymptotic stability.

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

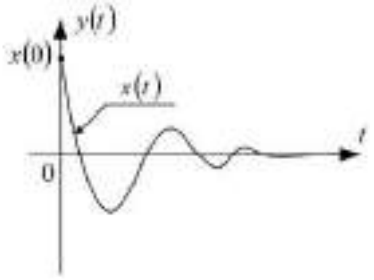


Fig. 15 – Transient in an asymptotic stable system.

Stronger than an asymptotic stability is an exponential stability.

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 numbers $\beta < 0$ and $\rho \geq 1$ that for any time $\forall t \geq 0$ the inequality is satisfied: $\|x(t)\| \leq \rho e^{\beta t} \cdot \|x(0)\|$.

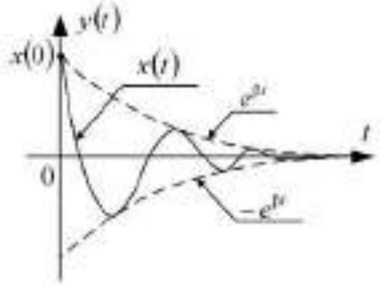


Fig. 16 – Transient in an exponentially stable system.

Constant β is the convergence degree and characterizes convergence velocity to equilibrium. From exponential stability implies asymptotic stability, and from asymptotic stability implies Lyapunov stability.

Ok, system can be stable or unstable. But in both these cases control purpose must be fulfilled. There are many different methods and approaches for developing control laws (algorithms) exists. Now we considered one of the most popular method «modal control».

Modal control

Behavior of the system is uniquely determined by the state matrix eigenvalues. To make system behavior in a certain way it is necessary to develop controller which delivers desired eigenvalues to a given state matrix A. We should construct reference (modal) model that has desired quality indicators.

The Plant is given by the system of differential equations:

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

Modal model is an autonomous dynamic system of the form:

$$\begin{cases} \dot{z}(t) = \Gamma z(t), \\ \eta(t) = \mathrm{H}z(t), \end{cases}$$

where $z(t) \in R^n$ – n-dimension state vector, $\eta(t) \in R^l$ – l-dimension output vector; Γ – square state matrix of dimension $n \times n$; H – output matrix $l \times n$. Matrices (Γ, H) are completely observable (observable matrix $\mathrm{No} = [\mathrm{H} \quad \mathrm{H}\Gamma \quad \mathrm{H}\Gamma^2 \quad \cdots \quad \mathrm{H}\Gamma^{n-1}]^{\mathrm{T}}$ should has the full rank). State matrix Γ characterized by eigenvalues $\lambda_i, i = \overline{1,n}$, providing a predetermined dynamic performance quality of the system. Eigenvalues of A matrix are not equal to eigenvalues of Γ matrix.

It is necessary to synthesize a control algorithm that generates control signals $u(t)$ for dynamic Plant with the quality indicators defined by the eigenvalues of matrix Γ. Let's choose proportional control:

$$u(t) = -\mathrm{K}x(t),$$

where K – matrix of linear stationary feedbacks.

Substituting control signal $u(t)$ in a vector-matrix description of the Plant we obtain a closed feedback system:

$$\begin{cases} \dot{x} = \mathrm{F}x, \\ y = Cx, \end{cases}$$

where $\mathrm{F} = A - B\mathrm{K}$ – matrix of the closed system.

To matrix K provides quality indicators for given dynamic system like in a reference model, it's necessary the condition of similarity is satisfied:

$$x(t) = \mathrm{M}z(t) \;\Rightarrow\; z(t) = \mathrm{M}^{-1}x(t)\,, t \geq 0,$$

where M – *coordinate transformation* or *similarity matrix*.

Output of the reference model is a control signal for the given model. Using this relation obtain the control law:

$$u(t) = -\mathrm{H}z(t) = -\mathrm{H}\mathrm{M}^{-1}x(t).$$

Let's introduce notation:

$$\mathrm{K} = \mathrm{H}\mathrm{M}^{-1} \;\Leftrightarrow\; \mathrm{H} = \mathrm{K}\mathrm{M}.$$

Substituting obtained expressions to Sylvester type matrix equation obtain:

$$\mathrm{M}\Gamma - A\mathrm{M} = -B\mathrm{K}\mathrm{M}.$$

With the notation $\mathrm{F} = A - B\mathrm{K}$ matrix equation leads to a similarity condition Γ and F:

$$M\Gamma = FM,$$

therefore matrix F has eigenvalues Γ.

Example:

$$A = \begin{bmatrix} 7 & 3 & 14 \\ 6 & 5 & -8 \\ 4 & -1 & -7 \end{bmatrix}, B = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, C = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}.$$

Required to ensure the transition time in a closed system $t_t = 1.7$ seconds.

```
A = [7 3 14; 6 5 -8; 4 -1 -7]; % MATLAB code
B = [0; 0; 1];
C = [1 0 0];
```

1. Checking system stability:

$$\det(A - sI) = s^3 - 5s^2 - 131s + 635 \implies \begin{cases} s_1 = -11.3921, \\ s_2 = 11.5776 > 0, \\ s_3 = 4.8145 > 0, \end{cases} \implies \text{system is unstable.}$$

```
roots(poly(A)); % MATLAB code or
eig(A);
```

2. Checking system for complete controllability (matrix $Nc = [B \quad AB \quad A^2B \quad \cdots \quad A^{n-1}B]$ should has the full rank):

$$\text{rank } N_c = \text{rank} \begin{bmatrix} 0 & 14 & -24 \\ 0 & -8 & 100 \\ 1 & -7 & 113 \end{bmatrix} = 3 = n \Rightarrow \text{system is completely controllable.}$$

```
rank(ctrb(A, B)); % MATLAB code
```

3. Form reference model:

To form the reference model, we should find a desired characteristic polynomial in accordance with a normalized transient time t_t^* for a n-order system. Normalized time can be calculated from transients of Butterworth (overcontrol not more than 15%) $Db(s) = \prod_{i=1}^{n} \left(s - \omega e^{j\left(\frac{\pi}{2} + \frac{2i-1}{2n}\pi\right)}\right)$ or Newton (overcontrol 0%) $Dn(s) = (s + \omega)^n$ polynomials:

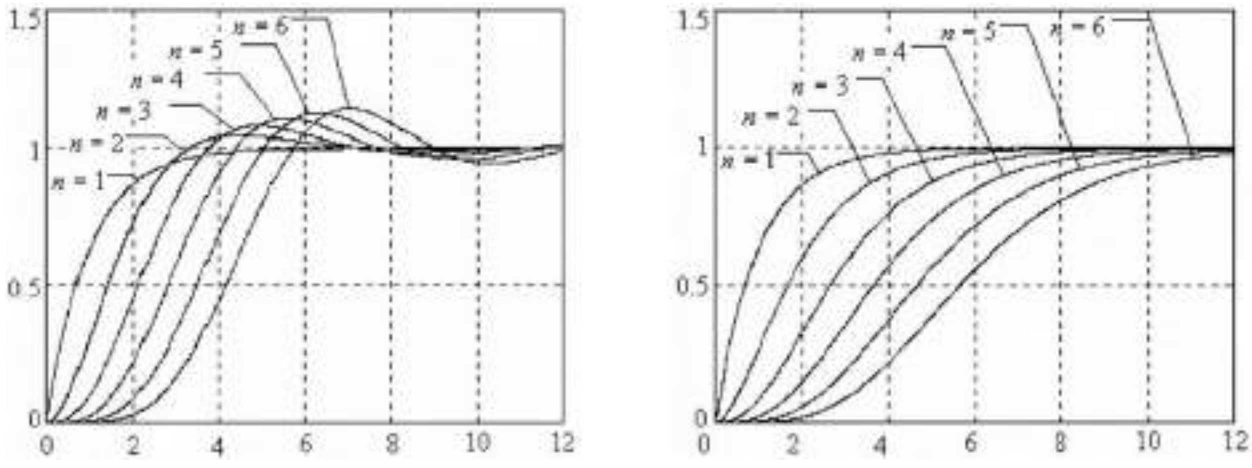


Fig. 17 – From left to right Butterworth and Newton polynomials transients accordingly.

In case $n = 3$: $t_t^* = 6.2$ seconds, $D_d(s) = s^3 + 3\omega s^2 + 3\omega^2 s + \omega^3$ – the desired characteristic polynomial (third-order Newton polynomial).

$\omega = \frac{t_t^*}{t_t} = \frac{6.2}{1.7} = 3.65, D_d(s) = s^3 + 10.94s^2 + 39.9s + 48.51.$

Now, write out the matrix Γ of reference model in a *canonical controllable form* with help the desired polynomial:

$$\Gamma = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -48.51 & -39.9 & -10.94 \end{bmatrix}, \mathrm{H} = [1 \quad 0 \quad 0].$$

```
G = [0 1 0; 0 0 1; -48.51 -39.9 -10.94]; % MATLAB code
H = [1 0 0];
```

4. Finding matrix transformation M:

The solution of the Sylvester matrix equation with respect to the matrix M:

$$\mathrm{M} = \begin{bmatrix} -0.1497 & -0.0084 & -0.0020 \\ 0.2111 & 0.0215 & 0.0028 \\ 0.0365 & -0.0054 & 0.0014 \end{bmatrix}.$$

```
M = sylv(-A,G,-B*H); % MATLAB code
```

5. Calculation of matrix K:

$$\mathrm{K} = \mathrm{HM}^{-1} = [36.2969 \quad 18.2453 \quad 15.94].$$

```
K = H * inv(M); % MATLAB code
```

6. Checking calculations:

$$\mathrm{F} = A - B\mathrm{K} = \begin{bmatrix} 7 & 3 & 14 \\ 6 & 5 & -8 \\ -32.2959 & -19.2453 & -22.94 \end{bmatrix},$$

$\det(\mathrm{F} - sI) = s^3 + 10.94s^2 + 39.9s + 48.51 = D_d(s)$ – characteristic polynomial coincides with the reference, hence controller coefficients found correctly.

```
F = A – B * K; % MATLAB code
poly(F)
```

7. Calculation of the direct linking coefficient:

$$\mathrm{K}_g = -(\mathrm{C}(\mathrm{A} - \mathrm{BK})^{-1}\mathrm{B})^{-1} = -0.5161.$$

```
Kg = -inv(C * inv(A – B * K) * B); % MATLAB code
```

8. Form control signal $u(t) = \mathrm{K}_g g(t) - \mathrm{K}x(t)$.

Discretizing

The last point of this lecture is how to use this controller in a real life. Let's consider approach to discretize continuous system and try to write simple program code. In continuous time, the Plant is described as follows:

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

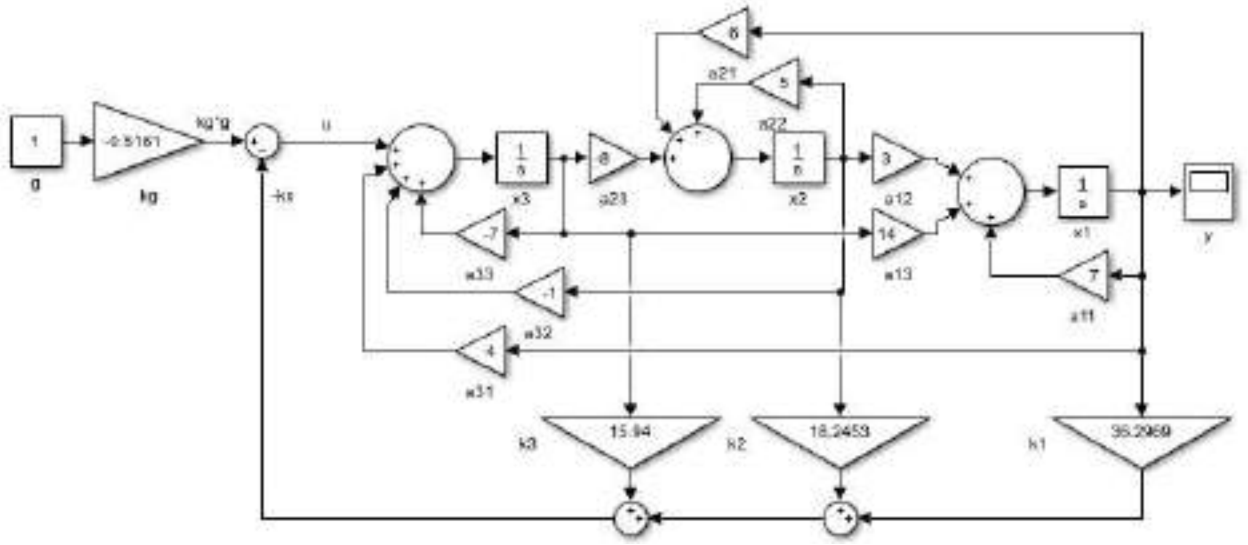


Fig. 18 – Scheme of closed system with a modal controller.

Digital electronic microcontrollers work in a discrete time, so:

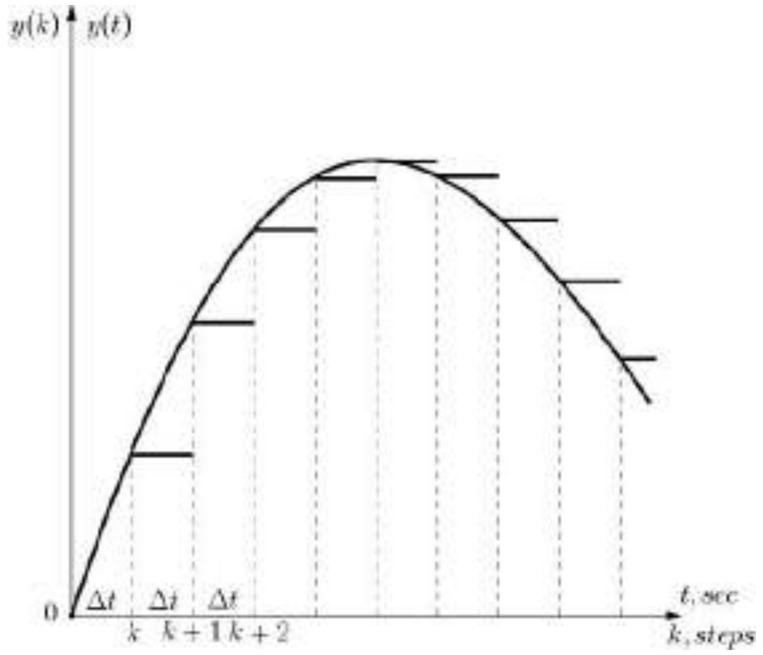


Fig. 19 – Continuous and discrete processes.

$T = \Delta t$ – discrete interval.

$$\begin{cases} x(k+1) = Ad \cdot x(k) + Bd \cdot u(k), \\ y(k) = Cd \cdot x(k). \end{cases}$$

Discrete matrix A we can find as a matrix exponent: $Ad = e^{AT}$, discrete matrix B we can find using formula: $Bd = A^{-1}(e^{AT} - I)B|_{\exists A^{-1}}$ and $Cd = C$.

```
T = 0.1; % MATLAB code
Ad = expm(A * T);
Bd = inv(A) * (Ad - eye(n)) * B;
Cd = C;
```

But, to avoid division by zero in A^{-1} we can decompose matrix exponent to series with k-members:

$e^{AT} \approx \sum_{i=0}^{k} \frac{A^i T^i}{i!} = Ad,$

$Bd \approx \left(\sum_{i=1}^{k} \frac{A^{i-1} T^i}{i!}\right) \cdot B.$

<u>Example:</u>

$A = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix}, B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, C = [1 \quad 0], T = 0.01, u = 1, Ad, Bd, Cd-?$

```
A = [0 1; -2 -3]; % MATLAB code
B = [0; 1];
C = [1 0];
T = 0.01;
Ad = 0;
Bd = 0;
k = 10;
for i = 0:1:k
        Ad = Ad + (A^i * T^i / factorial(i));
        if(i > 0)
                Bd = Bd + (A^(i – 1) * T^i / factorial(i)) * B;
        end;
end;
Cd = C;
```

$Ad = \begin{bmatrix} 0.9999 & 0.0099 \\ -0.0197 & 0.9703 \end{bmatrix}, Bd = \begin{bmatrix} 0 \\ 0.0099 \end{bmatrix}, Cd = [1 \quad 0].$

Ok, integrator is a memory element which saves previous value. Now, let’s construct transient in a discrete view:

```
x1 = 0; % MATLAB code
x2 = 0; %initial conditions
u = 1;
for m = 0:1:100000
        x1dot = Ad(1,1) * x1 + Ad(1,2) * x2 + Bd(1) * u;
        x2dot = Ad(2,1) * x2 + Ad(2,2) * x2 + Bd(2) * u;
        y(m) = Cd(1) * x1 + Cd(2) * x2;
        k(m) = m / T;
        x1 = x1dot;
        x2 = x2dot;
end;
plot(k,y);
```

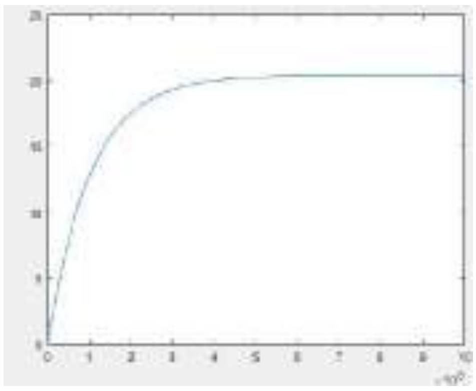


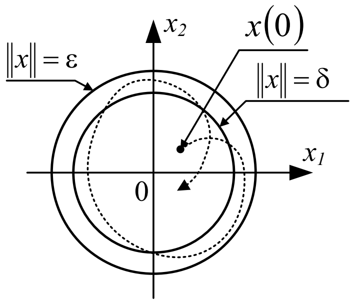
Fig. 20 – Transient in a discrete time.

Stability types and Lyapunov equations

Stability types

What is stability? Stability is the system ability to return to initial position after stopping action to system external disturbances. The first and the weakest kind of stability is Lyapunov stability.

1. Lyapunov stability guarantees bounded of all trajectories, but not guarantees convergence to some value. Geometrically it can be shown as the picture:



Pic. 1 – Lyapunov stability geometrical interpretation.

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.

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 $\|x(0)\| \le \delta(\varepsilon)$ for any time $\forall t \ge 0$ following inequality is satisfied: $\|x(t)\| \le \varepsilon$.

Roots stability criterion.

Let's consider the next continuous system:

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

Characteristic polynomial of matrix F is $\det[F - sI] = s^n + \alpha_{n-1}s^{n-1} + \cdots + \alpha_1 s + \alpha_0 = 0$, where s – is a differential operator, I – is an identity matrix, and $s_i, i = \overline{1, n}$ – roots of our polynomial.

So, if all roots have negative real parts $Res_i < 0, i = \overline{1, n}$, then the system is stable. On the complex plane they are:

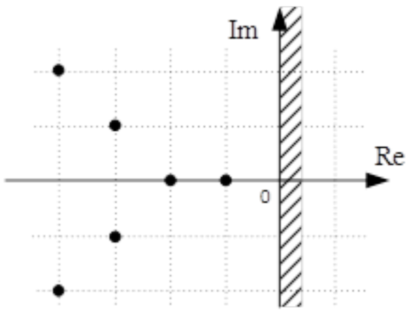


Fig. 2 – Roots distribution in a continuous stable system.

where Im – imaginary axis, Re – real axis. Imaginary axis is a *stability border*.

If one or more than one root is more than zero system is unstable.

Let consider this Root stability criterion in discrete case. In this case instead of function derivative we use the 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.

Let's consider the next discrete system:

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

Characteristic polynomial of matrix F_d is $\det[F_d - zI] = z^n + \alpha_{n-1} z^{n-1} + \dots + \alpha_1 z + \alpha_0 = 0$, where z – is a delay, I – is an identity matrix, and $z_i, i = \overline{1,n}$ – roots of our polynomial.

So, if all absolute values of roots less than one $|z_i| < 1, i = \overline{1,n}$, then the system is stable. On the complex plane they are:

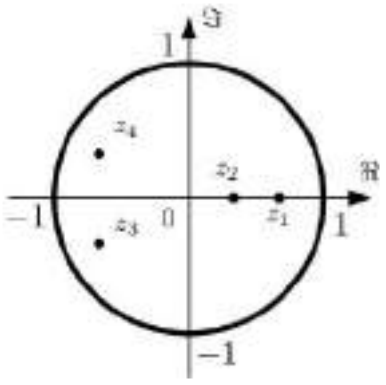


Fig. 3 – Roots distribution in a discrete stable system.

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.

Let's return to Lyapunov stability. As sub kinds of Lyapunov stability we can distinguish two stability borders.

The first one is the border of *neutral type*. Root stability criterion claims that the dynamic system is on the border of neutral type if one or two roots of system characteristic polynomial are equal to zero and rest roots have negative real parts.

In continuous case:

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

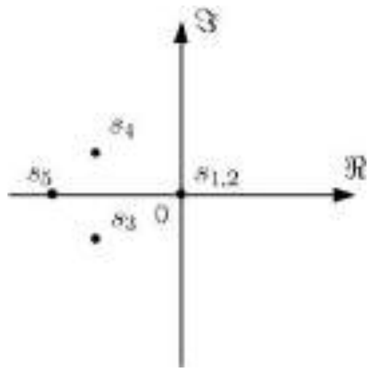


Fig. 4 – Continuous system is on the border of neutral type.

In discrete case, one or two roots are equal to one and rest roots are in the unit circle:

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

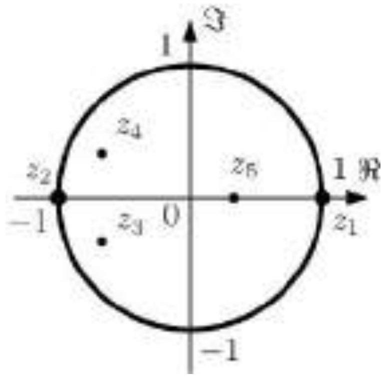


Fig. 5 – Discrete system is on the border of neutral type.

The second one is the border of *oscillatory type*. Root stability claims that the dynamic system is on the border of oscillatory type if the system characteristic polynomial has pair of purely imaginary roots and rest roots have negative real parts: $s_{1,2} = \pm j\omega, \omega > 0 \; Res_i < 0, i = \overline{3, n}$.

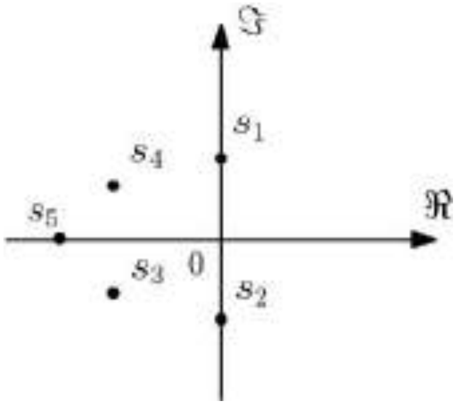


Fig. 6 – Continuous system is on the border of oscillatory type.

So, all these sub kinds are Lyapunov stability. The next kind of stability is asymptotic stability.

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

In discrete case $\lim_{m\to\infty} \|x(m)\| = 0$.

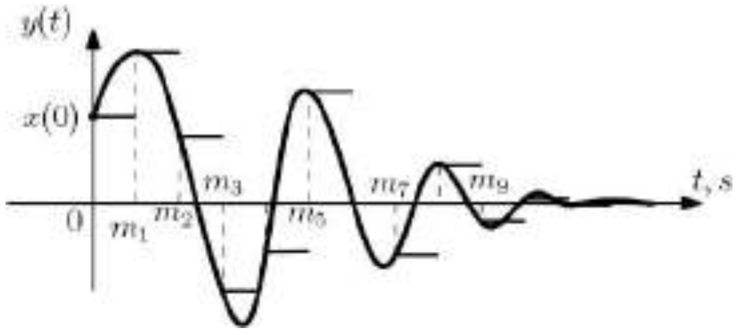


Fig. 7 – Asymptotically stable continuous and discrete processes.

Stronger than an asymptotic stability is exponential stability.

3. Exponential stability. The equilibrium $x = 0$ is exponential stable if for any motion trajectories $x(t)$ from the arbitrary initial conditions 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.

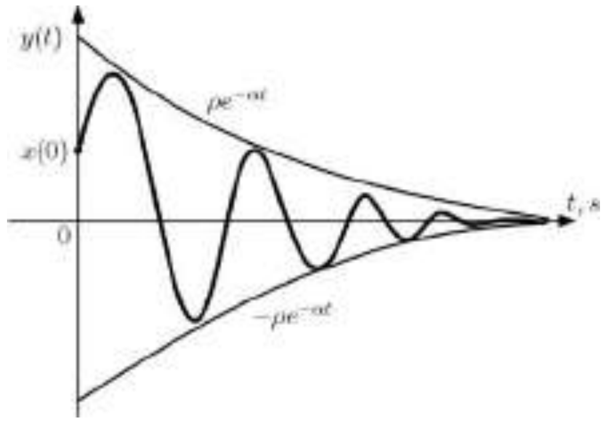


Fig. 8 – Exponentially stable continuous process.

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

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

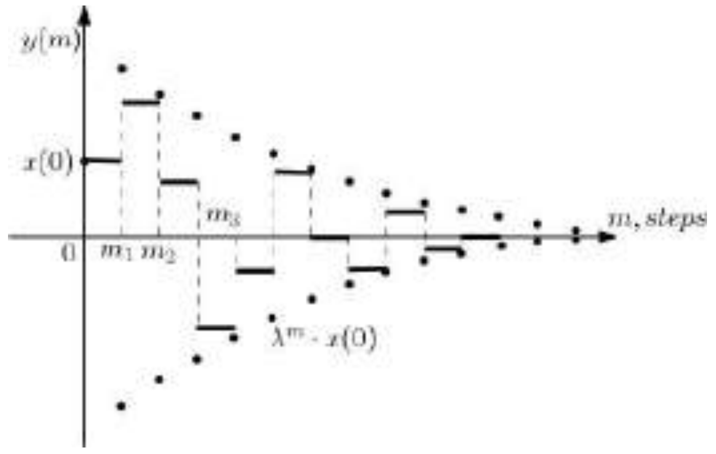


Fig. 9 – Exponentially stable discrete process.

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

From exponential stability goes asymptotic stability.

The next kind of stability and the strongest type is the *qualitative exponential stability*. We consider this type on the next lecture in details after considering Lyapunov equations, but now short brief about it.

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: $\|x(m) - \alpha^m x(0)\| \leq \rho((\alpha + r)^m - \alpha^m)\|x(0)\|, 0 \leq \alpha < 1 - r.$

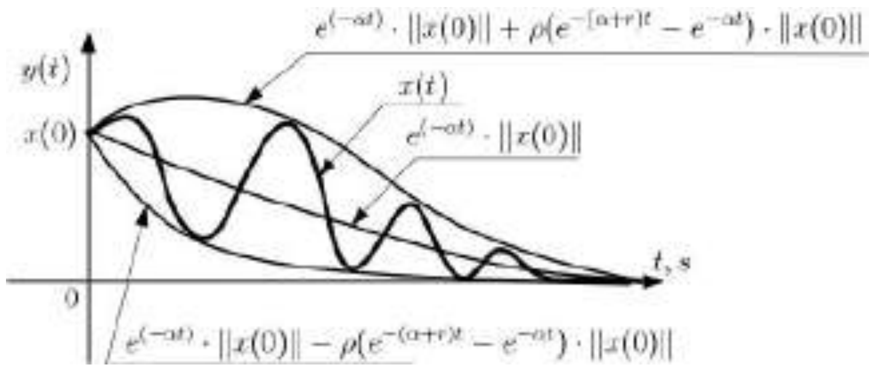


Fig. 10 – Qualitatively exponentially stable continuous process.

So, besides parameter α characterizes convergence to equilibrium in exponential stability, in qualitative exponential stability introduces parameter r characterizes trajectory average deviation.

Lyapunov equations

Now, let's consider Lyapunov functions for investigation stability of linear systems.

Lyapunov functions have the next 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.

The simplest and the most frequently used class of Lyapunov functions are quadratic forms: $V(x) = x^T P x$, P – is a positive definite symmetric square matrix of $n \times n$ dimension.

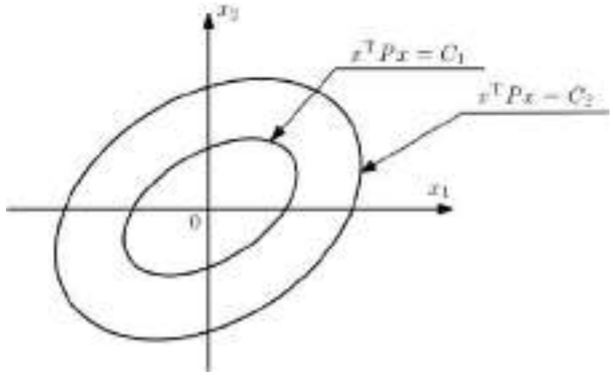


Fig. 11 – Lyapunov functions.

$C_2 < C_1$,

$P = I$ – identity matrix,

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

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

Lyapunov Theorem.

The equilibrium $x = 0$ is asymptotic 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: $\frac{dV(x(t))}{dt} < 0$.

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

Time derivative of Lyapunov function is equal to partial derivative of Lyapunov function with respect to x multiply to partial time derivative of x.

x – state vector with dimension of n, so:

$\frac{\partial V(x)}{\partial x} = \begin{bmatrix} \frac{\partial V(x)}{\partial x_1} & \dots & \frac{\partial V(x)}{\partial x_n} \end{bmatrix} = grad^T V(x)$.

Geometrically shown as follows:

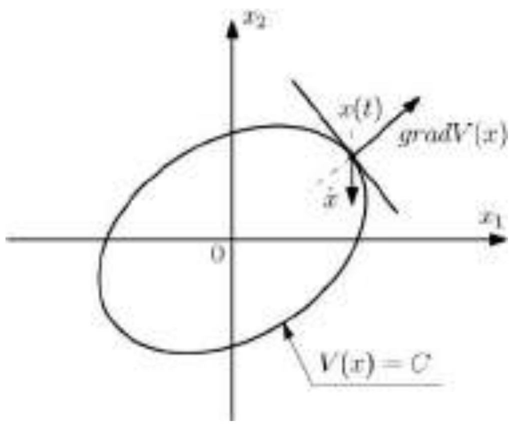


Fig. 12 – Derivative of Lyapunov function.

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

So, condition of Lyapunov theorem: $\frac{dV(x(t))}{dt} = grad^{\mathrm{T}}x \cdot \dot{x}$.

<u>Example:</u>

Consider following system: $\dot{x} = -x^3$. Let's choose Lyapunov function from the class of quadratic forms: $V(x) = x^2$.

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

This function is negative anytime, so, the system $\dot{x} = -x^3$ is asymptotically stable.

We can extend this result to the case of exponential stability. For this we should modify inequality from theorem as follows: $\dot{V}(x(t)) \leq -2\alpha V(x(t)), \alpha > 0$.

In case of qualitative exponential stability, we obtain this inequality:

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

Let's consider Rayleigh ratio and geometrical interpretation of Lyapunov functions:

$C_1^2\|x\|^2 \leq x^T P x \leq C_2^2\|x\|^2$, where $\dot{x} = Fx, x \in R^n, F - n \times n$, $V(x) = x^T P x$, P – is a positive definite symmetric square matrix of $n \times n$ dimension.

So, omitting intermediate calculations we get the inequality $\|x(t)\| \leq \frac{C_2}{C_1} e^{-\alpha t} \|x(0)\|, \rho = \frac{C_2}{C_1} \geq 1$.

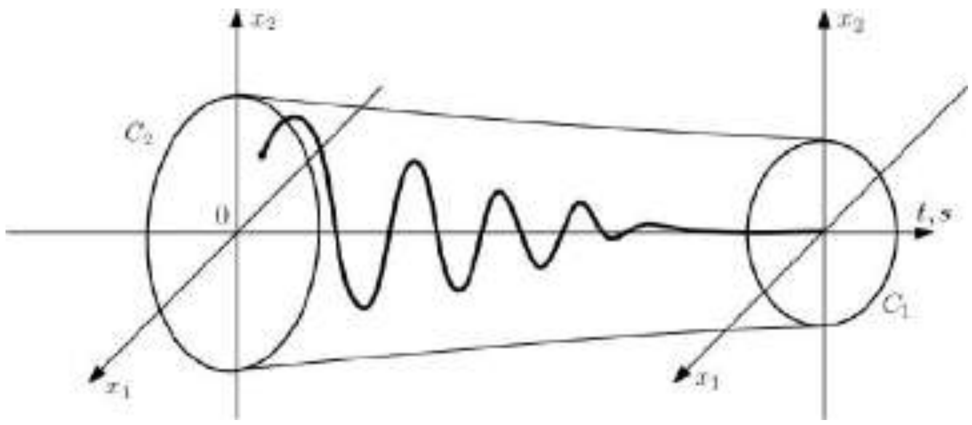


Fig. 12 – Geometrical interpretation of Rayleigh ratio.

From the Lyapunov theorem we can write Lyapunov equation omitting intermediate calculations:

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

where F – is a state matrix of closed system, P, Q – positive definite symmetric square matrices with dimensions are of the same of F.

Positive definite matrices are all eigenvalues of it more than zero.

For investigation system to asymptotic stable we should choose matrix Q, solve Lyapunov equation with respect to matrix P and check it for positive definition.

In case of exponential stability, we should to modify the Lyapunov equation as follows:

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

In case of qualitative exponential stability we obtain matrix equation of Riccati type:

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

Example. Investigation system to asymptotic stability:

$\dot{x} = Fx$,

$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & -2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$, is this system stable?

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

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

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

All eigenvalues of matrix $P > 0$ are more than zero, so system is asymptotically stable.

ITMO UNIVERSITY

Qualitative exponential stability for discrete and continuous linear systems

Ok, let's talk about *qualitative exponential stability*. This term is narrower than just exponential stability due to additional conditions. The conditions bounded changing values velocity of system state vector. Using additional conditions is possible to localize system process properties with some good characteristics: smaller oscillation and smoother processes. Despite these characteristics, developer can estimate *convergence velocity* and *stability margin* of the system.

Let's consider following linear discrete system:

$$x(m+1) = F\big(x(m)\big)\,. \tag{1}$$

So, equilibrium $x = 0$ of system (1) is *exponential stable*, if exists numbers $\rho > 0$, $\alpha > 0$, $d_x(\alpha) > 0$ that for all initial values of $\|x(0)\| \leq d_x(\alpha)$ for any number of discreetness interval $m > 0$ the following inequality is satisfied:

$$\|x(m)\| \leq \rho \cdot e^{-\alpha m} \cdot \|x(0)\|. \tag{2}$$

Let's introduce following notation: $\lambda = e^{-\alpha}$, where $0 < \lambda < 1$, so inequality (2) takes the form:

$$\|x(m)\| \leq \rho \cdot \lambda^m \cdot \|x(0)\|. \tag{3}$$

From these inequalities we can conclude, that all trajectories of exponential stable system are in «estimated tube» bounded by surfaces:

$$\|x(m)\|^2 = (\rho \cdot \lambda^m \cdot \|x(0)\|)^2. \tag{4}$$

those bounded by circles of radius $\rho \cdot \lambda^m \cdot \|x(0)\|$.

In case of quadratic norm (or Euclidean norm), the trajectories of exponential stable system are shown on this picture:

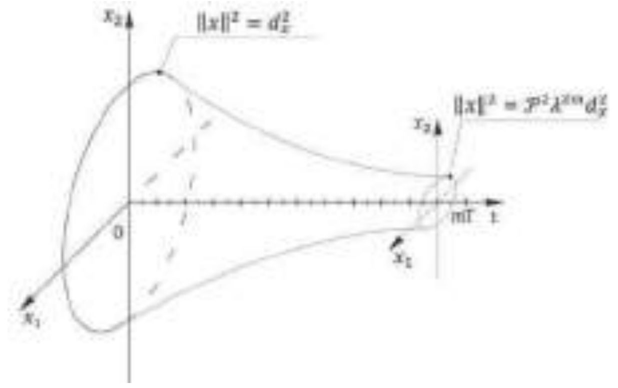


Fig. 1 – Estimated tube of exponential stable system.

Introduce local term of «*qualitative exponential stability*». Equilibrium $x = 0$ of system (1) 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:

$$\|x(m) - x(0)\| \leq \lambda_0 \rho \sum_{i=0}^{m-1} \lambda^i \|x(0)\| = \lambda_0 \rho \frac{1-\lambda^m}{1-\lambda} \|x(0)\| \tag{5}$$

Additional condition (5) constrains deviation of state vector current values from initial condition. The highest practical importance of qualitative exponential stability is under conditions: $\lambda_0 < 1$, because in this case inequality (5) is strongest, so processes in system have very qualitative parameters.

So, if inequalities (4) and (5) are valid, then all trajectories are bounded by surfaces (4) (center of these circles is at the origin) and:

$$\|x(m) - x(0)\|^2 = \left[\lambda_0 \rho \frac{1-\lambda^m}{1-\lambda} d_x\right]^2 \tag{6}$$

those bounded by circles of radiuses $\lambda_0 \rho \frac{1-\lambda^m}{1-\lambda} d_x$ with center of these circles is at the point $x(0)$.

«Estimated tube» cross-section of qualitative exponential stability system is shown on this picture:

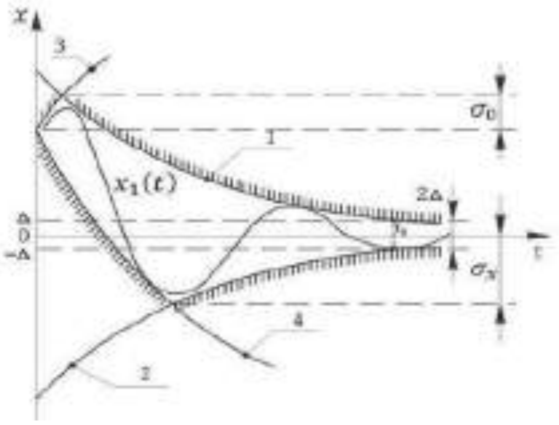


Fig. 2 – «Estimated tube» cross-section.

Estimated values of the first ejection and overcontrol can be derived from (4) and (6):

$$\sigma_{0x}^* = \frac{(\rho-1)\lambda_0}{\lambda+\lambda_0-1}. \tag{7}$$

$$\sigma_x^* = \frac{\lambda-\rho\lambda_0-1}{1-\lambda+\lambda_0}. \tag{8}$$

Ok, let's vector norm is determined by equation:

$$\|x\| = [\textstyle\sum_{i=1}^{n} |x_i|^v]^{\frac{1}{v}}, \tag{9}$$

where 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. So, if we have the system with two state vector components, then the surfaces of a constant level $\|x\|^v = 1$ ($x \in R^2$) are shown on the Fig. 3. Remark, for deterministic processes (not for stochastic) from the convergence by some norm follows convergence by any norm.

Ok, let's consider Lyapunov function $V(x)$ from a class of K^v and some conditions of this function is necessary: function is convex positively homogeneous power of v and following inequality is satisfied:

$$C_1^v \|x\|^v \le V(x) \le C_2^v \|x\|^v. \tag{10}$$

Let's consider Lyapunov functions from a class of quadratic forms:

$$V(x) = x^T P x, \tag{11}$$

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.

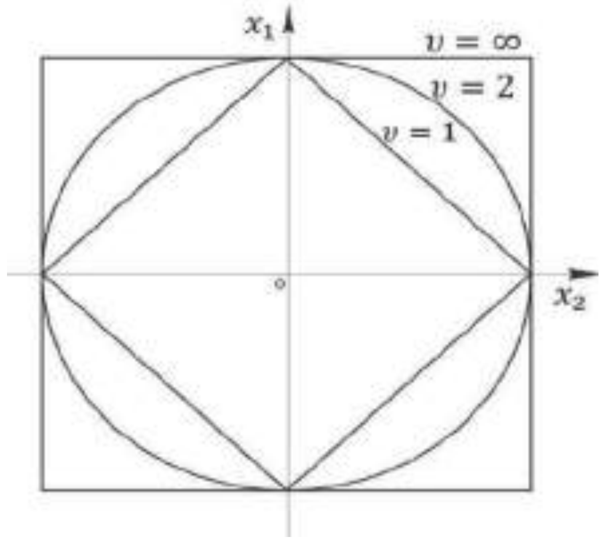


Fig. 3 – Geometric interpretation of different norms.

<u>The sufficient conditions of qualitative exponential stability</u>: for system (1) sufficiently existing number of $0 < \lambda < 1$, for any number of discreetness interval $m > 0$ the following inequality is satisfied:

$$V\big(x(m+1)\big) \le \lambda^{v} V\big(x(m)\big), \tag{12}$$

where $\rho = \frac{C_1}{C_2}$ and existing number $1-\lambda < \lambda_0 < 1+\lambda$, that for the system (1) following inequality is satisfied:

$$V\big(x(m+1) - x(m)\big) \le \lambda_0^{v} V\big(x(m)\big). \tag{13}$$

From these sufficient conditions, we can get two consequences.

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

$$V\left(x(m+1) - \frac{\lambda-\lambda_0+1}{2} x(m)\right) \le \left(\frac{\lambda+\lambda_0-1}{2}\right)^{v} V\big(x(m)\big). \tag{14}$$

<u>Consequence 2.</u> For qualitative exponential stability of system (1) 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 for system (1) is satisfied:

$$V\big(x(m+1) - (r+\alpha)x(m)\big) \le r^{v} V\big(x(m)\big), \tag{15}$$

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

At first, let's consider *geometric interpretation* of *exponential stability*. In case of quadratic form of Lyapunov function the condition of exponential stability is:

$$V\big(x(m+1)\big) \le \lambda^{2} V\big(x(m)\big). \tag{16}$$

From the condition follows, the next value of state vector $x(m+1)$ must belongs to area:

$$\Omega_x(\lambda) = \{\, x\colon x^T P x \le \lambda^2 x^T(m) P x(m)\} \tag{17}$$

if the previous value of state vector was on a surface $x^T P x = x^T(m) P x(m)$. This case is shown on the Fig. 4.

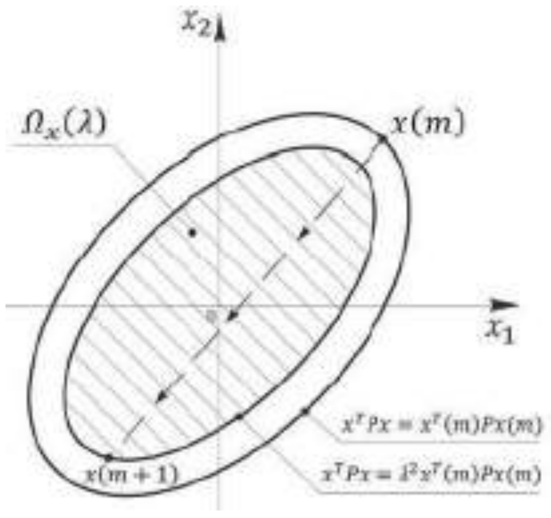

x_2
$\Omega_x(\lambda)$
$x(m)$
0
x_1
$x^T P x = x^T(m) P x(m)$
$x(m+1)$
$x^T P x = \lambda^2 x^T(m) P x(m)$

Fig. 4 – Geometric interpretation of exponential stability.

Let's consider *geometric interpretation* of *Consequence 1.*

1. System (1) must be exponential stable.

2. The inequality (13) must be valid. For Lyapunov functions from a class of K^2 the inequality (13) takes form:

$$\big(x(m+1)-x(m)\big)^T P\big(x(m+1)-x(m)\big) \le \lambda_0^2 x^T(m)Px(m). \tag{18}$$

From this condition follows, that each next value of state vector $x(m+1)$ with fixed $x(m)$ must belongs to area:

$$\Omega_x(\lambda_0) = \Big\{ x: \big(x-x(m)\big)^T P\big(x-x(m)\big) \le \lambda_0^2 x^T(m)Px(m)\Big\} \tag{19}$$

The border of this area is ellipsoid with center at the point $x(m)$.

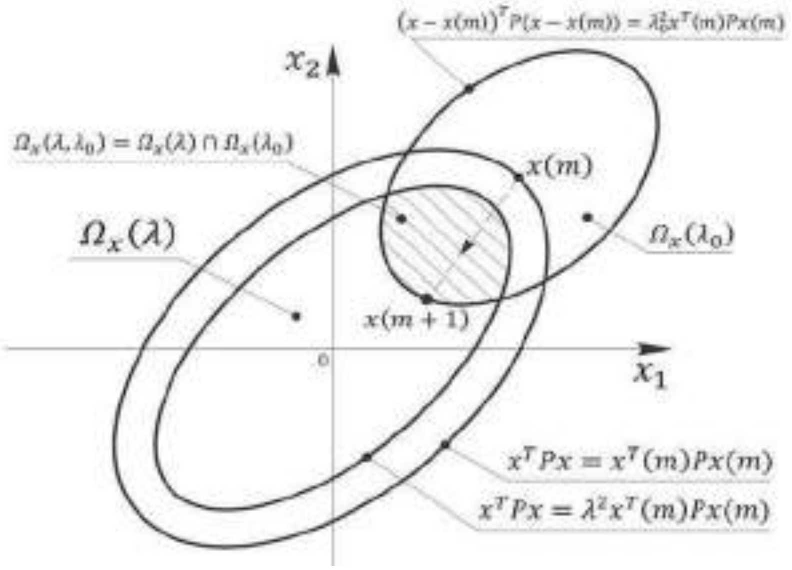

$(x-x(m))^T P(x-x(m)) = \lambda_0^2 x^T(m)Px(m)$
x_2
$\Omega_x(\lambda,\lambda_0) = \Omega_x(\lambda) \cap \Omega_x(\lambda_0)$
$x(m)$
$\Omega_x(\lambda)$
$\Omega_x(\lambda_0)$
$x(m+1)$
0
x_1
$x^T P x = x^T(m) P x(m)$
$x^T P x = \lambda^2 x^T(m) P x(m)$

Fig. 5 – Geometric interpretation of Consequence 1.

3. Inequalities (12) and (13) requires for each fixed arbitrary value $x(m)$ the next value of state vector $x(m+1)$ belongs to area:

$$\Omega_x(\lambda,\lambda_0) = \Omega_x(\lambda) \cap \Omega_x(\lambda_0). \tag{20}$$

As a conclude, local conditions of qualitative exponential stability distinguish from all values of state vector $\Omega_x(\lambda)$ some part $\Omega_x(\lambda, \lambda_0)$. This fact localized behavior of system motion trajectories.

And let's consider *geometric interpretation* of *Consequence 2*. For Lyapunov functions from a class of K^2 the inequality (15) takes form:

$$\big(x(m+1) - (r+\alpha)x(m)\big)^T P\big(x(m+1) - (r+\alpha)x(m)\big) \le r^2 x^T(m)Px(m), \tag{21}$$

From this condition follows, that each next value of state vector $x(m+1)$ with fixed $x(m)$ must belongs to area:

$$\Omega_x(r,\alpha) = \Big\{ x\colon \big(x - (r+\alpha)x(m)\big)^T P\big(x - (r+\alpha)x(m)\big) < r^2 x^T(m)Px(m) \Big\} \tag{22}$$

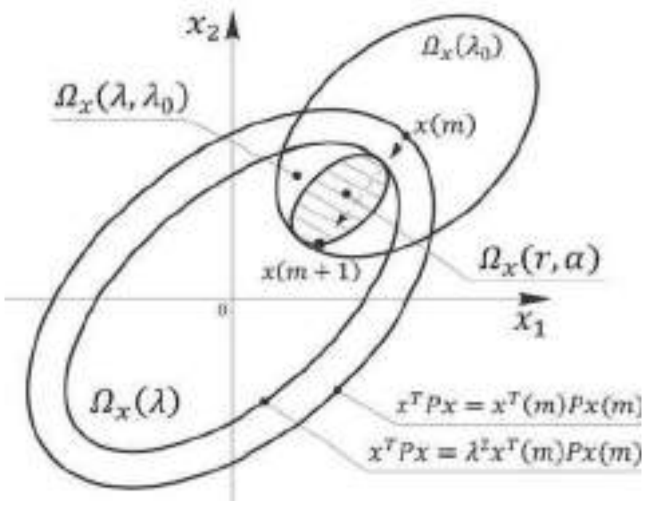


Fig. 6 – Geometric interpretation of Consequence 2.

In the Fig. 6 the area $\Omega_x(r,\alpha)$ is shaded. In case, when $\lambda = 2r+\alpha$, $\lambda_0 = 1-\alpha$ the area belongs to intersection areas $\Omega_x(\lambda_0)$ and $\Omega_x(\lambda)$. I mean $\Omega_x(r,\alpha) \subset \Omega_x(\lambda, \lambda_0)$.

As a conclude, if the system is satisfied for (r,α)-restrictions, then the strongest boundaries are for motion trajectories and system has good quality indicators.

Now, let's consider system (1) in matrix form:

$$x(m+1) = F \cdot \big(x(m)\big), \tag{23}$$

where F – square matrix of $n \times n$ dimension.

So, we should use matrix inequality and modified Lyapunov equation for investigation of qualitative exponential stability. In case of Lyapunov function from a quadratic form class, the inequality is:

$$(F - (r+\alpha)I)^T P (F - (r+\alpha)I) \le r^2 P, \tag{24}$$

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

The inequality (24) is valid in case of valid following modified Lyapunov equation:

$$(F - (r+\alpha)I)^T P (F - (r+\alpha)I) - r^2 P = -Q, \tag{25}$$

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

Let's consider connections of matrix inequalities and modified Lyapunov equations with areas of roots distribution in linear systems.

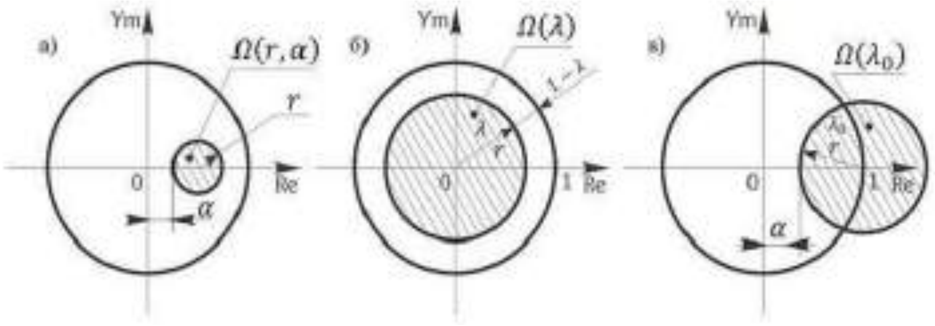


Fig. 7 – Areas of roots distribution.

In case of satisfied equation (25) the eigenvalues of system are in the shaded circle 7a. This case relates to Consequence 2.

In case of $-r = \alpha$ and $r = \lambda$ the eigenvalues of system are in the shaded circle 7b. This case relates to Consequence 1. The value $1 - \lambda$ is stability margin.

In case $\alpha + r = 1$ and $r = \lambda_0$ the eigenvalues of system are in the shaded circle 7c.

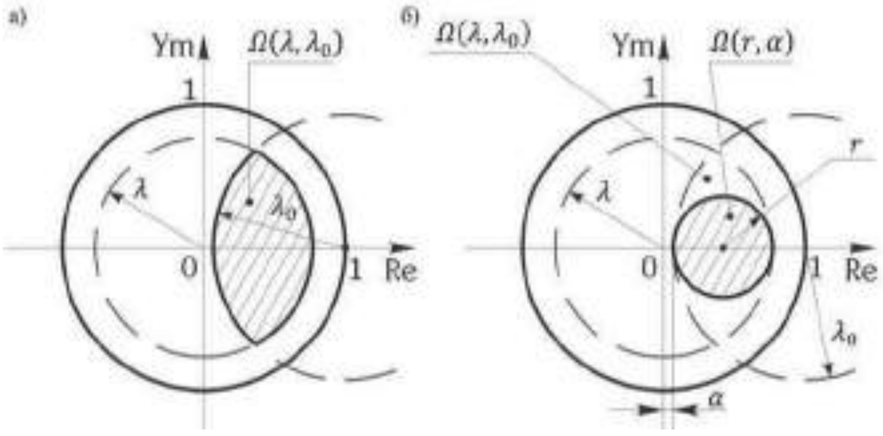


Fig. 8 – Intersections of stability areas.

The picture 8a relates to Consequence 1. The sufficient condition of qualitative exponential stable system is existing number $1 - \lambda < \lambda_0 < 1 + \lambda$ such following equation is satisfied:

$$(F - I)^T P(F - I) - \lambda_0^2 P = -Q. \tag{26}$$

The picture 8b relates to Consequence 2. The sufficient condition of qualitative exponential stable system is existing numbers $0 < r < 1$ and $0 < \alpha < 1 - 2r$ such equation (25) is satisfied.

Now, let's consider continuous case. Let's consider following continuous linear system:

$$\dot{x} = F(x), \tag{27}$$

where x – state vector of n-dimension.

Local conditions of qualitative exponential stability for continuous linear systems:

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

$$\dot{V}(x(t)) \le -2\alpha V(x(t)), \tag{28}$$

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

$$V(\dot{x}(t)) \le \lambda_0^2 V(x(t)), \tag{29}$$

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

Let's consider *geometric interpretation of Condition 1*. Rewrite (28) in the partial differential equation form:

$$\frac{\partial V(x)}{\partial x}\dot{x} \le -2\alpha V(x), \tag{30}$$

and we can image this condition on a complex plane:

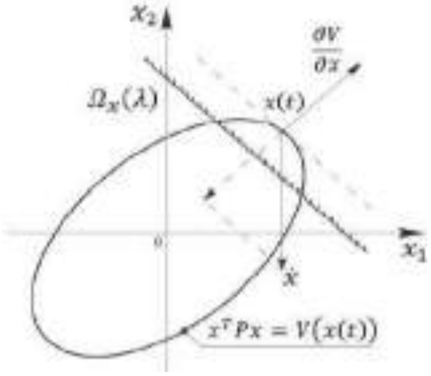


Fig. 9 – Geometric interpretation of exponential stability.

The second condition means that all possible values of state vector must belong to area $\Omega_x(\lambda_0)$ bounded by surface:

$$\dot{x}^T P \dot{x} = \lambda_0^2 V\big(x(t)\big). \tag{31}$$

Simultaneous performing both conditions means that values of state vector must belong to area $\Omega_x(\alpha,\lambda_0) = \Omega_x(\alpha) \cap \Omega_x(\lambda_0)$. It's shown on the Fig. 10.

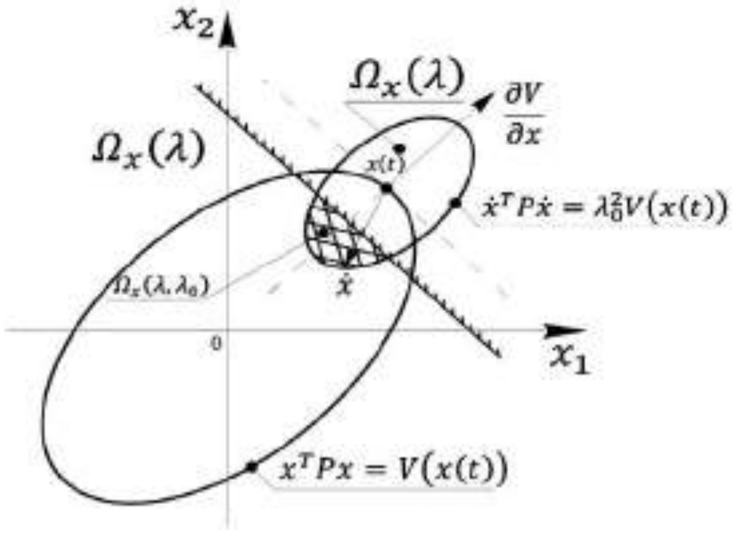

x_2
$\Omega_x(\lambda)$
$\frac{\partial V}{\partial x}$
$\Omega_x(\lambda)$
$x(t)$
$\dot{x}^T P \dot{x} = \lambda_0^2 V(x(t))$
$\Omega_x(\lambda,\lambda_0)$
$\dot{x}$
0
x_1
$x^T P x = V(x(t))$

Fig. 9 – Geometric interpretation of qualitative exponential stability.

Local conditions of qualitative exponential stability as one condition. Let's Lyapunov function from class K^2 of quadratic forms. System (27) is qualitative exponential stable if existing numbers $\alpha > 0$ and $r > 0$ such for any time $t > 0$ following inequality is satisfied:

$$V\big(\dot{x}(t) + (r+\alpha)x(t)\big) \le r^2 V\big(x(t)\big), \tag{32}$$

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

From condition (32) follows that 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:

$$\left(\dot{x} + (r + \alpha)x(t)\right)^T P\left(\dot{x} + (r + \alpha)x(t)\right) = r^2 V\left(x(t)\right), \tag{33}$$

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

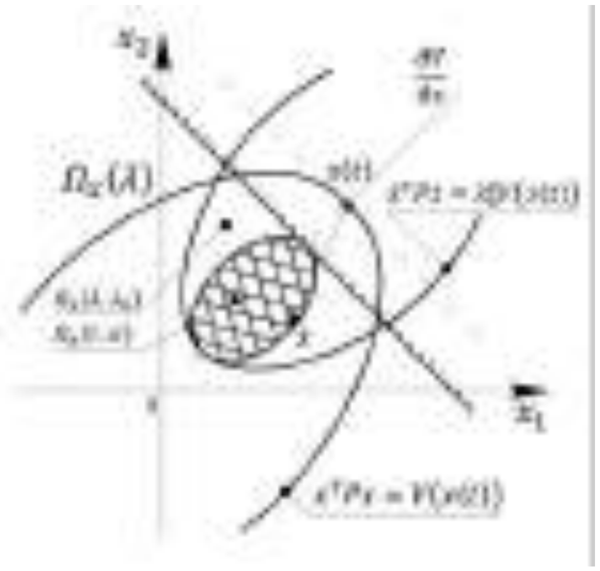


Fig. 10 – Geometric interpretation of qualitative exponential stability (r, α).

Now, let's consider roots distribution in linear continuous systems.

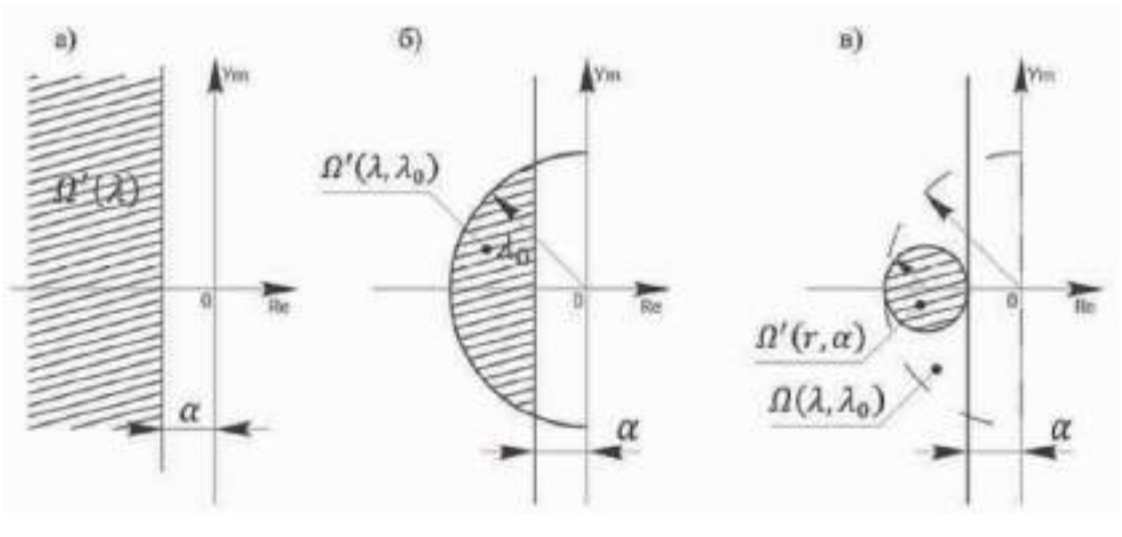


Fig. 11 – Roots distribution in linear continuous systems.

Fig. 11a – exponential stability, Fig. 11b – qualitative exponential stability, pic 11c – qualitative exponential stability (r, α).

So, in case considering system (27) in matrix form, the *sufficient condition* of qualitative exponential stability is existing numbers (r, α) $\lambda = \alpha, \lambda_0 = 2r + \alpha$, that Lyapunov equation is satisfied:

$$(F + (r + \alpha)I)^T P(F + (r + \alpha)I) - r^2 P = -Q, \tag{34}$$

or following matrix inequality:

$$(F + (r + \alpha)I)^T P(F + (r + \alpha)I) \leq r^2 P. \tag{35}$$

Port-Hamiltonian Control

In this section the port-Hamiltonian passivity-based control theory, which will applied to the control of the DFIM and the B2B, is presented. We start with a review of the basic ideas of passivity and of control by interconnection, move to the Interconnection and Damping Assignment—Passivity-based Control (IDA-PBC), and finally discuss two improvements of the basic IDA-PBC framework, namely Simultaneous Interconnection and Damping Assignment (SIDA), and a variant of the method which improves the robustness of the controller in front of uncertain parameters.

1 Introduction

According to one of the acceptions "to control" means "to exercise restraint or direction over". In an engineering context, we can translate this to "to stabilize a system in a desired equilibrium point or trajectory". Although a variety of techniques are available for linear control theory, most nonlinear control theory revolves around Lyapunov's method and its variants. Lyapunov theory was introduced originally as an analysis tool and became an useful technique for feedback control design.

Lyapunov-based control design is a quite difficult task which involves the construction of a suitable Lyapunov function. This function can be interpreted, in physical systems, as the energy (or storage) function. The main difference between many nonlinear control techniques is the way in which the appropriate Lyapunov function is constructed, as is the case, for instance, of backstepping, forwarding or adaptive control. Some other techniques also use the Lyapunov method to design controllers, for instance Sliding Mode Control, a technique for robust control where the trajectories are forced to reach a sliding surface.

Passivity-based Control (PBC) uses the fact that passive nonlinear systems are described by an storage function (which is a proper Lyapunov function). The control design main goal is then to reshape the original energy function by means of the controller. Based on PBC, the IDA-PBC (Interconnection and Damping Assignment–Passivity-based Control) technique, which uses the passive properties of Port Hamiltonian Systems, was presented.

2 Passivity-based control

Traditionally, control problems have been approached adopting a signal-processing viewpoint. This is very useful for linear time-invariant systems, where signals can be discriminated via filtering. However, for nonlinear systems, frequency mixing invalidates this approach due to the following reasons:

• Computations are far from obvious.

• Very complex controls are needed to quench the large set of undesirable signals, and the result is very inefficient, with a lot of energy being consumed and always on the verge of instability (a typical example is provided by bipedal walking machines).

Most of the problem stems from the fact that no information about the structure of the system is used. A change of control paradigm is needed, and this can be summarized in the catch expression "control systems as energy exchanging entities".

2.1 Energy-based control

Definition 1. The map $u \to y$ is passive if there exists a state function $H(x)$, bounded from below, and a nonnegative function $d(t) \geq 0$ such that

$$\underbrace{\int_0^t u^t(s)y(s)ds}_{\textit{energy supplied to the system}} = \underbrace{H(x(t)) - H(x(0))}_{\textit{stored energy}} + \underbrace{d(t)}_{\textit{dissipated energy}}. \qquad (1)$$

Example 1. (A mechanical system). The simplest example of passive system is probably the forced mass-spring-damper arrangement of Figure 1, where *q* is the mass position, *F(t)* is an external applied force, *m* is the mass and *b* and *k* are the damping and spring coefficients, respectively. One has

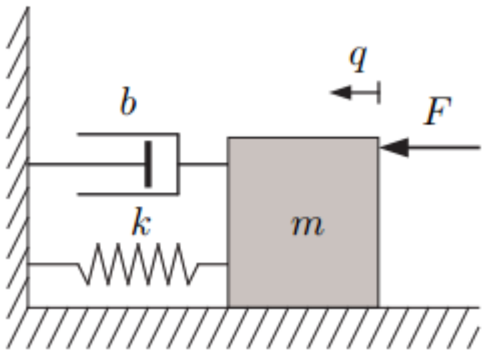


Figure 1: Example of a mechanical passive system

(with $v = \dot{q}$ as mechanical velocity)

$$\int_0^t F(s)v(s)ds = \int_0^t (m\dot{v}(s) + kq(s) + bv(s))ds$$

$$= \left(\frac{1}{2}mv^2(s) + \frac{1}{2}kq^2(s)\right)\Bigg|_0^t + b\int_0^t v^2(s)ds$$

$$= H(x(t)) - H(x(0)) + b\int_0^t v^2(s)ds.$$

Remark 1. Notice that, a passive system (1), if x^* is a global minimum of *H(x)* and *d(t) > 0*, and setting *u = 0*, *H(x)* will decrease in time and the system will reach x^* asymptotically. The rate of convergence can be increased if the energy is extracted from the system with

$$u = -K_{di}y$$

with $K_{di}^T = K_{di} > 0.$ △

The key idea of passivity-based control (PBC) is as follows; use feedback

$$u(t) = \beta(x(t)), \qquad (2)$$

where *β(x(t))* is a function depending on the states, so that the closed-loop system is again a passive system, with new energy function H_d, with respect to $\alpha \to y$, such that H_d has the global minimum at the desired point. Passivity for the closed-loop system is far from obvious: physically, the controller is injecting energy into the system. PBC is robust with respect to unmodeled dissipation, and has built-in safety: even if *H* is not known exactly, if passivity is preserved, the system will stop somewhere instead of running away and finally blowing up.

With (2), H_a is defined as (minus) the energy supplied to the system,

$$H_a = -\int_0^t \beta^T(x(s))y(s),$$

then the closed-loop system has energy function $H_d(x) = H(x) - H_a(x)$. One has the following *energy balance equation* (EBE), which yields an interpretation to PBC:

$$H_d(x(t)) = H(x(t)) - \int_0^t \beta^T(x(s))y(s).$$

Remark 2. For an affine dynamical system

$$\begin{cases} \dot{x} = f(x) + g(x)u \\ y = h(x) \end{cases},$$

the EBE is equivalent to the PDE

$$-\beta^T(x)h(x) = \partial H_a(f(x) + g(x)\beta(x)). \qquad (3)$$

Example 2. (Electrical system). As an example, consider the electrical system in Figure 2,

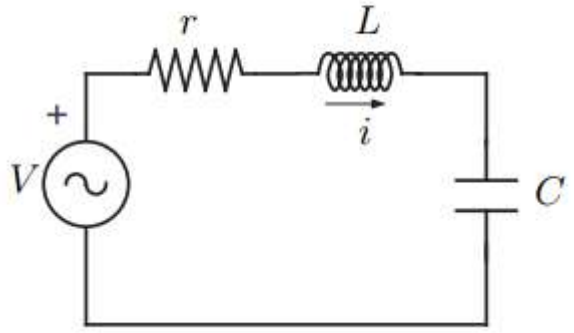


Figure 2: Example of an electrical passive system

$$\dot{x} = \begin{bmatrix} -\dfrac{x_2}{L} \\ -\dfrac{x_1}{C} - \dfrac{x_2}{L}r \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} V$$

where $x = [q, \lambda]^T$ is the state, $u = V$ is the control input and $y = \dfrac{\lambda}{L}$ (inductor current) is the passive output. The map $V \to i$ is passive with energy function

$$H(x) = \frac{1}{2C}x_1^2 + \frac{1}{2L}x_2^2$$

and dissipation

$$d(t)=\int_0^t \frac{r}{L^2}x_2^2(s)ds.$$

Notice that the natural minimum is [0, 0], but forced equilibrium points are of the form $[x_1^*, 0]$. The PDE (3) is in this case

$$\frac{x_2}{L}\partial H_a-\left(\frac{1}{C}x_1+\frac{r}{L}x_2-\beta(x)\right)\partial H_a=-\frac{1}{L}x_2\beta(x).$$

Since $x_2^*=0$ is already a minimum of *H*, its only necessary to shape the energy in *x1*. Hence, taking $H_a = H_a(x_1)$ and solving the above PDE

$$\beta(x_1)=\frac{\partial H_a}{\partial x_1}$$

i.e. it defines a closed-loop control. Then, one has to choose H_a so that H_d has the minimum at x_1^*. The simplest solution is

$$H_a(x_1)=\frac{1}{2C_a}x_1^2-\left(\frac{1}{C}+\frac{1}{C_a}\right)x_1^*x_1$$

where C_a is a design parameter. The closed-loop energy H_d can then be computed and it is seen that it has a minimum at $[x_1^*, 0]$ if $C_a > -C$. Finally, the control is computed as

$$u=\frac{\partial H_a}{\partial x_1}=-\frac{1}{C_a}x_1+\left(\frac{1}{C}+\frac{1}{C_a}\right)x_1^*.$$

This control is an energy-balancing PBC that stabilizes x∗ under stated parameter restrictions.

Example 3. (Electrical system). Consider now the slightly different circuit of Figure 3.

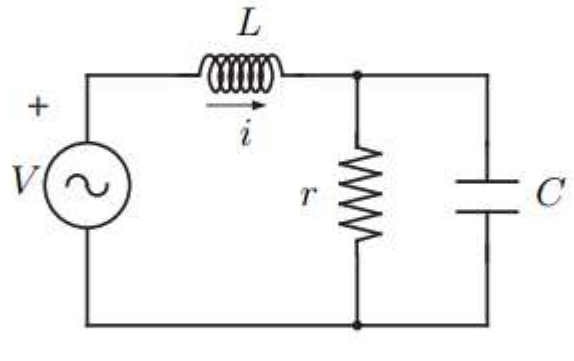


Figure 3: Example of an electrical passive system

With the same states, energy, input and outputs than the preceding system, the equations of the motion are now

$$\dot{x}=\begin{bmatrix}-\frac{1}{rC}x_1+\frac{1}{L}x_2\\ -\frac{1}{C}x_1\end{bmatrix}+\begin{bmatrix}0\\1\end{bmatrix}V$$

Only the dissipation structure has changed, but the admissible equilibria are of the form

$$x^* = \left[CV^d, \frac{L}{r} V^d \right]^T$$

for any constant V^d. The power delivered by the source, $p = V_i = V \frac{x_2}{L}$, is nonzero at any equilibrium point except for the trivial one. Hence, the source has to provide an infinite amount of energy to keep any nontrivial equilibrium point, a task which is clearly not feasible. This situation will reappear later into the discussion of invariant and Casimir functions. Notice that pure mechanical systems are free of this problem, since any equilibrium has velocities equal to zero and hence no power in necessary to keep the system at the equilibrium point.

2.2 Control as an interconnection

To give a physical interpretation of PBC, one can think the controller as a *system exchanging energy with the plant.* Consider two systems, ∑ and ∑c, exchanging energy through an interconnection network given by ∑l, as depicted in Figure 4.

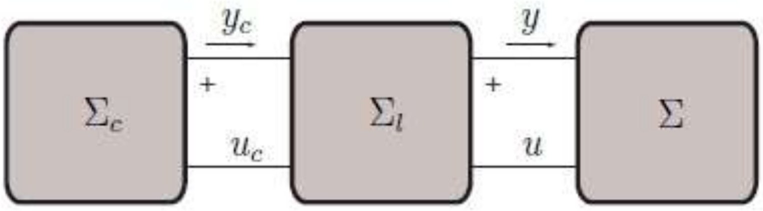


Figure 4: Network interpretation of control

The condition for the interconnection to be power continuous is

$$u_c^T(t) y_c(t) + u^T(t) y(t) = 0 \ \forall t$$

Example 4. (Feedback interconnection). As an example, consider the typical negative feedback interconnection displayed in Figure 5

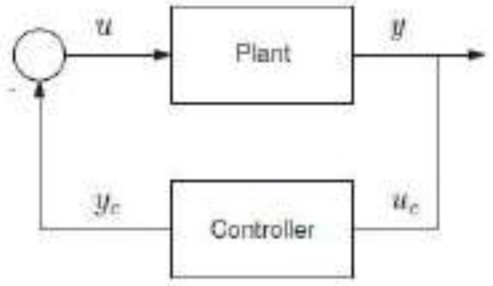


Figure 5: Typical negative feedback interconnection

The interconnection is given by

$$u_c = y$$
$$u = -y_c$$

and is clearly power continuous.

Suppose now that some extra inputs $u \to u+v$, $u_c \to u_c + v_c$ are added to the interconnected system. Then is easy to show the following.

Remark 3. Let Σ and Σ_c have the state variables x and ξ. If Σ and Σ_c are passive with energy functions $H(x)$ and $H_c(\xi)$ and Σ_I is power preserving, then the map $[v, v_c] \to [y, y_c]$ is passive for the interconnected system with energy function $H_d(x, \xi) = H(x)+H_c(\xi)$. Or, in short, $\triangle$

Proposition 1. Power continuous interconnection of passive system yields passive system.

The resulting system of the interconnection of the plant and the controller is a passive system with energy function

$$H_d(x,\xi) = H(x) + H_c(\xi)$$

but this is not very useful unless the energy function depends only on x. To solve this, the dynamics are restricted to a submanifold of the (x, ξ) space parametrized by x:

$$\Omega_K = (x,\xi);\ \xi = F(x) + K,$$

and dynamically invariant:

$$(\partial F\dot{x})_{\xi=F(x)+K} = 0.$$

Instead of solving this in general, it is convenient to formulate the problem for a port-controlled Hamiltonian systems.

2.3 Casimir functions and the dissipation obstacle

A port-controlled system in explicit form given by (1), remind,

$$\begin{cases} \dot{x} = (J(x) - R(x))\partial H(x) + g(x)u \\ y = g^T(x)\partial H(x) \end{cases}$$

with $J^T = -J$, $R^T = R \geq 0$ and $H > 0$, satisfy the following relation

$$\dot{H} = -(\partial H)^T R\partial H + y^T u.$$

Integrating, from 0 to t, the energy balance equation, is recovered

$$H(x(t)) - H(x(0)) = \int_0^t u^T(s)y(s)ds - \int_0^t (\partial H)^T R\partial H.$$

More precise results about the possibility of obtaining invariant manifolds expressing the controller variables in terms of the variables of the system can be formulated if both systems and controller are PCHS. Let thus

$$\sum : \begin{cases} \dot{x} = (J(x) - R(x))\partial H(x) + g(x)u \\ y = g^T(x)\partial H(x) \end{cases}$$

define the plant and

$$\sum\nolimits_c : \begin{cases} \dot{\xi} = (J_c(\xi) - R_c(\xi))\partial H_c(\xi) + g_c(\xi)u_c \\ y_c = g_c^T(\xi)\partial H_c(\xi) \end{cases}$$

define the controller. With the power preserving, standard negative feedback interconnection $u = -y_c$, $u_c = y$, one gets

$$\begin{bmatrix} \dot{x} \\ \dot{\xi} \end{bmatrix} = \begin{bmatrix} J(x) - R(x) & -g(x)g_c(\xi)^T \\ g_c(\xi)g(x)^T & J_c(\xi) - R_c(\xi) \end{bmatrix} \begin{bmatrix} \partial H_d(x) \\ \partial H_d(\xi) \end{bmatrix}$$

where $H_d(x, \xi) = H(x) + H_c(\xi)$. Let us look now for invariant manifolds of the form

$$C_K(x, \xi) = F(x) - \xi + K.$$

Condition $\dot{C}_K = 0$ yields

$$[\partial F, I_m]^T \begin{bmatrix} J(x) - R(x) & -g(x)g_c(\xi)^T \\ g_c(\xi)g(x)^T & J_c(\xi) - R_c(\xi) \end{bmatrix} \begin{bmatrix} \partial H_d(x) \\ \partial H_d(\xi) \end{bmatrix} = 0.$$

In order to keep the freedom to choose H_c, one demands that the above equation is satisfied on C_K for every Hamiltonian, *i.e.* one imposes on F the following system of PDE's:

$$[\partial F, I_m]^T \begin{bmatrix} J(x) - R(x) & -g(x)g_c(\xi)^T \\ g_c(\xi)g(x)^T & J_c(\xi) - R_c(\xi) \end{bmatrix} = 0.$$

Functions $C_K(x, \xi)$ such that F satisfies the above PDE on $C_K = 0$ are called Casimir. They are invariants associated to the structure of the system (J, R, g, J_c, R_c, g_c), independently of the Hamiltonian function.

One can show that the PDE for F has solution iff, on $C_K = 0$,

1. $(\partial F)^T J \partial F = J_c$,
2. $R\partial F = 0$,
3. $R_c = 0$,
4. $(\partial F)^T J = g_c g^T$.

Conditions 2 and 3 are easy to understand: essentially, no Casimir functions exist in presence of dissipation. Given the structure of the PDE, $R_c = 0$ is unavoidable, but one can have an effective $R = 0$ just by demanding that the coordinates on which the Casimir depends do not have dissipation, and hence condition 2.

If the preceding conditions are fulfilled, an easy computation shows that the dynamics on C_K is given by

$$\dot{x} = (J(x) - R(x))\partial H_d$$

with $H_d(x) = H(x) + H_c(F(x) + K)$. Notice that, due to condition 2,

$$R\partial H_c(F(x) + K) = \underbrace{R(\partial F)}_{=0} \frac{\partial H_c}{\partial \xi}(F(x) + K) = 0,$$

so, in energy-balancing PBC, dissipation is only admissible for those coordinates which do not require energy shaping.

For regulation problems in mechanical systems, where the state consists of positions and velocities, dissipation only appear associated to the later, while energy shaping is necessary only in the position part, since the kinetic energy already has the minimum at the desired point (that is, at velocity equal to zero). Hence, the dissipation obstacle is always absent for mechanical regulation problems. For the first of the two simple RLC circuits considered previously (Figures 3.2 and 3.3), dissipation appears in a coordinate, x_2, which already has the minimum at the desired point. For the second one, the minimum of the energy has to be moved for both coordinates, and hence the dissipation obstacle is unavoidable.

3 Interconnection and damping assignment – Passivity-based control

The Interconnection and damping assignment–Passivity-based control (IDA-PBC) was introduced to combine the passivity properties of PCHS with control by interconnection and energy-based control. This technique has been applied to many different plants: mechanical systems, magnetic levitation systems, mass-balance systems, electric machines, power converter.

The key idea is that using the Hamiltonian framework, solving the PDE associated to the energy-balance equation (3) can be done with an appropriate selection of the interconnection J and dissipation R matrices and the energy function H of the desired closed-loop system (which will be denoted with subindex d: J_d, R_d and H_d).

3.1 IDA-PBC technique

One can get a method with more freedom if not only the energy function is changed but also the interconnection J and dissipation R, *i.e.* if one aims at a closed-loop system of the form

$$\dot{x} = (J_d(x) - R_d(x))\partial H_d(x), \tag{4}$$

where $J_d = -J_d^T$ is the *desired* interconnection matrix, $R_d = R_d^T \geq 0$ is the *desired* dissipation matrix and Hd (with a minimum at x^*) is the *desired* Hamiltonian function.

Proposition 2. Consider the system

$$\dot{x} = f(x) + g(x)u \tag{5}$$

Assume there are matrices $J_d = -J_d^T$, $R_d = R_d^T \geq 0$ and a smooth function H_d that verify the so-called matching equation

$$f(x) + g(x)u = (J_d(x) - R_d(x))\partial H_d(x). \tag{6}$$

Then the closed-loop system with control $u = \beta(x)$,

$$\beta(x) = (g^T(x)g(x))^{-1} g(x)((J_d(x) - R_d(x))\partial H_d(x) - f(x)) \tag{7}$$

is asymptotically stable.

Proof. Substituting (7) into (5) the closed-loop system becomes

$$\dot{x} = (J_d(x) - R_d(x))\partial H_d,$$

which, following Proposition 1, is asymptotically stable.

It is thus clear that the problem is how to solve the matching equation (6). Notice that there is a huge amount of freedom in selecting J_d, R_d and H_d satisfying the previous assumptions ($J_d = -J_d^T$, $R_d = R_d^T \geq 0$ and $x^* = \arg\min H_d$). **In Non-Parameterized IDA**, the structure and damping matrices ($J_d(x)$ and $R_d(x)$) are fixed, the matching equation is pre-multiplied by a left annihilator of $g(x)$ and the resulting PDE in H_d is then solved.

- **In Algebraic IDA** the desired Hamiltonian function H_d is first selected (for example a quadratic function in the error terms) and then the resulting algebraic equations are solved for J_d and *Rd.*
- **In Parameterized IDA**, applicable mainly to underactuated mechanical systems the knowledge of a priori structure of the desired Hamiltonian is used to obtain a more easy to solve PDE, giving constraints on J_d and R_d.
- **In Interlaced Algebraic-Parameterized IDA** the PDE is evaluated in some subspace (where the solution can be easily computed) and then matrices J_d, R_d are found which ensure a valid solution of the matching equation.

There is not a *best* method to solve the matching equation. Each control problem requires an individual study to find out which of the above strategies provides an acceptable solution of the matching equation.

The first papers on IDA-PBC introduced new matrices J_a, R_a and a Hamiltonian function H_a such that

$$J_d(x) \triangleq J(x) + J_a(x),$$
$$R_d(x) \triangleq R(x) + R_a(x),$$
$$H_d(x) \triangleq H(x) + H_a(x)$$

referred to as the structure matrix, damping matrix and Hamiltonian function, respectively, contributed by the controller. With this notation, and using a PCHS description of the system (5), the matching equation (6) becomes

$$(J(x) + J_a(x) - R(x) - R_a(x))\partial H_a = -(J_a(x) - R_a(x))\partial H + g(x)u, \qquad (8)$$

where the available degrees of freedom for the design are the matrices are J_a, R_a and the function H_a.

In order to clarify the methodology, and to compare later the classic IDA-PBC controllers to the designed ones using the new approaches presented in this Thesis (see subsection 3.2 and Section 4), we present here two examples: a classical DC motor and a nonlinear toy model.

Example 5. (A DC motor). Consider a permanent magnet DC motor (or either a field DC motor (Fig. 5.1) for which the field dynamics, λ_f, is neglected).

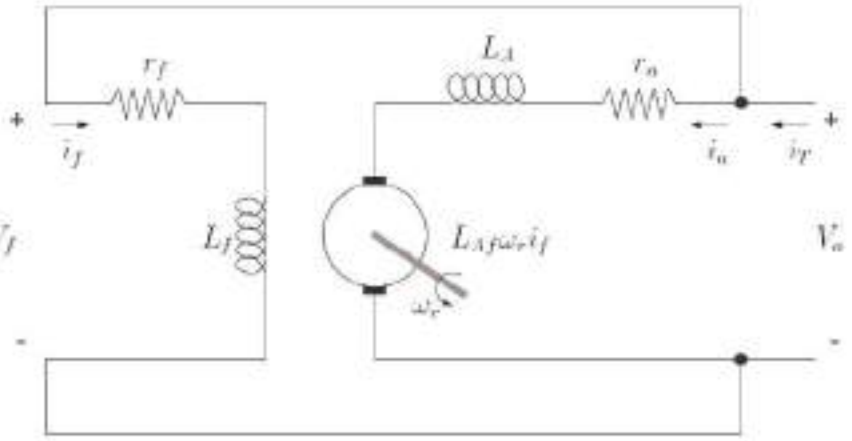


Figure 5.1: Shunt-connected dc machine

From the PCHS model of the DC motor and using K = $L_{Af}\, i_f = ct$, called the torque constant, the port-controlled Hamiltonian system is described by

$$\dot{x} = (J - R)\partial H(x) + g + g_u u$$

with the variables $x \in R^2$

$$x = [\lambda, p_m]^T$$

where λ is the inductor flux (or λ_a in the generic case), and p_m is the angular momentum. The interconnection, dissipation and port matrices are

$$J = \begin{bmatrix} 0 & -K \\ K & 0 \end{bmatrix} \; R = \begin{bmatrix} r & 0 \\ 0 & B_r \end{bmatrix} \; g = \begin{bmatrix} 0 \\ -\tau_L \end{bmatrix} \; g_u = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

with the control input $u = v$ (to simplify the notation the voltage v_a in now called v). Notice that the system inputs have been split according to whether they can be controlled or not when the machine acts as a motor. In this case, the mechanical torque can be seen as an external perturbation. r and B_r represent the electrical and mechanical losses respectively, and the Hamiltonian function is given by

$$H(x) = \frac{1}{2L}\lambda^2 + \frac{1}{2J_m} p_m^2,$$

where L is the inductance and J_m the inertia of the motor.

Assume that the control objective is a desired speed ω^d. In terms of ω^d, the equilibrium values of i and v are

$$i^* = \frac{1}{K}(B_r\omega^d + \tau_L)$$
$$u^* = ri^* + K\omega^d.$$

To apply the IDA-PBC technique (following the algebraic approach) a desired Hamiltonian function H_d is fixed as

$$H_d(x) = H(x) = \frac{1}{2L}(\lambda - \lambda^*)^2 + \frac{1}{2J_m}(p_m - p_m^*)^2,$$

which implies (recall the energy and co-energy variables relationship, $\lambda = Li$ and $p_m = J_m\omega$)

$$\partial H_d = \begin{bmatrix} i - i^* \\ \omega - \omega d \end{bmatrix}.$$

In order to solve the matching equation of the IDA-PBC method, we consider generalized interconnection and dissipation matrices given by

$$J_d - R_d = \begin{bmatrix} -r_d & -j_d \\ j_d & -b_d \end{bmatrix}. \tag{9}$$

The first row of the matching equation will yield the desired control action, while the second row imposes

$$j_d(i - i^*) - b_d(\omega - \omega^d) = K_i - B_r\omega - \tau_L.$$

Setting $b_d = B_r$, and using the equilibrium point expression, j_d is computed as

$$j_d = K,$$

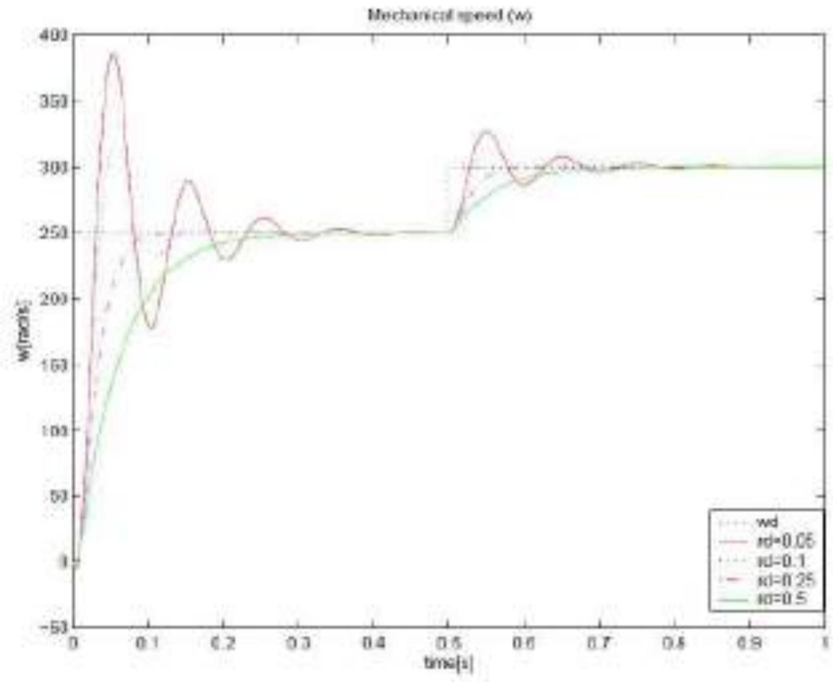


Figure 6: Simulation results: Mechanical speed ω, for different r_d values

where r_d is a still free parameter to tune the controller. Finally, substituting into the first row of the matching equation,

$$u = -r_d(i - i^*) - ri + K\omega^d \tag{10}$$

Notice that this is just a proportional + constant compensation controller.

Figures 6-8 show the system behavior with the control law (10). The motor parameters are: $r = 0.05\Omega$, $L = 2\text{mH}$, $K = 0.07\text{N}\cdot\text{m}\cdot\text{A}^{-1}$, $B_r = 0.0001\text{N}\cdot\text{m}\cdot\text{rad}^{-1}\text{s}^{-1}$, $J_m = 0.0006\text{Kg}\cdot\text{m}^2$ and the nominal torque is $\tau_L = 2\text{N}\cdot\text{m}$. The desired mechanical speed is fixed at $\omega_d = 250\text{rad}\cdot\text{s}^{-1}$ for $0\text{s} < t \leq 0.5\text{s}$ and changes at $\omega_d = 300\text{rad}\cdot\text{s}^{-1}$ for $0.5\text{s} < t \leq 1\text{s}$.

Figure 6 shows the mechanical speed for different damping r_d values. Notice that for a higher value of r_d the transient becomes more damped, which gives a physical interpretation of the R_d matrix (9). Figure 7 shows the inductor current i, with a similar behaviour to that of ω. Finally, the space-state trajectory for $0\text{s} < t \leq 0.5\text{s}$, which converges to the equilibrium point, is depicted in Figure 8

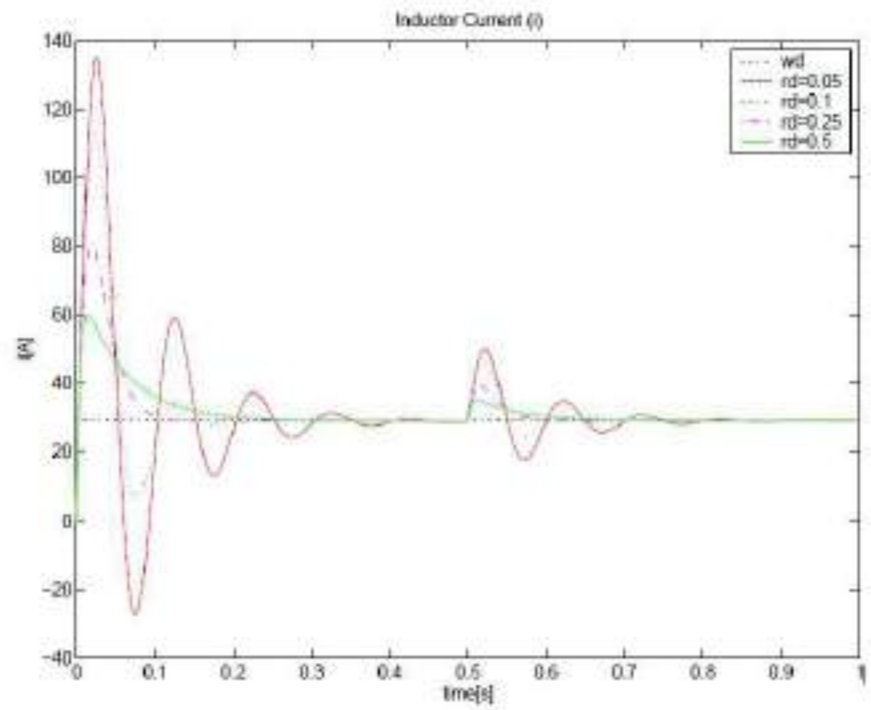


Figure 7: Simulation results: Inductor current *i*, for different r_d values

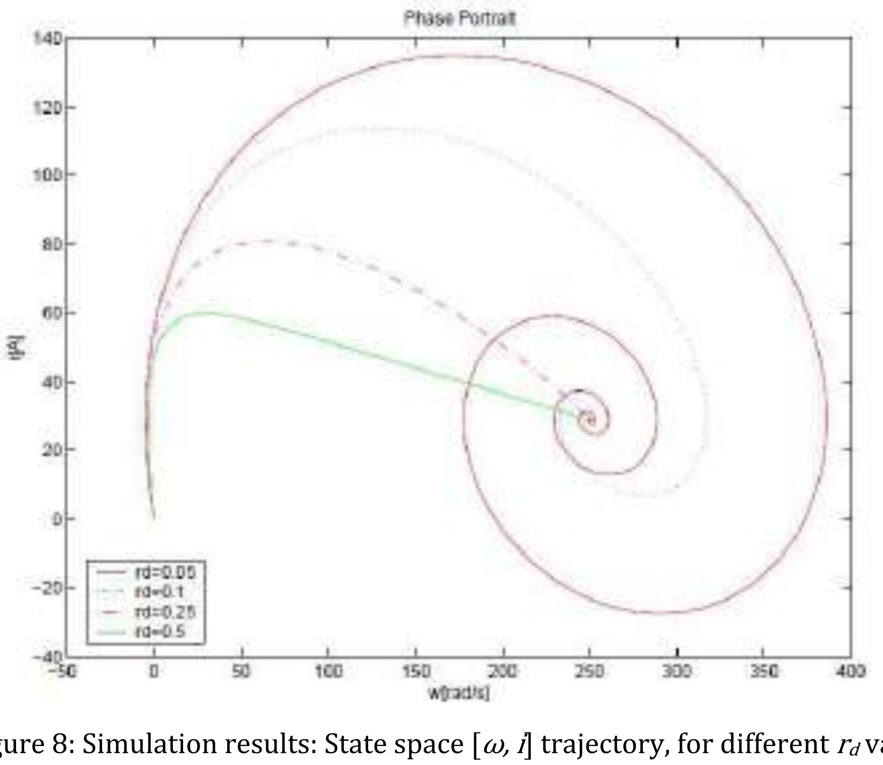


Figure 8: Simulation results: State space [ω, *i*] trajectory, for different r_d values

Example 6. (A toy model). Consider the following 2-dimensional nonlinear control system

$$\begin{aligned} \dot{x}_1 &= -x_1 + \xi x_2^2, \\ \dot{x}_2 &= -x_1 x_2 + u, \end{aligned} \qquad (11)$$

where $\xi > 0$. This can be cast into PCHS form

$$\dot{x}_1 = (J - R)\partial H + gu \tag{12}$$

with

$$J = \begin{bmatrix} 0 & x_2 \\ -x_2 & 0 \end{bmatrix},\ R = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix},\ g = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$H(x) = \frac{1}{2}x_1^2 + \frac{1}{2}\xi x_2^2$$

The control objective is to regulate, for example, x_2 to a desired value x_2^d (nevertheless the control law for a regulated x_1 yields the same control law). The equilibrium of (11) corresponding to this is given by

$$x_1^* = \xi(x_2^d)^2,\ u^* = \xi(x_2^d)^3.$$

Using the IDA-PBC technique, also within the algebraic approach, we match (12) to

$$\dot{x}_1 = (J_d - R_d)\partial H_d$$

with

$$J = \begin{bmatrix} 0 & \alpha(x) \\ -\alpha(x) & 0 \end{bmatrix},\ R_d = \begin{bmatrix} 1 & 0 \\ 0 & r \end{bmatrix},$$

and

$$H_d(x) = \frac{1}{2}(x_1 - x_1^*)^2 + \frac{1}{2\gamma}(x_2 - x_2^d)^2,$$

where α(x_1, x_2) is a function to be determined by the matching procedure and γ > 0, r > 0 are adjustable parameters.

From the first row of the matching equation $(J - R)\partial H + gu = (J_d - R_d)\partial H_d$ one gets

$$-x_1 + \xi x_2^2 = -(x_1 - x_1^*) + \frac{\alpha}{\gamma}(x_2 - x_2^d),$$

from which

$$\alpha(x_1, x_2) = \frac{\gamma}{x_2 - x_2^d}(\xi x_2^2 - x_1^*) = \gamma\xi(x_2 - x_2^d).$$

Substituting this into the second row of the matching equation

$$-x_1x_2 + u = -\alpha(x_1 - x_1^*) - \frac{r}{\gamma}(x_2 - x_2^d),$$

yields the feedback control law

$$u = x_1x_2 - \gamma\xi(x_1 - x_1^*)(x_2 + x_2^d) - \frac{r}{\gamma}(x_2 - x_2^d). \tag{13}$$

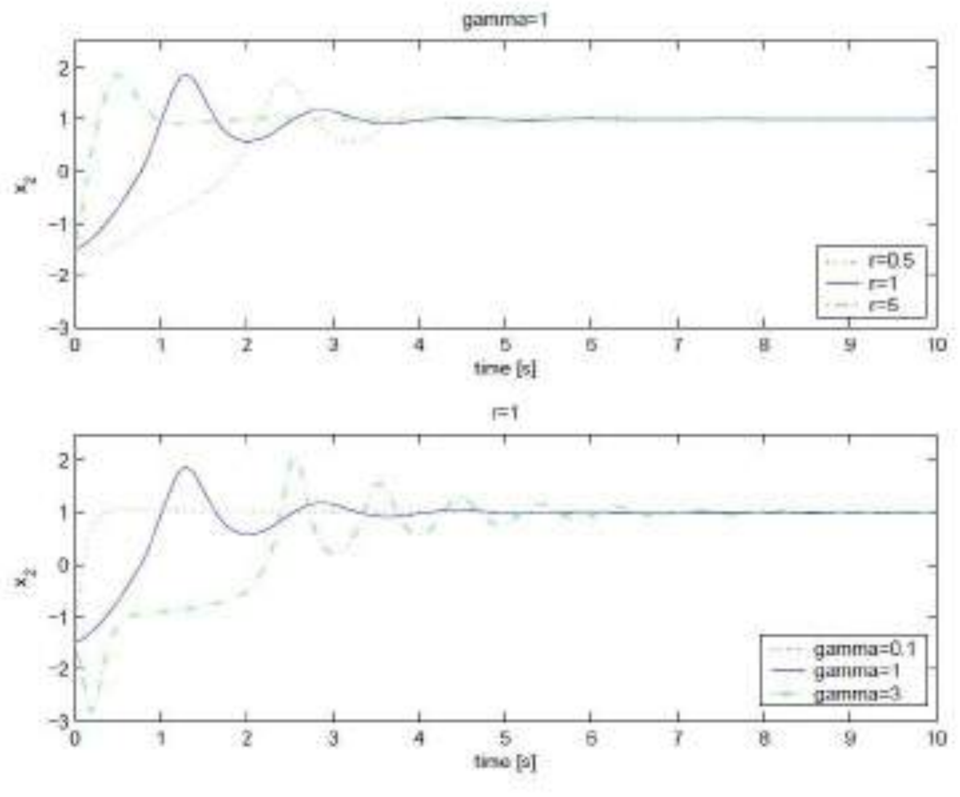


Figure 9: Simulation results: x_2 behaviour for different r and γ values

This control law yields a closed-loop system which is Hamiltonian with (J_d,R_d,H_d), and which has $(x_1^*,\ x_2^d)$ as a globally asymptotically stable equilibrium point.

Figures 9 to 12 show the behavior of the system controlled by the IDA-PBC controller (13). The parameters are ξ = 2 and x_2^d $x^d{}_2$= 1. Figures 9 and 10 show $x_2(t)$ and $x_1(t)$ for different r and γ values, while in Figures 11 and 12 the phase portrait is depicted. Notice that the γ parameter has more influence on the trajectories. This is due to the fact that γ modifies the Hamiltonian in the x_2 direction (see Figure 13) and tuning this parameter makes trajectories of x_2 restricted (or semi-bounded).

3.2 Simultaneous IDA-PBC

The standard two–stage procedure used in IDA-PBC, consisting of splitting the control action into the sum of energy-shaping and damping injection terms, is not without loss of generality, and effectively reduces the set of systems that can be stabilized with IDA–PBC. This assertion is, of course, not surprising since it is clear that, to achieve strict passivity, the procedure described above is just one of many other possible ways. This point is illustrated with the IDA–PBC design methodology proposed in (see the previous subsection). To enlarge the set of systems that can be stabilized via IDA–PBC we suggest to carry out *simultaneously* the energy shaping and the damping injection stages and refer to this variation of the method as SIDA–PBC.

As we said before, the key for the success of IDA-PBC is the solution of the matching

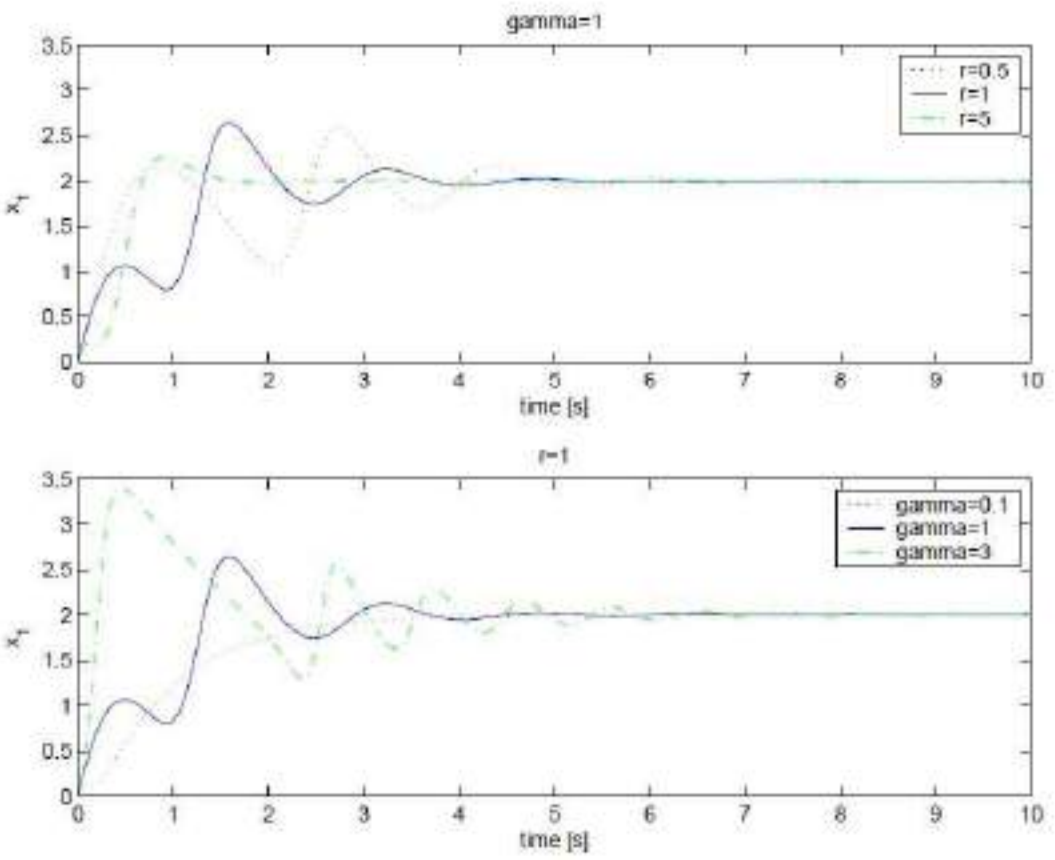


Figure 10: Simulation results: x_1 behaviour for different r and γ values

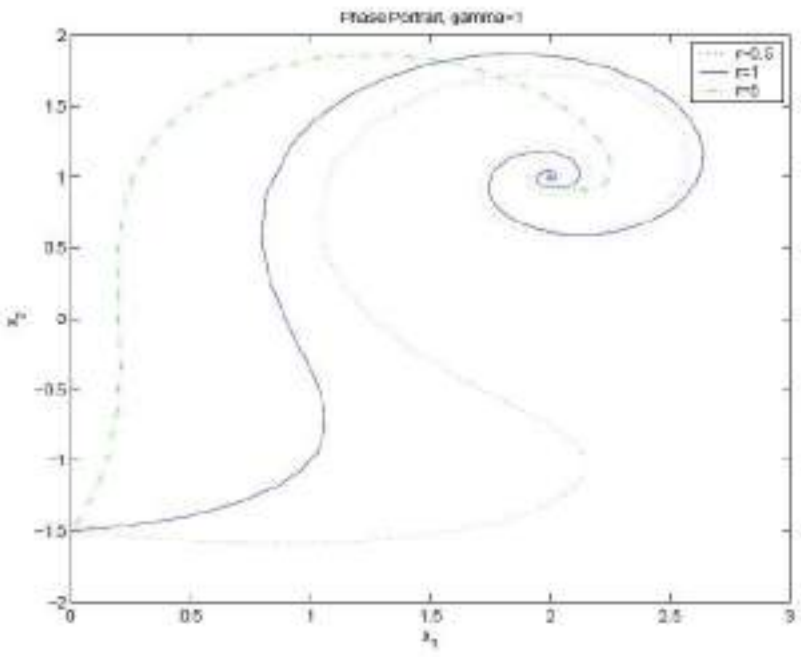


Figure 11: Simulation results: State space [x_1, x_2] trajectory, for different r values

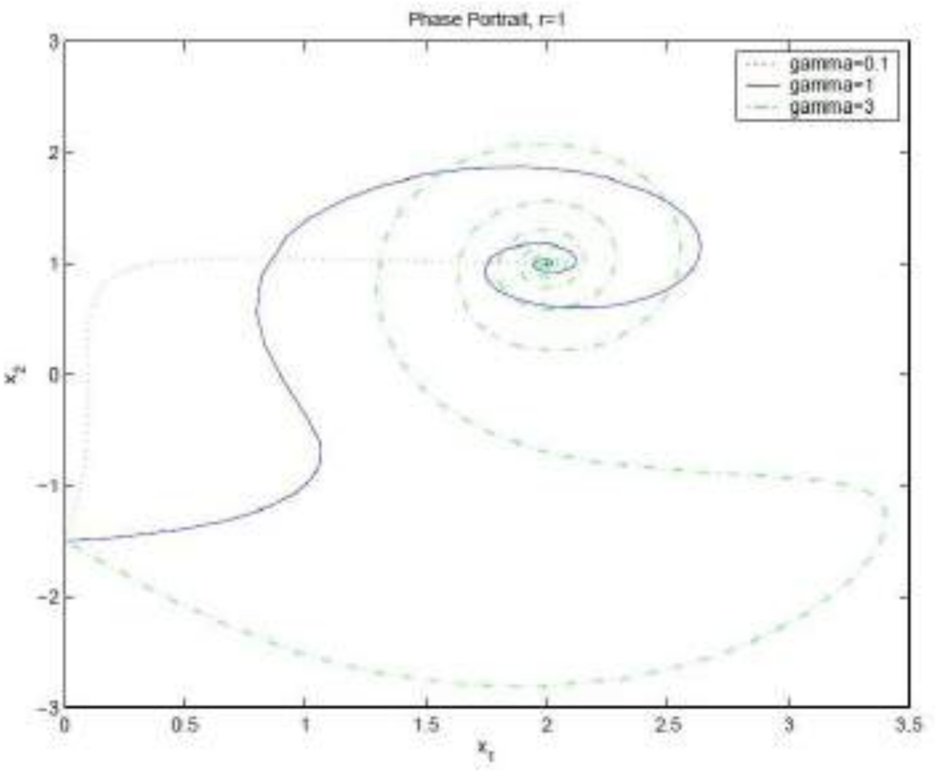


Figure 12: Simulation results: State space [x_1, x_2] trajectory, for different γ values

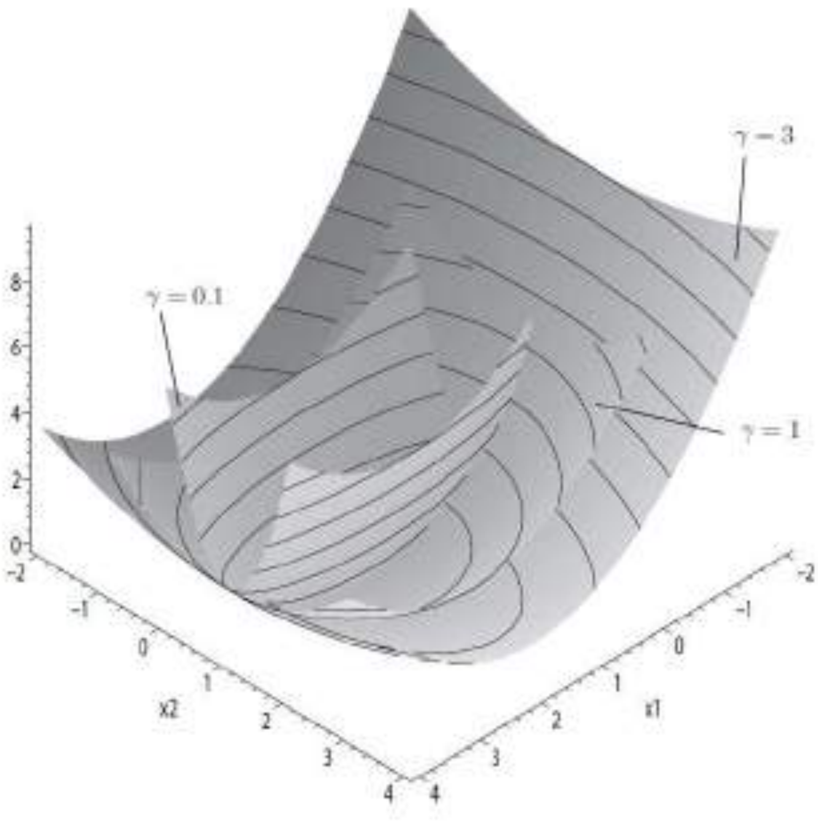


Figure 13: Desired Hamiltonian function H_d for different γ values

equation

$$f(x)+g(x)u=(J_d-R_d)\partial H_d. \tag{14}$$

With the motivation of enlarging the class of systems for which this equation is solvable we propose to avoid the decomposition of the control into energy–shaping and damping injection terms. Instead, we suggest to carry out *simultaneously* both stages and replace (14), with the SIDA–PBC matching equations

$$f(x)+g(x)u=F_d(x)\partial H_d, \tag{15}$$

and to replace the constraints

$$J_d^T(x)=-J_d(x), \quad R_d(x)=R_d(x)^T\geq 0 \tag{16}$$

by the *strictly weaker* condition

$$F_d(x)+F_d^T(x)\leq 0, \tag{17}$$

and define the control as

$$u=\left[g^T(x)g(x)\right]^{-1}g^T(x)(F_d(x)\partial H_d-f(x)). \tag{18}$$

Since the set of skew–symmetric matrices is strictly contained in the set of matrices with negative semi–definite symmetric part, it is clear that the set of functions $\{f(x), g(x)\}$ for which (14) (subject to the constraint (16)) is solvable is strictly smaller than the set for which (15), subject to (17), is solvable.

Remark 4. Similarly to IDA–PBC, application of SIDA–PBC also yields a closed–loop PCH system of the form (4) with

$$J_d(x,t)=\frac{1}{2}\left[F_d(x,t)-F_d^T(x,t)\right], \quad R_d(x,t)=\frac{1}{2}\left[F_d(x,t)+F_d^T(x,t)\right].$$

The SIDA-PBC can be summarized in the following Proposition.

Proposition 3. A dynamical system in an affine the form

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

with the control law (18)

$$u=\left[g^T(x)g(x)\right]^{-1}g^T(x)(F_d(x)\partial H_d-f(x)),$$

is asymptotically stable to x^* *iff*

$$x^*=\arg\min H_d$$

and

$$F_d(x)+F_d(x)^T\leq 0.$$

Example 7. (A toy model) Now we apply this technique to the toy model described before. We have to solve the new matching equation (15), which implies the control law (18). The model (11) can be written in the form $\dot{x}=f(x)+g(x)u$ with

$$f(x)=\begin{bmatrix}-x_1+\xi x_2^2\\ -x_1x_2\end{bmatrix},\ g=\begin{bmatrix}0\\1\end{bmatrix}.$$

Splitting the F_d matrix as

$$F_d = \begin{bmatrix} F_{11} & F_{12} \\ F_{21} & F_{22} \end{bmatrix},$$

the control law (18) has the following form

$$u = F_{21}\partial_{x1}H_d + F_{22}\partial_{x2}H_d + x_1x_2$$

where F_{21} and F_{22} are free parameters satisfying (17) and $x^* =$ arg min $H_d(x)$. Notice that we have more degrees of freedom than in the conventional IDA-PBC technique.

In this case the more evident choice is to take a quadratic energy function, for example

$$H_d = \frac{1}{2}(x_1 - x_1^*)^2 + \frac{\xi}{2}(x_2 - x_2^*)^2$$

which implies

$$\partial_{x1}H_d = x_1 - x_1^*, \text{and } \partial_{x2}H_d = \xi(x_2 - x_2^*).$$

Setting $F_{21} = -x_2$ and $F_{22} = -\frac{k}{\xi}$, the control law yields

$$u = x_1^* x_2 - k(x_2 - x_2^*). \qquad (19)$$

The F_{11} and F_{12} are still free and must satisfy the matching equation for the x_1 dynamics,

$$-x_1 + \xi x_2^2 = F_{11}(x_1 - x_1^*) + F_{12}\xi(x_2 - x_2^*).$$

In order to simplify the calculations, we set $F_{11} = -1$, which implies

$$F_{22} = x_2 + x_2^*.$$

Finally, to prove stability we only have to be sure that the $F + F^T$ matrix is negative-semidefinite, *i.e.*

$$F + F^T = \begin{bmatrix} -2 & x_2^* \\ x_2^* & -\frac{k}{\xi} \end{bmatrix} \le 0$$

and, applying Schur's inequality,

$$k \ge \frac{1}{4}\xi x_2^{*2} = \frac{1}{4}x_1^*.$$

Figure 14 shows the simulation results of the control law (19). The parameter values are $\xi = 2$, $x_2^d = 2$ and $k = 1$. Notice that the system goes to the desired fixed point x^*.

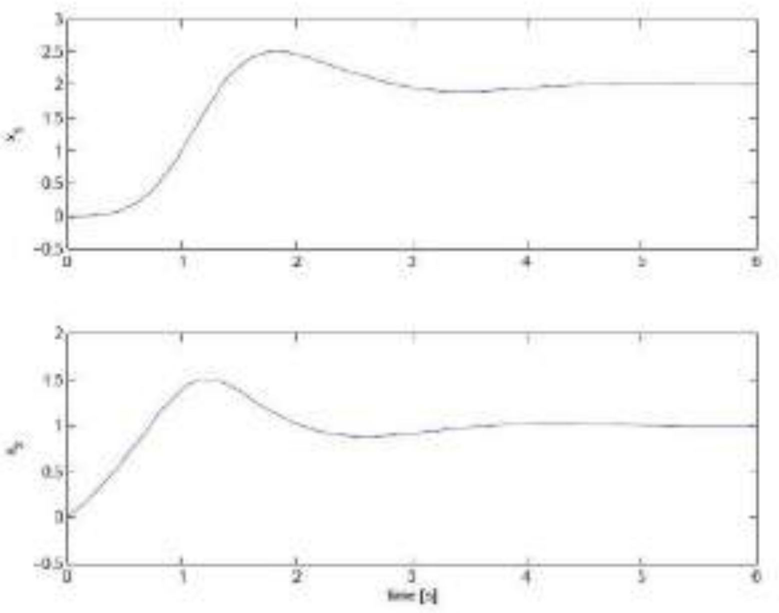


Figure 14: Simulation results: x_1 and x_2 for a SIDA-PBC controller

4 Improving the robustness of the IDA-PBC technique

One of the problems of the IDA-PBC technique for practical applications is the robustness of the designed controllers.

In this case, the robustness problem was partially solved adding an integral term to the error of the passive output. This dynamical extension partially solves the problem for relative degree one outputs but the main problem remains open for higher relative degree outputs. In this case the dynamical extension is not clear because, in general, it breaks the skew-symmetric property of the J_d matrix.

4.1 Adding an integral term

In this subsection we explain why the integral term can be used in a PCHS framework for relative degree one outputs, or in other words, passive outputs. To expose the basic idea, consider a fully actuated control system of the form

$$\begin{cases} \dot{x}_1 = f_1(x_1, x_2) \\ \dot{x}_2 = f_2(x_1, x_2) + g(x_1, x_2)u \end{cases} \tag{20}$$

where $x_1 \in R^n$, $x_2 \in R^m$ and $u \in R^m$, and g is full rank. Assume the IDA-PBC technique can be applied to (20) so that in closed-loop the system becomes

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} J_d - R_d \end{bmatrix} \begin{bmatrix} \partial_1 H_d \\ \partial_2 H_d \end{bmatrix}$$

with control law

$$u = g^{-1}((O_{m\times n} \quad I_{m\times m})(J_d - R_d)\partial H_d - f_2).$$

Under the stated assumptions, the x_2 are relative degree one outputs. We can easily add a dynamical extension to them by means of

$$\dot{z} = -a\partial_2 H_d, \tag{21}$$

where $a \in \mathbb{R}^{m\times m}$. The whole closed loop system can be rewritten in Hamiltonian form as

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{z} \end{bmatrix} = \begin{bmatrix} \begin{matrix} J_d & -R_d \\ 0 & -a \end{matrix} & \begin{matrix} 0 \\ a^T \\ 0 \end{matrix} \end{bmatrix} \begin{bmatrix} \partial_1 H_{dz} \\ \partial_2 H_{dz} \\ \partial_z H_{dz} \end{bmatrix}$$

with a new Hamiltonian function

$$H_{dz} = H_d + \frac{k}{2} z^T z.$$

The new controller is

$$v = u + g^{-1} k a^T z = u - g^{-1} k a^T a \int \partial_2 H_{dz}.$$

Notice that (21) forces $x_2 = x_2^*$ to remain a fixed point of the extended system.

The same procedure, when applied to the higher relative degree output x1, requires a closed loop system of the form

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{z} \end{bmatrix} = \begin{bmatrix} \begin{matrix} J_d & -R_d \\ -a & 0 \end{matrix} & \begin{matrix} 0 \\ b \\ 0 \end{matrix} \end{bmatrix} \begin{bmatrix} \partial_1 H_{dz} \\ \partial_2 H_{dz} \\ \partial_z H_{dz} \end{bmatrix}$$

where now $a \in \mathbb{R}^{n\times n}$ and $b \in \mathbb{R}^{m\times n}$. The a term is used to force the equilibrium point x_1^* of the output, while b is necessary to put the integral action into the control law. In this case stability cannot be proved using the PCHS properties, since the a, b terms break the semi-definite positiveness of the dissipation matrix:

$$R_{dz} = \begin{bmatrix} \begin{matrix} & R_d & \\ a/2 & & -b^T/2 \end{matrix} & \begin{matrix} a^T/2 \\ -b/2 \\ 0 \end{matrix} \end{bmatrix}.$$

Indeed, consider a matrix of the form

$$M = \begin{bmatrix} A & B^T \\ B & D \end{bmatrix}.$$

A simple application of Schur's complement shows that if $D = 0$, then $B \neq 0$ implies $M < 0$. In our case, this would mean $a = 0$ and $b = 0$, which makes no sense.

4.2 Influence of unknown parameters on the PCHS structure

In this subsection we point out the kind of problems that can appear in the closed-loop structure obtained by IDA-PBC methods for relative degree one outputs, when nominal values are used in a system with uncertain parameters.

Although the IDA-PBC method has some built-in robustness coming from its PCHS structure, the use of a nominal u for systems with uncertain parameters can give a closed loop system which is not exactly PCHS. One may thing that for nominal parameters in a small neighborhood of the actual ones, the "*J*—*R*" structure will not be destroyed; however, we will see that the resulting closed-loop system has interconnection and dissipation matrices depending on the state of the system, even if the closed-loop system for the actual parameter values does not; this has as a consequence that the effect of small parameter changes is not uniform in state space and, in particular, is unbounded in a neighborhood of the desired regulation point. In addition to this, the closed-loop system obtained with a nominal control does not have, in general, x^* as a fixed point. As is well known from elemental control theory, this last problem can be corrected by adding control terms proportional to the integral of the error. Integral control has been discussed in the PCHS setting in the previous subsection 4.1, where it is shown that adding as state variable the integral of the natural passive output of the closed-loop system yields a system which is again PCHS.

Consider the dynamical system (20) of subsection 4.1,

$$\begin{cases} \dot{x}_1 = f_1(x_1, x_2) \\ \dot{x}_2 = f_2(x_1, x_2) + g(x_1, x_2)u \end{cases} \tag{22}$$

where $x_1 \in R^n$, $x_2 \in R^m$, $u \in R^m$ and det $g \neq 0$, so that the x_2 are relative degree one outputs which we want to regulate to desired values x_2^*. Given x_2^*, the fixed point values of x_1 and u are obtained by equaling to zero the right-hand sides of (22).

Applying the IDA-PBC technique, we match the system to the desired partitioned PCHS

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} J_{d11} - R_{d11} & -J_{d21}^T - R_{d21}^T \\ J_{d21} - R_{d21} & J_{d22} - R_{d22} \end{bmatrix} \begin{bmatrix} \partial_1 H_d \\ \partial_2 H_d \end{bmatrix},$$

where each $J_{d}\cdot$ and $R_{d}\cdot$ represents the interconnection and dissipative terms of the J_d and R_d matrices, respectively. This implies that J_{d11} and J_{d22} must be skew-symmetric and similarly $R_{d11} = R^T_{d11} \geq 0$ and $R_{d22} = R^T_{d22} \geq 0$. Hence, the desired interconnection and damping matrices are

$$J_d = \begin{bmatrix} J_{d11} & -J_{d21}^T \\ J_{d21} & J_{d22} \end{bmatrix}, R_d = \begin{bmatrix} R_{d11} & R_{d21}^T \\ R_{d21} & R_{d22} \end{bmatrix}.$$

Equaling the first x_1 rows of both systems yields the IDA-PBC matching equation

$$f_1 = (J_{d11} - R_{d11})\partial_1 H_d + (-J_{d21}^T - R_{d21}^T)\partial_2 H_d. \tag{23}$$

Assume that this equation can be solved, giving J_d, R_d and H_d satisfying the proper structural and control objective conditions. Substituting then into the equation coming from the last x_2 rows, one gets the feedback control

$$u = g^{-1}\left[(J_{d21} - R_{d21})\partial_1 H_d + (J_{d22}^T - R_{d22}^T)\partial_2 H_d - f_2 \right].$$

Assume now that the system (22) depends on some uncertain constant parameters ξ, for which we assume nominal values $\hat{\xi}$. The unknown parameters creep into the formalism through f_i (and f_u), making the solution to the matching equation (23) depend on them, and also through the desired values x^*_1, which appear in H_d and which may depend on ξ due to the fact that they must obey $f_1(x^*_1, x^*_2) = 0$. Hence, the nominal control is given by

$$\hat{u} = \hat{g}^{-1}\left[(\hat{J}_{d21} - \hat{R}_{d21})\partial_1 \hat{H}_d + (\hat{J}_{d22} - \hat{R}_{d22})\partial_2 \hat{H}_d - \hat{f}_2 \right].$$

The closed-loop system computed with the nominal control is

$$\dot{x}_1 = (J_{d11} - R_{d11})\partial_1 H_d + (-J_{d21}^T - R_{d21}^T)\partial_2 H_d, \tag{24}$$

$$\dot{x}_2 = f_2 - g\hat{g}^{-1}\hat{f}_2 + g\hat{g}^{-1}\left[(\hat{J}_{d21} - \hat{R}_{d21})\partial_1 \hat{H}_d + (\hat{J}_{d22} - \hat{R}_{d22})\partial_2 \hat{H}_d \right].$$

In the equation for x_1, (24), we can change H_d by $\hat{H}_d$ and put the balance terms into δ_1; denoting $\delta_2 = f_2 - g\hat{g}^{-1}\hat{f}_2$, , we get a system of the form

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} \begin{bmatrix} \partial_1 \hat{H}_d \\ \partial_2 \hat{H}_d \end{bmatrix} + \begin{bmatrix} \delta_1 \\ \delta_2 \end{bmatrix}.$$

The components of δ_1 can be made proportional to components of $\partial_2 \hat{H}_d$ by dividing by the corresponding factors; likewise, the components of δ_2 can be made proportional to components of $\partial_1 \hat{H}_d$ (one has a large amount of freedom in selecting the components of $\partial \hat{H}_d$ to which the extra terms are made proportional). After doing this, one gets

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} B_{11} & B_{12} + \tilde{B}_{12} \\ B_{21} + \tilde{B}_{21} & B_{22} \end{bmatrix} \begin{bmatrix} \partial_1 \hat{H}_d \\ \partial_2 \hat{H}_d \end{bmatrix} = \hat{A}_d \partial \hat{H}_d. \tag{25}$$

Notice that there are no singularities in the differential equations (25), since the singular terms in $\hat{A}_d$ are canceled by $\partial \hat{H}_d$.

Since any matrix can be decomposed into symmetric and skew-symmetric parts, we write

$$\hat{A}_d = \hat{J}_d - \hat{R}_d,\ \hat{J}_d^T = -\hat{J}_d,\ \hat{R}_d^T = -\hat{R}_d.$$

Due to the $\tilde{B}_{21}$ and $\tilde{B}_{12}$ terms, the corresponding elements of $\hat{J}_d$ and $\hat{R}_d$ will contain terms which are singular at $x_1 = \hat{x}_1^*$ or $\hat{x}_2 = x_2^*$. This is no formal problem for $\hat{J}_d$, but the presence of off-diagonal singular terms in $\hat{R}_d$ will destroy its positive semidefiniteness at least in a neighborhood of $(\hat{x}_1^*, x_2^*)$. Notice, however, that due to the presence of δ_1, δ_2 the closed-loop system has fixed points which differ from $(\hat{x}_1^*, x_2^*)$; if $\hat{R}_d$ is positive semidefinite in a neighborhood of the closed loop fixed points, LaSalle's theorem can still be invoked to proof local asymptotic stability, albeit not for the desired regulation point.

In order to ensure the regularization objective in presence of the unknown parameter, an integral term is introduced in basic control theory. For relative degree one outputs, this can be given a Hamiltonian form as well (see previous subsection). Keeping the unknown parameters assumption, we can rewrite the closed-loop system as follows. First of all, we write $u = \hat{u} + v$ in the original system. This yields

$$\dot{x} = (\hat{J}_d - \hat{R}_d)\partial_x \hat{H}_d + gv.$$

Because of $\partial_2 H_d|_{x_2 = x_2^*} = 0,$ we can enlarge the state space with $z \in R^u$ so that

$$\dot{z} = -a\partial_2 H_d = -a\partial_2 \hat{H}_d,$$

with $a = a^T \in R^{m\times m}$ also diagonal and positive definite. The closed-loop enlarged system can be written as

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{z} \end{bmatrix} = \begin{bmatrix} \hat{J}_d & -\hat{R}_d & \begin{matrix} 0 \\ a \\ 0 \end{matrix} \\ 0 & -a^T & \end{bmatrix} \partial \hat{H}_{dz},$$

where

$$\hat{H}_{dz} = \hat{H}_d + \frac{1}{2} z^T z.$$

As discussed in subsection 4.1, due to the equation for z˙, the only fixed points of the new closed-loop system are those with $x_2 = \dot{x}_2$. The equation for $\dot{x}_1$ determines then x^*_1 in terms of $\dot{x}_2$ and the actual parameter values; finally, the equation for $\dot{x}_2$ sets the equilibrium value of z, z^*, in terms of the nominal parameter values. However,

$$R_{dz} = \begin{bmatrix} & \hat{R}_d & & 0 \\ & & & 0 \\ 0 & & 0 & 0 \end{bmatrix}$$

has the same singularity problems that $\hat{R}_d$ in a neighborhood of x_u^*, and a proof of stability based on LaSalle's theorem cannot be given. Nevertheless, we will present an example in the next Section where the desired regulation point seems to be asymptotically stable.

Example 8. (A toy model again). To illustrate the quite general previous remarks, consider the toy model studied in subsection 3.1, equation (11),

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

where $\xi > 0$ is an uncertain parameter. The control objective is to regulate x_2 to a desired value x^d_2. The IDA-PBC control law obtained was (13),

$$u = x_1 x_2 - \gamma\xi(x_1 - x_1^*)(x_2 + x_2^d) - \frac{r}{\gamma}(x_2 - x_2^d).$$

This control law yields a closed-loop system which is Hamiltonian with (J_d,R_d,H_d), and which has (x_1^*, x_2^d) as a globally asymptotically stable equilibrium point. However, if we use an estimated value $\hat{\xi}$ of the uncertain parameter **ξ**, the feedback control is

$$\hat{u} = x_1 x_2 - \gamma\hat{\xi}(x_1 - \hat{x}_1^*)(x_2 + x_2^d) - \frac{r}{\gamma}(x_2 - x_2^d),$$

where

$$\hat{x}_1^* = \hat{\xi}(x_2^d)^2 = \frac{\hat{\xi}}{\xi} x_1^*.$$

For later convenience, we also define

$$\hat{\alpha} = \gamma\hat{\xi}(x_2 + x_2^d).$$

Using this $\hat{u}$, the closed-loop system equation for $\dot{x}_2$ is

$$\begin{aligned}\dot{x}_2 &= -\gamma\hat{\xi}(x_1-\hat{x}_1^*)(x_2+x_2^d)-\frac{r}{\gamma}(x_2-x_2^d)\\ &= -\hat{\alpha}(x_1-\hat{x}_1^*)-r\frac{1}{\gamma}(x_2-x_2^d)\\ &= -\hat{\alpha}\partial_1\hat{H}_d - r\partial_2\hat{H}_d,\end{aligned}$$

where

$$\hat{H}_d = \frac{1}{2}(x_1-\hat{x}_1^*)^2+\frac{1}{2\gamma}(x_2-x_2^d)^2.$$

The equation for $\dot{x}_1$ is not changed by the feedback, but can be rewritten as

$$\begin{aligned}\dot{x}_1 &= -x_1+\xi x_2^2\\ &= -(x_1-\hat{x}_1^*)-\hat{x}_1^*+\xi x_2^2+(\xi-\hat{\xi})x_2^2\\ &= -\partial_1\hat{H}_d+\hat{\xi}(x_2+x_2^d)(x_2-x_2^d)+(\xi-\hat{\xi})x_2^2\\ &= -\partial_1\hat{H}_d+\hat{\alpha}\frac{1}{\gamma}(x_2-x_2^d)+(\xi-\hat{\xi})x_2^2\\ &= -\partial_1\hat{H}_d+\hat{\alpha}\partial_2\hat{H}_d+(\xi-\hat{\xi})x_2^2.\end{aligned}$$

These two equations can be cast into Hamiltonian form as

$$\begin{aligned}\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} &= \begin{bmatrix}-1 & \hat{\alpha}+\gamma(\xi-\hat{\xi})\dfrac{x_2^2}{x_2-x_2^d}\\ -\hat{\alpha} & -r\end{bmatrix}\begin{bmatrix}\partial_1\hat{H}_d\\ \partial_2\hat{H}_d\end{bmatrix}\\ &= \hat{A}_d\partial\hat{H}_d = (\hat{J}_d-\hat{R}_d)\partial\hat{H}_d,\end{aligned}$$

where $\hat{J}_d$ is the skew-symmetric part, giving the closed-loop interconnection matrix, and

$$\hat{R}_d = -\frac{1}{2}(\hat{A}_d+\hat{A}_d^T) = \begin{bmatrix}1 & -\dfrac{\gamma}{2}\dfrac{(\xi-\hat{\xi})x_2^2}{x_2-x_2^d}\\ -\dfrac{\gamma}{2}\dfrac{(\xi-\hat{\xi})x_2^2}{x_2-x_2^d} & r\end{bmatrix}.$$

One has

$$tr\hat{R}_d = 1+r>0,$$

$$\det\hat{R}_d = r-\frac{\gamma^2}{4}\frac{(\xi-\hat{\xi})^2}{(x_2-x_2^d)^2}x_2^4.$$

Hence, in order to ensure that $\hat{R}_d \geq 0$, it is necessary that

$$\frac{(x_2-x_2^d)^2}{x_2^4} \geq \frac{\gamma^2(\xi-\hat{\xi})^2}{4r}, \tag{26}$$

which is globally true if $\hat{\xi} = \xi$ but fails in a neighborhood of $x_2 = x_2^d$, as well as for $|x_2|$ large enough, if $\xi \neq \hat{\xi}$.

Notice that, for $\hat{\xi} \neq \xi$, the closed-loop system does not have $x_2 = x_2$, $x_1 = \hat{x}_1$ as a fixed point, even though these are critical points of $\hat{H}_d$, due to the $1/\left(x_2 - x^d{}_2\right)$ term in $\hat{A}_d$. In

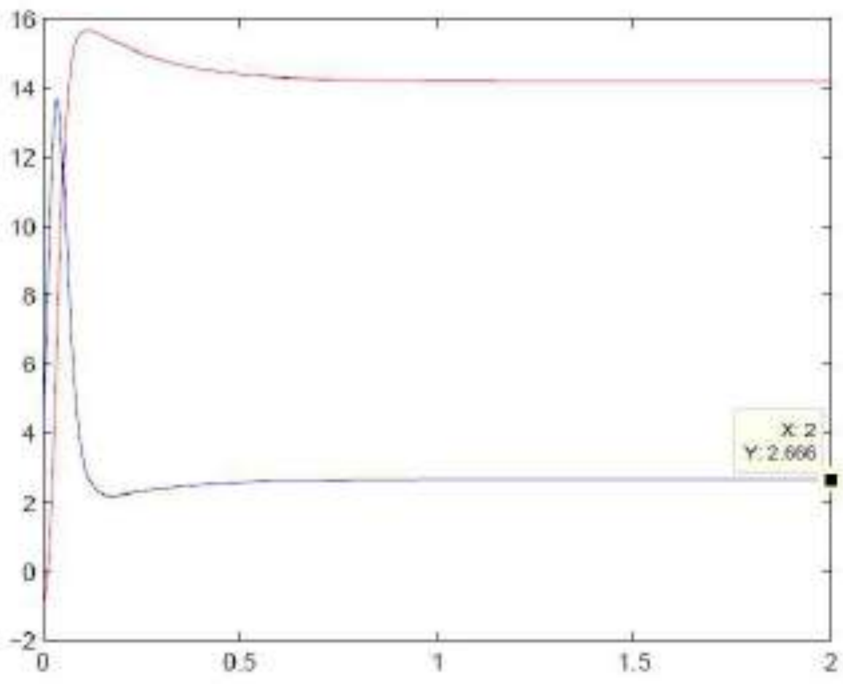


Figure 15: Simulation results: IDA-PBC controller for a toy model

general, due to the state dependence of $\hat{A}_d$, other solutions may appear anyway. In fact, computing the fixed points yields the relation (depending only on the actual value of ξ)

$$x_1 = \xi x_2^2,$$

while the value of x_2 comes from the solutions to

$$0 = \gamma^2 \hat{\xi}(\xi x_2^2 - \hat{x}_1^*)(x_2 - x_2^2) + r(x_2 - x_2^d).$$

If $\hat{\xi} = \xi$, one gets

$$\gamma^2 \xi^2 (x_2^2 - (x_2^2)^2)(x_2 + x_2^d) + r(x_2 - x_2^d) = 0$$

which only has a real solution, namely $x_2 = x^d{}_2$. For $\xi \neq \hat{\xi}$ one has, in general, three solutions, at least one of them real, all different from $x^d{}_2$.

Figure 15 shows a simulation of the controller. The asymptotic value of x_2 is ~ 2.666 instead of $x^d{}_2 = 2$, while x_1 goes to ξ × (2.666)², as expected. As discussed in this Chapter, local asymptotic stability can be proved using LaSalle's theorem, but extensive simulations with very broad initial conditions seem to indicate that the stability is in fact global.

Following the general theory, an integral term is introduced next, so that the equation for x_2 gets modified by an az term while the dynamics of z is

$$\dot{z} = -a\frac{1}{\gamma}(x_2 + x_2^d).$$

All the fixed points of the closed-loop system have $x_2 = x^d{}_2$; from the equation for $\dot{x}_1$, one gets again $x_1 = x*_1 = \xi\left(x^d{}_2\right)^2$. Finally, the equation for $\dot{x}_2$ determines now the fixed point value of z, z^*, which depends on the nominal value $\hat{\xi}$, instead of determining the fixed point for x_2.

Figure 16 shows a simulation of the new controller, for the same parameter values than the simulation for the old controller and $a = 50$. The variable z, the integral of the error in x_2, starts from zero an goes asymptotically to z^*. A longer transitory appears, as is characteristic of integral controllers. Simulations with initial values in a wide range of points, seem to point to the global stability of the closed loop system.

However, if r is decreased oscillations do appear. For instance, for $r = 20$ and the same values of all the other parameters, one gets the response displayed in Figure 17. The disappearance of the oscillations when r is increased corresponds to a (reversed) Hopf bifurcation. In fact, linearizing the closed loop system around $(\xi\left(x^d{}_2\right)^2, x^d{}_2, z^*)$ yields asystem which is asymptotically stable as long as

$$\frac{r}{\gamma} + \gamma\hat{\xi}(x_2^d)^2(\xi - \hat{\xi}) - (\xi - \gamma\hat{\xi})^2(x_2^d)^2 > 0,$$

which is true for r sufficiently large. Numerical simulations seem to imply that the fixed point of the nonlinear system is globally asymptotically stable. Computing the time derivative of $\hat{H}_{dz}$,

$$\frac{d}{dt}\hat{H}_{dz} = -(x_1 - \hat{x}_1^*)^2 + (\xi - \hat{\xi})x_2^2(x_1 - \hat{x}_1^*) - \frac{r}{\gamma^2}(x_2 - x_2^d)^2, \tag{27}$$

it can be seen that the region where (27) is nonpositive is much larger than what is implied by (26), due to the state-space dependence of the closed-loop dissipation matrix; in fact, for r large enough, the nonpositive region is pushed away from the desired regulation point, except for a bounded shrinking region whose boundary contains the later and which contains most of the periodic orbit. Although the details are quite particular to this example, we hope to obtain some insight into any existing mechanism which could be generalized.

4.3 Robust control via structure modification

As discussed in the previous Section, it is not clear how to generalize the integral extension for higher relative degree outputs in the PCHS framework. We present here a different approach, which can be applied to a larger class of systems. Examples include the DC motor, the electrical part of a doubly-fed induction machine or the buck power converter.

Consider a dynamical system of the form

$$\begin{cases} \dot{x}_o = f_0(x_o, x_u, \xi) \\ \dot{x}_u = f_u(x_o, x_u) + g(x_o, x_u)u \end{cases} \tag{28}$$

where $x_o \in \mathrm{R}^o$ are higher order relative degree outputs, $x_u \in \mathrm{R}^u$, $u \in \mathrm{R}^p$ are the controls and ξ is an uncertain parameter. To simplify the presentation we consider $p = u = o$ and that g is full rank.

As a control target we fix a desired $x^d{}_o$, which implies that the fixed point value of x_u is given by the following equation

$$f_0(x_o^d, x_u^*, \xi) = 0, \tag{29}$$

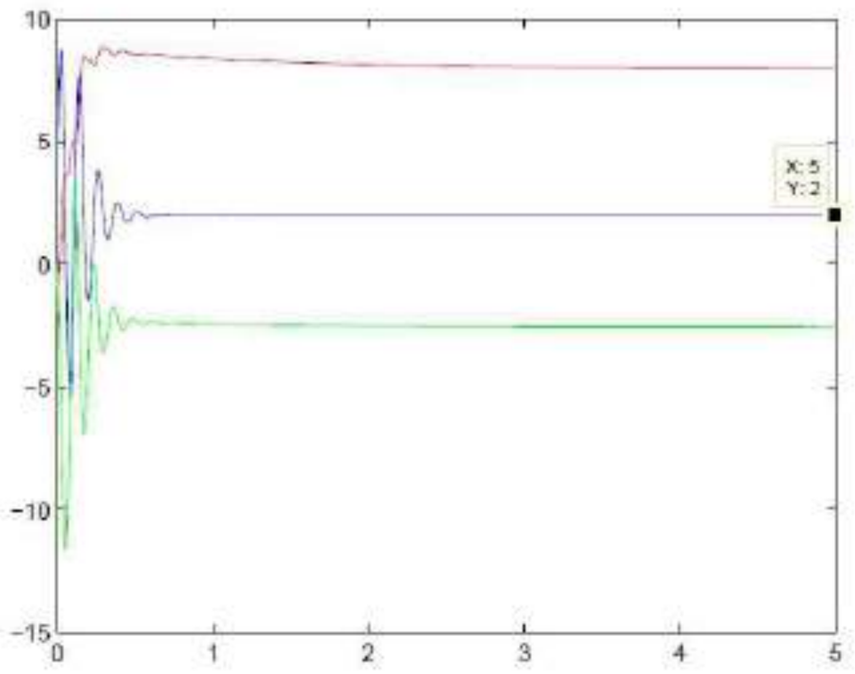


Figure 16: Simulation results: IDA-PBC+integral controller (with $r = 50$) for a toy model

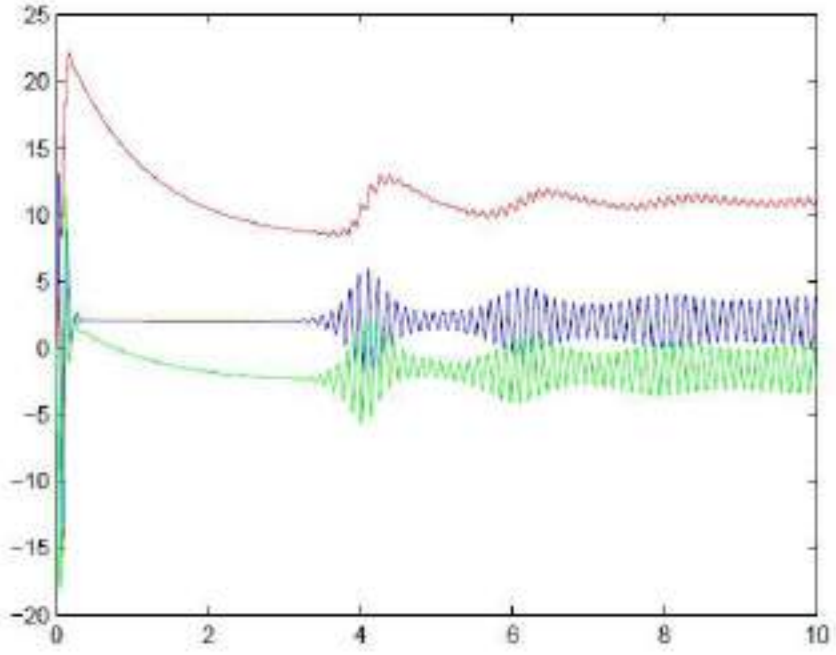


Figure 17: Simulation results: IDA-PBC+integral controller (with $r = 20$) for a toy model

and depends thus on the uncertain parameter ξ.

Applying the IDA-PBC technique, we match the system to the desired port Hamiltonian structure, where $(J_d - R_d)$ is partitioned as

$$\begin{bmatrix} \dot{x}_o \\ \dot{x}_u \end{bmatrix} = \begin{bmatrix} J_{doo} - R_{doo} & -J_{doo}^T - R_{doo}^T \\ J_{duo} - R_{duo} & J_{duu} - R_{duu} \end{bmatrix} \begin{bmatrix} \partial_o H_d \\ \partial_u H_d \end{bmatrix}.$$

Each $J_{d\cdot\cdot}$ and $R_{d\cdot\cdot}$ represents the interconnection and dissipative terms of the J_d and R_d matrices, respectively. This implies that J_{doo} and J_{duu} must be skew-symmetric and similarly $R_{doo} = R^T_{doo} \geq 0$ and $R_{duu} = R^T_{duu} \geq 0$. Hence, the desired interconnection and damping matrices are

$$J_d = \begin{bmatrix} J_{doo} & -J_{doo}^T \\ J_{duo} & J_{duu} \end{bmatrix}, R_d = \begin{bmatrix} R_{doo} & -R_{doo}^T \\ R_{duo} & R_{duu} \end{bmatrix}.$$

Notice that we need a H_d such that $\partial H_d\big|_{x=x^*} = 0$ to obtain an equilibrium point in $x^* = (x_o^d, x_u^*)$. Equaling the u rows of the IDA-PBC the control lawyields

$$u = g^{-1}\left[(J_{duo} - R_{duo})\partial_o H_d + (J_{duu} - R_{duu})\partial_u H_d - f_u\right].$$

Since H_d is a free function, it is chosen so that $\partial_o H_d$ does not depend on ξ (Notice that x_u^* depends on it, equation (29)). In the same way, ξ can appear in $\partial_u H_d$ through x_u^*, which can be removed from the control law setting

$$J_{duu} - R_{duu} = 0,$$

and the robustified IDA-PBC control law is

$$u = g^{-1}\left[(J_{duo} - R_{duo})\partial_o H_d - f_u\right].$$

As we set $R_{duu} = 0$, again Schur's complement shows that in order to keep the semipositiveness of R_d, we are forced to $R_{duo} = 0$, and consequently

$$u = g^{-1}\left[(J_{duo}\partial_o H_d - f_u\right]. \qquad (30)$$

From the o rows of the IDA-PBC, the following equation must be satisfied, were we fixed $R_{duo} = 0$,

$$f_o = (J_{doo} - R_{doo})\partial_o H_d - J_{duo}^T \partial_u H_d. \qquad (31)$$

Selecting J_{duo} full rank,

$$\partial_u H_d = -(J_{duo}^T)^{-1}\left[f_o - (J_{doo} - R_{doo})\partial_o H_d\right]. \qquad (32)$$

Rewriting f_o as

$$f_o = A(x)\partial_o H_d + B(x_u),$$

and choosing $(J_{doo} - R_{doo})$ so that

$$A(x) = J_{doo} - R_{doo},$$

the PDE (32) simplifies to

$$\partial_o H_d = -(J_{duo}^T)^{-1} B(x_u).$$

Notice that J_{duo} must be a function of x_u only, $J_{duo} = J_{duo}(xu)$. Fixing a part of the Hamiltonian and then finding the rest of H_d solving the PDE was also proposed. Stability can be discussed, using LaSalle's theorem. Dissipativity is assured if

$$R_{doo} = R_{doo}^T > 0.$$

This is equivalent to

$$A(x) + A(x)^T < 0.$$

Notice that this condition depends only on f_o, irrespectively of u. Convergence to the equilibrium point, defined by $\partial^2 H_d \big|_{x=x^*} = 0$, follows from the condition

$$\partial^2 H_d \big|_{x=x^*} > 0,$$

or, in other words,

$$\partial_u (-(J_{duo}^T)^{-1} B(x_u))\big|_{x=x^*} > 0.$$

We can summarize this Section in the following Proposition.

Proposition 4. Consider a dynamical system given by (28), so that f_o can be expressed as,

$$f_o = A(x)\partial_o H_d + B(x_u) \qquad (33)$$

where $\partial_o H_d$ is a design function of x_o such that

$$\partial_o H_d(x_o)\big|_{x_o = x_o^d} = 0$$

and

$$\partial_o^2 H_d(x_o)\big|_{x_o = x_o^d} > 0. \qquad (34)$$

Then the control law

$$u = g^{-1}\left[J_{duo}(x_u)\partial_o H_d - f_u \right], \qquad (35)$$

where $J_{duo}(xu)$ is another design function of x_u, is robustly stable in front of variations of ξ as long as

$$A(x) + A(x)^T < 0, \qquad (36)$$

$$(-(J_{duo}^T)^{-1} B(x_u))\big|_{x=x^*} = 0, \qquad (37)$$

and

$$\partial_u (-(J_{duo}^T)^{-1} B(x_u))\big|_{x=x^*} > 0. \qquad (38)$$

Notice that condition (36) implies that the dynamics of the output variables x_o is dissipative, and this is the only dissipation of the closed-loop system (due to $R_{duu} = R_{duo} = 0$).

Proposition 5. (The toy model). Consider once more the toy model studied in subsection 3.1, equation (11), where $\xi > 0$ is an uncertain parameter. In this case, differing from the previous subsection, the desired output is fixed by $x^d{}_1$. Notice that x1 is now a relative degree two output, and theintegral term discussion is not clear.

Applying the classical IDA-PBC method to the system, the following feedback control law is obtained

$$u = x_1x_2 - \gamma\xi(x_1 - x_1^d)(x_2 + x_2^*) - \frac{r}{\gamma}(x_2 - x_2^*),$$

where r > 0, γ > 0 are control parameters. Notice that the control law *u* depends on x^d_1 and x^*_2, where x^*_2 is function of ξ,

$$x_2^* = \sqrt{\frac{1}{\xi}x_1^d}.$$

In this case the control law is not robust with respect to an uncertain $\hat{\xi}$.

Let us calculate a new controller following the previous discussion. In this case the x_o output variable is x_1 and the x_u variable is x_2. First we fix $\partial_o H_d$ as

$$\partial_o H_d = x_1 - x_1^d$$

which ensures conditions (34) and (35). Then from (33), A(*x*) and B(x_u) must be

$$A(x) = -1\ B(x_u) = \xi(x_2^2 - x_2^{*2}).$$

Notice that condition (36) is achieved.

The easiest choice of J_{duo} is a free constant, for instance *k* > 0, but for this nonlinear example it is necessary to add a more complicated a(x_2) function.

The *a*(*x*) function is included to avoid stability restrictions on the space-state. The same procedure with *a* = 1 implies

$$\partial^2 H_d\big|_{x=x^*} = \begin{bmatrix} 1 & 0 \\ 0 & 2\frac{\xi}{k}x_2^* \end{bmatrix}$$

which is negative for x*$_2$ < 0. Consequently, the globally asymptotically stability is not achieved.

Then the final choice is

$$J_{duo} = J_{duo}^T = -a(x_2)k$$

with *k* > 0 and

$$a(x) = \begin{cases} 1, & x_2 > 0 \\ b, & x_2 = 0 \\ -1, & x_2 < 0 \end{cases} \tag{39}$$

where b ∈ [−1, 1] is a parameter that would be used to choose the equilibrium point of x_2 (see discussion on the closed-loop dynamics at the end of the example). This selection ensures conditions (37)

$$\frac{1}{a(x_2)k}\xi(x_2^2 - x_2^{*2})\bigg|_{x_2=x_2^*} = 0$$

and (38),

$$\left(\frac{1}{a(x_2)k}\xi(x_2^2 - x_2^{*2})\right)\bigg|_{x_2=x_2^*} > 0$$

Finally, the controller is obtained from (3.30) yields

$$u = x_1 x_2 - ak(x_1 - x_1^d). \tag{40}$$

The example seen as a classic IDA-PBC design. This design can be also obtained following the traditional IDA-PBC method in order to show the Hamiltonian structure of the closed–loop system. Consider the matching equation of the system (11) with the PCHS dynamics (4) where

$$J_d = \begin{bmatrix} 0 & a(x_2)k \\ -a(x_2)k & 0 \end{bmatrix}, R_d = \begin{bmatrix} r_c & 0 \\ 0 & 0 \end{bmatrix}$$

where $k > 0$, $a(x_2)$ is described in (39) and a Hamiltonian function such that

$$\partial_x H_d = \begin{bmatrix} x_1 - x_1^d \\ \partial_{x2} H_d \end{bmatrix}. \tag{41}$$

From the second row of the matching equation we obtain the same robust control law as (40)

$$u = x_1 x_2 - ak(x_1 - x_1^d)$$

which does not depend on ξ. From the first row we must compute $\partial_{x_2} H_d$ and verify the stability properties of the closed loop system. The matching equation yields

$$-x_1 + \xi x_2^2 = -r_c(x_1 - x_1^d) + ak\partial_{x_2} H_d$$

and, using $r_c = 1$ and $x_1^d = \xi x_2^{*2}$,

$$\partial_{x_2} H_d = \frac{\xi}{ak}(x_2^2 - x_2^{*2}). \tag{42}$$

To show stability, with the desired structure (4) and $J_d = -J^T{}_d$, $R_d = R^T{}_d > 0$ we only need positiveness of the Hessian of H_d evaluated at x^*,

$$\partial_x^2 H_d\big|_{x=x^*} = \begin{bmatrix} 1 & 0 \\ 0 & 2\frac{\xi}{k}a(x_2^*)x_2^* \end{bmatrix},$$

which is true for all x^*_2, as long as $k > 0$ and ξ > 0. Notice that

$$\partial_x^2 H_d = \frac{\xi}{ak}(\theta(x_2)(x_2^2 - x_2^{*2}) + 2a(x_2)x_2),$$

where Θ(x_2) is the Heaviside function.

The Hamiltonian function can be found integrating $\partial_x H_d$ (equation (41) with (42))

$$H_d = a(x_2)\frac{\xi}{k}x_2\left(\frac{1}{3}x_2^2 - x_2^{*2}\right) + \frac{1}{2}(x_1 - x_1^d)^2,$$

which has two local minima, both with first coordinate $x^d{}_1$ .H_d is depicted in Figure 3.18 (using the same simulation parameters than for Figures 19 and 20). Notice that two equilibrium points appear, given by

$$x_2^* = \pm\sqrt{\frac{1}{\xi}x_1^d},$$

and these points yield the same value for x^d_1. In the classical controller, this ambiguity did not appear, basically because we were fixing the desired value of x^*_2 , while in the robust controller both values of x_2 are possible.

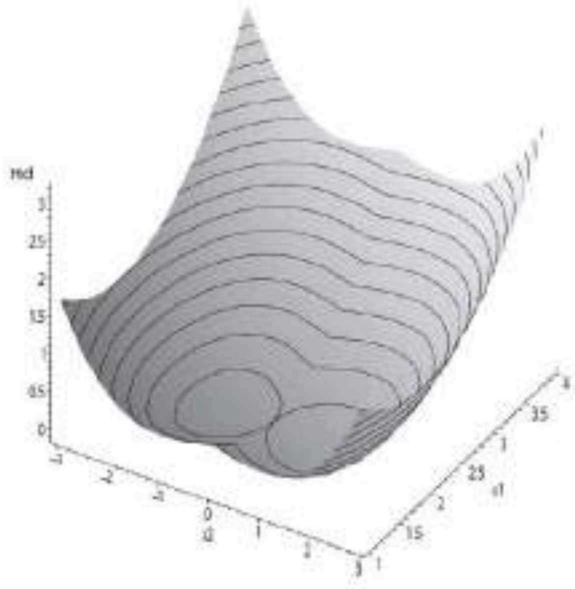


Figure 18: Desired Hamiltonian function, H_d

Simulations. Figures 19 and 20 show simulation results testing both controllers, the robust method presented above and the classic IDA-PBC. The parameters are $\xi = 2$, $\hat{\xi} = 1$, with initial conditions $x(0) = (0,-1.5)$, and the desired output is $x^d_1 = 2$. The control parameter for the robust control law is $k = 10$, while for the classical IDA-PBC $r = 1$ and $\gamma = 1$ are selected.

The robust controller achieves the desired value of x1 even with a wrong parameter estimation, while the classical IDA-PBC controller is sensible to the ξ variations. Notice that the variations on change the x^*_2 equilibrium point.

Study of the closed-loop dynamics. Now we focus on to study of the dynamical behavior of the controller designed above. Fig.21 shows the phase portrait of the closed–loop system (the values of the parameters are as above).

Two stable fixed points, $x^* = (2,\pm 1)$, are present. To select x^*_2 , let us to write the system (11) with the feedback control law (40),

$$\begin{cases} \dot{x}_1 = -x_1 + \xi x_2^2 \\ \dot{x}_2 = -ak(x_1 - x_1^d) \end{cases}$$

The dynamics after reaching $x_2 = 0$ there is described by

$$\dot{x}_1 = -x_1,$$

so x_1 tends to $x_1 = 0$, and simultaneously the x_2 dynamics is

$$\dot{x}_2 = -ak(x_1 - x_1^d)$$

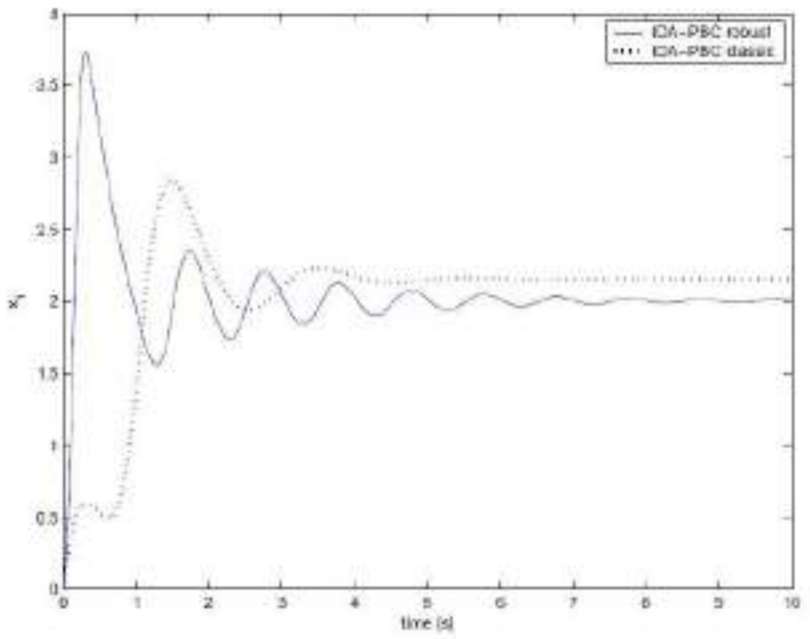


Figure 19: Comparison between the robust method and the classic IDA-PBC, behavior of x_1

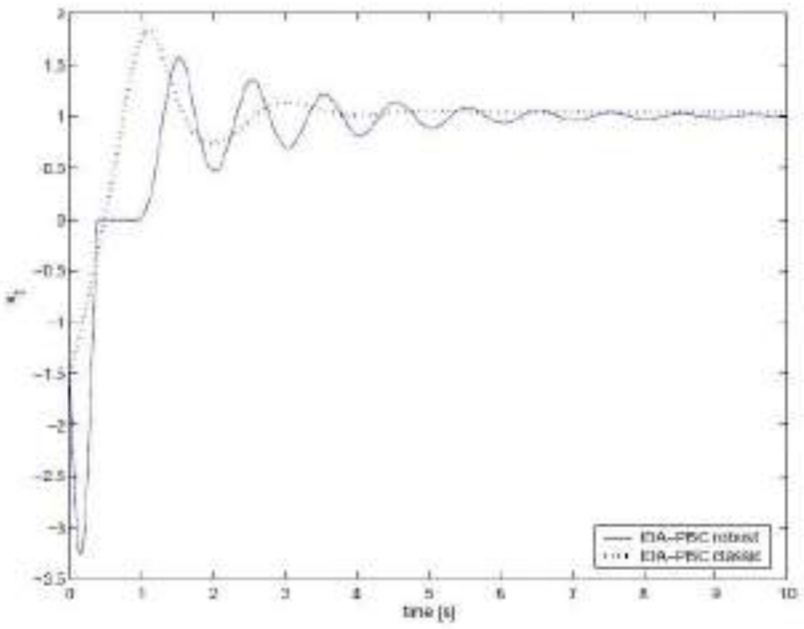


Figure 20: Comparison between the robust method and the classic IDA-PBC, behavior of x_2

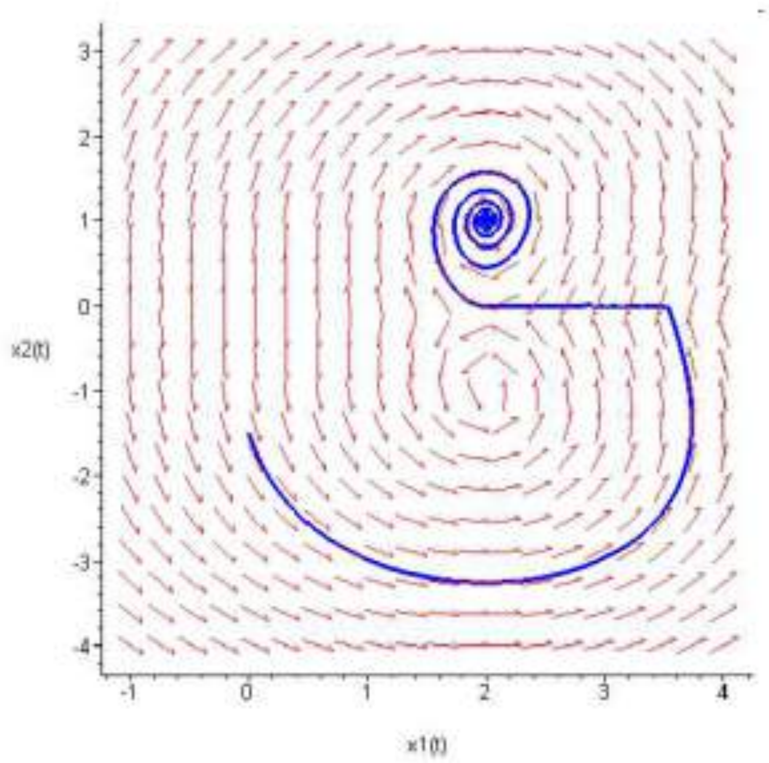


Figure 21: State space; trajectory and vector field

where $k > 0$ and $a(x_2) = b$ with $b \in [-1, 1]$. Notice that for $x_1 > x^d{}_1$, and sufficiently far of the equilibrium point, $x_2 = 0$ is an attractor set. Besides, for $x_1 < x^d{}_1$ and $x_2 = 0$ the dynamics of x_2 for $b = 1$ is increasing, while if $b = -1$ the dynamics of x_2 decreases. In other words, for $b = 1$

$$\lim_{t\to\infty} x_2 = +\sqrt{\frac{1}{\xi} x_1^d}$$

and for $b = -1$

$$\lim_{t\to\infty} x_2 = -\sqrt{\frac{1}{\xi} x_1^d}.$$

Figure 22 shows a phase portrait of two different simulations, for $b = 1$ with a continuous line and $b = -1$ with a dotted line. The behavior is as expected from the discussion above, for $b = 1$, x_2 tends to $+\sqrt{\frac{1}{\xi} x_1^d}$ while for b = −1 x_2 tends to $-\sqrt{\frac{1}{\xi} x_1^d}$. In Figure 23 the same simulations are depicted in function of time.

For numerical simulations, we modify $a(x)$ (39) as

$$a(x) = \begin{cases} 1, & x_2 > \varepsilon \\ b, & -\varepsilon < x_2 < \varepsilon \\ -1, & x_2 < \varepsilon \end{cases}$$

where $\epsilon > 0$ is a constant, so that numerical errors do not bring the trajectory to the wrong fixed point.

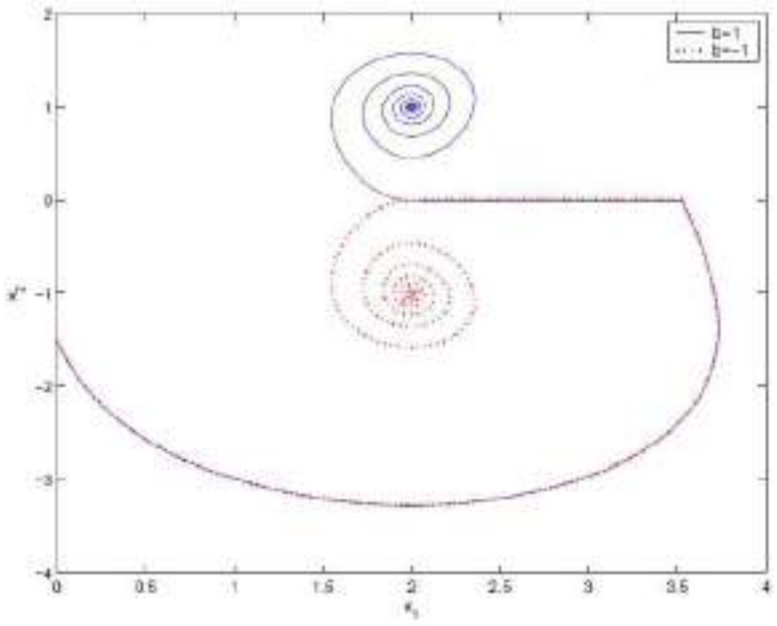


Figure 22: Phase portrait of x for two different b values. $b = 1$ with a continuous line and $b = -1$ with a dotted line

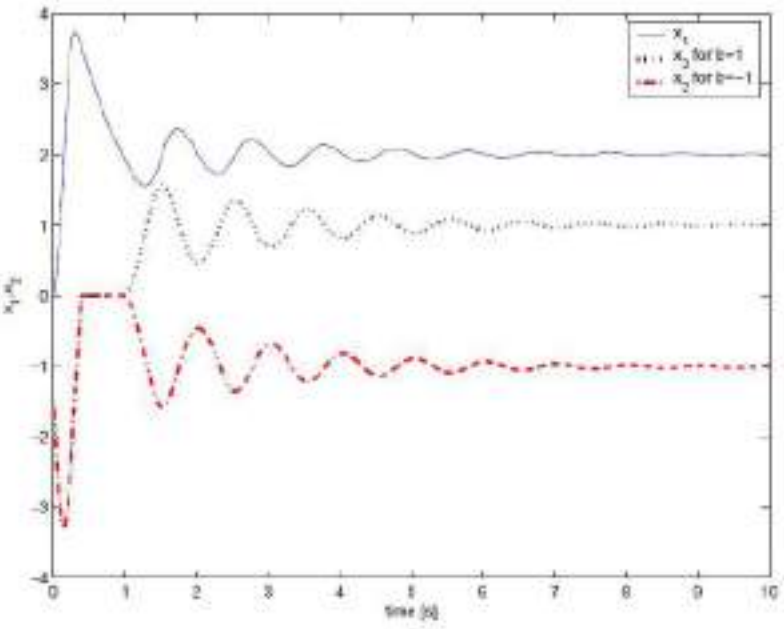


Figure 23: Simulations for two different b values

Example 9. (The DC motor). This robust IDA-PBC technique can also be applied to the DC motor speed control problem. Consider the DC motor described earlier, in PCHS form given by

$$\dot{x} = (J - R)\partial H(x) + g + g_u u$$

with $x \in \mathrm{R}^2$

$$x = [\lambda, p_m]^T$$

and where λ is the inductor flux and p_m is the angular momentum. The interconnection, dissipation and port matrices are

$$J = \begin{bmatrix} 0 & -K \\ K & 0 \end{bmatrix} \; R = \begin{bmatrix} r & 0 \\ 0 & B_r \end{bmatrix} \; g = \begin{bmatrix} 0 \\ -\tau_L \end{bmatrix} \; g_u = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

with the control input $u = v$, r and B_r represent the electrical and mechanical losses respectively, and the Hamiltonian function is given by

$$H(x) = \frac{1}{2L}\lambda^2 + \frac{1}{2J_m} p_m^2,$$

where L is the inductance and J_m the inertia of the motor. Assume that the control objective is a desired speed ω^d and in that the unknown parameter is the external torque τ_L.

Following the procedure described in Proposition 4, where the x_o (output) variable is themechanical speed ω and the x_u variable is the inductor current i, we choose

$$\partial_o H_d = \frac{1}{J_m}(p_m - p_m^d) = \omega - \omega^d, \tag{43}$$

which ensures (34). Now f_o from equation (33) can be written as

$$f_o = K_i - B_r\omega - \tau_L = A(x)(\omega - \omega^d) + B(x),$$

and into taking account $\tau_L = K_{i^*} - B_r\omega^d$, , $A(x)$ and $B(x)$ are given by

$$A = -B_r,$$
$$B = K(i - i^*),$$

which fulfill the conditions (36), (37) and (38), with

$$J_{duo} = \gamma > 0. \tag{44}$$

Finally, the control law is obtained from (35),

$$u = \gamma(\omega - \omega^d) - ri - K_\omega$$

with (43), (44) and

$$f_u = -ri - K_\omega.$$

Figure 24 shows the behaviour of the DC motor with the IDA-PBC robust control law. The motor parameters are: $r = 0.05\Omega$, $L = 2mH$, $K = 0.07$N·m·A^{-1}, Br=0.01N·m·rad^{-1}s^{-1}, $J_m = 0.0006$Kg·m^2, the nominal torque is $\tau_L = 1.25$N·m and γ =0.05. The system starts at $\omega = 170$rad·s^{-1} with $\omega^d = 120$rad·s^{-1}. For $t = 1$s the desired mechanical speed is changed to $\omega^d = 170$rad·s−1, and for $t = 2$s the external torque decreases until $\tau_L = 0.25$N·m. Notice that the mechanical speed regulation is achieved even with the change of the external torque.

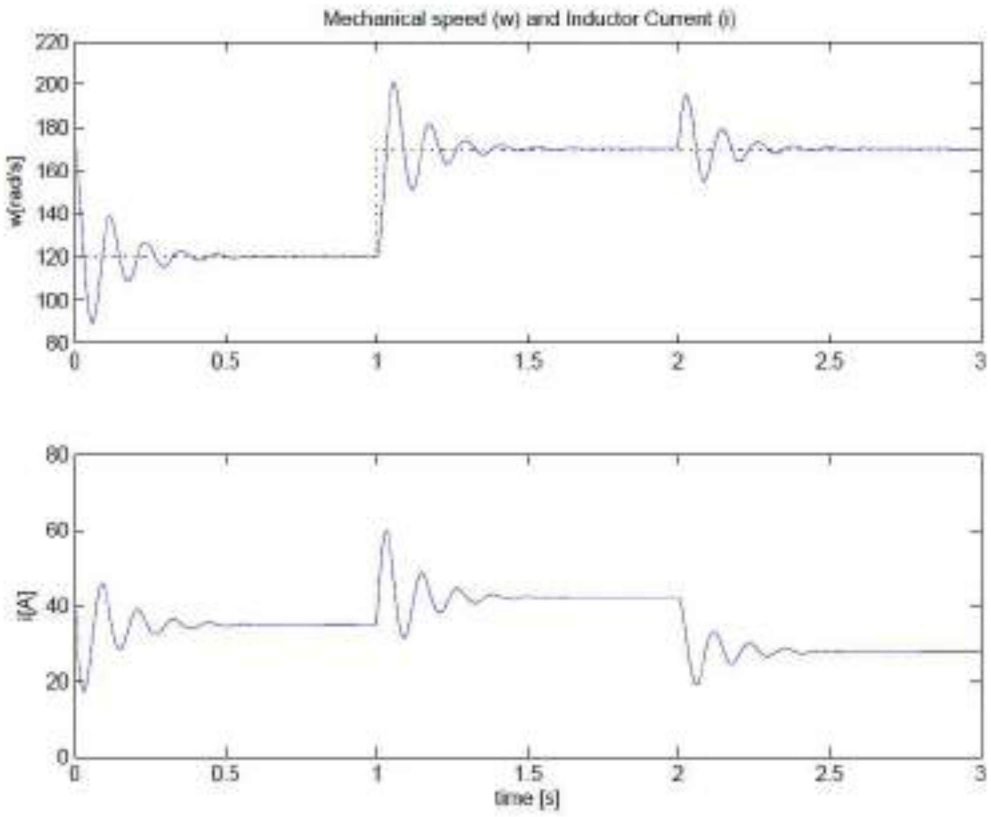
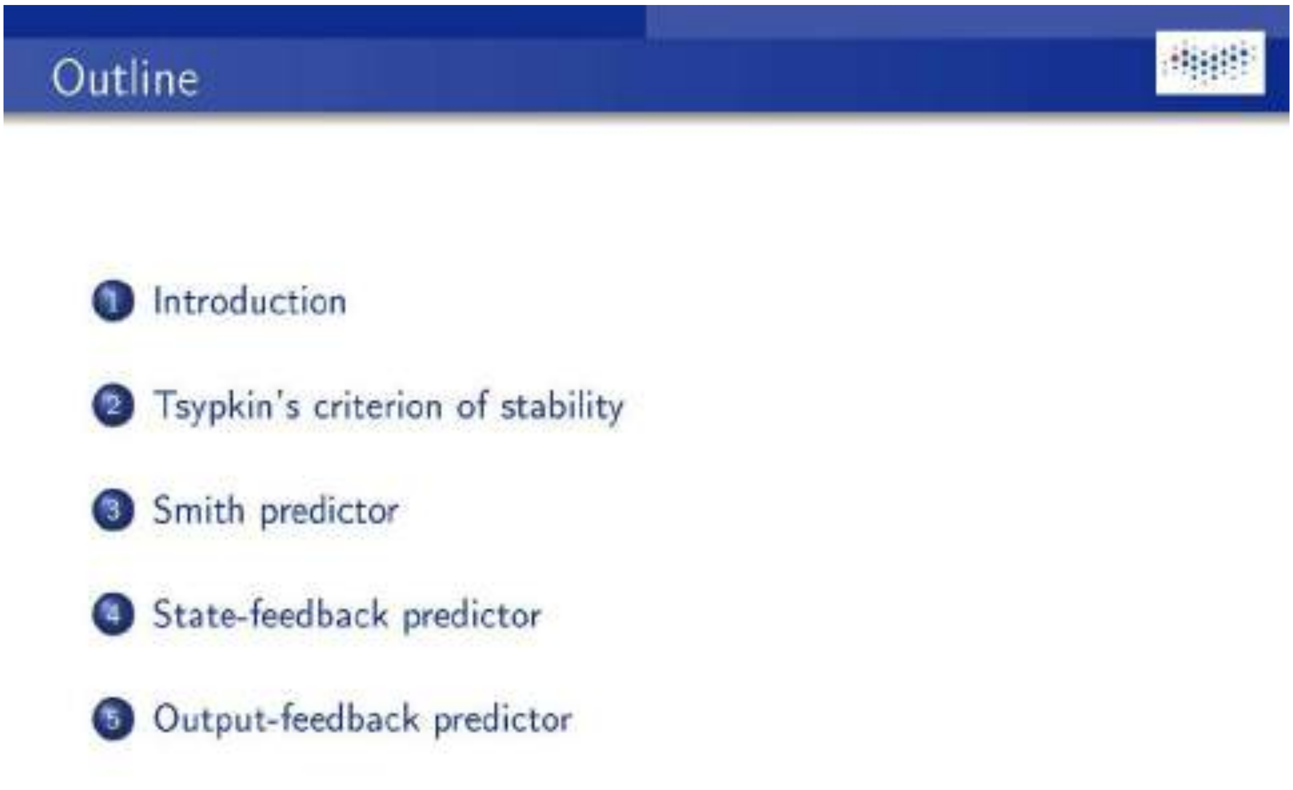


Figure 24: Simulations of the IDA-PBC robust for a DC motor

We have witnessed that one of the advantages of the port-Hamiltonian (pH) framework is that the Hamiltonian can be used as a basis to construct a candidate Lyapunov function, thus providing insight into various system properties like stability, passivity, finite L_2 gain, etc.

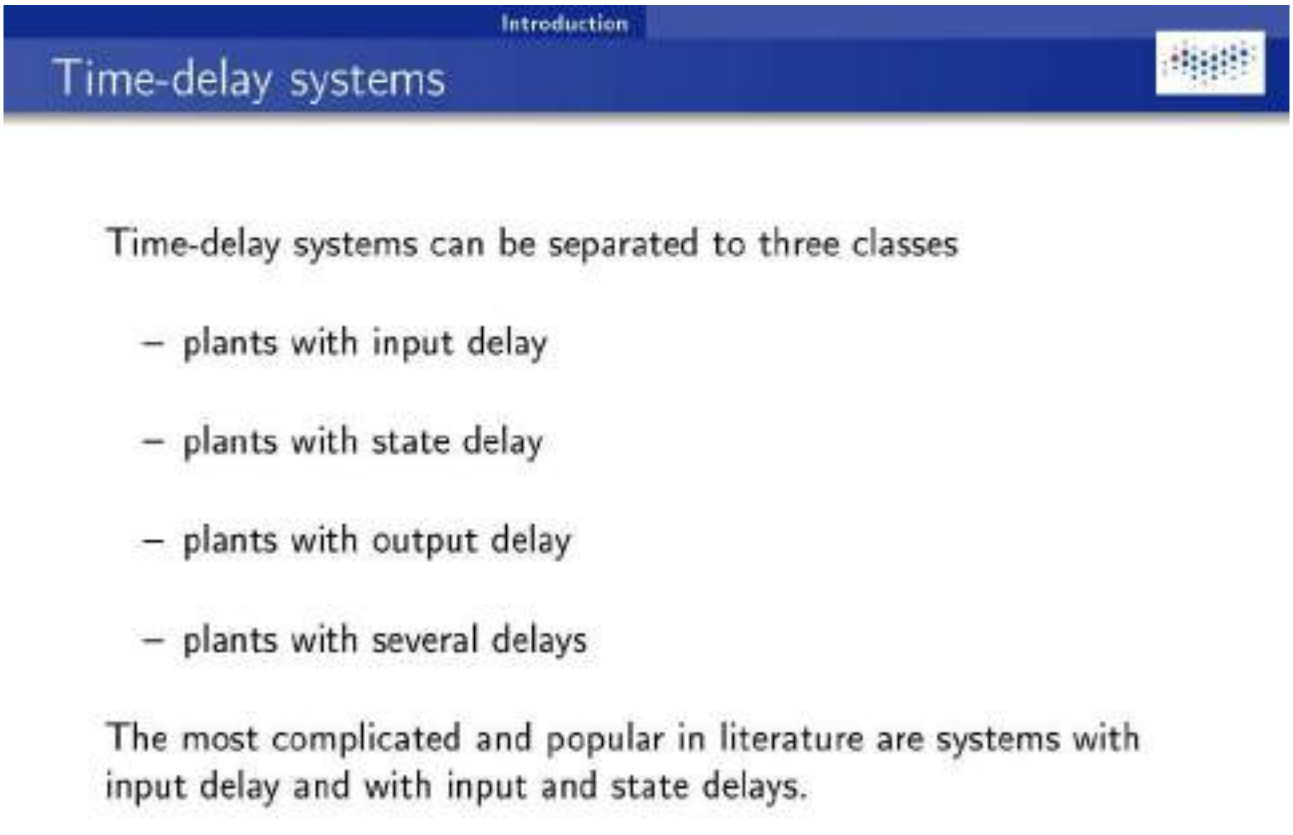
Another key feature of pH systems is that a power-preserving interconnection of pH systems results in another pH system, with total Hamiltonian being the sum of the Hamiltonian functions and with a Dirac structure defined by the composition of the Dirac structures of the subsystems. These features have led to a research focus on the control of port-Hamiltonian systems.

Control approaches for time-delay systems



Outline

1. Introduction
2. Tsypkin's criterion of stability
3. Smith predictor
4. State-feedback predictor
5. Output-feedback predictor



Introduction

Time-delay systems

Time-delay systems can be separated to three classes

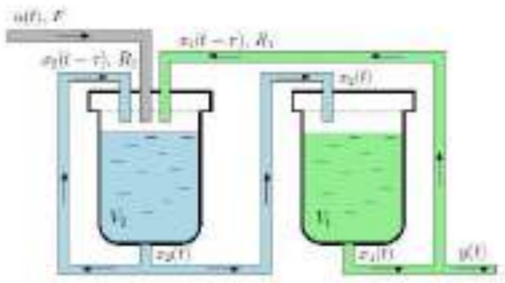
- plants with input delay
- plants with state delay
- plants with output delay
- plants with several delays

The most complicated and popular in literature are systems with input delay and with input and state delays.

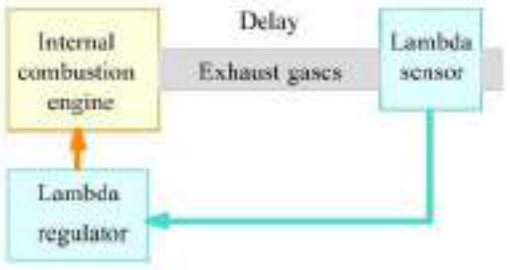
Introduction

Technical systems with time delays

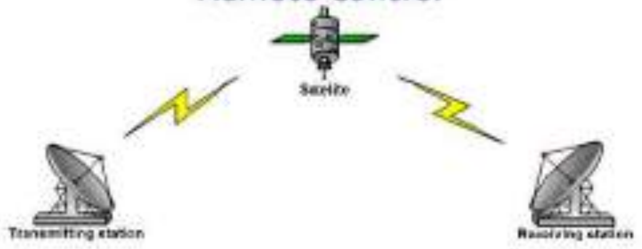
Chemical reactor



Combustion engine



Remote control

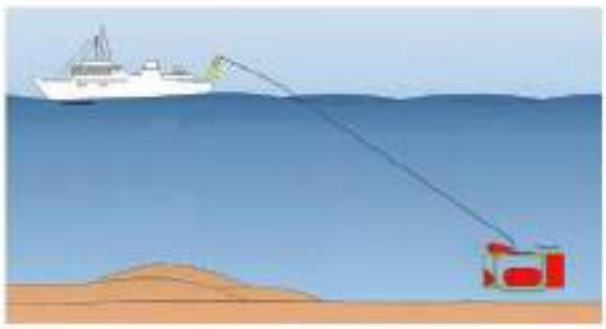


Introduction

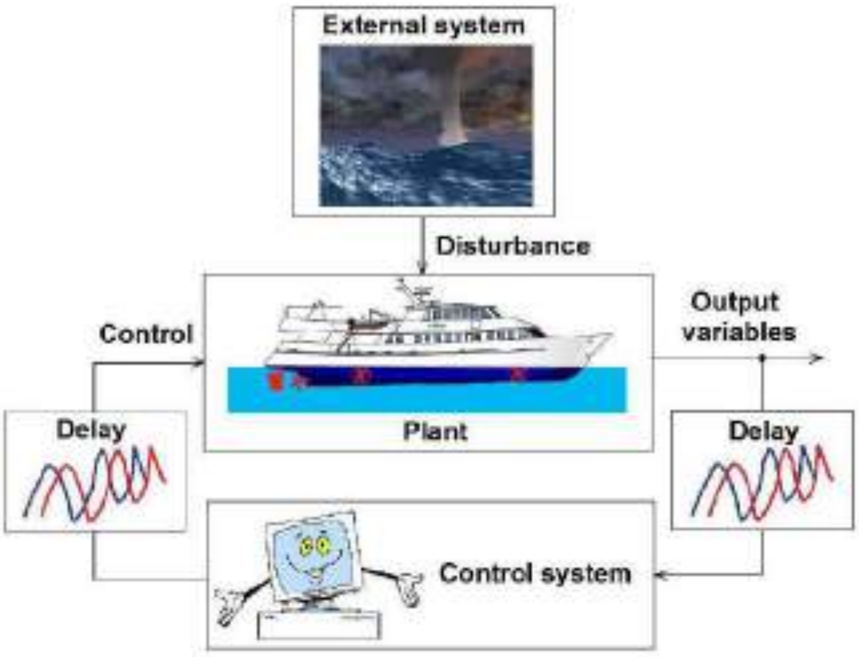
Systems with delays and external disturbances

Towing of an underwater vehicle

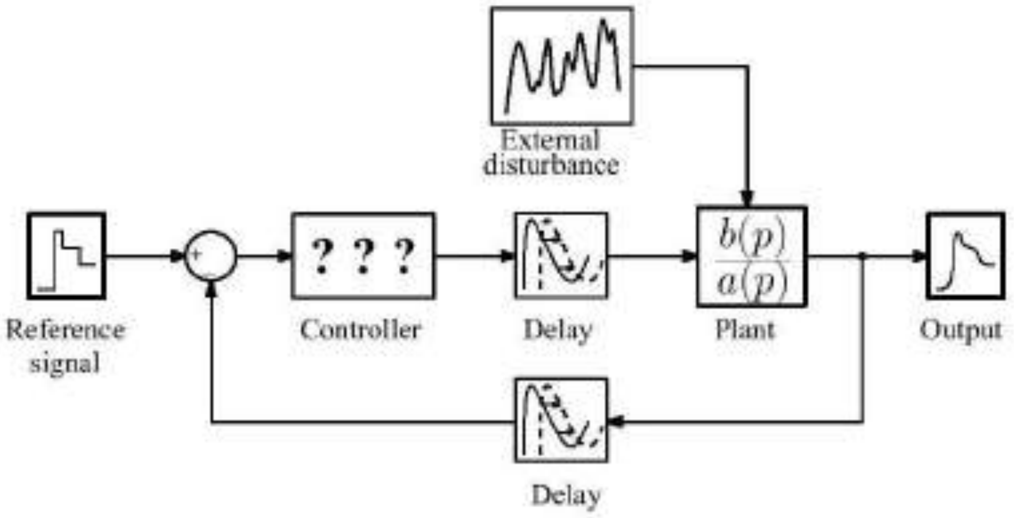
Extraction of nodules



Present-day view at the problem

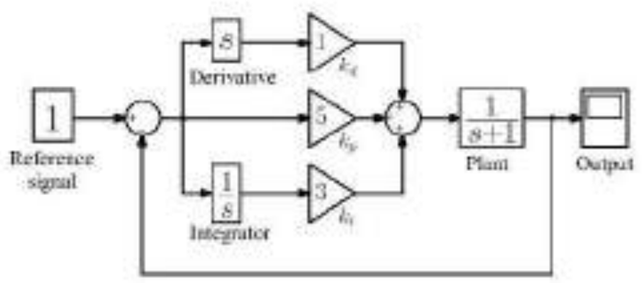


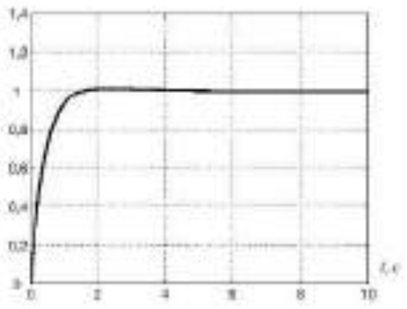
General problem formulation



Closed-loop system with delays

Closed-loop system with PID-controller

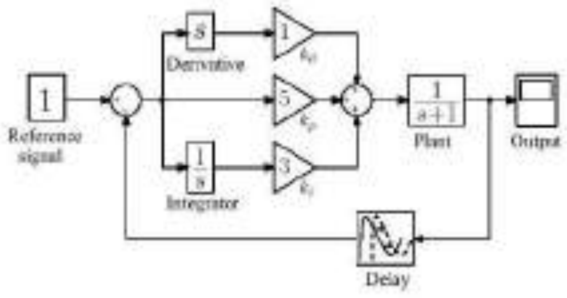


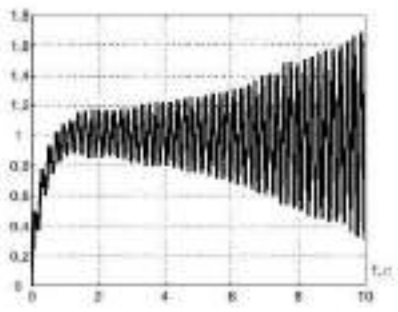


Structural scheme of the system

Transients for the output

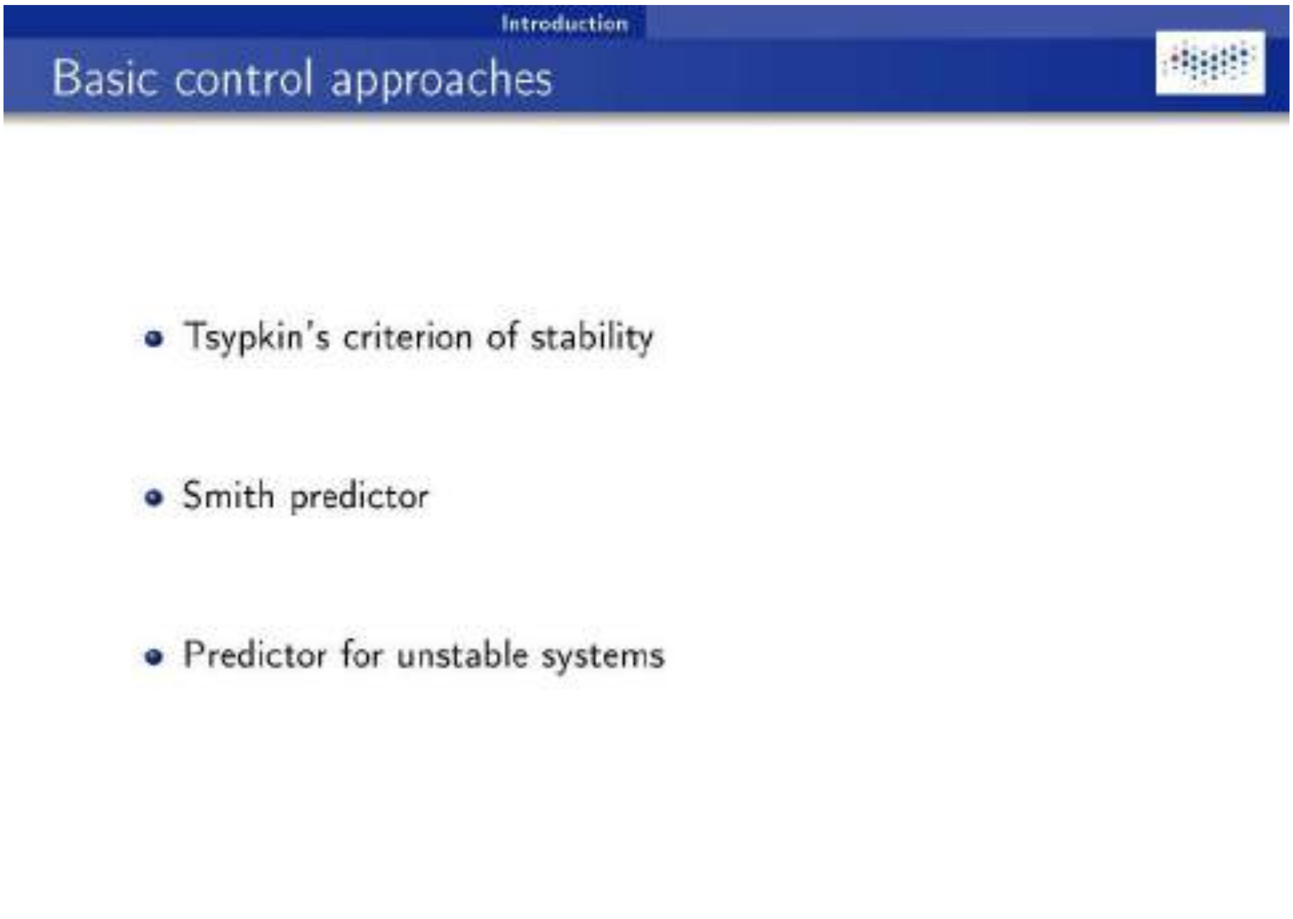
Closed-loop system with delay





Structural scheme of the system

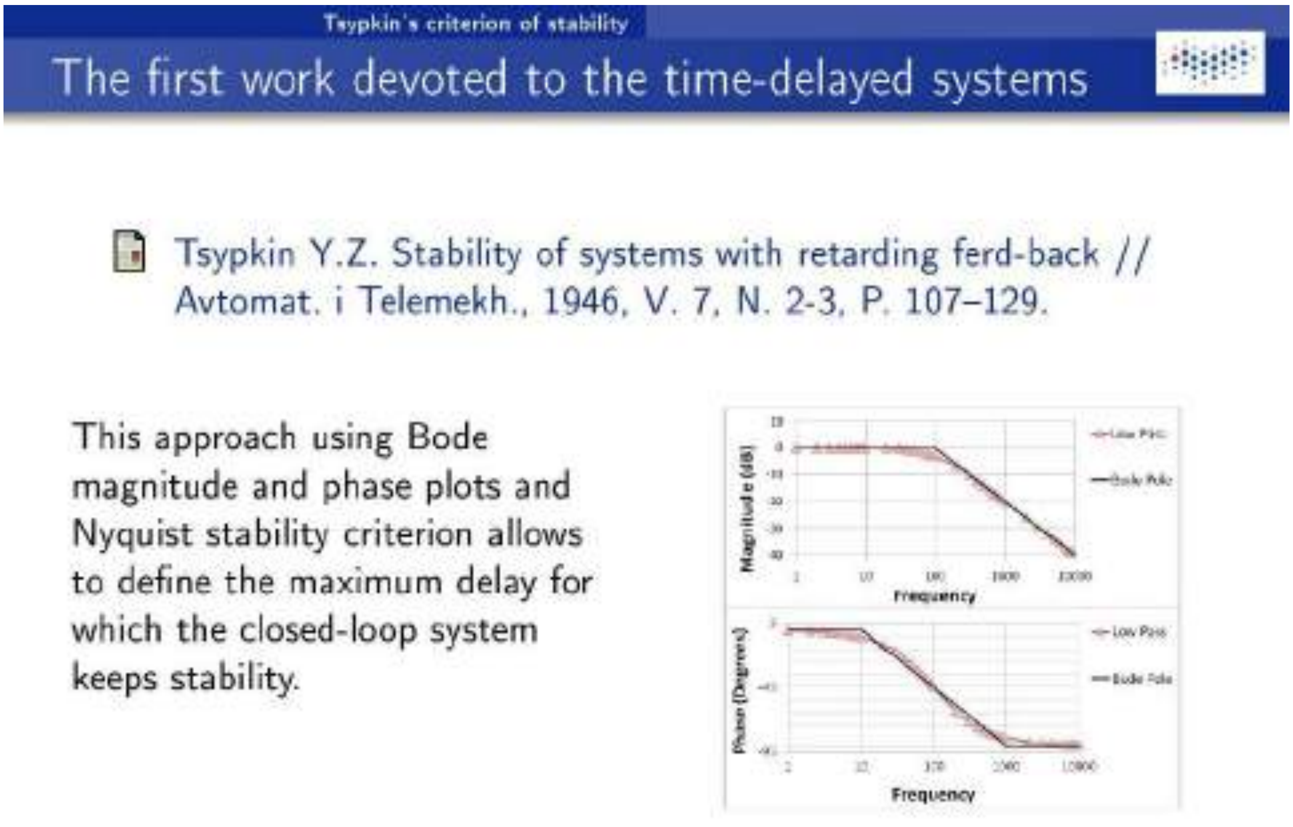
Transients for the output



Introduction

Basic control approaches

- Tsypkin's criterion of stability
- Smith predictor
- Predictor for unstable systems

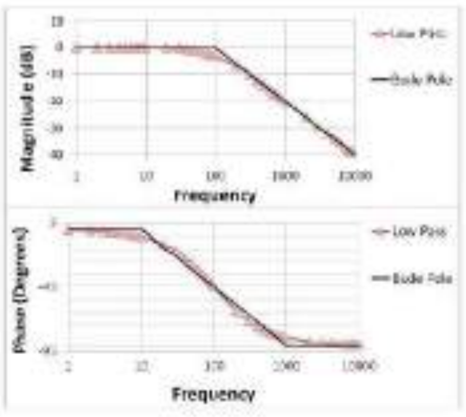


Tsypkin's criterion of stability

The first work devoted to the time-delayed systems

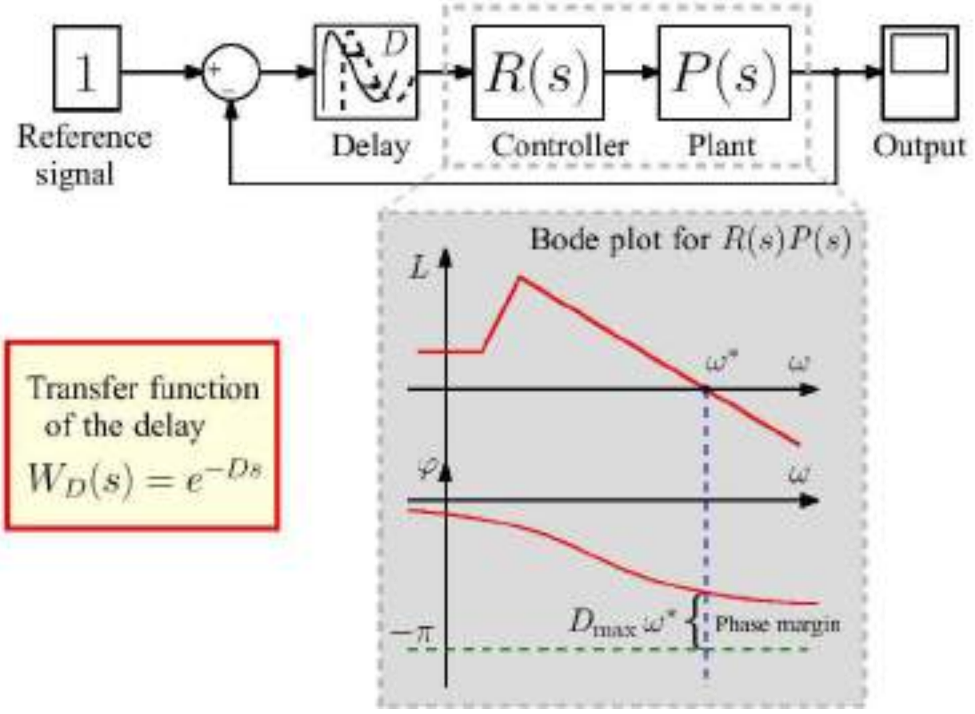
Tsypkin Y.Z. Stability of systems with retarding ferd-back // Avtomat. i Telemekh., 1946, V. 7, N. 2-3, P. 107–129.

This approach using Bode magnitude and phase plots and Nyquist stability criterion allows to define the maximum delay for which the closed-loop system keeps stability.



Tsypkin's criterion of stability

Maximum allowable delay in the closed loop



Smith predictor

Smith predictor

Smith predictor is a special structure of the controller proposed by Otto Smith in 1957.

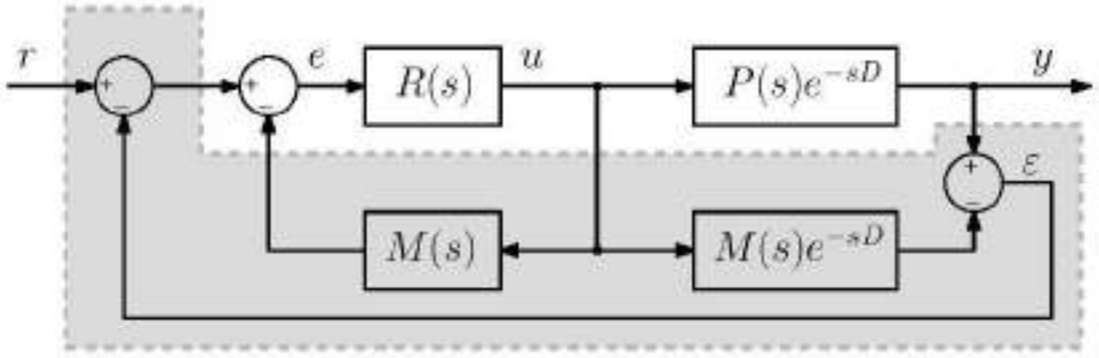
Smith O.J.M., Closer control of loops with dead time // Chem. Eng. Prog., 1959, N. 53, P. 217–219.

Smith O.J.M., A controller to overcome dead time // ISA, 1959, V. 6, P. 28–33.

The main goal of the Smith predictor is to predict which signal will appear before it will happen.

Smith predictor

Smith predictor



Control system supplied with Smith predictor

$M(s)$ is a model of the plant
e^{-sD} is a transfer function of the delay
$R(s)$ is a structure of the nominal controller
$P(s)e^{-sD}$ is a transfer function of the plant with the input delay

Smith predictor

Smith predictor

Assume that the model of the plan is ideal, i.e. $M(s) = P(s)$. Then the error between real output and output estimate will be zero ($\varepsilon = 0$). Thus, we have

$$y = Pe^{-sD}\left(\frac{R}{1+RM}\right)r = \left(\frac{PR}{1+RP}e^{-sD}\right)r. \qquad (1)$$

The term $\left(\frac{PR}{1+RP}\right)$ is a transfer function of the closed-loop system without delay.
It means that the delay does not exist in the feedback loop and does not affect the stability and performance of the closed-loop system. In other words controller does the job independently on the time delay. The delay exist only in a numerator of the transfer function that means the output after regulation is delayed.

Smith predictor

Consider the Smith predictor without assumption $\varepsilon = 0$. In this case the model of the closed-loop system will be

$$y = Pe^{-sD}R\,(r - \varepsilon - Mu)\,,\ \varepsilon = y - Me^{-sD}u,$$
$$y = Pe^{-sD}u, \tag{2}$$

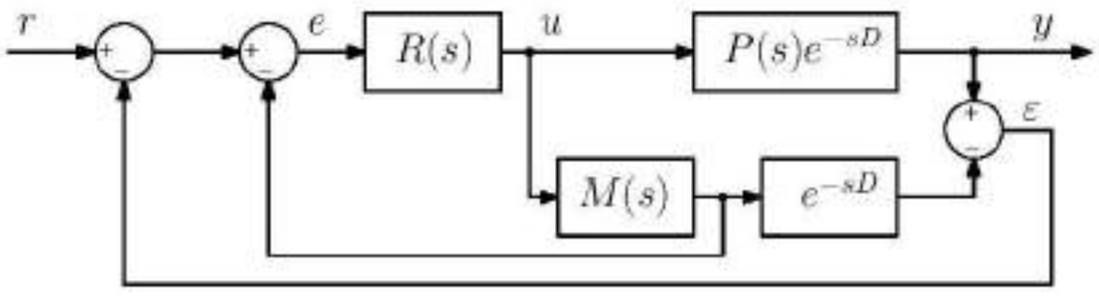
hence

$$y = \left[\frac{PR}{1 + RM + R(P - M)e^{-sD}}\right] e^{-sD}\,r. \tag{3}$$

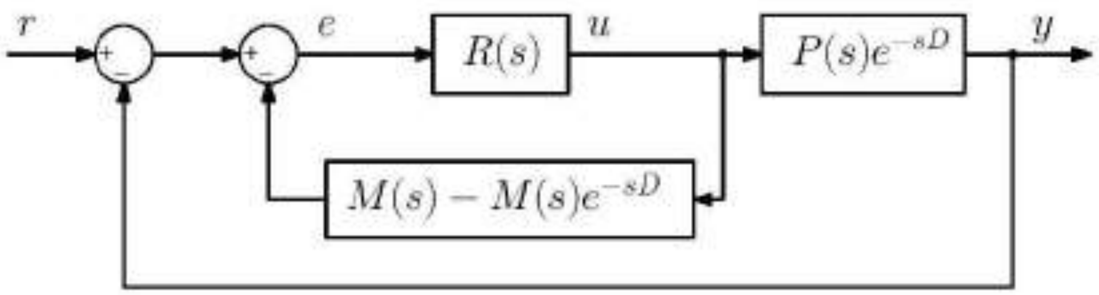
One can see that the error $M - P$ converges to zero if the model is precise, and the exponential term in denominator associated with the delay disappears (in square brackets (3)).

Modified Smith predictor

Using topological transformations one can get several equivalent structures of Smith predictor.

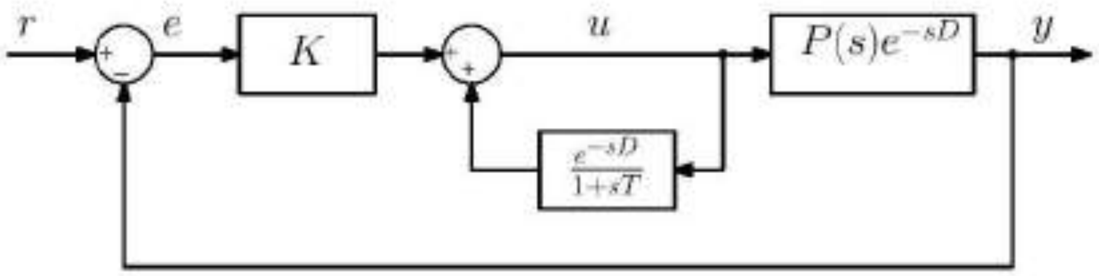


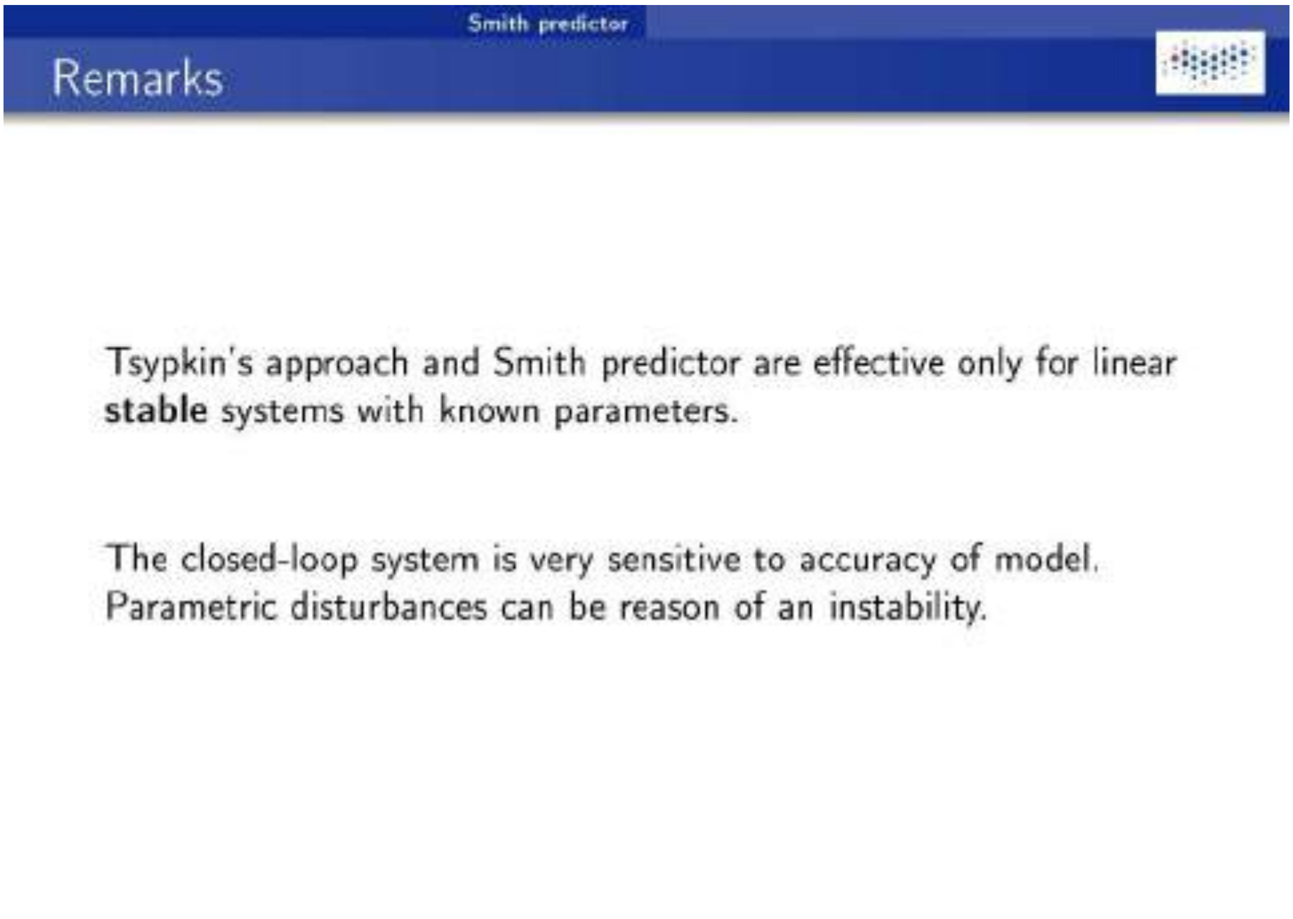
Modified Smith predictor



Modified Smith predictor

Predictable proportional-integral controller (PPI-controller) is a modified Smith predictor which is widely utilized in automatic control. Its structure presented on the figure below



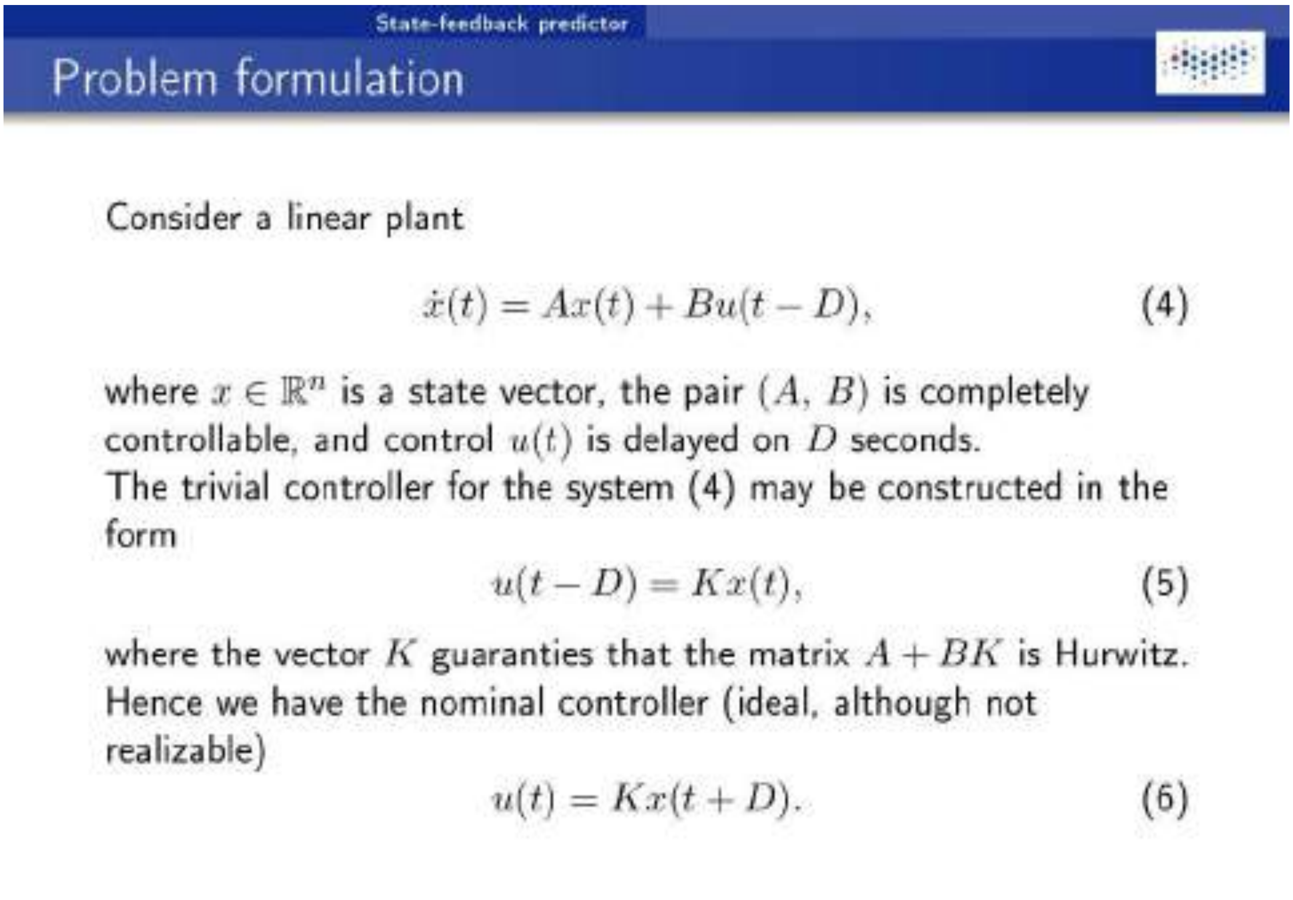


Smith predictor

Remarks

Tsypkin's approach and Smith predictor are effective only for linear **stable** systems with known parameters.

The closed-loop system is very sensitive to accuracy of model. Parametric disturbances can be reason of an instability.



State-feedback predictor

Problem formulation

Consider a linear plant

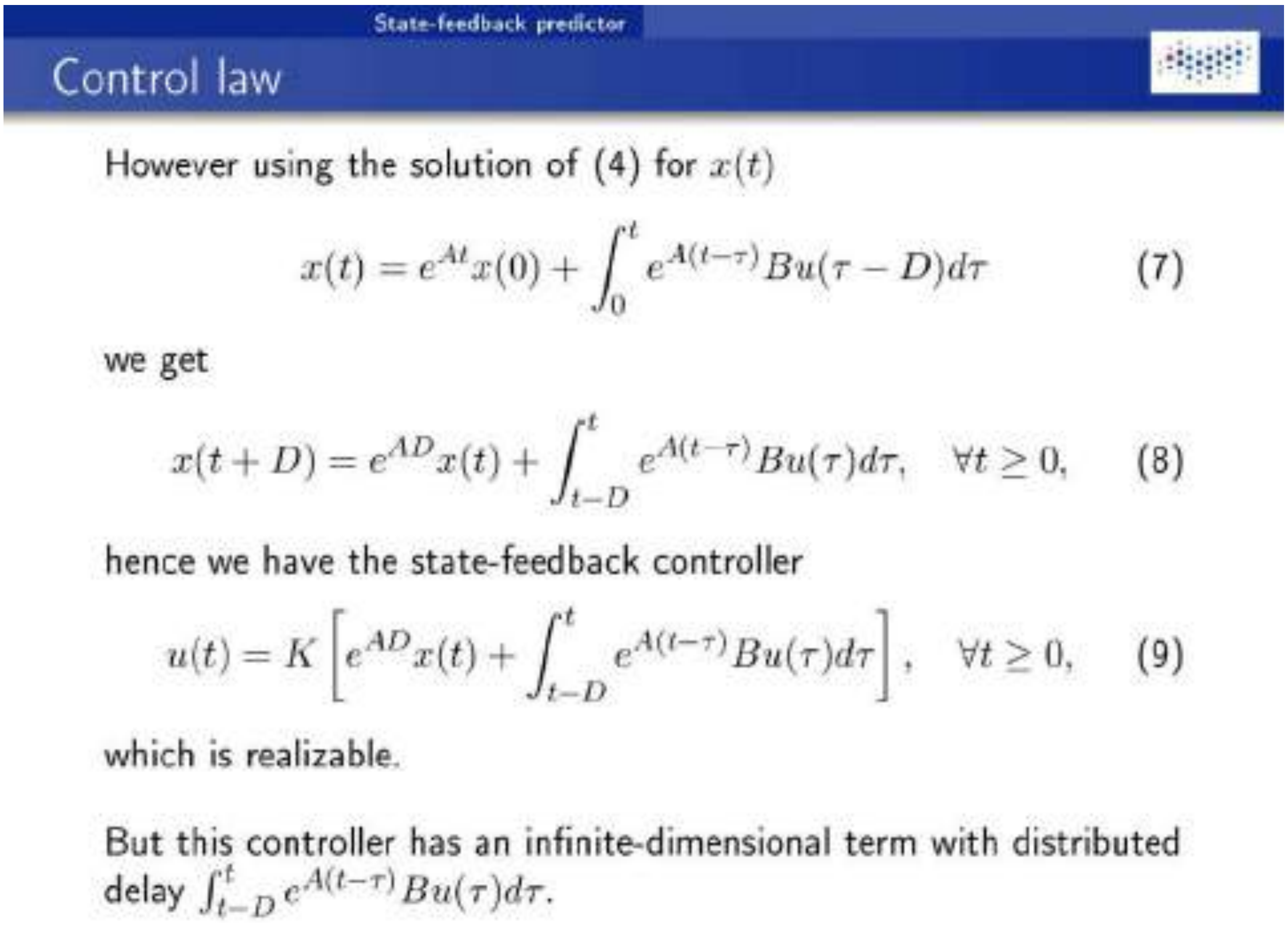
$$\dot{x}(t) = Ax(t) + Bu(t - D), \tag{4}$$

where $x \in \mathbb{R}^n$ is a state vector, the pair (A, B) is completely controllable, and control $u(t)$ is delayed on D seconds.
The trivial controller for the system (4) may be constructed in the form

$$u(t - D) = Kx(t), \tag{5}$$

where the vector K guaranties that the matrix $A + BK$ is Hurwitz.
Hence we have the nominal controller (ideal, although not realizable)

$$u(t) = Kx(t + D). \tag{6}$$



State-feedback predictor

Control law

However using the solution of (4) for $x(t)$

$$x(t) = e^{At}x(0) + \int_0^t e^{A(t-\tau)}Bu(\tau - D)d\tau \qquad (7)$$

we get

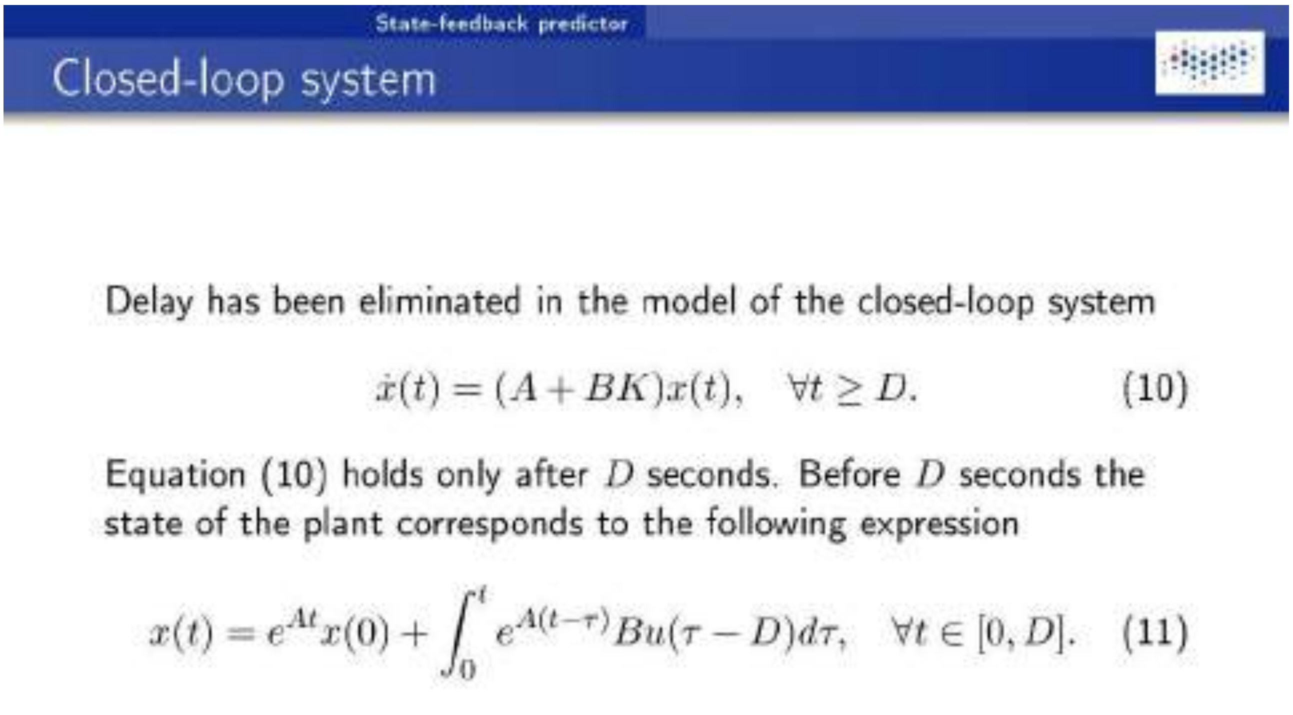
$$x(t+D) = e^{AD}x(t) + \int_{t-D}^t e^{A(t-\tau)}Bu(\tau)d\tau, \quad \forall t \geq 0, \qquad (8)$$

hence we have the state-feedback controller

$$u(t) = K\left[e^{AD}x(t) + \int_{t-D}^t e^{A(t-\tau)}Bu(\tau)d\tau\right], \quad \forall t \geq 0, \qquad (9)$$

which is realizable.

But this controller has an infinite-dimensional term with distributed delay $\int_{t-D}^t e^{A(t-\tau)}Bu(\tau)d\tau$.



State-feedback predictor

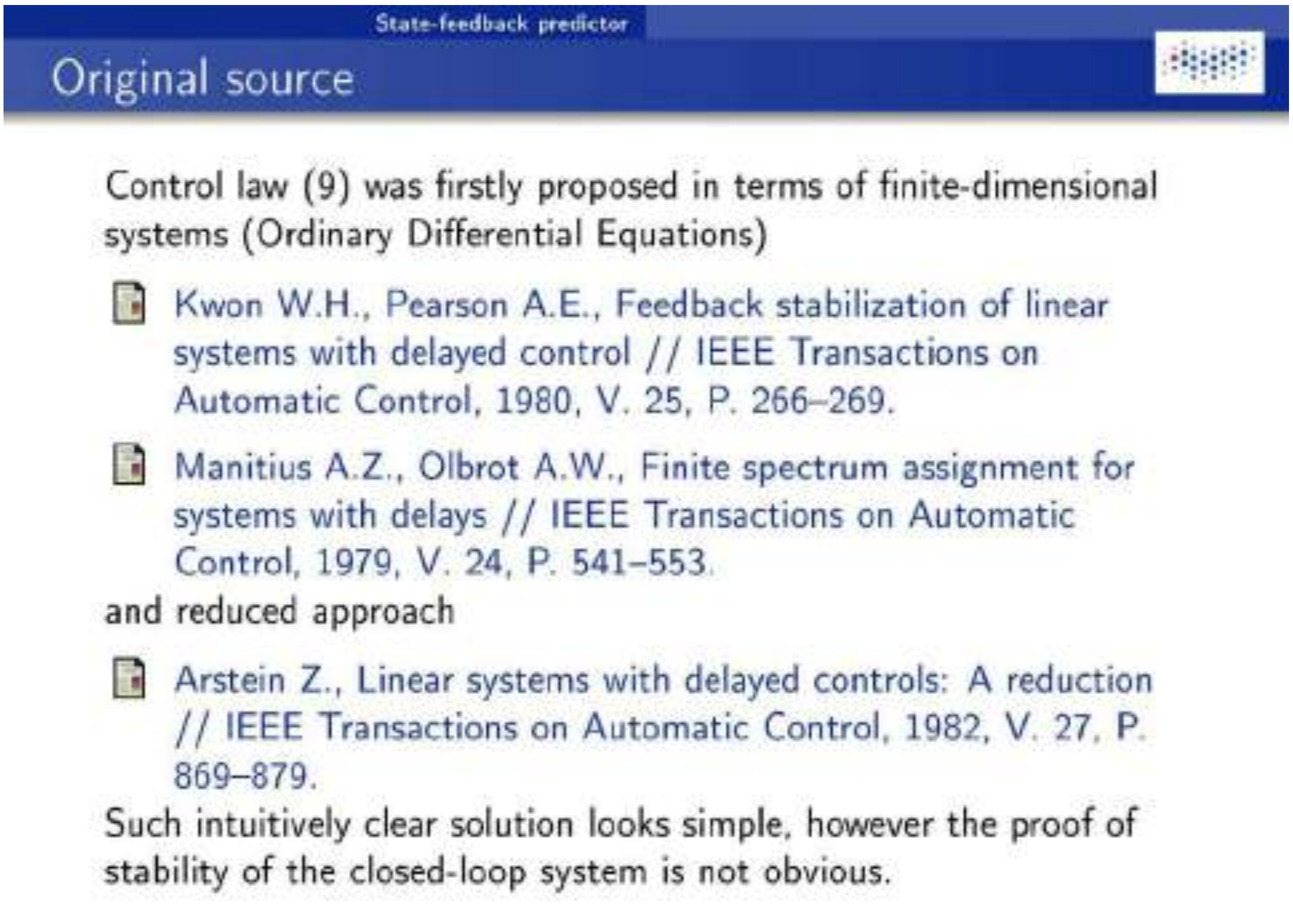
Closed-loop system

Delay has been eliminated in the model of the closed-loop system

$$\dot{x}(t) = (A + BK)x(t), \quad \forall t \geq D. \qquad (10)$$

Equation (10) holds only after D seconds. Before D seconds the state of the plant corresponds to the following expression

$$x(t) = e^{At}x(0) + \int_0^t e^{A(t-\tau)}Bu(\tau - D)d\tau, \quad \forall t \in [0, D]. \qquad (11)$$



State-feedback predictor

Original source

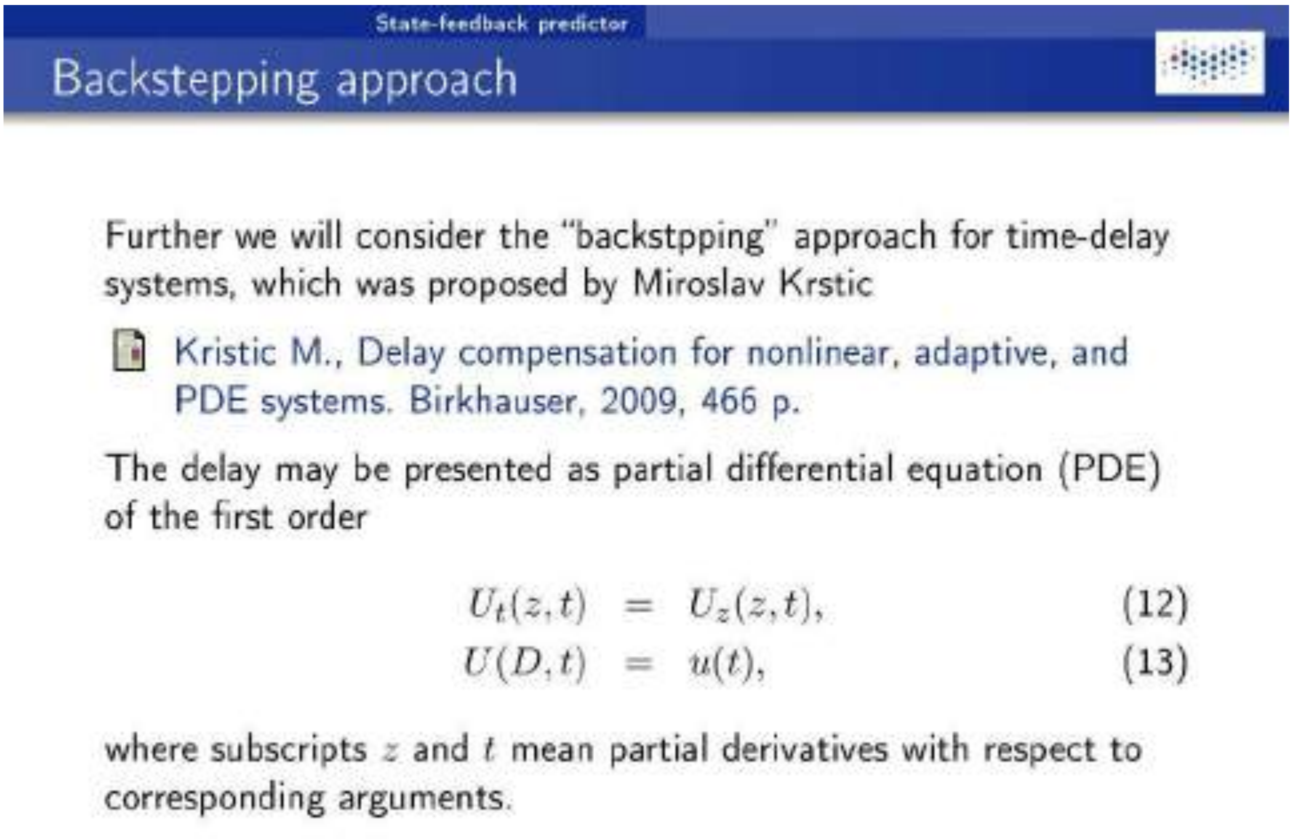
Control law (9) was firstly proposed in terms of finite-dimensional systems (Ordinary Differential Equations)

- Kwon W.H., Pearson A.E., Feedback stabilization of linear systems with delayed control // IEEE Transactions on Automatic Control, 1980, V. 25, P. 266–269.
- Manitius A.Z., Olbrot A.W., Finite spectrum assignment for systems with delays // IEEE Transactions on Automatic Control, 1979, V. 24, P. 541–553.

and reduced approach

- Arstein Z., Linear systems with delayed controls: A reduction // IEEE Transactions on Automatic Control, 1982, V. 27, P. 869–879.

Such intuitively clear solution looks simple, however the proof of stability of the closed-loop system is not obvious.



State-feedback predictor

Backstepping approach

Further we will consider the "backstpping" approach for time-delay systems, which was proposed by Miroslav Krstic

- Kristic M., Delay compensation for nonlinear, adaptive, and PDE systems. Birkhauser, 2009, 466 p.

The delay may be presented as partial differential equation (PDE) of the first order

$$U_t(z,t) = U_z(z,t), \qquad (12)$$

$$U(D,t) = u(t), \qquad (13)$$

where subscripts z and t mean partial derivatives with respect to corresponding arguments.

State-feedback predictor

PDE model of the delay

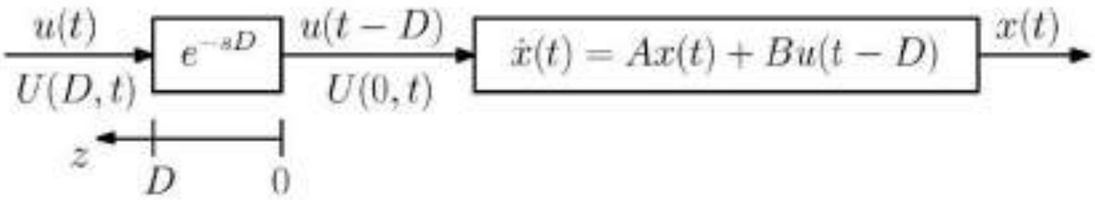
Solution of (12), (13) is

$$U(z,t) = u(t+z-D), \tag{14}$$

where the output of the delay

$$U(0,t) = u(t-D) \tag{15}$$

describes the delayed control signal



Linear plant with the input delay

State-feedback predictor

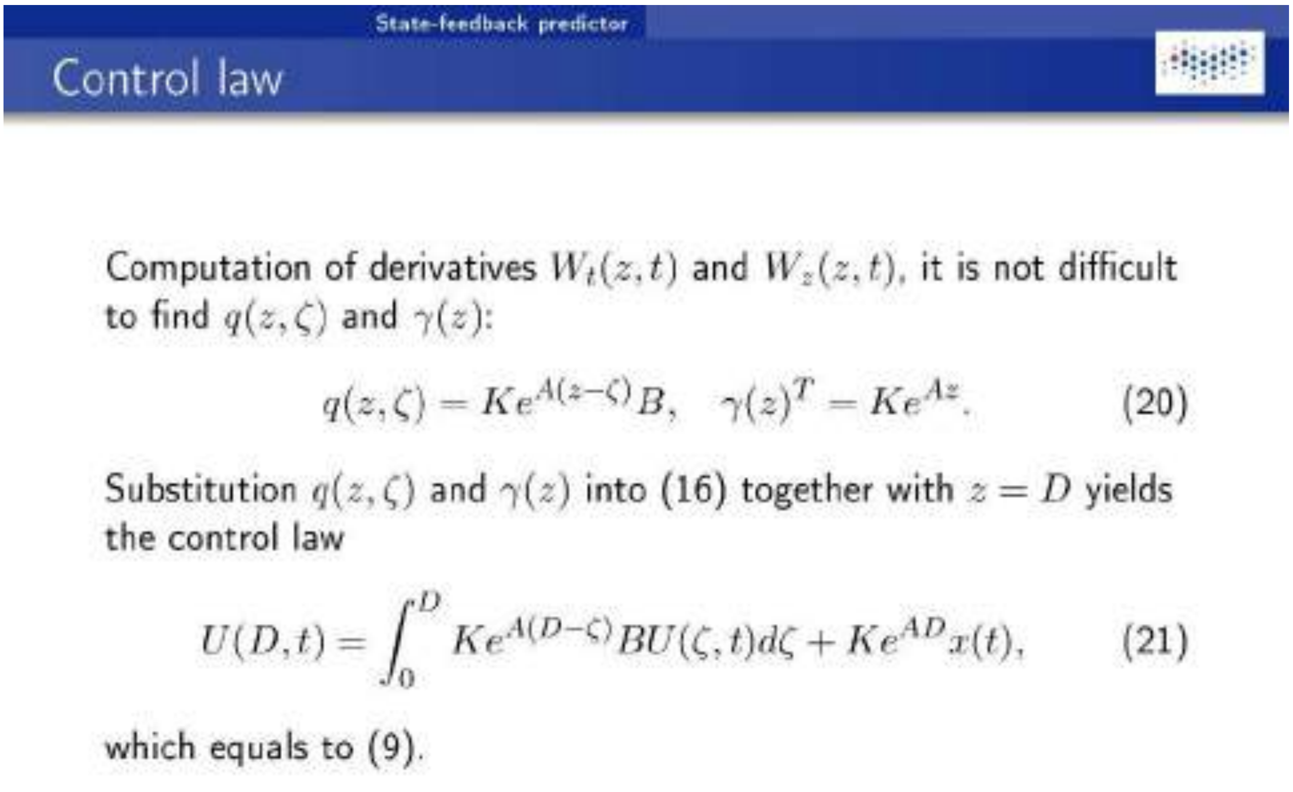
Backstepping transformation

Consider the following transformation [1]

$$W(z,t) = U(z,t) - \int_0^z q(z,\zeta)U(\zeta,t)d\zeta - \gamma(z)^T x(t), \tag{16}$$

which maps the system (4), (12)–(15) to internally stable system

$$\begin{aligned} \dot{x}(t) &= (A+BK)x(t) + BW(0,t), && (17)\\ W_t(z,t) &= W_z(z,t), && (18)\\ W(D,t) &= 0. && (19) \end{aligned}$$



State-feedback predictor

Control law

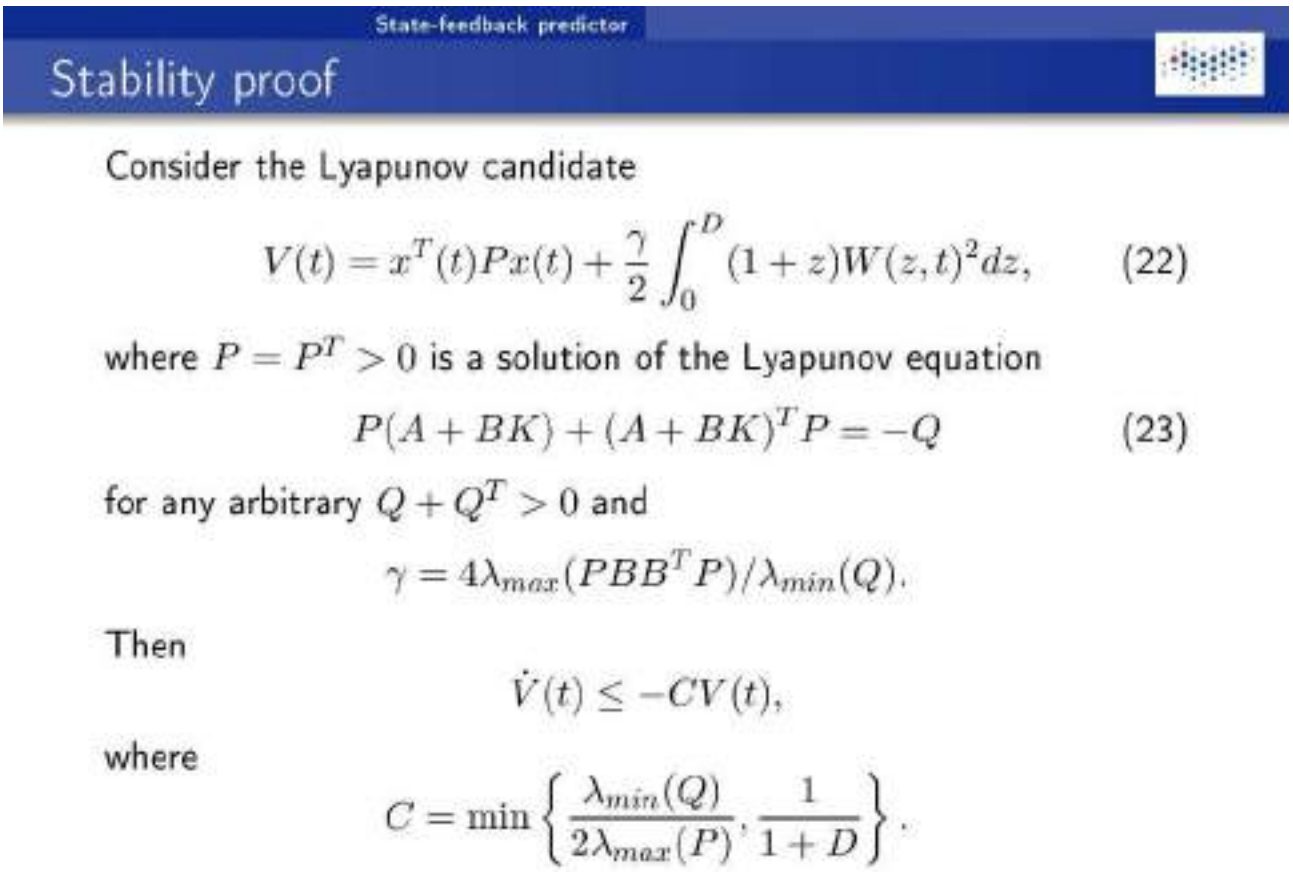
Computation of derivatives $W_t(z,t)$ and $W_z(z,t)$, it is not difficult to find $q(z,\zeta)$ and $\gamma(z)$:

$$q(z,\zeta) = Ke^{A(z-\zeta)}B, \quad \gamma(z)^T = Ke^{Az}. \tag{20}$$

Substitution $q(z,\zeta)$ and $\gamma(z)$ into (16) together with $z = D$ yields the control law

$$U(D,t) = \int_0^D Ke^{A(D-\zeta)}BU(\zeta,t)d\zeta + Ke^{AD}x(t), \tag{21}$$

which equals to (9).



State-feedback predictor

Stability proof

Consider the Lyapunov candidate

$$V(t) = x^T(t)Px(t) + \frac{\gamma}{2}\int_0^D (1+z)W(z,t)^2dz, \tag{22}$$

where $P = P^T > 0$ is a solution of the Lyapunov equation

$$P(A+BK) + (A+BK)^TP = -Q \tag{23}$$

for any arbitrary $Q + Q^T > 0$ and

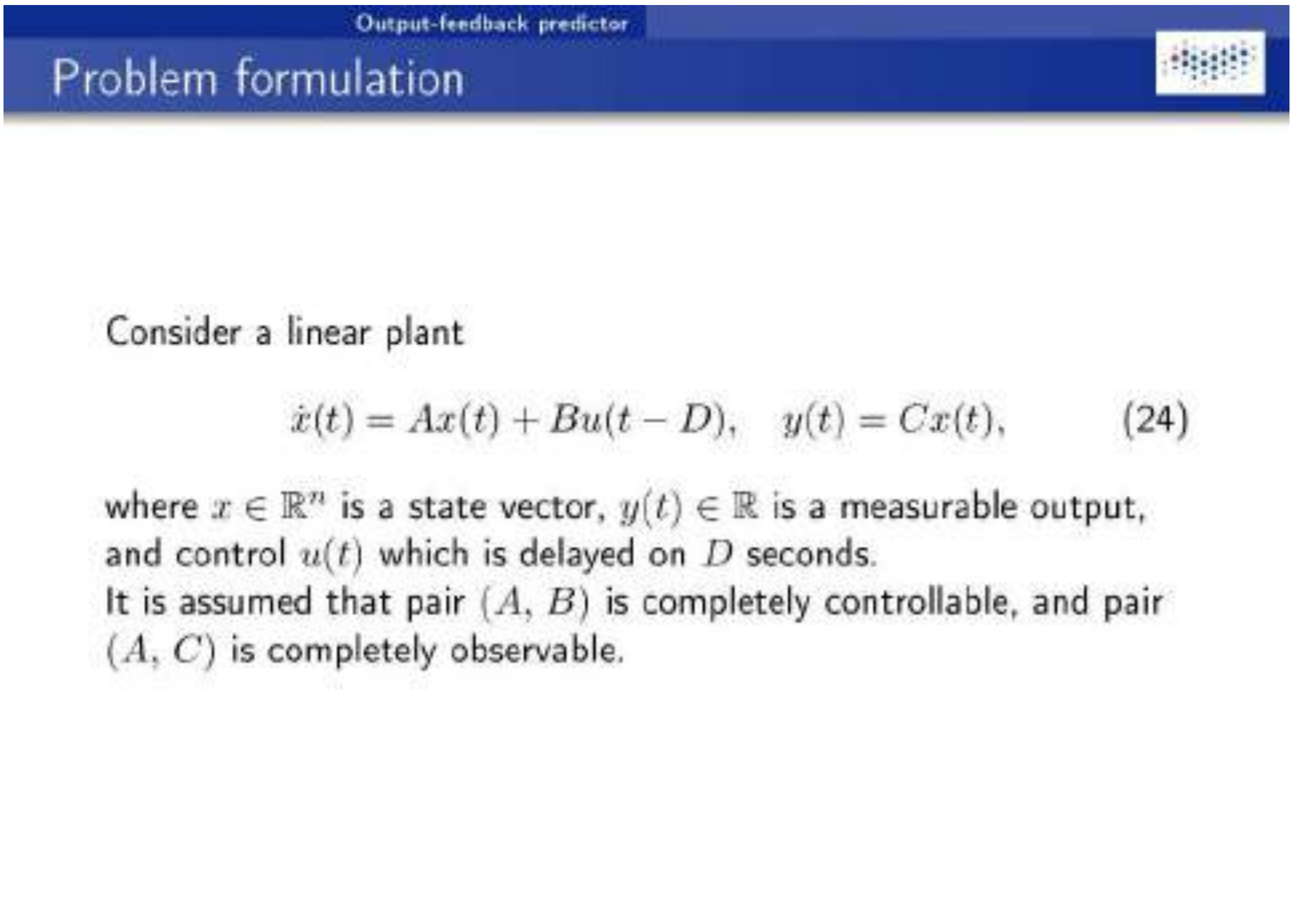
$$\gamma = 4\lambda_{max}(PBB^TP)/\lambda_{min}(Q).$$

Then

$$\dot{V}(t) \leq -CV(t),$$

where

$$C = \min\left\{\frac{\lambda_{min}(Q)}{2\lambda_{max}(P)}, \frac{1}{1+D}\right\}.$$



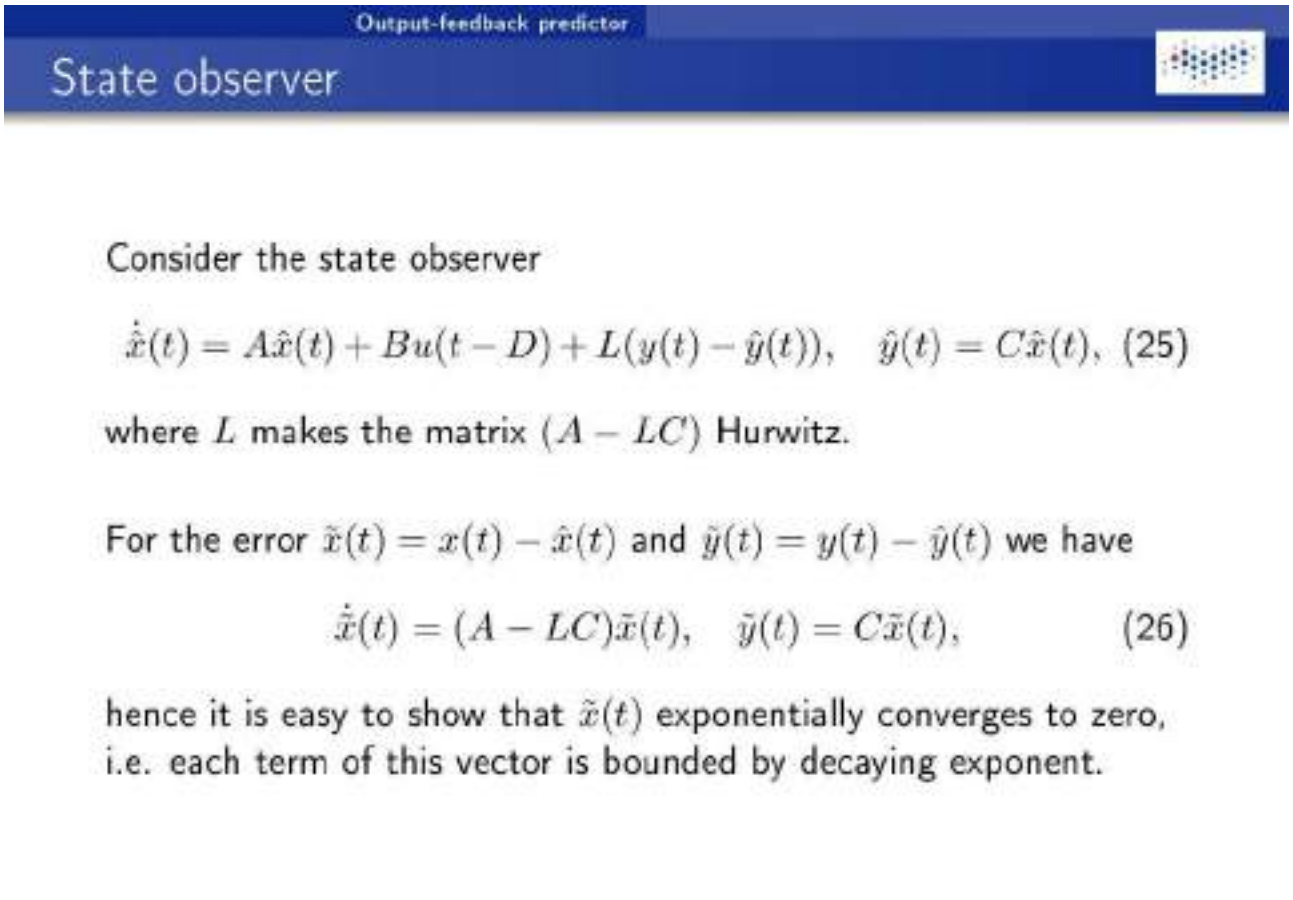
Output-feedback predictor

Problem formulation

Consider a linear plant

$$\dot{x}(t) = Ax(t) + Bu(t-D), \quad y(t) = Cx(t), \tag{24}$$

where $x \in \mathbb{R}^n$ is a state vector, $y(t) \in \mathbb{R}$ is a measurable output, and control $u(t)$ which is delayed on D seconds.
It is assumed that pair $(A,\, B)$ is completely controllable, and pair $(A,\, C)$ is completely observable.



Output-feedback predictor

State observer

Consider the state observer

$$\dot{\hat{x}}(t) = A\hat{x}(t) + Bu(t-D) + L(y(t) - \hat{y}(t)), \quad \hat{y}(t) = C\hat{x}(t), \tag{25}$$

where L makes the matrix $(A - LC)$ Hurwitz.

For the error $\tilde{x}(t) = x(t) - \hat{x}(t)$ and $\tilde{y}(t) = y(t) - \hat{y}(t)$ we have

$$\dot{\tilde{x}}(t) = (A - LC)\tilde{x}(t), \quad \tilde{y}(t) = C\tilde{x}(t), \tag{26}$$

hence it is easy to show that $\tilde{x}(t)$ exponentially converges to zero, i.e. each term of this vector is bounded by decaying exponent.

Output-feedback predictor

Backstepping transformation

Consider backstepping transformation like (16)

$$\hat{W}(z,t) = U(z,t) - Ke^{Az}\hat{x}(t) - K\int_0^z e^{A(z-\zeta)}BU(\zeta,t)d\zeta$$
$$+ K\int_z^D e^{A(z+D-\zeta)}L\tilde{Y}(\zeta,t)d\zeta, \quad (27)$$
$$\tilde{Y}(z,t) = \tilde{y}(t+z-D), \quad (28)$$
$$\tilde{Y}_t(z,t) = \tilde{Y}_z(z,t), \quad (29)$$
$$\tilde{Y}(D,t) = \tilde{y}(t). \quad (30)$$

Output-feedback predictor

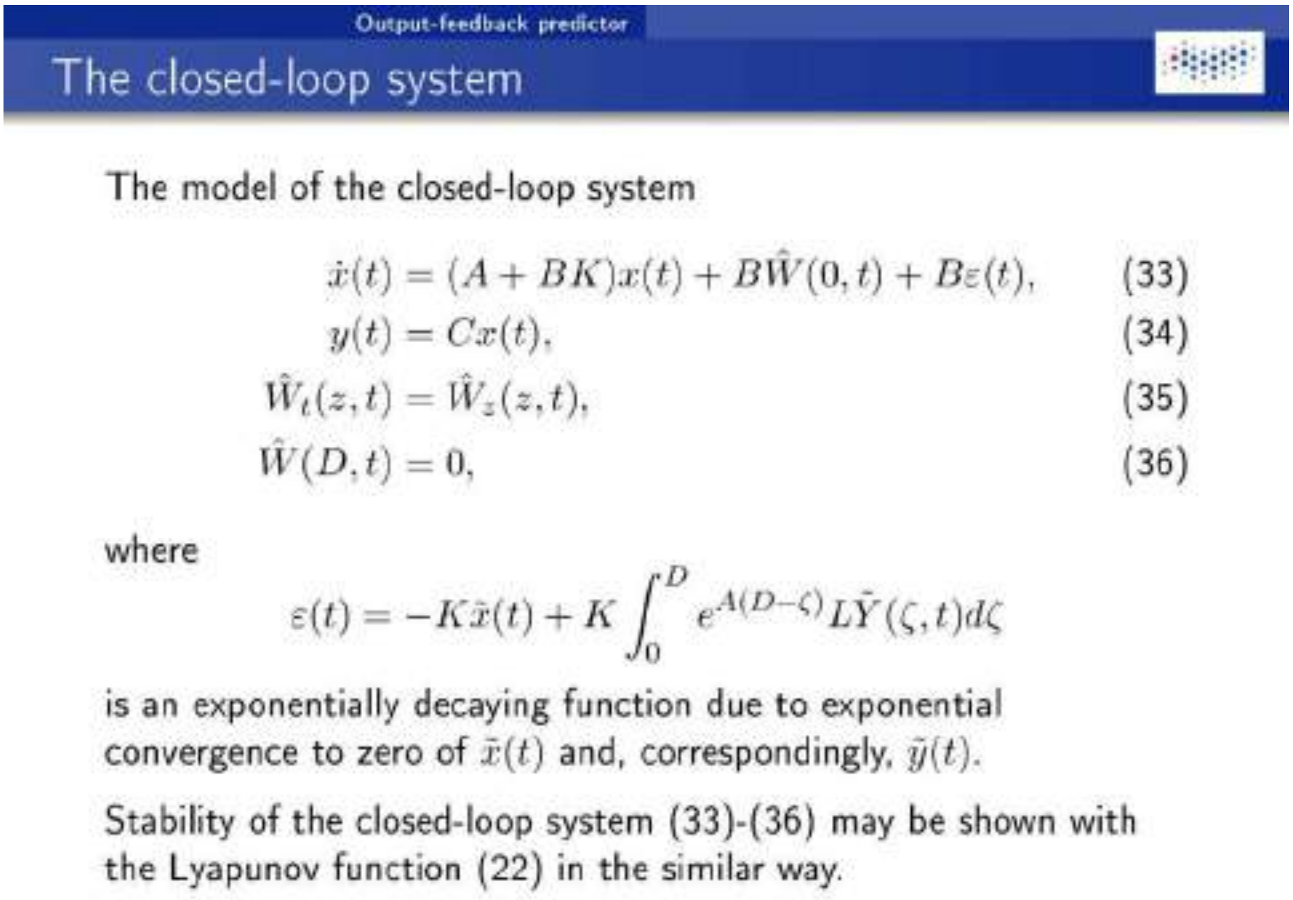
Control law

Choosing $z = D$ and equating $\hat{W}(D,t)$ to zero in (27) we get a realizable control law

$$u(t) = Ke^{AD}\hat{x}(t) + K\int_{t-D}^{t} e^{A(t-\tau)}Bu(\tau)d\tau, \quad (31)$$

which uses estimates of the state $\hat{x}(t)$.
Substitute in (24) the transformation (27) with $z = 0$:

$$\dot{x}(t) = Ax(t) + BK\hat{x}(t) + K\int_0^D e^{A(D-\zeta)}L\tilde{Y}(\zeta,t)d\zeta + B\hat{W}(0,t)$$
$$= (A+BK)x(t) + B\hat{W}(0,t)$$
$$- BK\tilde{x}(t) + BK\int_0^D e^{A(D-\zeta)}L\tilde{Y}(\zeta,t)d\zeta$$
$$= (A+BK)x(t) + B\hat{W}(0,t) + B\varepsilon(t). \quad (32)$$



Output-feedback predictor

The closed-loop system

The model of the closed-loop system

$$\dot{x}(t) = (A + BK)x(t) + B\hat{W}(0,t) + B\varepsilon(t), \tag{33}$$

$$y(t) = Cx(t), \tag{34}$$

$$\hat{W}_t(z,t) = \hat{W}_z(z,t), \tag{35}$$

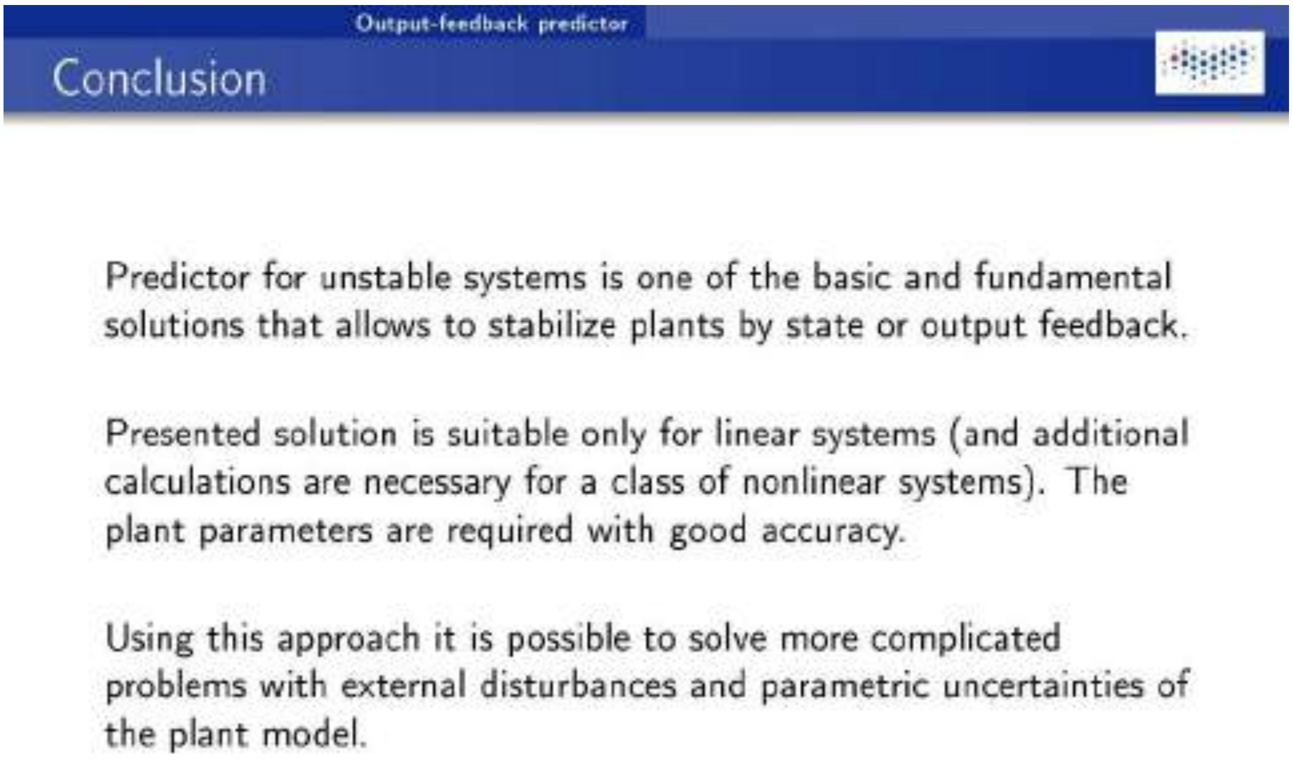
$$\hat{W}(D,t) = 0, \tag{36}$$

where

$$\varepsilon(t) = -K\tilde{x}(t) + K\int_0^D e^{A(D-\zeta)} L\tilde{Y}(\zeta,t)d\zeta$$

is an exponentially decaying function due to exponential convergence to zero of $\tilde{x}(t)$ and, correspondingly, $\tilde{y}(t)$.

Stability of the closed-loop system (33)-(36) may be shown with the Lyapunov function (22) in the similar way.



Output-feedback predictor

Conclusion

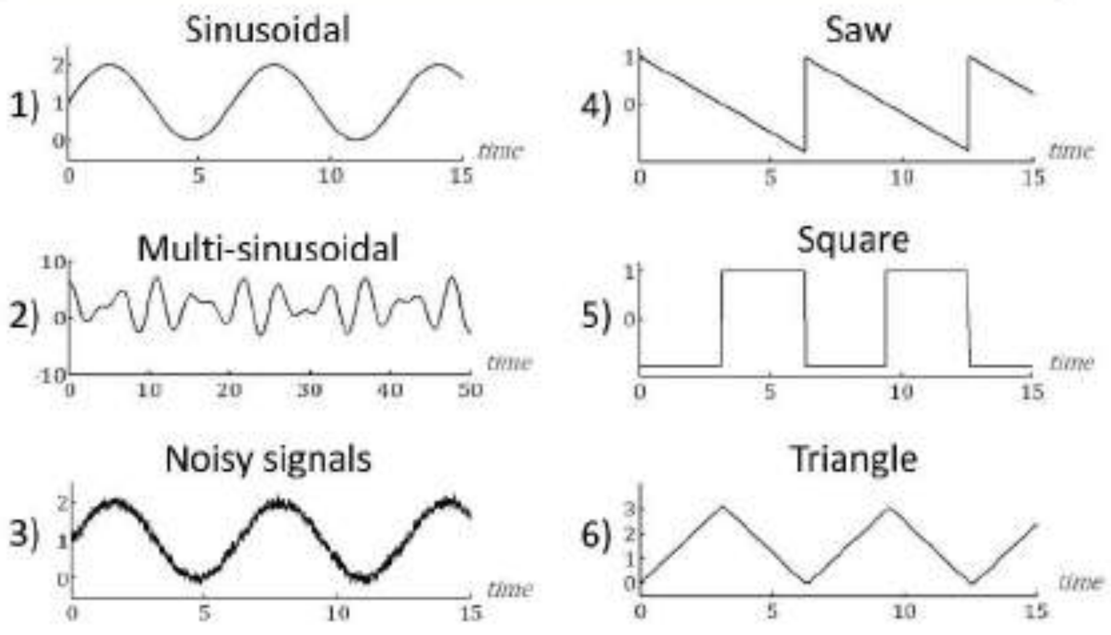
Predictor for unstable systems is one of the basic and fundamental solutions that allows to stabilize plants by state or output feedback.

Presented solution is suitable only for linear systems (and additional calculations are necessary for a class of nonlinear systems). The plant parameters are required with good accuracy.

Using this approach it is possible to solve more complicated problems with external disturbances and parametric uncertainties of the plant model.

Periodic signals online parameter estimation

Problem



Application

1. Measuring systems
2. Compensation system of
   – hard drives
   – ship
   – active suspension vehicle

History

1. Hsu L., Ortega R., Damm G. A globally convergent frequency estimator // *IEEE Transactions on Automatic Control. 1999.*

1. Marino R., Tomei R. Global Estimation of Unknown Frequencies // *IEEE Transactions on Automatic Control. 2002. Vol. 47.*

1. Obregon-Pulido G., Castillo-Toledo B., Loukianov A. A. Globally Convergent Estimator for n–Frequencies // *IEEE Transactions on Automatic Control. 2002. Vol. 47.*

1. Xia X. Global frequency estimation using adaptive identifiers» // *IEEE Transactions on Automatic Control, 2004. Vol. 47, No. 7, pp. 1188-1193.*

1. M. Hou. Estimation of Sinusoidal Frequencies and Amplitudes Using Adaptive Identifier and Observer // *IEEE Transactions On Automatic Control, Vol. 52, No. 3, March 2007*

Introduction

Input signal (1)

Sinusoidal
$$y(t) \;=\; \sigma + \mu sin(\omega t + \varphi) + \delta(t)$$

Multi-sinusoidal
$$y(t) = \sigma + \delta(t) + \sum_{i=1}^{k} \mu_i \, sin(\omega_i t + \varphi_i)$$

Triangle
$$y_{tr}(t) = \sum_{k=1}^{\infty} a_k^{tr} \sin((2k-1)\omega t)$$

Saw
$$y_{sw}(t) = \sum_{k=1}^{\infty} a_k^{sw} \sin(k\omega t)$$

Square
$$y_{sq}(t) = \sum_{k=1}^{\infty} a_k^{sq} \sin((2k-1)\omega t)$$

Introduction

Input signal $$y(t) = \sigma + \mu \sin(\omega t + \phi) \quad (1)$$

$$\dddot{y}(t) = -\omega^2 \dot{y}(t) \quad (2)$$

Linear filter (3)

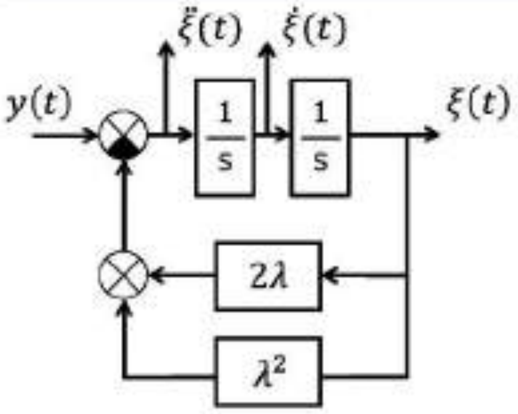
(4)

$$\dot{\xi}(s) = \frac{\lambda^2 s}{(s+\lambda)^2} y(s) \qquad \dddot{\xi}(s) = \frac{\lambda^2 s^3}{(s+\lambda)^2} y(s) \quad (5)$$

"Ideal" identification law

$$\dot{\hat{\theta}} = k\dot{\xi}^2 (\theta - \hat{\theta}) \quad (6)$$

Linear filter



Filter outputs

$$\xi(s) = \frac{\lambda^2}{(s+\lambda)^2} y(s)$$

$$\dot{\xi}(s) = \frac{\lambda^2 s}{(s+\lambda)^2} y(s)$$

$$\ddot{\xi}(s) = \frac{\lambda^2 s^2}{(s+\lambda)^2} y(s)$$

For signal (1) the following equation is satisfied

$$\dddot{\xi}(t) = \theta \dot{\xi}(t) + \varepsilon_{\xi}(t), \;\; \theta = -\omega^2$$

Additional variable

$$\chi(t) = \hat{\theta}(t) - k\dot{\xi}(t)\ddot{\xi}(t)$$

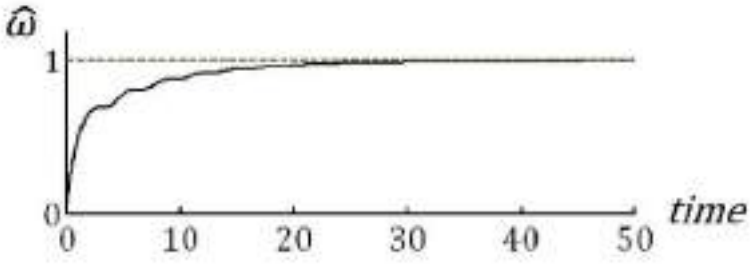
Estimation algorithm

$$\hat{\omega}(t) = \sqrt{|\hat{\theta}(t)|} \tag{7}$$

$$\hat{\theta}(t) = \chi(t) + k\dot{\xi}(t)\ddot{\xi}(t) \tag{8}$$

$$\dot{\chi}(t) = -k\dot{\xi}^2(t)\hat{\theta}(t) - k\ddot{\xi}^2(t) \tag{9}$$

Estimation error $$|\tilde{\omega}(t)| \le \rho e^{-\beta t}, \forall t \ge 0 \tag{10}$$



With disturbance

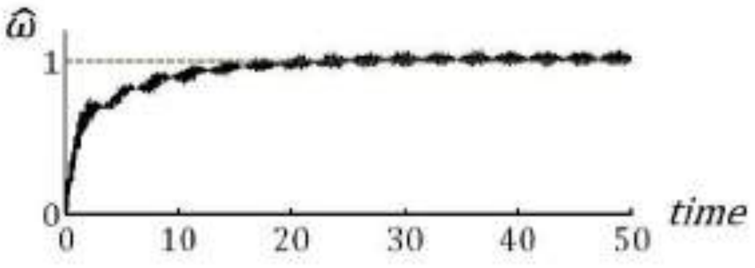
Estimation algorithm

$$\hat{\omega}(t) = \sqrt{|\hat{\theta}(t)|} \tag{7}$$

$$\hat{\theta}(t) = \chi(t) + k\dot{\xi}(t)\ddot{\xi}(t) \tag{8}$$

$$\dot{\chi}(t) = -k\dot{\xi}^2(t)\hat{\theta}(t) - k\ddot{\xi}^2(t) \tag{9}$$

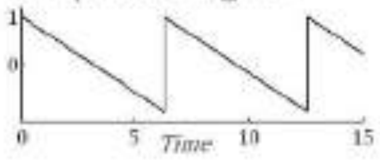
Estimation error $$|\tilde{\omega}(t)| \le \rho e^{-\beta t} + C_\omega, \forall t \ge 0 \tag{11}$$



Special signals

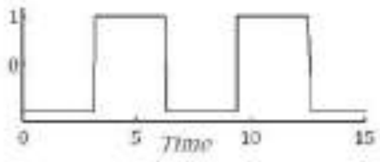
Graph of a signal

Fourier series decomposition



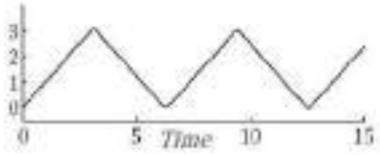
Saw

$$y_{sw}(t) = \sum_{k=1}^{\infty} \frac{2}{\pi}\frac{A}{k}\sin(k\omega t) \quad (12)$$



Square

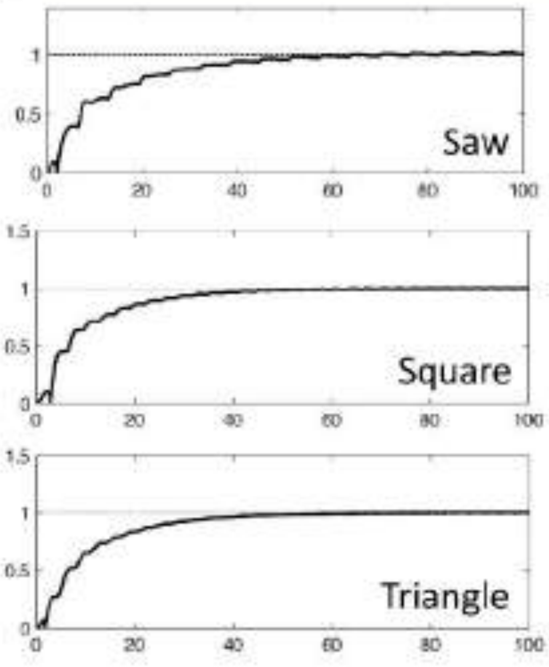
$$y_{sq}(t) = \sum_{k=1}^{\infty} \frac{4}{\pi}\frac{A}{2k-1}\sin((2k-1)\omega t) \quad (13)$$



Triangle

$$y_{tr}(t) = \sum_{k=1}^{\infty} \frac{8}{\pi^2}\frac{(-1)^{k-1}A}{(2k-1)^2}\sin((2k-1)\omega t) \quad (14)$$

Estimation of the frequency



$$\hat{\omega}(t) = \sqrt{\left|\hat{\theta}(t)\right|} \quad (15)$$

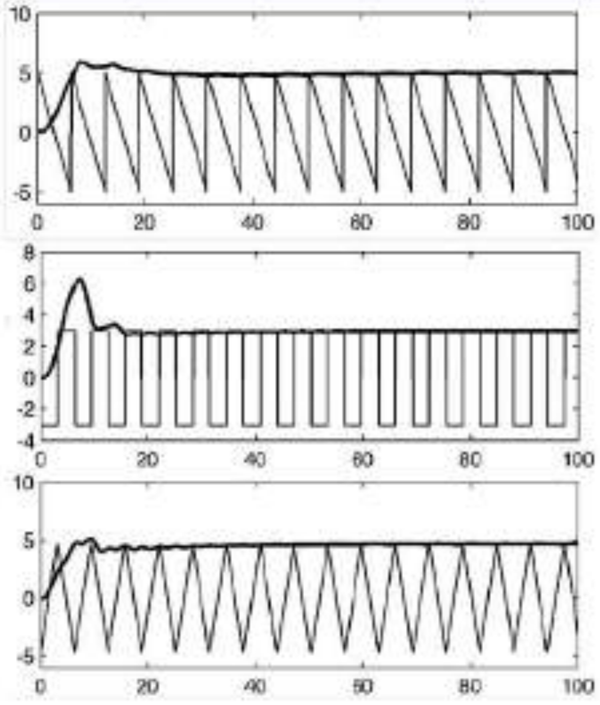
$$\hat{\theta}(t) = \chi(t) + k\xi_1(t)\xi_2(t) \quad (16)$$

$$\dot{\chi}(t) = -k\xi_1^2(t)\hat{\theta}(t) + k\xi_2^2(t) \quad (17)$$

$$\xi_1(s) = \frac{s\lambda^4}{(s+\lambda)^4}y(s) \quad (18)$$

$$\xi_2(s) = s\xi_1(s) \quad (19)$$

Estimation of the amplitude



$$\hat{A}_{sw}(t) = \frac{\pi\hat{\alpha}(t)}{2} \quad (20)$$

$$\hat{A}_{sq}(t) = \frac{\pi\hat{\alpha}(t)}{4} \quad (21)$$

$$\hat{A}_{tr}(t) = \frac{\pi^2\hat{\alpha}(t)}{8} \quad (22)$$

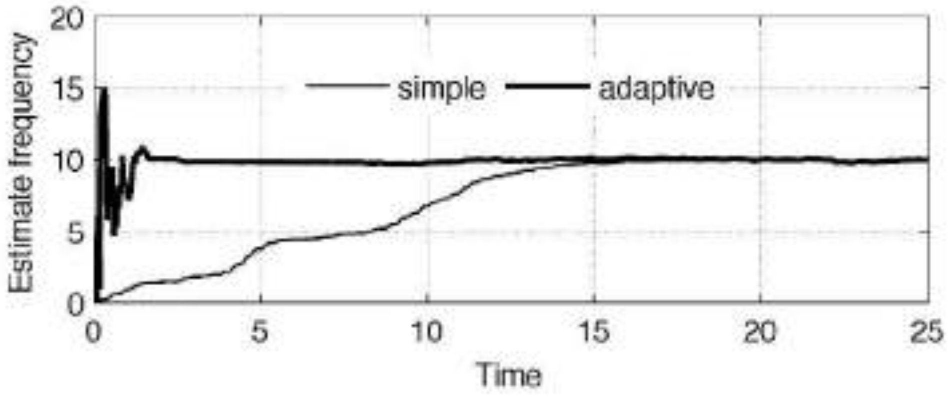
$$\hat{\alpha}(t) = \frac{\hat{\mu}(t)}{\hat{L}_f(t)} \quad (23)$$

$$\hat{\mu}(t) = \sqrt{\xi_1^2(t) + \left(\frac{\xi_2(t)}{\eta(t)}\right)^2} \quad (24)$$

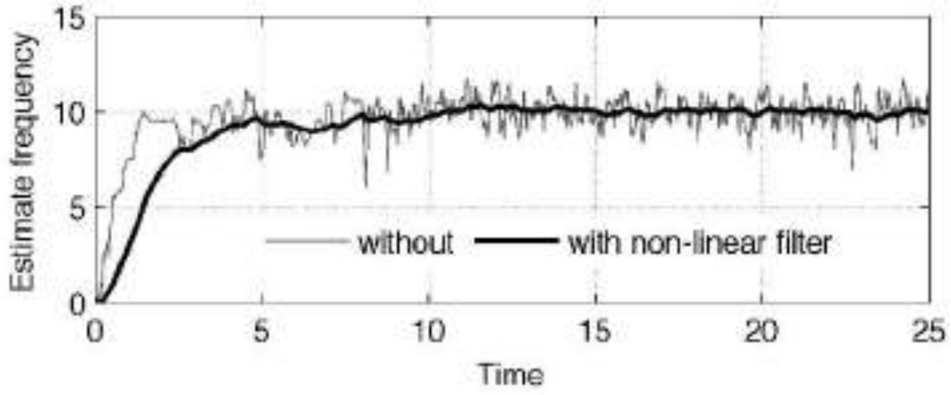
$$\eta(t) = \begin{cases} \omega_{min}, & \hat{\omega} < \omega_{min} \\ \hat{\omega}, & \hat{\omega} \geq \omega_{min} \end{cases} \quad (25)$$

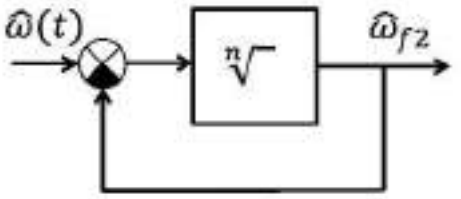
Adaptive algorithm

$$k = 10e^{-3t} + 0.001 \quad (33)$$



Non-linear filtration





$$\widehat{\omega}_{f2}(t) = \int_0^t \sqrt[n]{\widehat{\omega}(\tau) - \widehat{\omega}_{f2}(\tau)}d\tau \quad (34)$$

Conclusions

Algorithm works online in continuous time

Advantages

✓Different types of input signals

✓The algorithm works in the presence of noise

✓Speed and accuracy of convergence can be adjusted

Adaptive and robust control

Outline

- Introduction to adaptive and robust control
- Lyapunov functions method short tutorial
- Simple example of adaptive controller design
- Simple example of robust controller design
- Error models

1. Introduction

Problems and motivation

mathematical models have limited accuracy over the whole range of plants operating

Aircraft

DC motors

DC motor dynamics

$$\dot{I} = -\frac{R}{L}I - \frac{k_E}{L}\omega + \frac{1}{L}U,$$
$$\dot{\omega} = \frac{k_M}{J}I - \frac{1}{J}M_L,$$
$$\dot{\alpha} = \omega$$

Spark ignition engines

Fuel evaporation process dynamics

$$\dot{m}_{ff} = -\frac{1}{T_f}m_{ff} + \frac{K_f}{T_f}m_{fi}$$
$$m_{fc} = m_{ff} + (1 - K_f)m_{fi}$$

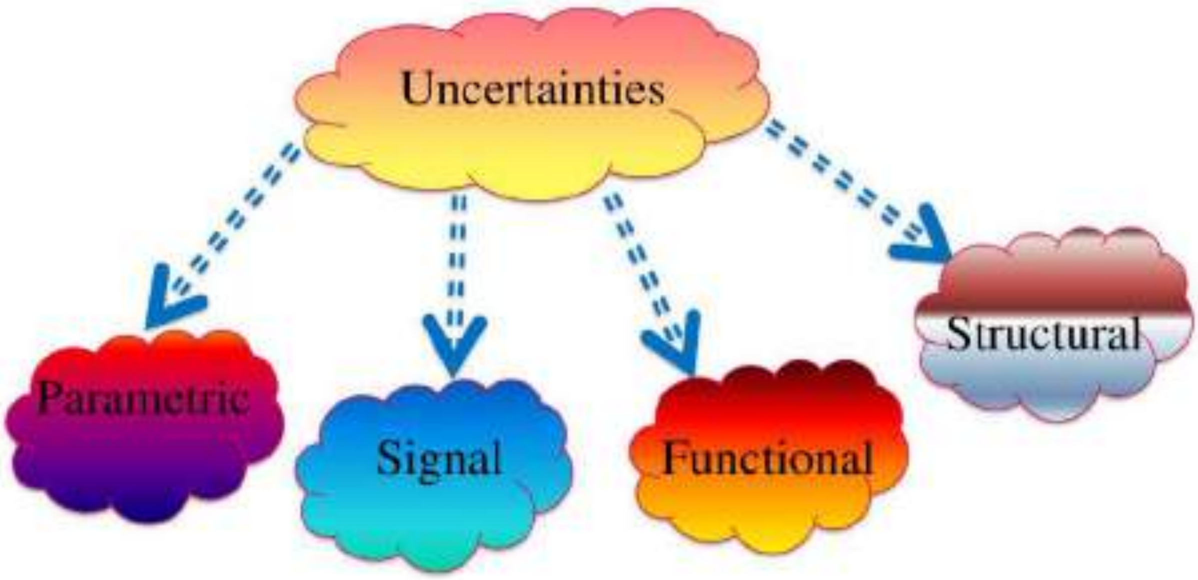
In this context, the approaches of control theory that can come up with the problems of plants uncertainties are of special interest.

Can the control system choose the correct control to improve the performance of the plant operating in presence of uncertainties?

How to design an adaptive control?

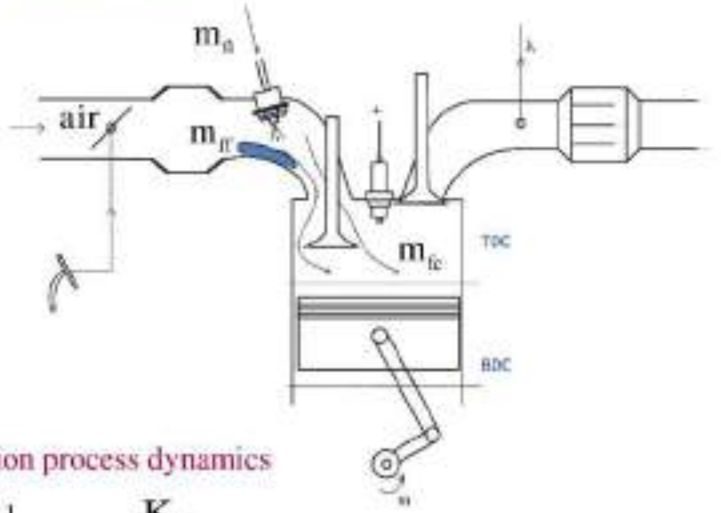
Definitions, clarifications:

1. Model with uncertainties that is the potential basis for controller design belongs to some **class of models** and is called **nominal.**
2. Characteristics of the nominal model are called **nominal.**
3. Uncertainties – unknown or not known precisely characteristics, structure or parameters of the plant .
4. Uncertainties of the plant ≡ uncertainties of the model.



Parametric uncertainties imply that the parameters of the plant model are constant and unknown.

Spark ignition engines



Fuel evaporation process dynamics

$$\dot{m}_{ff} = -\frac{1}{T_f} m_{ff} + \frac{K_f}{T_f} m_{fi}$$

$$m_{fc} = m_{ff} + (1 - K_f) m_{fi}$$

Signal uncertainties imply that the plant model contains unknown functions of time.

DC motors

DC motor dynamics

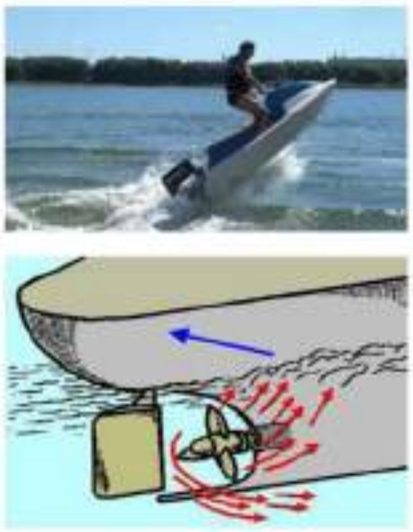
$$\dot{I} = -\frac{R}{L} I - \frac{k_E}{L}\omega + \frac{1}{L} U,$$

$$\dot{\omega} = \frac{k_M}{J} I - \frac{1}{J} M_L,$$

$$\dot{\alpha} = \omega$$

$$\boxed{R = R(temperature) = R(time)}$$

Functional uncertainties imply that plant model contains unknown functions of state.



Tail-shaft dynamics equation

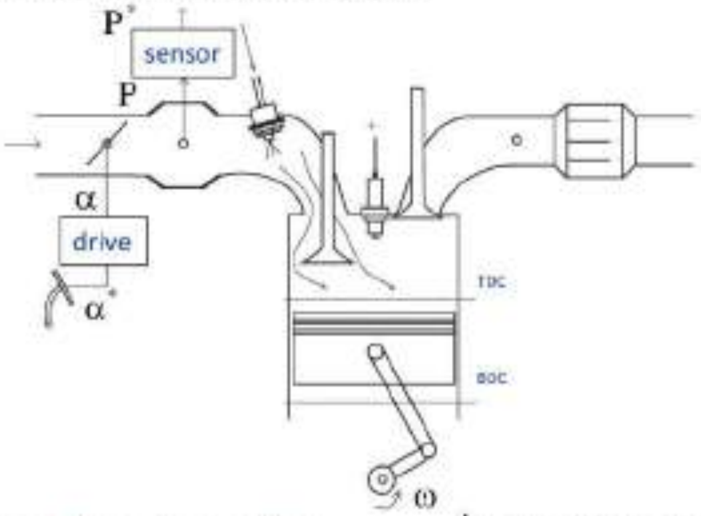
$$J\dot{\omega} = M - M_V$$

M is the engine effective torque

M_V is the viscous friction

$$\boxed{M_V = M_V(\omega) \approx c_0 + c_1\omega + c_2\omega^2}$$

Structural uncertainties imply that the plant model contains unknown structures.



Manifold air pressure dynamics equation: $\dot{\mathbf{P}} + k_1\eta_c(\omega)\mathbf{P} = k_2\eta_t(\mathbf{P})\varphi_1(\mathbf{P})\varphi_2(\boldsymbol{\alpha})$

Pressure sensor dynamics: $\dot{\mathbf{P}}^* = -a\mathbf{P}^* + b\mathbf{P}$

Throttle drive dynamics: $\dot{\alpha} = -c\alpha + d\alpha^*$

Definitions:

Adaptive and robust control are the controls providing desired performance of the plant operating in presence of uncertainties

1. Adaptive control implies the compensation of uncertainties.
2. Robust control does not imply the compensation of uncertainties, but using high gain control.

2. Lyapunov functions method short tutorial

Universal approach of stability analysis for autonomous plants

$$\dot{x} = f(x), \quad x(0), \tag{1}$$

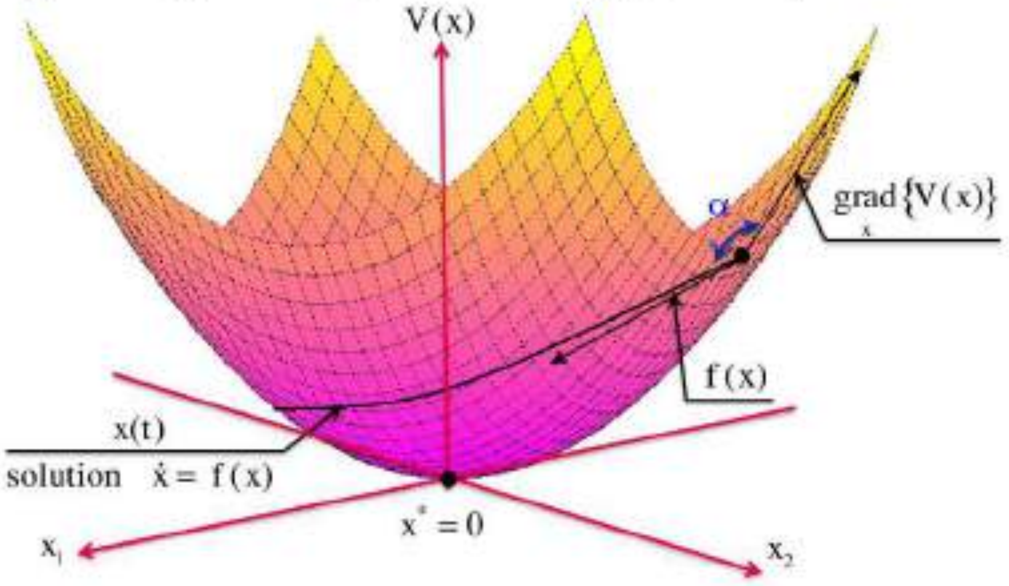
at equilibrium x^*, where $x \in R^n$ is the state vector, $f \in R^n$ is the continuous nonlinear mapping.

Lyapunov functions $V(x)$:

1. $V(x)$ is monotonic ;
2. $V(x) > 0$, if $\|x\| \neq 0$,
   $V(0) = 0$;
3. $V(x) \in C^1$ (continuous and diferentiable) .

Time derivative of Lyapunov function in amount of (1):

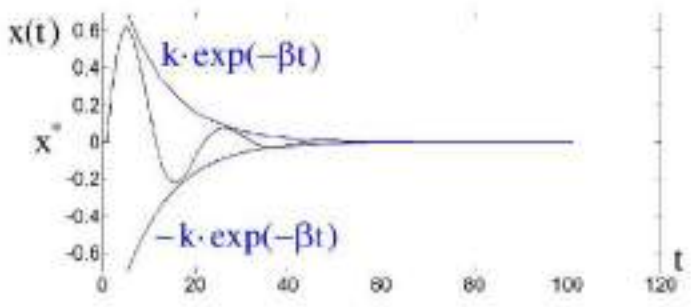
$$\dot{V}(x) = \frac{\partial V(x)}{\partial x}\dot{x} = \frac{\partial V(x)}{\partial x} f(x) = \underset{x}{\text{grad}}\{V(x)\} f(x) = \left\|\underset{x}{\text{grad}}\{V(x)\}\right\| \|f(x)\| \cos\alpha$$



Stability criterias:

1. If $\dot{V}(x) \le 0$, then the equilibrium $x^* = 0$ is Lyapunov stable;
2. If $\dot{V}(x) < 0$, then the equilibrium $x^* = 0$ is asymptotically stable;
3. If $\dot{V}(x) \le -\beta V(x),\ \beta > 0$, then the equilibrium $x^* = 0$ is exponentially stable;

$$\dot{V}(x) \le -\beta V(x) \quad \Rightarrow \quad V(x) \le \exp(-\beta t) V(0)$$



Examples of Lyapunov functions:

1. Linear system

$$\dot{x} = Ax, \quad x(0) \tag{2}$$

where A is the time-invariant matrix.

Lyapunov function candidate

$$V(x) = x^T Px, \tag{3}$$

where $P = P^T \succ 0$ is the time-invariant matrix.

$$\dot{V}(x) = \dot{x}^T Px + x^T P\dot{x} = x^T A^T Px + x^T PAx =$$
$$= x^T \left(A^T P + PA\right)x = -x^T Qx < 0$$

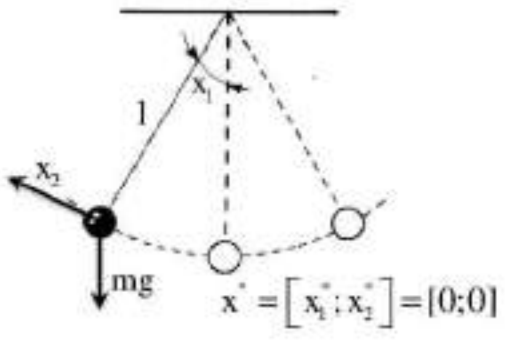
Conclusion: If there exists $P = P^T \succ 0$ such that

$$A^T P + PA = -Q, \tag{4}$$

where $Q = Q^T \succ 0$, system (2) is asymptotically stable.

2. Pendulum

$$\dot{x}_1 = x_2,$$
$$\dot{x}_2 = -gl\sin(x_1) - \frac{k}{m}x_2 \tag{5}$$



where g is the gravity acceleration, l is the length of rod, k is the friction coefficient.

Lyapunov function candidate: **sum of potential and kinetic energy**

$$V(x) = mg\left(1 - \cos(x_1)\right)l + \frac{mx_2^2}{2}. \tag{6}$$

Time derivative:

$$\dot{V}(x) = mg\sin(x_1)\dot{x}_1 l + mx_2\dot{x}_2 = mg\sin(x_1)x_2 l - mx_2 gl\sin(x_1) - mx_2^2.$$

or

$$\dot{V}(x) = -mx_2^2 < 0 \tag{7}$$

Conclusion: pendulum is asymptotically stable at the equilibrium $x^* = [0;0]$.

3. Simple example of adaptive controller design

Motivation

Can the classic theory come up with the problems of uncertainties?

3. Simple example of adaptive controller design

Motivation

Problem statement:

Plant:

$$\dot{x} = \theta x + u, \tag{8}$$

where x is the scalar state, u is the control, θ is the known parameter.

Objective is to design a control providing the following limiting equality:

$$\lim_{t \to \infty} x = 0. \tag{9}$$

Solution:

$$u = -\theta x - \lambda x, \tag{10}$$

where λ is the positive constant parameter.

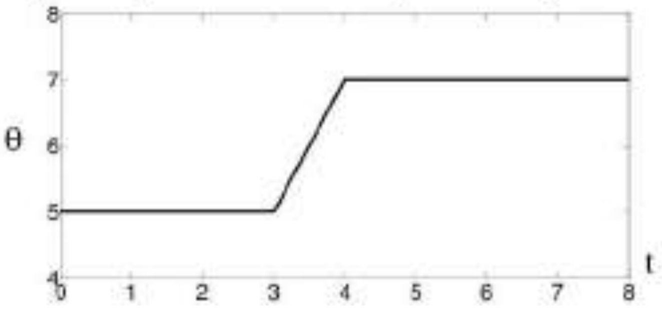
$$u=-\theta x-\lambda x, \quad \dot{x}=\theta x+u \quad \Rightarrow \dot{x}=-\lambda x \Rightarrow x(t)=\exp(-\lambda t)\,x(0). \qquad (11)$$

Let us the design parameter is $\lambda=1$ and plant parameter $\theta=5$, i.e.

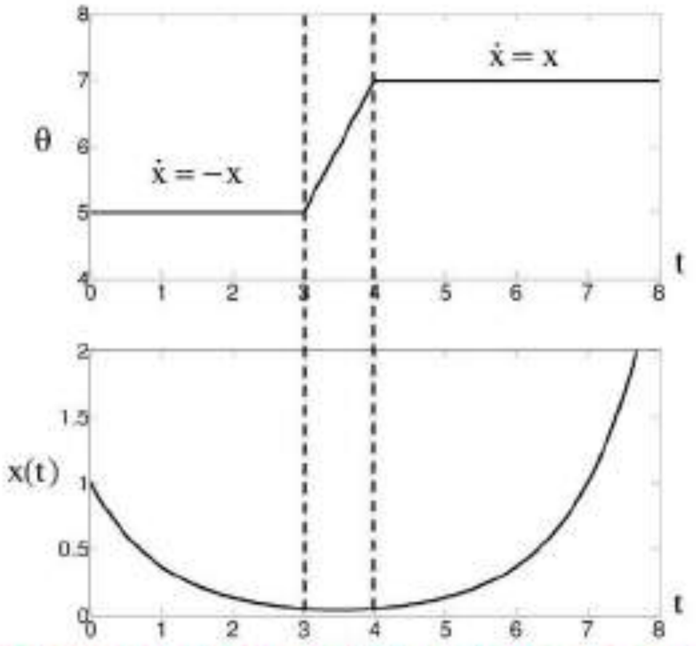
control: $u=-6x$

system $\dot{x}=-x$ is stable.

Now let us imagine, the plant parameter θ unpredictably be changed from 5 to 13:



Control : $u=-6x$



Classical control is not reliable and does not work properly in presence of uncertainties

Problem statement of adaptive control:

Plant:

$$\dot{x} = \theta x + u, \tag{12}$$

where θ is the unknown parameter.

Objective is to design a control providing the following limiting equality:

$$\lim_{t\to\infty} \left(x_M - x \right) = 0, \tag{13}$$

where x_M is the output of reference model

$$\dot{x}_M = -\lambda x_M + \lambda g, \tag{14}$$

g is the reference signal, λ is the positive parameter responsible for transient time.

Solution:

1. Let the parameter θ is known.

Let us for the error signal $\varepsilon = x_M - x$ and take its derivative in amount of plant and reference model equations:

$$\dot{\varepsilon} = \dot{x}_M - \dot{x} = \left(-\lambda x_M + \lambda g \right) - \left(\theta x + u \right)$$

What control to choose to provide the exponential decaying of error?

Solution:

1. Let the parameter θ is known.

Let us for the error signal $\varepsilon = x_M - x$ and take its derivative in amount of plant and reference model equations:

$$\dot{\varepsilon} = \dot{x}_M - \dot{x} = (-\lambda x_M + \lambda g) - (\theta x + u)$$

Let $\dot{\varepsilon} = -\lambda\varepsilon = -\lambda x_M + \lambda x \Rightarrow \varepsilon(t) = \exp(-\lambda t)\varepsilon(0)$. Therefore

$$(-\lambda x_M + \lambda g) - (\theta x + u) = -\lambda x_M + \lambda x$$

$$\Downarrow$$

$$\boxed{u = -\theta x - \lambda x + \lambda g} \qquad (15)$$

Solution:

2. Let the parameter θ is unknown. Therefore the control

$$u = -\theta x - \lambda x + \lambda g$$

is not implementable. Substitute estimate $\hat{\theta}$ for θ and obtain implementable adjustable control:

$$\boxed{u = -\hat{\theta} x - \lambda x + \lambda g} \qquad (16)$$

Substitute (16) into the plant equation $\dot{x} = \theta x + u$:

$$\dot{x} = \theta x - \hat{\theta} x - \lambda x + \lambda g, \qquad (17)$$

Take the derivative of the error

$$\dot{\varepsilon} = \dot{x}_M - \dot{x} = (-\lambda x_M + \lambda g) - \left(\theta x - \hat{\theta} x - \lambda x + \lambda g\right)$$

Signal Error Model $$\boxed{\dot{\varepsilon} = -\lambda\varepsilon - \tilde{\theta} x.} \qquad (18)$$

where $\tilde{\theta} = \theta - \hat{\theta}$ is the parametric error.

Solution:

3. Let us choose the algorithm generating estimate $\hat{\theta}$:

$$\boxed{\dot{\hat{\theta}} = \Omega(t)} \qquad (19)$$

where $\Omega(t)$ is implementable (measurable) function.

Taking into account that $\tilde{\theta} = \theta - \hat{\theta}$ and

$$\dot{\tilde{\theta}} = -\dot{\hat{\theta}}$$

we get

Parametric Error Model

$$\boxed{\dot{\tilde{\theta}} = -\Omega(t)} \qquad (20)$$

How to choose the function $\Omega(t)$???

Solution:

4. Models

Signal Error Model $\boxed{\dot{\varepsilon} = -\lambda\varepsilon - \tilde{\theta}x,}$ (18)

Parametric Error Model $\boxed{\dot{\tilde{\theta}} = -\Omega(t)}$ (20)

Choose the Lyapunov function candidate

$$V(\varepsilon, \tilde{\theta}) = \frac{1}{2}\varepsilon^2 + \frac{1}{2\gamma}\tilde{\theta}^2, \qquad \gamma > 0 \qquad (21)$$

and take its time derivative using (18) and (20):

$$\dot{V}(\varepsilon, \tilde{\theta}) = \varepsilon\dot{\varepsilon} + \frac{1}{\gamma}\tilde{\theta}\dot{\tilde{\theta}} = -\lambda\varepsilon^2 - \tilde{\theta}x\varepsilon - \frac{1}{\gamma}\tilde{\theta}\Omega(t)$$

If $\Omega(t) = -\gamma x\varepsilon$ then $\dot{V}(\varepsilon, \tilde{\theta}) = -\lambda\varepsilon^2 < 0$

$$\boxed{\dot{\hat{\theta}} = -\gamma x\varepsilon} \qquad (22)$$

Summary

Adjustable controller:

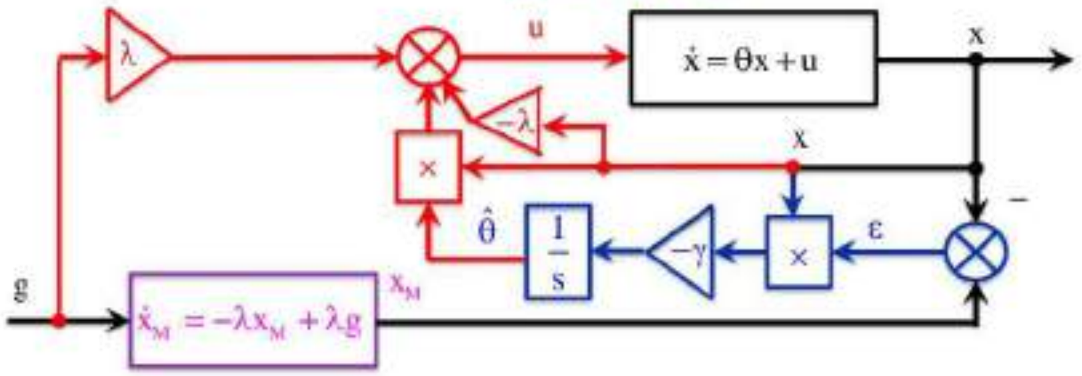
$$u = -\hat{\theta}x - \lambda x + \lambda g \tag{16}$$

Adaptation algorithm:

$$\dot{\hat{\theta}} = -\gamma x \varepsilon \tag{22}$$

with $\varepsilon = x_M - x$ and reference model

$$\dot{x}_M = -\lambda x_M + \lambda g. \tag{14}$$



Summary

Properties of the closed-loop system:

1. All signals in the system are bounded;
2. Control error $\varepsilon = x_M - x$ asymptotically tends to zero;
3. Parametric error $\tilde{\theta} = \theta - \hat{\theta}$ in general case tends to a constant;

$$V(\varepsilon, \tilde{\theta}) = \frac{1}{2}\varepsilon^2 + \frac{1}{2\gamma}\tilde{\theta}^2,$$

$$\dot{V}(\varepsilon, \tilde{\theta}) = -\lambda\varepsilon^2 < 0$$

4. There is an optimal adaptation gain γ corresponding the fastest parametrical convergence;

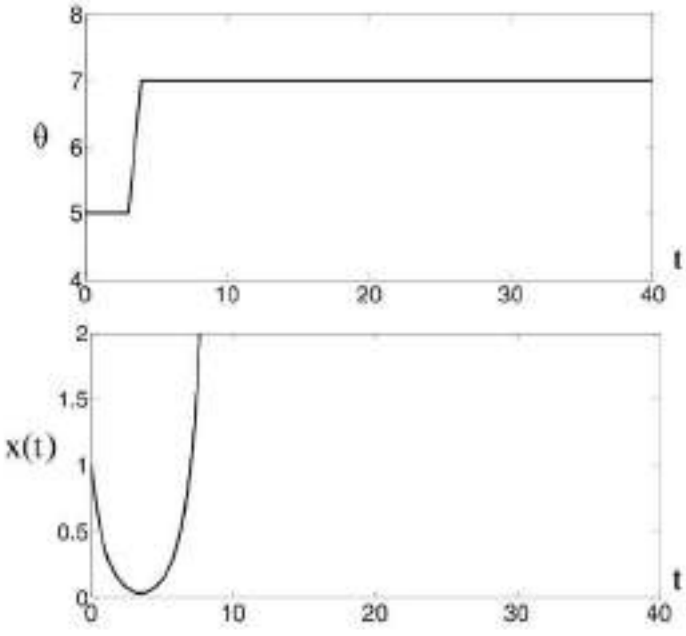
5. There can be parametric drift phenomena in presence of noise, i.e., if

$$\dot{x} = \theta x + u + \delta,$$

where δ is bounded disturbance, $\hat{\theta} \to \infty$.

Example: Classical stabilizing control for unstable plant $\dot{x} = 5x + u$

$$u = -6x$$

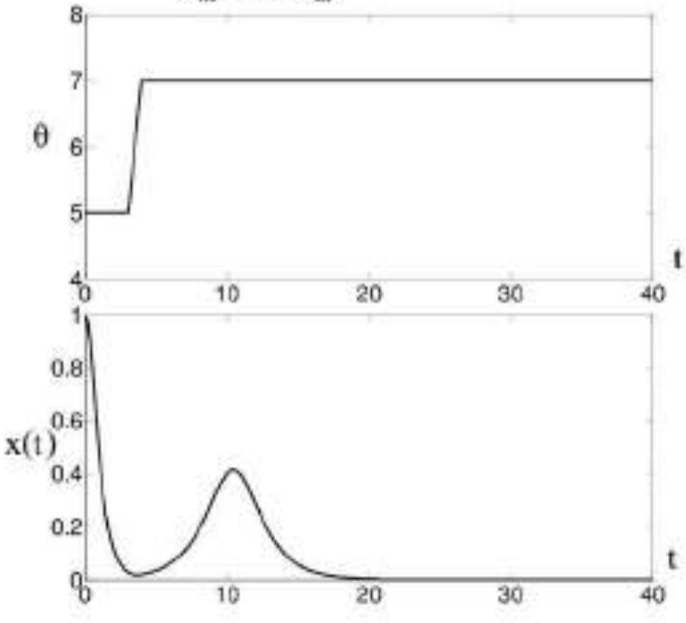


Example: Adaptive stabilizing control for unstable plant $\dot{x} = 5x + u$

$$u = -6x - \hat{\theta}x,$$

$$\dot{\hat{\theta}} = -2x\varepsilon, \quad \varepsilon = x_M - x,$$

$$\dot{x}_M = -6x_M$$

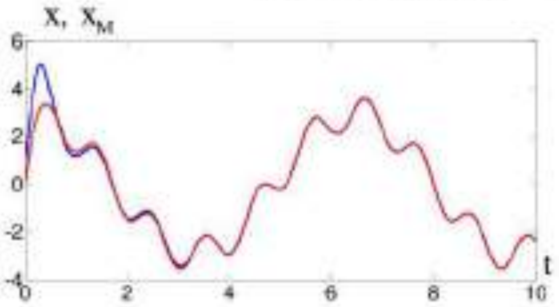


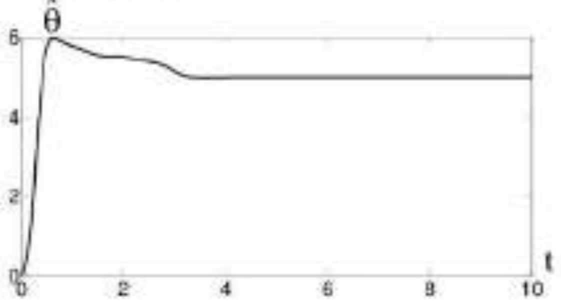
Example: Adaptive tracking control for unstable plant $\dot{x} = 5x + u$

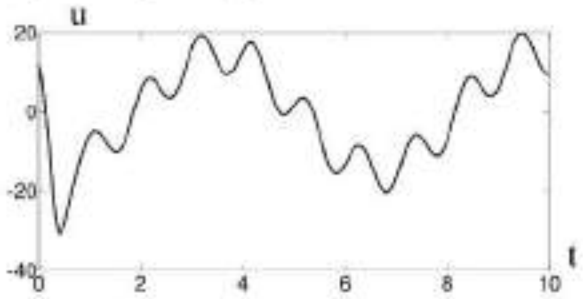
$$u = -6x - \hat{\theta}x + 6g,$$

$$\dot{\hat{\theta}} = -2x\varepsilon, \quad \varepsilon = x_M - x,$$

$$\dot{x}_M = -6x_M + 6g, \quad g = \sin 6t + 3\cos t$$







Definitions:

Adaptive and robust control are the controls providing desired performance of the plant operating in presence of uncertainties

1. Adaptive control implies the compensation of uncertainties.

2. Robust control does not imply the compensation of uncertainties, but using high gain control.

4. Simple example of robust controller design

Problem statement of adaptive control:

Plant:

$$\dot{x} = \theta x + u + \delta, \qquad |\delta| \le \bar{\delta} \tag{23}$$

where θ is the unknown parameter, $\delta(t)$ is unpredictable bounded noise.

Objective is to design a control providing the following inequality:

$$|x_M(t) - x(t)| \le \Delta \text{ for any } t \ge T, \tag{24}$$

where x_M is the output of reference model

$$\dot{x}_M = -\lambda x_M + \lambda g, \tag{25}$$

g is the reference signal, λ is the positive parameter responsible for transient time.

Solution:

How to prevent unbounded growth of the estimates $\hat{\theta}$ in presence of noise in adaptation algorithm $\dot{\hat{\theta}} = -\gamma x \varepsilon$?

Solution:

Proposition 1.

Adjustable controller:

$$u = -\hat{\theta}x - \lambda x + \lambda g \tag{26}$$

~~Adaptation algorithm:~~ $\longrightarrow$ **Nonlinear static feedback:**

$$\hat{\theta} = -\gamma x\varepsilon \tag{27}$$

with $\varepsilon = x_M - x$ and reference model

$$\dot{x}_M = -\lambda x_M + \lambda g. \tag{28}$$

Substitution of (27) into (26) gives "high-gain" type controller:

$$u = \gamma x^2\varepsilon - \lambda x + \lambda g.$$

Then substitute this control into disturbed plant $\dot{x} = \theta x + u + \delta$.

$$\dot{x} = \theta x + \gamma x^2\varepsilon - \lambda x + \lambda g + \delta.$$

Solution:

Proposition 1.

Again, take the derivative of the error $\varepsilon = x_M - x$

$$\dot{\varepsilon} = \dot{x}_M - \dot{x} = \left(-\lambda x_M + \lambda g\right) - \left(\theta x + \gamma x^2\varepsilon - \lambda x + \lambda g + \delta\right)$$

$$\boxed{\dot{\varepsilon} = -\lambda\varepsilon - \theta x - \gamma x^2\varepsilon - \delta} \tag{29}$$

Choose the Lyapunov function candidate

$$V(\varepsilon, \tilde{\theta}) = \frac{1}{2}\varepsilon^2$$

and take its time derivative using (29):

$$\dot{V}(\varepsilon, \tilde{\theta}) = \varepsilon\dot{\varepsilon} = -\lambda\varepsilon^2 - \theta x\varepsilon - \gamma x^2\varepsilon^2 - \delta\varepsilon = -\frac{\lambda}{2}\varepsilon^2 - \frac{\lambda}{2}\varepsilon^2 - \theta x\varepsilon - \gamma x^2\varepsilon^2 - \delta\varepsilon =$$

Solution:

Proposition 1.

Again, take the derivative of the error $\varepsilon = x_M - x$

$$\dot{\varepsilon} = \dot{x}_M - \dot{x} = \left(-\lambda x_M + \lambda g\right) - \left(\theta x + \gamma x^2 \varepsilon - \lambda x + \lambda g + \delta\right)$$

$$\boxed{\dot{\varepsilon} = -\lambda\varepsilon - \theta x - \gamma x^2 \varepsilon - \delta} \qquad (29)$$

Choose the Lyapunov function candidate

$$V(\varepsilon) = \frac{1}{2}\varepsilon^2$$

and take its time derivative using (29):

$$\dot{V}(\varepsilon) = \varepsilon\dot{\varepsilon} = -\lambda\varepsilon^2 - \theta x\varepsilon - \gamma x^2\varepsilon^2 - \delta\varepsilon = -\frac{\lambda}{2}\varepsilon^2 - \frac{\lambda}{2}\varepsilon^2 - \theta x\varepsilon - \gamma x^2\varepsilon^2 - \delta\varepsilon =$$

$$= -\frac{\lambda}{2}\varepsilon^2 - \frac{\lambda}{2}\varepsilon^2 - \delta\varepsilon \pm \frac{1}{2\lambda}\delta^2 - \gamma x^2\varepsilon^2 - \theta x\varepsilon \pm \frac{\theta^2}{4\gamma}$$

Solution:

Proposition 1.

$$\dot{V}(\varepsilon) = -\frac{\lambda}{2}\varepsilon^2 - \left(\sqrt{\frac{\lambda}{2}}\varepsilon + \sqrt{\frac{1}{2\lambda}}\delta\right)^2 + \frac{1}{2\lambda}\delta^2 - \left(\sqrt{\gamma}x\varepsilon + \frac{\theta}{2\sqrt{\gamma}}\right)^2 + \frac{\theta^2}{4\gamma}$$

$$\dot{V}(\varepsilon) \le -\frac{\lambda}{2}\varepsilon^2 + \frac{1}{2\lambda}\delta^2 + \frac{\theta^2}{4\gamma}$$

$$\dot{V}(\varepsilon) \le -\frac{\lambda}{2}\varepsilon^2 + \frac{1}{2\lambda}\bar{\delta}^2 + \frac{\theta^2}{4\gamma}$$

$|\delta(t)| \le \bar{\delta}$

$$\dot{V}(\varepsilon) \le -\frac{\lambda}{2}\varepsilon^2 + \bar{\Delta}$$

where $\bar{\Delta} = \frac{1}{2\lambda}\bar{\delta}^2 + \frac{\theta^2}{4\gamma}$ is constant

Solution:

Proposition 1.

$$\dot{V}(\varepsilon) = -\frac{\lambda}{2}\varepsilon^2 - \left(\sqrt{\frac{\lambda}{2}}\varepsilon + \sqrt{\frac{1}{2\lambda}}\delta\right)^2 + \frac{1}{2\lambda}\delta^2 - \left(\sqrt{\gamma}x\varepsilon + \frac{\theta}{2\sqrt{\gamma}}\right)^2 + \frac{\theta^2}{4\gamma}$$

$$\dot{V}(\varepsilon) \le -\frac{\lambda}{2}\varepsilon^2 + \frac{1}{2\lambda}\delta^2 + \frac{\theta^2}{4\gamma}$$

$$\dot{V}(\varepsilon) \le -\frac{\lambda}{2}\varepsilon^2 + \frac{1}{2\lambda}\bar{\delta}^2 + \frac{\theta^2}{4\gamma}$$

$$|\delta(t)| \le \bar{\delta}$$

$$\bar{\Delta} = \frac{1}{2\lambda}\bar{\delta}^2 + \frac{\theta^2}{4\gamma}$$

where $\bar{\Delta} = \frac{1}{2\lambda}\bar{\delta}^2 + \frac{\theta^2}{4\gamma}$ is constant

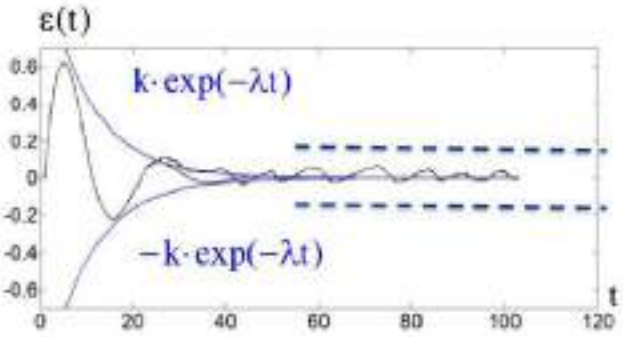
$$\boxed{\dot{V}(\varepsilon) \le -\lambda V(\varepsilon) + \bar{\Delta}} \qquad V(\varepsilon) = \frac{1}{2}\varepsilon^2 \quad (30)$$

Solution:

Proposition 1.

$$\dot{V}(\varepsilon) \le -\lambda V(\varepsilon) + \bar{\Delta} \qquad \Rightarrow \qquad V(t) \le \exp(-\lambda t)V(0)\left(1 - \frac{\bar{\Delta}}{\lambda}\right) + \frac{\bar{\Delta}}{\lambda}V(0)$$

Exponential convergence of ε to bounded set is proved.



Summary

Adjustable controller:

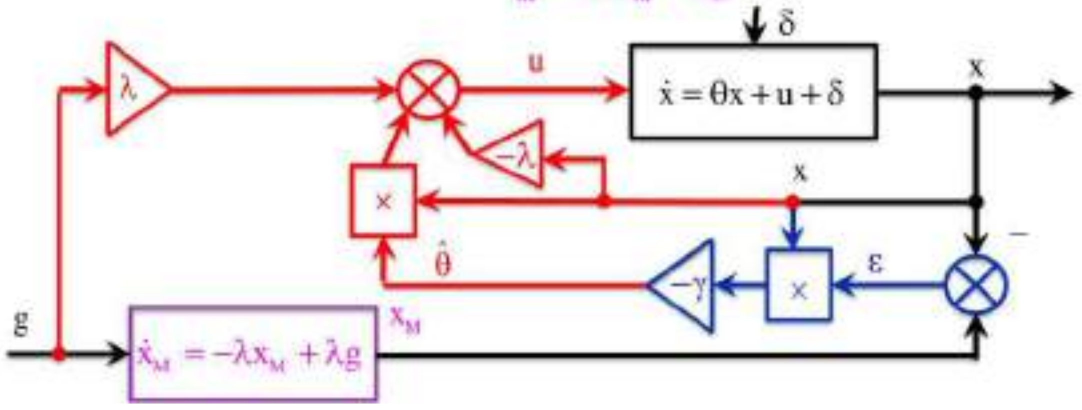
$$u = -\hat{\theta}x - \lambda x + \lambda g \qquad (26)$$

Nonlinear static feedback :

$$\dot{\hat{\theta}} = -\gamma x \varepsilon \qquad (27)$$

with $\varepsilon = x_M - x$ and reference model

$$\dot{x}_M = -\lambda x_M + \lambda g. \qquad (28)$$


δ
u
$\dot{x} = \theta x + u + \delta$
x
λ
−λ
×
x
$\hat{\theta}$
−γ
×
ε
−
g
x_M
$\dot{x}_M = -\lambda x_M + \lambda g$

Summary

Properties of the closed-loop robust system:

1. All signals in the system are bounded;
2. Control error $\varepsilon = x_M - x$ exponentially tends to the neighborhood of zero;
3. The radius of neighborhood can be arbitrary reduced by

How?

$$\dot{V}(\varepsilon) \le -\lambda V(\varepsilon) + \bar{\Delta} \quad \text{where} \quad \bar{\Delta} = \frac{1}{2\lambda}\bar{\delta}^2 + \frac{\theta^2}{4\gamma}$$

Summary

Properties of the closed-loop robust system:

1. All signals in the system are bounded;
2. Control error $\varepsilon = x_M - x$ exponentially tends to the neighborhood of zero;
3. The radius of neighborhood can be arbitrary reduced by

λ or γ

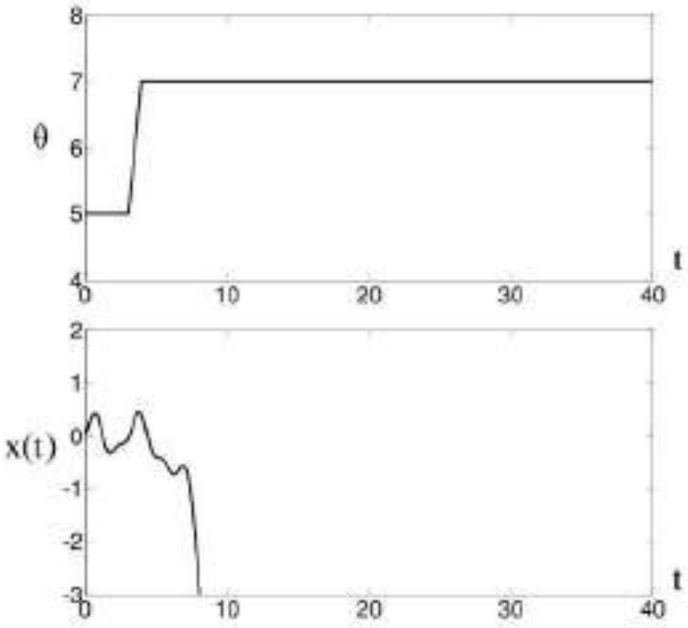
$$\dot V(\varepsilon) \le -\lambda V(\varepsilon) + \bar\Delta \quad \text{where} \quad \bar\Delta = \frac{1}{2\lambda}\bar{\bar\delta}^2 + \frac{\theta^2}{4}$$

4. There is no compensation of uncertainty!
   Even, if the plant is not disturbed ($\delta = \bar\delta = 0$), the error $\varepsilon = x_M - x$ does not go to zero!

Example: Classical stabilizing control for unstable plant $\boxed{\dot x = \theta x + u + \delta}$

$u = -6x$ $\boxed{\theta = 5}$

$\boxed{\delta = 0{,}5\sin(4t) + 0{,}75\cos(2t)}$



Example: Robust stabilizing control for unstable plant

$\boxed{\dot{x} = \theta x + u + \delta}$

$$u = -6x - \hat{\theta}x,$$

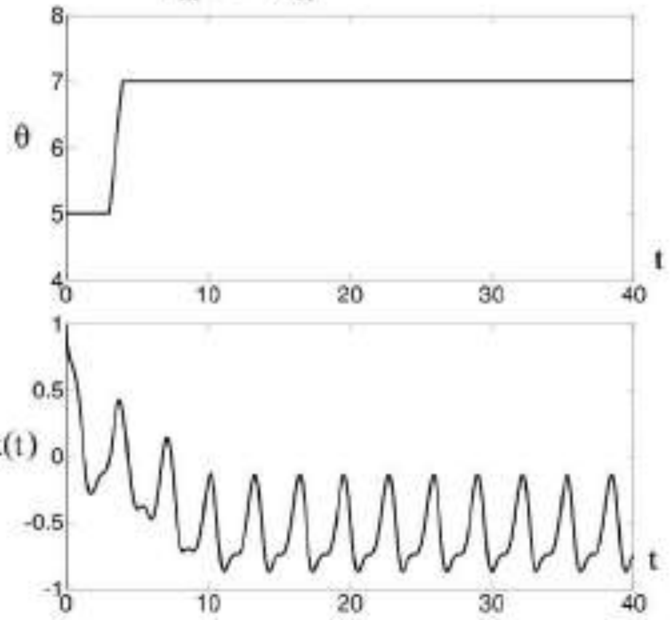
$\boxed{\theta = 5}$

$$\dot{\hat{\theta}} = -\gamma x \varepsilon, \qquad \varepsilon = x_M - x,$$

$\boxed{\delta = 0{,}5\sin(4t) + 0{,}75\cos(2t)}$

$$\dot{x}_M = -6x_M$$

$\gamma = 2$



Example: Robust stabilizing control for unstable plant

$\boxed{\dot{x} = \theta x + u + \delta}$

$$u = -6x - \hat{\theta}x,$$

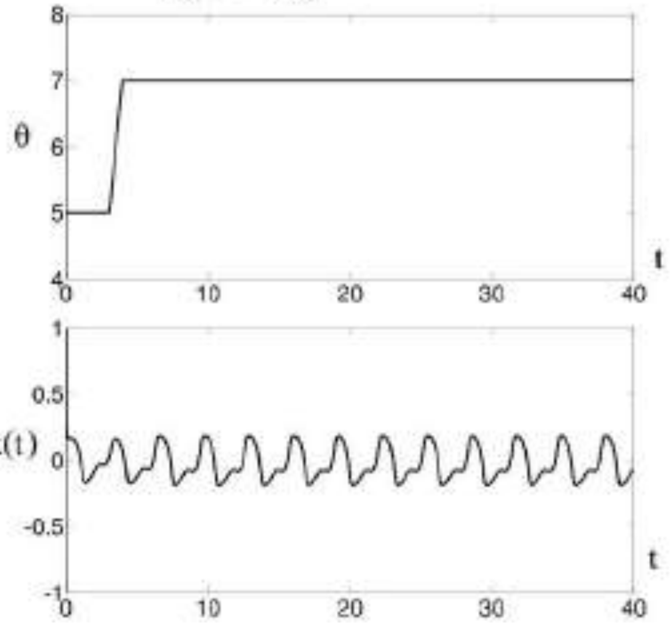
$\boxed{\theta = 5}$

$$\dot{\hat{\theta}} = -\gamma x \varepsilon, \qquad \varepsilon = x_M - x,$$

$\boxed{\delta = 0{,}5\sin(4t) + 0{,}75\cos(2t)}$

$$\dot{x}_M = -6x_M$$

$\gamma = 200$



Example: Robust tracking control for unstable plant $\boxed{\dot{x} = \theta x + u + \delta}$

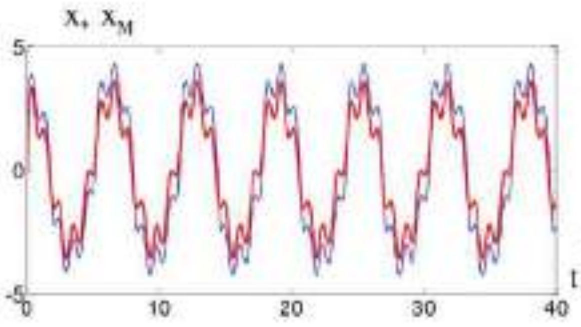
$$u = -6x - \hat{\theta}x + 6g,$$

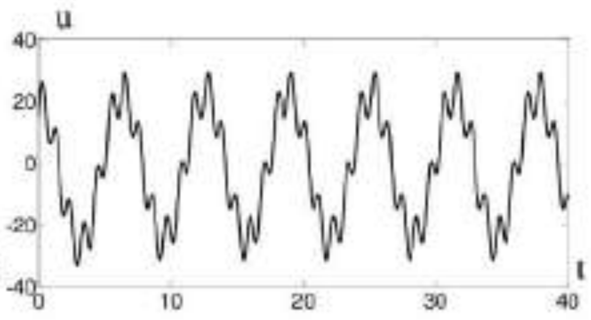
$\boxed{\theta = 5}$

$$\dot{\hat{\theta}} = -2x\varepsilon, \quad \varepsilon = x_M - x,$$

$\boxed{\delta = 0{,}5\sin(4t) + 0{,}75\cos(2t)}$

$$\dot{x}_M = -6x_M + 6g, \quad g = \sin 6t + 3\cos t$$





Adaptive control provides the complete compensation of uncertainties,

but can be not reliable under disturbance condition $\hat{\theta} \to \infty$

Adaptive control provides the complete compensation of uncertainties,

but can be not reliable under disturbance condition $\hat{\theta} \to \infty$

trade off ?

Robust control guarantee the strongest exponential stability,

but does not compensate the uncertainties, therefore $\varepsilon \not\to 0$

Solution:

Proposition 2

Adjustable controller:

$$u = -\hat{\theta}x - \lambda x + \lambda g \tag{31}$$

~~Adaptation algorithm:~~ $\longrightarrow$ **Robust modification of AA:**

$$\dot{\hat{\theta}} = -\gamma x \varepsilon - \sigma \hat{\theta} \tag{32}$$

where σ is a positive feedback gain,

$\varepsilon = x_M - x$, x_M is the output of reference model

$$\dot{x}_M = -\lambda x_M + \lambda g. \tag{33}$$

Then substitute control (31) into disturbed plant $\dot{x} = \theta x + u + \delta$.

$$\dot{x} = \theta x - \hat{\theta} x - \lambda x + \lambda g + \delta.$$

$$\dot{x} = \tilde{\theta} x - \lambda x + \lambda g + \delta. \qquad (\tilde{\theta} = \theta - \hat{\theta})$$

Solution:

Proposition 2.

Again, form take the derivative of the error $\varepsilon = x_M - x$

$$\dot\varepsilon = \dot x_M - \dot x = \left(-\lambda x_M + \lambda g\right) - \left(\tilde\theta x - \lambda x + \lambda g + \delta\right)$$

Signal Error Model $\boxed{\dot\varepsilon = -\lambda\varepsilon - \tilde\theta x - \delta}$ (34)

$$\dot{\hat\theta} = -\gamma x\varepsilon - \sigma\hat\theta \longrightarrow \dot{\tilde\theta} = -\dot{\hat\theta}$$

Parametric Error Model $\boxed{\dot{\tilde\theta} = \gamma x\varepsilon + \sigma\hat\theta}$ (35)

Choose the Lyapunov function candidate

$$V(\varepsilon,\tilde\theta) = \frac{1}{2}\varepsilon^2 + \frac{1}{2\gamma}\tilde\theta^2, \qquad \gamma > 0 \tag{36}$$

Solution:

Proposition 2.

Take the time derivative of Lyapunov function using (34) and (35):

Signal Error Model $\boxed{\dot\varepsilon = -\lambda\varepsilon - \tilde\theta x - \delta}$

Parametric Error Model $\boxed{\dot{\tilde\theta} = \gamma x\varepsilon + \sigma\hat\theta}$

$$\dot V(\varepsilon,\tilde\theta) = \varepsilon\dot\varepsilon + \frac{1}{\gamma}\tilde\theta\dot{\tilde\theta} = -\lambda\varepsilon^2 - \tilde\theta x\varepsilon - \frac{1}{\gamma}\tilde\theta\Omega(t)$$

$$\dot V(\varepsilon,\tilde\theta) = \varepsilon\dot\varepsilon + \frac{1}{\gamma}\tilde\theta\dot{\tilde\theta} = \left(-\lambda\varepsilon^2 - \tilde\theta x\varepsilon - \delta\varepsilon\right) + \frac{1}{\gamma}\tilde\theta\left(\gamma x\varepsilon + \sigma\hat\theta\right)$$

$$\dot V(\varepsilon,\tilde\theta) = -\lambda\varepsilon^2 - \delta\varepsilon + \frac{\sigma}{\gamma}\tilde\theta\hat\theta$$

$$\dot V(\varepsilon,\tilde\theta) = -\lambda\varepsilon^2 - \delta\varepsilon - \frac{\sigma}{\gamma}\tilde\theta^2 + \frac{\sigma}{\gamma}\tilde\theta\theta \longleftarrow \tilde\theta = \theta - \hat\theta$$

Solution:

Proposition 2.

$$\dot{V}(\varepsilon,\tilde{\theta}) = -\frac{\lambda}{2}\varepsilon^2 \boxed{-\frac{\lambda}{2}\varepsilon^2 - \delta\varepsilon} - \frac{\sigma}{2\gamma}\tilde{\theta}^2 \boxed{-\frac{\sigma}{2\gamma}\tilde{\theta}^2 + \frac{\sigma}{\gamma}\tilde{\theta}\theta}$$

$$\dot{V}(\varepsilon,\tilde{\theta}) = -\frac{\lambda}{2}\varepsilon^2 - \frac{\sigma}{2\gamma}\tilde{\theta}^2 \boxed{-\frac{\lambda}{2}\varepsilon^2 - \delta\varepsilon \pm \frac{1}{2\lambda}\delta^2} \boxed{-\frac{\sigma}{2\gamma}\tilde{\theta}^2 + \frac{\sigma}{\gamma}\tilde{\theta}\theta \pm \frac{\sigma}{2\gamma}\theta^2}$$

$$\dot{V}(\varepsilon,\tilde{\theta}) = -\frac{\lambda}{2}\varepsilon^2 - \frac{\sigma}{2\gamma}\tilde{\theta}^2 \boxed{-\left(\sqrt{\frac{\lambda}{2}}\varepsilon + \sqrt{\frac{1}{2\lambda}}\delta\right)^2 + \frac{1}{2\lambda}\delta^2} \boxed{-\frac{\sigma}{2\gamma}\left(\tilde{\theta}-\theta\right)^2 + \frac{\sigma}{2\gamma}\theta^2}$$

$$\dot{V}(\varepsilon,\tilde{\theta}) \le -\frac{\lambda}{2}\varepsilon^2 - \frac{\sigma}{2\gamma}\tilde{\theta}^2 + \frac{1}{2\lambda}\delta^2 + \frac{\sigma}{2\gamma}\theta^2$$

$$\boxed{|\delta(t)| \le \bar{\delta}}$$

$$\dot{V}(\varepsilon,\tilde{\theta}) \le -\frac{\lambda}{2}\varepsilon^2 - \frac{\sigma}{2\gamma}\tilde{\theta}^2 + \frac{1}{2\lambda}\bar{\delta}^2 + \frac{\sigma}{2\gamma}\theta^2$$

Solution:

Proposition 2.

$$\dot{V}(\varepsilon,\tilde{\theta}) \le -\frac{\lambda}{2}\varepsilon^2 - \frac{\sigma}{2\gamma}\tilde{\theta}^2 + \frac{1}{2\lambda}\bar{\delta}^2 + \frac{\sigma}{2\gamma}\theta^2$$

$$\dot{V}(\varepsilon,\tilde{\theta}) \le -\frac{\lambda}{2}\varepsilon^2 - \frac{\sigma}{2\gamma}\tilde{\theta}^2 + \bar{\Delta}$$

where $\bar{\Delta} = \frac{1}{2\lambda}\bar{\delta}^2 + \frac{\sigma}{2\gamma}\theta^2$ is a constant.

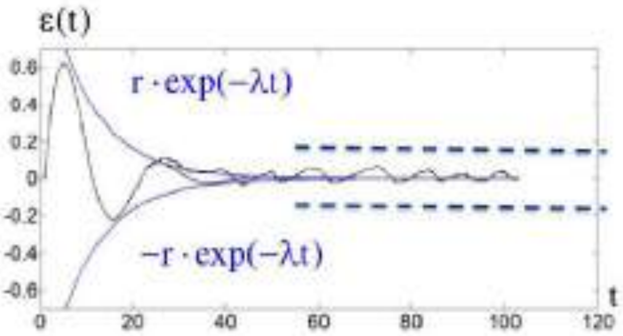
$$\dot{V}(\varepsilon,\tilde{\theta}) \le -kV(\varepsilon,\tilde{\theta}) + \bar{\Delta} \qquad k = \min\left\{\lambda, \frac{\sigma}{\gamma}\right\} \qquad (37)$$

Solution:

Proposition 2.

$$\dot V(\varepsilon) \le -kV(\varepsilon) + \bar\Delta \qquad \Rightarrow \qquad V(t) \le \exp(-kt)V(0)\left(1-\frac{\bar\Delta}{k}\right) + \frac{\bar\Delta}{k}V(0)$$

Exponential convergence of ε to bounded set is proved.



Summary

Adjustable controller:

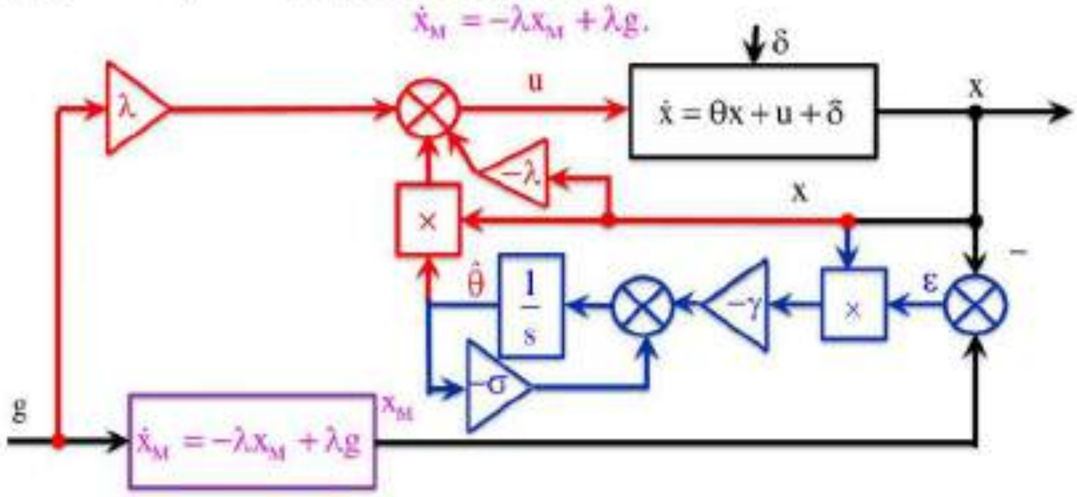
$$u = -\hat\theta x - \lambda x + \lambda g \tag{31}$$

Robust modification of adaptation algorithm:

$$\dot{\hat\theta} = -\gamma x\varepsilon - \sigma\hat\theta \tag{32}$$

with $\varepsilon = x_M - x$ and reference model

$$\dot x_M = -\lambda x_M + \lambda g. \tag{33}$$



Summary

Properties of the closed-loop robust system:

1. All signals in the system are bounded;
2. Control error $\varepsilon = x_M - x$ exponentially tends to the neighborhood of zero;
3. The radius of neighborhood can be arbitrary reduced by

How?

$$\dot{V}(\varepsilon) \le -kV(\varepsilon) + \bar{\Delta} \quad \text{where} \quad \bar{\Delta} = \frac{1}{2\lambda}\bar{\delta}^2 + \frac{\sigma}{2\gamma}\theta^2$$

Summary

Properties of the closed-loop robust system:

1. All signals in the system are bounded;
2. Control error $\varepsilon = x_M - x$ exponentially tends to the neighborhood of zero;
3. The radius of neighborhood can be arbitrary reduced by

λ or γ or $\sigma \downarrow$

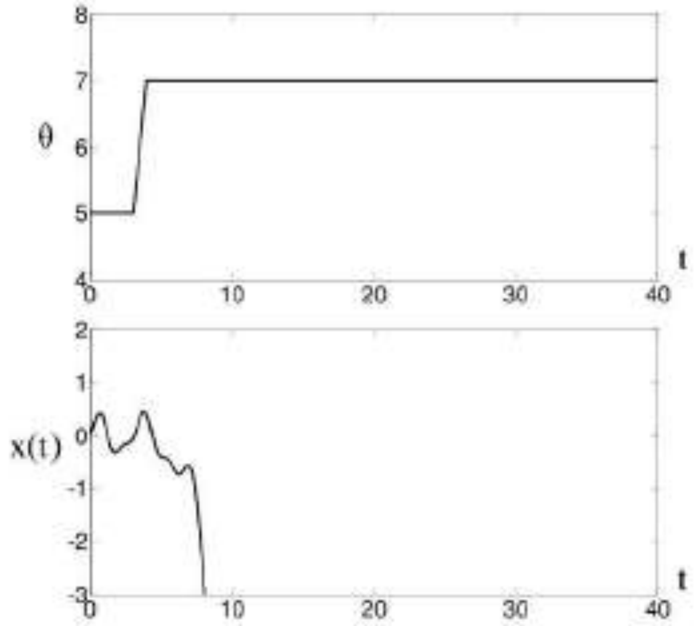
$$\dot{V}(\varepsilon) \le -kV(\varepsilon) + \bar{\Delta} \quad \text{where} \quad \bar{\Delta} = \frac{1}{2\lambda}\bar{\delta}^2 + \frac{\sigma}{2\gamma}\theta^2$$

4. Algorithm provides the compensation of uncertainty.
   If the plant is not disturbed ($\delta = \bar{\delta} = 0$), the error $\varepsilon = x_M - x$ can go to zero, if $\sigma = 0$.

Example: Classical stabilizing control for unstable plant $\boxed{\dot{x} = \theta x + u + \delta}$

$$u = -6x \qquad \boxed{\theta = 5}$$

$$\boxed{\delta = 0{,}5\sin(4t) + 0{,}75\cos(2t)}$$



Example: Adaptive robust stabilizing control for the plant $\boxed{\dot{x} = \theta x + u + \delta}$

$$u = -6x - \hat{\theta}x,$$

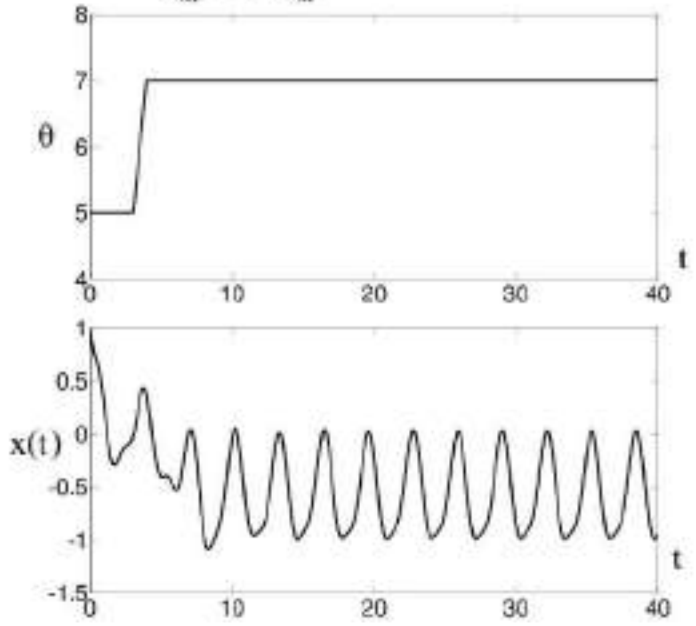
$$\dot{\hat{\theta}} = -\gamma x\varepsilon - \hat{\theta}, \qquad \varepsilon = x_M - x,$$

$$\dot{x}_M = -6x_M$$

$$\boxed{\theta = 5}$$

$$\boxed{\delta = 0{,}5\sin(4t) + 0{,}75\cos(2t)}$$

$\gamma = 2$



Example: Adaptive robust stabilizing control for the plant $\boxed{\dot{x} = \theta x + u + \delta}$

$$u = -6x - \hat{\theta}x,$$

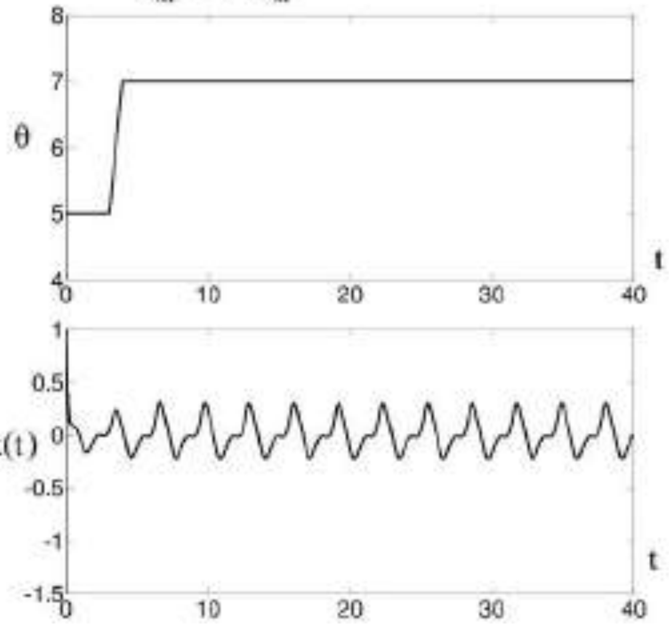
$\boxed{\theta = 5}$

$$\dot{\hat{\theta}} = -\gamma x \varepsilon - \hat{\theta}, \qquad \varepsilon = x_M - x,$$

$\boxed{\delta = 0{,}5\sin(4t) + 0{,}75\cos(2t)}$

$$\dot{x}_M = -6x_M$$

$\gamma = 200$



Example: Adaptive robust tracking control for the plant $\boxed{\dot{x} = \theta x + u + \delta}$

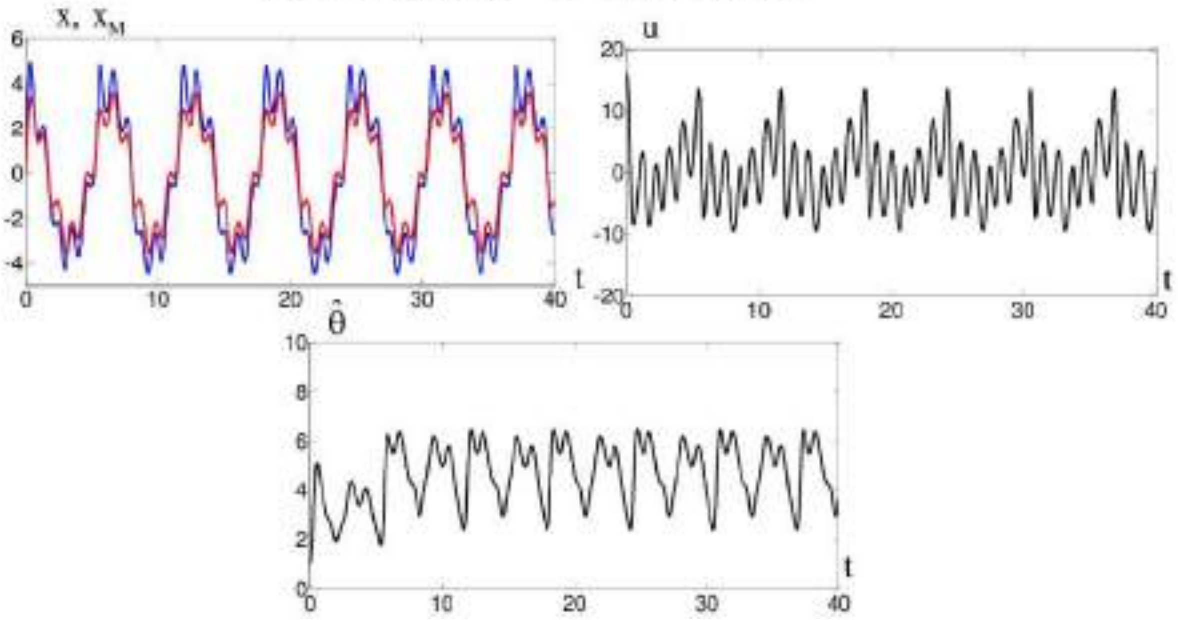
$$u = -6x - \hat{\theta}x + 6g,$$

$\boxed{\theta = 5}$

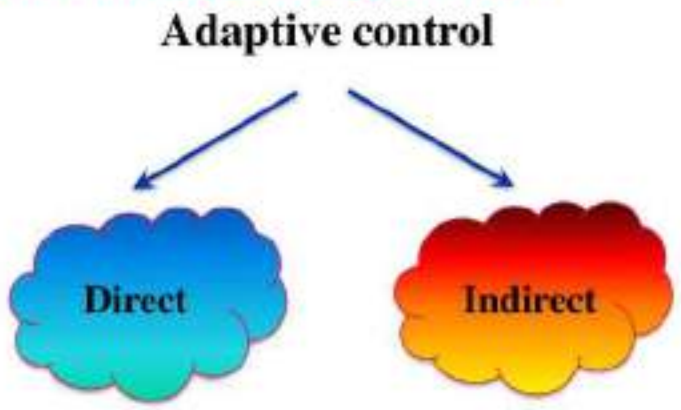
$$\dot{\hat{\theta}} = -2x\varepsilon - \hat{\theta}, \qquad \varepsilon = x_M - x,$$

$\boxed{\delta = 0{,}5\sin(4t) + 0{,}75\cos(2t)}$

$$\dot{x}_M = -6x_M + 6g, \qquad g = \sin 6t + 3\cos t$$



5. Direct and indirect adaptation



Parameters are adjusted in the framework of minimization of the control error

Parameters are adjusted in the framework of minimization of the identification error

Direct adaptive control

Adjustable controller:

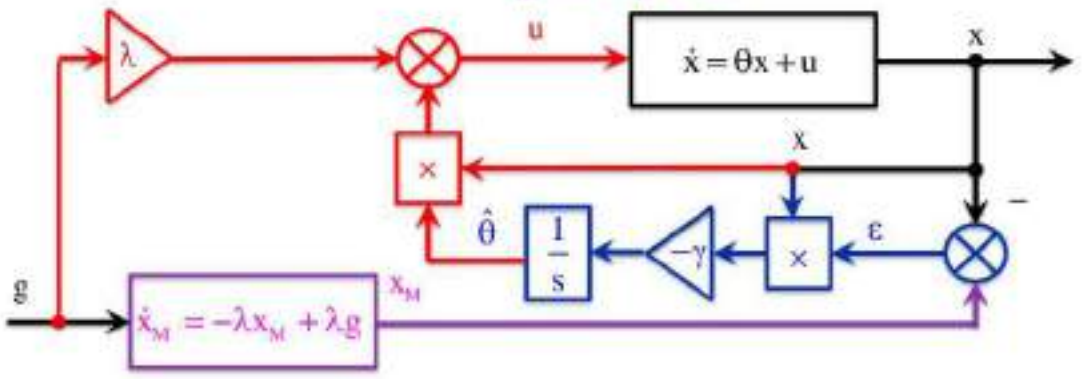
$$u = -\hat{\theta}x - \lambda x + \lambda g \qquad (1)$$

Adaptation algorithm:

$$\dot{\hat{\theta}} = -\gamma x \varepsilon \qquad (2)$$

with $\varepsilon = x_M - x$ and reference model

$$\dot{x}_M = -\lambda x_M + \lambda g. \qquad (3)$$



Indirect adaptive control

Problem statement:

Plant:

$$\dot{x} = \theta x + u, \tag{4}$$

where θ is the unknown parameter.

Objective is to design a control providing the following limiting equality:

$$\lim_{t\to\infty}\left(x_M - x\right) = 0, \tag{5}$$

where x_M is the output of reference model

$$\dot{x}_M = -\lambda x_M + \lambda g, \tag{6}$$

g is the reference signal, λ is the positive parameter responsible for transient time.

Indirect adaptive control

Solution:

1. Let the parameter θ is known.

Let us for the error signal $\varepsilon = x_M - x$ and take its derivative in amount of plant and reference model equations:

$$\dot{\varepsilon} = \dot{x}_M - \dot{x} = \left(-\lambda x_M + \lambda g\right) - \left(\theta x + u\right)$$

Let $\dot{\varepsilon} = -\lambda\varepsilon = -\lambda x_M + \lambda x \;\Rightarrow\; \varepsilon(t) = \exp(-\lambda t)\varepsilon(0)$. Therefore

$$\left(-\lambda x_M + \lambda g\right) - \left(\theta x + u\right) = -\lambda x_M + \lambda x$$

$$\Downarrow$$

$$\boxed{u = -\theta x - \lambda x + \lambda g} \tag{6}$$

Indirect adaptive control

Solution:

2. Let the parameter θ is unknown. Therefore the control

$$u = -\theta x - \lambda x + \lambda g$$

is not implementable. Substitute estimate $\hat{\theta}$ for θ and obtain implementable adjustable control:

$$\boxed{u = -\hat{\theta} x - \lambda x + \lambda g} \tag{7}$$

Indirect adaptive control

Solution:

Statement: the plant $\dot{x} = \theta x + u$ can be represented in another parametric form:

$$\begin{aligned} x &= (k+\theta)\xi_1 + \xi_2, \\ \dot{\xi}_1 &= -k\xi_1 + x, \\ \dot{\xi}_2 &= -k\xi_2 + u, \end{aligned} \tag{8}$$

where k is some positive constant.

Indirect adaptive control

Solution:

Statement: the plant $\dot{x} = \theta x + u$ can be represented in another parametric form:

$$\begin{aligned} x &= (k+\theta)\xi_1 + \xi_2, \\ \dot{\xi}_1 &= -k\xi_1 + x, \\ \dot{\xi}_2 &= -k\xi_2 + u, \end{aligned} \tag{8}$$

where k is some positive constant.

Proof: Take the derivative of $x = (k+\theta)\xi_1 + \xi_2$:

$$\dot{x} = (k+\theta)\dot{\xi}_1 + \dot{\xi}_2$$

$$\dot{x} = (k+\theta)(-k\xi_1 + x) + \left(-k\xi_2 + u\right)$$

$$\dot{x} = -k^2\xi_1 + kx - k\theta\xi_1 + \theta x - k\xi_2 + u$$

$$\dot{x} = -kx + kx + \dot{x} \quad \Rightarrow \quad 0 = 0$$

Ok!!!

Indirect adaptive control

Solution:

Form the error $e = x - \hat{x}$, where $\hat{x}$ is generated by adjustable parametric model

$$\begin{aligned} \hat{x} &= (k+\hat{\theta})\xi_1 + \xi_2, \\ \dot{\xi}_1 &= -k\xi_1 + x, \\ \dot{\xi}_2 &= -k\xi_2 + u, \end{aligned} \tag{9}$$

where k is some positive constant.

Take the derivative of the error in amount of plant $\dot{x} = \theta x + u$:

$$\dot{e} = \dot{x} - \dot{\hat{x}} = \theta x + u - (k+\hat{\theta})\dot{\xi}_1 - \dot{\xi}_2$$

$$\dot{e} = \theta x + \cancel{u} - (k+\hat{\theta})(-k\xi_1 + x) - (-k\xi_2 + \cancel{u})$$

$$\dot{e} = \tilde{\theta}x + k^2\xi_1 - kx + k\hat{\theta}\xi_1 - \hat{\theta}x + k\xi_2$$

Signal Error Model $$\boxed{\dot{e} = -ke + \tilde{\theta}x} \tag{10}$$

Indirect adaptive control

Solution:

3. Let us choose the algorithm generating estimate $\hat{\theta}$:

$$\boxed{\dot{\hat{\theta}} = \Omega(t)} \tag{11}$$

where $\Omega(t)$ is implementable (measurable) function.

Taking into account that $\tilde{\theta} = \theta - \hat{\theta}$ and

$$\dot{\tilde{\theta}} = -\dot{\hat{\theta}}$$

we get

Parametric Error Model

$$\boxed{\dot{\tilde{\theta}} = -\Omega(t)} \tag{12}$$

Indirect adaptive control

Solution:

4. Models

Signal Error Model $\boxed{\dot{e} = -ke + \tilde{\theta}x}$ (13)

Parametric Error Model $\boxed{\dot{\tilde{\theta}} = -\Omega(t)}$ (14)

Choose the Lyapunov function candidate

$$V(e,\tilde{\theta}) = \frac{1}{2}e^2 + \frac{1}{2\gamma}\tilde{\theta}^2, \qquad \gamma > 0 \tag{15}$$

and take its time derivative using (18) and (20):

$$\dot{V}(e,\tilde{\theta}) = e\dot{e} + \frac{1}{\gamma}\tilde{\theta}\dot{\tilde{\theta}} = -\lambda e^2 + \tilde{\theta}xe - \frac{1}{\gamma}\tilde{\theta}\Omega(t)$$

If $\Omega(t) = \gamma xe$, then $\dot{V}(e,\tilde{\theta}) = -\lambda e^2 < 0$

Indirect adaptive control
Summary

Adjustable controller:

$$u=-\hat{\theta}x-\lambda x+\lambda g \tag{7}$$

Adaptation algorithm:

$$\dot{\hat{\theta}}=\gamma x e \tag{15}$$

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

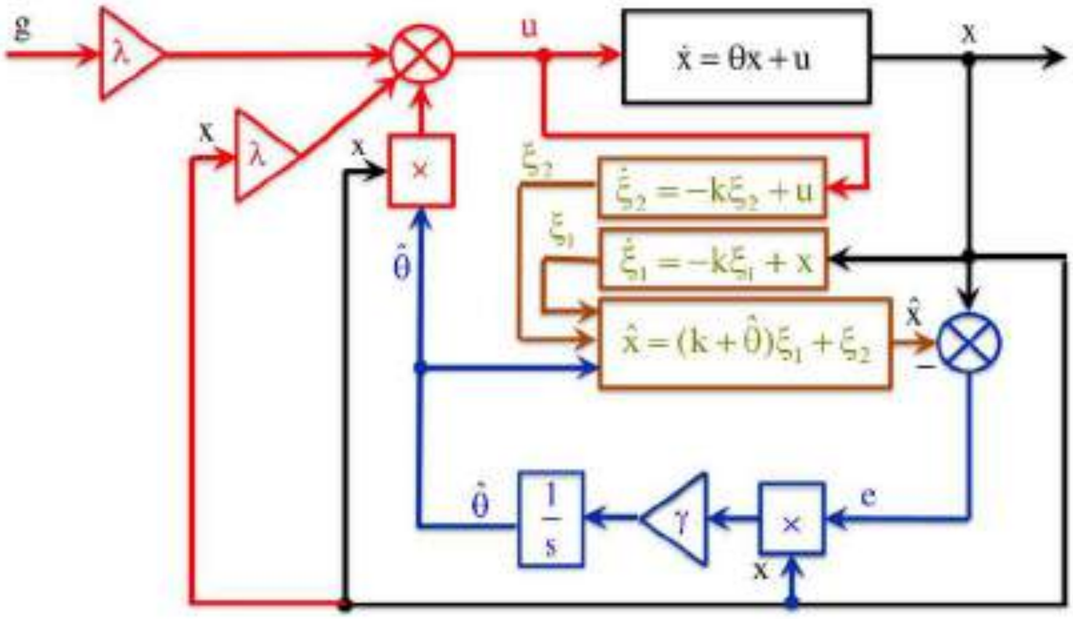
Adjustable model:

$$\hat{x}=(k+\hat{\theta})\xi_1+\xi_2$$

Filters:

$$\dot{\xi}_1=-k\xi_1+x, \qquad \dot{\xi}_2=-k\xi_2+u \tag{9}$$

Indirect adaptive control
Summary



ITMO UNIVERSITY

Indirect adaptive control

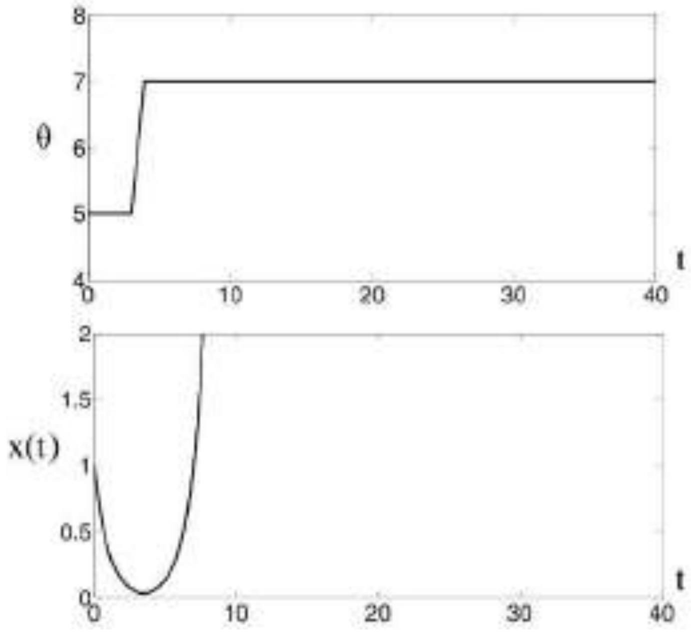
Summary

Properties of the closed-loop system:

1. All signals in the system are bounded;
2. Control error $\varepsilon = x_M - x$ asymptotically tends to zero;
3. Identification error $e = x - \hat{x}$ asymptotically tends to zero;
4. Parametric error $\tilde{\theta} = \theta - \hat{\theta}$ in general case tends to a constant;
$$V(e,\tilde{\theta}) = \frac{1}{2}e^2 + \frac{1}{2\gamma}\tilde{\theta}^2,$$
$$\dot{V}(e,\tilde{\theta}) = -\lambda e^2 < 0$$
5. There is an optimal adaptation gain γ corresponding the fastest parametrical convergence;

Example: Classical stabilizing control for unstable plant $\dot{x} = 5x + u$

$$u = -6x$$



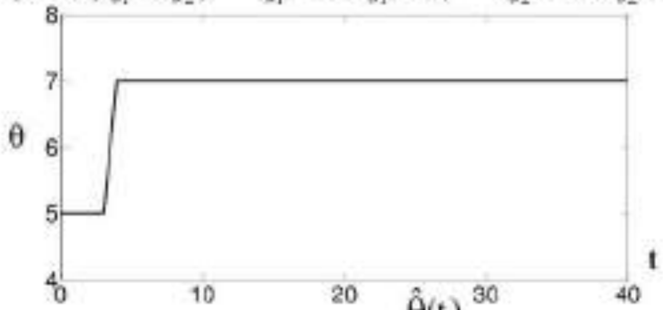
Example: Adaptive stabilizing control for unstable plant $\dot{x} = 5x + u$

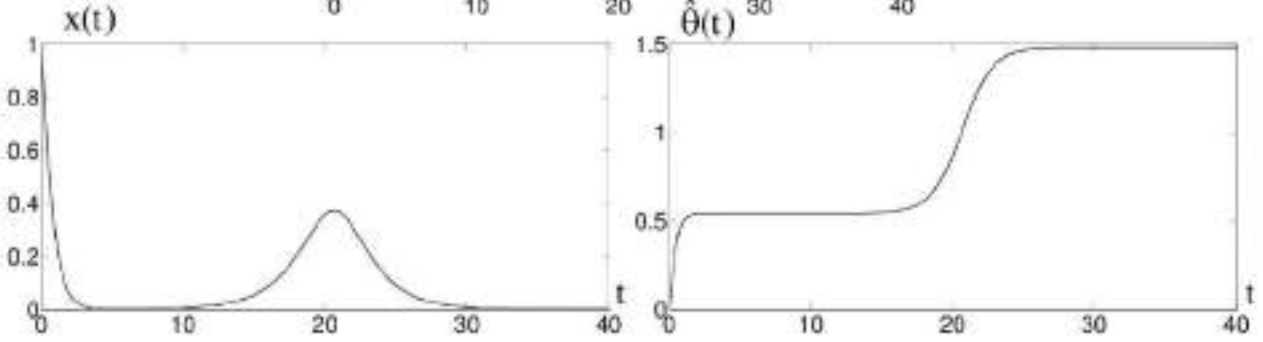
$$u = -6x - \hat{\theta}x,$$

$$\dot{\hat{\theta}} = 0.8xe, \quad e = x - \hat{x},$$

$$\hat{x} = (k + \hat{\theta})\xi_1 + \xi_2, \quad \dot{\xi}_1 = -k\xi_1 + x, \quad \dot{\xi}_2 = -k\xi_2 + u$$

$k = 3$



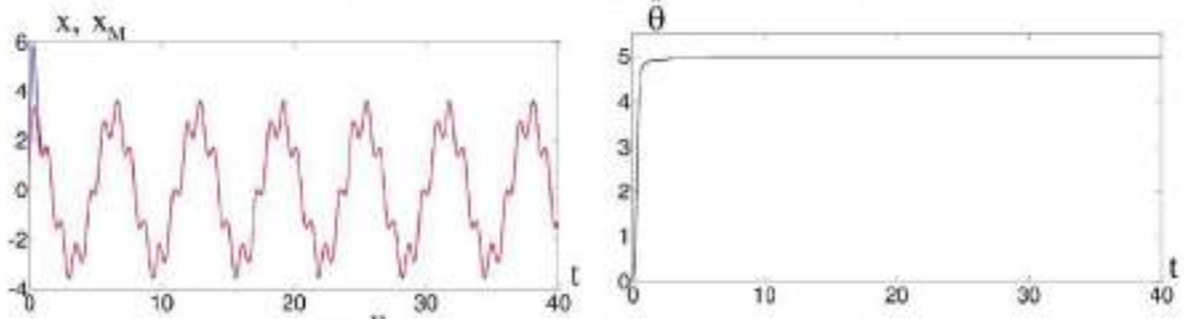


Example: Adaptive tracking control for unstable plant $\dot{x} = 5x + u$

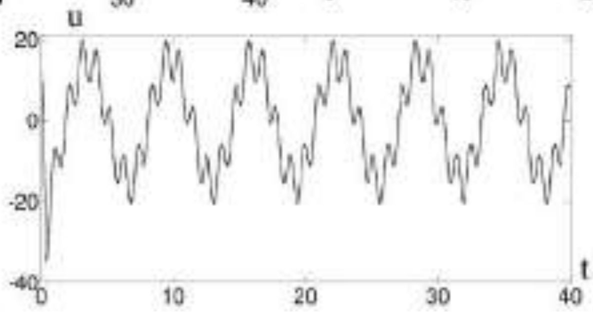
$$u = -6x - \hat{\theta}x + 6g,$$

$$\dot{\hat{\theta}} = 0.8xe, \quad e = x - \hat{x},$$

$$\hat{x} = (k + \hat{\theta})\xi_1 + \xi_2, \quad \dot{\xi}_1 = -k\xi_1 + x, \quad \dot{\xi}_2 = -k\xi_2 + u$$



$k = 3$



Generalized algorithm of adaptive and robust controller design

How to design an adaptive control?

Generalized algorithm of adaptive and robust controller design

1. Problem statement of adaptive control:

Plant:

$$\dot{x} = f(\theta, x, u, \delta), \qquad x(0), \tag{34}$$

where θ is the vector of unknown parameters (or functions),

$f \in R^n$ is continuous nonlinear mapping, $\delta \in R^m : \|\delta\| \le \bar{\delta}$ is the disturbance.

Objective is to design a control u providing the following inequality:

$$|x_M(t) - x(t)| \le \Delta \text{ for any } t \ge T, \tag{35}$$

where x_M is the output of reference model

$$\dot{x}_M = -\lambda x_M + \lambda g, \tag{36}$$

g is the reference signal, λ is the positive parameter.

Generalized algorithm of adaptive and robust controller design

2. Nonadaptive controller design:

Let the plant parameters (functions) θ be known.

Luggage of classical control theory

$\Rightarrow$

Nonadaptive control

$$u = U(\theta, x, e, g), \tag{37}$$

where $e = x_M - x$ is the control error, U is the nonlinear static or dynamical scalar function.

Generalized algorithm of adaptive and robust controller design

3. Adaptive robust controller design

Parameters (functions) θ are unknown.

Generalized algorithm of adaptive and robust controller design

3. Adjustable controller design

Parameters (functions) θ are unknown.

Substitute estimates $\hat{\theta}$ for θ in control (37) and obtain adjustable controller:

$$\boxed{u = U(\hat{\theta}, x, e, g)} \tag{38}$$

Substitute (38) into the plant $\dot{x} = f(\theta, x, u, \delta)$:

$$\dot{x} = f(\theta, x, U(\hat{\theta}, x, e, g), \delta)$$

Form the error $e = x_M - x$ and take its derivative:

Generalized algorithm of adaptive and robust controller design

Form the error $e = x_M - x$ and take its derivative:

$$\dot{e} = \dot{x}_M - \dot{x} = (-\lambda x_M + \lambda g) - f(\theta, x, U(\hat{\theta}, x, \varepsilon, g), \delta)$$

$$\Downarrow$$

Signal Error Model $\boxed{\dot{e} = E(e, \tilde{\theta}, t)}$ (39)

where E is the nonlinear static vector function,

$\tilde{\theta} = \theta - \hat{\theta}$ is the parametric error.

Generalized algorithm of adaptive and robust controller design

4. Adaptation algorithm design

Form the parametric error model

Parametric Error Model $\boxed{\dot{\tilde{\theta}} = \Omega(e,t),}$ (40)

where Ω is the implementable (measurable) function to be determined.

Adaptation algorithm:

$$\boxed{\dot{\hat{\theta}} = -\Omega(e,t),} \qquad \dot{\tilde{\theta}} = -\dot{\hat{\theta}} \tag{41}$$

Generalized algorithm of adaptive and robust controller design

5. Determination of Ω

Signal Error Model $\boxed{\dot{e} = E\left(e,\tilde{\theta},t\right)}$ (39)

Parametric Error Model $\boxed{\dot{\tilde{\theta}} = \Omega(e,t),}$ (40)

Choose a Lyapunov function candidate

$$V = V(e,\tilde{\theta},t).$$

Take its derivative in amount of (39) and (40) $\dot{V}(e,\tilde{\theta})$.

Generalized algorithm of adaptive and robust controller design

$$\begin{tikzcd}
\boxed{\dot{e} = E\left(e,\tilde{\theta},t\right)} \arrow[dr] & & \boxed{\dot{\hat{\theta}} = \Omega\left(e,t\right),} \arrow[dl] \\
& \dot{V}(e,\tilde{\theta}). &
\end{tikzcd}$$

Condition

$$\dot{V}(e,\tilde{\theta}) < 0.$$

gives

$$\Downarrow$$

Adaptation algorithm: $\Omega\left(e,t\right)$

$$\boxed{\dot{\hat{\theta}} = -\Omega\left(e,t\right)} \tag{41}$$

Generalized algorithm of adaptive and robust controller design

Summary

Adjustable control

$$u = U\,(\hat{\theta}, x, \varepsilon, g) \tag{38}$$

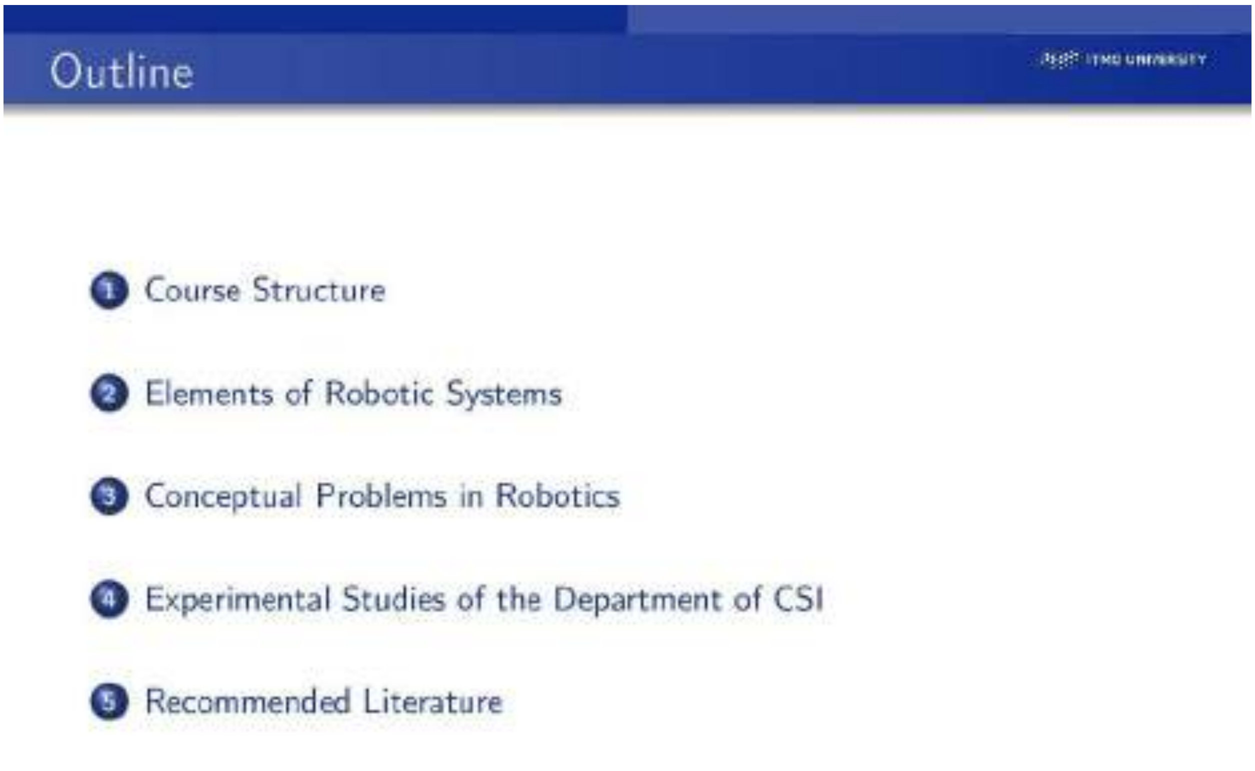
Adaptation algorithm

$$\dot{\hat{\theta}} = -\Omega\left(e,t\right) \tag{41}$$

Generalized algorithm of adaptive and robust controller design

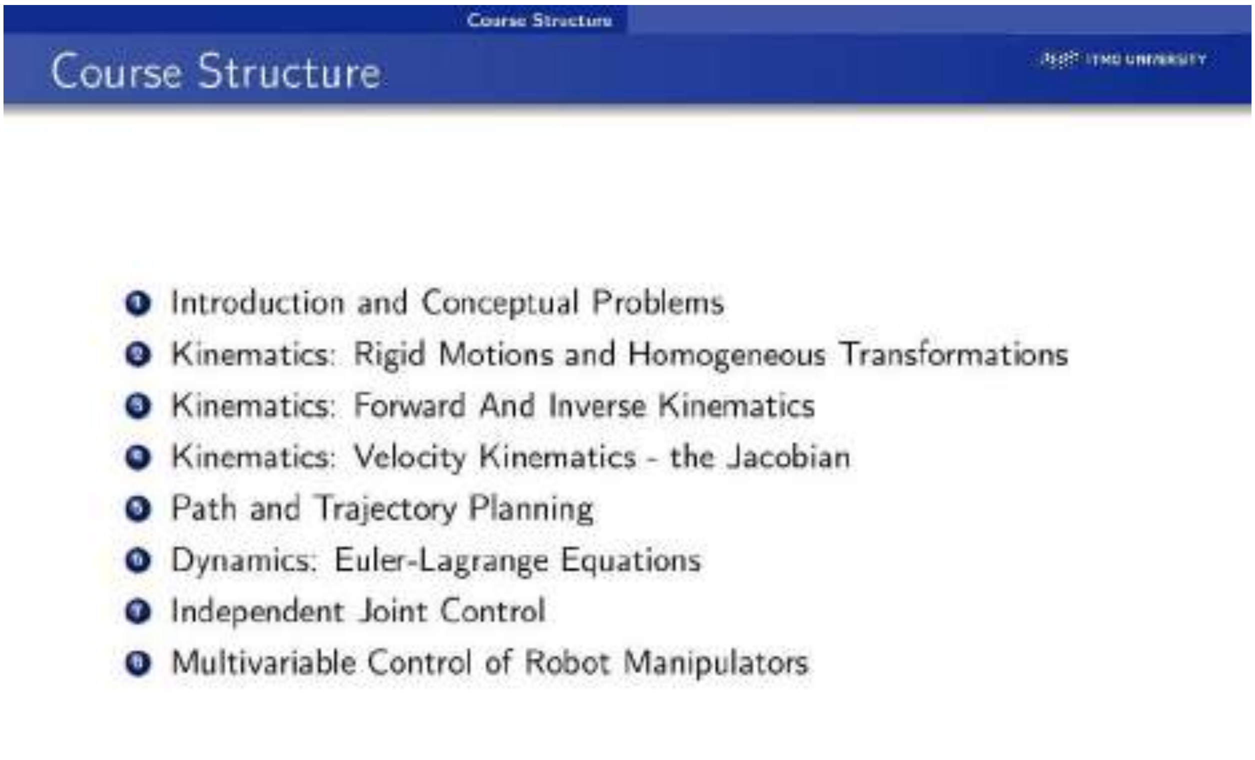
There is no any universal approach of Lyapunov function choice!

Introduction and Conceptual Problems



Outline

1. Course Structure
2. Elements of Robotic Systems
3. Conceptual Problems in Robotics
4. Experimental Studies of the Department of CSI
5. Recommended Literature



Course Structure

Course Structure

1. Introduction and Conceptual Problems
2. Kinematics: Rigid Motions and Homogeneous Transformations
3. Kinematics: Forward And Inverse Kinematics
4. Kinematics: Velocity Kinematics - the Jacobian
5. Path and Trajectory Planning
6. Dynamics: Euler-Lagrange Equations
7. Independent Joint Control
8. Multivariable Control of Robot Manipulators

Elements of Robotic Systems

Types of Robot Joints

ITMO UNIVERSITY

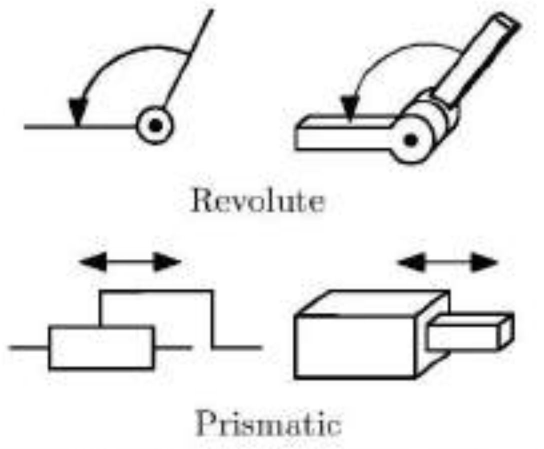


Fig.: Symbolic representation of robot joints types: revolute (relative rotation between adjacent links) and prismatic (relative displacement between adjacent links)

Elements of Robotic Systems

Revolute Joints vs. Prismatic Joints

ITMO UNIVERSITY

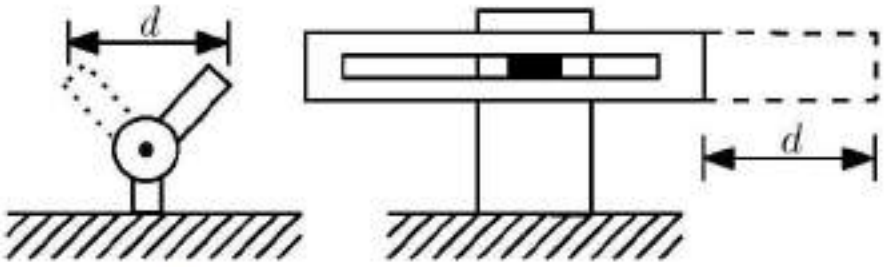
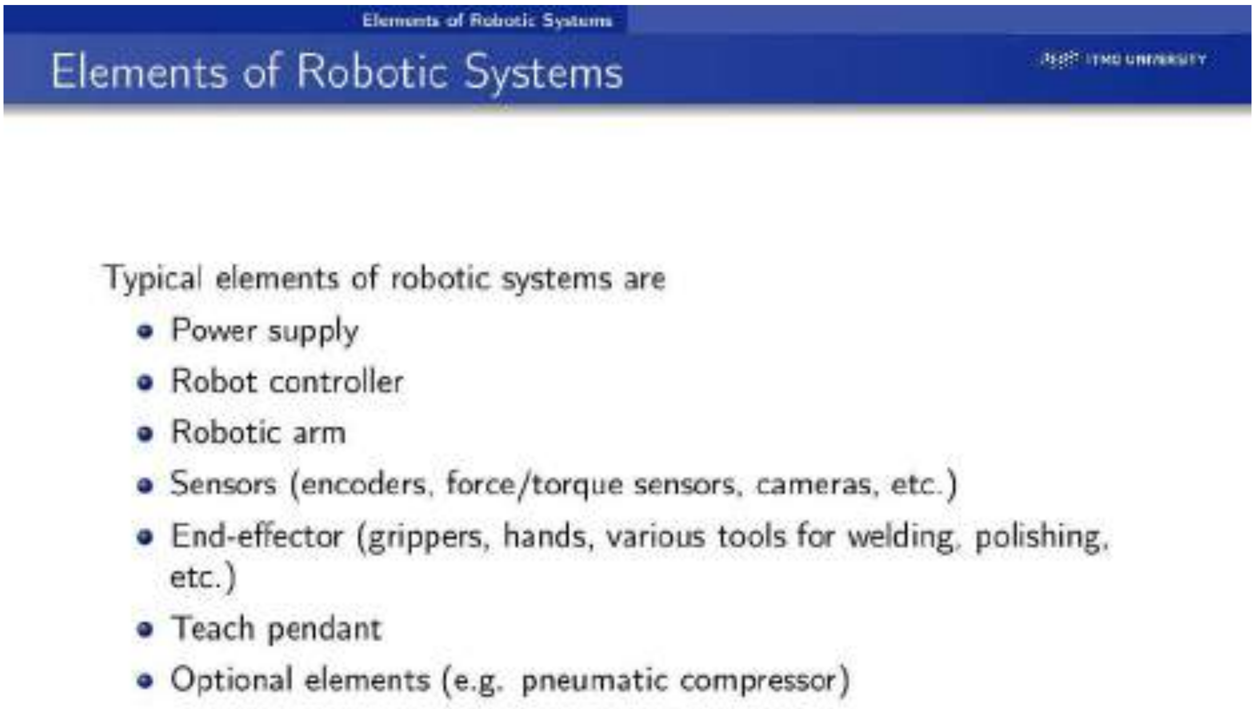


Fig.: Revolute Joints vs. Prismatic Joints



Elements of Robotic Systems

Elements of Robotic Systems

Typical elements of robotic systems are

- Power supply
- Robot controller
- Robotic arm
- Sensors (encoders, force/torque sensors, cameras, etc.)
- End-effector (grippers, hands, various tools for welding, polishing, etc.)
- Teach pendant
- Optional elements (e.g. pneumatic compressor)

Elements of Robotic Systems

Examples of Various End-Effectors

Fig.: 3-fingered hand

Fig.: 2-fingered gripper of the KUKA youBot robot

Fig.: Pneumatic gripper by Festo

Elements of Robotic Systems

Articulated 3-DOF Manipulator

ITMO UNIVERSITY

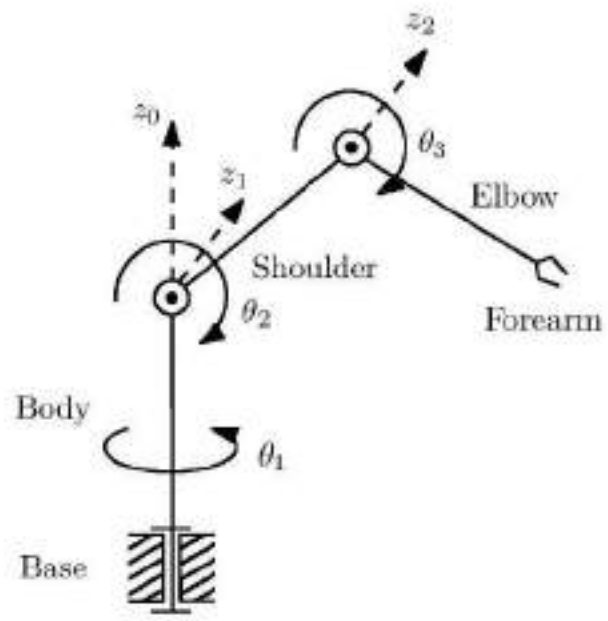


Fig.: Kinematic chain of articulated 3-DOF manipulator

Elements of Robotic Systems

Articulated 3-DOF Manipulator (2)

ITMO UNIVERSITY

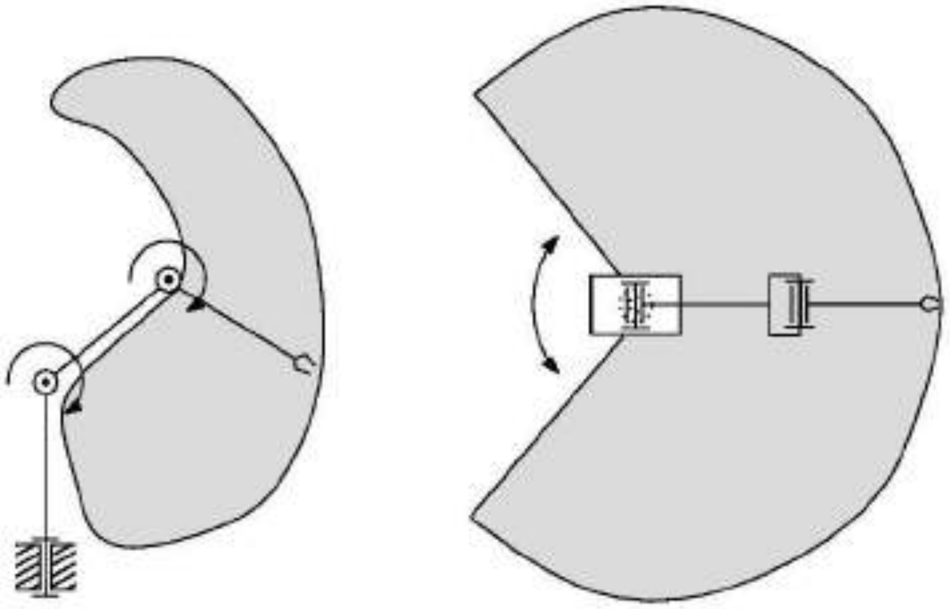
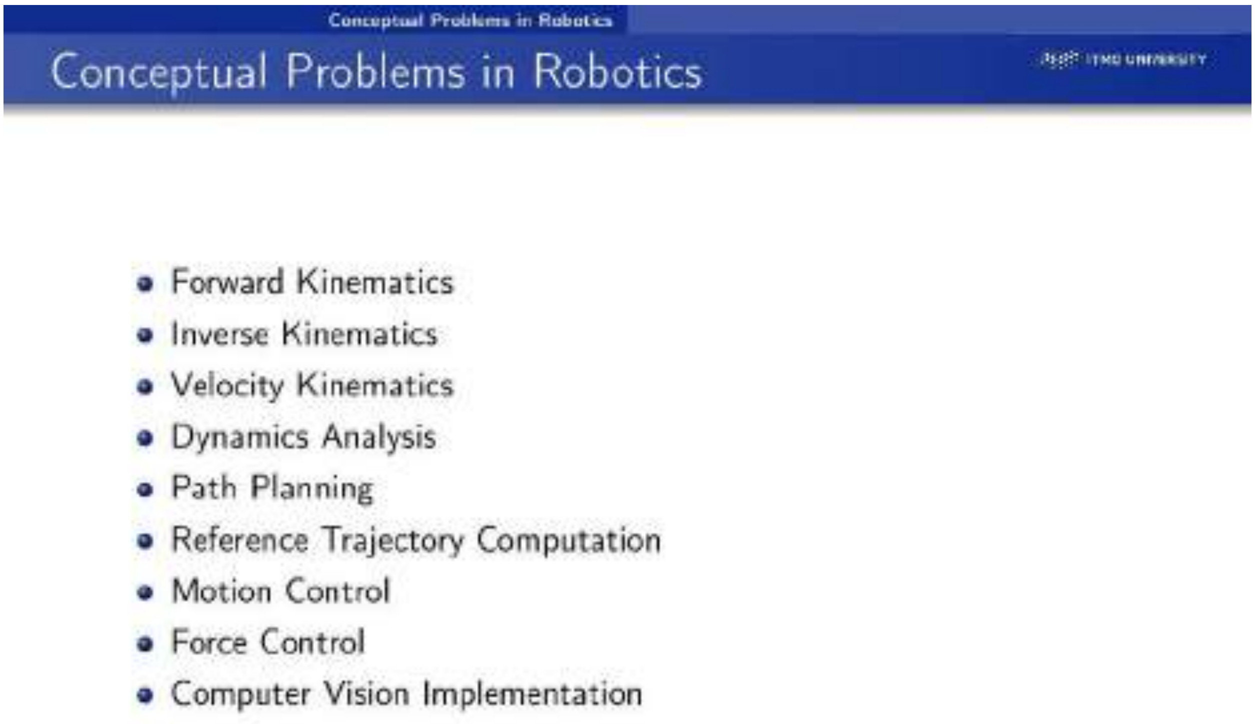
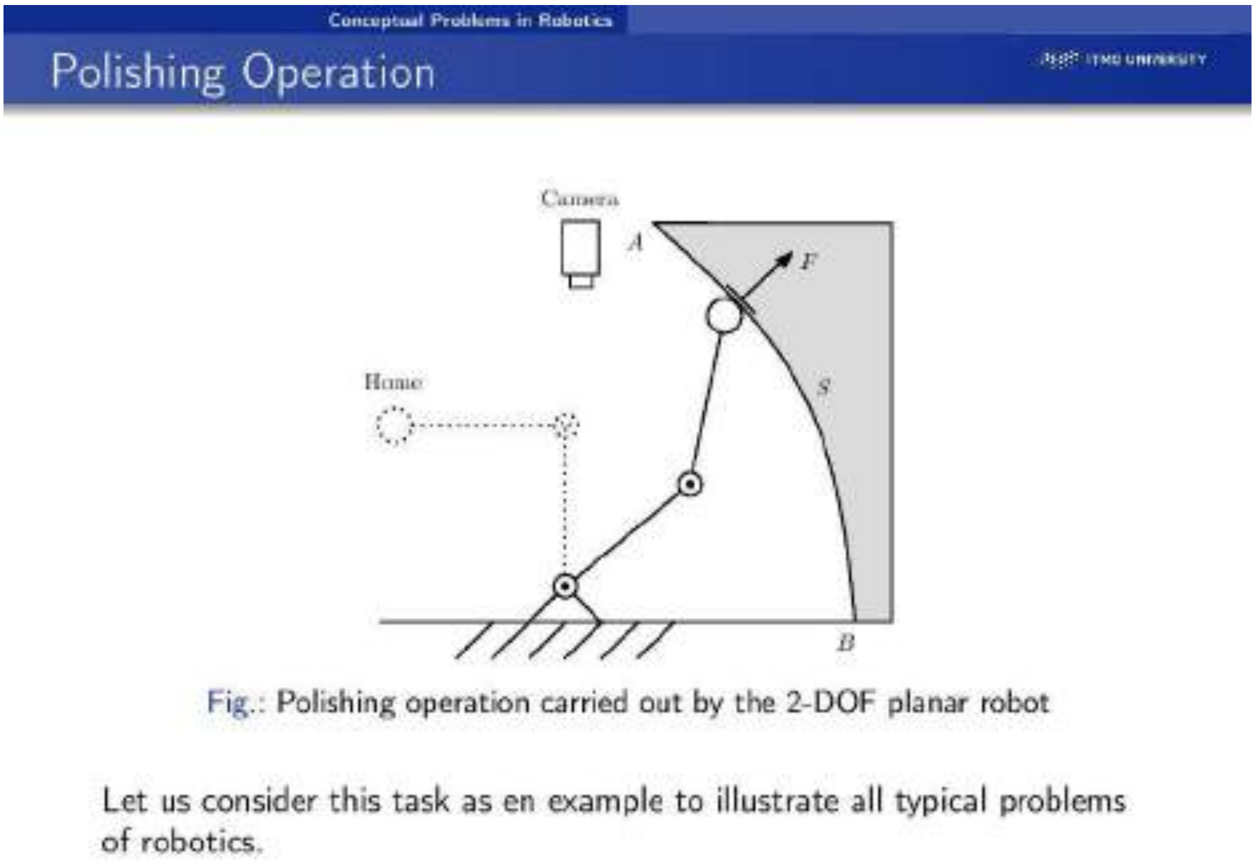


Fig.: Workspace of the articulated 3-DOF manipulator



Conceptual Problems in Robotics

- Forward Kinematics
- Inverse Kinematics
- Velocity Kinematics
- Dynamics Analysis
- Path Planning
- Reference Trajectory Computation
- Motion Control
- Force Control
- Computer Vision Implementation



Polishing Operation

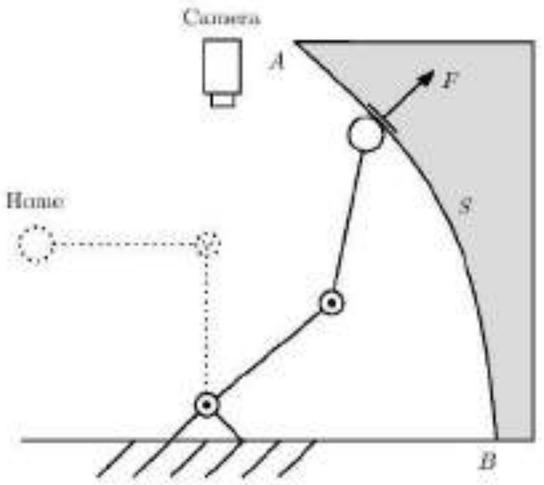
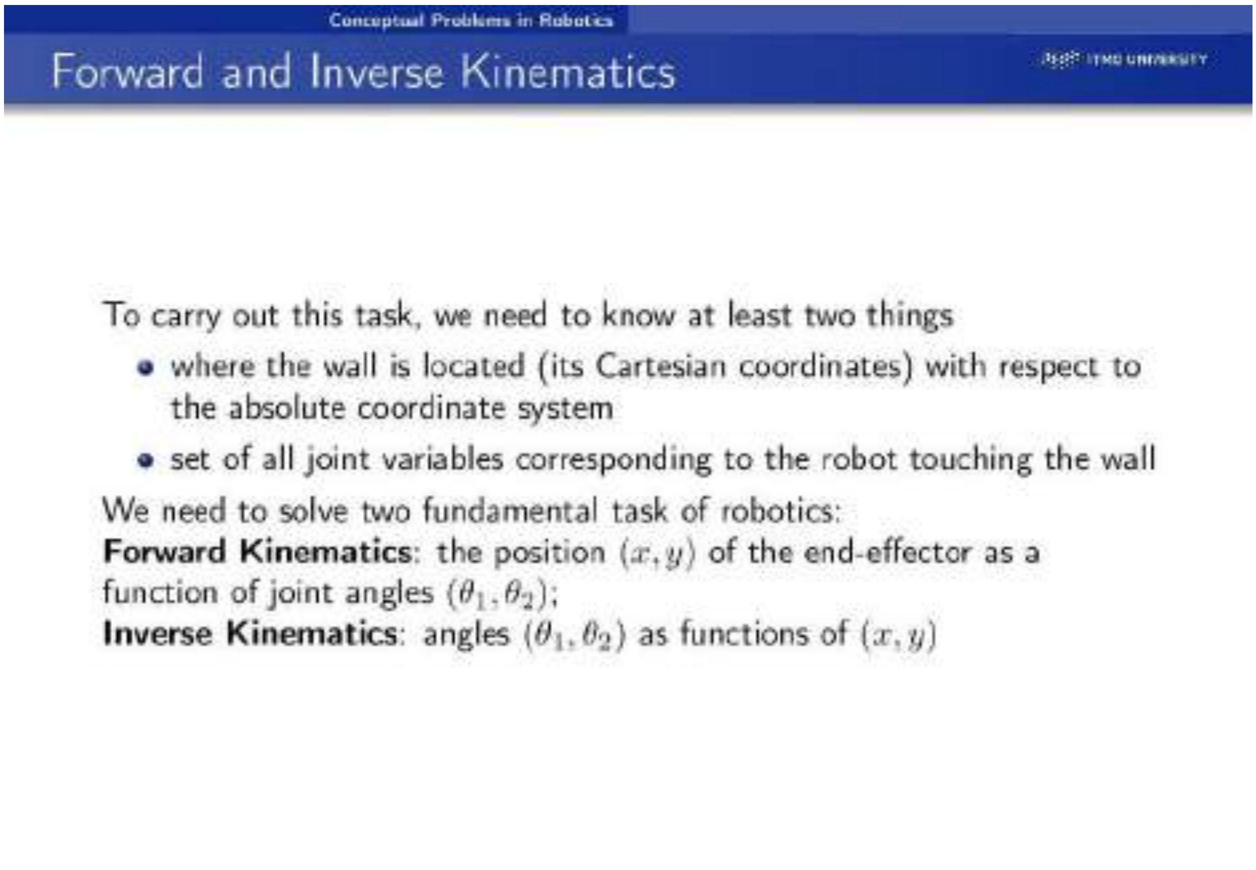


Fig.: Polishing operation carried out by the 2-DOF planar robot

Let us consider this task as en example to illustrate all typical problems of robotics.



Conceptual Problems in Robotics

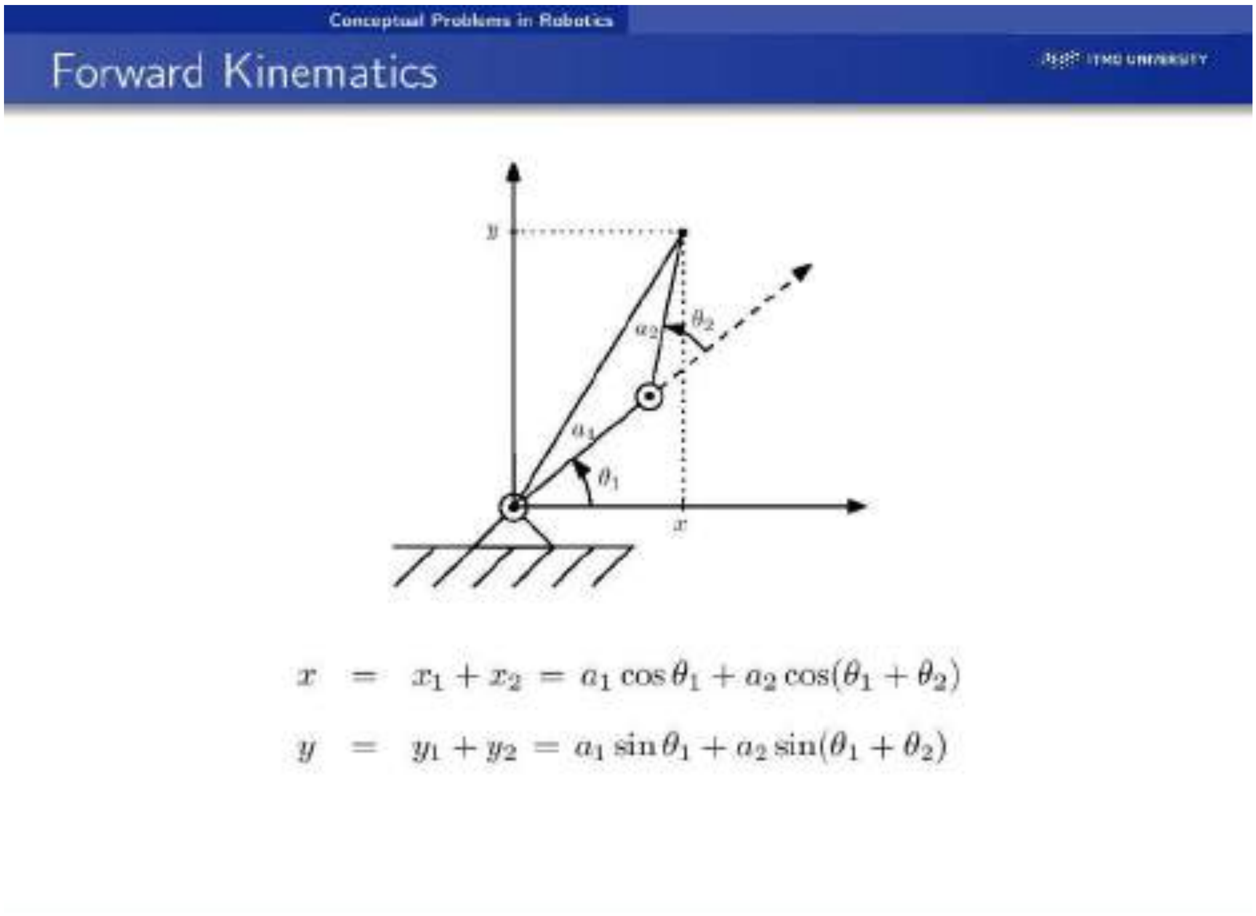
Forward and Inverse Kinematics

ITMO UNIVERSITY

To carry out this task, we need to know at least two things

- where the wall is located (its Cartesian coordinates) with respect to the absolute coordinate system
- set of all joint variables corresponding to the robot touching the wall

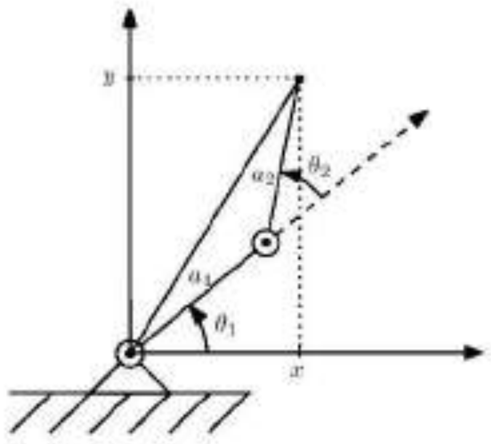
We need to solve two fundamental task of robotics:
Forward Kinematics: the position (x, y) of the end-effector as a function of joint angles (θ_1, θ_2);
Inverse Kinematics: angles (θ_1, θ_2) as functions of (x, y)



Conceptual Problems in Robotics

Forward Kinematics

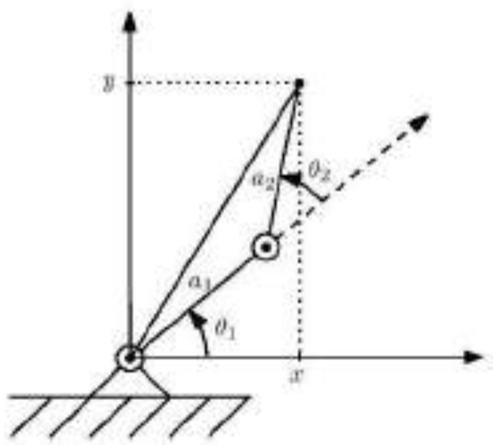
ITMO UNIVERSITY



$$
\begin{aligned}
x &= x_1 + x_2 = a_1 \cos\theta_1 + a_2 \cos(\theta_1 + \theta_2) \\
y &= y_1 + y_2 = a_1 \sin\theta_1 + a_2 \sin(\theta_1 + \theta_2)
\end{aligned}
$$

Conceptual Problems in Robotics

Forward Kinematics (Tool Frame Orientation)



$$\underbrace{\begin{bmatrix} x_2 \cdot x_0 \\ x_2 \cdot y_0 \end{bmatrix}}_{x_2^0} = \begin{bmatrix} \cos(\theta_1 + \theta_2) \\ \sin(\theta_1 + \theta_2) \end{bmatrix}, \quad \underbrace{\begin{bmatrix} y_2 \cdot x_0 \\ y_2 \cdot y_0 \end{bmatrix}}_{y_2^0} = \begin{bmatrix} -\sin(\theta_1 + \theta_2) \\ \cos(\theta_1 + \theta_2) \end{bmatrix}$$

Conceptual Problems in Robotics

Inverse Kinematics

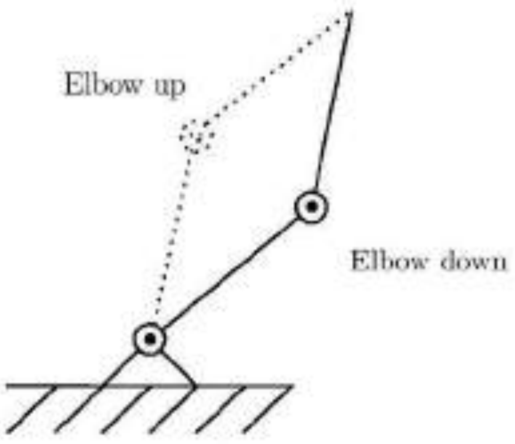
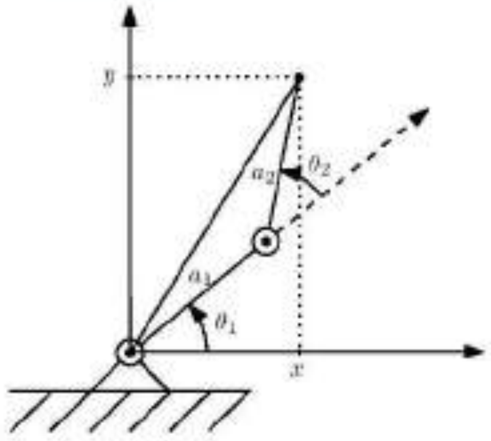


Fig.: Uncertainty of inverse kinematic solution

Conceptual Problems in Robotics

Inverse Kinematics

ITMO UNIVERSITY

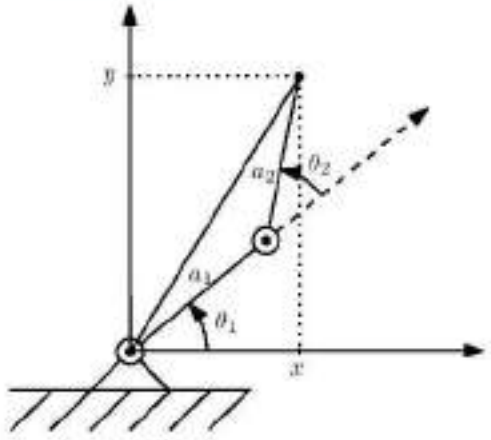


$$\vec{c} = \vec{a}_1 + \vec{a}_2 \quad \Rightarrow \|\vec{c}\|^2 = \|\vec{a}_1 + \vec{a}_2\|^2$$
$$x^2 + y^2 = (\vec{a}_1 + \vec{a}_2) \cdot (\vec{a}_1 + \vec{a}_2)$$
$$x^2 + y^2 = a_1^2 + a_2^2 + 2\vec{a}_1\vec{a}_2$$
$$x^2 + y^2 = a_1^2 + a_2^2 + 2a_1a_2\cos\theta_2$$

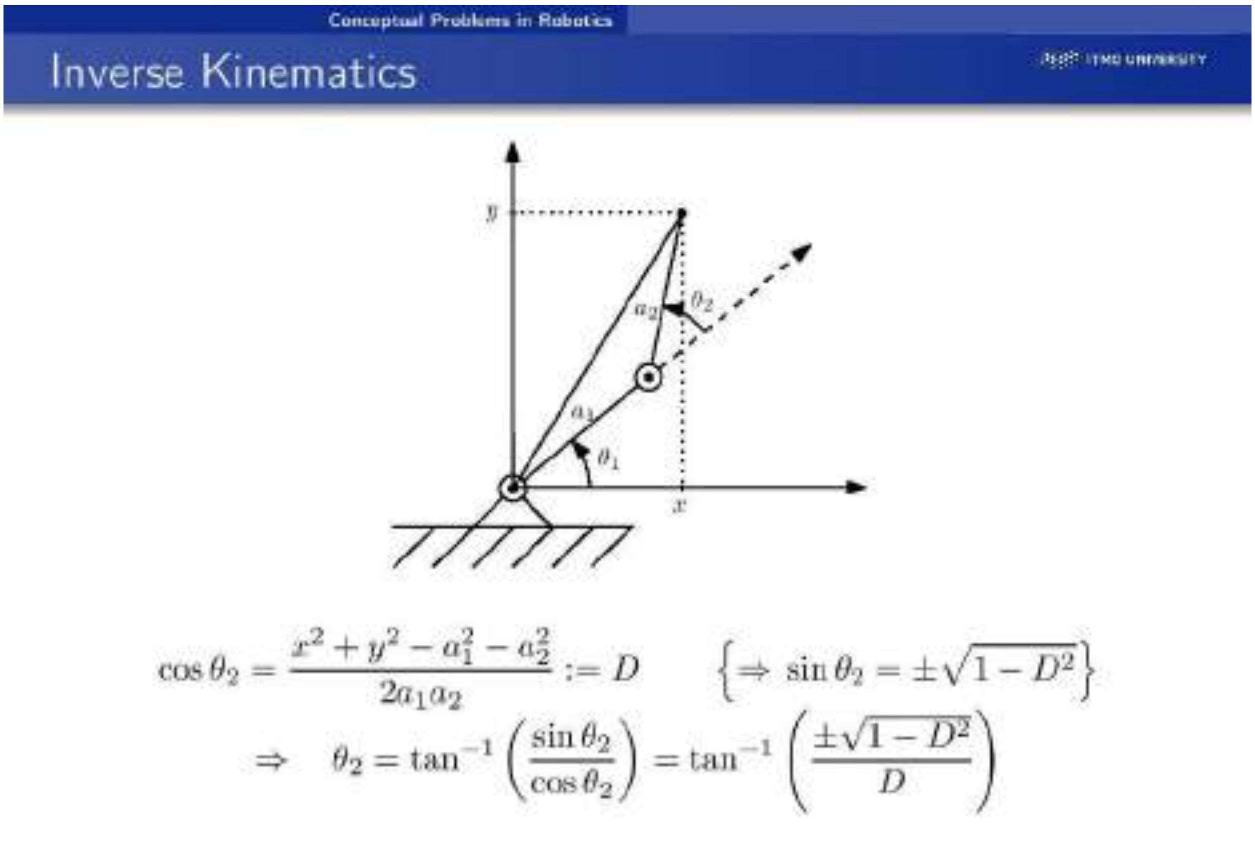
Conceptual Problems in Robotics

Inverse Kinematics

ITMO UNIVERSITY

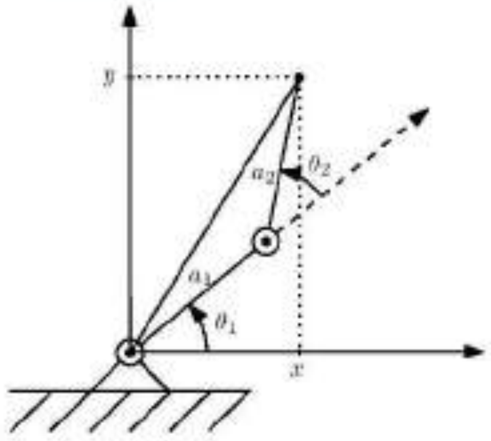


$$\cos\theta_2 = \frac{x^2 + y^2 - a_1^2 - a_2^2}{2a_1a_2} := D \qquad \left\{(\sin\theta_2)^2 + (\cos\theta_2)^2 = 1\right\}$$



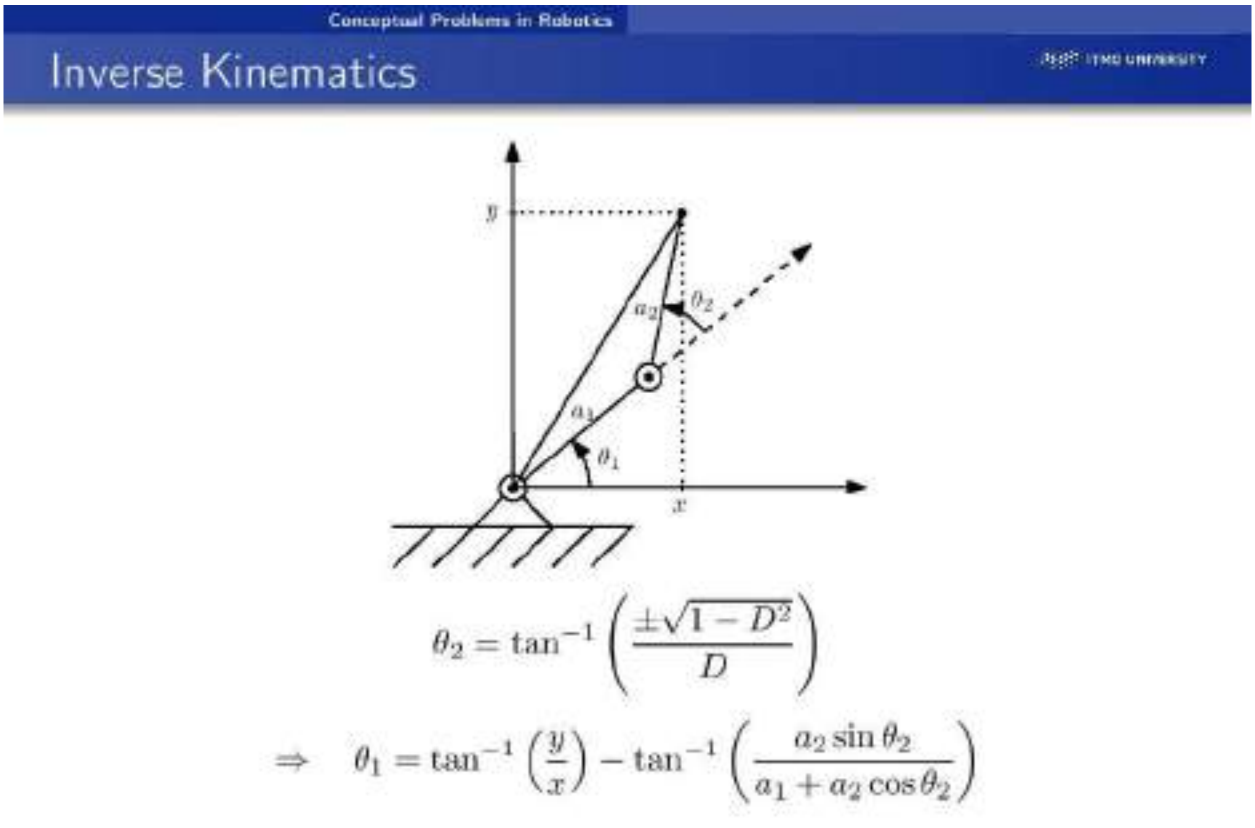
Conceptual Problems in Robotics

Inverse Kinematics



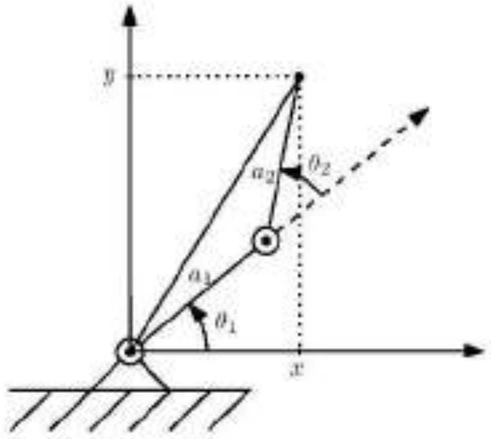
$$\cos\theta_2 = \frac{x^2+y^2-a_1^2-a_2^2}{2a_1a_2} := D \qquad \left\{\Rightarrow \sin\theta_2 = \pm\sqrt{1-D^2}\right\}$$

$$\Rightarrow \quad \theta_2 = \tan^{-1}\left(\frac{\sin\theta_2}{\cos\theta_2}\right) = \tan^{-1}\left(\frac{\pm\sqrt{1-D^2}}{D}\right)$$



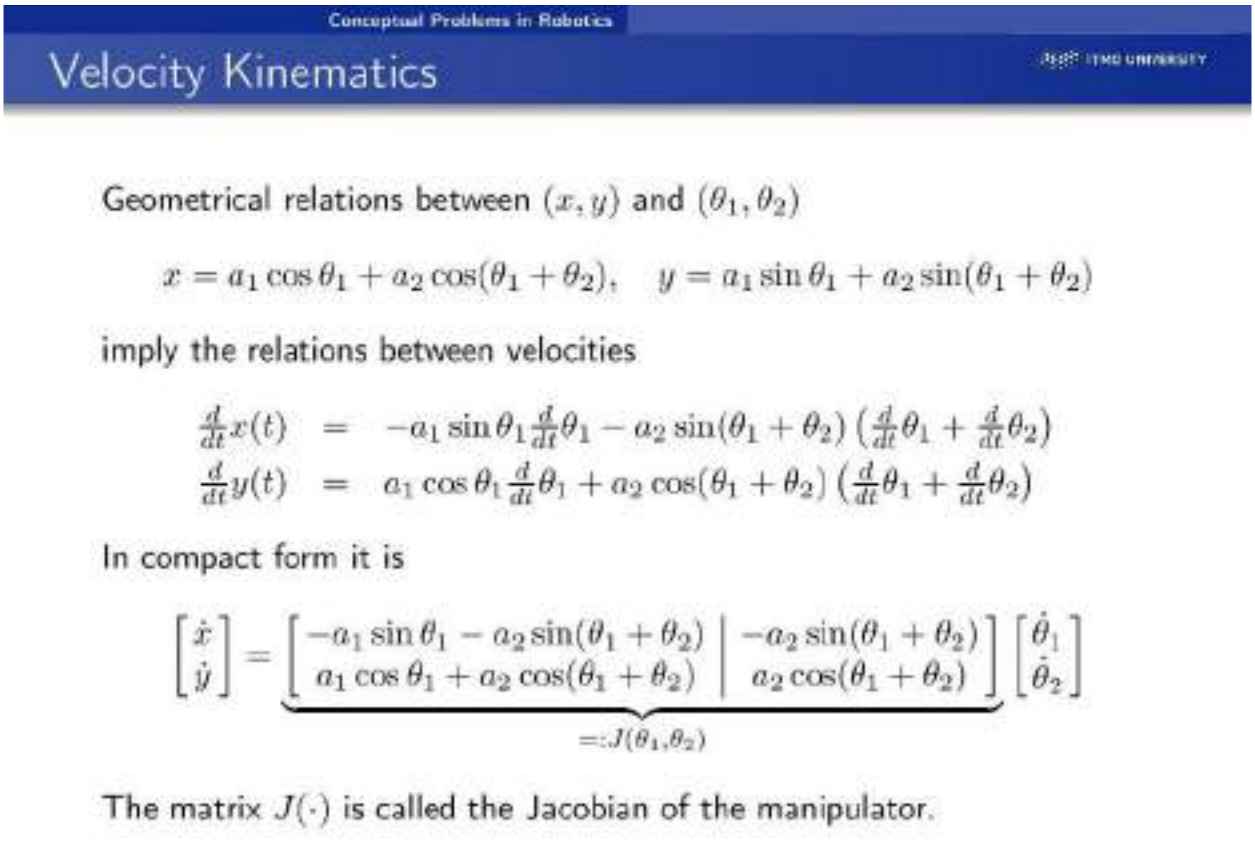
Conceptual Problems in Robotics

Inverse Kinematics



$$\theta_2 = \tan^{-1}\left(\frac{\pm\sqrt{1-D^2}}{D}\right)$$

$$\Rightarrow \quad \theta_1 = \tan^{-1}\left(\frac{y}{x}\right) - \tan^{-1}\left(\frac{a_2\sin\theta_2}{a_1+a_2\cos\theta_2}\right)$$



Conceptual Problems in Robotics

Velocity Kinematics

ITMO UNIVERSITY

Geometrical relations between (x, y) and (θ_1, θ_2)

$$x = a_1 \cos\theta_1 + a_2\cos(\theta_1+\theta_2), \quad y = a_1\sin\theta_1 + a_2\sin(\theta_1+\theta_2)$$

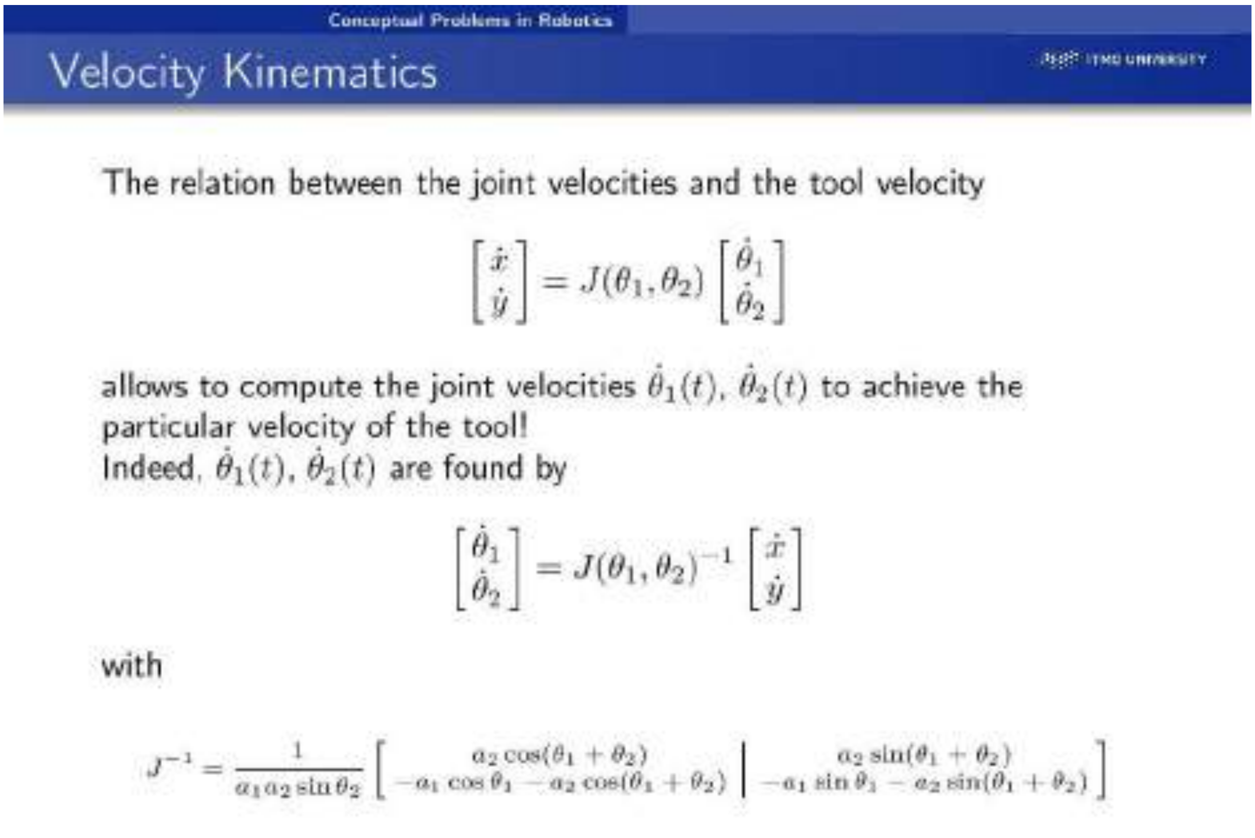
imply the relations between velocities

$$\begin{aligned}\tfrac{d}{dt}x(t) &= -a_1\sin\theta_1\tfrac{d}{dt}\theta_1 - a_2\sin(\theta_1+\theta_2)\left(\tfrac{d}{dt}\theta_1 + \tfrac{d}{dt}\theta_2\right)\\ \tfrac{d}{dt}y(t) &= a_1\cos\theta_1\tfrac{d}{dt}\theta_1 + a_2\cos(\theta_1+\theta_2)\left(\tfrac{d}{dt}\theta_1 + \tfrac{d}{dt}\theta_2\right)\end{aligned}$$

In compact form it is

$$\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix} = \underbrace{\left[\begin{array}{c|c}-a_1\sin\theta_1 - a_2\sin(\theta_1+\theta_2) & -a_2\sin(\theta_1+\theta_2)\\ a_1\cos\theta_1 + a_2\cos(\theta_1+\theta_2) & a_2\cos(\theta_1+\theta_2)\end{array}\right]}_{=:J(\theta_1,\theta_2)}\begin{bmatrix}\dot{\theta}_1\\ \dot{\theta}_2\end{bmatrix}$$

The matrix $J(\cdot)$ is called the Jacobian of the manipulator.



Conceptual Problems in Robotics

Velocity Kinematics

ITMO UNIVERSITY

The relation between the joint velocities and the tool velocity

$$\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix} = J(\theta_1,\theta_2)\begin{bmatrix}\dot{\theta}_1\\ \dot{\theta}_2\end{bmatrix}$$

allows to compute the joint velocities $\dot{\theta}_1(t)$, $\dot{\theta}_2(t)$ to achieve the particular velocity of the tool!
Indeed, $\dot{\theta}_1(t)$, $\dot{\theta}_2(t)$ are found by

$$\begin{bmatrix}\dot{\theta}_1\\ \dot{\theta}_2\end{bmatrix} = J(\theta_1,\theta_2)^{-1}\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix}$$

with

$$J^{-1} = \frac{1}{a_1a_2\sin\theta_2}\left[\begin{array}{c|c}a_2\cos(\theta_1+\theta_2) & a_2\sin(\theta_1+\theta_2)\\ -a_1\cos\theta_1 - a_2\cos(\theta_1+\theta_2) & -a_1\sin\theta_1 - a_2\sin(\theta_1+\theta_2)\end{array}\right]$$

Velocity Kinematics

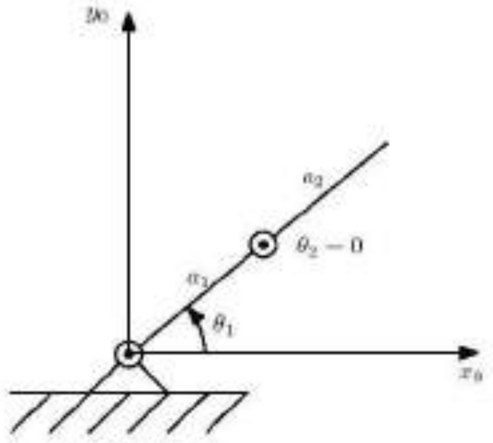
It is clear that the inverse of the Jacobian

$$J^{-1} = \frac{1}{a_1 a_2 \sin\theta_2}\left[\begin{array}{c|c} a_2\cos(\theta_1+\theta_2) & a_2\sin(\theta_1+\theta_2) \\ -a_1\cos\theta_1 - a_2\cos(\theta_1+\theta_2) & -a_1\sin\theta_1 - a_2\sin(\theta_1+\theta_2) \end{array}\right]$$

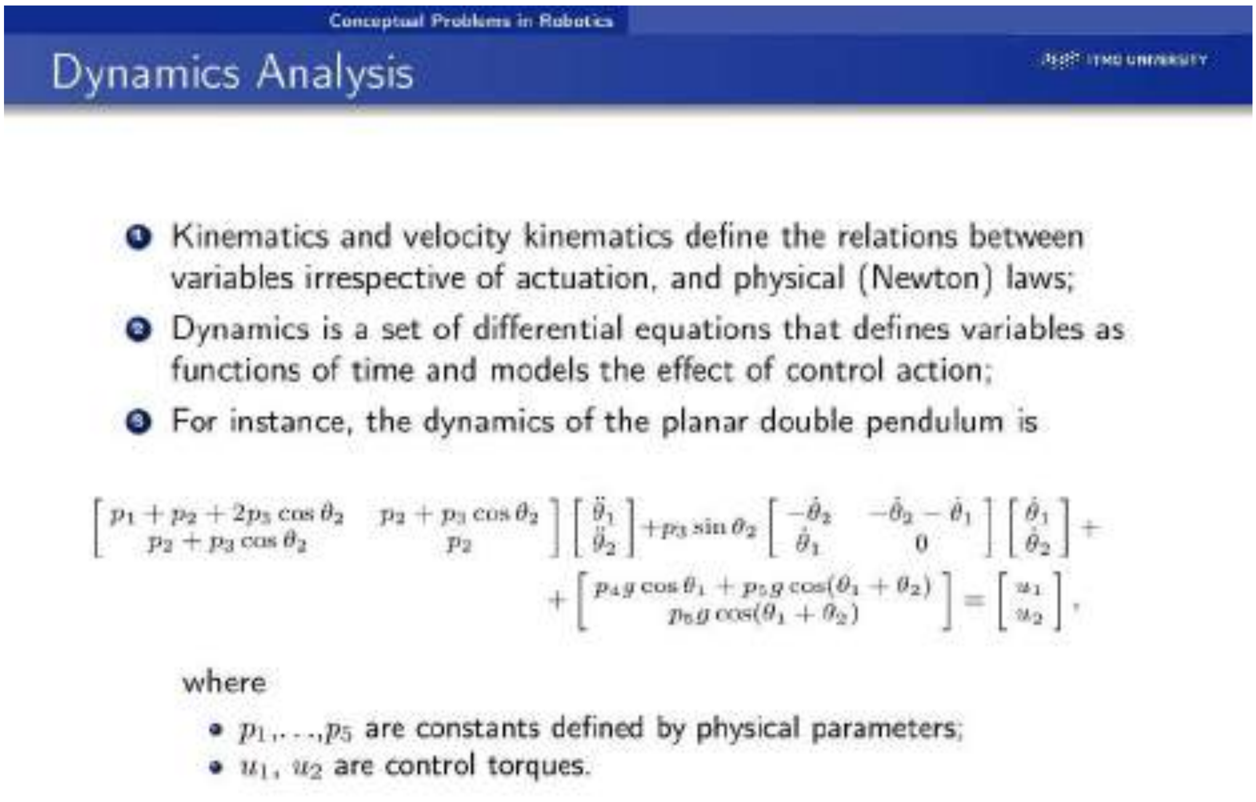
is not defined when

$$\sin\theta_2 = 0 \quad \{\Rightarrow\ \theta_2 = \pi\cdot k,\ k = \ldots -1, 0, 1, \ldots\}$$

Singular Configurations



If $\theta_2 = 0$, then the Jacobian $J(\cdot)$ looses rank and cannot be inverted.



Conceptual Problems in Robotics

Dynamics Analysis

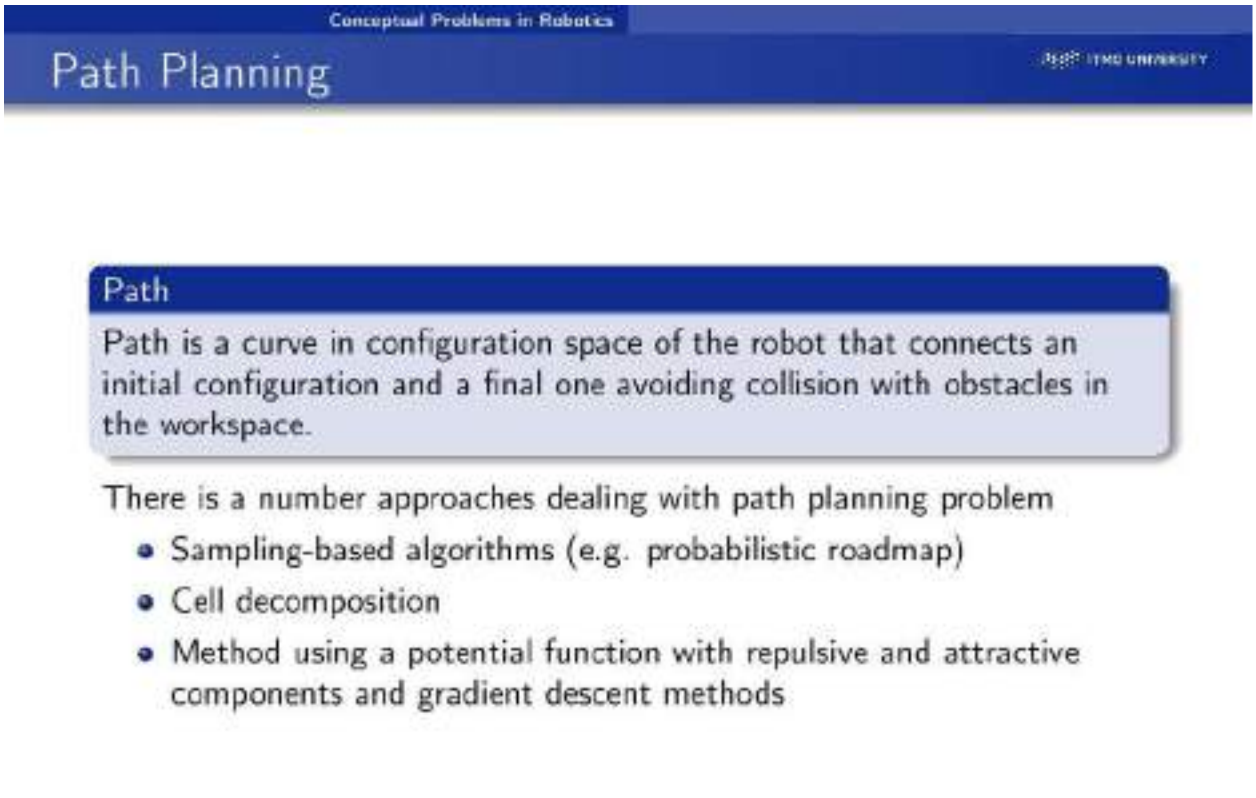
ITMO UNIVERSITY

1. Kinematics and velocity kinematics define the relations between variables irrespective of actuation, and physical (Newton) laws;
2. Dynamics is a set of differential equations that defines variables as functions of time and models the effect of control action;
3. For instance, the dynamics of the planar double pendulum is

$$\begin{bmatrix} p_1+p_2+2p_3\cos\theta_2 & p_2+p_3\cos\theta_2 \\ p_2+p_3\cos\theta_2 & p_2 \end{bmatrix}\begin{bmatrix}\ddot\theta_1\\ \ddot\theta_2\end{bmatrix} + p_3\sin\theta_2\begin{bmatrix}-\dot\theta_2 & -\dot\theta_2-\dot\theta_1\\ \dot\theta_1 & 0\end{bmatrix}\begin{bmatrix}\dot\theta_1\\ \dot\theta_2\end{bmatrix} + \\ + \begin{bmatrix} p_4 g\cos\theta_1 + p_5 g\cos(\theta_1+\theta_2) \\ p_5 g\cos(\theta_1+\theta_2)\end{bmatrix} = \begin{bmatrix}u_1\\ u_2\end{bmatrix},$$

where

- $p_1,\dots,p_5$ are constants defined by physical parameters;
- u_1, u_2 are control torques.



Conceptual Problems in Robotics

Path Planning

ITMO UNIVERSITY

Path

Path is a curve in configuration space of the robot that connects an initial configuration and a final one avoiding collision with obstacles in the workspace.

There is a number approaches dealing with path planning problem

- Sampling-based algorithms (e.g. probabilistic roadmap)
- Cell decomposition
- Method using a potential function with repulsive and attractive components and gradient descent methods

Reference Trajectory Computation

Trajectory

Trajectory is a parametrization of time for generalized positions, velocities and accelerations of the robot.

One of the methods of trajectory generation is interpolation by spline functions.

Motion Control

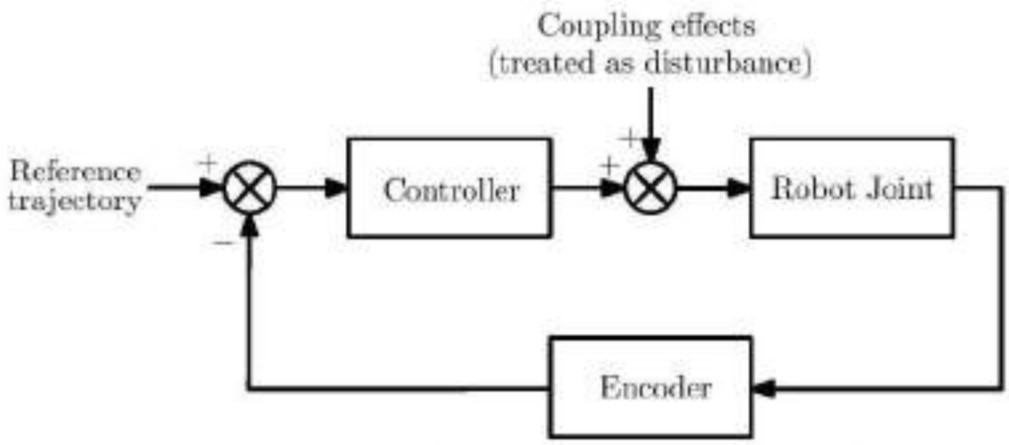
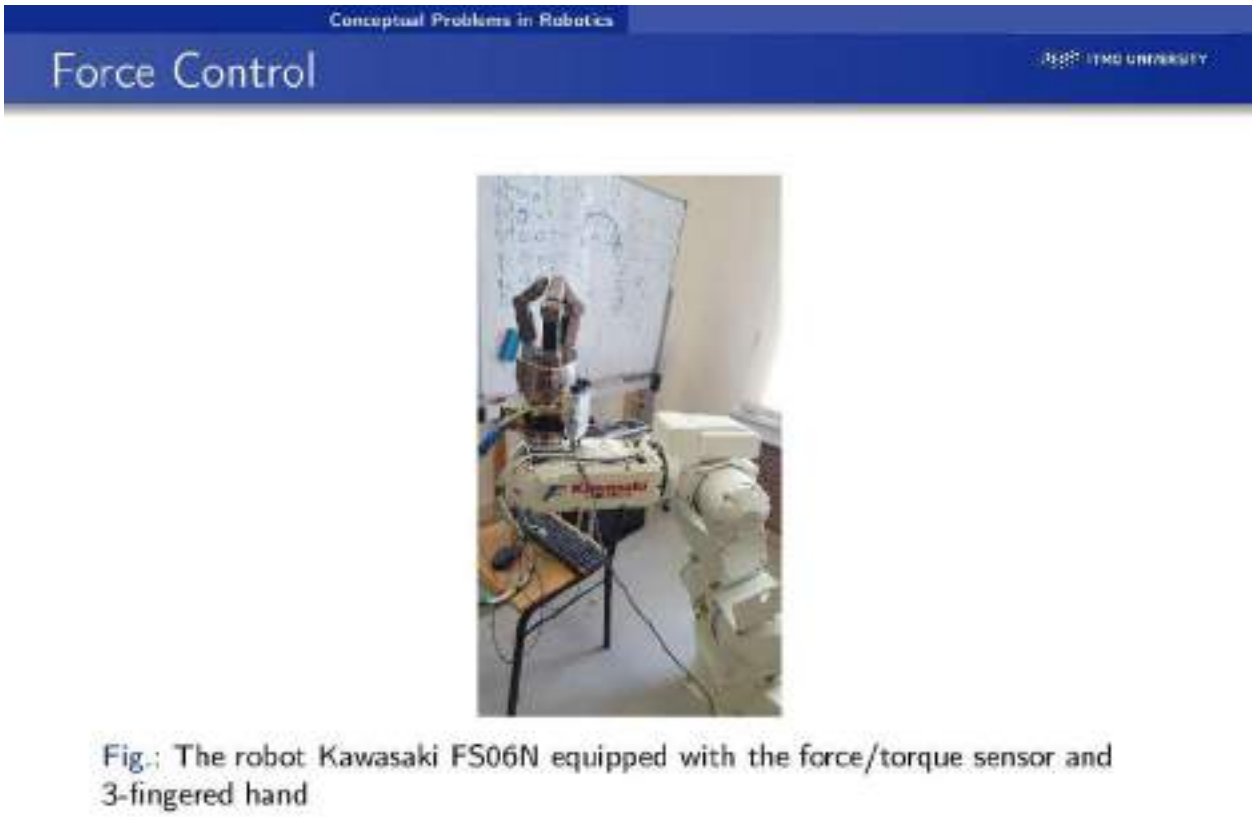


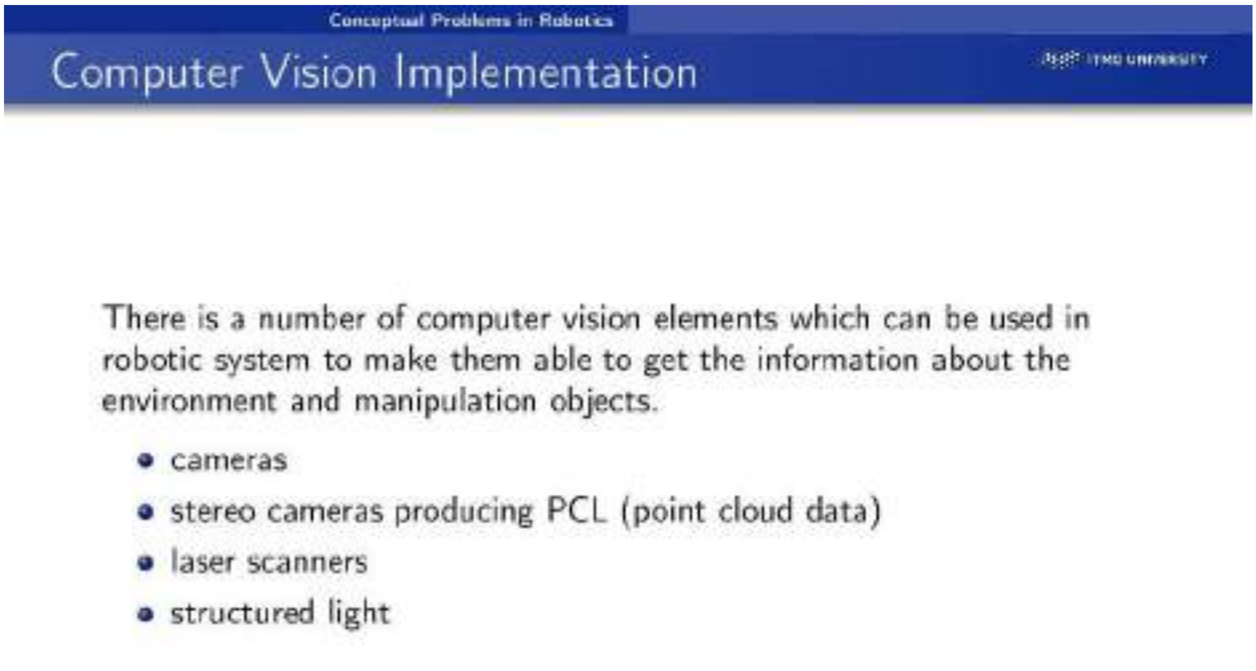
Fig.: Basic scheme of the independent joint control



Conceptual Problems in Robotics

Force Control

Fig.: The robot Kawasaki FS06N equipped with the force/torque sensor and 3-fingered hand



Conceptual Problems in Robotics

Computer Vision Implementation

There is a number of computer vision elements which can be used in robotic system to make them able to get the information about the environment and manipulation objects.

- cameras
- stereo cameras producing PCL (point cloud data)
- laser scanners
- structured light

Experimental Studies of the Department of CSI

Robotic Boat Setup

ITMO UNIVERSITY

Fig.: Robotic boat

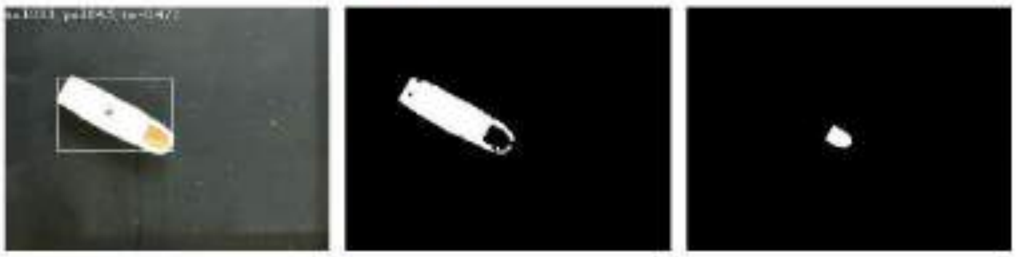


Fig.: Computer vision system

Experimental Studies of the Department of CSI

Manipulators

ITMO UNIVERSITY

Kawasaki FS06N

Mitsubishi MELFA RV-3SDB

KUKA youBot

Experimental Studies of the Department of CSI

Trajectory Planning Using Spline Functions

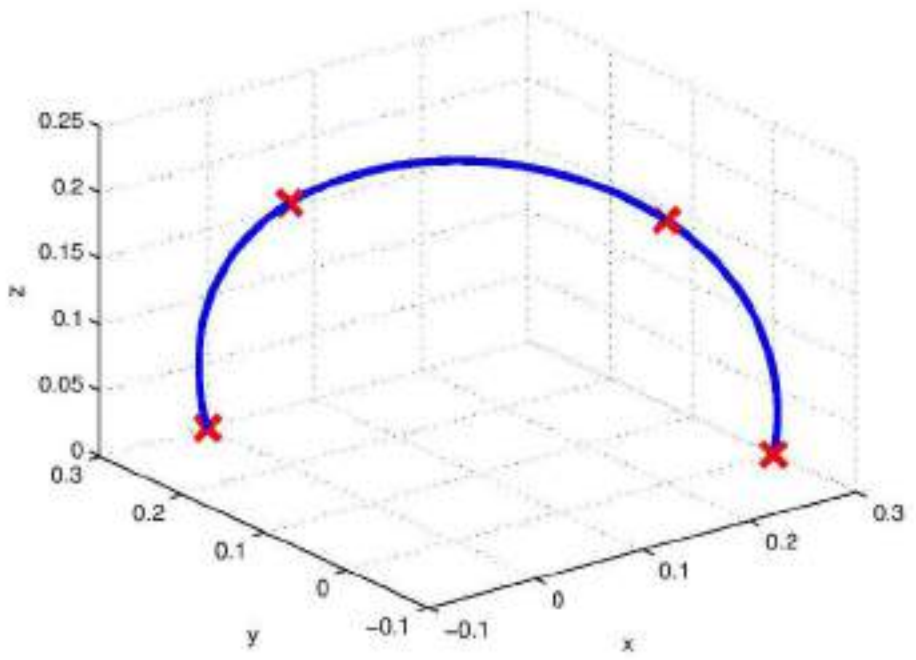
Task Formulation

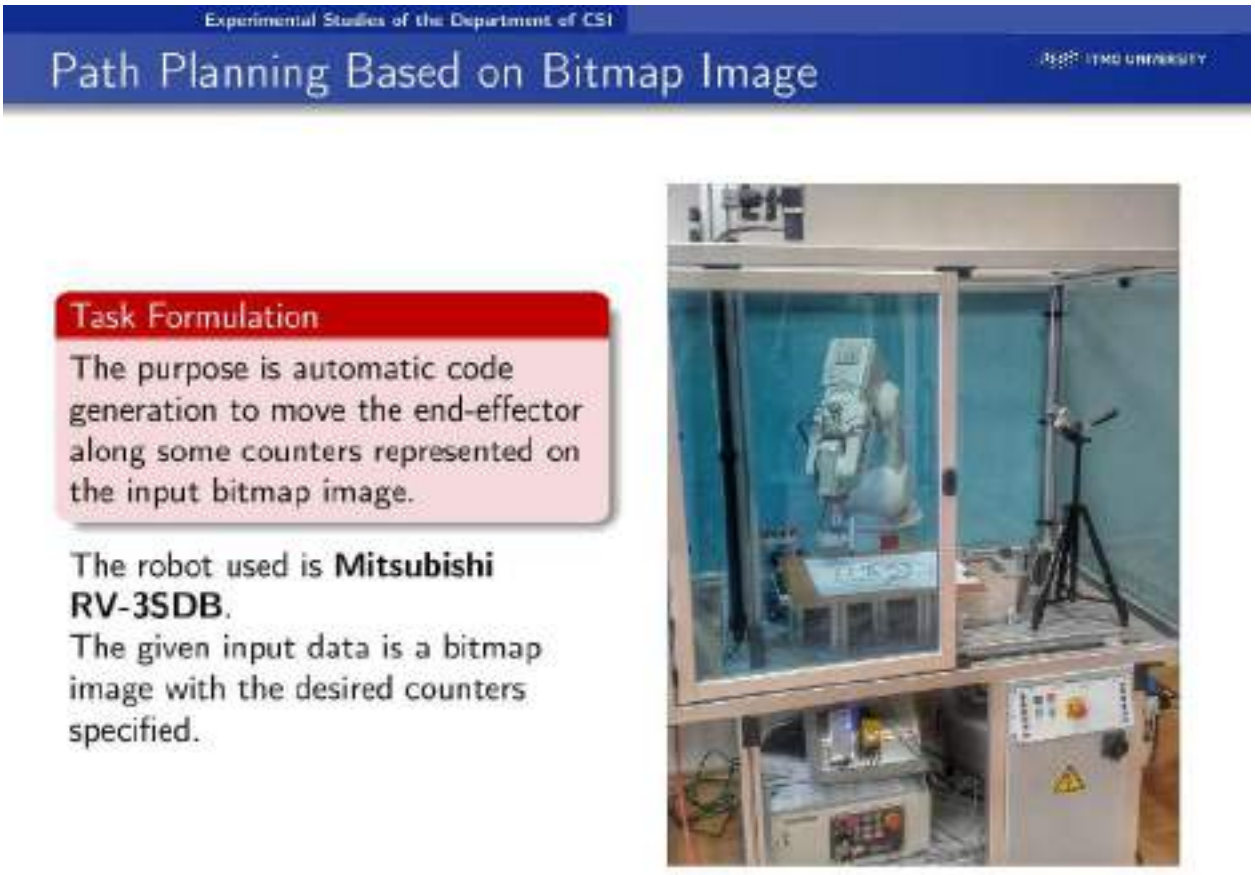
The purpose is to carry out trajectory planning based on four reference points using spline functions.

The robot used is **Kawasaki FS06N**.
The given input data is

- Denavit-Hartenberg parameters of the robot
- Cartesian coordinates of the four reference points: starting, departing, arriving and final points
- time moments assigned to the given reference points

Experimental Studies of the Department of CSI





Experimental Studies of the Department of CSI

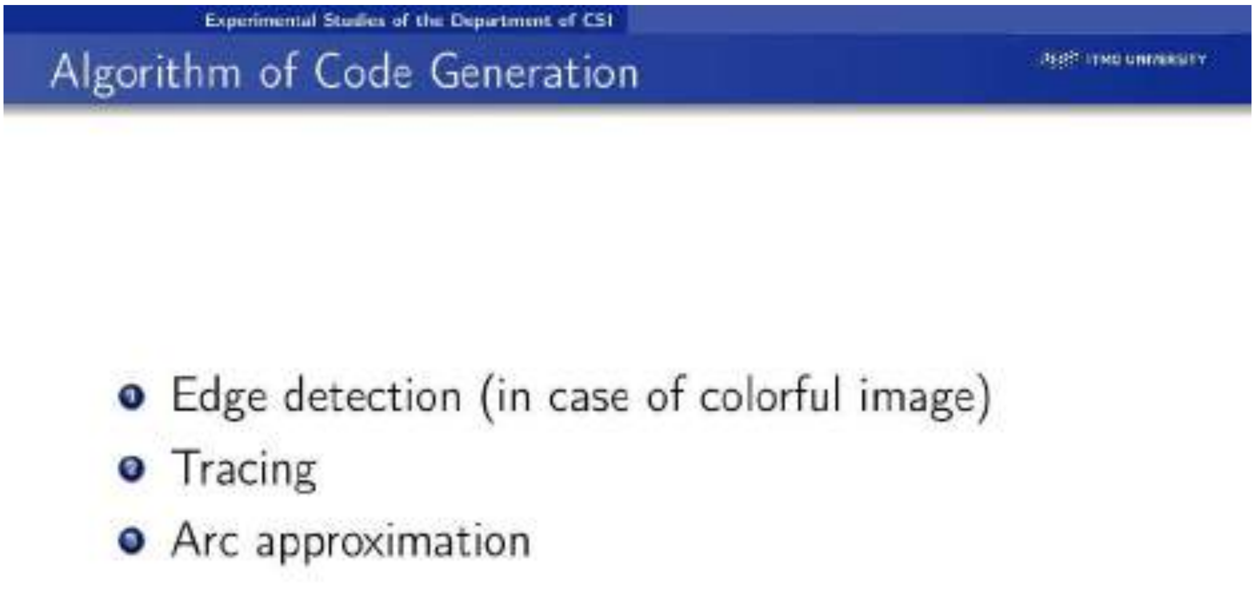
Path Planning Based on Bitmap Image

ITMO UNIVERSITY

Task Formulation

The purpose is automatic code generation to move the end-effector along some counters represented on the input bitmap image.

The robot used is **Mitsubishi RV-3SDB**.
The given input data is a bitmap image with the desired counters specified.



Experimental Studies of the Department of CSI

Algorithm of Code Generation

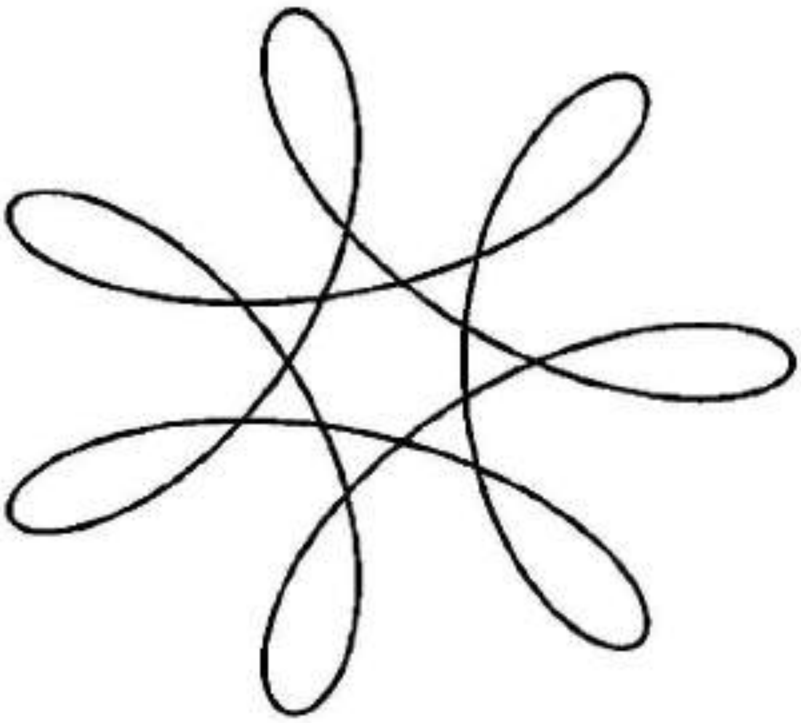
ITMO UNIVERSITY

- Edge detection (in case of colorful image)
- Tracing
- Arc approximation

Experimental Studies of the Department of CSI

Drawing a Hypotrochoid

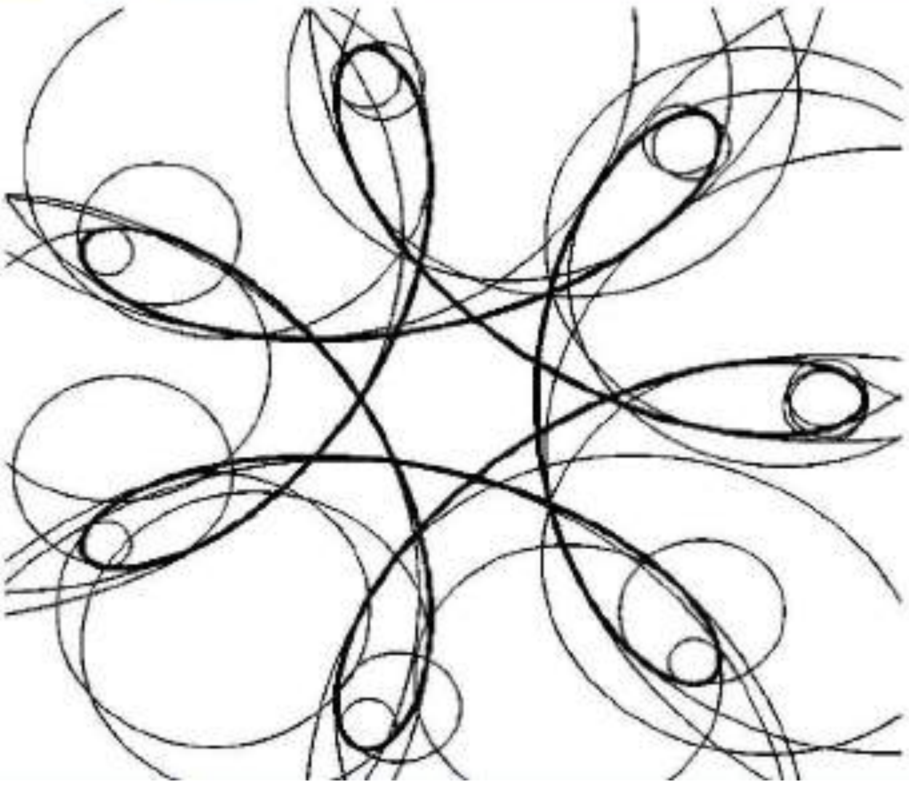
ITMO UNIVERSITY



Experimental Studies of the Department of CSI

Drawing a Hypotrochoid

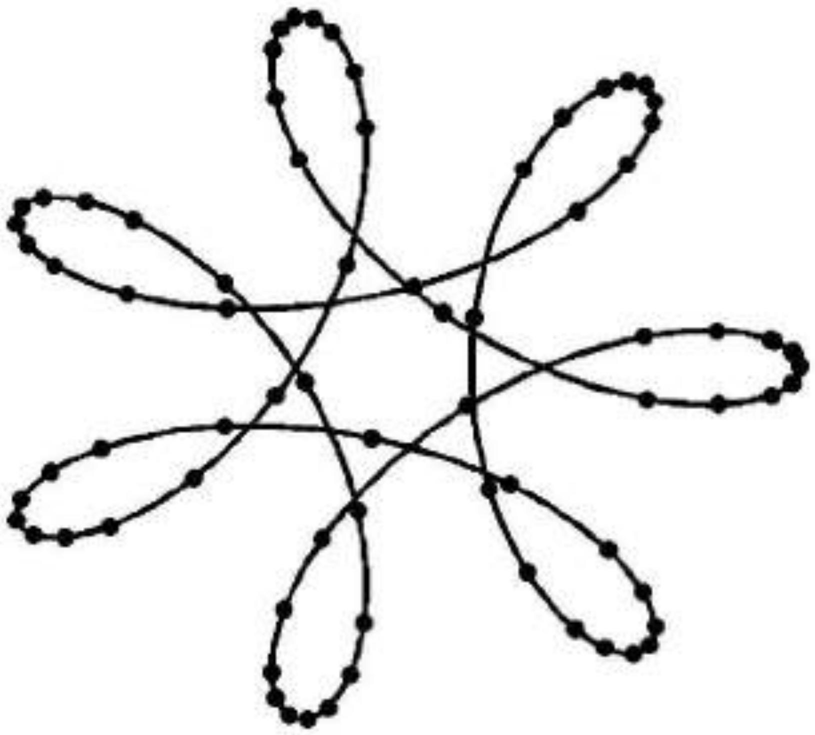
ITMO UNIVERSITY



Experimental Studies of the Department of CSI

Drawing a Hypotrochoid

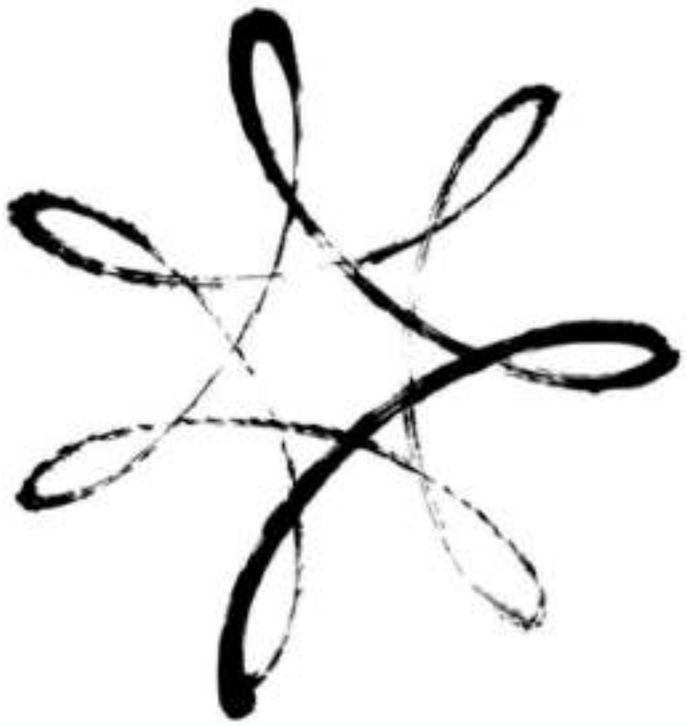
ITMO UNIVERSITY



Experimental Studies of the Department of CSI

Drawing a Hypotrochoid

ITMO UNIVERSITY



Experimental Studies of the Department of CSI

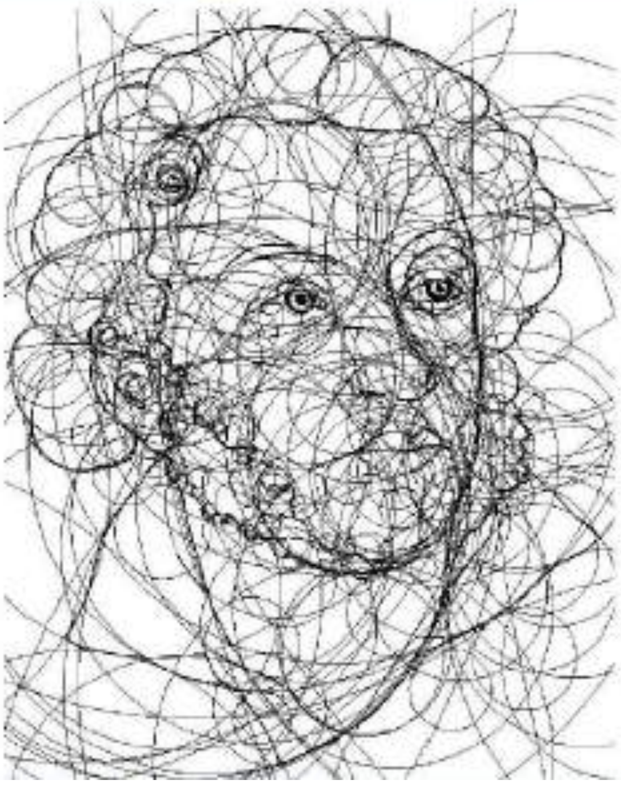
Drawing a Portrait of Alexander Pushkin

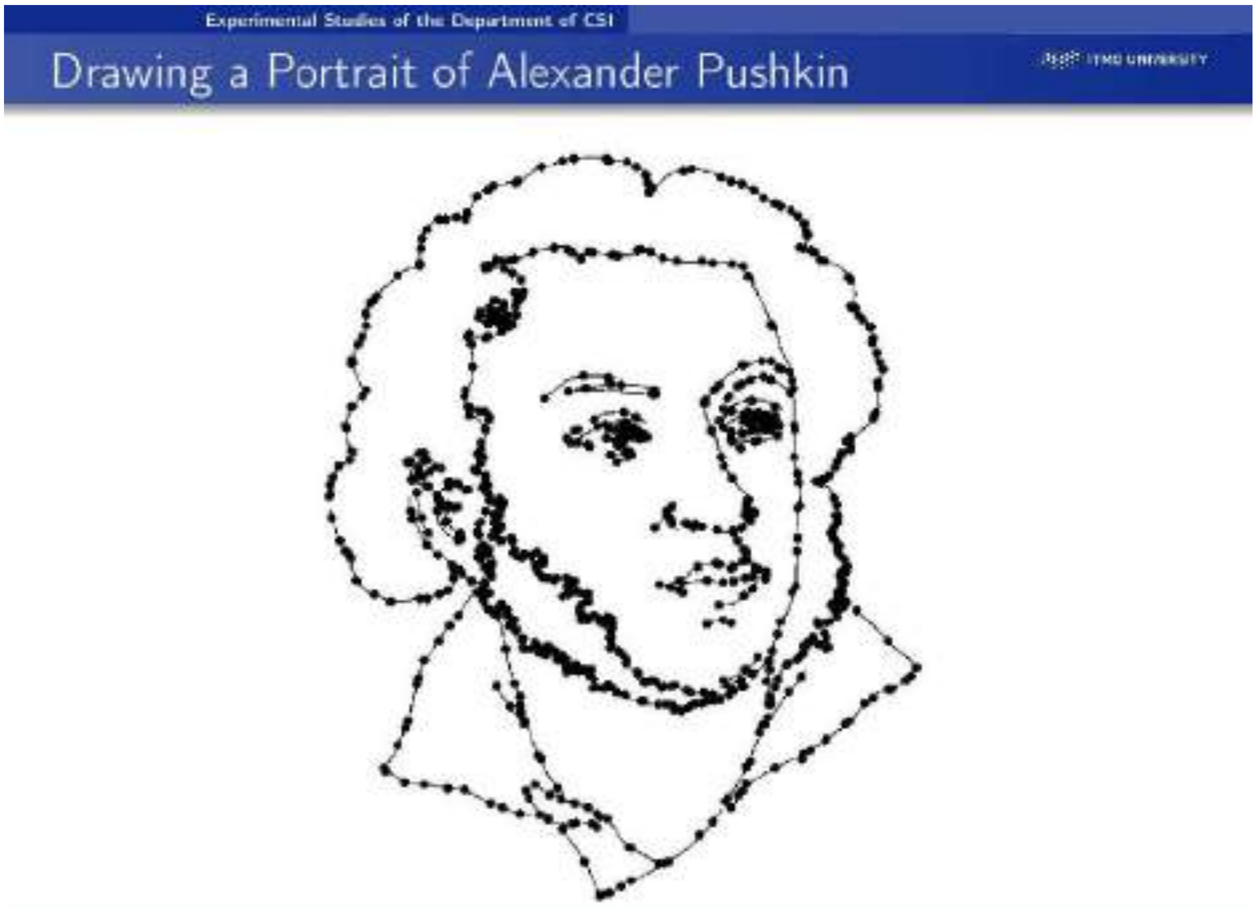
ITMO UNIVERSITY

Experimental Studies of the Department of CSI

Drawing a Portrait of Alexander Pushkin

ITMO UNIVERSITY

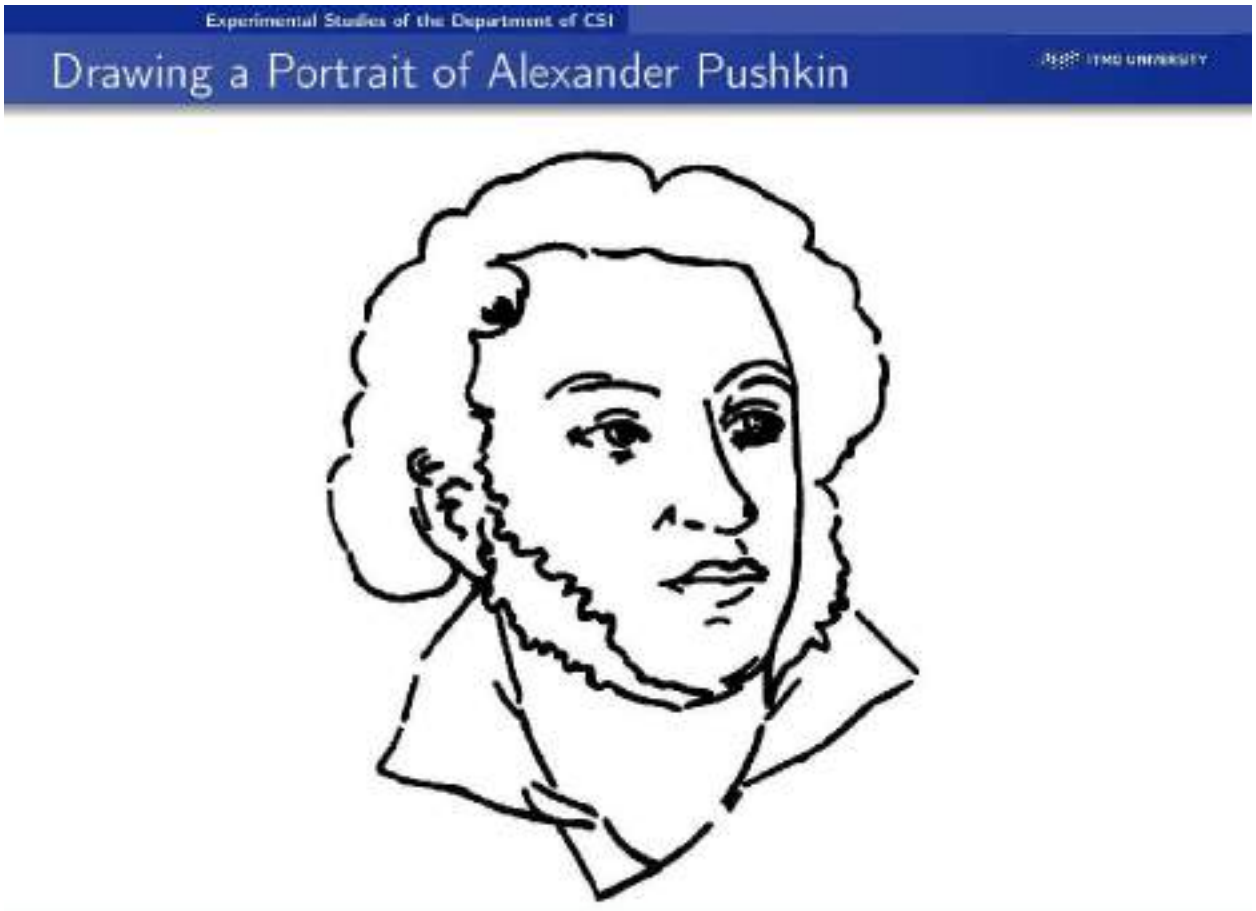




Experimental Studies of the Department of CSI

Drawing a Portrait of Alexander Pushkin

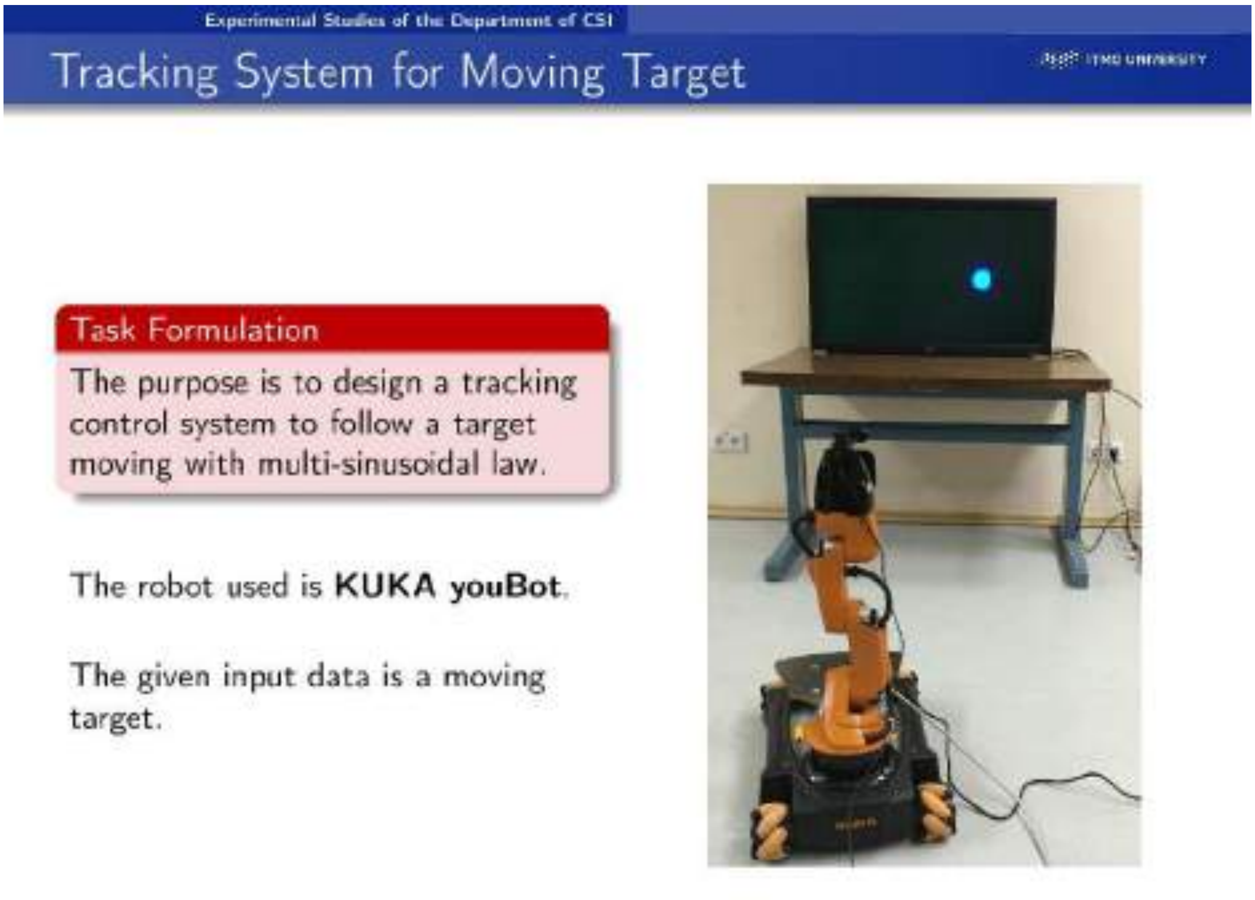
ITMO UNIVERSITY



Experimental Studies of the Department of CSI

Drawing a Portrait of Alexander Pushkin

ITMO UNIVERSITY



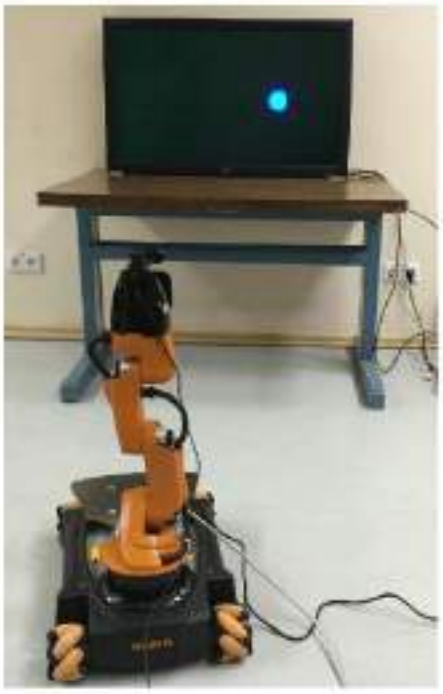
Tracking System for Moving Target

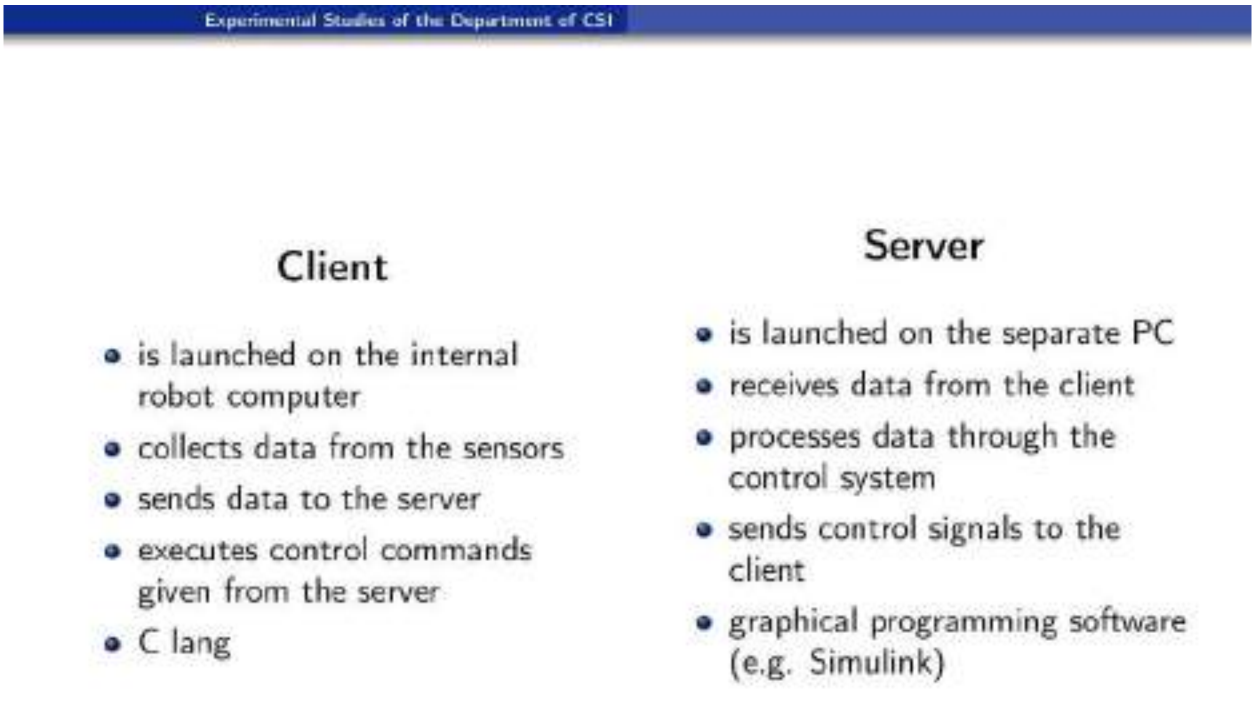
Task Formulation

The purpose is to design a tracking control system to follow a target moving with multi-sinusoidal law.

The robot used is **KUKA youBot**.

The given input data is a moving target.



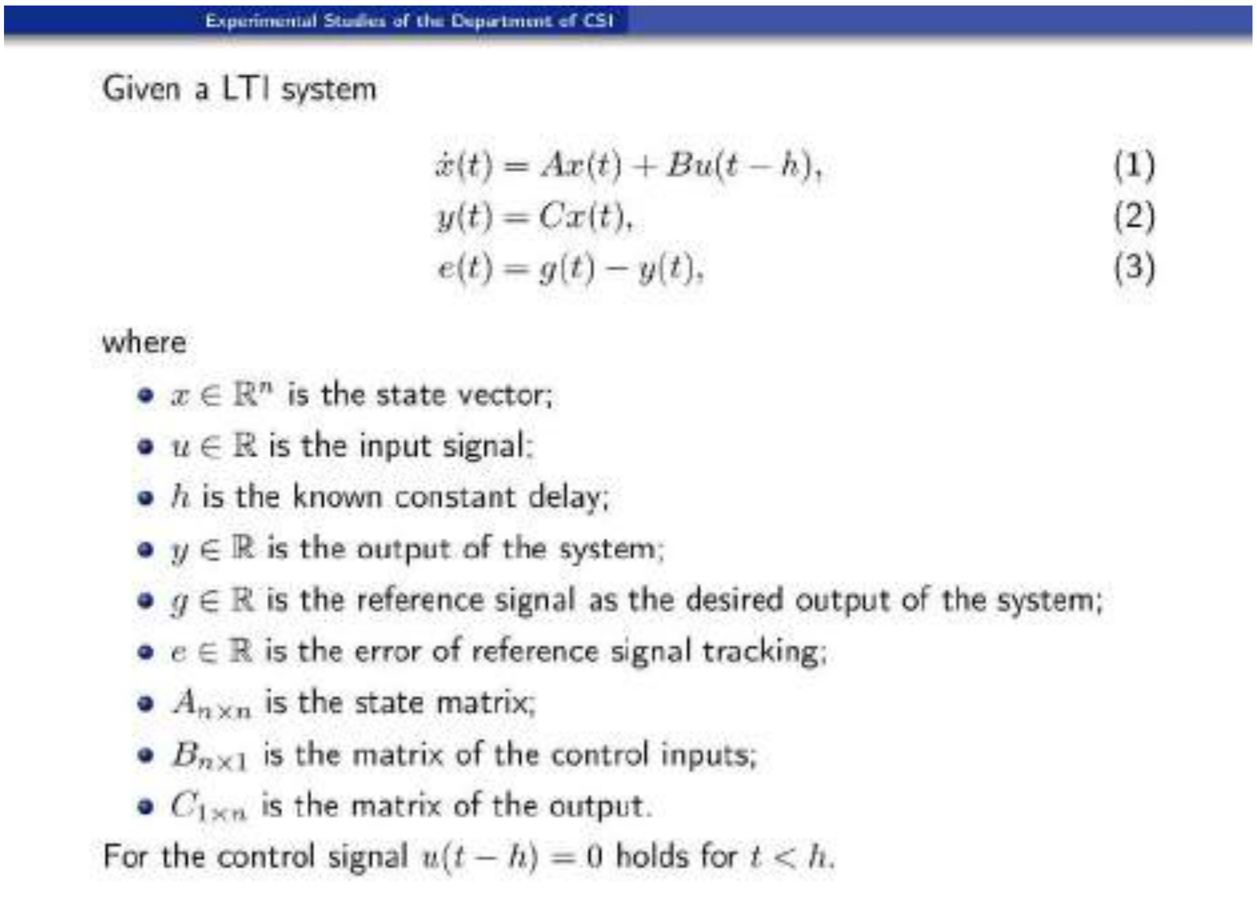


Client

- is launched on the internal robot computer
- collects data from the sensors
- sends data to the server
- executes control commands given from the server
- C lang

Server

- is launched on the separate PC
- receives data from the client
- processes data through the control system
- sends control signals to the client
- graphical programming software (e.g. Simulink)



Experimental Studies of the Department of CSI

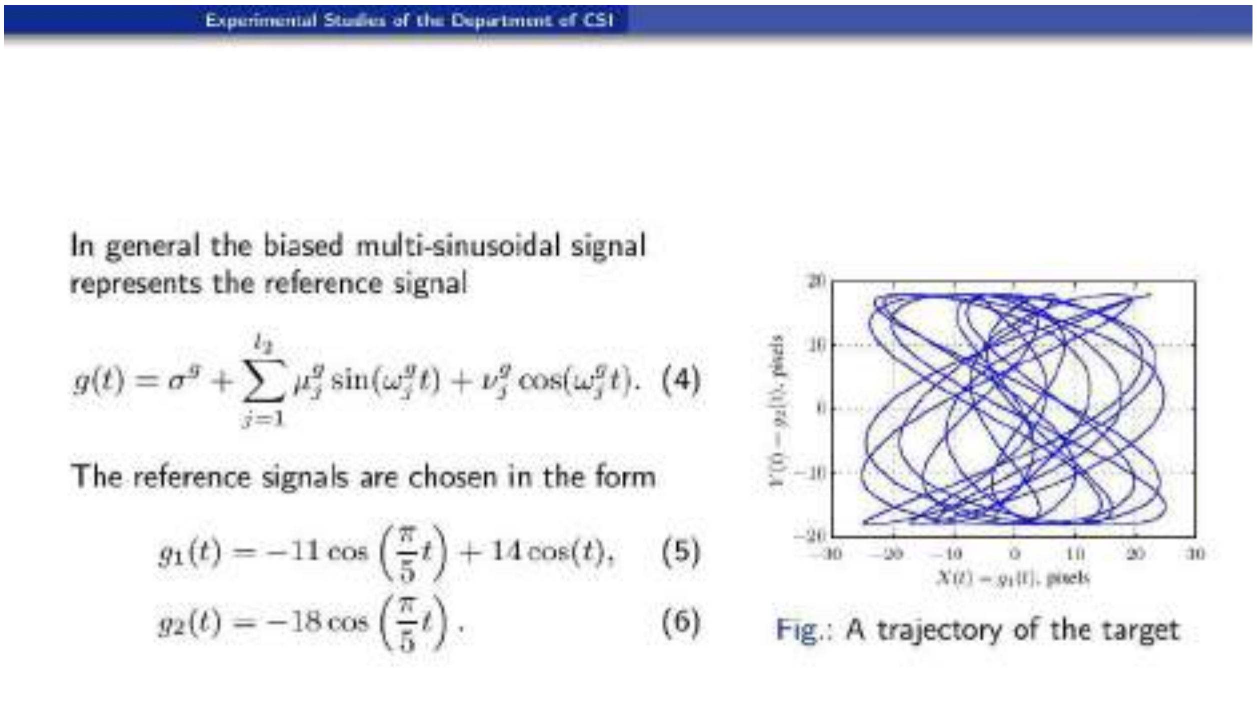
Given a LTI system

$$\dot{x}(t) = Ax(t) + Bu(t-h), \quad (1)$$
$$y(t) = Cx(t), \quad (2)$$
$$e(t) = g(t) - y(t), \quad (3)$$

where

- $x \in \mathbb{R}^n$ is the state vector;
- $u \in \mathbb{R}$ is the input signal;
- h is the known constant delay;
- $y \in \mathbb{R}$ is the output of the system;
- $g \in \mathbb{R}$ is the reference signal as the desired output of the system;
- $e \in \mathbb{R}$ is the error of reference signal tracking;
- $A_{n\times n}$ is the state matrix;
- $B_{n\times 1}$ is the matrix of the control inputs;
- $C_{1\times n}$ is the matrix of the output.

For the control signal $u(t-h) = 0$ holds for $t < h$.



Experimental Studies of the Department of CSI

In general the biased multi-sinusoidal signal represents the reference signal

$$g(t) = \sigma^g + \sum_{j=1}^{l_g} \mu_j^g \sin(\omega_j^g t) + \nu_j^g \cos(\omega_j^g t). \quad (4)$$

The reference signals are chosen in the form

$$g_1(t) = -11\cos\left(\frac{\pi}{5}t\right) + 14\cos(t), \quad (5)$$
$$g_2(t) = -18\cos\left(\frac{\pi}{5}t\right). \quad (6)$$

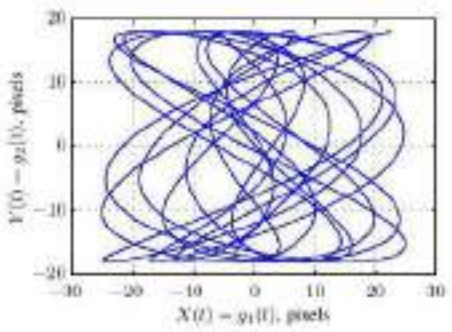
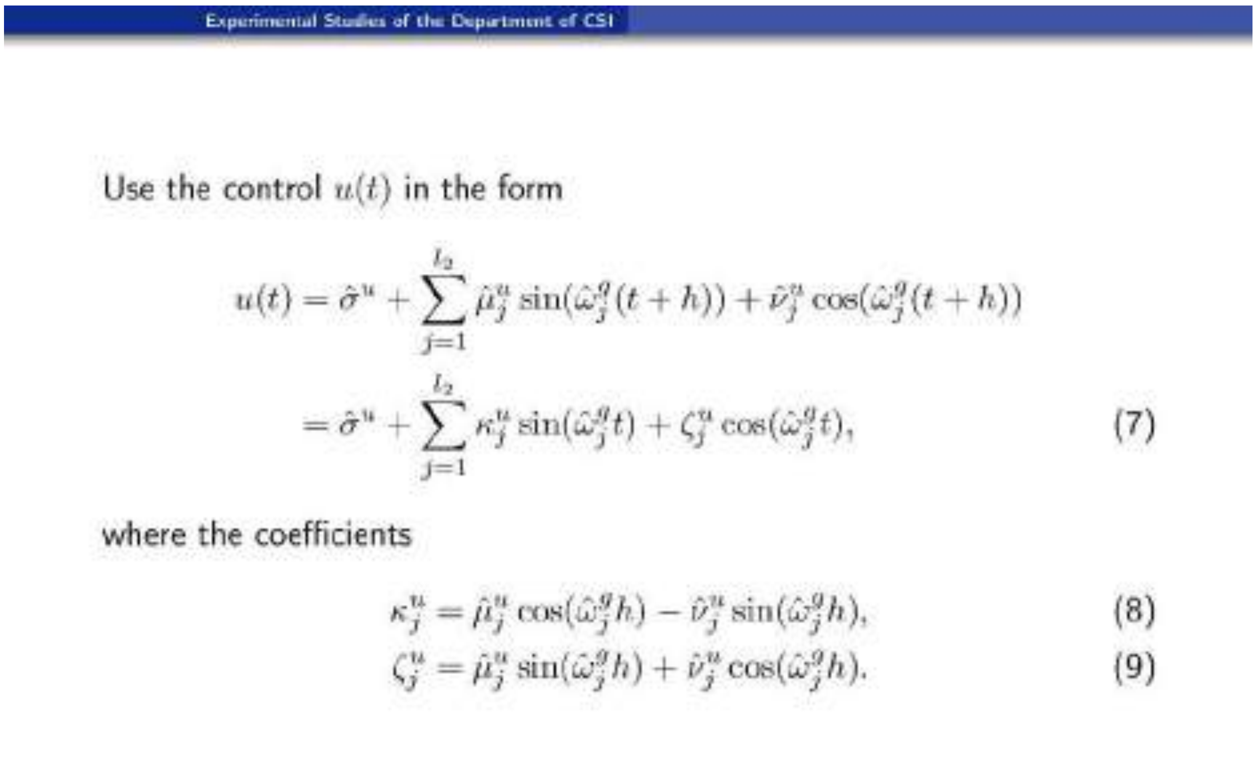


Fig.: A trajectory of the target

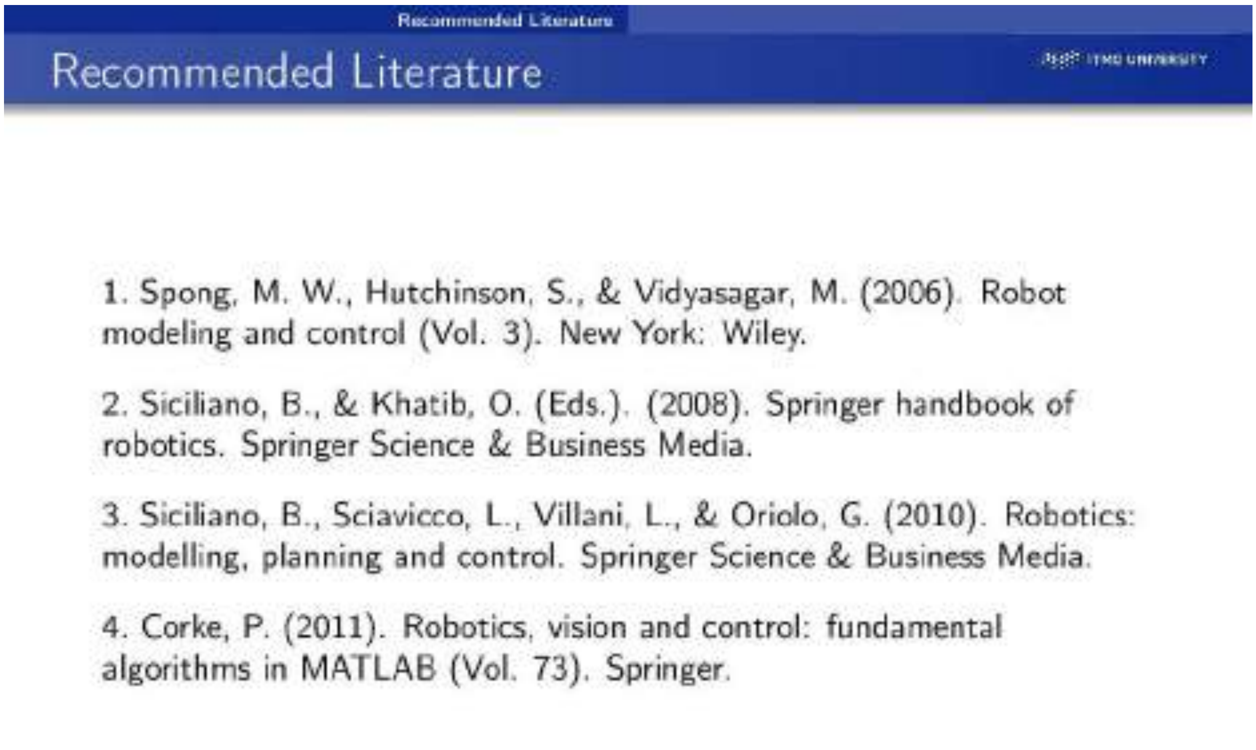


Use the control $u(t)$ in the form

$$u(t) = \hat{\sigma}^u + \sum_{j=1}^{l_2} \hat{\mu}_j^u \sin(\hat{\omega}_j^g (t+h)) + \hat{\nu}_j^u \cos(\hat{\omega}_j^g (t+h))$$
$$= \hat{\sigma}^u + \sum_{j=1}^{l_2} \kappa_j^u \sin(\hat{\omega}_j^g t) + \zeta_j^u \cos(\hat{\omega}_j^g t), \quad (7)$$

where the coefficients

$$\kappa_j^u = \hat{\mu}_j^u \cos(\hat{\omega}_j^g h) - \hat{\nu}_j^u \sin(\hat{\omega}_j^g h), \quad (8)$$
$$\zeta_j^u = \hat{\mu}_j^u \sin(\hat{\omega}_j^g h) + \hat{\nu}_j^u \cos(\hat{\omega}_j^g h). \quad (9)$$



Recommended Literature

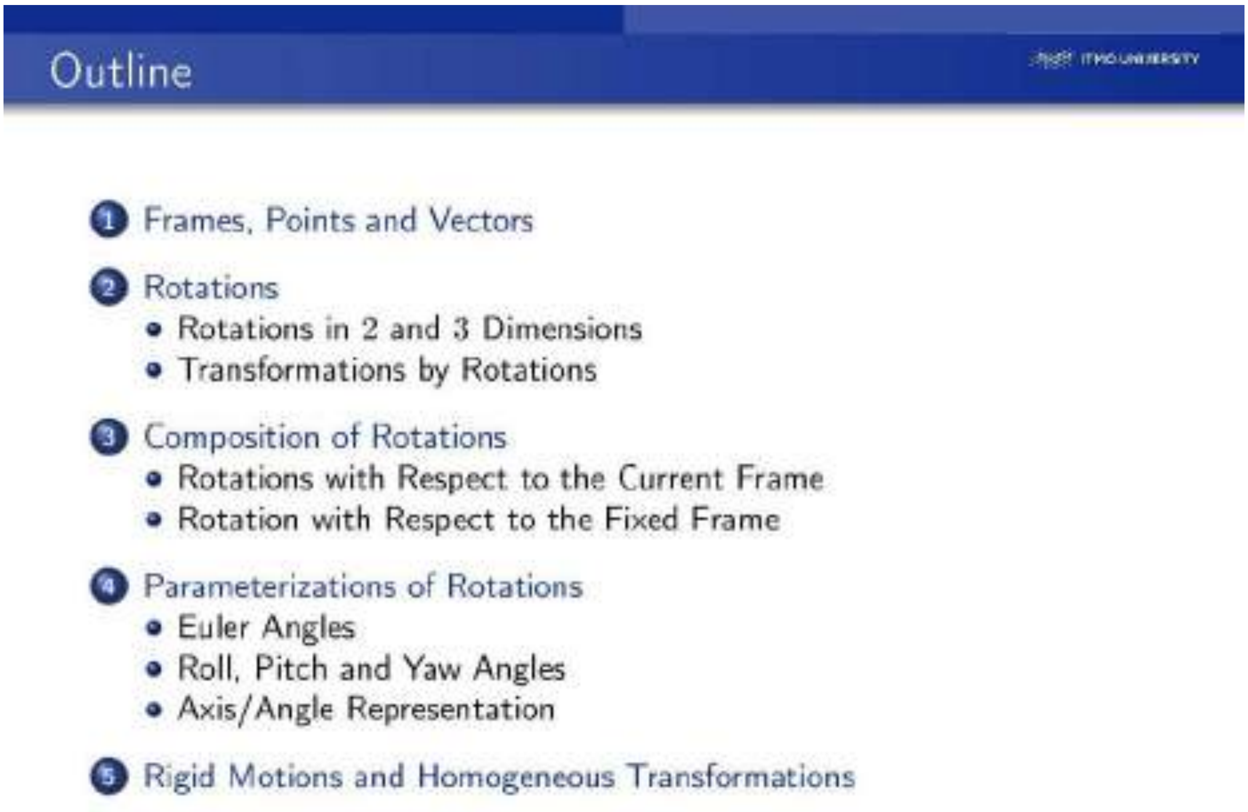
1. Spong, M. W., Hutchinson, S., & Vidyasagar, M. (2006). Robot modeling and control (Vol. 3). New York: Wiley.

2. Siciliano, B., & Khatib, O. (Eds.). (2008). Springer handbook of robotics. Springer Science & Business Media.

3. Siciliano, B., Sciavicco, L., Villani, L., & Oriolo, G. (2010). Robotics: modelling, planning and control. Springer Science & Business Media.

4. Corke, P. (2011). Robotics, vision and control: fundamental algorithms in MATLAB (Vol. 73). Springer.

Kinematics: Rigid motions and Homogeneous Transformations



Outline

1. Frames, Points and Vectors
2. Rotations
   - Rotations in 2 and 3 Dimensions
   - Transformations by Rotations
3. Composition of Rotations
   - Rotations with Respect to the Current Frame
   - Rotation with Respect to the Fixed Frame
4. Parameterizations of Rotations
   - Euler Angles
   - Roll, Pitch and Yaw Angles
   - Axis/Angle Representation
5. Rigid Motions and Homogeneous Transformations

Frames, Points and Vectors

Frames, Points and Vectors

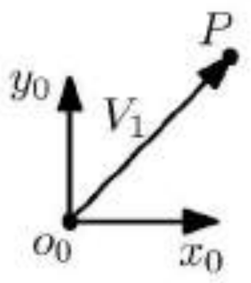


Fig.: A coordinate frame in R^2: the point $P = [x_p^0, y_p^0]$ can be associated with the vector $\vec{V}_1$. Here x_p^0 denotes x-coordinate of point P in (o_0, x_0, y_0)-frame, i.e. in the 0-frame.

Frames, Points and Vectors

Frames, Points and Vectors

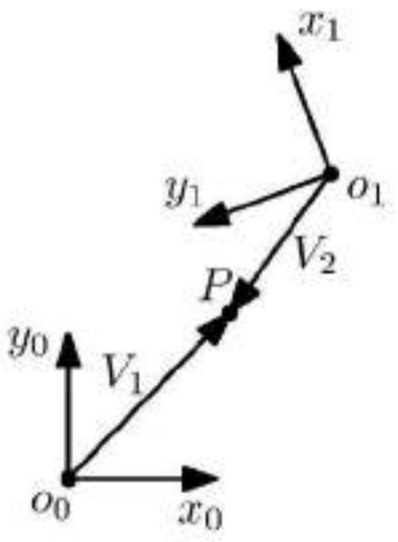


Fig.: Two coordinate frames in R^2: the point $P = [x_p^0, y_p^0] = [x_p^1, y_p^1]$ can be associated with vectors $\vec{V_1}$ and $\vec{V_2}$. Here x_p^1 denotes x-coordinate of point P in the 1-frame.

Frames, Points and Vectors

Frames, Points and Vectors

- **A point** corresponds a particular location in the space
- **A point** has different representation (coordinates) in different frames
- **A vector** is defined by direction and magnitude
- **Vectors** with the same direction and magnitude are the same

Frames, Points and Vectors

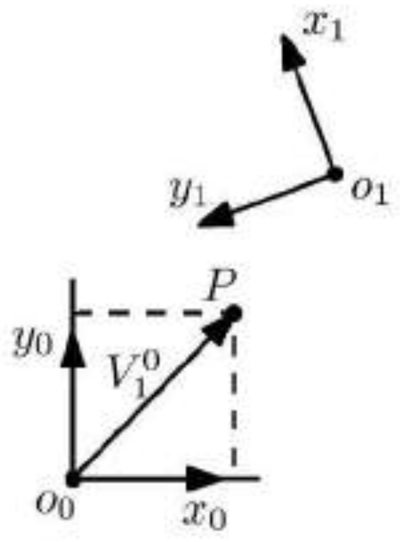


Fig.: The coordinates of the vector $\vec{V}_1$ in the 0-frame $[x_p^0, y_p^0]$.
What would be coordinates of $\vec{V}_1$ in the 1-frame?

Frames, Points and Vectors

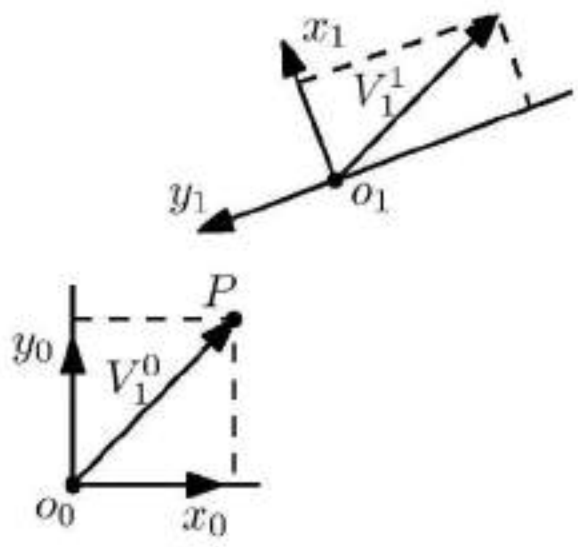


Fig.: We need to consider the vector $\vec{V}_1^1$ of the same direction and magnitude as $\vec{V}_1$ but with the origin in o_1. Conclusion: we can sum vectors only if they are expressed in parallel frames.

Rotations in 2D

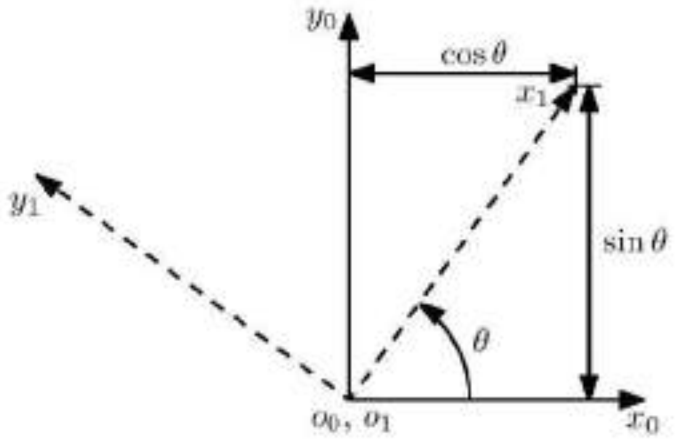


Fig.: To find an appropriate way to parametrize rotations in 2-D, let us track vectors $x_1^0(\cdot)$, $y_1^0(\cdot)$ as θ varies (courtesy Spong).

$$x_1^0(\theta) = \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix}, \quad y_1^0(\theta) = x_1^0\left(\theta + \frac{\pi}{2}\right) = \begin{bmatrix} \cos\left(\theta + \frac{\pi}{2}\right) \\ \sin\left(\theta + \frac{\pi}{2}\right) \end{bmatrix} = \begin{bmatrix} -\sin(\theta) \\ \cos(\theta) \end{bmatrix}$$

Rotations in 2D

The matrix

$$R(\theta) = \left[x_1^0(\theta) \,|\, y_1^0(\theta)\right] = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

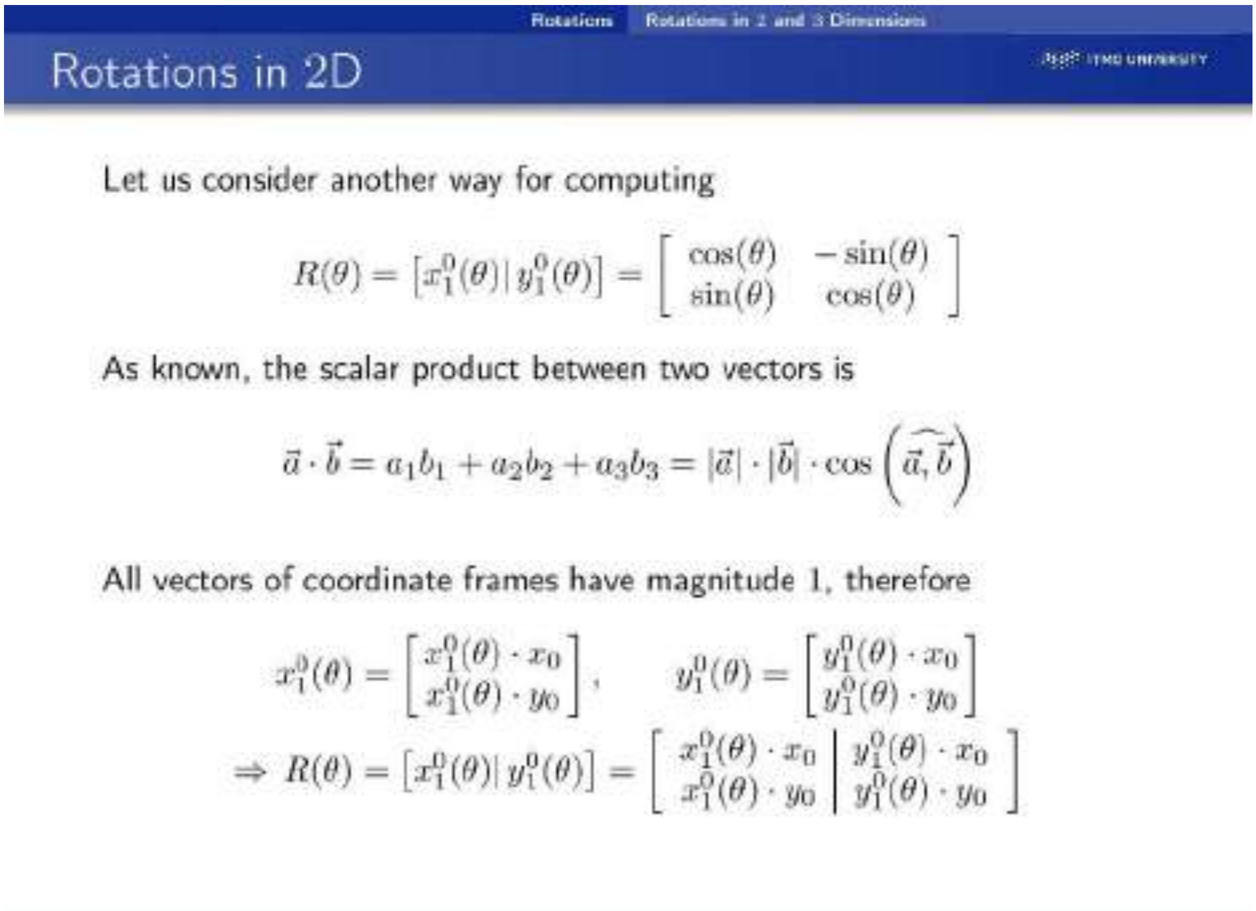
is called **rotation matrix**.
It has a number of interesting properties:

- $\det R(\theta) = \cos^2(\theta) + \sin^2(\theta) = 1$
- $R(\theta)^{-1} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} = R^T(\theta)$

A $n \times n$-matrix X that satisfies the property, $X^{-1} = X^T$

$$\Rightarrow \left\{XX^T = I_n,\ \det(XX^T) = 1 = \det(X)\det(X^T) = \det(X)^2\right\}$$

is called orthogonal, $X \in \mathcal{O}(n)$. If $\det X = 1 \Rightarrow X \in \mathcal{SO}(n)$.



Rotations | Rotations in 2 and 3 Dimensions

Rotations in 2D

Let us consider another way for computing

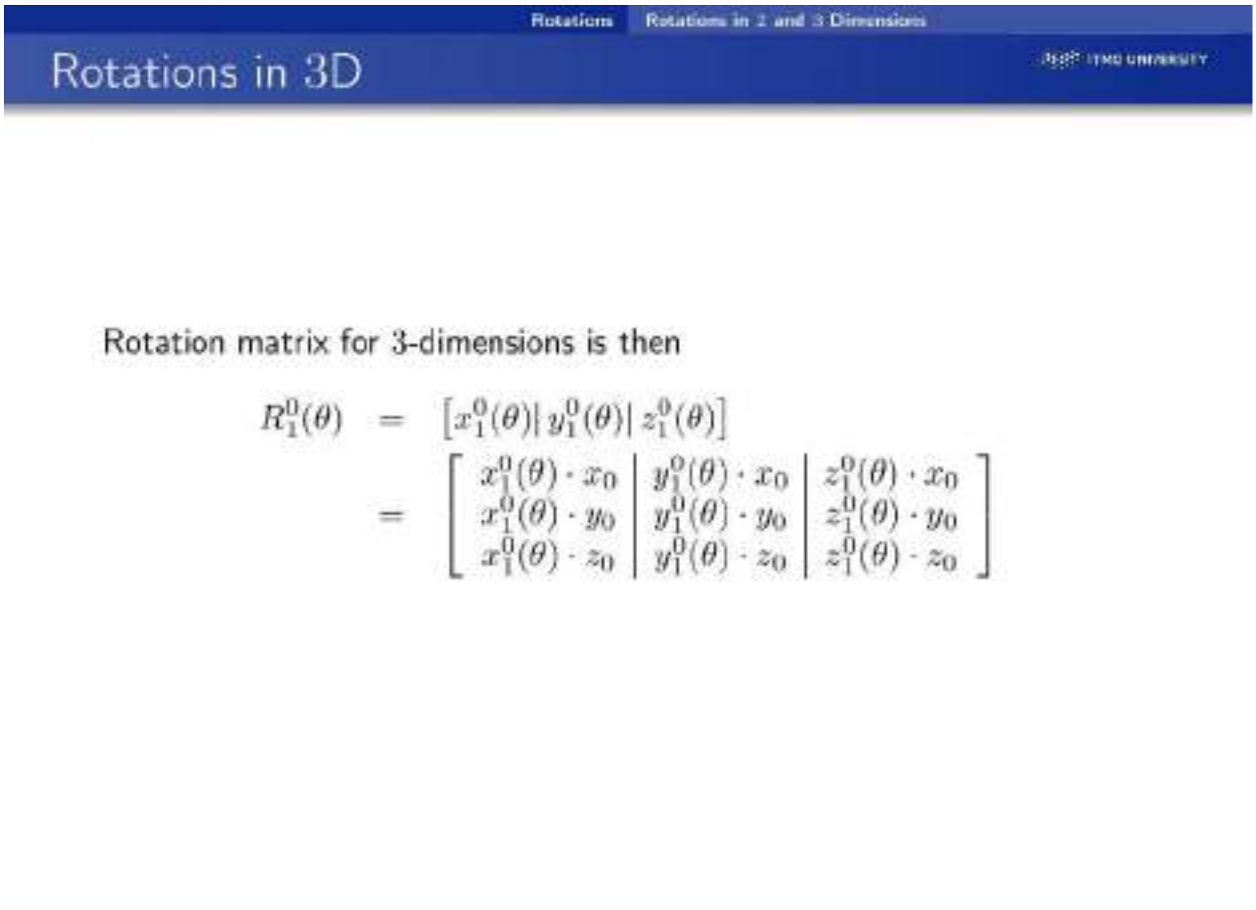
$$R(\theta) = [x_1^0(\theta) \,|\, y_1^0(\theta)] = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

As known, the scalar product between two vectors is

$$\vec{a} \cdot \vec{b} = a_1 b_1 + a_2 b_2 + a_3 b_3 = |\vec{a}| \cdot |\vec{b}| \cdot \cos\left(\widehat{\vec{a}, \vec{b}}\right)$$

All vectors of coordinate frames have magnitude 1, therefore

$$x_1^0(\theta) = \begin{bmatrix} x_1^0(\theta) \cdot x_0 \\ x_1^0(\theta) \cdot y_0 \end{bmatrix}, \qquad y_1^0(\theta) = \begin{bmatrix} y_1^0(\theta) \cdot x_0 \\ y_1^0(\theta) \cdot y_0 \end{bmatrix}$$
$$\Rightarrow R(\theta) = [x_1^0(\theta) \,|\, y_1^0(\theta)] = \left[\begin{array}{c|c} x_1^0(\theta) \cdot x_0 & y_1^0(\theta) \cdot x_0 \\ x_1^0(\theta) \cdot y_0 & y_1^0(\theta) \cdot y_0 \end{array}\right]$$

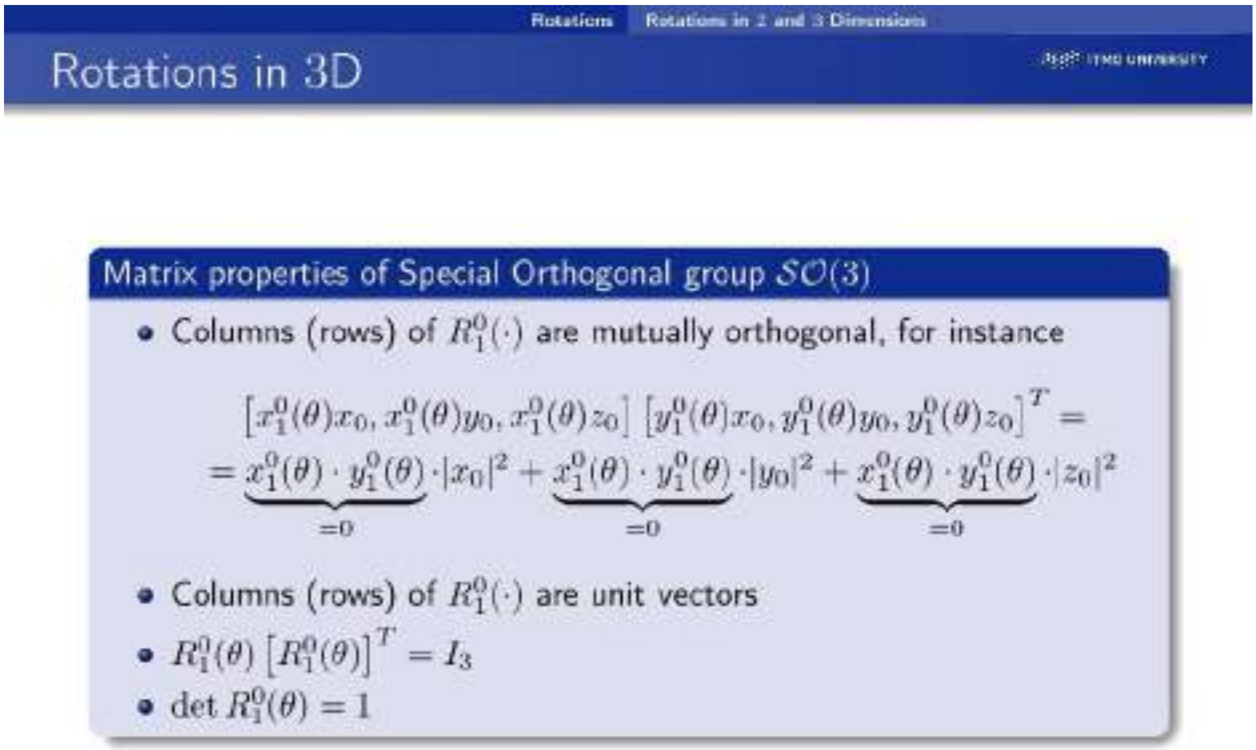


Rotations | Rotations in 2 and 3 Dimensions

Rotations in 3D

Rotation matrix for 3-dimensions is then

$$\begin{aligned} R_1^0(\theta) &= [x_1^0(\theta) \,|\, y_1^0(\theta) \,|\, z_1^0(\theta)] \\ &= \left[\begin{array}{c|c|c} x_1^0(\theta) \cdot x_0 & y_1^0(\theta) \cdot x_0 & z_1^0(\theta) \cdot x_0 \\ x_1^0(\theta) \cdot y_0 & y_1^0(\theta) \cdot y_0 & z_1^0(\theta) \cdot y_0 \\ x_1^0(\theta) \cdot z_0 & y_1^0(\theta) \cdot z_0 & z_1^0(\theta) \cdot z_0 \end{array}\right] \end{aligned}$$



Rotations | Rotations in 2 and 3 Dimensions

Rotations in 3D

Matrix properties of Special Orthogonal group $\mathcal{SO}(3)$

- Columns (rows) of $R_1^0(\cdot)$ are mutually orthogonal, for instance

$$\left[x_1^0(\theta)x_0, x_1^0(\theta)y_0, x_1^0(\theta)z_0\right]\left[y_1^0(\theta)x_0, y_1^0(\theta)y_0, y_1^0(\theta)z_0\right]^T =$$
$$= \underbrace{x_1^0(\theta)\cdot y_1^0(\theta)}_{=0}\cdot|x_0|^2 + \underbrace{x_1^0(\theta)\cdot y_1^0(\theta)}_{=0}\cdot|y_0|^2 + \underbrace{x_1^0(\theta)\cdot y_1^0(\theta)}_{=0}\cdot|z_0|^2$$

- Columns (rows) of $R_1^0(\cdot)$ are unit vectors
- $R_1^0(\theta)\left[R_1^0(\theta)\right]^T = I_3$
- $\det R_1^0(\theta) = 1$

Rotations | Rotations in 2 and 3 Dimensions

Example 2.1: Rotation in 3D

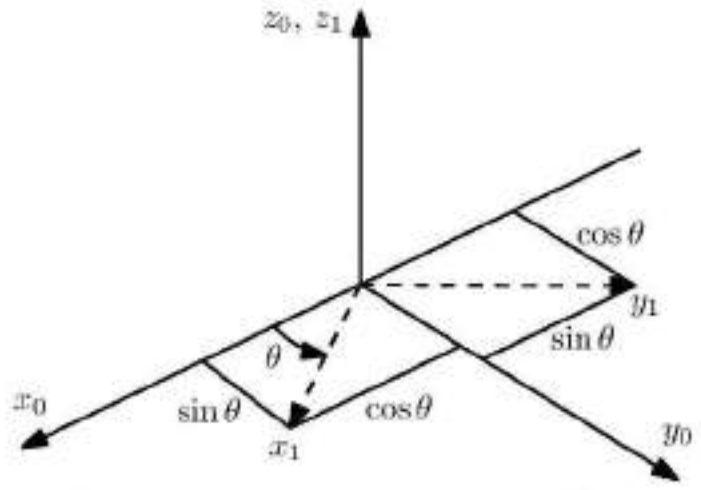
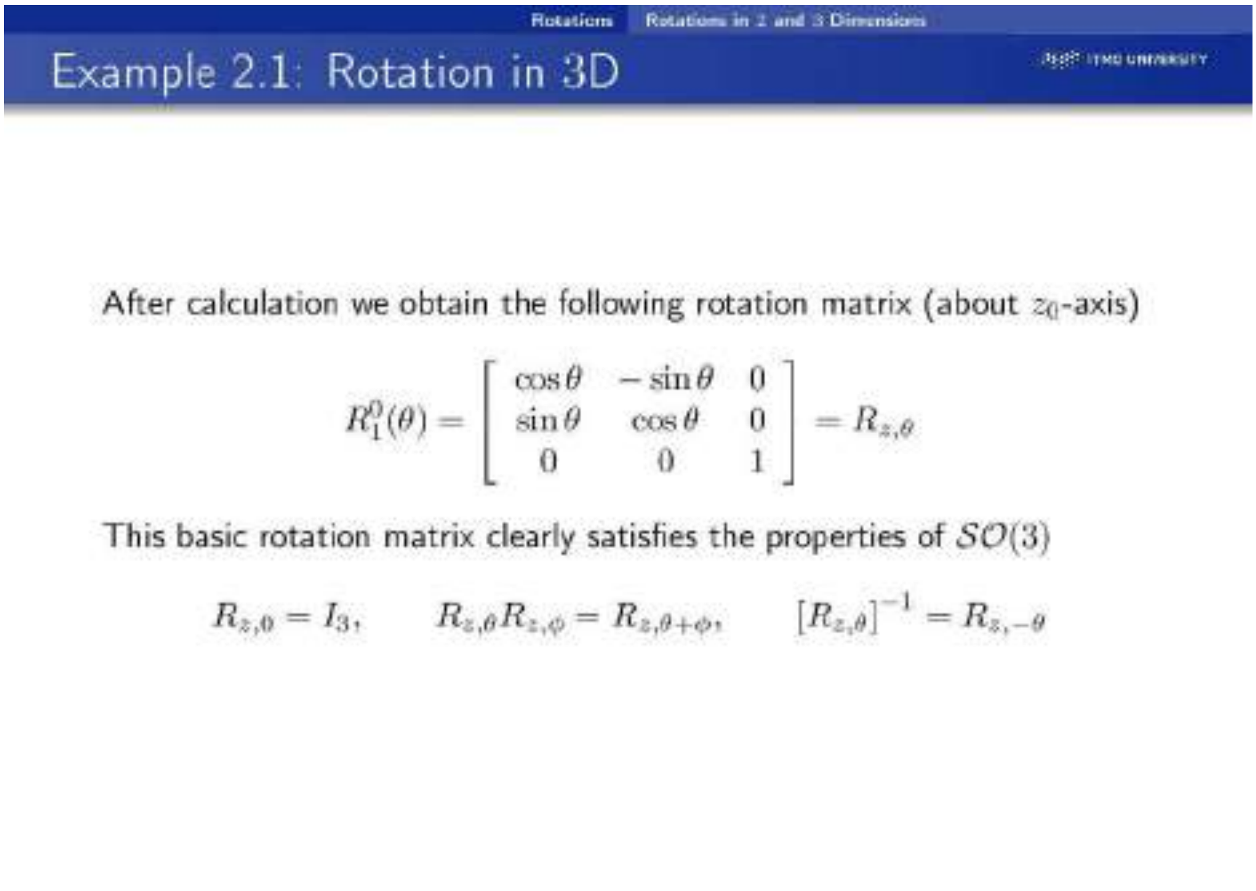


Fig.: Frame 1 is rotated about z_0-axis by an angle θ (courtesy Spong).

$$\begin{array}{rclrclrcl}
x_1^0(\theta)x_0 &=& \cos\theta & y_1^0(\theta)x_0 &=& -\sin\theta & z_1^0(\theta)x_0 &=& 0\\
x_1^0(\theta)y_0 &=& \sin\theta & y_1^0(\theta)y_0 &=& \cos\theta & z_1^0(\theta)y_0 &=& 0\\
x_1^0(\theta)z_0 &=& 0 & y_1^0(\theta)z_0 &=& 0 & z_1^0(\theta)z_0 &=& 1
\end{array}$$



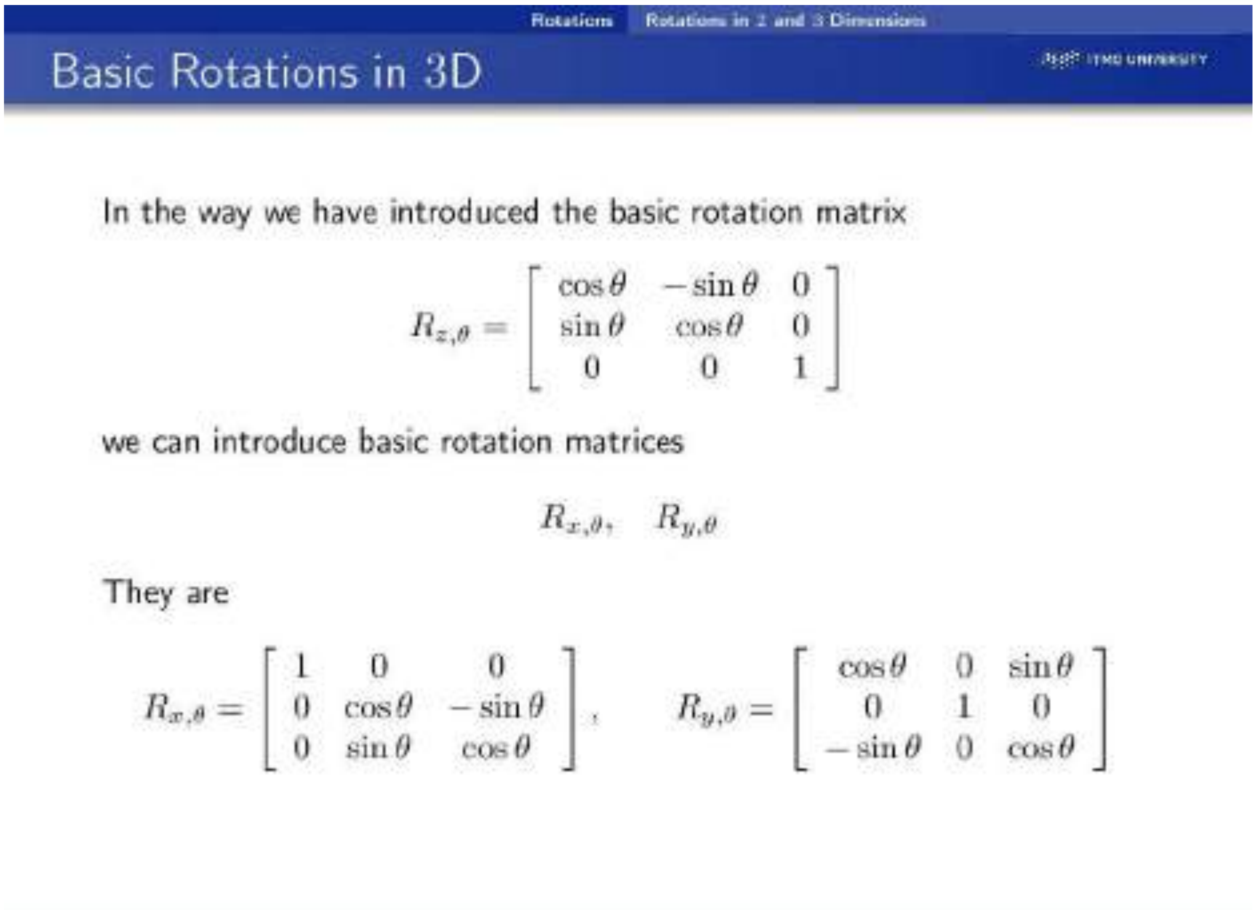
Example 2.1: Rotation in 3D

After calculation we obtain the following rotation matrix (about z_0-axis)

$$R_1^0(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} = R_{z,\theta}$$

This basic rotation matrix clearly satisfies the properties of $\mathcal{SO}(3)$

$$R_{z,0} = I_3, \qquad R_{z,\theta}R_{z,\phi} = R_{z,\theta+\phi}, \qquad \left[R_{z,\theta}\right]^{-1} = R_{z,-\theta}$$



Basic Rotations in 3D

In the way we have introduced the basic rotation matrix

$$R_{z,\theta} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

we can introduce basic rotation matrices

$$R_{x,\theta}, \quad R_{y,\theta}$$

They are

$$R_{x,\theta} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}, \qquad R_{y,\theta} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

Transformations by Rotations in 3D

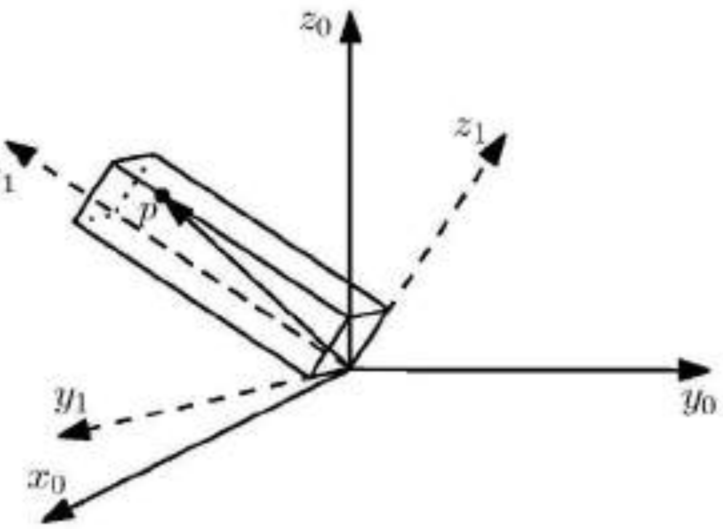


Fig.: The 0-frame is our world, the 1-frame is attached to a rigid body (courtesy Spong).

What will happen with points of the body (say p) if we rotate the body, i.e. the 1-frame?

Transformations by Rotations in 3D

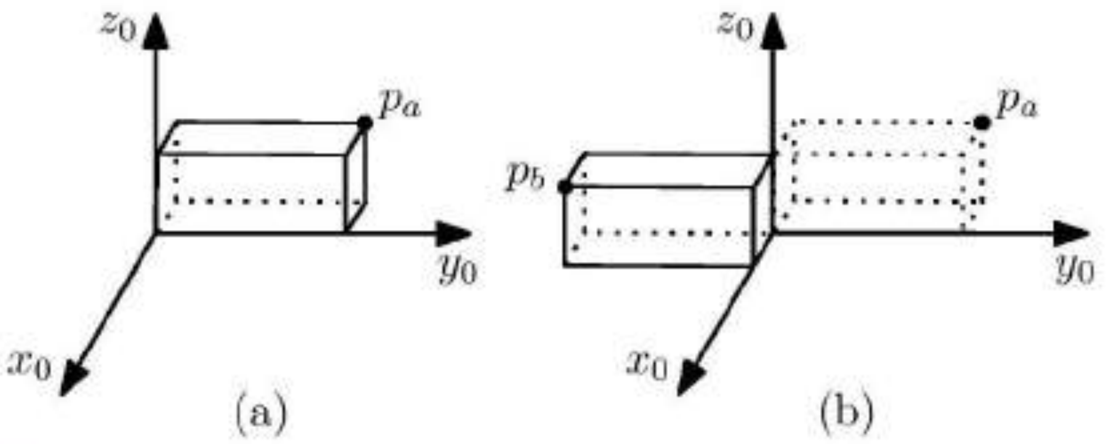
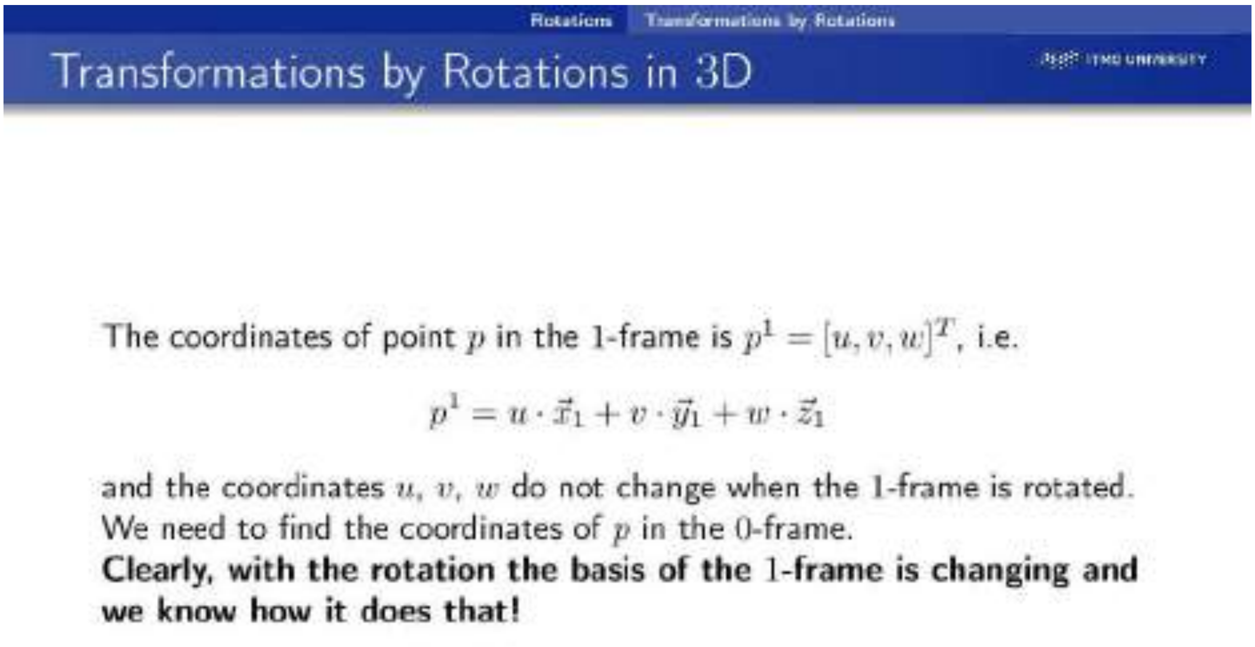


Fig.: We are interested in coordinates of the point p, which is constant in the body attached 1-frame but it changes in the 0-frame (courtesy Spong).

How to trace the change of position of body point p in the 0-frame?

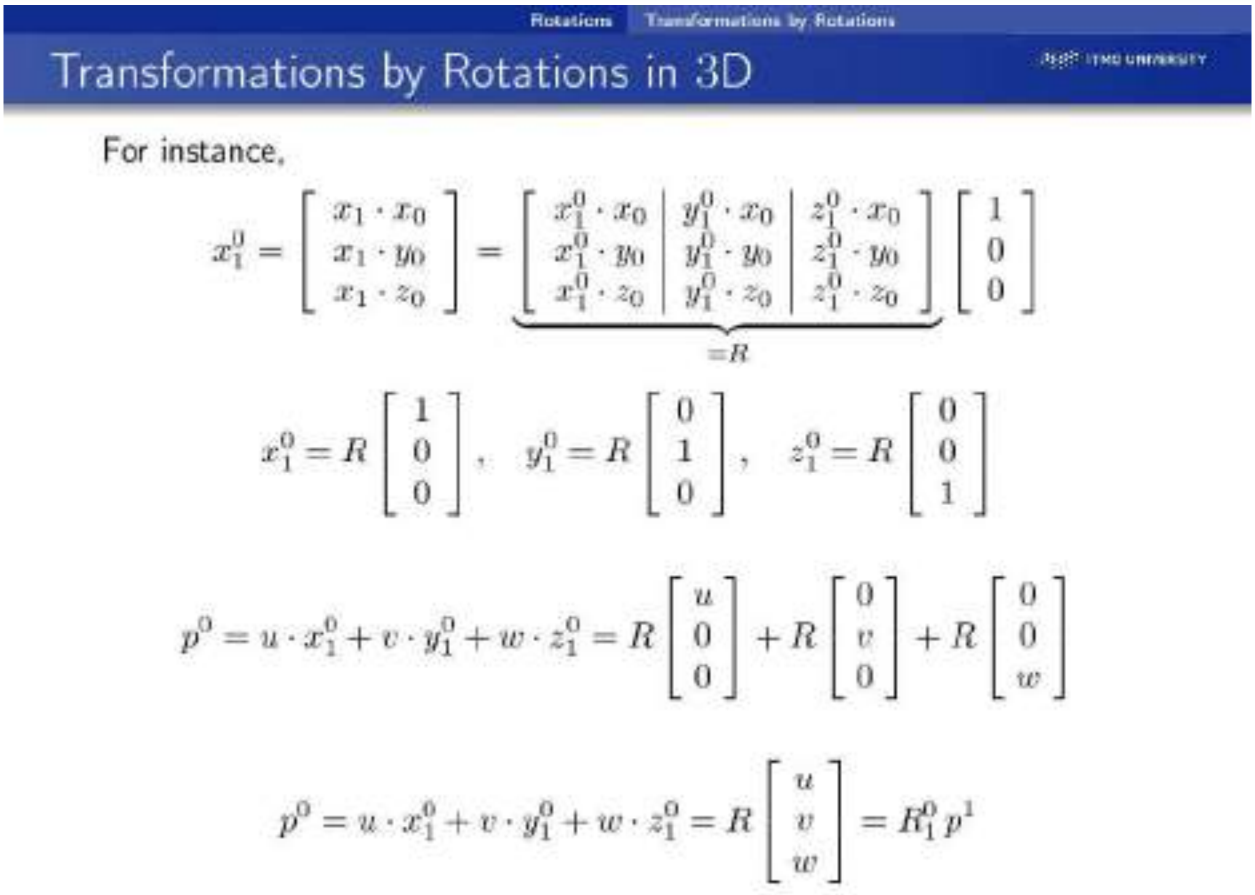


Transformations by Rotations in 3D

The coordinates of point p in the 1-frame is $p^1 = [u, v, w]^T$, i.e.

$$p^1 = u \cdot \vec{x}_1 + v \cdot \vec{y}_1 + w \cdot \vec{z}_1$$

and the coordinates u, v, w do not change when the 1-frame is rotated. We need to find the coordinates of p in the 0-frame.
Clearly, with the rotation the basis of the 1-frame is changing and we know how it does that!



Transformations by Rotations in 3D

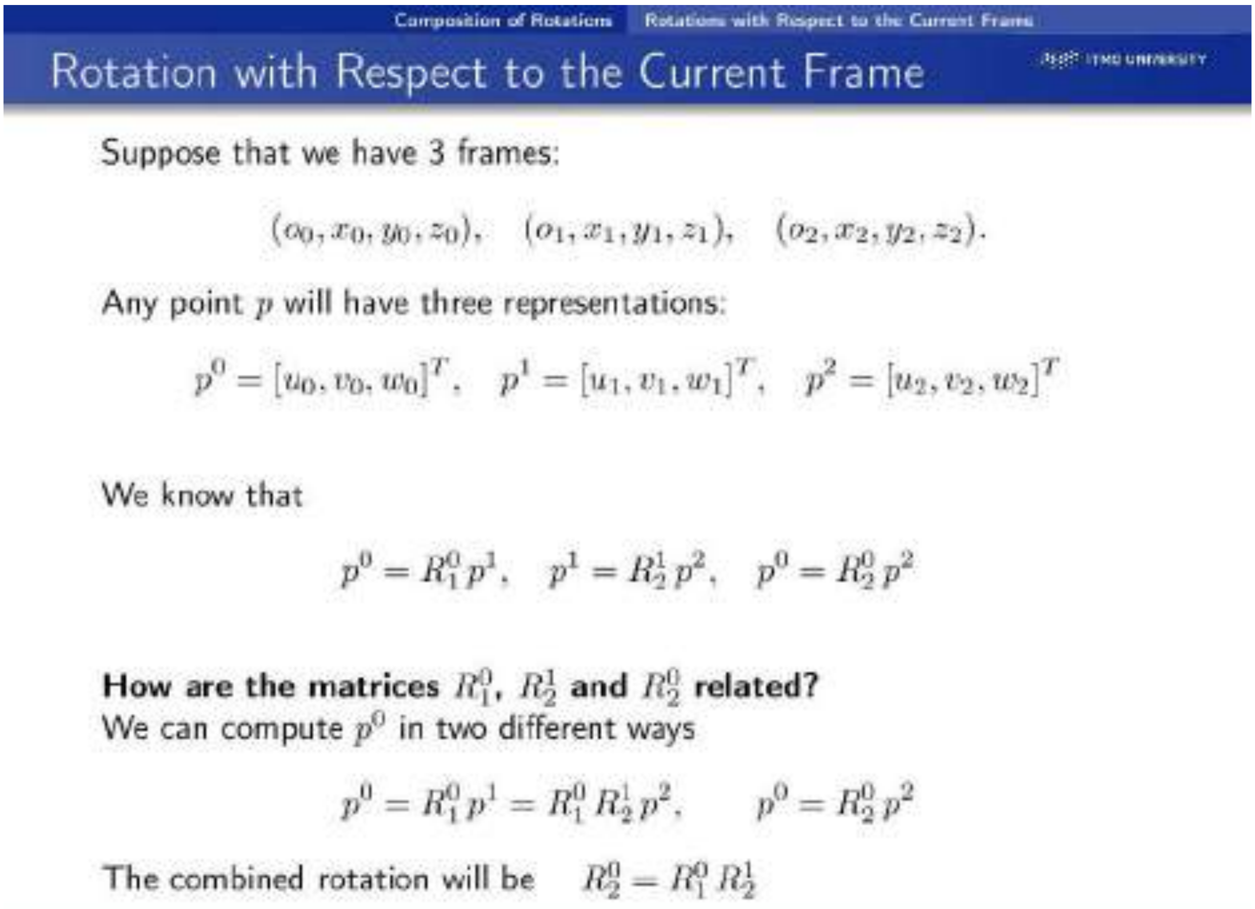
For instance,

$$x_1^0 = \begin{bmatrix} x_1 \cdot x_0 \\ x_1 \cdot y_0 \\ x_1 \cdot z_0 \end{bmatrix} = \underbrace{\left[\begin{array}{c|c|c} x_1^0 \cdot x_0 & y_1^0 \cdot x_0 & z_1^0 \cdot x_0 \\ x_1^0 \cdot y_0 & y_1^0 \cdot y_0 & z_1^0 \cdot y_0 \\ x_1^0 \cdot z_0 & y_1^0 \cdot z_0 & z_1^0 \cdot z_0 \end{array}\right]}_{=R} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

$$x_1^0 = R \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad y_1^0 = R \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad z_1^0 = R \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

$$p^0 = u \cdot x_1^0 + v \cdot y_1^0 + w \cdot z_1^0 = R \begin{bmatrix} u \\ 0 \\ 0 \end{bmatrix} + R \begin{bmatrix} 0 \\ v \\ 0 \end{bmatrix} + R \begin{bmatrix} 0 \\ 0 \\ w \end{bmatrix}$$

$$p^0 = u \cdot x_1^0 + v \cdot y_1^0 + w \cdot z_1^0 = R \begin{bmatrix} u \\ v \\ w \end{bmatrix} = R_1^0 p^1$$



Rotation with Respect to the Current Frame

Suppose that we have 3 frames:

$$(o_0, x_0, y_0, z_0), \quad (o_1, x_1, y_1, z_1), \quad (o_2, x_2, y_2, z_2).$$

Any point p will have three representations:

$$p^0 = [u_0, v_0, w_0]^T, \quad p^1 = [u_1, v_1, w_1]^T, \quad p^2 = [u_2, v_2, w_2]^T$$

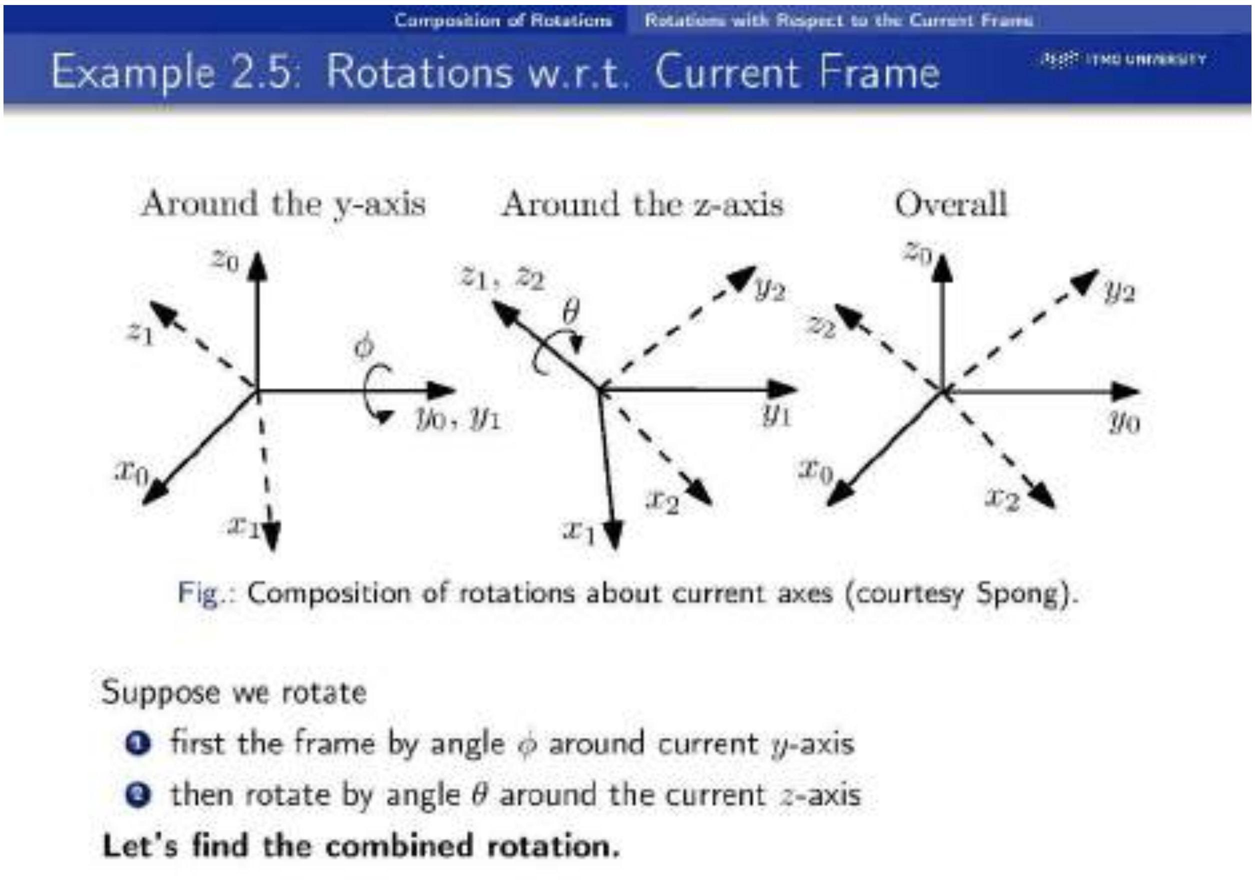
We know that

$$p^0 = R_1^0 p^1, \quad p^1 = R_2^1 p^2, \quad p^0 = R_2^0 p^2$$

How are the matrices R_1^0, R_2^1 and R_2^0 related?
We can compute p^0 in two different ways

$$p^0 = R_1^0 p^1 = R_1^0 R_2^1 p^2, \qquad p^0 = R_2^0 p^2$$

The combined rotation will be $R_2^0 = R_1^0 R_2^1$



Example 2.5: Rotations w.r.t. Current Frame

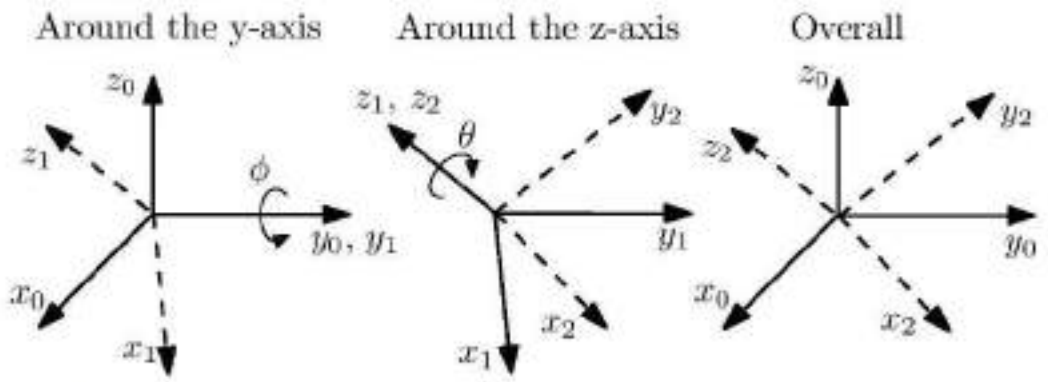
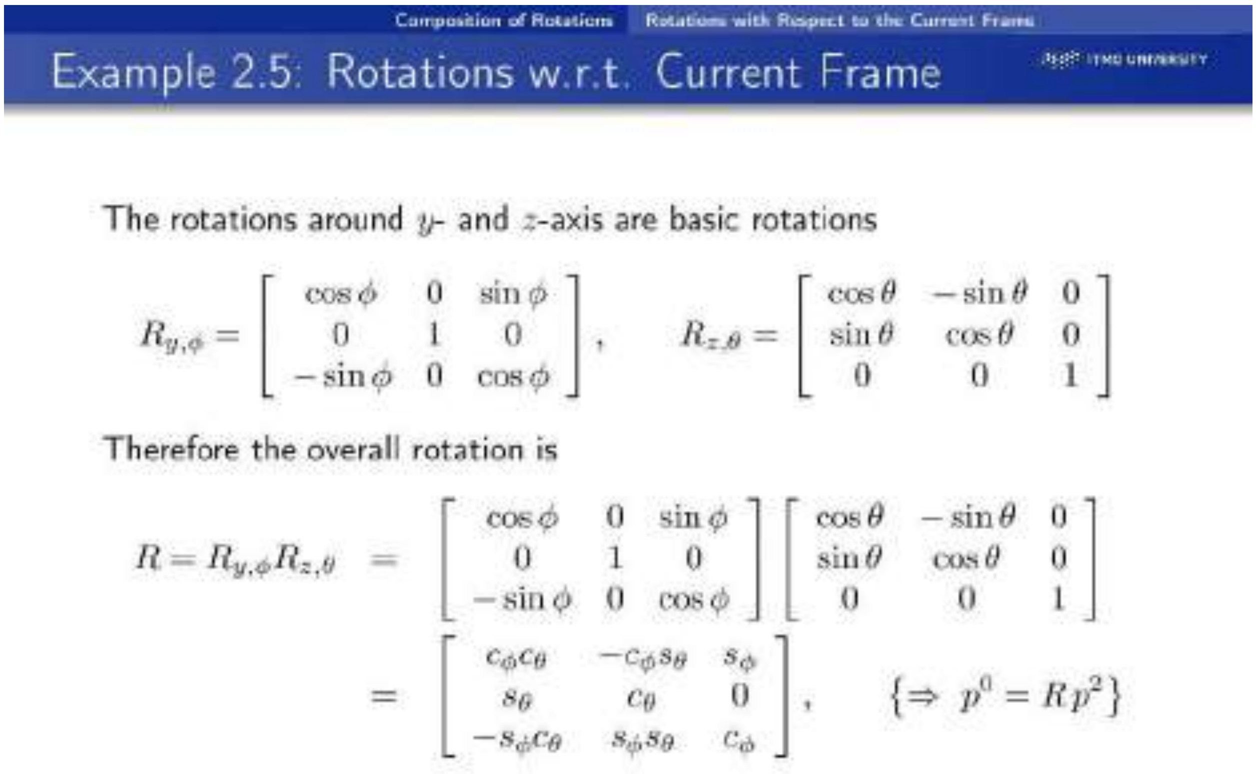


Fig.: Composition of rotations about current axes (courtesy Spong).

Suppose we rotate

1. first the frame by angle ϕ around current y-axis
2. then rotate by angle θ around the current z-axis

Let's find the combined rotation.



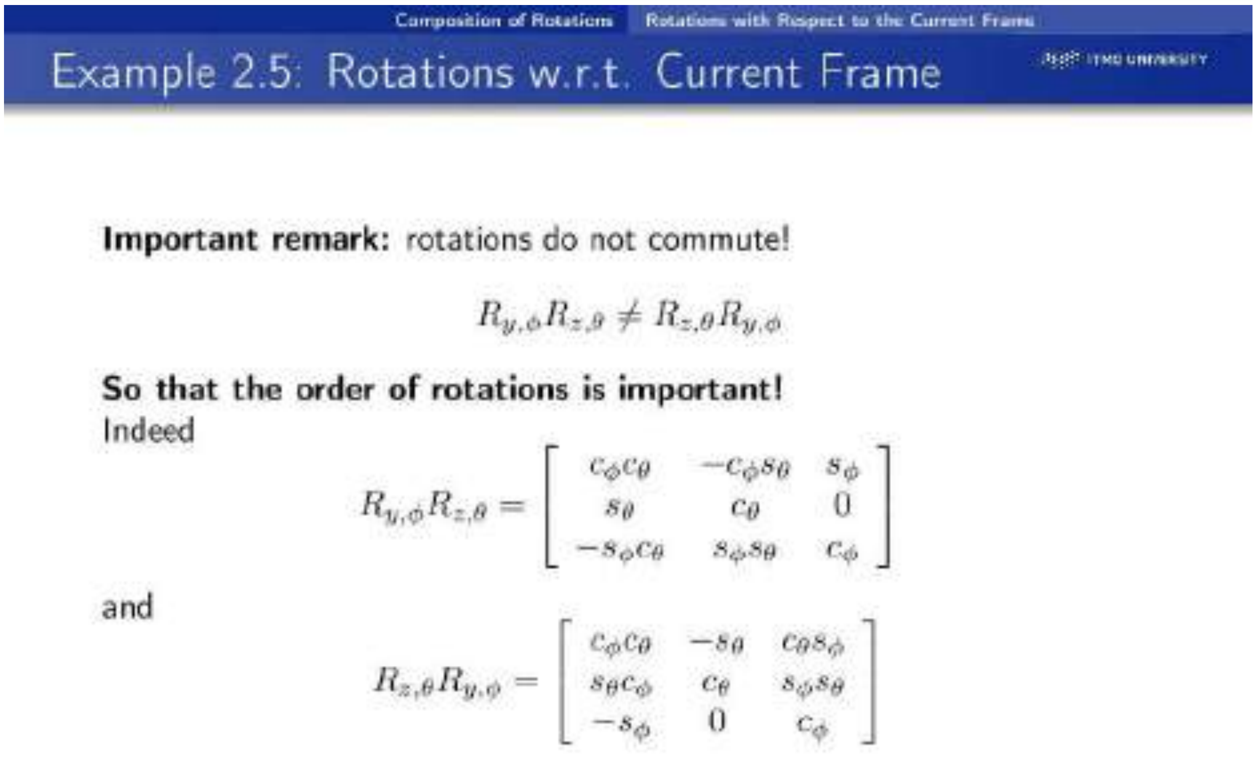
Example 2.5: Rotations w.r.t. Current Frame

The rotations around y- and z-axis are basic rotations

$$R_{y,\phi} = \begin{bmatrix} \cos\phi & 0 & \sin\phi \\ 0 & 1 & 0 \\ -\sin\phi & 0 & \cos\phi \end{bmatrix}, \qquad R_{z,\theta} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Therefore the overall rotation is

$$\begin{aligned} R = R_{y,\phi}R_{z,\theta} &= \begin{bmatrix} \cos\phi & 0 & \sin\phi \\ 0 & 1 & 0 \\ -\sin\phi & 0 & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} c_\phi c_\theta & -c_\phi s_\theta & s_\phi \\ s_\theta & c_\theta & 0 \\ -s_\phi c_\theta & s_\phi s_\theta & c_\phi \end{bmatrix}, \qquad \{\Rightarrow\ p^0 = Rp^2\} \end{aligned}$$



Example 2.5: Rotations w.r.t. Current Frame

Important remark: rotations do not commute!

$$R_{y,\phi}R_{z,\theta} \neq R_{z,\theta}R_{y,\phi}$$

So that the order of rotations is important!
Indeed

$$R_{y,\phi}R_{z,\theta} = \begin{bmatrix} c_\phi c_\theta & -c_\phi s_\theta & s_\phi \\ s_\theta & c_\theta & 0 \\ -s_\phi c_\theta & s_\phi s_\theta & c_\phi \end{bmatrix}$$

and

$$R_{z,\theta}R_{y,\phi} = \begin{bmatrix} c_\phi c_\theta & -s_\theta & c_\theta s_\phi \\ s_\theta c_\phi & c_\theta & s_\phi s_\theta \\ -s_\phi & 0 & c_\phi \end{bmatrix}$$

Rotation with Respect to the Fixed Frame

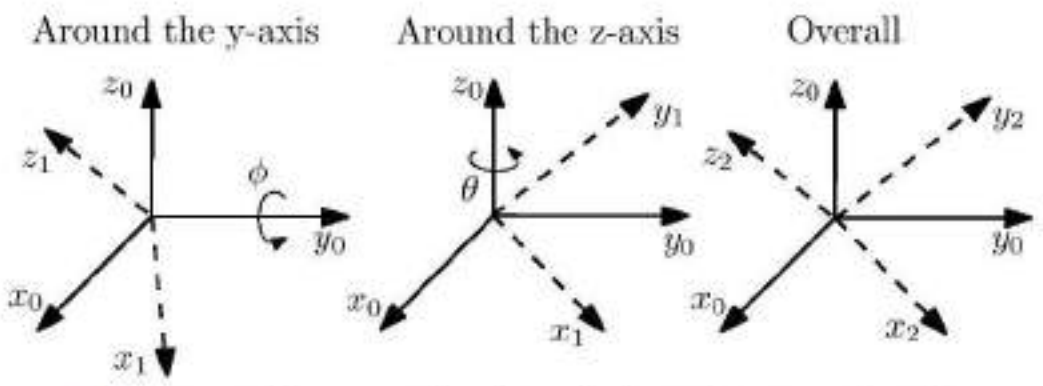


Fig.: Composition of rotations about fixed axes (courtesy Spong).

How to compute the rotation if the basic rotations are done with respect to fixed frames?

Similarity Transformation

Given two frames and the rotation

$$(o_0, x_0, y_0, z_0), \quad (o_1, x_1, y_1, z_1), \quad p^0 = R_1^0 p^1$$

Given a linear transform A, for which we know how it acts on vectors of the 0-frame

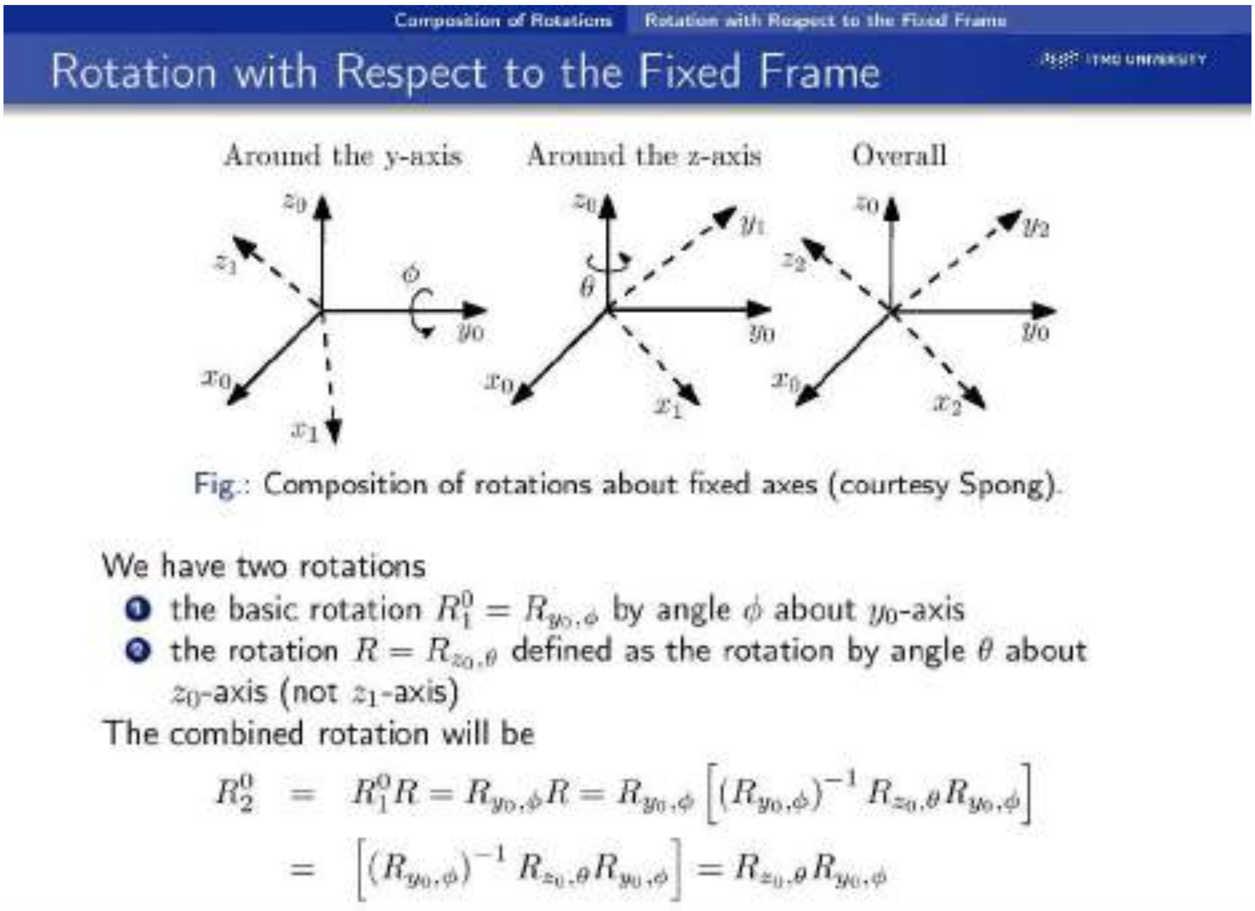
$$\vec{a}^0 = A\vec{b}^0$$

How does it act on the vectors of the 1-frame?
To compute its action, we need to observe that vectors in both frames are in one-to-one correspondence, i.e

- given $\vec{b}^0$, then $\vec{b}^1 = \left[R_1^0\right]^{-1}\vec{b}^0$
- given $\vec{b}^1$, then $\vec{b}^0 = R_1^0\vec{b}^1$

To define A acting in the 1-frame, use its definition in 0-frame

$$\underbrace{\left[R_1^0\right]^{-1}\vec{a}^0}_{\vec{a}^1} = \left[R_1^0\right]^{-1}A\vec{b}^0 = \underbrace{\left[R_1^0\right]^{-1}AR_1^0}_{B}\vec{b}^1$$



Rotation with Respect to the Fixed Frame

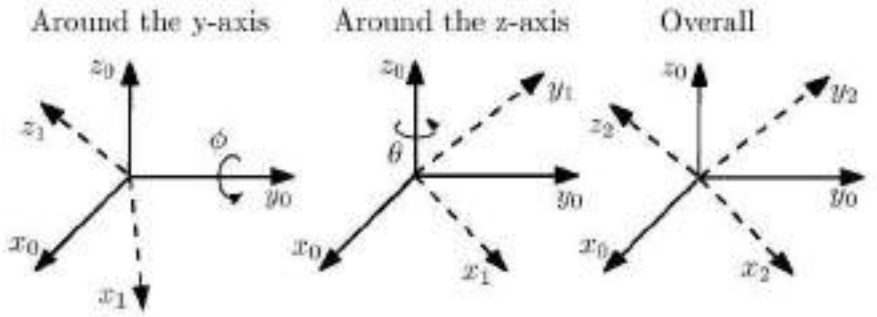


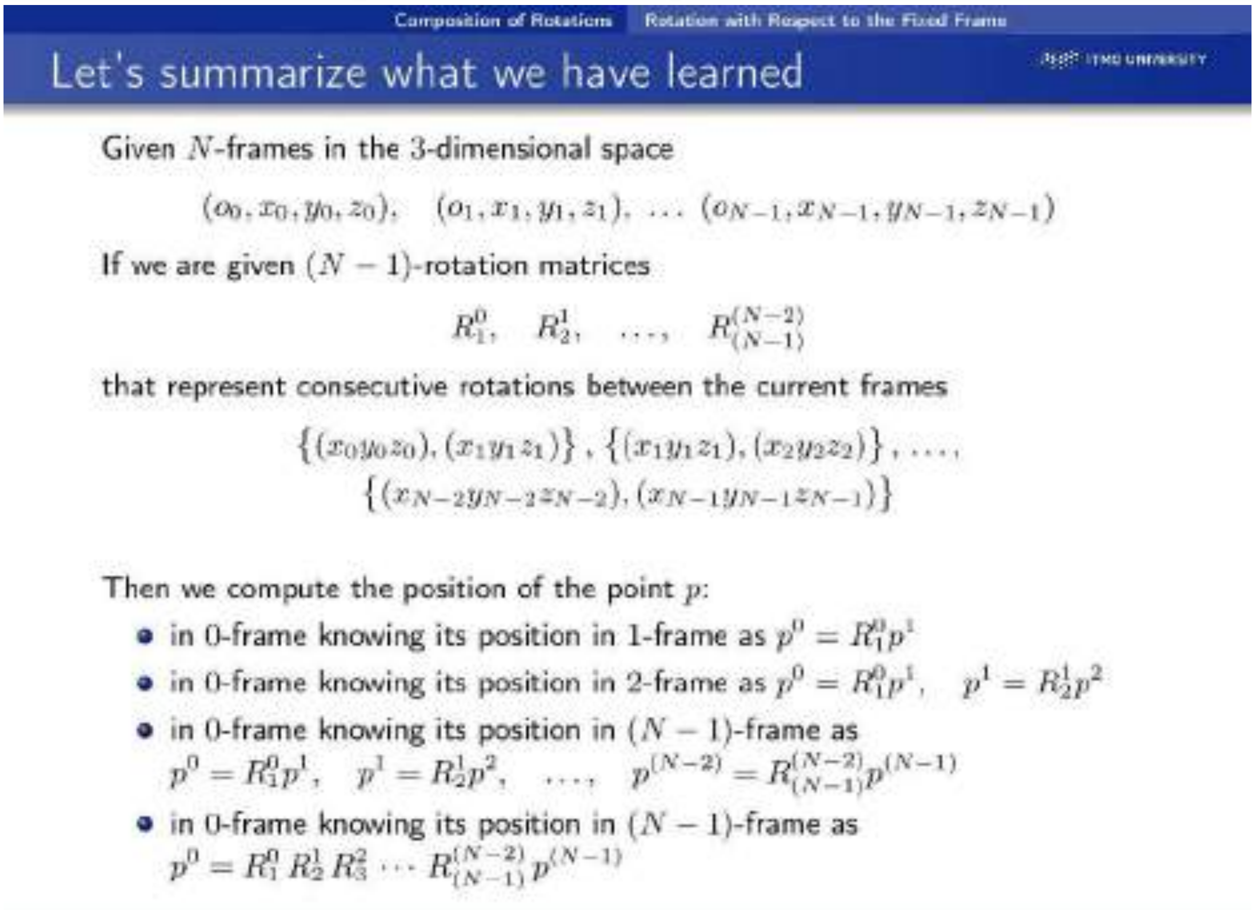
Fig.: Composition of rotations about fixed axes (courtesy Spong).

We have two rotations

1. the basic rotation $R_1^0 = R_{y_0,\phi}$ by angle ϕ about y_0-axis
2. the rotation $R = R_{z_0,\theta}$ defined as the rotation by angle θ about z_0-axis (not z_1-axis)

The combined rotation will be

$$\begin{aligned} R_2^0 &= R_1^0 R = R_{y_0,\phi} R = R_{y_0,\phi}\left[(R_{y_0,\phi})^{-1} R_{z_0,\theta} R_{y_0,\phi}\right] \\ &= \left[(R_{y_0,\phi})^{-1} R_{z_0,\theta} R_{y_0,\phi}\right] = R_{z_0,\theta} R_{y_0,\phi} \end{aligned}$$



Let's summarize what we have learned

Given N-frames in the 3-dimensional space

$$(o_0, x_0, y_0, z_0), \quad (o_1, x_1, y_1, z_1), \ \ldots \ (o_{N-1}, x_{N-1}, y_{N-1}, z_{N-1})$$

If we are given $(N-1)$-rotation matrices

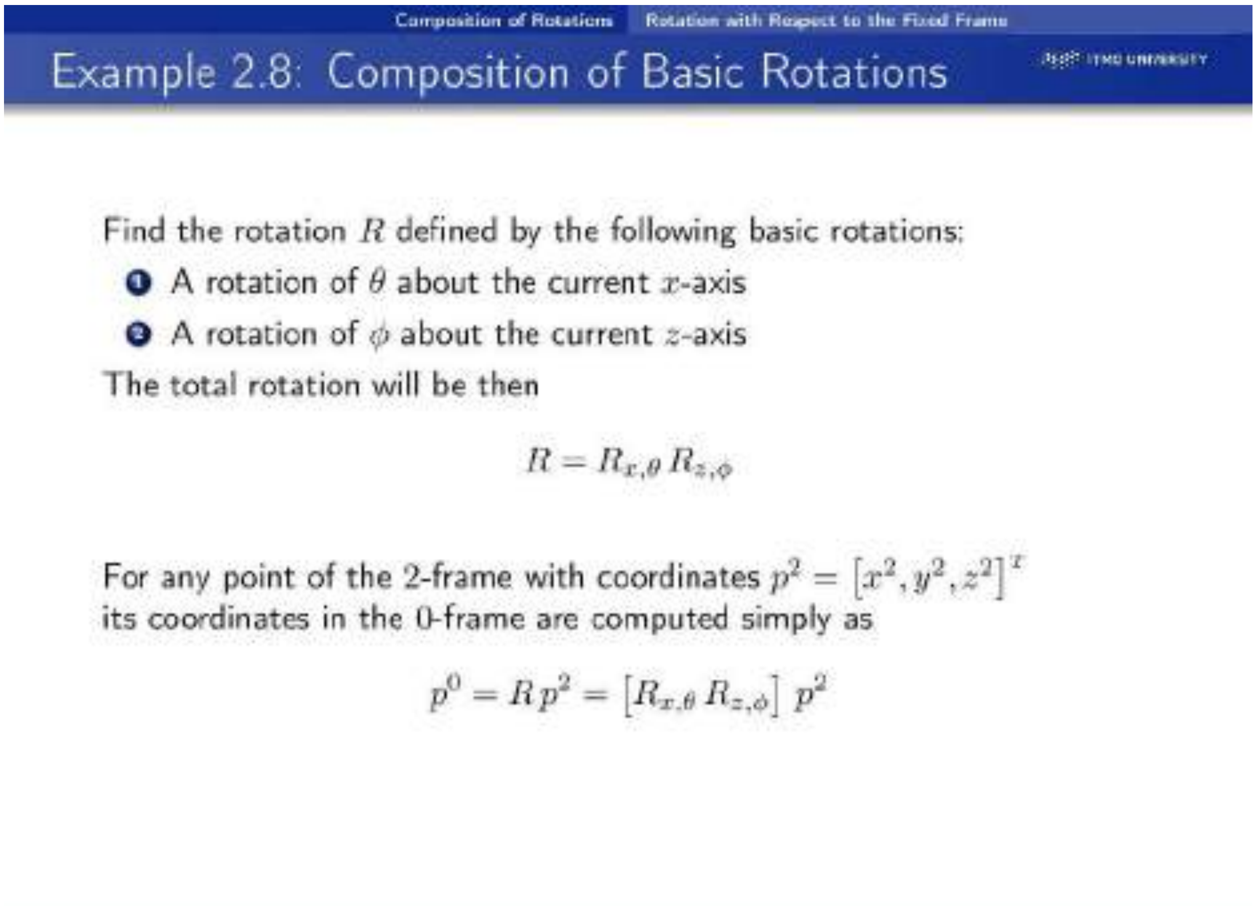
$$R_1^0, \quad R_2^1, \quad \ldots, \quad R_{(N-1)}^{(N-2)}$$

that represent consecutive rotations between the current frames

$$\left\{(x_0 y_0 z_0), (x_1 y_1 z_1)\right\}, \left\{(x_1 y_1 z_1), (x_2 y_2 z_2)\right\}, \ldots, \\ \left\{(x_{N-2} y_{N-2} z_{N-2}), (x_{N-1} y_{N-1} z_{N-1})\right\}$$

Then we compute the position of the point p:

- in 0-frame knowing its position in 1-frame as $p^0 = R_1^0 p^1$
- in 0-frame knowing its position in 2-frame as $p^0 = R_1^0 p^1, \quad p^1 = R_2^1 p^2$
- in 0-frame knowing its position in $(N-1)$-frame as $p^0 = R_1^0 p^1, \quad p^1 = R_2^1 p^2, \quad \ldots, \quad p^{(N-2)} = R_{(N-1)}^{(N-2)} p^{(N-1)}$
- in 0-frame knowing its position in $(N-1)$-frame as $p^0 = R_1^0 R_2^1 R_3^2 \cdots R_{(N-1)}^{(N-2)} p^{(N-1)}$


Composition of Rotations | Rotation with Respect to the Fixed Frame

Example 2.8: Composition of Basic Rotations

Find the rotation R defined by the following basic rotations:

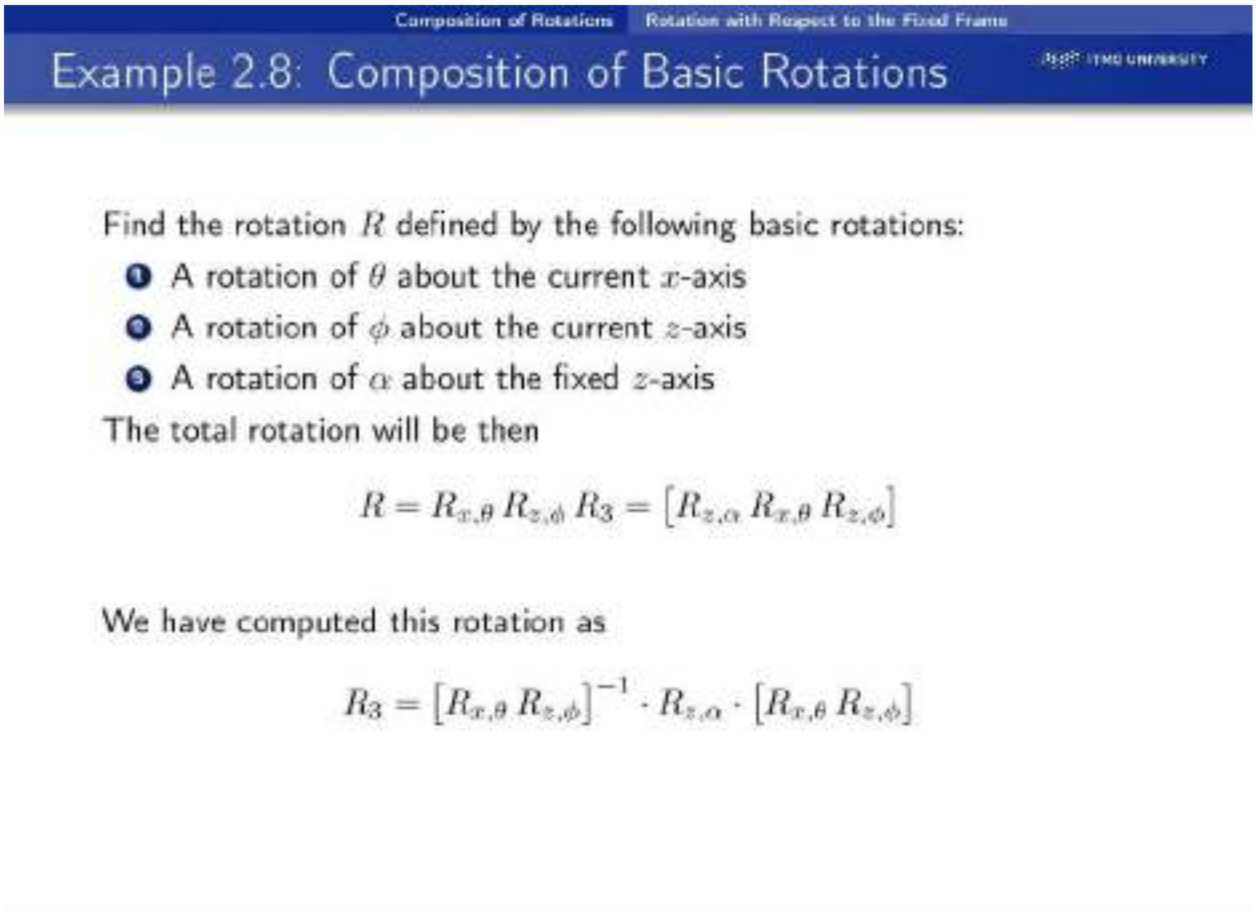
1. A rotation of θ about the current x-axis
2. A rotation of ϕ about the current z-axis

The total rotation will be then

$$R = R_{x,\theta}\, R_{z,\phi}$$

For any point of the 2-frame with coordinates $p^2 = \left[x^2, y^2, z^2\right]^T$ its coordinates in the 0-frame are computed simply as

$$p^0 = R\, p^2 = \left[R_{x,\theta}\, R_{z,\phi}\right]\, p^2$$


Composition of Rotations | Rotation with Respect to the Fixed Frame

Example 2.8: Composition of Basic Rotations

Find the rotation R defined by the following basic rotations:

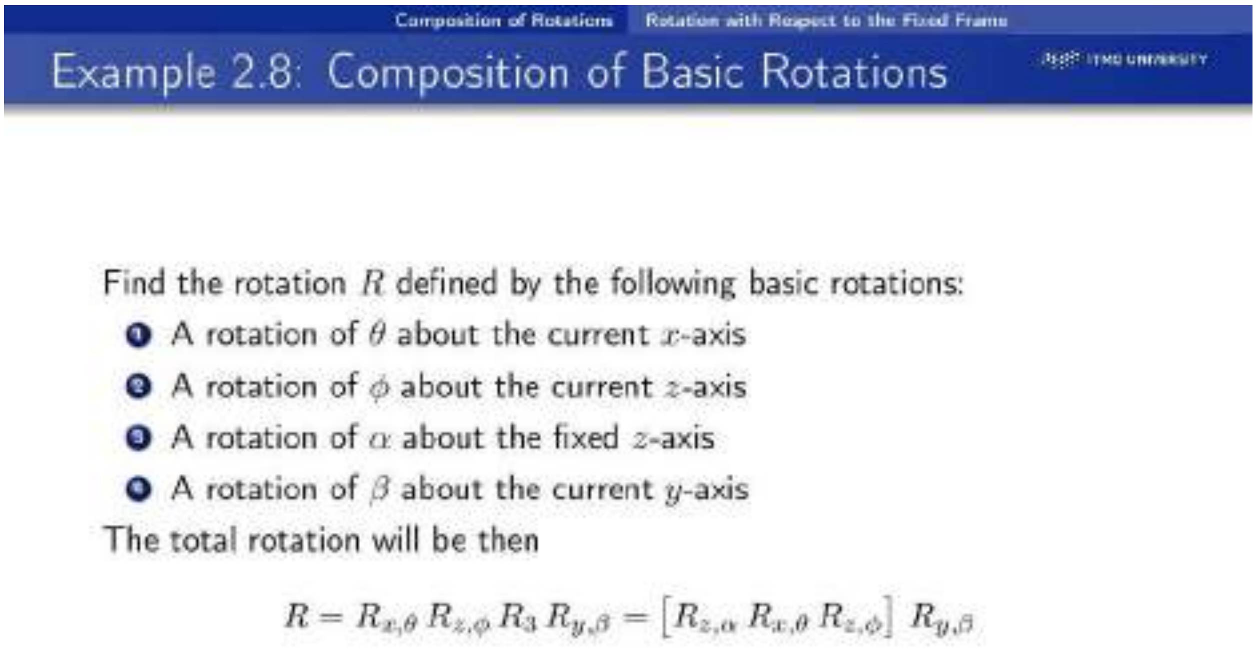
1. A rotation of θ about the current x-axis
2. A rotation of ϕ about the current z-axis
3. A rotation of α about the fixed z-axis

The total rotation will be then

$$R = R_{x,\theta}\, R_{z,\phi}\, R_3 = \left[R_{z,\alpha}\, R_{x,\theta}\, R_{z,\phi}\right]$$

We have computed this rotation as

$$R_3 = \left[R_{x,\theta}\, R_{z,\phi}\right]^{-1} \cdot R_{z,\alpha} \cdot \left[R_{x,\theta}\, R_{z,\phi}\right]$$



Composition of Rotations | Rotation with Respect to the Fixed Frame

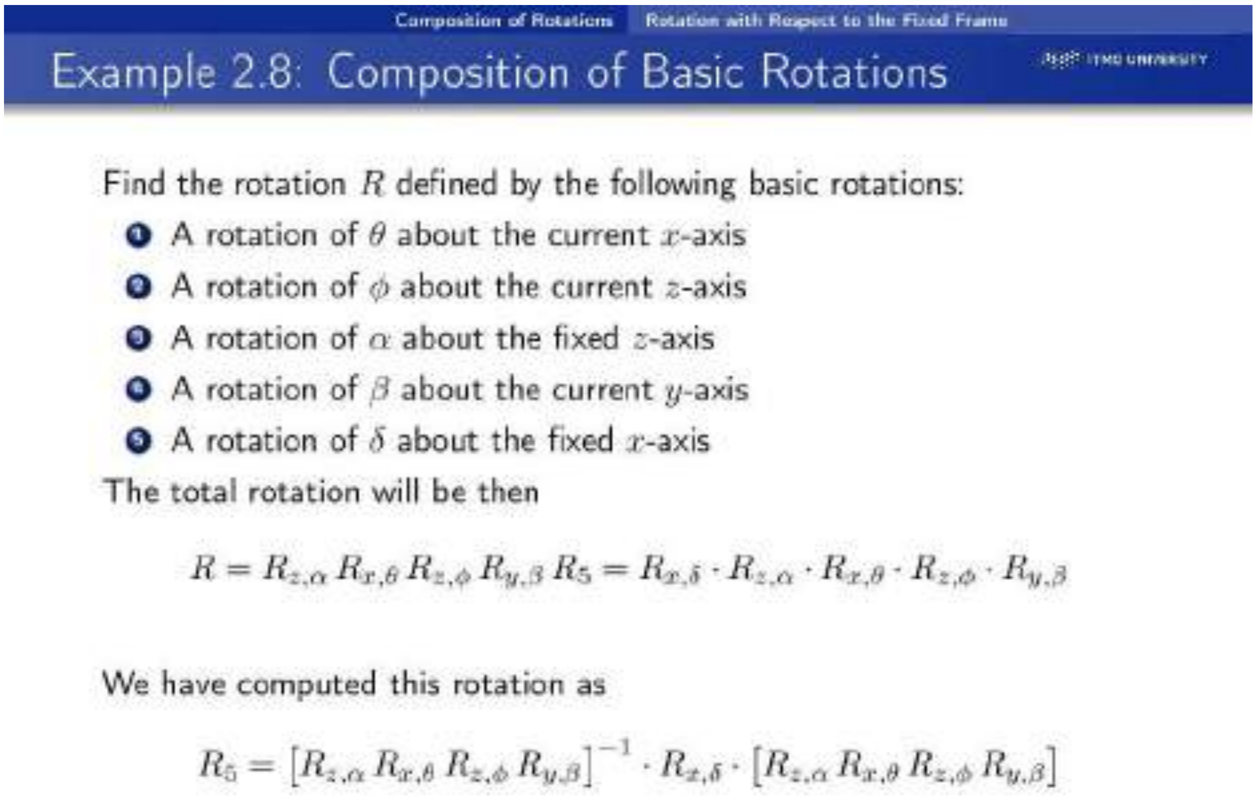
Example 2.8: Composition of Basic Rotations

Find the rotation R defined by the following basic rotations:

1. A rotation of θ about the current x-axis
2. A rotation of ϕ about the current z-axis
3. A rotation of α about the fixed z-axis
4. A rotation of β about the current y-axis

The total rotation will be then

$$R = R_{x,\theta}\, R_{z,\phi}\, R_3\, R_{y,\beta} = \left[R_{z,\alpha}\, R_{x,\theta}\, R_{z,\phi}\right]\, R_{y,\beta}$$



Composition of Rotations | Rotation with Respect to the Fixed Frame

Example 2.8: Composition of Basic Rotations

Find the rotation R defined by the following basic rotations:

1. A rotation of θ about the current x-axis
2. A rotation of ϕ about the current z-axis
3. A rotation of α about the fixed z-axis
4. A rotation of β about the current y-axis
5. A rotation of δ about the fixed x-axis

The total rotation will be then

$$R = R_{z,\alpha}\, R_{x,\theta}\, R_{z,\phi}\, R_{y,\beta}\, R_5 = R_{x,\delta} \cdot R_{z,\alpha} \cdot R_{x,\theta} \cdot R_{z,\phi} \cdot R_{y,\beta}$$

We have computed this rotation as

$$R_5 = \left[R_{z,\alpha}\, R_{x,\theta}\, R_{z,\phi}\, R_{y,\beta}\right]^{-1} \cdot R_{x,\delta} \cdot \left[R_{z,\alpha}\, R_{x,\theta}\, R_{z,\phi}\, R_{y,\beta}\right]$$

Parameterizations of Rotations

A rigid body has at most 3 rotational degrees of freedom.

Any rotation matrix R

- has dimension 3×3, i.e. it has 9 quantities
- belongs to $\mathcal{SO}(3)$, i.e. we have the following constraints:
  - its 3 columns are unit vectors (3 equations)
  - its 3 columns are orthogonal to each other (3 equations)

We have 6 independent equations and 9 unknowns that describe arbitrary rotations in 3D.

There are 3 free variables to be assigned!

Euler Angles

Around the z-axis (ϕ) Around the y-axis Around the z-axis (ψ)

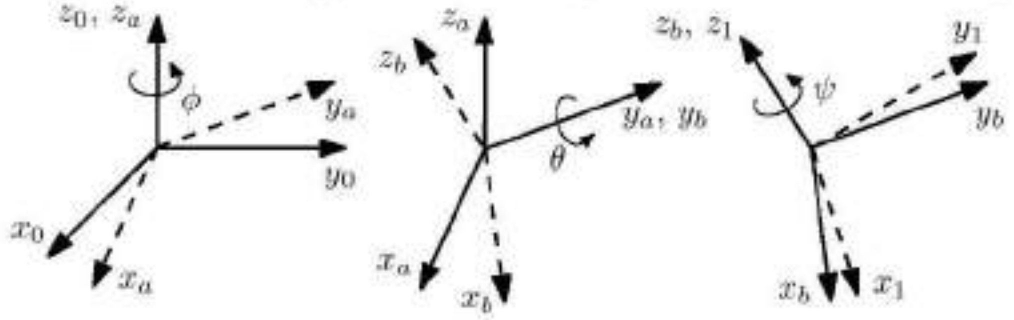
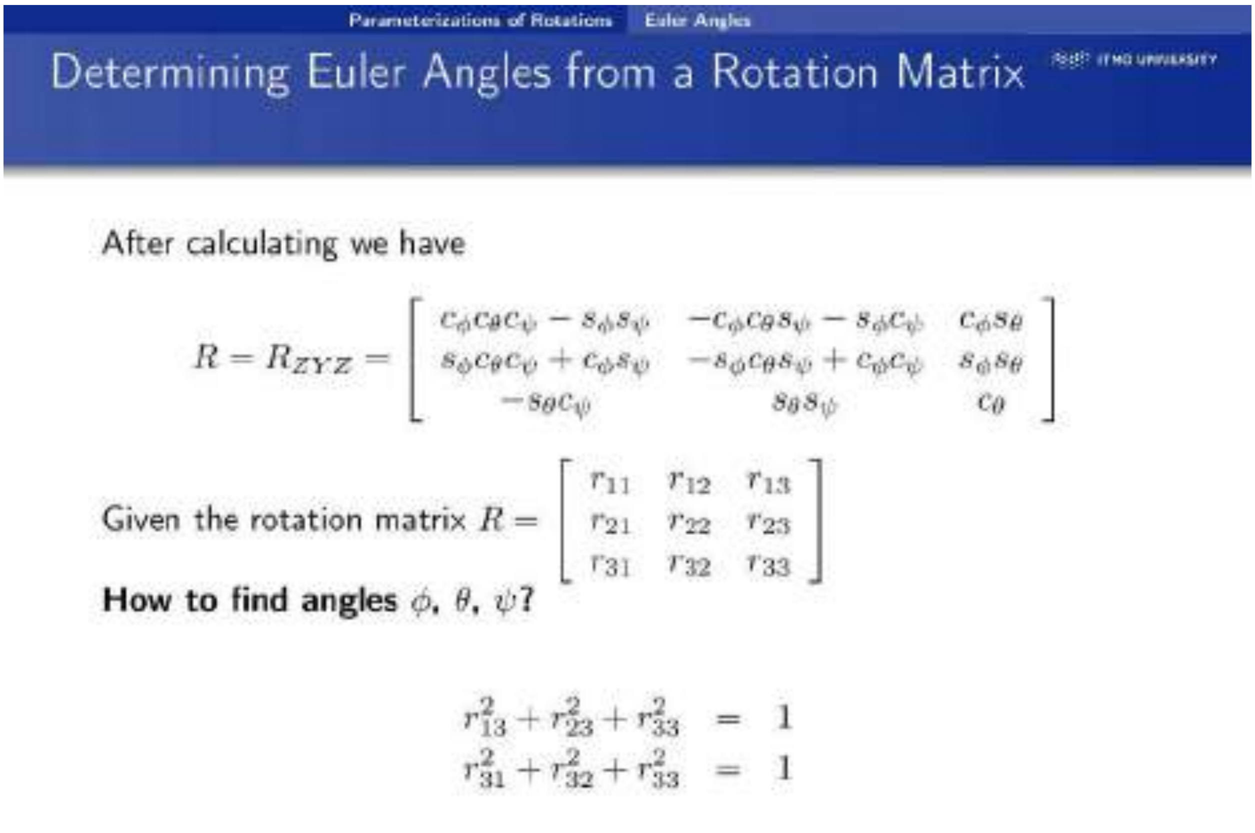


Fig.: Euler angle representation by 3 rotations about current axes (courtesy Spong).

$$
\begin{aligned}
R_{ZYZ} &= R_{z,\phi} \cdot R_{y,\theta} \cdot R_{z,\psi} \\
&= \begin{bmatrix} c_\phi & -s_\phi & 0 \\ s_\phi & c_\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix} \cdot \begin{bmatrix} c_\psi & -s_\psi & 0 \\ s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}
\end{aligned}
$$



Determining Euler Angles from a Rotation Matrix

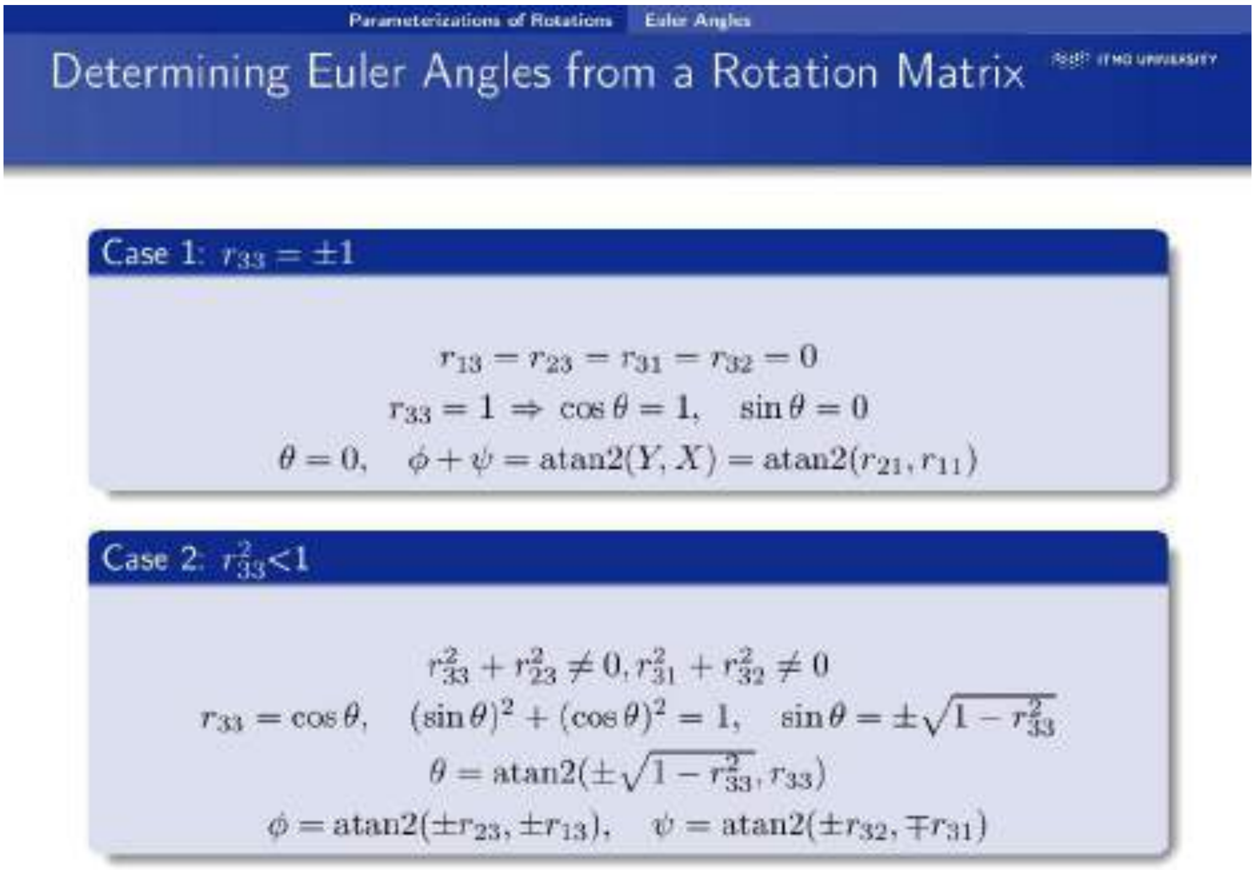
After calculating we have

$$R = R_{ZYZ} = \begin{bmatrix} c_\phi c_\theta c_\psi - s_\phi s_\psi & -c_\phi c_\theta s_\psi - s_\phi c_\psi & c_\phi s_\theta \\ s_\phi c_\theta c_\psi + c_\phi s_\psi & -s_\phi c_\theta s_\psi + c_\phi c_\psi & s_\phi s_\theta \\ -s_\theta c_\psi & s_\theta s_\psi & c_\theta \end{bmatrix}$$

Given the rotation matrix $R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$

How to find angles ϕ, θ, ψ?

$$\begin{aligned} r_{13}^2 + r_{23}^2 + r_{33}^2 &= 1 \\ r_{31}^2 + r_{32}^2 + r_{33}^2 &= 1 \end{aligned}$$



Determining Euler Angles from a Rotation Matrix

Case 1: $r_{33} = \pm 1$

$$r_{13} = r_{23} = r_{31} = r_{32} = 0$$
$$r_{33} = 1 \Rightarrow \cos\theta = 1, \quad \sin\theta = 0$$
$$\theta = 0, \quad \phi + \psi = \text{atan2}(Y, X) = \text{atan2}(r_{21}, r_{11})$$

Case 2: $r_{33}^2 < 1$

$$r_{33}^2 + r_{23}^2 \neq 0, r_{31}^2 + r_{32}^2 \neq 0$$
$$r_{33} = \cos\theta, \quad (\sin\theta)^2 + (\cos\theta)^2 = 1, \quad \sin\theta = \pm\sqrt{1 - r_{33}^2}$$
$$\theta = \text{atan2}(\pm\sqrt{1 - r_{33}^2}, r_{33})$$
$$\phi = \text{atan2}(\pm r_{23}, \pm r_{13}), \quad \psi = \text{atan2}(\pm r_{32}, \mp r_{31})$$

Roll, Pitch and Yaw Angles

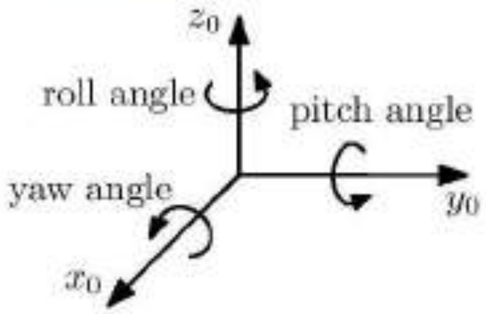


Fig.: Yaw, pitch and roll angles of 3 rotations about the fixed axes x, y and z.

$$\begin{aligned}
R_{XYZ} &= R_{Z,\phi} \cdot R_{Y,\theta} \cdot R_{X,\psi} \\
&= \begin{bmatrix} c_\phi & -s_\phi & 0 \\ s_\phi & c_\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\psi & -s_\psi \\ 0 & s_\psi & c_\psi \end{bmatrix} \\
&= \begin{bmatrix} c_\phi c_\theta & -s_\phi c_\psi + c_\phi s_\theta s_\psi & s_\phi s_\psi + c_\phi s_\theta c_\psi \\ s_\phi c_\theta & c_\phi c_\psi + s_\phi s_\theta s_\psi & -c_\phi s_\psi + s_\phi s_\theta c_\psi \\ -s_\theta & c_\theta s_\psi & c_\theta c_\psi \end{bmatrix}
\end{aligned}$$

Axis/Angle Representation for a Rotation Matrix

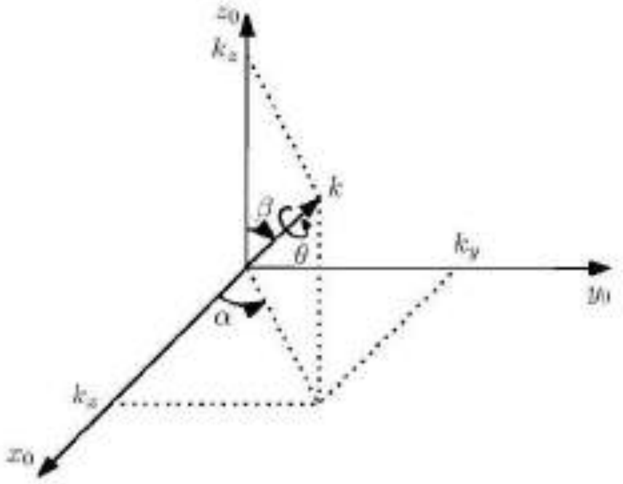
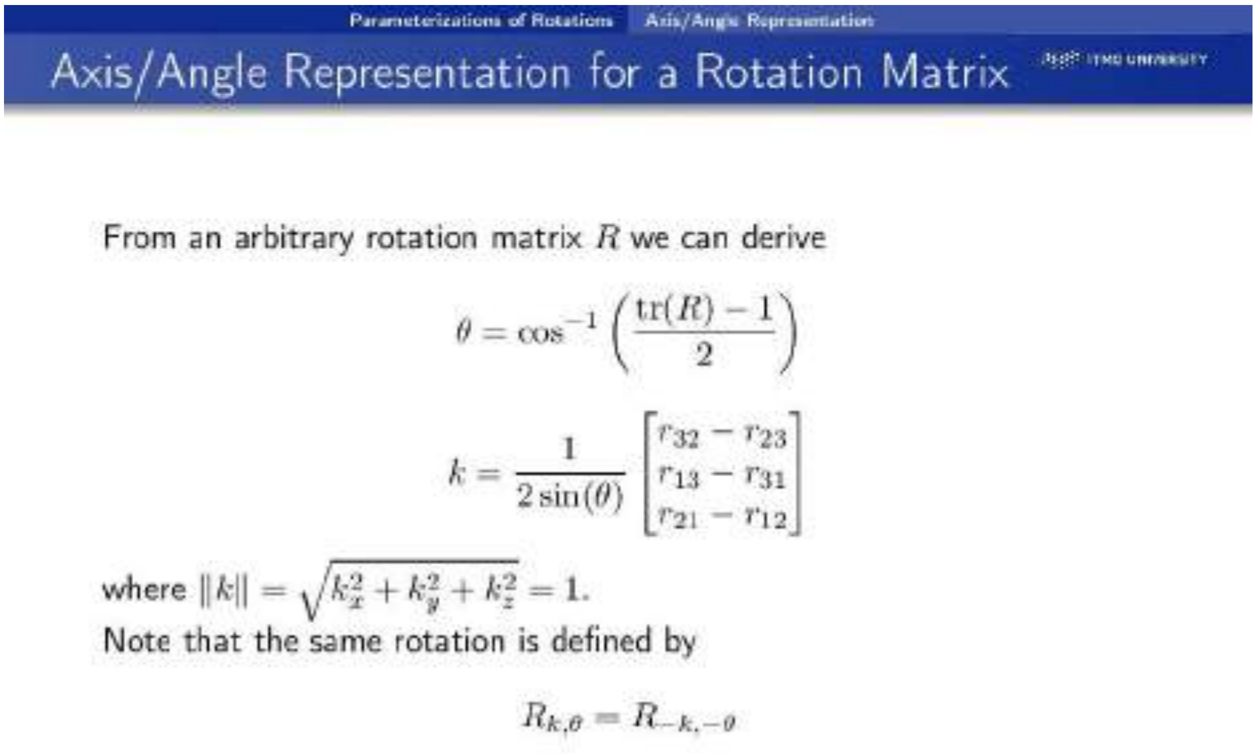


Fig.: Any rotation matrix can be expressed by a single rotation θ about a suitable axis $k = [k_x, k_y, k_z]^T$ of unit length (courtesy Spong).

$$R_{k,\theta} = R \cdot R_{z,\theta} \cdot R^{-1}, \qquad R = R_{z,\alpha} \cdot R_{y,\beta}$$



Parameterizations of Rotations | Axis/Angle Representation

Axis/Angle Representation for a Rotation Matrix

ITMO UNIVERSITY

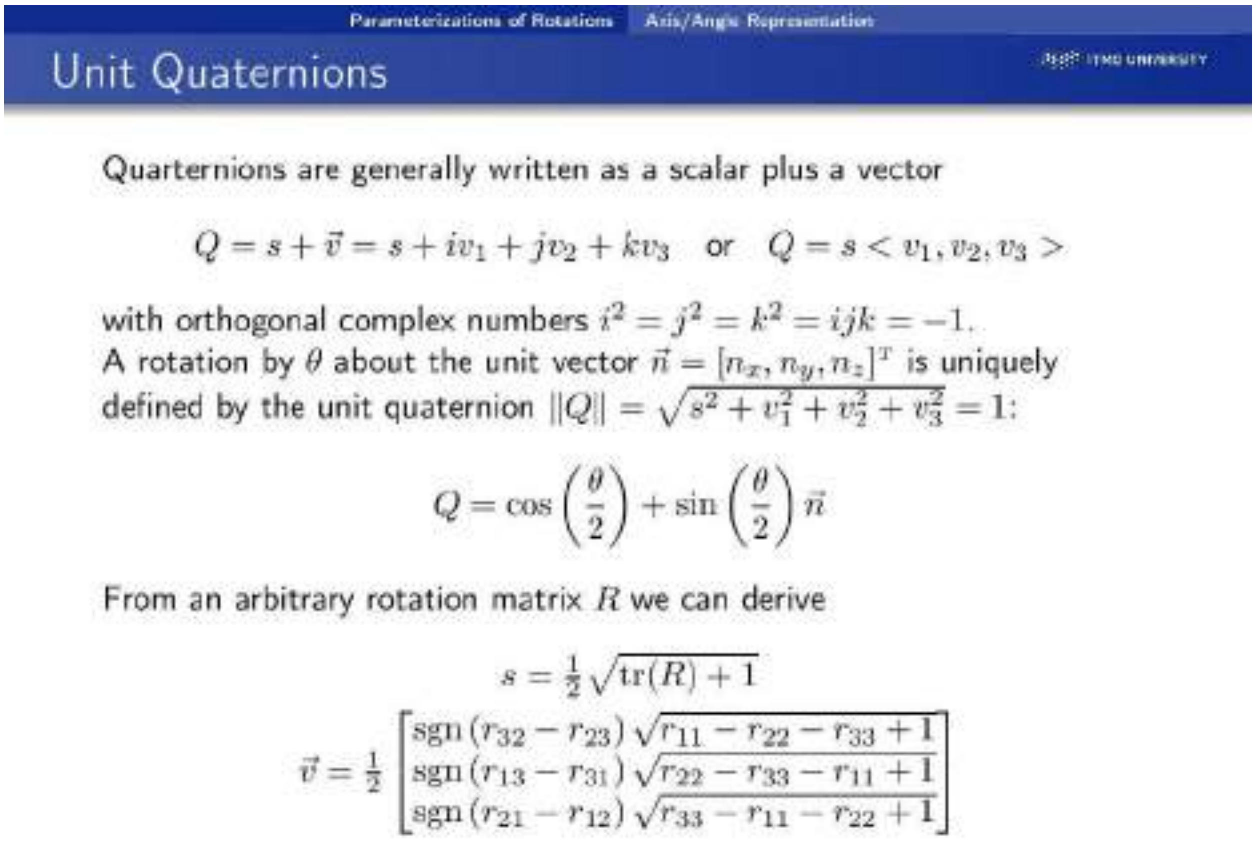
From an arbitrary rotation matrix R we can derive

$$\theta = \cos^{-1}\left(\frac{\mathrm{tr}(R)-1}{2}\right)$$

$$k = \frac{1}{2\sin(\theta)}\begin{bmatrix} r_{32}-r_{23} \\ r_{13}-r_{31} \\ r_{21}-r_{12} \end{bmatrix}$$

where $\|k\| = \sqrt{k_x^2+k_y^2+k_z^2} = 1$.
Note that the same rotation is defined by

$$R_{k,\theta} = R_{-k,-\theta}$$



Parameterizations of Rotations | Axis/Angle Representation

Unit Quaternions

ITMO UNIVERSITY

Quarternions are generally written as a scalar plus a vector

$$Q = s + \vec{v} = s + iv_1 + jv_2 + kv_3 \quad \text{or} \quad Q = s < v_1, v_2, v_3 >$$

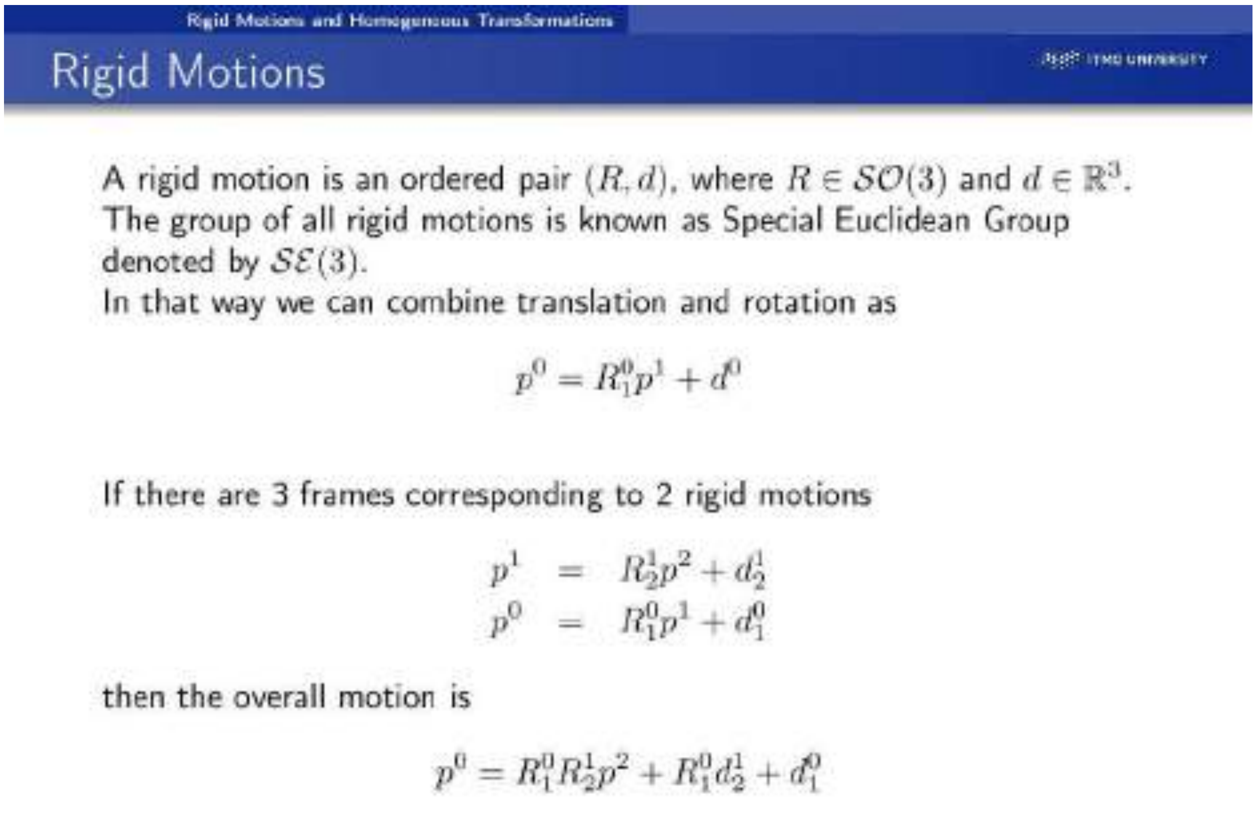
with orthogonal complex numbers $i^2 = j^2 = k^2 = ijk = -1$.
A rotation by θ about the unit vector $\vec{n} = [n_x, n_y, n_z]^T$ is uniquely defined by the unit quaternion $\|Q\| = \sqrt{s^2+v_1^2+v_2^2+v_3^2} = 1$:

$$Q = \cos\left(\frac{\theta}{2}\right) + \sin\left(\frac{\theta}{2}\right)\vec{n}$$

From an arbitrary rotation matrix R we can derive

$$s = \tfrac{1}{2}\sqrt{\mathrm{tr}(R)+1}$$

$$\vec{v} = \tfrac{1}{2}\begin{bmatrix} \mathrm{sgn}\,(r_{32}-r_{23})\sqrt{r_{11}-r_{22}-r_{33}+1} \\ \mathrm{sgn}\,(r_{13}-r_{31})\sqrt{r_{22}-r_{33}-r_{11}+1} \\ \mathrm{sgn}\,(r_{21}-r_{12})\sqrt{r_{33}-r_{11}-r_{22}+1} \end{bmatrix}$$



Rigid Motions

A rigid motion is an ordered pair (R, d), where $R \in \mathcal{SO}(3)$ and $d \in \mathbb{R}^3$.
The group of all rigid motions is known as Special Euclidean Group denoted by $\mathcal{SE}(3)$.
In that way we can combine translation and rotation as

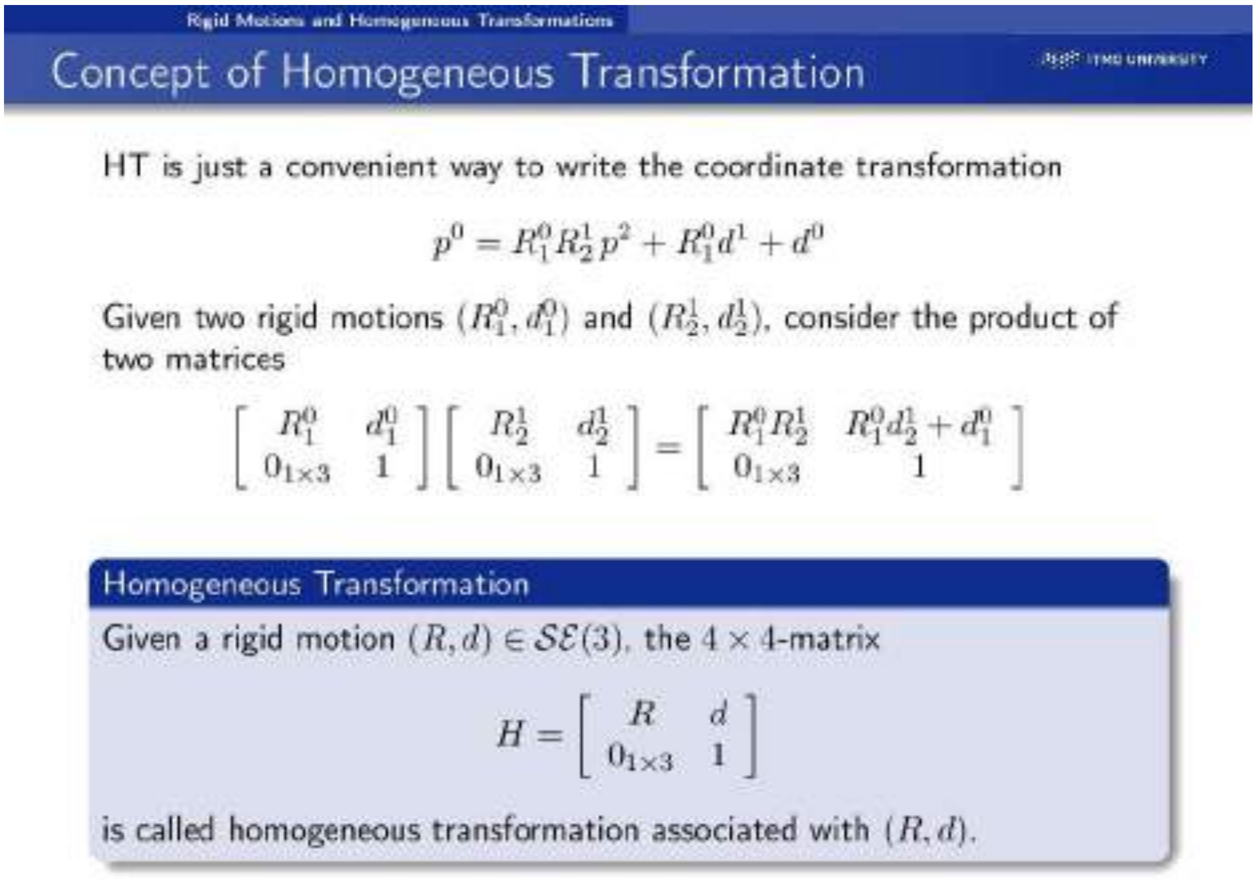
$$p^0 = R_1^0 p^1 + d^0$$

If there are 3 frames corresponding to 2 rigid motions

$$\begin{aligned} p^1 &= R_2^1 p^2 + d_2^1 \\ p^0 &= R_1^0 p^1 + d_1^0 \end{aligned}$$

then the overall motion is

$$p^0 = R_1^0 R_2^1 p^2 + R_1^0 d_2^1 + d_1^0$$



Concept of Homogeneous Transformation

HT is just a convenient way to write the coordinate transformation

$$p^0 = R_1^0 R_2^1 p^2 + R_1^0 d^1 + d^0$$

Given two rigid motions (R_1^0, d_1^0) and (R_2^1, d_2^1), consider the product of two matrices

$$\begin{bmatrix} R_1^0 & d_1^0 \\ 0_{1\times3} & 1 \end{bmatrix} \begin{bmatrix} R_2^1 & d_2^1 \\ 0_{1\times3} & 1 \end{bmatrix} = \begin{bmatrix} R_1^0 R_2^1 & R_1^0 d_2^1 + d_1^0 \\ 0_{1\times3} & 1 \end{bmatrix}$$

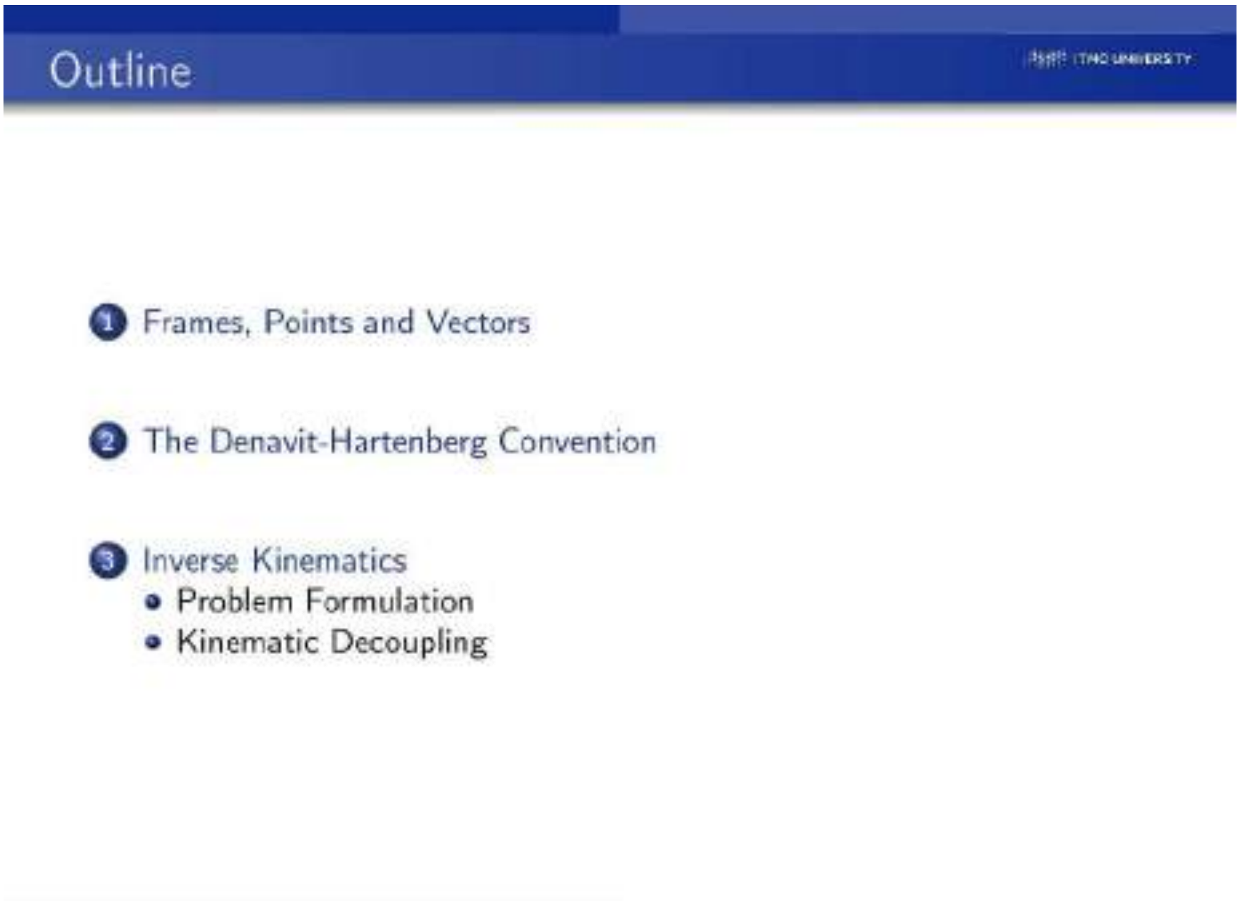
Homogeneous Transformation

Given a rigid motion $(R, d) \in \mathcal{SE}(3)$, the 4×4-matrix

$$H = \begin{bmatrix} R & d \\ 0_{1\times3} & 1 \end{bmatrix}$$

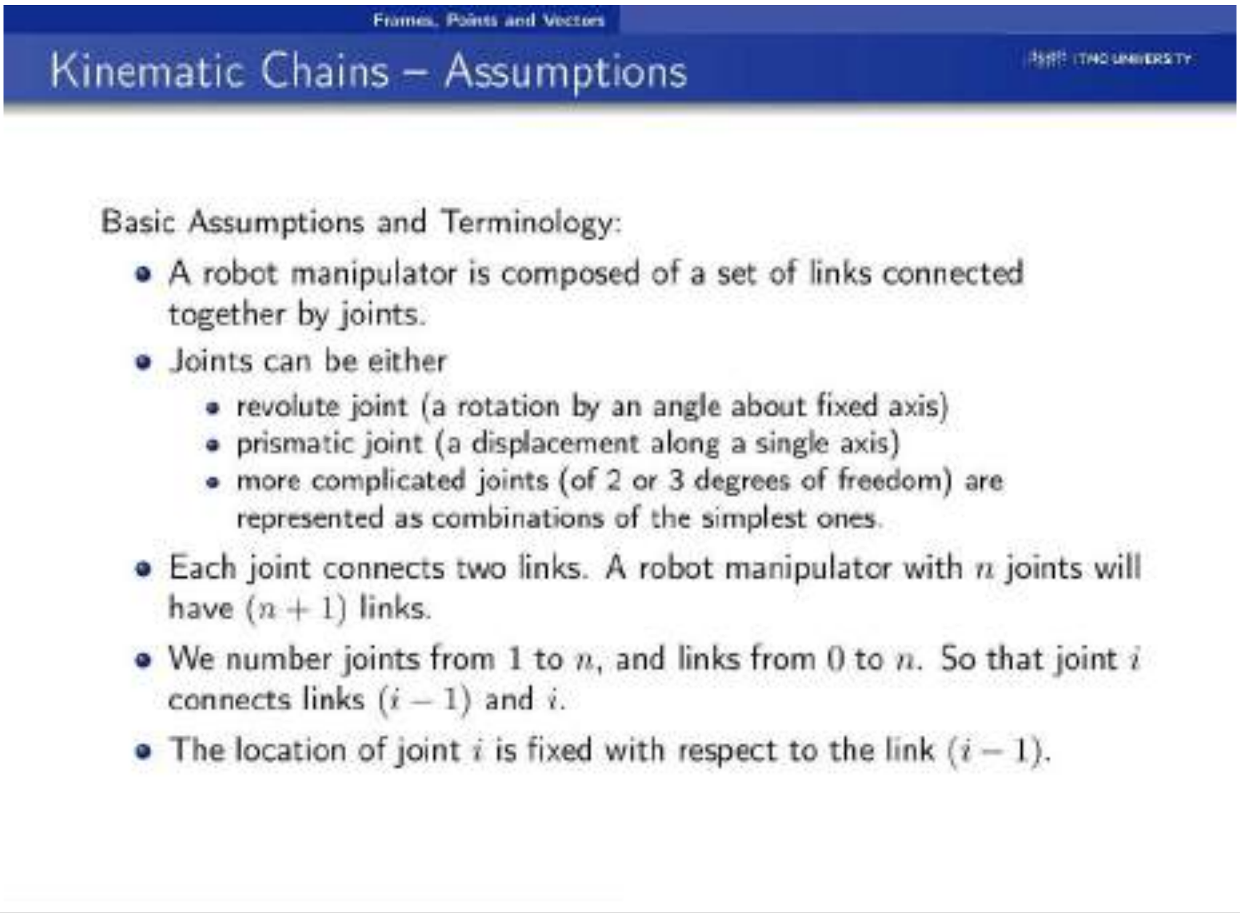
is called homogeneous transformation associated with (R, d).

Kinematics: Forward and Inverse Kinematics



Outline

1. Frames, Points and Vectors
2. The Denavit-Hartenberg Convention
3. Inverse Kinematics
   - Problem Formulation
   - Kinematic Decoupling

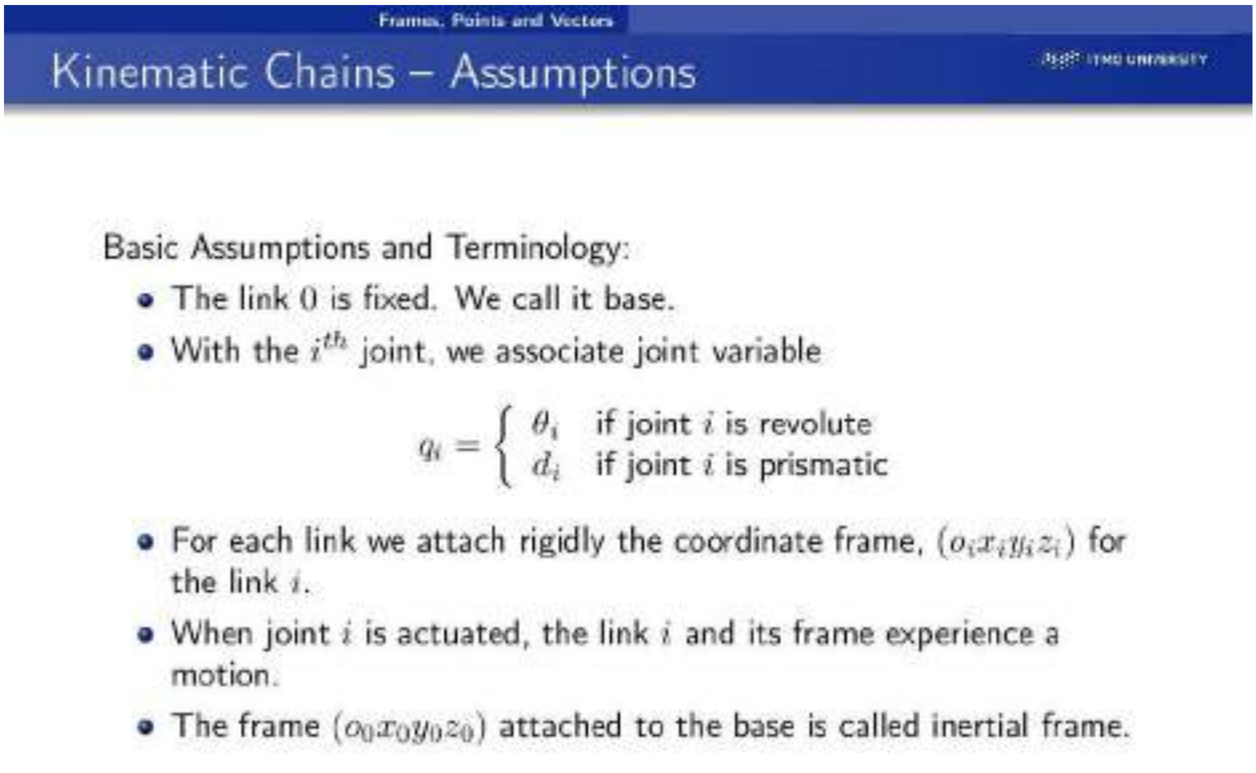


Frames, Points and Vectors

Kinematic Chains – Assumptions

Basic Assumptions and Terminology:

- A robot manipulator is composed of a set of links connected together by joints.
- Joints can be either
  - revolute joint (a rotation by an angle about fixed axis)
  - prismatic joint (a displacement along a single axis)
  - more complicated joints (of 2 or 3 degrees of freedom) are represented as combinations of the simplest ones.
- Each joint connects two links. A robot manipulator with n joints will have $(n+1)$ links.
- We number joints from 1 to n, and links from 0 to n. So that joint i connects links $(i-1)$ and i.
- The location of joint i is fixed with respect to the link $(i-1)$.



Kinematic Chains – Assumptions

Basic Assumptions and Terminology:

- The link 0 is fixed. We call it base.
- With the i^{th} joint, we associate joint variable

$$q_i = \begin{cases} \theta_i & \text{if joint } i \text{ is revolute} \\ d_i & \text{if joint } i \text{ is prismatic} \end{cases}$$

- For each link we attach rigidly the coordinate frame, $(o_i x_i y_i z_i)$ for the link i.
- When joint i is actuated, the link i and its frame experience a motion.
- The frame $(o_0 x_0 y_0 z_0)$ attached to the base is called inertial frame.

Kinematic Chains – Assumptions

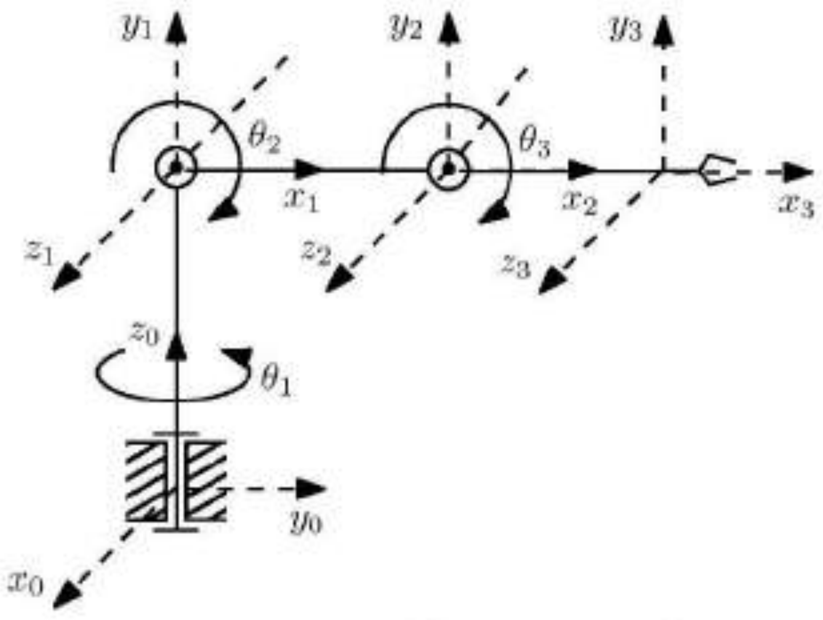
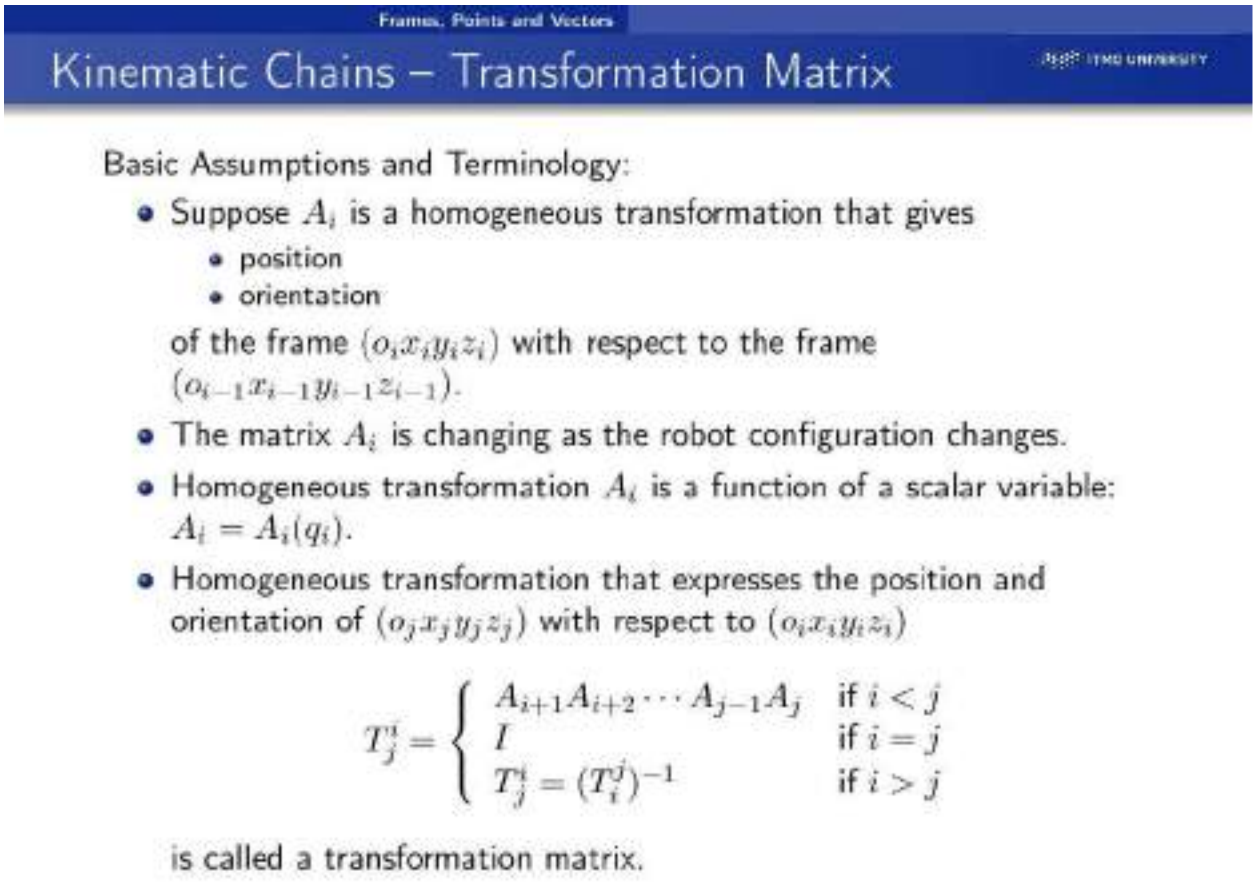


Fig.: Coordinate frames attached to elbow manipulator (courtesy Spong).



Frames, Points and Vectors

Kinematic Chains – Transformation Matrix

ITMO UNIVERSITY

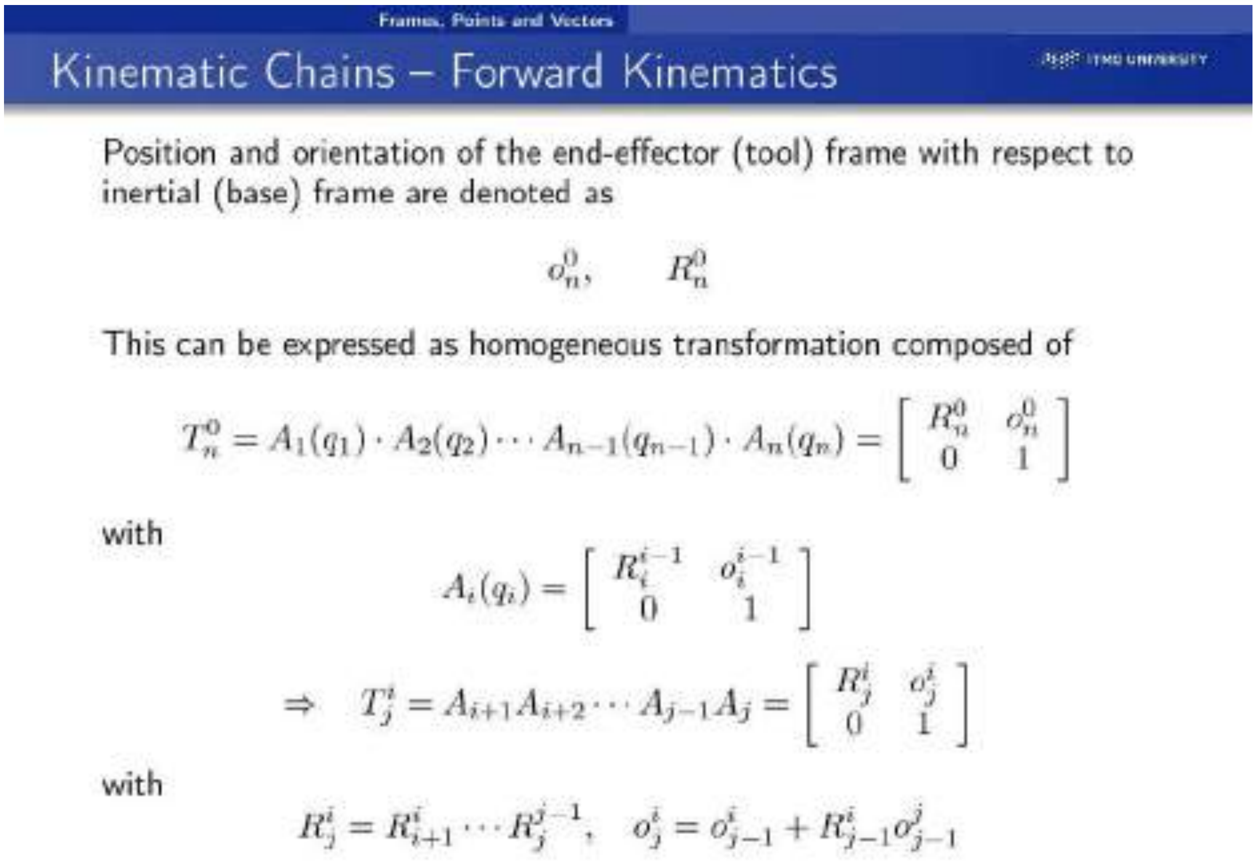
Basic Assumptions and Terminology:

- Suppose A_i is a homogeneous transformation that gives
  - position
  - orientation

  of the frame $(o_i x_i y_i z_i)$ with respect to the frame $(o_{i-1} x_{i-1} y_{i-1} z_{i-1})$.
- The matrix A_i is changing as the robot configuration changes.
- Homogeneous transformation A_i is a function of a scalar variable: $A_i = A_i(q_i)$.
- Homogeneous transformation that expresses the position and orientation of $(o_j x_j y_j z_j)$ with respect to $(o_i x_i y_i z_i)$

$$T_j^i = \begin{cases} A_{i+1} A_{i+2} \cdots A_{j-1} A_j & \text{if } i < j \\ I & \text{if } i = j \\ T_j^i = (T_i^j)^{-1} & \text{if } i > j \end{cases}$$

is called a transformation matrix.



Frames, Points and Vectors

Kinematic Chains – Forward Kinematics

ITMO UNIVERSITY

Position and orientation of the end-effector (tool) frame with respect to inertial (base) frame are denoted as

$$o_n^0, \qquad R_n^0$$

This can be expressed as homogeneous transformation composed of

$$T_n^0 = A_1(q_1) \cdot A_2(q_2) \cdots A_{n-1}(q_{n-1}) \cdot A_n(q_n) = \begin{bmatrix} R_n^0 & o_n^0 \\ 0 & 1 \end{bmatrix}$$

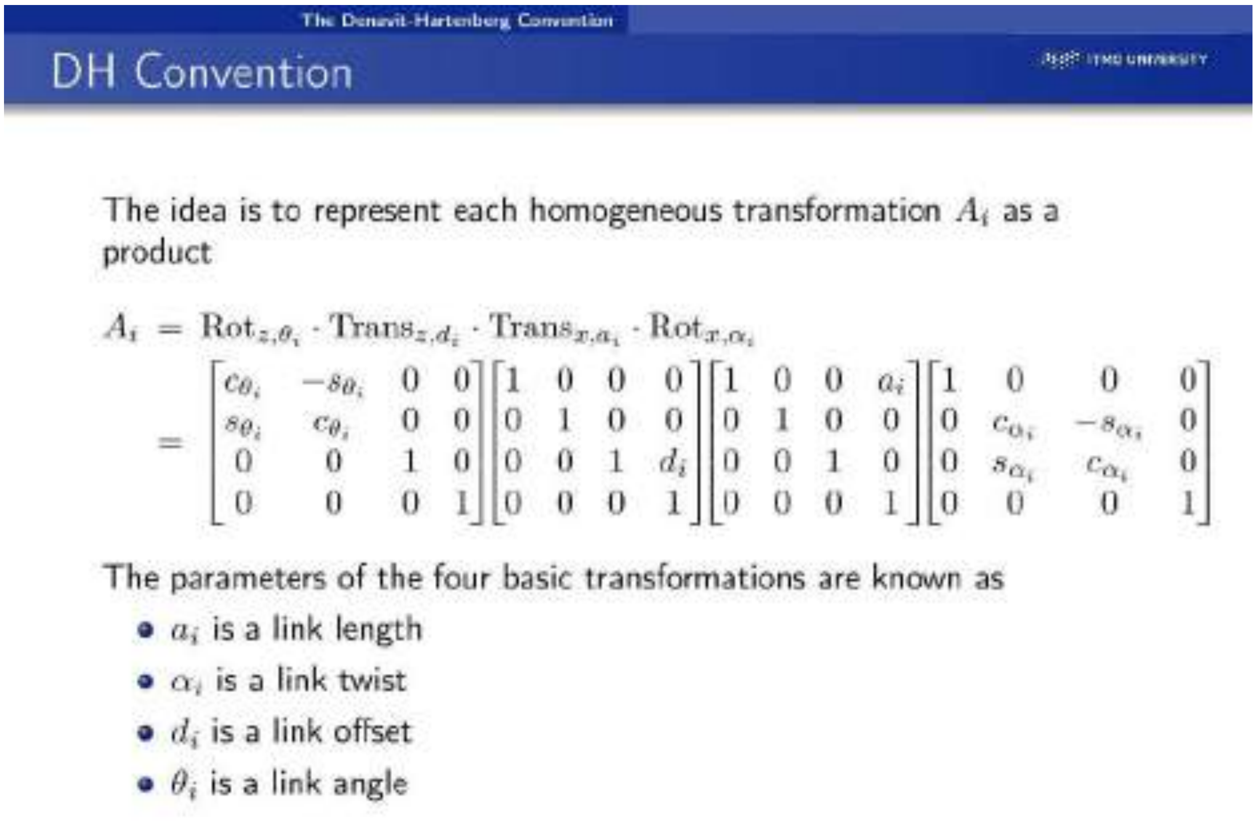
with

$$A_i(q_i) = \begin{bmatrix} R_i^{i-1} & o_i^{i-1} \\ 0 & 1 \end{bmatrix}$$

$$\Rightarrow \quad T_j^i = A_{i+1} A_{i+2} \cdots A_{j-1} A_j = \begin{bmatrix} R_j^i & o_j^i \\ 0 & 1 \end{bmatrix}$$

with

$$R_j^i = R_{i+1}^i \cdots R_j^{j-1}, \quad o_j^i = o_{j-1}^i + R_{j-1}^i o_{j-1}^j$$



The Denavit-Hartenberg Convention

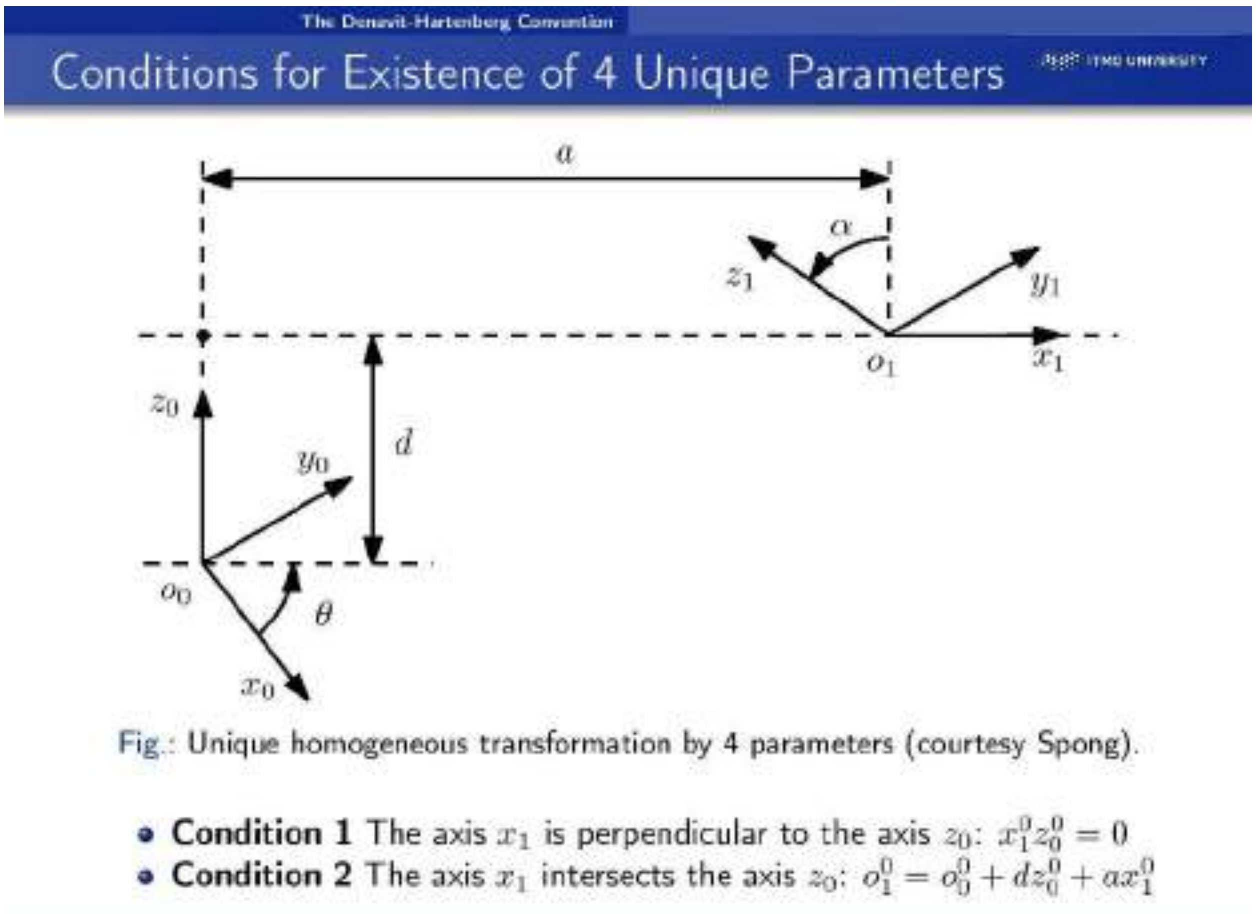
DH Convention

The idea is to represent each homogeneous transformation A_i as a product

$$A_i = \mathrm{Rot}_{z,\theta_i} \cdot \mathrm{Trans}_{z,d_i} \cdot \mathrm{Trans}_{x,a_i} \cdot \mathrm{Rot}_{x,\alpha_i}$$
$$= \begin{bmatrix} c_{\theta_i} & -s_{\theta_i} & 0 & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_{\alpha_i} & -s_{\alpha_i} & 0 \\ 0 & s_{\alpha_i} & c_{\alpha_i} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The parameters of the four basic transformations are known as

- a_i is a link length
- α_i is a link twist
- d_i is a link offset
- θ_i is a link angle



The Denavit-Hartenberg Convention

Conditions for Existence of 4 Unique Parameters

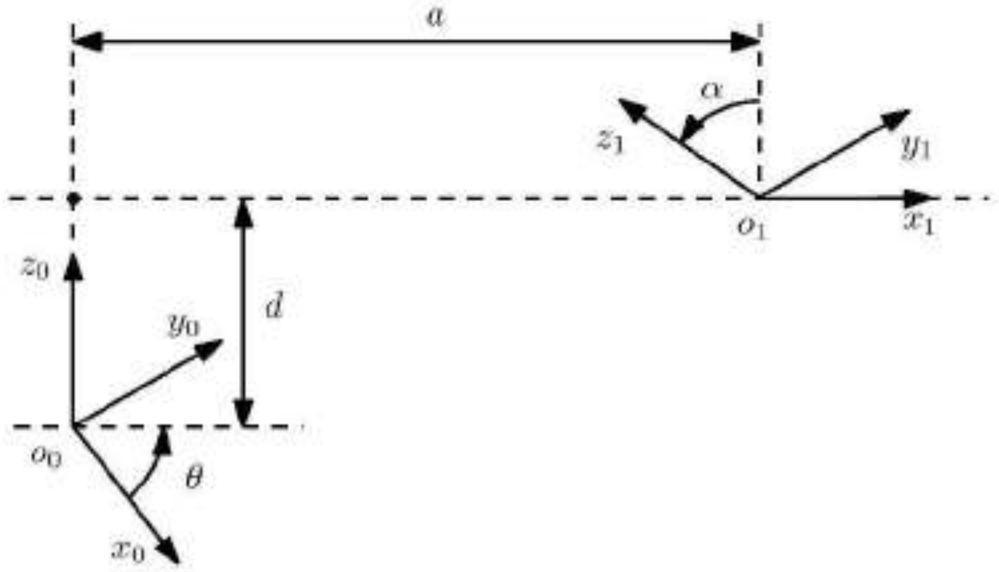
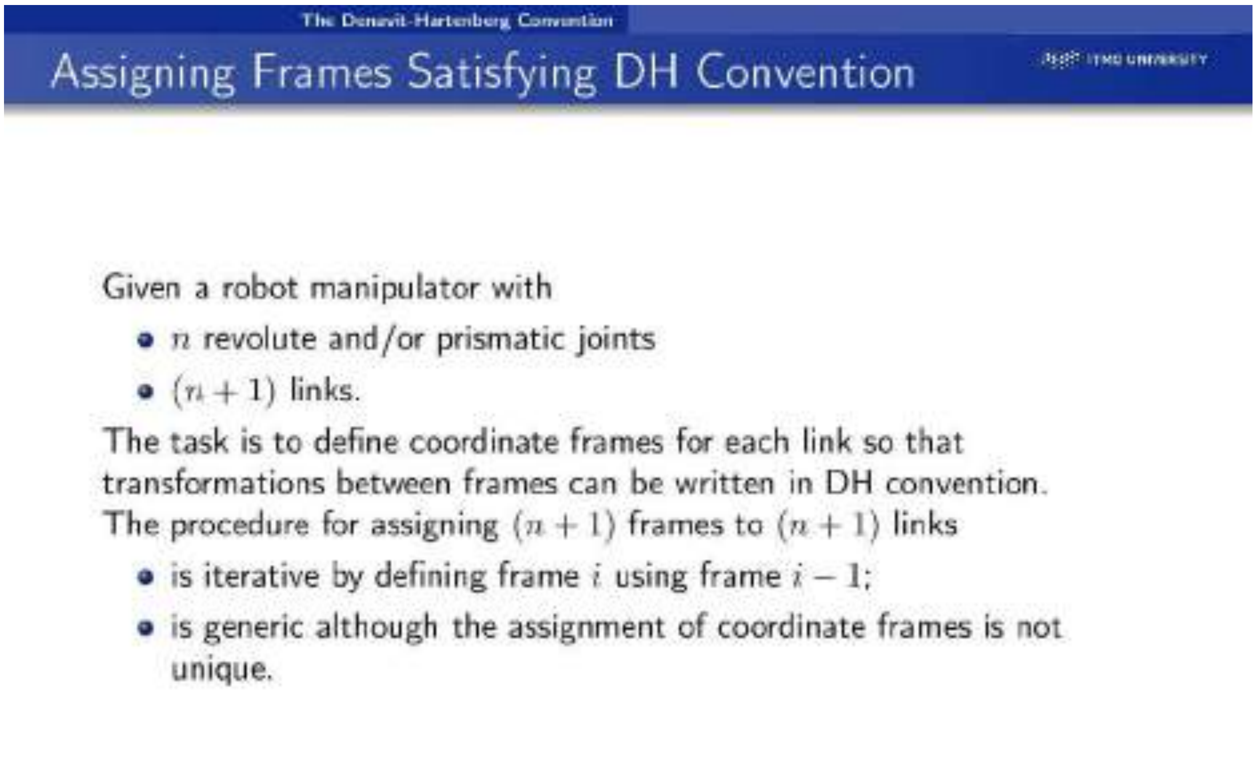


Fig.: Unique homogeneous transformation by 4 parameters (courtesy Spong).

- **Condition 1** The axis x_1 is perpendicular to the axis z_0: $x_1^0 z_0^0 = 0$
- **Condition 2** The axis x_1 intersects the axis z_0: $o_1^0 = o_0^0 + dz_0^0 + ax_1^0$



The Denavit-Hartenberg Convention

Assigning Frames Satisfying DH Convention

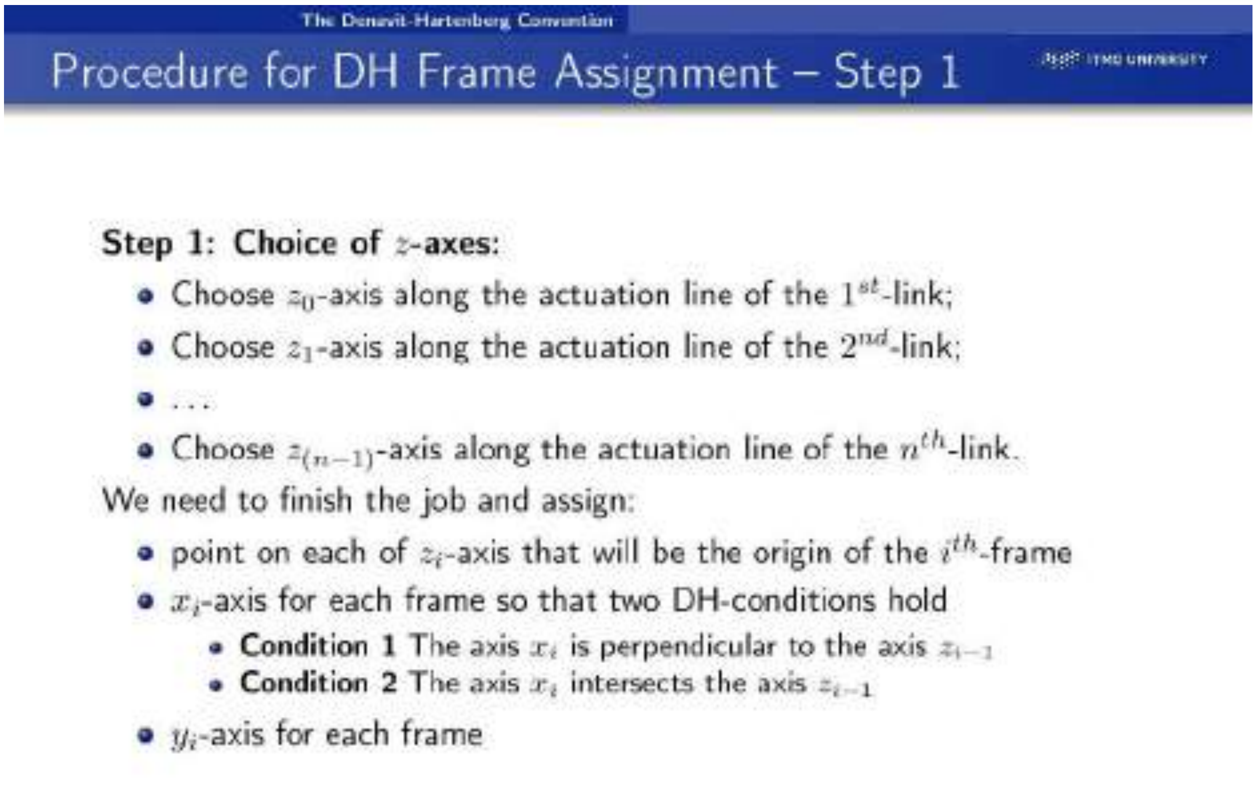
ITMO UNIVERSITY

Given a robot manipulator with

- n revolute and/or prismatic joints
- $(n+1)$ links.

The task is to define coordinate frames for each link so that transformations between frames can be written in DH convention.
The procedure for assigning $(n+1)$ frames to $(n+1)$ links

- is iterative by defining frame i using frame $i-1$;
- is generic although the assignment of coordinate frames is not unique.



The Denavit-Hartenberg Convention

Procedure for DH Frame Assignment – Step 1

ITMO UNIVERSITY

Step 1: Choice of z-axes:

- Choose z_0-axis along the actuation line of the 1^{st}-link;
- Choose z_1-axis along the actuation line of the 2^{nd}-link;
- ...
- Choose $z_{(n-1)}$-axis along the actuation line of the n^{th}-link.

We need to finish the job and assign:

- point on each of z_i-axis that will be the origin of the i^{th}-frame
- x_i-axis for each frame so that two DH-conditions hold
  - **Condition 1** The axis x_i is perpendicular to the axis z_{i-1}
  - **Condition 2** The axis x_i intersects the axis z_{i-1}
- y_i-axis for each frame

The Denavit-Hartenberg Convention

Procedure for DH Frame Assignment – Step 2

ITMO UNIVERSITY

Step 2 – Choice of x-axes and origins o:

- Suppose that we have chosen the $(i-1)^{th}$-frame and need to proceed with the i^{th}-frame.
- For the i^{th}-frame, the z_i axis is already fixed.
- To meet conditions both conditions the x_i-axis must intersect z_{i-1} and $x_i \perp z_{i-1}$ and $x_i \perp z_i$
- 3 cases for assigning the new origin o_i and the x_i-axis:
  1. z_i and z_{i-1} are not coplanar:
     $\Rightarrow$ one common perpendicular line exists between both vectors
  2. z_i and z_{i-1} are parallel:
     $\Rightarrow$ infinitely many common perpendicular lines to choose from
  3. z_i and z_{i-1} intersect:
     $\Rightarrow$ normal vector of the plane spanned by z_i and z_{i-1}

The Denavit-Hartenberg Convention

Procedure for DH Frame Assignment – Step 3

ITMO UNIVERSITY

Step 3 – Choice of y-axes: If we have already chosen the vectors z_i, x_i and the point o_i for the i^{th}-frame, y_i can be assigned by cross-product operation: $\vec{y_i} = \vec{z_i} \times \vec{x_i}$

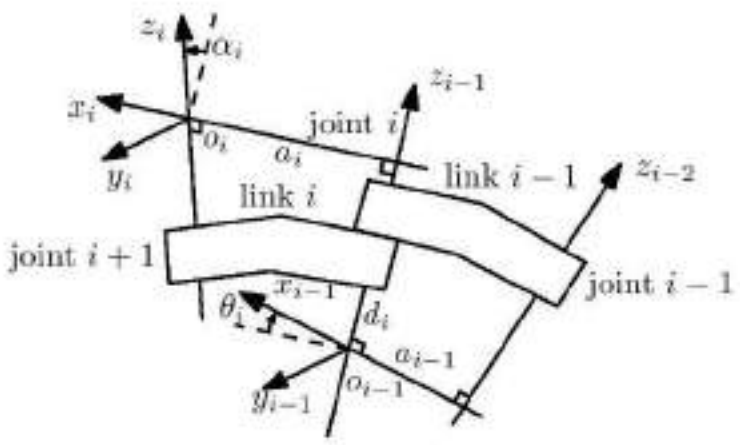


Fig.: Illustration of Denavit-Hartenberg frame assignment (courtesy Spong).

Assigning the End-Effector Frame n (Tool Frame)

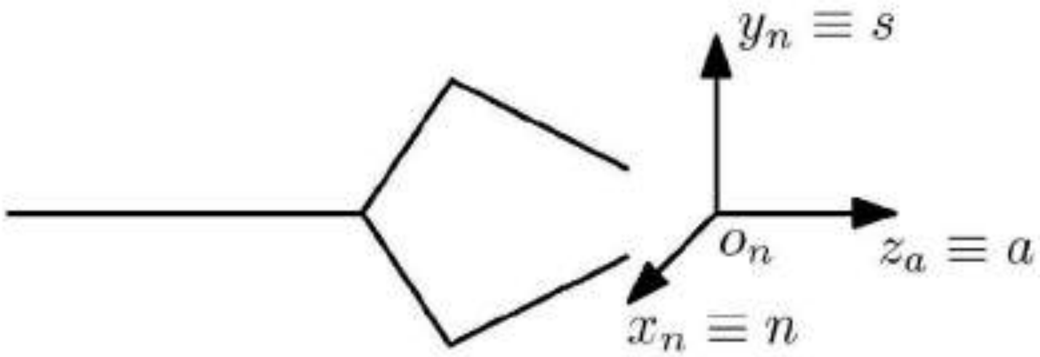


Fig.: Tool frame assignment (courtesy Spong).

For most robots z_{n-1} and z_n coincide so the final transformation is

- translation by d_n along z_{n-1}-axis
- rotation by θ_n about z_n-axis

Example: Planar Two-Link Manipulator

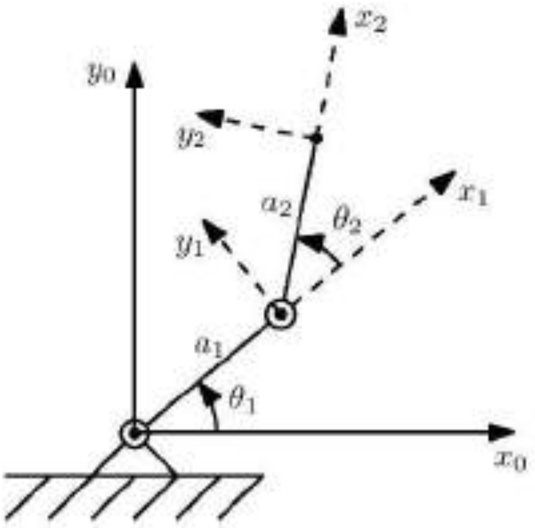


Fig.: Planar two-link manipulator with z-axes pointing out (courtesy Spong).

Example: Three-Link Cylindrical Manipulator

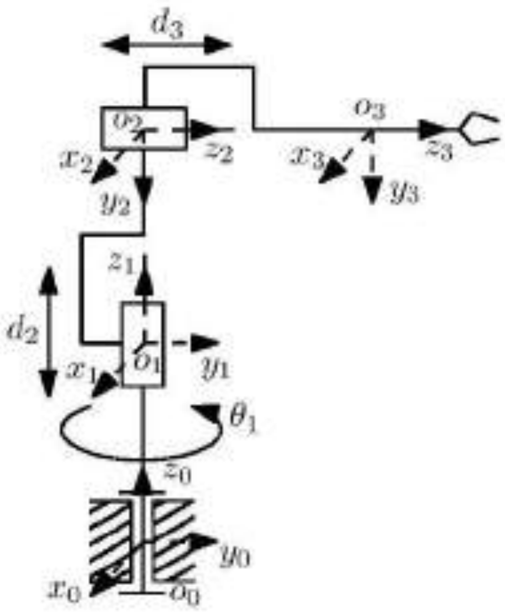


Fig.: Three-link cylindrical manipulator (courtesy Spong).

DH-parameters:

Link	a_i	α_i	d_i	θ_i
1	0	0	d_1	θ_1
2	0	$-\frac{\pi}{2}$	d_2	0
3	0	0	d_3	0

Example: Three-Link Cylindrical Manipulator

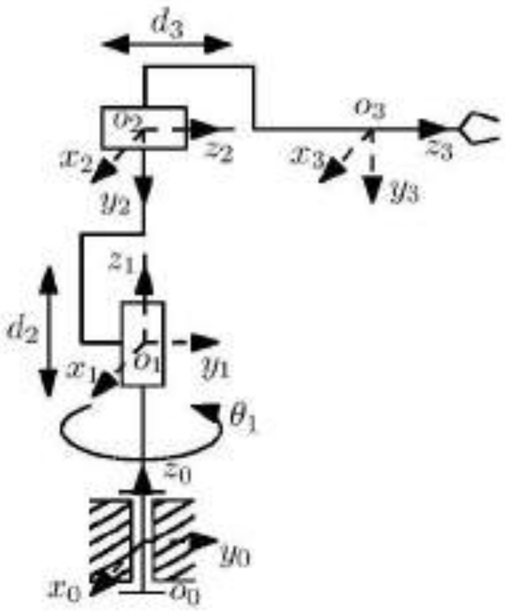


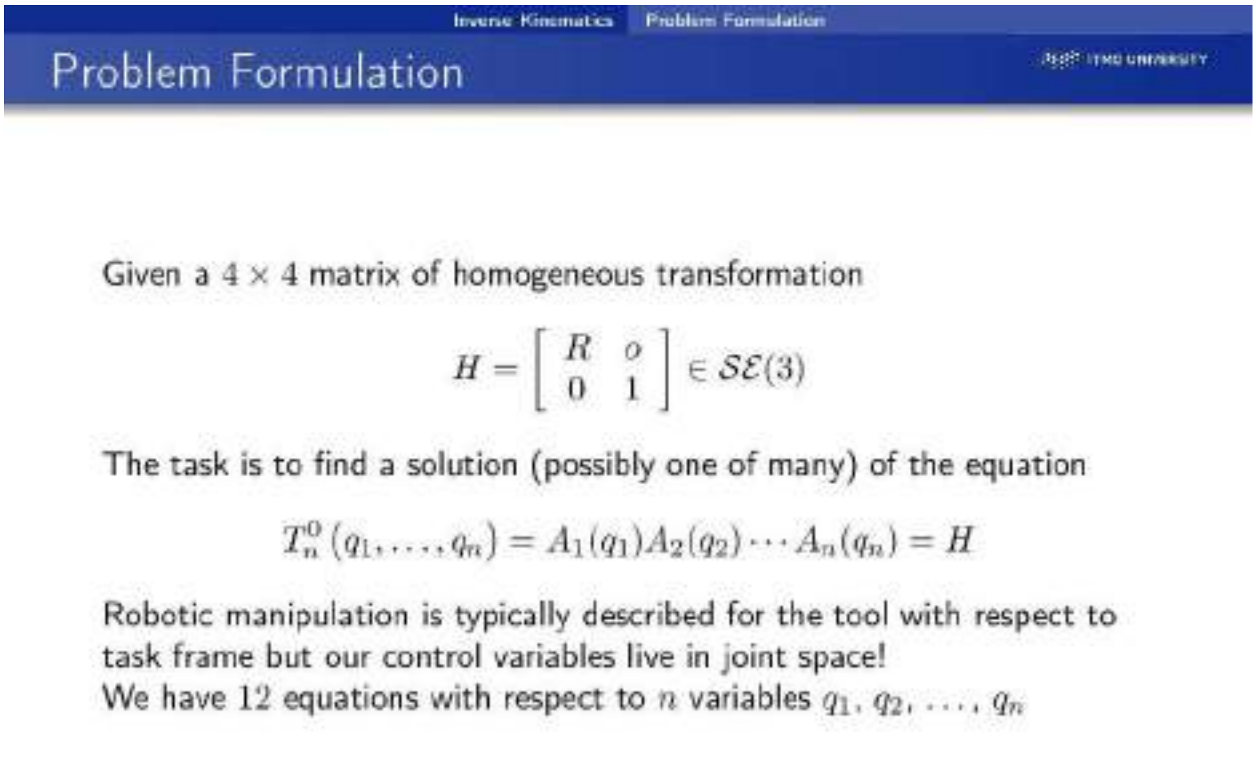
Fig.: Three-link cylindrical manipulator (courtesy Spong).

$$A_1 = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(-\frac{\pi}{2}) & -\sin(-\frac{\pi}{2}) & 0 \\ 0 & \sin(-\frac{\pi}{2}) & \cos(-\frac{\pi}{2}) & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_3^0 = A_1 A_2 A_3$$


Inverse Kinematics
Problem Formulation
ITMO UNIVERSITY

Problem Formulation

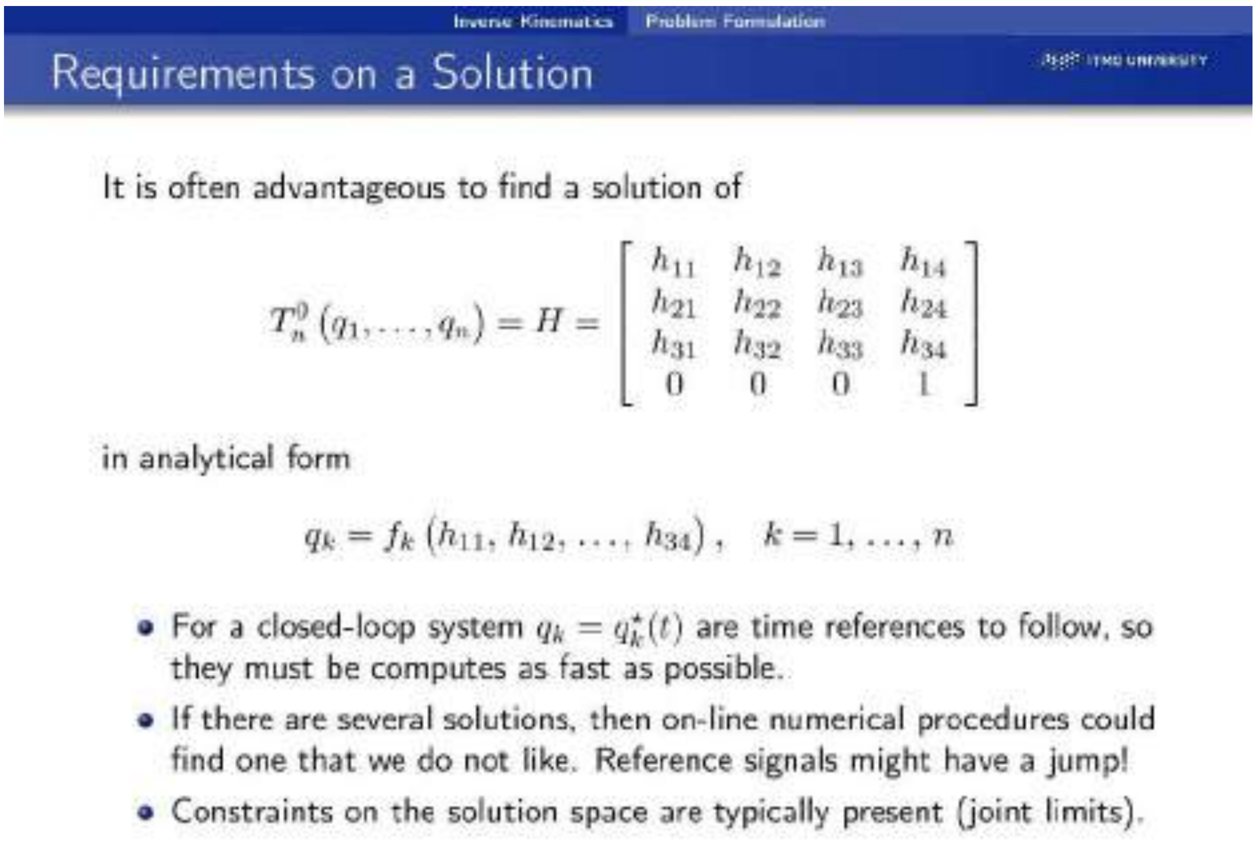
Given a 4×4 matrix of homogeneous transformation

$$H = \begin{bmatrix} R & o \\ 0 & 1 \end{bmatrix} \in \mathcal{SE}(3)$$

The task is to find a solution (possibly one of many) of the equation

$$T_n^0\left(q_1, \ldots, q_n\right) = A_1(q_1)A_2(q_2)\cdots A_n(q_n) = H$$

Robotic manipulation is typically described for the tool with respect to task frame but our control variables live in joint space!
We have 12 equations with respect to n variables $q_1, q_2, \ldots, q_n$


Inverse Kinematics
Problem Formulation
ITMO UNIVERSITY

Requirements on a Solution

It is often advantageous to find a solution of

$$T_n^0\left(q_1, \ldots, q_n\right) = H = \begin{bmatrix} h_{11} & h_{12} & h_{13} & h_{14} \\ h_{21} & h_{22} & h_{23} & h_{24} \\ h_{31} & h_{32} & h_{33} & h_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

in analytical form

$$q_k = f_k\left(h_{11}, h_{12}, \ldots, h_{34}\right), \quad k = 1, \ldots, n$$

- For a closed-loop system $q_k = q_k^\star(t)$ are time references to follow, so they must be computes as fast as possible.
- If there are several solutions, then on-line numerical procedures could find one that we do not like. Reference signals might have a jump!
- Constraints on the solution space are typically present (joint limits).

Kinematic Decoupling

Given $R \in \mathcal{SO}(3)$ and $o \in \mathbb{R}^3$

- Problem of inverse kinematics is quite difficult in general
- For particular robot manipulators with
  - at least 6 joints
  - the last 3 joint axes intersecting in one point (spherical wrist, wrist center)

  we can separate the two sub-problems
  - inverse position kinematics
  - inverse orientation kinematics

Kinematic Decoupling

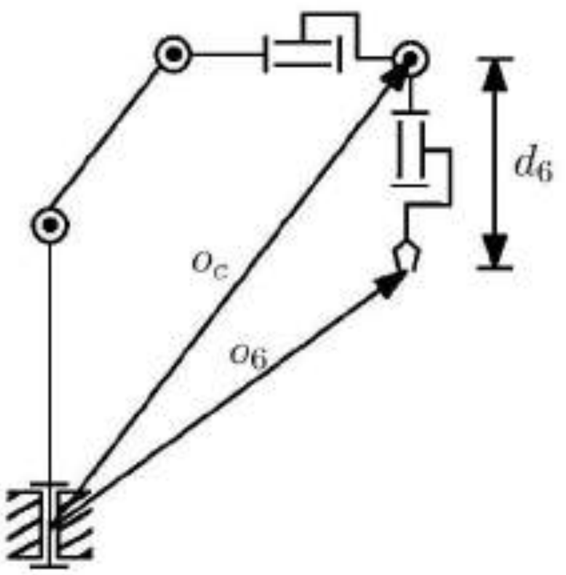


Fig.: Kinematic decoupling at the wrist center (courtesy Spong).

How to compute o_c^0 and R_3^0, R_6^3 ?

Kinematic Decoupling

If the last three joint axes intersect in one point then

$$o = o_c^0 + d_6 \cdot R \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = o_c^0 + d_6 \cdot \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

with o and R being desired position and orientation of the tool frame. Therefore,

$$o_c^0 = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = o - d_6 \cdot R \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} o_x - d_6 r_{13} \\ o_y - d_6 r_{23} \\ o_z - d_6 r_{33} \end{bmatrix}$$

We know that

$$R = R_3^0 \cdot R_6^3 \quad \Rightarrow \quad R_6^3 = \left[R_3^0\right]^{-1} R = \left[R_3^0\right]^T R$$

Inverse Position Kinematics – Elbow Manipulator

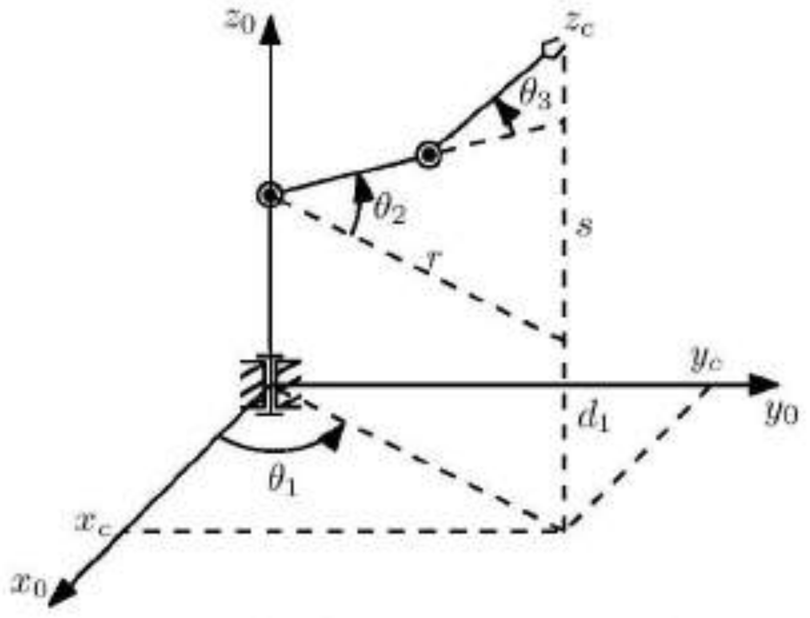


Fig.: Inverse position problem: find angles θ_1, θ_2 and θ_3 (courtesy Spong).

Inverse Position Kinematics – Elbow Manipulator

Angle θ_1:

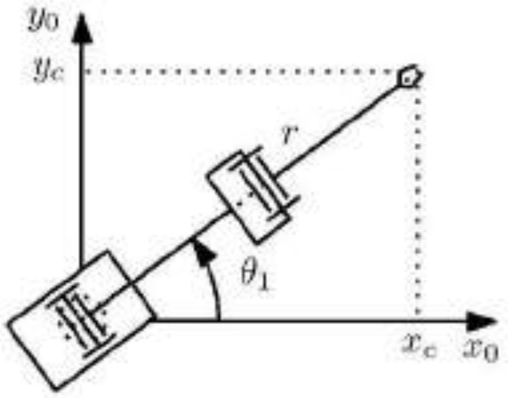


Fig.: Projection of the wrist center onto x_0-y_0 plane (courtesy Spong).

$$\theta_1 = \text{atan2}(y_c, x_c)$$

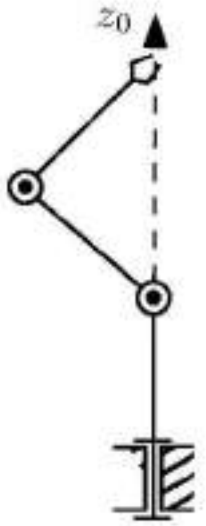


Fig.: Singular configuration (courtesy Spong).

Inverse Position Kinematics – Elbow Manipulator

Angles θ_2 and θ_3:

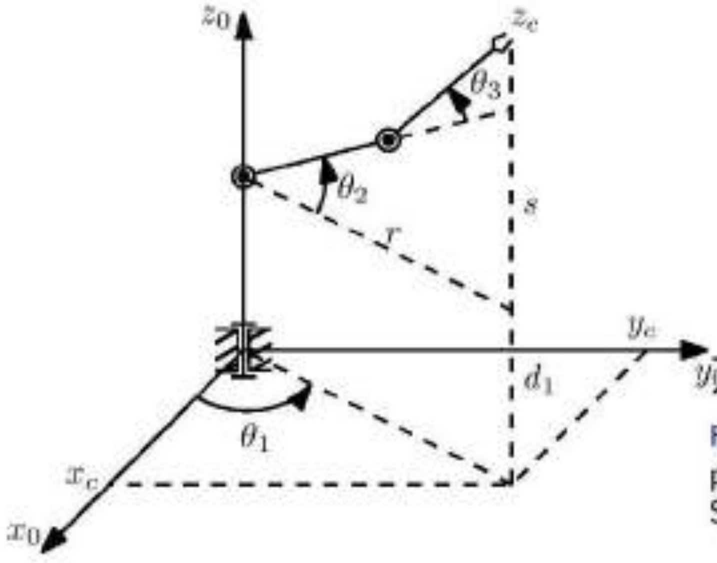


Fig.: Inverse position problem (courtesy Spong).

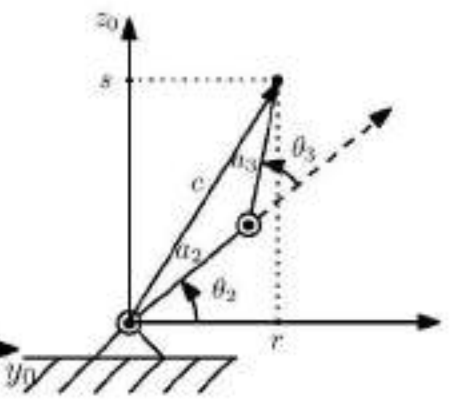


Fig.: Projection onto moving plane of link 2 and 3 (courtesy Spong).

Inverse Position Kinematics – Elbow Manipulator

Angles θ_2 and θ_3:

Consider the vector sum

$$\begin{aligned}
\vec{c} &= [r,\ s]^T = \vec{a}_2 + \vec{a}_3 \\
\|\vec{c}\|^2 &= \|\vec{a}_2 + \vec{a}_3\|^2 \\
r^2 + s^2 &= (\vec{a}_2 + \vec{a}_3) \cdot (\vec{a}_2 + \vec{a}_3) \\
r^2 + s^2 &= a_2^2 + a_3^2 + 2\vec{a}_2\vec{a}_3 \\
r^2 + s^2 &= a_2^2 + a_3^2 + 2a_2a_3 \cos\theta_3 \\
\cos\theta_3 &= \frac{r^2 + s^2 - a_2^2 - a_3^2}{2a_2a_3} = D \\
&= \frac{(x_c^2 + y_c^2) + (z_c - d_1)^2 - a_2^2 - a_3^2}{2a_2a_3} \\
1 &= (\sin\theta_3)^2 + (\cos\theta_3)^2 \\
\sin\theta_3 &= \pm\sqrt{1 - D^2}
\end{aligned}$$

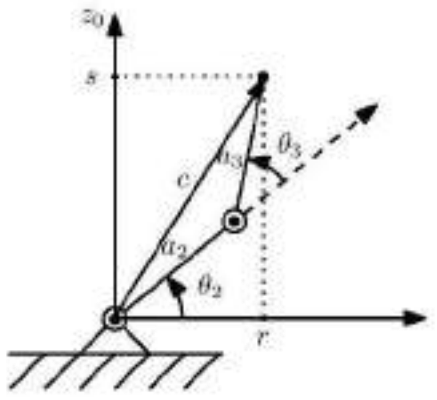


Fig.: Projection onto moving plane of link 2 and 3 (courtesy Spong).

Inverse Position Kinematics – Elbow Manipulator

Angles θ_2 and θ_3:

$$\begin{aligned}
\theta_3 &= \operatorname{atan2}(\pm\sqrt{1 - D^2}, D) \\
\theta_2 &= \operatorname{atan2}(s, r) \\
&\quad - \operatorname{atan2}(a_3 \sin\theta_3, a_2 + a_3 \cos\theta_3)
\end{aligned}$$

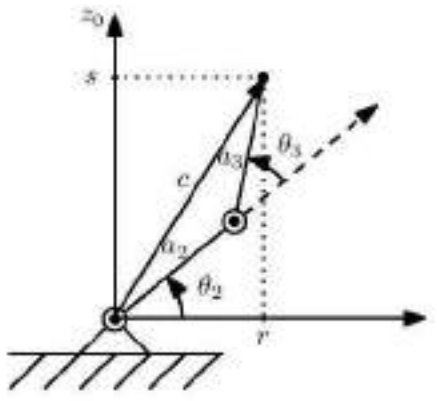
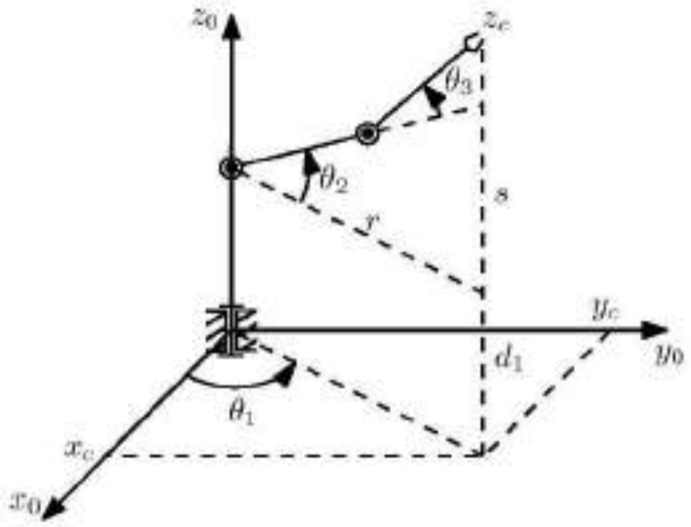


Fig.: Projection onto moving plane of link 2 and 3 (courtesy Spong).

Inverse Orientation Kinematics – Elbow Manipulator



What is the rotation matrix from 0-frame to 3-frame (wrist center)?

- Link 1: $a = 0$, $\alpha = \frac{\pi}{2}$, $d = d_1$, $\theta = \theta_1 \quad \Rightarrow \quad H_1^0$
- Link 2: $a = a_2$, $\alpha = 0$, $d = 0$, $\theta = \theta_2 \quad \Rightarrow \quad H_2^1$
- Link 3: $a = a_3$, $\alpha = 0$, $d = 0$, $\theta = \theta_3 \quad \Rightarrow \quad H_3^2$

Inverse Orientation Kinematics – Elbow Manipulator

Given $R \in SO(3)$ and $o \in R^3$,

$$\text{We have computed } H_3^0 = H_1^0 H_2^1 H_3^2 \quad \Rightarrow \quad R_3^0, \quad o_3^0 = o_c^0$$

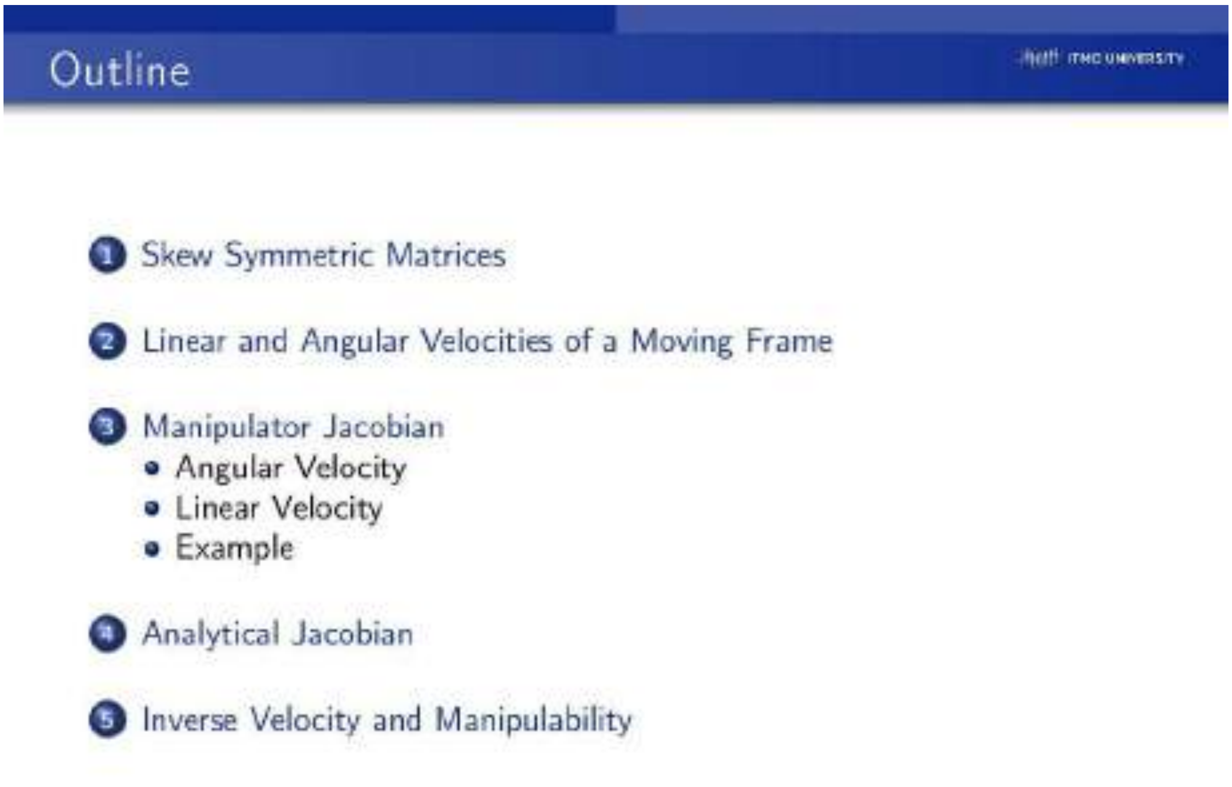
Therefore, we can compute the rotation matrix

$$R_6^3 = \left[R_3^0\right]^{-1} \cdot R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

We need to compute, e.g., Euler angles (ϕ, θ, ψ) such that

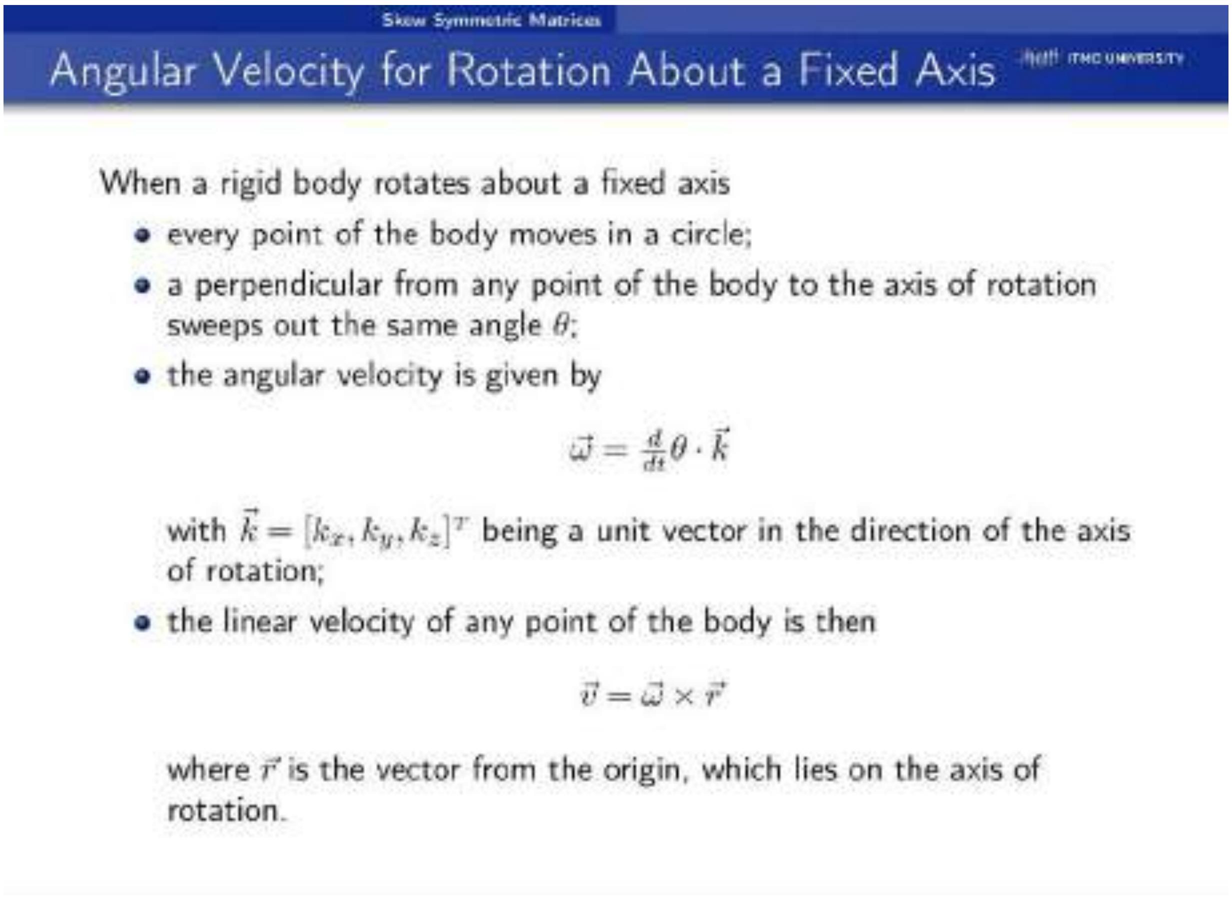
$$R_6^3 = R_{z,\phi} \cdot R_{y,\theta} \cdot R_{z,\psi}$$

Kinematics: Velocity Kinematics - the Jacobian



Outline

1. Skew Symmetric Matrices
2. Linear and Angular Velocities of a Moving Frame
3. Manipulator Jacobian
   - Angular Velocity
   - Linear Velocity
   - Example
4. Analytical Jacobian
5. Inverse Velocity and Manipulability



Skew Symmetric Matrices

Angular Velocity for Rotation About a Fixed Axis

When a rigid body rotates about a fixed axis

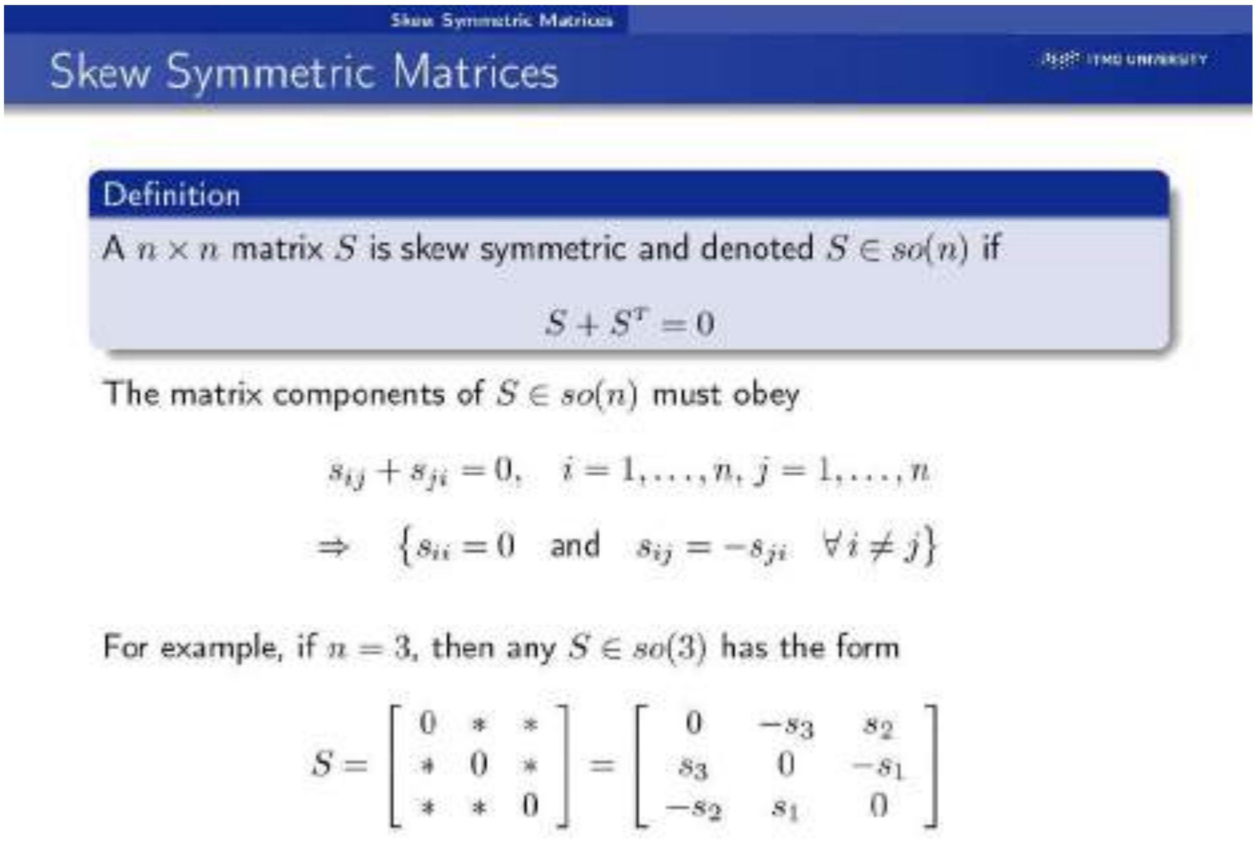
- every point of the body moves in a circle;
- a perpendicular from any point of the body to the axis of rotation sweeps out the same angle θ;
- the angular velocity is given by

$$\vec{\omega} = \frac{d}{dt}\theta \cdot \vec{k}$$

  with $\vec{k} = [k_x, k_y, k_z]^T$ being a unit vector in the direction of the axis of rotation;
- the linear velocity of any point of the body is then

$$\vec{v} = \vec{\omega} \times \vec{r}$$

  where $\vec{r}$ is the vector from the origin, which lies on the axis of rotation.



Skew Symmetric Matrices

Skew Symmetric Matrices

ITMO UNIVERSITY

Definition

A $n \times n$ matrix S is skew symmetric and denoted $S \in so(n)$ if

$$S + S^T = 0$$

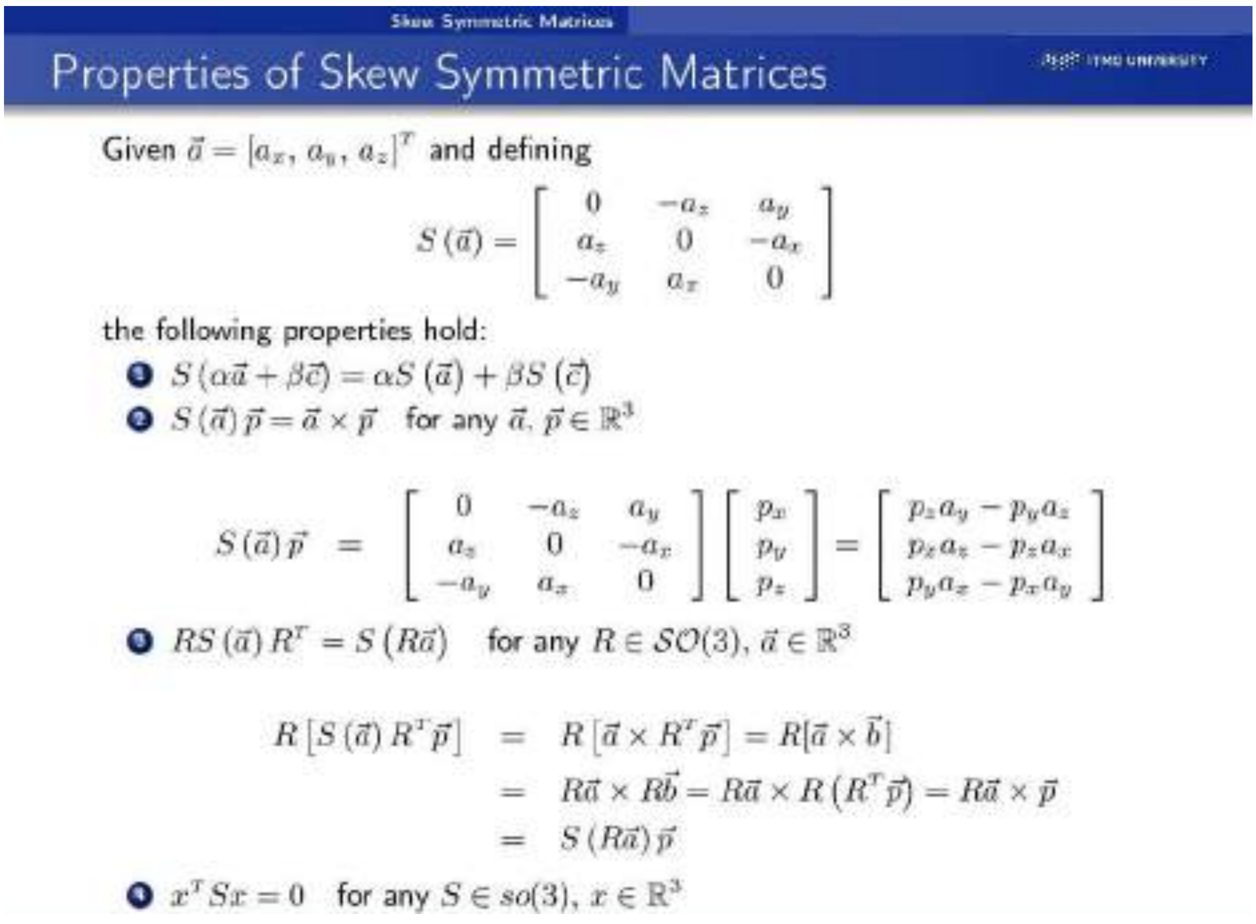
The matrix components of $S \in so(n)$ must obey

$$s_{ij} + s_{ji} = 0, \quad i = 1, \ldots, n, \; j = 1, \ldots, n$$

$$\Rightarrow \quad \{ s_{ii} = 0 \quad \text{and} \quad s_{ij} = -s_{ji} \quad \forall i \neq j \}$$

For example, if $n = 3$, then any $S \in so(3)$ has the form

$$S = \begin{bmatrix} 0 & * & * \\ * & 0 & * \\ * & * & 0 \end{bmatrix} = \begin{bmatrix} 0 & -s_3 & s_2 \\ s_3 & 0 & -s_1 \\ -s_2 & s_1 & 0 \end{bmatrix}$$



Skew Symmetric Matrices

Properties of Skew Symmetric Matrices

ITMO UNIVERSITY

Given $\vec{a} = [a_x, a_y, a_z]^T$ and defining

$$S(\vec{a}) = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix}$$

the following properties hold:

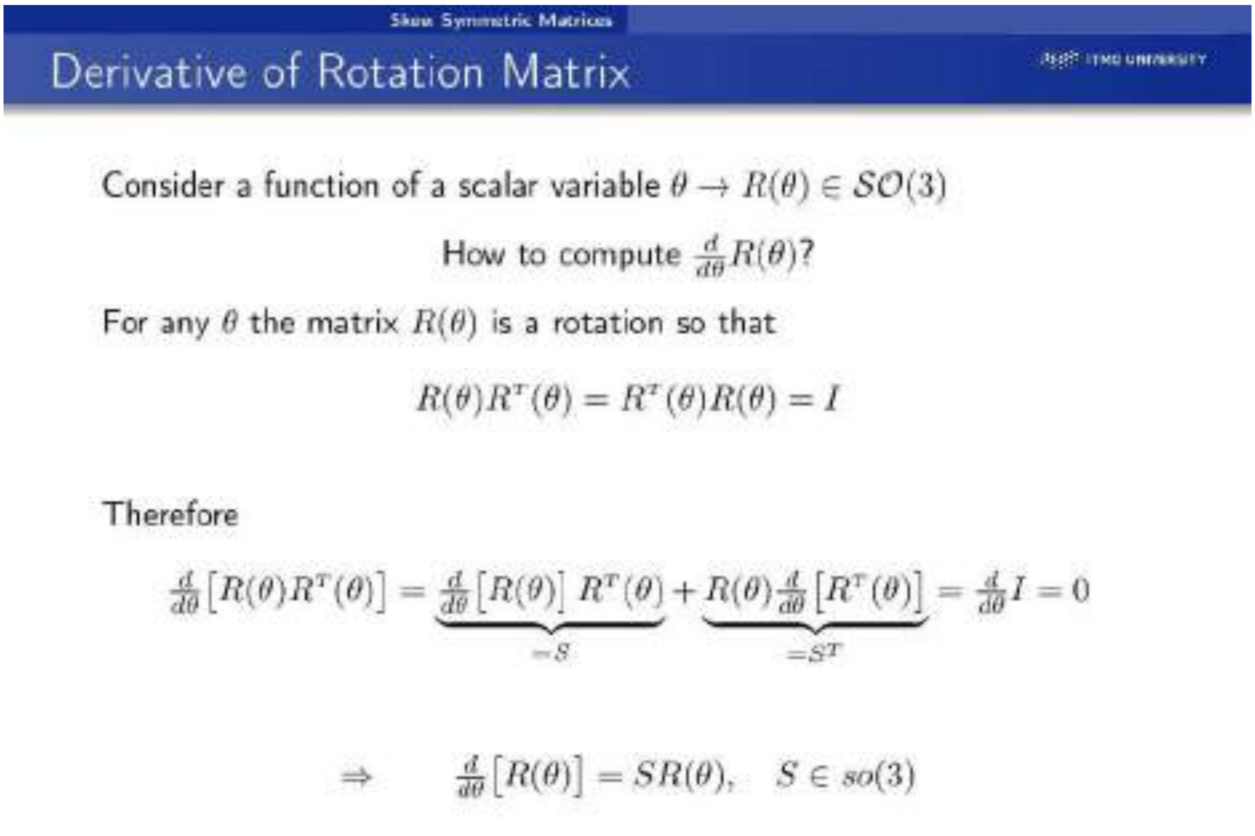
1. $S(\alpha\vec{a} + \beta\vec{c}) = \alpha S(\vec{a}) + \beta S(\vec{c})$
2. $S(\vec{a})\vec{p} = \vec{a} \times \vec{p}$ for any $\vec{a}, \vec{p} \in \mathbb{R}^3$

$$S(\vec{a})\vec{p} \;=\; \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} = \begin{bmatrix} p_z a_y - p_y a_z \\ p_x a_z - p_z a_x \\ p_y a_x - p_x a_y \end{bmatrix}$$

3. $RS(\vec{a})R^T = S(R\vec{a})$ for any $R \in SO(3)$, $\vec{a} \in \mathbb{R}^3$

$$\begin{aligned} R\left[S(\vec{a})R^T\vec{p}\right] &= R\left[\vec{a} \times R^T\vec{p}\right] = R[\vec{a} \times \vec{b}] \\ &= R\vec{a} \times R\vec{b} = R\vec{a} \times R\left(R^T\vec{p}\right) = R\vec{a} \times \vec{p} \\ &= S(R\vec{a})\vec{p} \end{aligned}$$

4. $x^T S x = 0$ for any $S \in so(3)$, $x \in \mathbb{R}^3$



Skew Symmetric Matrices

Derivative of Rotation Matrix

ITMO UNIVERSITY

Consider a function of a scalar variable $\theta \to R(\theta) \in \mathcal{SO}(3)$

How to compute $\frac{d}{d\theta}R(\theta)$?

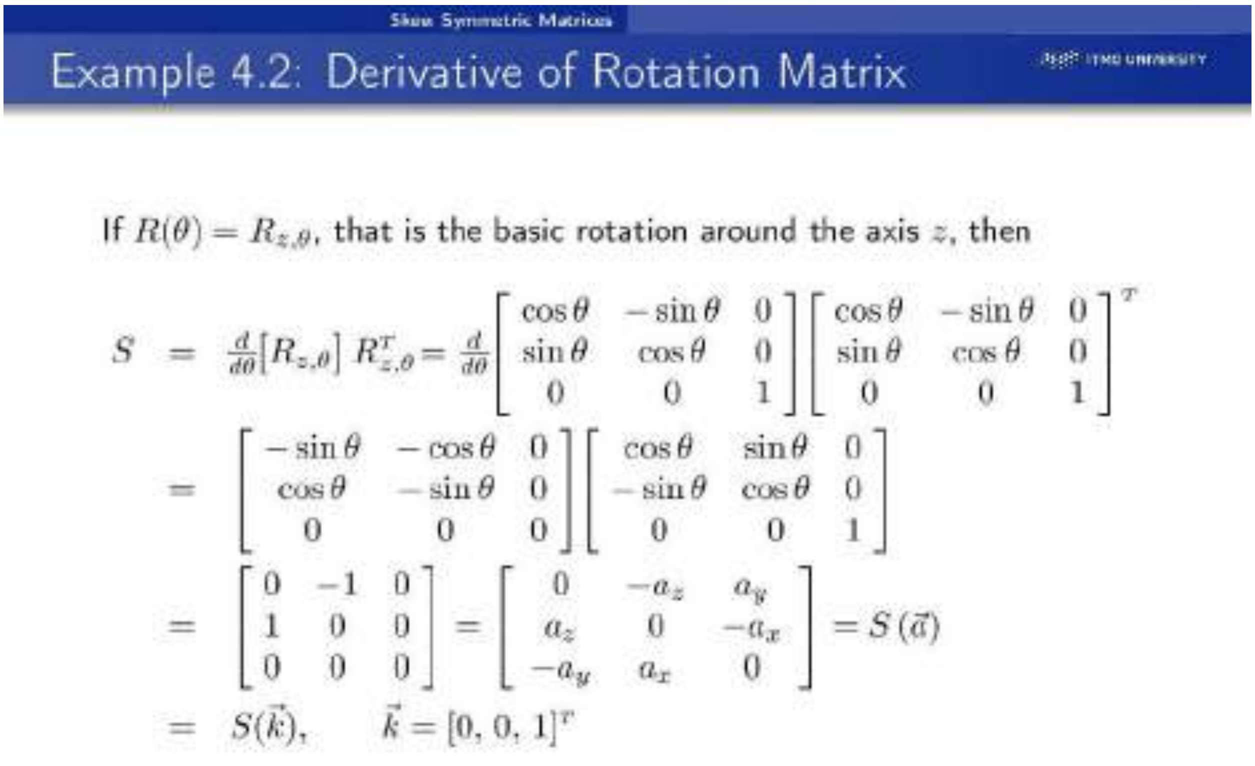
For any θ the matrix $R(\theta)$ is a rotation so that

$$R(\theta)R^T(\theta) = R^T(\theta)R(\theta) = I$$

Therefore

$$\tfrac{d}{d\theta}\left[R(\theta)R^T(\theta)\right] = \underbrace{\tfrac{d}{d\theta}\left[R(\theta)\right]\, R^T(\theta)}_{=S} + \underbrace{R(\theta)\tfrac{d}{d\theta}\left[R^T(\theta)\right]}_{=S^T} = \tfrac{d}{d\theta}I = 0$$

$$\Rightarrow \qquad \tfrac{d}{d\theta}\left[R(\theta)\right] = SR(\theta), \quad S \in so(3)$$



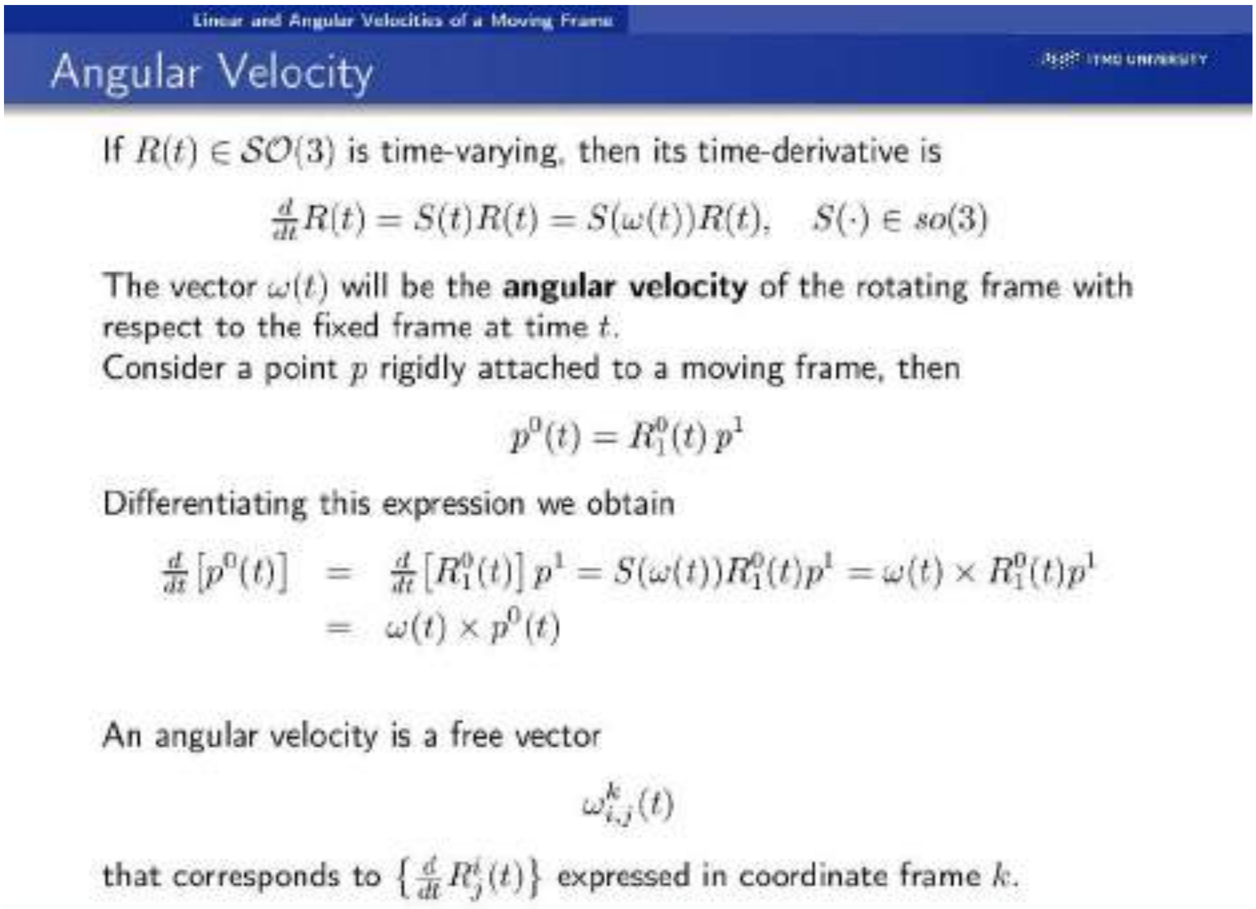
Skew Symmetric Matrices

Example 4.2: Derivative of Rotation Matrix

ITMO UNIVERSITY

If $R(\theta) = R_{z,\theta}$, that is the basic rotation around the axis z, then

$$\begin{aligned} S &= \tfrac{d}{d\theta}\left[R_{z,\theta}\right] R^T_{z,\theta} = \tfrac{d}{d\theta}\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}^T \\ &= \begin{bmatrix} -\sin\theta & -\cos\theta & 0 \\ \cos\theta & -\sin\theta & 0 \\ 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} = S(\vec{a}) \\ &= S(\vec{k}), \qquad \vec{k} = [0,\, 0,\, 1]^T \end{aligned}$$



Linear and Angular Velocities of a Moving Frame

Angular Velocity

ITMO UNIVERSITY

If $R(t) \in \mathcal{SO}(3)$ is time-varying, then its time-derivative is

$$\tfrac{d}{dt}R(t) = S(t)R(t) = S(\omega(t))R(t), \quad S(\cdot) \in so(3)$$

The vector $\omega(t)$ will be the **angular velocity** of the rotating frame with respect to the fixed frame at time t.
Consider a point p rigidly attached to a moving frame, then

$$p^0(t) = R_1^0(t)\, p^1$$

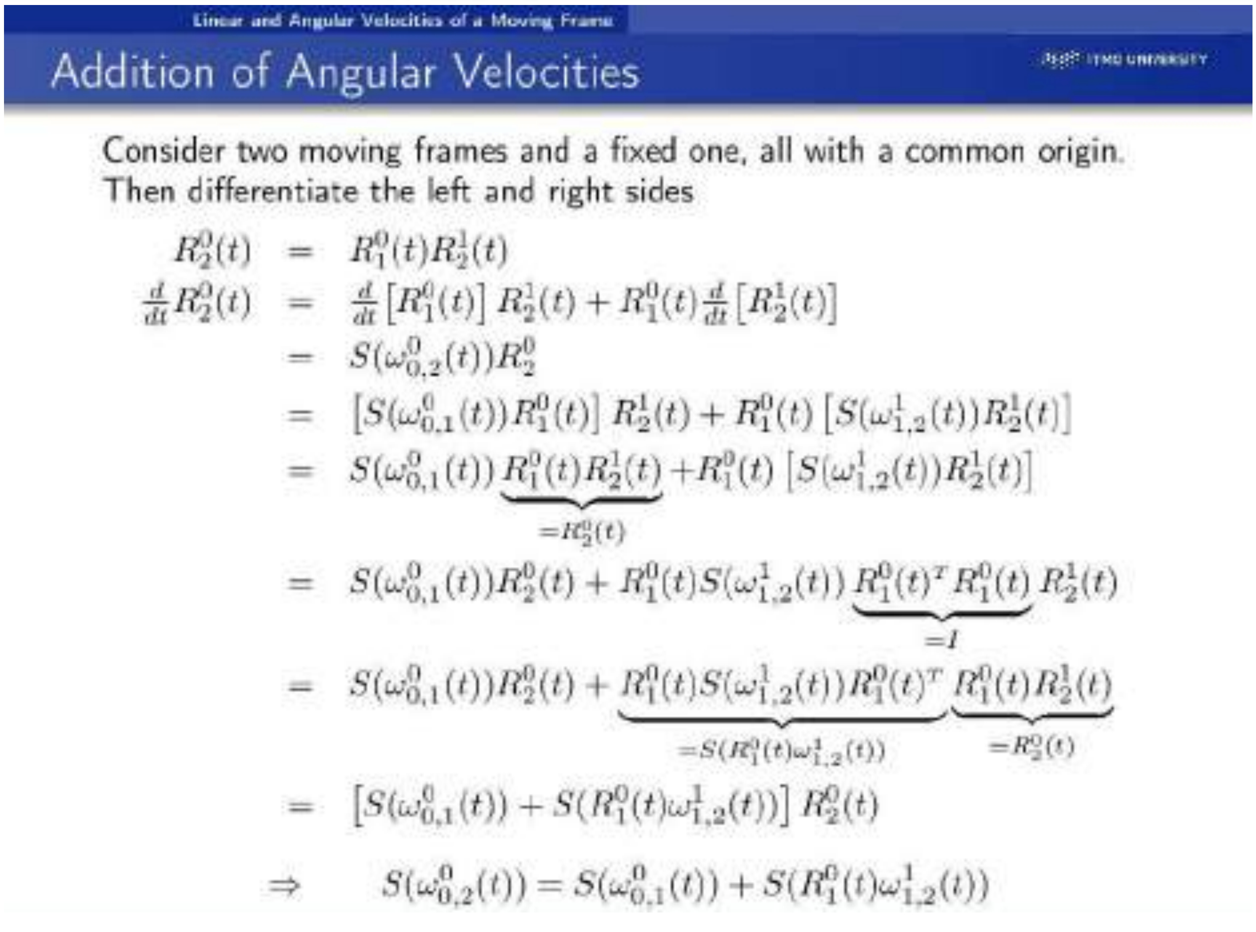
Differentiating this expression we obtain

$$\begin{aligned} \tfrac{d}{dt}\left[p^0(t)\right] &= \tfrac{d}{dt}\left[R_1^0(t)\right] p^1 = S(\omega(t))R_1^0(t)p^1 = \omega(t) \times R_1^0(t)p^1 \\ &= \omega(t) \times p^0(t) \end{aligned}$$

An angular velocity is a free vector

$$\omega_{i,j}^k(t)$$

that corresponds to $\left\{\tfrac{d}{dt}R_j^i(t)\right\}$ expressed in coordinate frame k.



Linear and Angular Velocities of a Moving Frame

Addition of Angular Velocities

ITMO UNIVERSITY

Consider two moving frames and a fixed one, all with a common origin. Then differentiate the left and right sides

$$\begin{aligned} R_2^0(t) &= R_1^0(t)R_2^1(t) \\ \tfrac{d}{dt}R_2^0(t) &= \tfrac{d}{dt}\left[R_1^0(t)\right] R_2^1(t) + R_1^0(t)\tfrac{d}{dt}\left[R_2^1(t)\right] \\ &= S(\omega_{0,2}^0(t))R_2^0 \\ &= \left[S(\omega_{0,1}^0(t))R_1^0(t)\right] R_2^1(t) + R_1^0(t)\left[S(\omega_{1,2}^1(t))R_2^1(t)\right] \\ &= S(\omega_{0,1}^0(t))\underbrace{R_1^0(t)R_2^1(t)}_{=R_2^0(t)} + R_1^0(t)\left[S(\omega_{1,2}^1(t))R_2^1(t)\right] \\ &= S(\omega_{0,1}^0(t))R_2^0(t) + R_1^0(t)S(\omega_{1,2}^1(t))\underbrace{R_1^0(t)^T R_1^0(t)}_{=I} R_2^1(t) \\ &= S(\omega_{0,1}^0(t))R_2^0(t) + \underbrace{R_1^0(t)S(\omega_{1,2}^1(t))R_1^0(t)^T}_{=S(R_1^0(t)\omega_{1,2}^1(t))} \underbrace{R_1^0(t)R_2^1(t)}_{=R_2^0(t)} \\ &= \left[S(\omega_{0,1}^0(t)) + S(R_1^0(t)\omega_{1,2}^1(t))\right] R_2^0(t) \end{aligned}$$

$$\Rightarrow \qquad S(\omega_{0,2}^0(t)) = S(\omega_{0,1}^0(t)) + S(R_1^0(t)\omega_{1,2}^1(t))$$

Linear and Angular Velocities of a Moving Frame

Addition of Angular Velocities

The relation

$$S(\omega^0_{0,2}(t)) = S(\omega^0_{0,1}(t)) + S(R^0_1(t)\omega^1_{1,2}(t))$$

together with the property $S(a) + S(c) = S(a + c)$ imply that the angular velocity can be computed as

$$\omega^0_{0,2}(t) = \omega^0_{0,1}(t) + R^0_1(t)\omega^1_{1,2}(t) = \omega^0_{0,1}(t) + \omega^0_{1,2}(t)$$

Given n-moving frames with the same origins as for the fixed one

$$R^0_n(t) = R^0_1(t)R^1_2(t)\cdots R^{n-1}_n(t) \;\Rightarrow\; \frac{d}{dt}R^0_n(t) = S(\omega^0_{0,n}(t))R^0_n(t)$$

$$\begin{aligned} \omega^0_{0,n}(t) &= \omega^0_{0,1}(t) + \omega^0_{1,2}(t) + \omega^0_{2,3}(t) + \cdots + \omega^0_{n-1,n}(t) \\ &= \omega^0_{0,1} + R^0_1\omega^1_{1,2} + R^0_2\omega^2_{2,3} + \cdots + R^0_{n-1}\omega^{n-1}_{n-1,n} \end{aligned}$$

Linear and Angular Velocities of a Moving Frame

Linear Velocity of a Point

Given moving and fixed frames related by a homogeneous transformation

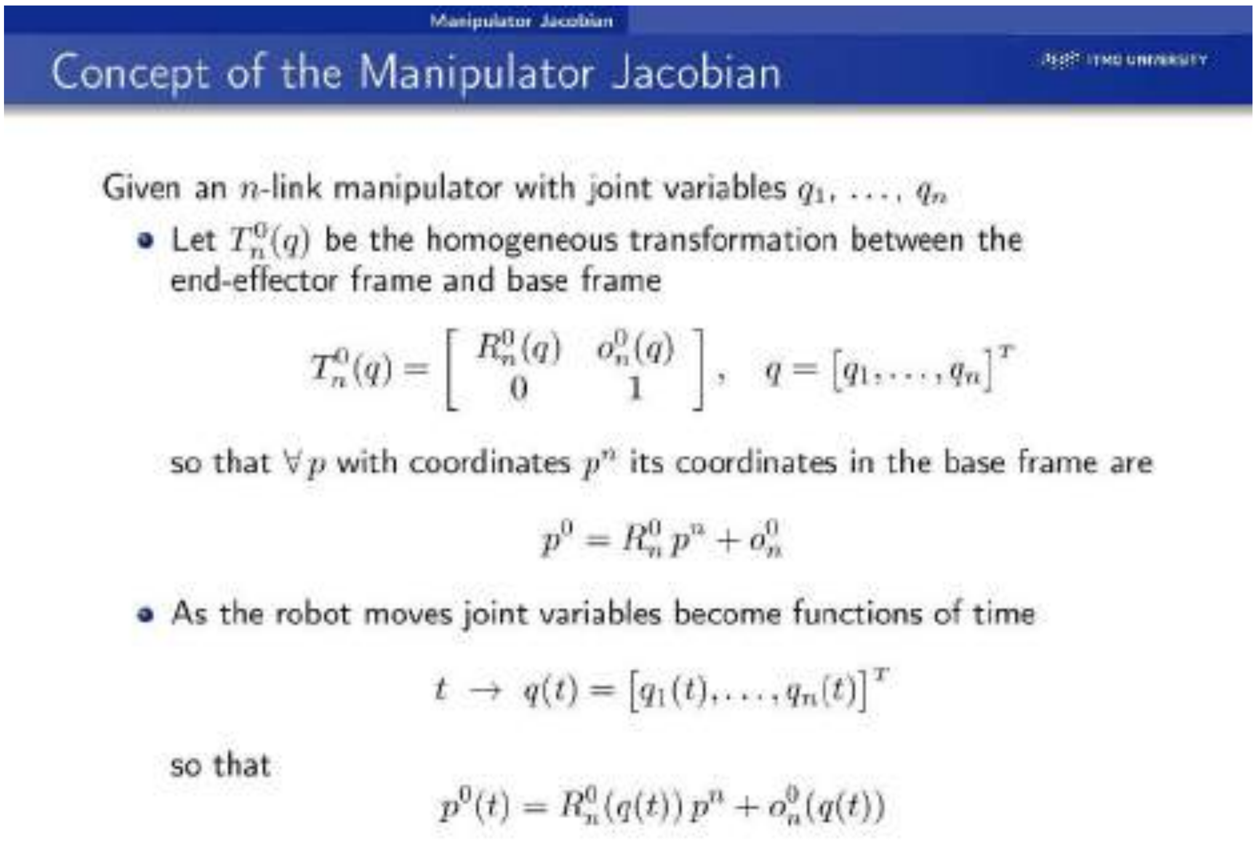
$$H^0_1(t) = \begin{bmatrix} R^0_1(t) & o^0_1(t) \\ 0 & 1 \end{bmatrix}$$

that is, coordinates of each point of the moving frame are

$$p^0(t) = R^0_1(t)p^1 + o^0_1(t)$$

Hence

$$\begin{aligned} \frac{d}{dt}p^0(t) &= \frac{d}{dt}\left[R^0_1(t)\right]p^1 + \frac{d}{dt}\left[o^0_1(t)\right] \\ &= S(\omega^0_1(t))R^0_1(t)p^1 + \frac{d}{dt}\left[o^0_1(t)\right] \\ &= \omega^0_1(t) \times \left[R^0_1(t)p^1\right] + \frac{d}{dt}\left[o^0_1(t)\right] \end{aligned}$$



Manipulator Jacobian

Concept of the Manipulator Jacobian

ITMO UNIVERSITY

Given an n-link manipulator with joint variables $q_1, \ldots, q_n$

- Let $T_n^0(q)$ be the homogeneous transformation between the end-effector frame and base frame

$$T_n^0(q) = \begin{bmatrix} R_n^0(q) & o_n^0(q) \\ 0 & 1 \end{bmatrix}, \quad q = \left[q_1, \ldots, q_n\right]^T$$

  so that $\forall p$ with coordinates p^n its coordinates in the base frame are

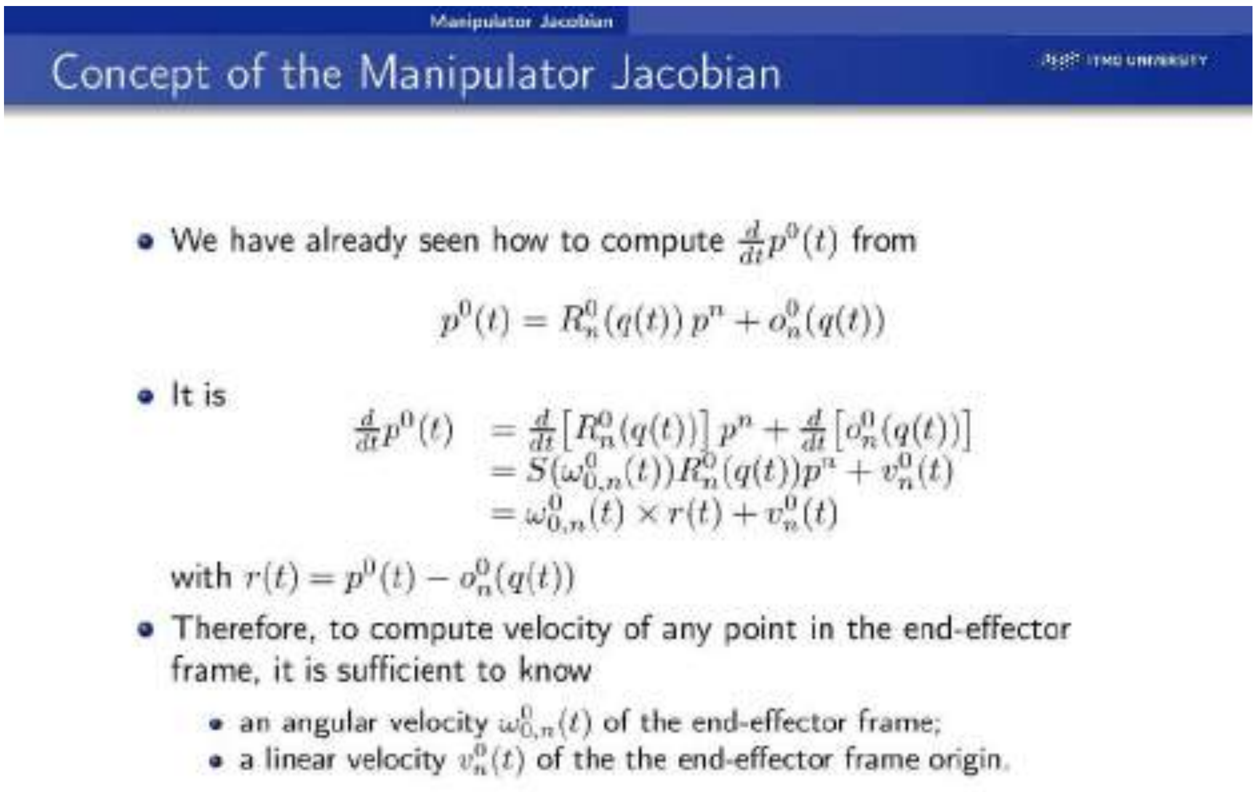
$$p^0 = R_n^0 \, p^n + o_n^0$$

- As the robot moves joint variables become functions of time

$$t \;\rightarrow\; q(t) = \left[q_1(t), \ldots, q_n(t)\right]^T$$

  so that

$$p^0(t) = R_n^0(q(t))\, p^n + o_n^0(q(t))$$



Manipulator Jacobian

Concept of the Manipulator Jacobian

ITMO UNIVERSITY

- We have already seen how to compute $\frac{d}{dt}p^0(t)$ from

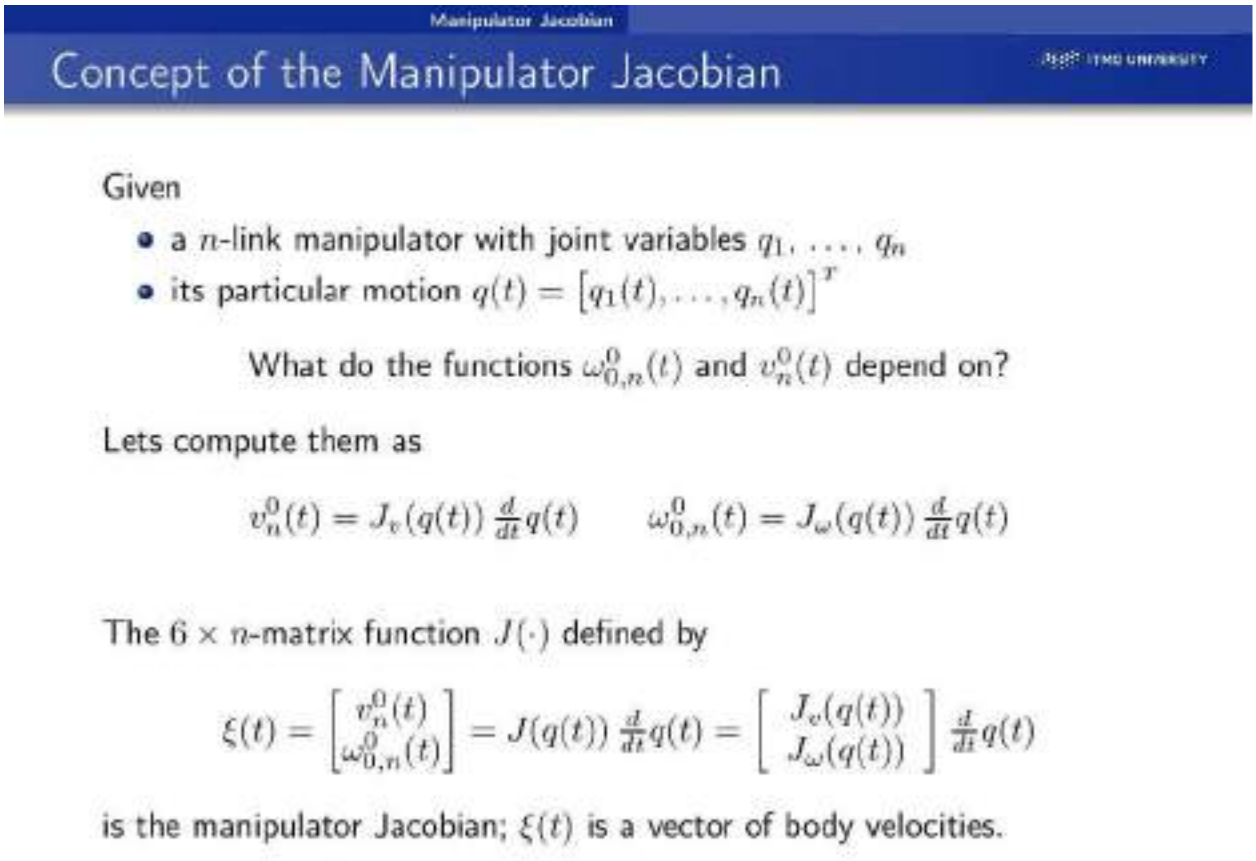
$$p^0(t) = R_n^0(q(t))\, p^n + o_n^0(q(t))$$

- It is

$$\begin{aligned} \tfrac{d}{dt}p^0(t) &= \tfrac{d}{dt}\left[R_n^0(q(t))\right] p^n + \tfrac{d}{dt}\left[o_n^0(q(t))\right] \\ &= S(\omega_{0,n}^0(t))R_n^0(q(t))p^n + v_n^0(t) \\ &= \omega_{0,n}^0(t) \times r(t) + v_n^0(t) \end{aligned}$$

  with $r(t) = p^0(t) - o_n^0(q(t))$

- Therefore, to compute velocity of any point in the end-effector frame, it is sufficient to know
  - an angular velocity $\omega_{0,n}^0(t)$ of the end-effector frame;
  - a linear velocity $v_n^0(t)$ of the the end-effector frame origin.



Manipulator Jacobian

Concept of the Manipulator Jacobian

ITMO UNIVERSITY

Given

- a n-link manipulator with joint variables $q_1, \dots, q_n$
- its particular motion $q(t) = \left[q_1(t), \dots, q_n(t)\right]^T$

What do the functions $\omega^0_{0,n}(t)$ and $v^0_n(t)$ depend on?

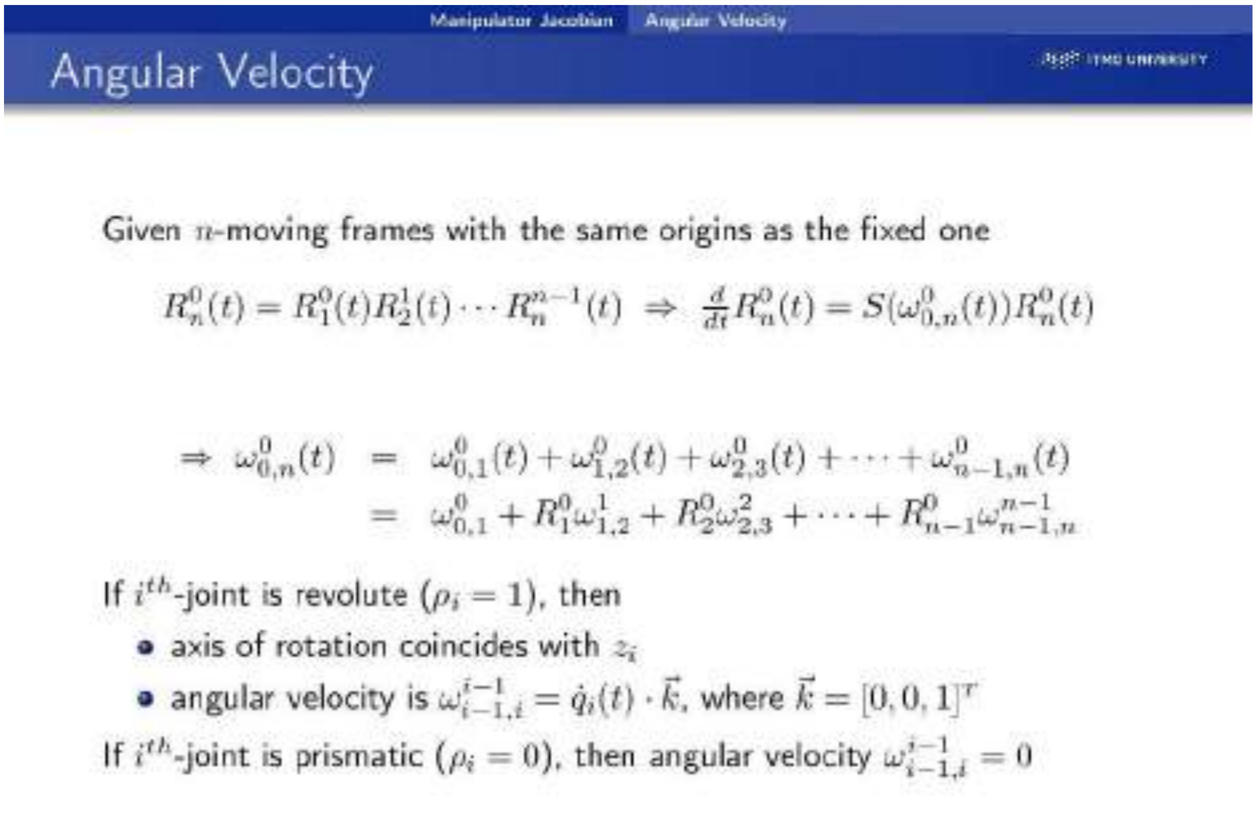
Lets compute them as

$$v^0_n(t) = J_v(q(t))\,\tfrac{d}{dt}q(t) \qquad \omega^0_{0,n}(t) = J_\omega(q(t))\,\tfrac{d}{dt}q(t)$$

The $6 \times n$-matrix function $J(\cdot)$ defined by

$$\xi(t) = \begin{bmatrix} v^0_n(t) \\ \omega^0_{0,n}(t) \end{bmatrix} = J(q(t))\,\tfrac{d}{dt}q(t) = \begin{bmatrix} J_v(q(t)) \\ J_\omega(q(t)) \end{bmatrix} \tfrac{d}{dt}q(t)$$

is the manipulator Jacobian; $\xi(t)$ is a vector of body velocities.



Manipulator Jacobian | Angular Velocity

Angular Velocity

ITMO UNIVERSITY

Given n-moving frames with the same origins as the fixed one

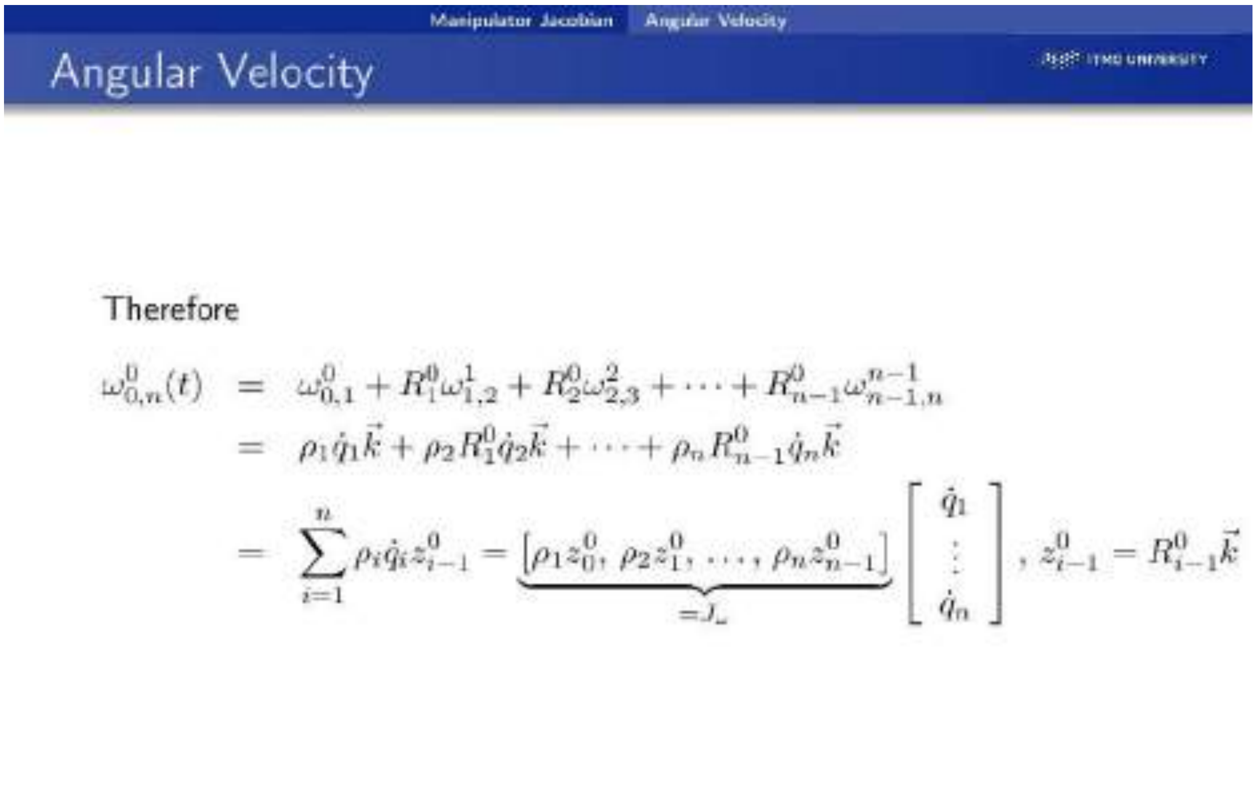
$$R^0_n(t) = R^0_1(t) R^1_2(t) \cdots R^{n-1}_n(t) \;\Rightarrow\; \tfrac{d}{dt}R^0_n(t) = S(\omega^0_{0,n}(t))R^0_n(t)$$

$$\begin{aligned} \Rightarrow\; \omega^0_{0,n}(t) &= \omega^0_{0,1}(t) + \omega^0_{1,2}(t) + \omega^0_{2,3}(t) + \cdots + \omega^0_{n-1,n}(t) \\ &= \omega^0_{0,1} + R^0_1\omega^1_{1,2} + R^0_2\omega^2_{2,3} + \cdots + R^0_{n-1}\omega^{n-1}_{n-1,n} \end{aligned}$$

If i^{th}-joint is revolute ($\rho_i = 1$), then

- axis of rotation coincides with z_i
- angular velocity is $\omega^{i-1}_{i-1,i} = \dot{q}_i(t) \cdot \vec{k}$, where $\vec{k} = [0, 0, 1]^T$

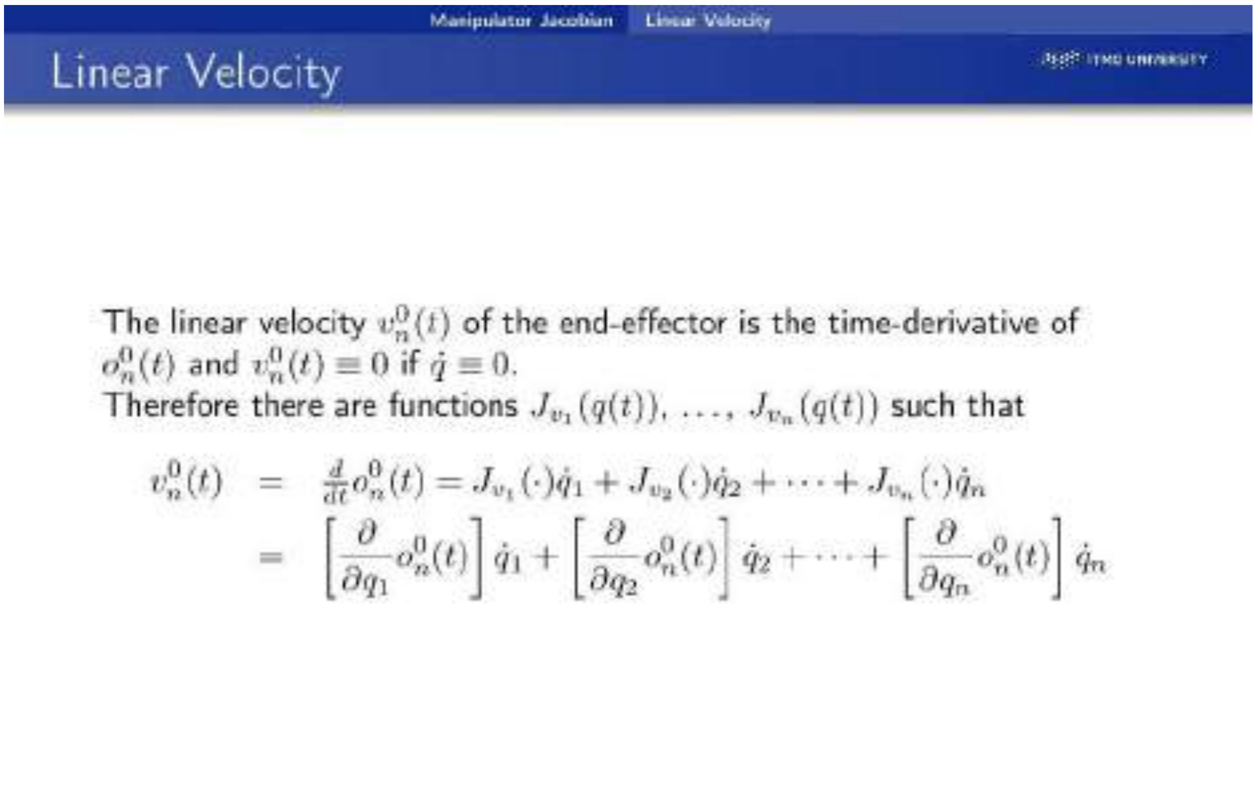
If i^{th}-joint is prismatic ($\rho_i = 0$), then angular velocity $\omega^{i-1}_{i-1,i} = 0$



Angular Velocity

Therefore

$$\begin{aligned}
\omega_{0,n}^{0}(t) &= \omega_{0,1}^{0} + R_1^0\omega_{1,2}^{1} + R_2^0\omega_{2,3}^{2} + \cdots + R_{n-1}^{0}\omega_{n-1,n}^{n-1} \\
&= \rho_1\dot{q}_1\vec{k} + \rho_2 R_1^0\dot{q}_2\vec{k} + \cdots + \rho_n R_{n-1}^{0}\dot{q}_n\vec{k} \\
&= \sum_{i=1}^{n}\rho_i\dot{q}_i z_{i-1}^{0} = \underbrace{\left[\rho_1 z_0^0,\ \rho_2 z_1^0,\ \ldots,\ \rho_n z_{n-1}^{0}\right]}_{=J_\omega}\begin{bmatrix}\dot{q}_1\\ \vdots\\ \dot{q}_n\end{bmatrix},\ z_{i-1}^{0} = R_{i-1}^{0}\vec{k}
\end{aligned}$$



Linear Velocity

The linear velocity $v_n^0(t)$ of the end-effector is the time-derivative of $o_n^0(t)$ and $v_n^0(t) \equiv 0$ if $\dot{q} \equiv 0$.
Therefore there are functions $J_{v_1}(q(t)),\ \ldots,\ J_{v_n}(q(t))$ such that

$$\begin{aligned}
v_n^0(t) &= \tfrac{d}{dt}o_n^0(t) = J_{v_1}(\cdot)\dot{q}_1 + J_{v_2}(\cdot)\dot{q}_2 + \cdots + J_{v_n}(\cdot)\dot{q}_n \\
&= \left[\frac{\partial}{\partial q_1}o_n^0(t)\right]\dot{q}_1 + \left[\frac{\partial}{\partial q_2}o_n^0(t)\right]\dot{q}_2 + \cdots + \left[\frac{\partial}{\partial q_n}o_n^0(t)\right]\dot{q}_n
\end{aligned}$$

Linear Velocity

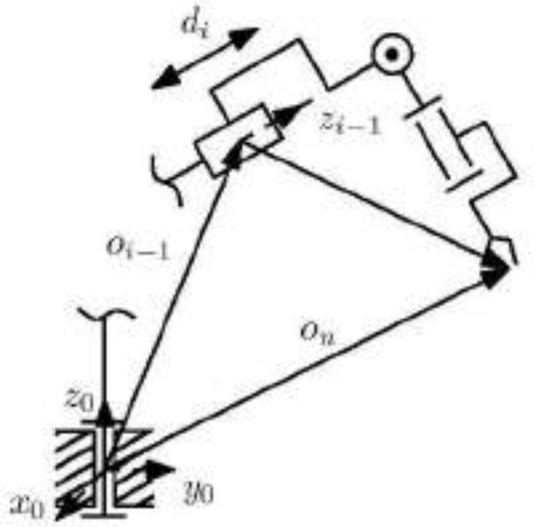


Fig.: Motion of the end effector due to prismatic joint i along the z_{i-1}-axis with velocity $\frac{d}{dt}d_i$, whereas all other joints are kept fixed (courtesy Spong).

$$\frac{d}{dt}o_n^0(t) = \frac{d}{dt}d_i(t)R_{i-1}^0\vec{k} = \frac{d}{dt}d_i(t)\,z_{i-1} = J_{v_i}\dot{d}_i$$

Linear Velocity

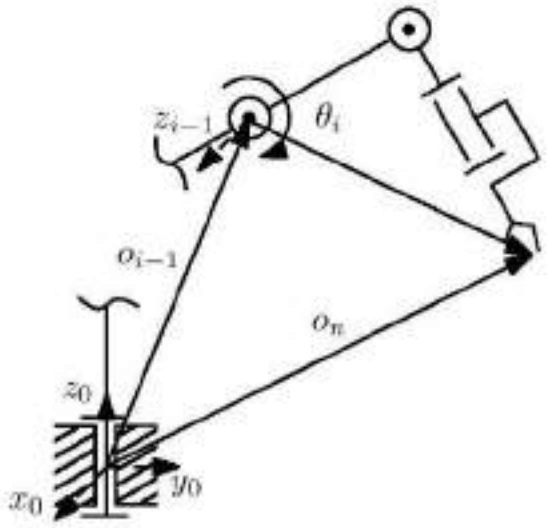
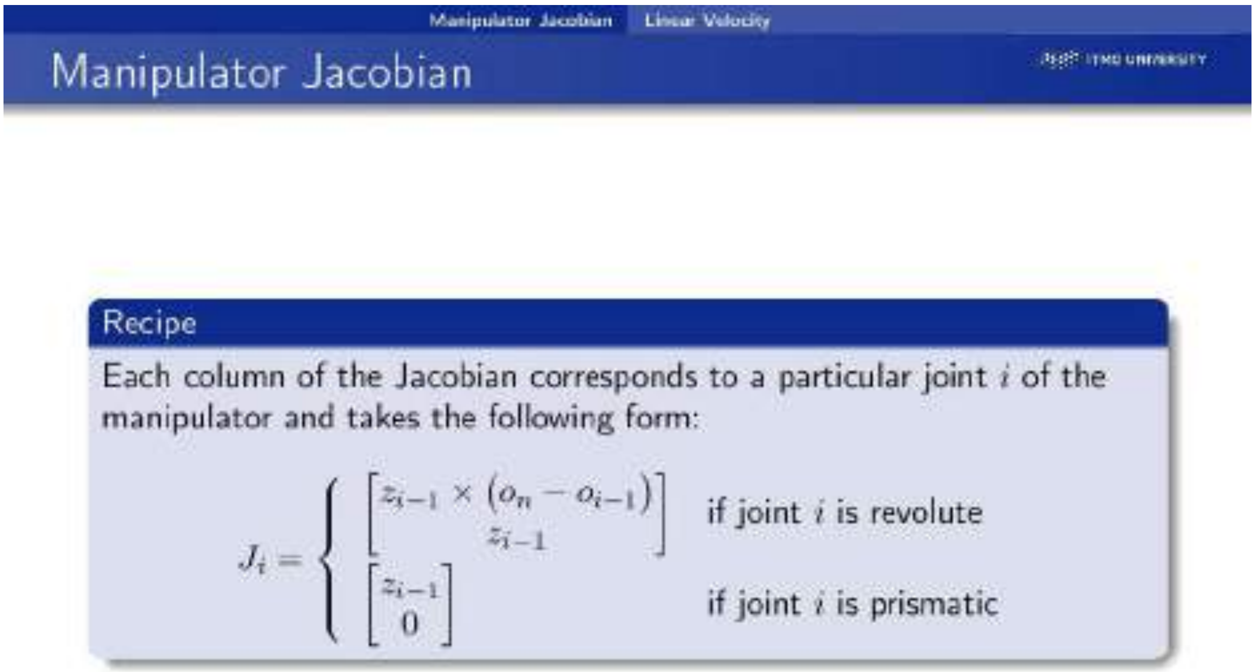


Fig.: Motion of the end effector due to revolute joint i about the z_{i-1}-axis with angular velocity $\frac{d}{dt}\theta_i z_{i-1}$, all other joints are kept fixed (courtesy Spong).

$$\frac{d}{dt}o_n^0(t) = R_{i-1}^0\left[\omega_{i-1,n}^{i-1}\times o_n^{i-1}\right] = \left[\dot{\theta}_i z_{i-1}\right]\times\left[o_n - o_{i-1}\right] = J_{v_i}\dot{\theta}_i$$

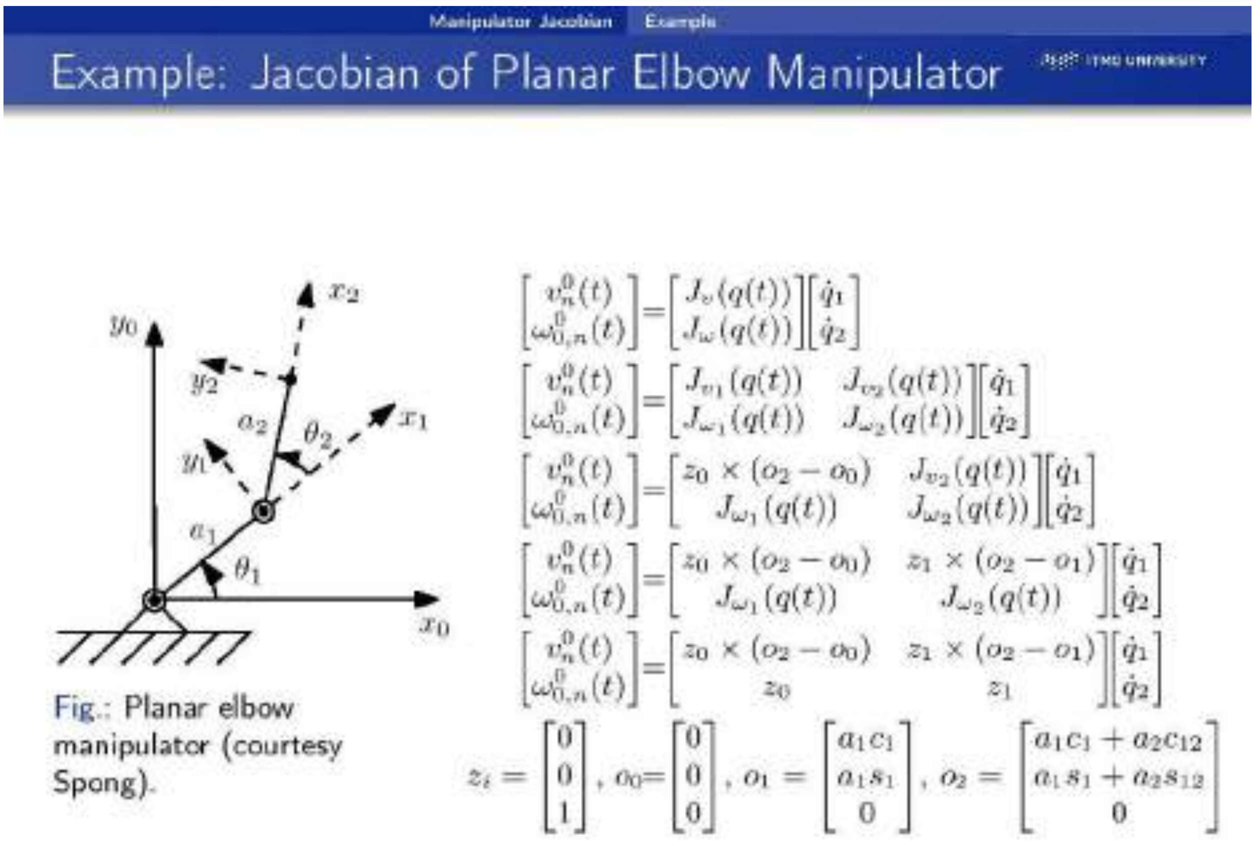


Manipulator Jacobian

Recipe

Each column of the Jacobian corresponds to a particular joint i of the manipulator and takes the following form:

$$J_i = \begin{cases} \begin{bmatrix} z_{i-1} \times (o_n - o_{i-1}) \\ z_{i-1} \end{bmatrix} & \text{if joint } i \text{ is revolute} \\ \begin{bmatrix} z_{i-1} \\ 0 \end{bmatrix} & \text{if joint } i \text{ is prismatic} \end{cases}$$



Example: Jacobian of Planar Elbow Manipulator

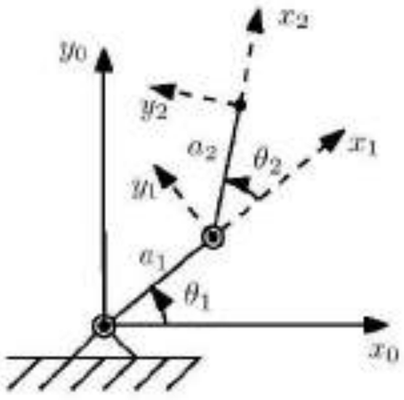


Fig.: Planar elbow manipulator (courtesy Spong).

$$\begin{bmatrix} v_n^0(t) \\ \omega_{0,n}^0(t) \end{bmatrix} = \begin{bmatrix} J_v(q(t)) \\ J_\omega(q(t)) \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}$$

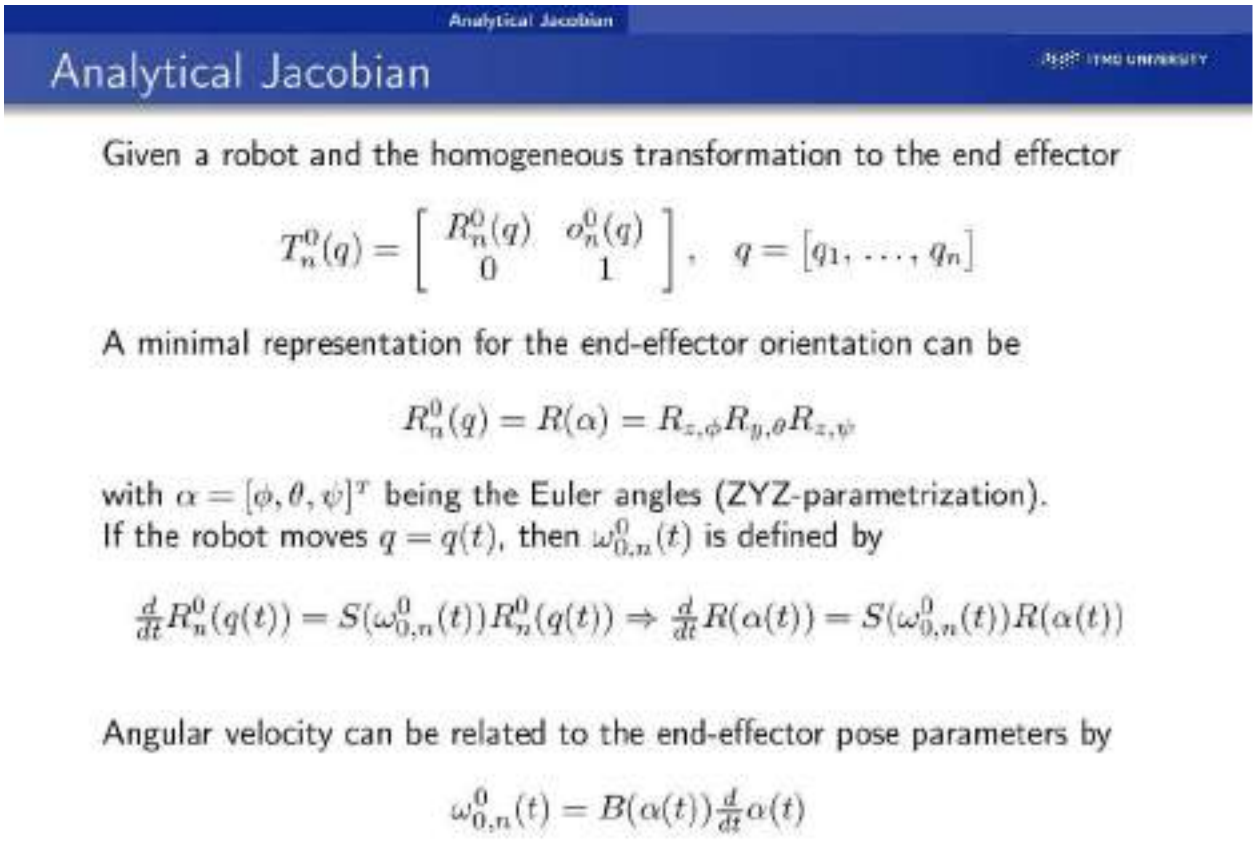
$$\begin{bmatrix} v_n^0(t) \\ \omega_{0,n}^0(t) \end{bmatrix} = \begin{bmatrix} J_{v_1}(q(t)) & J_{v_2}(q(t)) \\ J_{\omega_1}(q(t)) & J_{\omega_2}(q(t)) \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}$$

$$\begin{bmatrix} v_n^0(t) \\ \omega_{0,n}^0(t) \end{bmatrix} = \begin{bmatrix} z_0 \times (o_2 - o_0) & J_{v_2}(q(t)) \\ J_{\omega_1}(q(t)) & J_{\omega_2}(q(t)) \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}$$

$$\begin{bmatrix} v_n^0(t) \\ \omega_{0,n}^0(t) \end{bmatrix} = \begin{bmatrix} z_0 \times (o_2 - o_0) & z_1 \times (o_2 - o_1) \\ J_{\omega_1}(q(t)) & J_{\omega_2}(q(t)) \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}$$

$$\begin{bmatrix} v_n^0(t) \\ \omega_{0,n}^0(t) \end{bmatrix} = \begin{bmatrix} z_0 \times (o_2 - o_0) & z_1 \times (o_2 - o_1) \\ z_0 & z_1 \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}$$

$$z_i = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \ o_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \ o_1 = \begin{bmatrix} a_1 c_1 \\ a_1 s_1 \\ 0 \end{bmatrix}, \ o_2 = \begin{bmatrix} a_1 c_1 + a_2 c_{12} \\ a_1 s_1 + a_2 s_{12} \\ 0 \end{bmatrix}$$



Analytical Jacobian

Given a robot and the homogeneous transformation to the end effector

$$T_n^0(q) = \begin{bmatrix} R_n^0(q) & o_n^0(q) \\ 0 & 1 \end{bmatrix}, \quad q = [q_1, \dots, q_n]$$

A minimal representation for the end-effector orientation can be

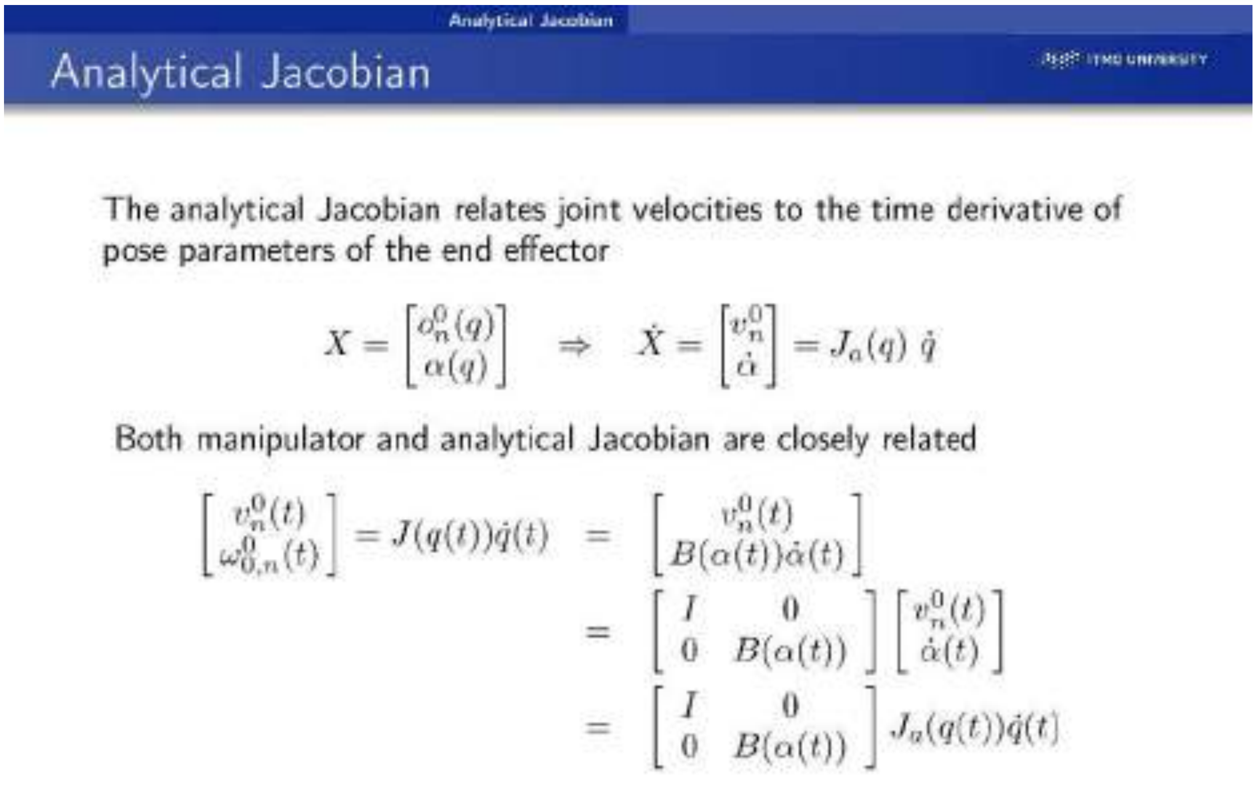
$$R_n^0(q) = R(\alpha) = R_{z,\phi} R_{y,\theta} R_{z,\psi}$$

with $\alpha = [\phi, \theta, \psi]^T$ being the Euler angles (ZYZ-parametrization).
If the robot moves $q = q(t)$, then $\omega_{0,n}^0(t)$ is defined by

$$\tfrac{d}{dt} R_n^0(q(t)) = S(\omega_{0,n}^0(t)) R_n^0(q(t)) \Rightarrow \tfrac{d}{dt} R(\alpha(t)) = S(\omega_{0,n}^0(t)) R(\alpha(t))$$

Angular velocity can be related to the end-effector pose parameters by

$$\omega_{0,n}^0(t) = B(\alpha(t)) \tfrac{d}{dt} \alpha(t)$$



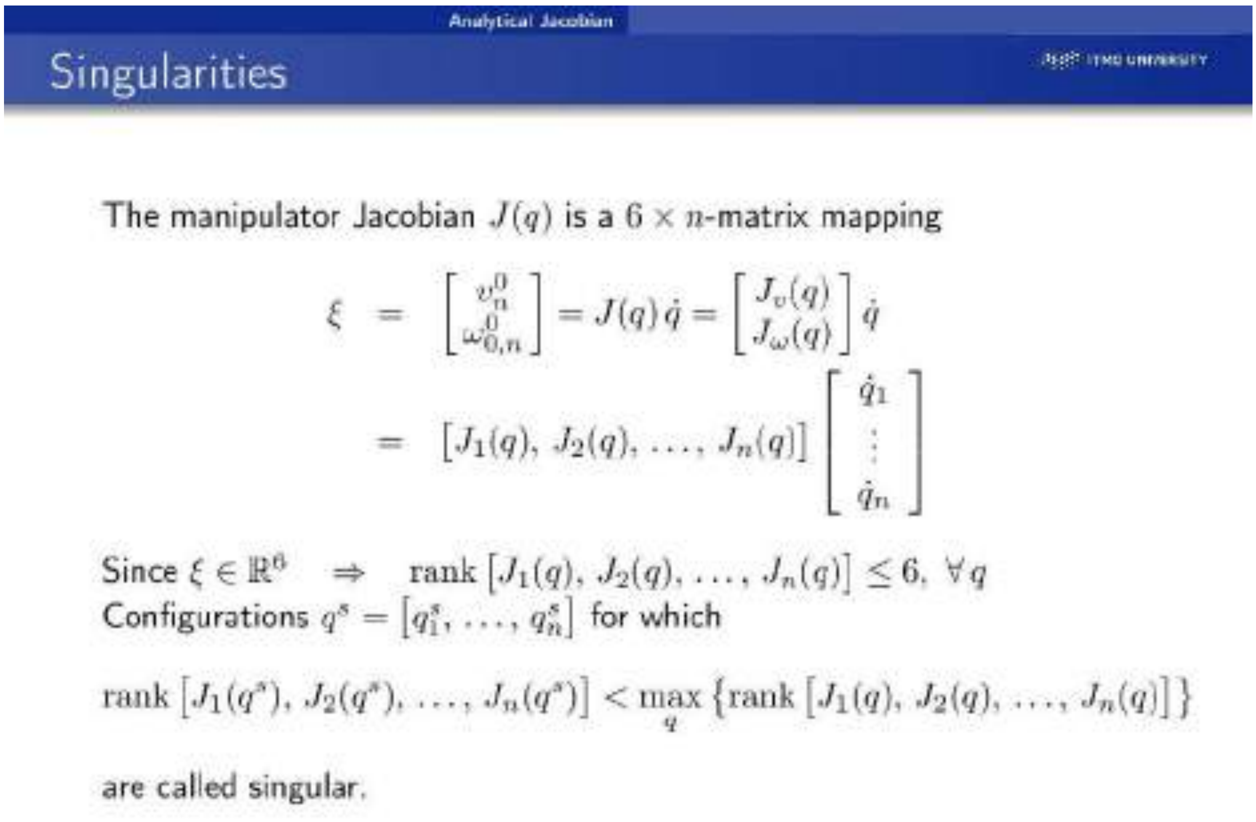
Analytical Jacobian

The analytical Jacobian relates joint velocities to the time derivative of pose parameters of the end effector

$$X = \begin{bmatrix} o_n^0(q) \\ \alpha(q) \end{bmatrix} \quad \Rightarrow \quad \dot{X} = \begin{bmatrix} v_n^0 \\ \dot{\alpha} \end{bmatrix} = J_a(q)\, \dot{q}$$

Both manipulator and analytical Jacobian are closely related

$$\begin{aligned} \begin{bmatrix} v_n^0(t) \\ \omega_{0,n}^0(t) \end{bmatrix} = J(q(t))\dot{q}(t) \quad &= \begin{bmatrix} v_n^0(t) \\ B(\alpha(t))\dot{\alpha}(t) \end{bmatrix} \\ &= \begin{bmatrix} I & 0 \\ 0 & B(\alpha(t)) \end{bmatrix} \begin{bmatrix} v_n^0(t) \\ \dot{\alpha}(t) \end{bmatrix} \\ &= \begin{bmatrix} I & 0 \\ 0 & B(\alpha(t)) \end{bmatrix} J_a(q(t))\dot{q}(t) \end{aligned}$$



Analytical Jacobian

Singularities

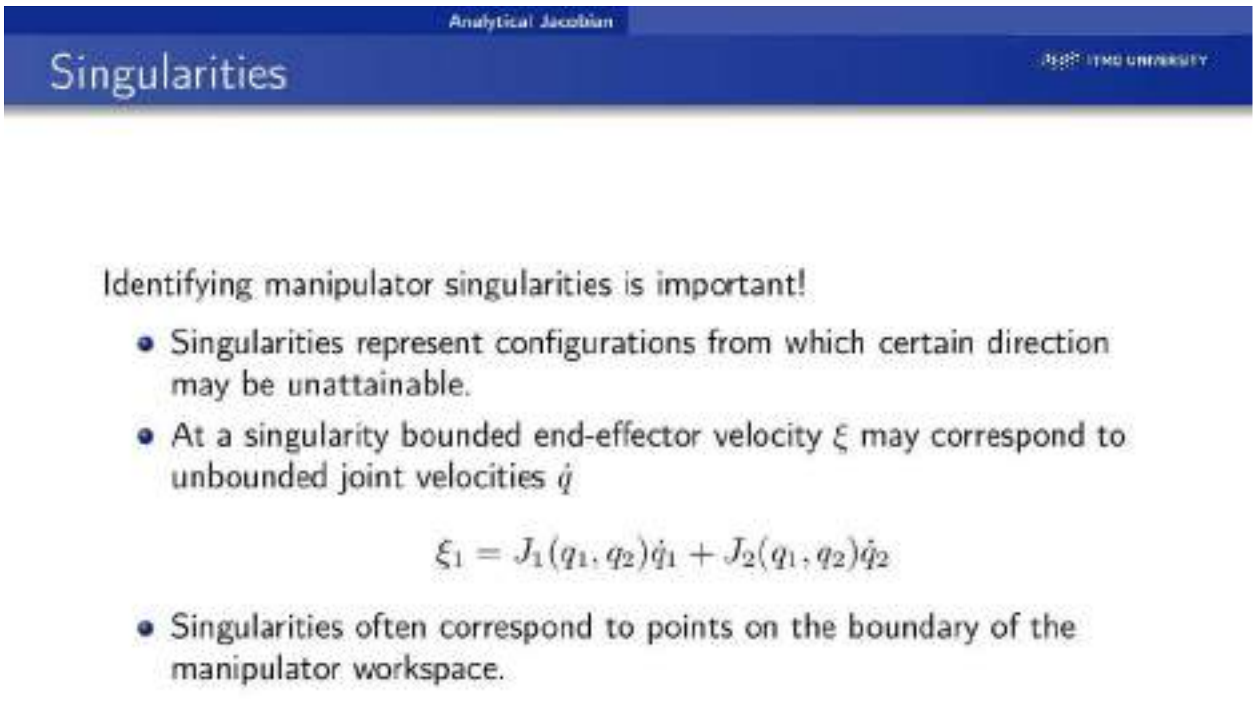
The manipulator Jacobian $J(q)$ is a $6 \times n$-matrix mapping

$$\begin{aligned} \xi &= \begin{bmatrix} v_n^0 \\ \omega_{0,n}^0 \end{bmatrix} = J(q)\,\dot q = \begin{bmatrix} J_v(q) \\ J_\omega(q) \end{bmatrix} \dot q \\ &= \left[J_1(q),\, J_2(q),\, \dots,\, J_n(q)\right] \begin{bmatrix} \dot q_1 \\ \vdots \\ \dot q_n \end{bmatrix} \end{aligned}$$

Since $\xi \in \mathbb{R}^6 \quad \Rightarrow \quad \operatorname{rank}\left[J_1(q),\, J_2(q),\, \dots,\, J_n(q)\right] \le 6,\ \forall q$
Configurations $q^s = \left[q_1^s,\, \dots,\, q_n^s\right]$ for which

$$\operatorname{rank}\left[J_1(q^s),\, J_2(q^s),\, \dots,\, J_n(q^s)\right] < \max_q \left\{\operatorname{rank}\left[J_1(q),\, J_2(q),\, \dots,\, J_n(q)\right]\right\}$$

are called singular.



Analytical Jacobian

Singularities

Identifying manipulator singularities is important!

- Singularities represent configurations from which certain direction may be unattainable.
- At a singularity bounded end-effector velocity ξ may correspond to unbounded joint velocities $\dot q$

$$\xi_1 = J_1(q_1, q_2)\dot q_1 + J_2(q_1, q_2)\dot q_2$$

- Singularities often correspond to points on the boundary of the manipulator workspace.

Analytical Jacobian

Example: Singularities of Elbow Manipulator

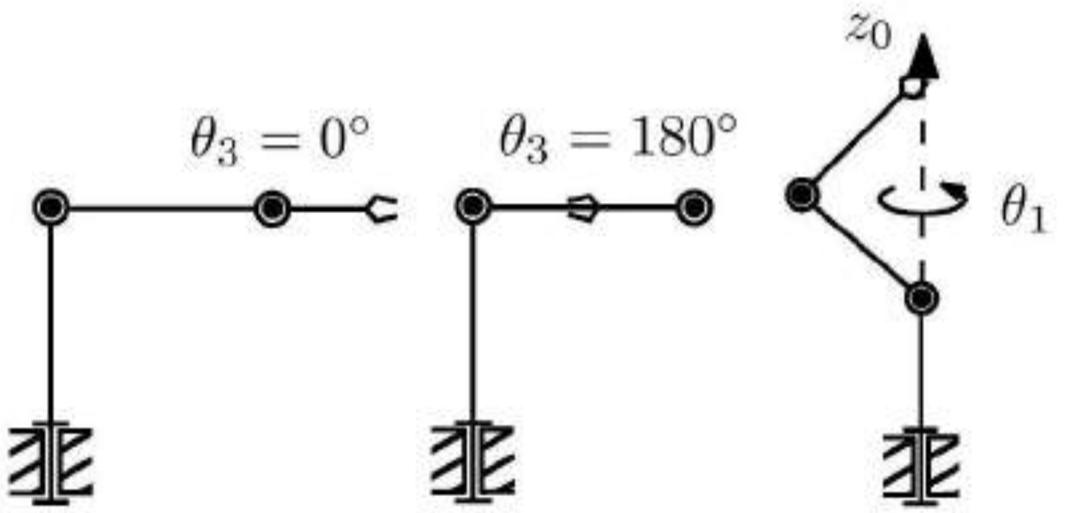


Fig.: Singular configurations of the planar elbow manipulator (courtesy Spong).

Inverse Velocity and Manipulability

Assigning Joint Velocities

The manipulator Jacobian $J(q)$ a $6 \times n$-matrix mapping

$$\xi = \begin{bmatrix} v_n^0 \\ \omega_{0,n}^0 \end{bmatrix} = J(q)\,\dot{q} = [J_1(q), J_2(q), \ldots, J_n(q)] \begin{bmatrix} \dot{q}_1 \\ \vdots \\ \dot{q}_n \end{bmatrix}$$

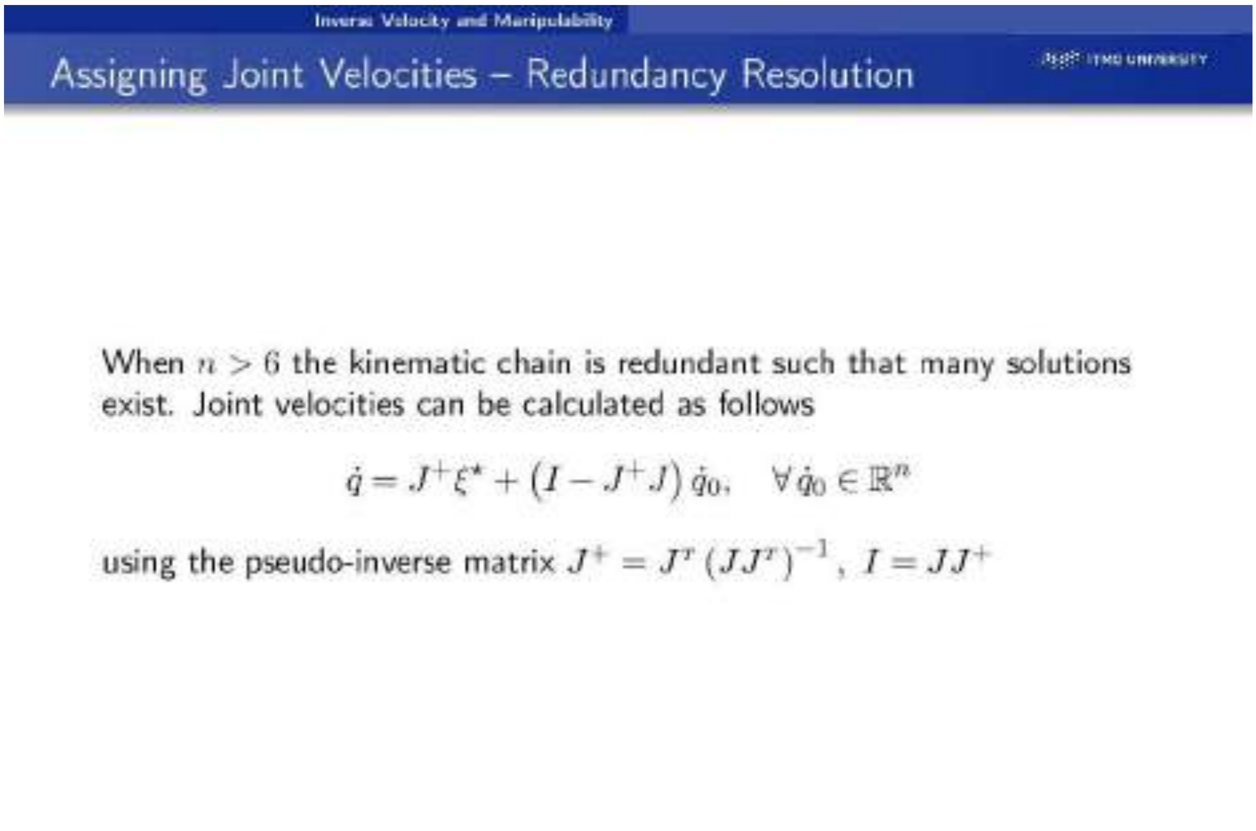
If the Jacobian is square and full rank, we can invert it

$$\dot{q} = J^{-1}(q)\xi^*$$

Hence, the joint angles might be assigned by integration

$$q(t) = \int_0^t J^{-1}(q(\tau))\xi^*(\tau)d\tau + q(0)$$

or in simple discrete form: $q(t_{k+1}) = q(t_k) + J^{-1}(q(t_k))\xi^*(t_k)\Delta t$.



Inverse Velocity and Manipulability

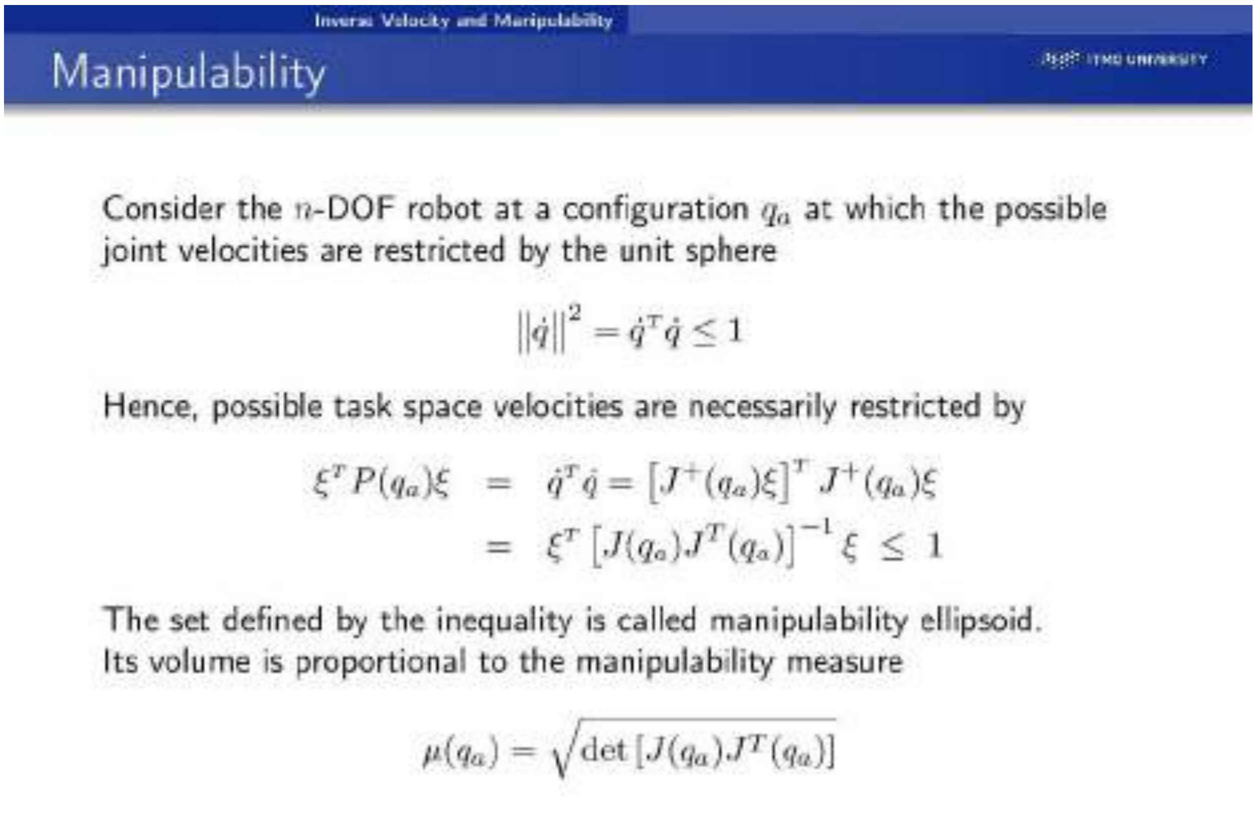
Assigning Joint Velocities – Redundancy Resolution

ITMO UNIVERSITY

When $n > 6$ the kinematic chain is redundant such that many solutions exist. Joint velocities can be calculated as follows

$$\dot{q} = J^{+}\xi^{\star} + \left(I - J^{+}J\right)\dot{q}_0, \quad \forall \dot{q}_0 \in \mathbb{R}^n$$

using the pseudo-inverse matrix $J^{+} = J^{T}\left(JJ^{T}\right)^{-1}$, $I = JJ^{+}$



Inverse Velocity and Manipulability

Manipulability

ITMO UNIVERSITY

Consider the n-DOF robot at a configuration q_a at which the possible joint velocities are restricted by the unit sphere

$$\left\|\dot{q}\right\|^2 = \dot{q}^T\dot{q} \leq 1$$

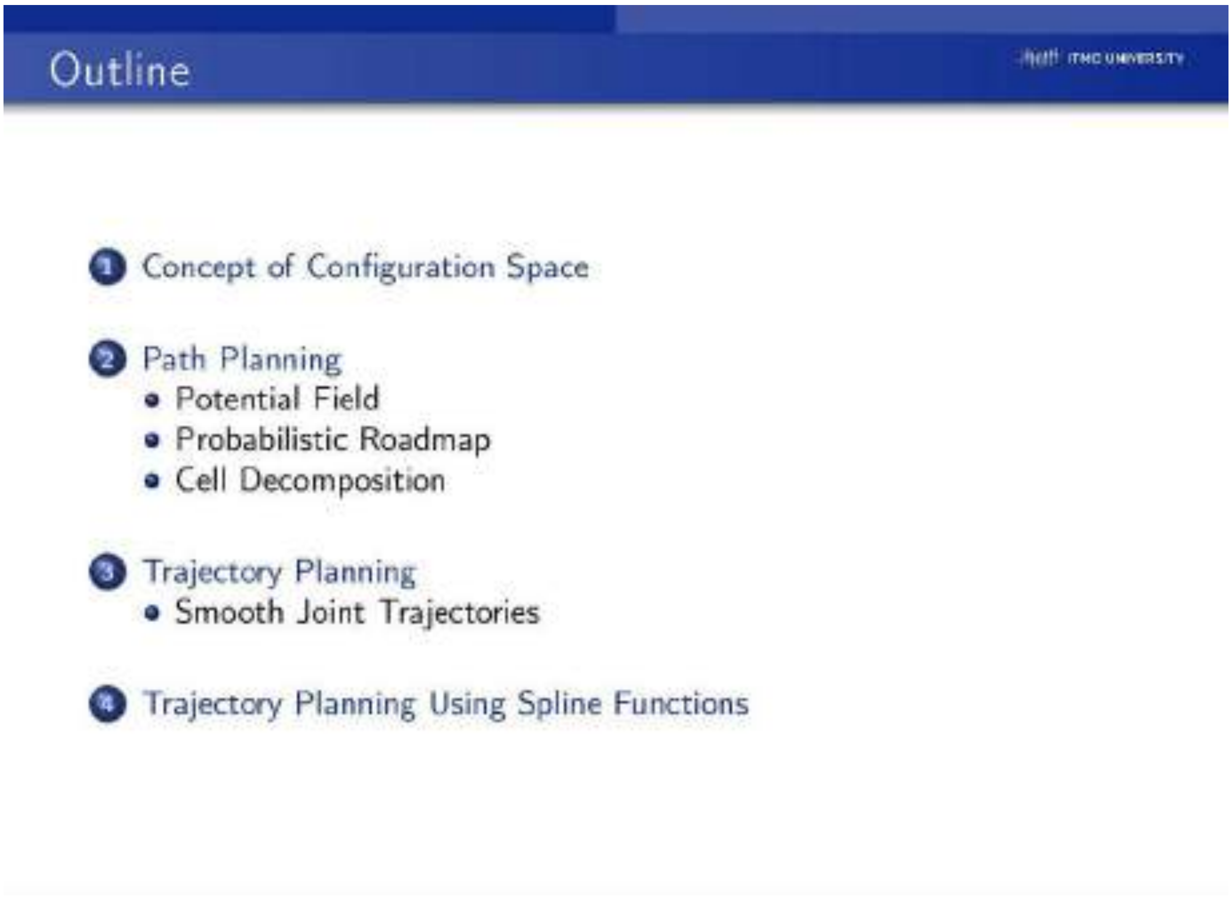
Hence, possible task space velocities are necessarily restricted by

$$\begin{aligned} \xi^T P(q_a)\xi &= \dot{q}^T\dot{q} = \left[J^{+}(q_a)\xi\right]^T J^{+}(q_a)\xi \\ &= \xi^T\left[J(q_a)J^T(q_a)\right]^{-1}\xi \leq 1 \end{aligned}$$

The set defined by the inequality is called manipulability ellipsoid. Its volume is proportional to the manipulability measure

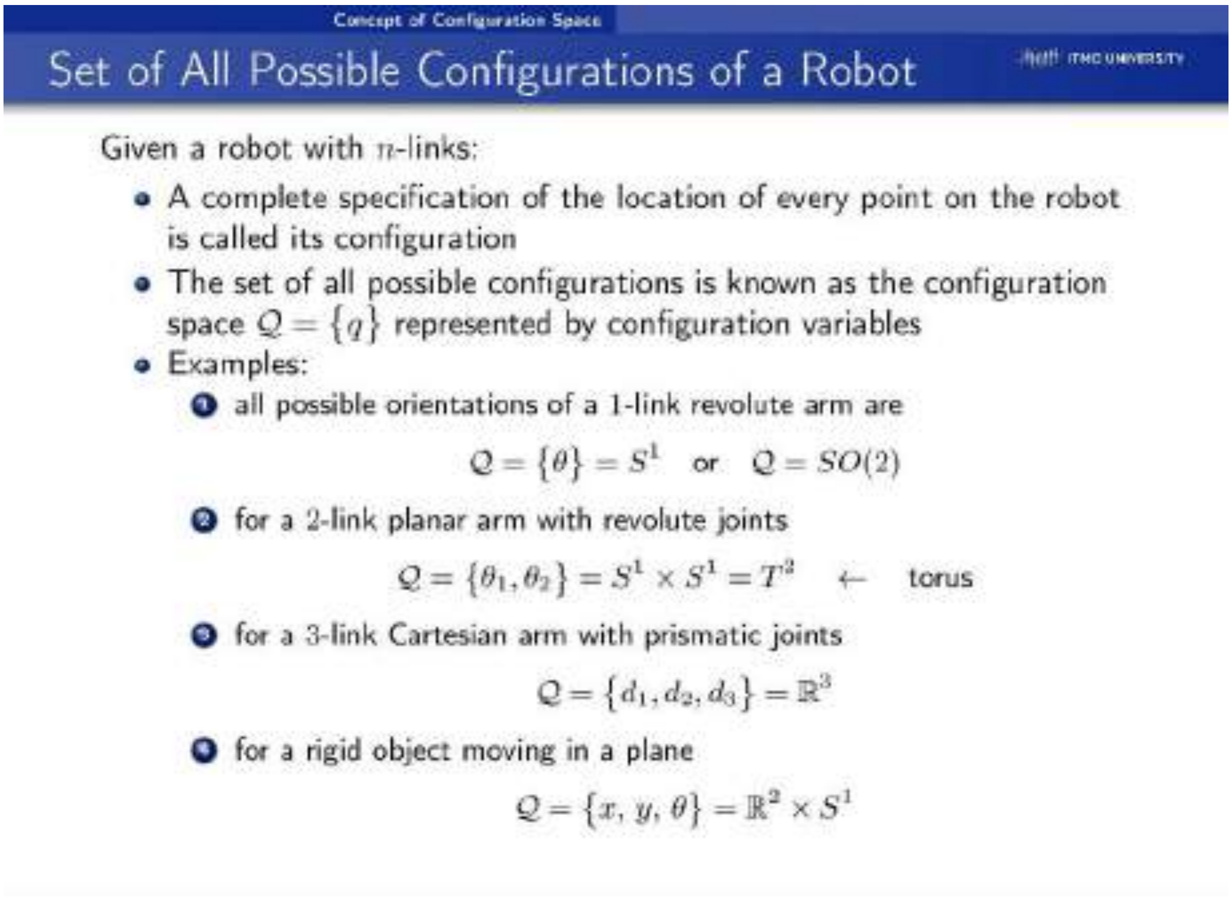
$$\mu(q_a) = \sqrt{\det\left[J(q_a)J^T(q_a)\right]}$$

Path and Trajectory Planning



Outline

ITMO UNIVERSITY



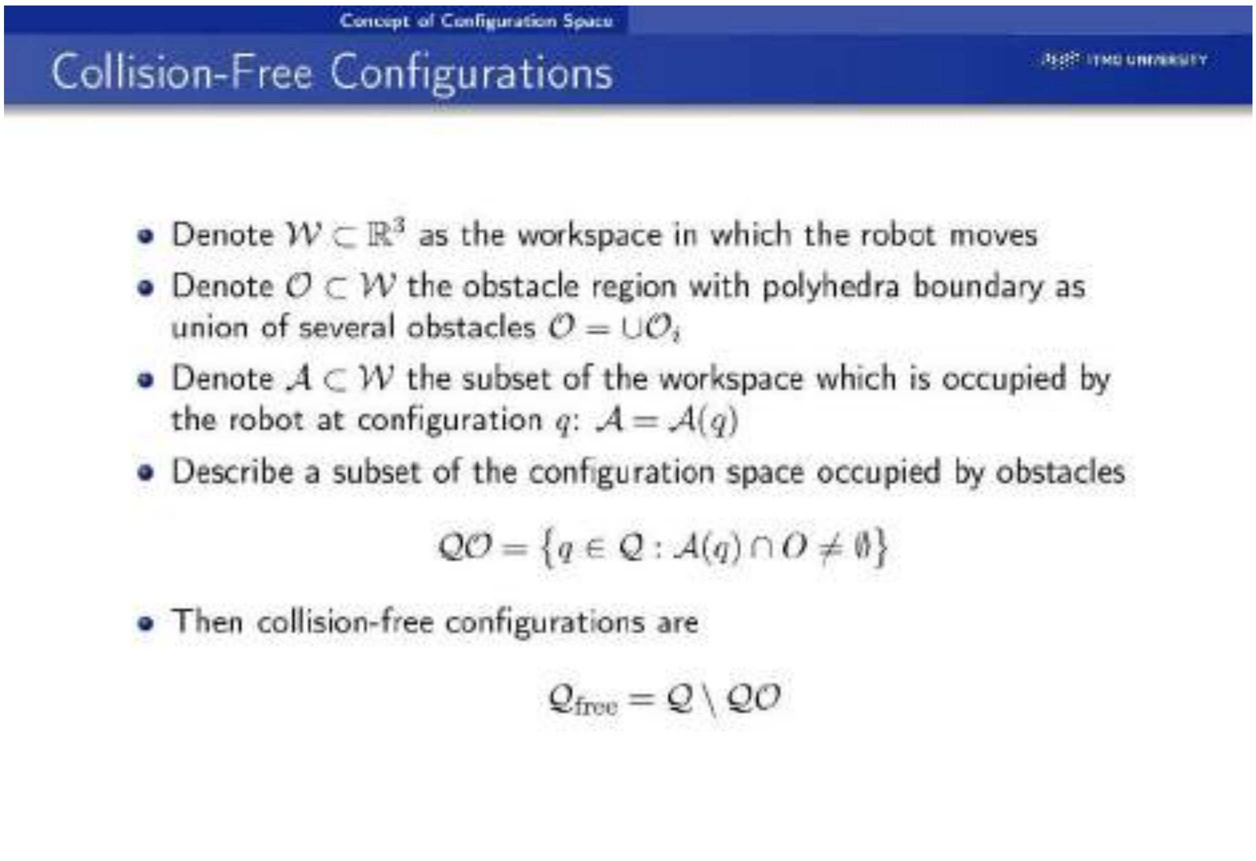
Concept of Configuration Space

Set of All Possible Configurations of a Robot

ITMO UNIVERSITY

Given a robot with n-links:

- A complete specification of the location of every point on the robot is called its configuration
- The set of all possible configurations is known as the configuration space $\mathcal{Q} = \{q\}$ represented by configuration variables
- Examples:
  1. all possible orientations of a 1-link revolute arm are
  $$\mathcal{Q} = \{\theta\} = S^1 \quad \text{or} \quad \mathcal{Q} = SO(2)$$
  2. for a 2-link planar arm with revolute joints
  $$\mathcal{Q} = \{\theta_1, \theta_2\} = S^1 \times S^1 = T^2 \quad \leftarrow \quad \text{torus}$$
  3. for a 3-link Cartesian arm with prismatic joints
  $$\mathcal{Q} = \{d_1, d_2, d_3\} = \mathbb{R}^3$$
  4. for a rigid object moving in a plane
  $$\mathcal{Q} = \{x, y, \theta\} = \mathbb{R}^2 \times S^1$$



Concept of Configuration Space

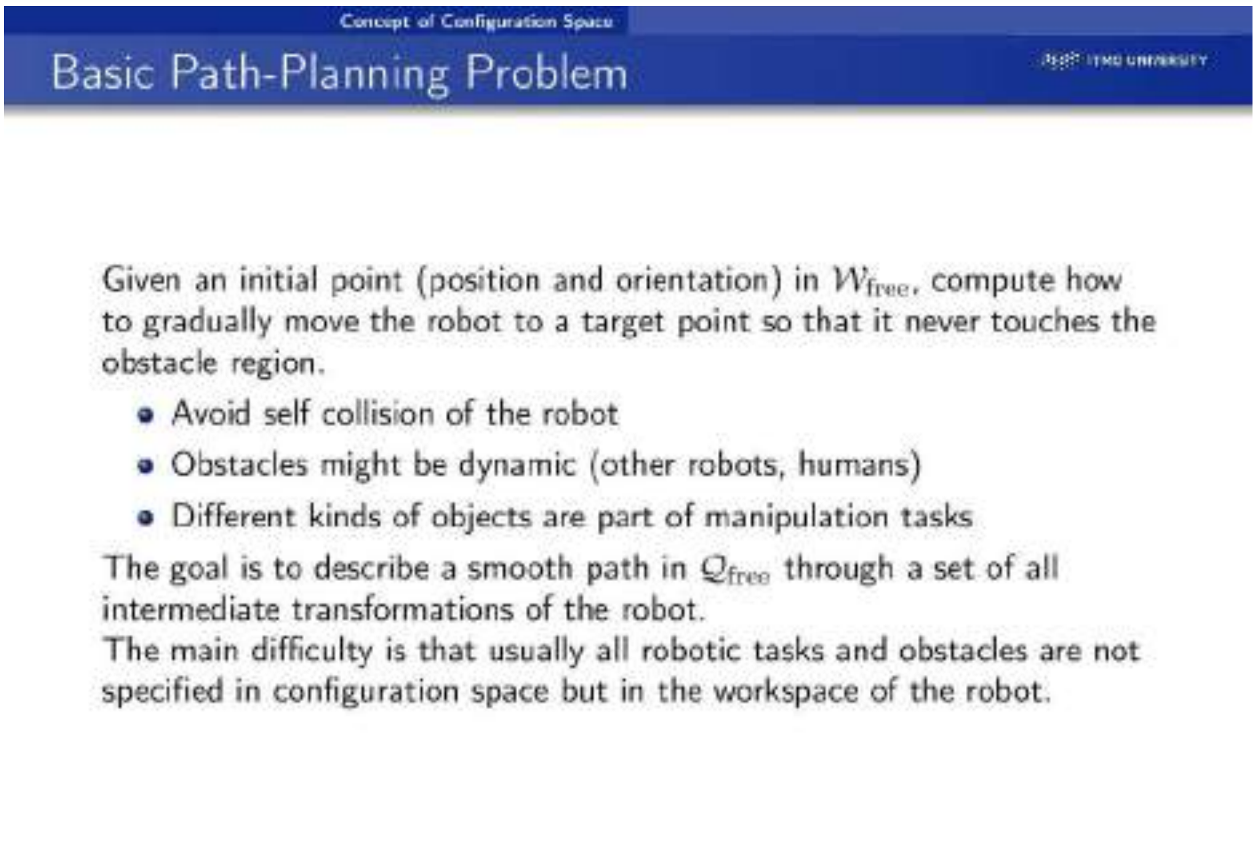
Collision-Free Configurations

- Denote $\mathcal{W} \subset \mathbb{R}^3$ as the workspace in which the robot moves
- Denote $\mathcal{O} \subset \mathcal{W}$ the obstacle region with polyhedra boundary as union of several obstacles $\mathcal{O} = \cup \mathcal{O}_i$
- Denote $\mathcal{A} \subset \mathcal{W}$ the subset of the workspace which is occupied by the robot at configuration q: $\mathcal{A} = \mathcal{A}(q)$
- Describe a subset of the configuration space occupied by obstacles

$$\mathcal{QO} = \{ q \in \mathcal{Q} : \mathcal{A}(q) \cap \mathcal{O} \neq \emptyset \}$$

- Then collision-free configurations are

$$\mathcal{Q}_{\text{free}} = \mathcal{Q} \setminus \mathcal{QO}$$



Concept of Configuration Space

Basic Path-Planning Problem

Given an initial point (position and orientation) in $\mathcal{W}_{\text{free}}$, compute how to gradually move the robot to a target point so that it never touches the obstacle region.

- Avoid self collision of the robot
- Obstacles might be dynamic (other robots, humans)
- Different kinds of objects are part of manipulation tasks

The goal is to describe a smooth path in $\mathcal{Q}_{\text{free}}$ through a set of all intermediate transformations of the robot.
The main difficulty is that usually all robotic tasks and obstacles are not specified in configuration space but in the workspace of the robot.

Concept of Configuration Space

Example: End Effector that Translates in the Plane

ITMO UNIVERSITY

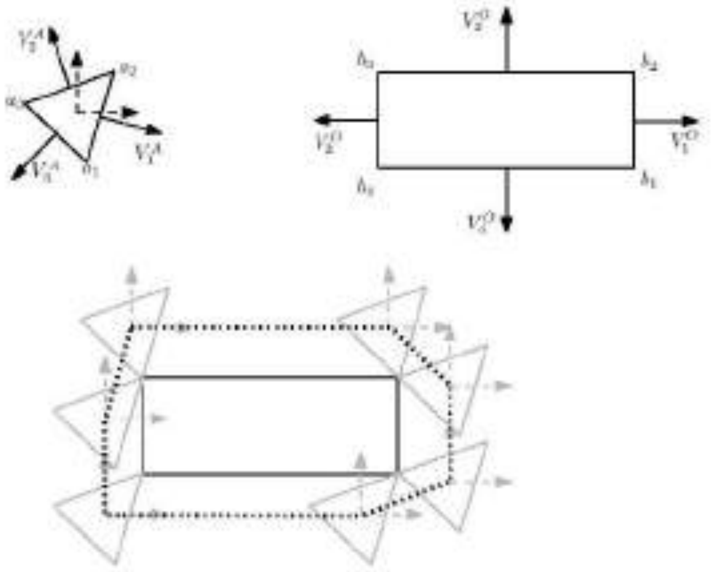


Fig.: (**a**) The triangle-shaped robot end effector moves in a 2-D workspace that contains a rectangular obstacle; (**b**) the boundary of the configuration space obstacle region $\mathcal{QO}$ is the dashed convex hull

Concept of Configuration Space

Example: Two-Link Planar Arm

ITMO UNIVERSITY

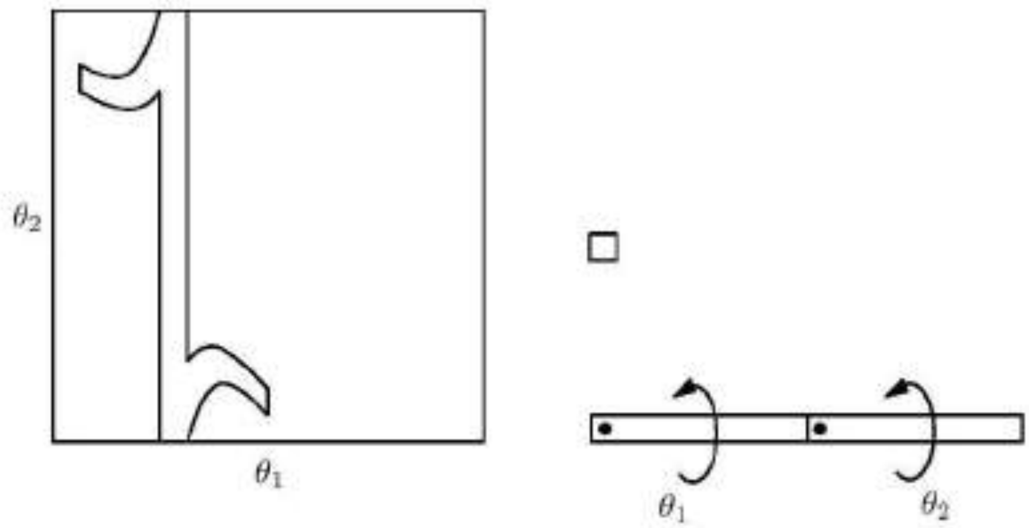
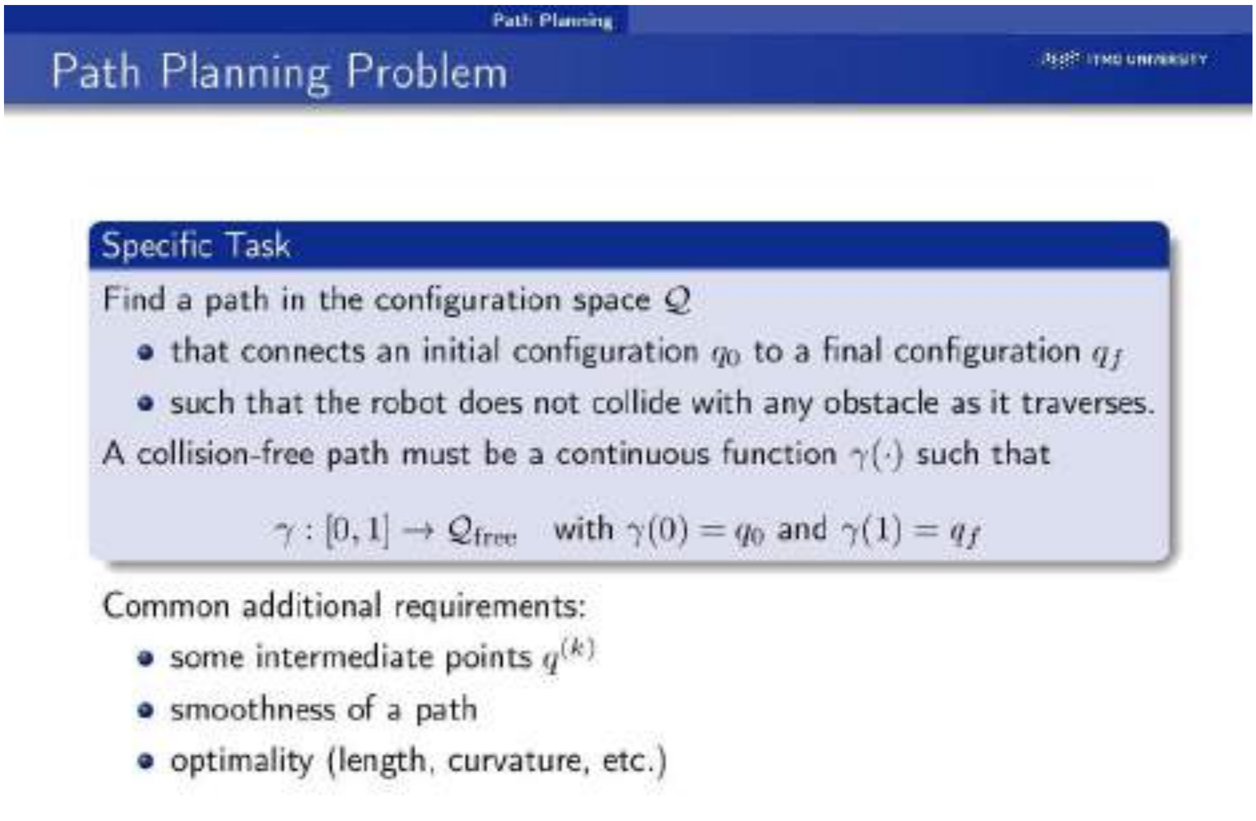


Fig.: (**a**) The two-link robot moves in a 2-D workspace that contains a small obstacle; (**b**) the configuration space obstacle region $\mathcal{QO}$ is gray

The computation of $\mathcal{QO}$ is more difficult for robots with revolute joints.



Path Planning Problem

Specific Task

Find a path in the configuration space $\mathcal{Q}$

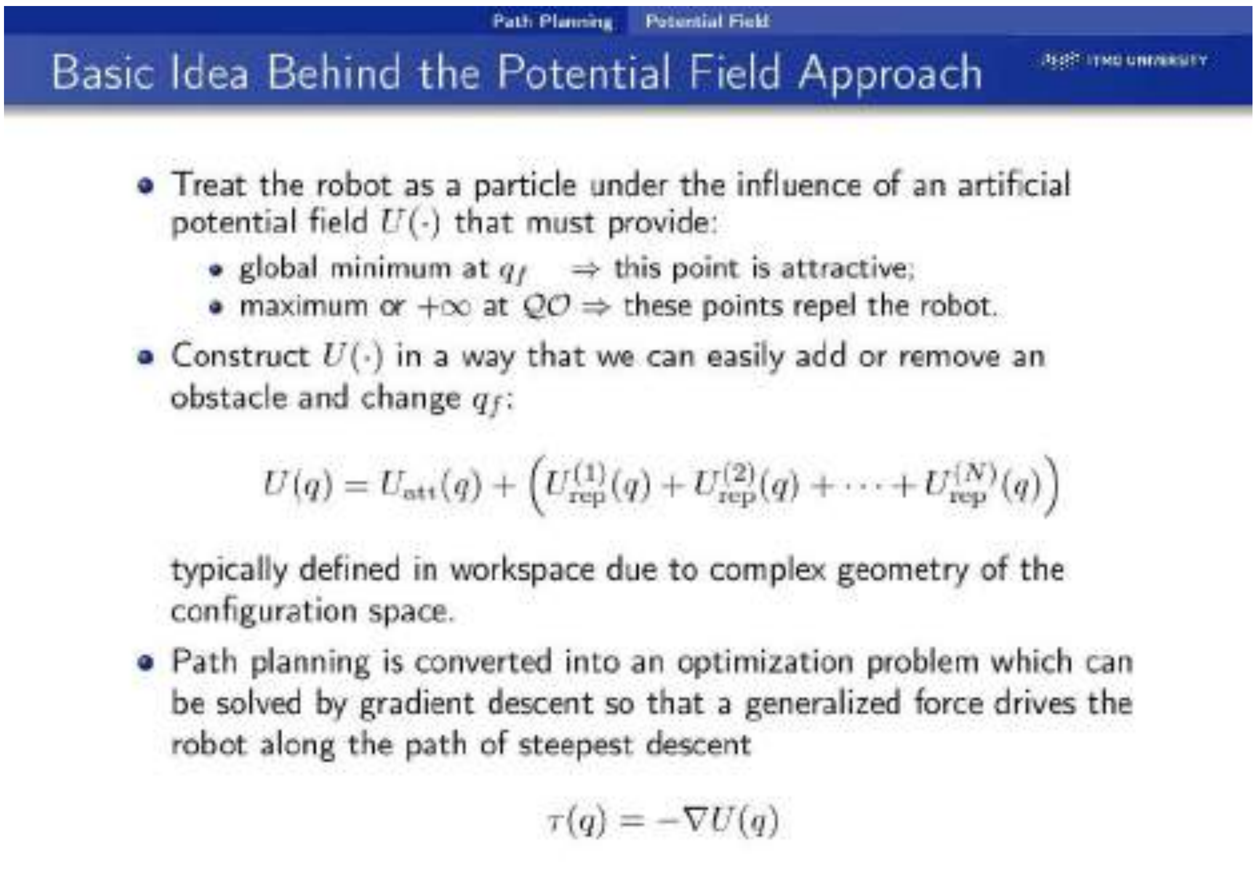
- that connects an initial configuration q_0 to a final configuration q_f
- such that the robot does not collide with any obstacle as it traverses.

A collision-free path must be a continuous function $\gamma(\cdot)$ such that

$$\gamma : [0,1] \to \mathcal{Q}_{\text{free}} \quad \text{with } \gamma(0) = q_0 \text{ and } \gamma(1) = q_f$$

Common additional requirements:

- some intermediate points $q^{(k)}$
- smoothness of a path
- optimality (length, curvature, etc.)



Basic Idea Behind the Potential Field Approach

- Treat the robot as a particle under the influence of an artificial potential field $U(\cdot)$ that must provide:
  - global minimum at q_f $\Rightarrow$ this point is attractive;
  - maximum or $+\infty$ at $\mathcal{QO} \Rightarrow$ these points repel the robot.
- Construct $U(\cdot)$ in a way that we can easily add or remove an obstacle and change q_f:

$$U(q) = U_{\text{att}}(q) + \left(U_{\text{rep}}^{(1)}(q) + U_{\text{rep}}^{(2)}(q) + \cdots + U_{\text{rep}}^{(N)}(q)\right)$$

  typically defined in workspace due to complex geometry of the configuration space.
- Path planning is converted into an optimization problem which can be solved by gradient descent so that a generalized force drives the robot along the path of steepest descent

$$\tau(q) = -\nabla U(q)$$

Example: Potential Field for a Particle in the Plane

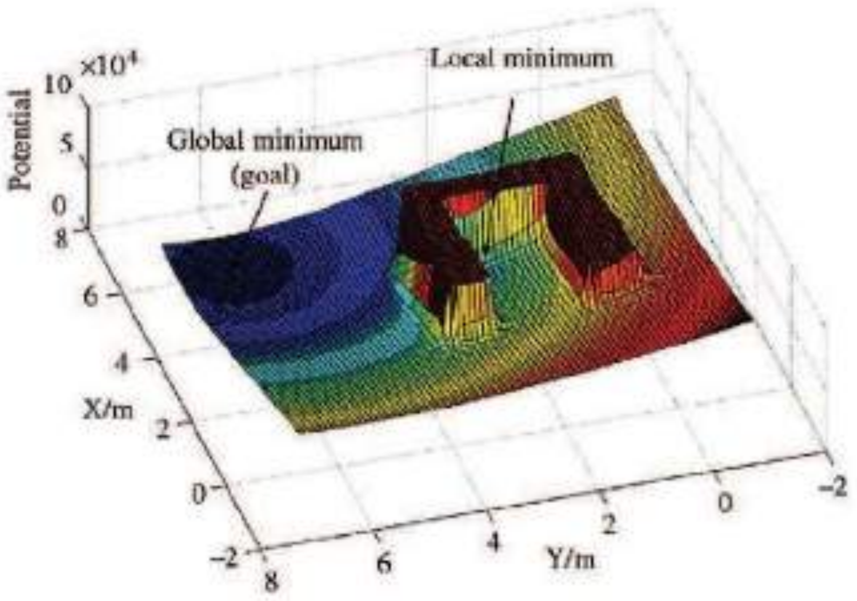


Fig.: Example of an artificial potential field for a particle in the plane

Probabilistic Roadmap Method

- Sample randomly the configuration space $\mathcal{Q}$
- Discard samples that belong to $\mathcal{QO}$
- Connect pairs of configuration nodes, for instance
  1. Introduce a distance measure $d(\cdot)$ in $\mathcal{Q}$, e.g. 2-norm
  2. Choose a radius $\varepsilon > 0$ and identify k nearest neighbors to a particular node
  3. Connect nodes and discard path segments for which collision occurs

  The result is a roadmap of several disjoint components.
- Enhance the roadmap by connecting as many disjoint components as possible (e.g. sample additional nodes)
- Connect q_0 and q_f to the roadmap and run a path smoothing algorithm

Example: PRM for a 2-D Configuration Space

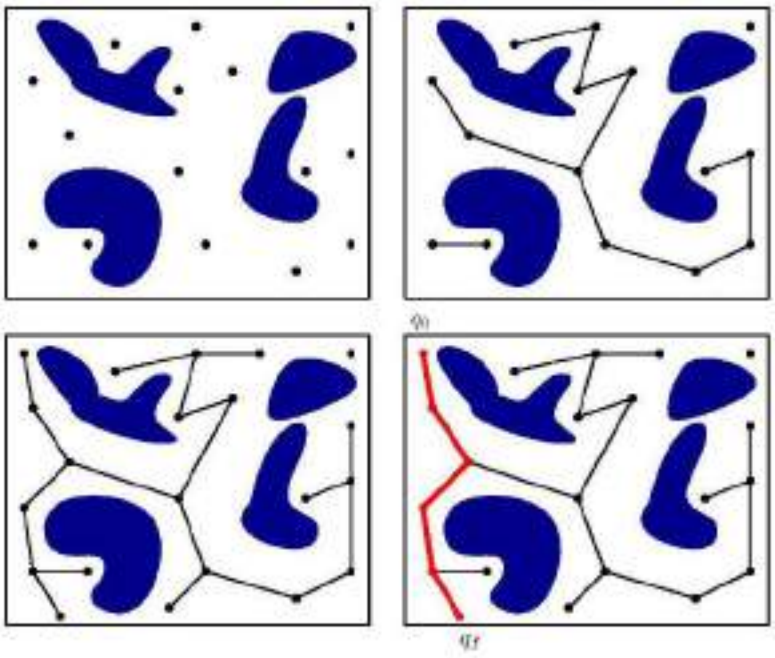


Fig.: Steps in constructing a probabilistic roadmap for a 2-D configuration space containing polygonal obstacles

Cell Decomposition

Principle

The idea is to decompose free configuration space into cells that are trapezoids or triangles. Planning in each cell is trivial because it is convex. A roadmap is made by placing a point in the center of each cell and each boundary between cells. Any graph search algorithm can be used to find a collision-free path quickly.

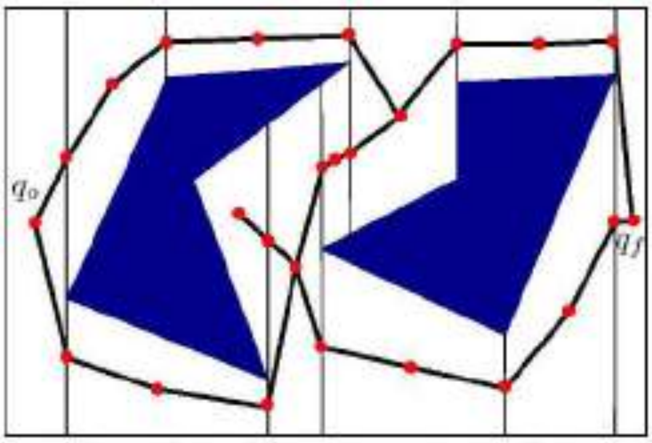


Fig.: The roadmap derived from the vertical cell decomposition.

Cell Decomposition Types

There are two types of cell decomposition:

- exact cell decomposition
- approximate cell decomposition

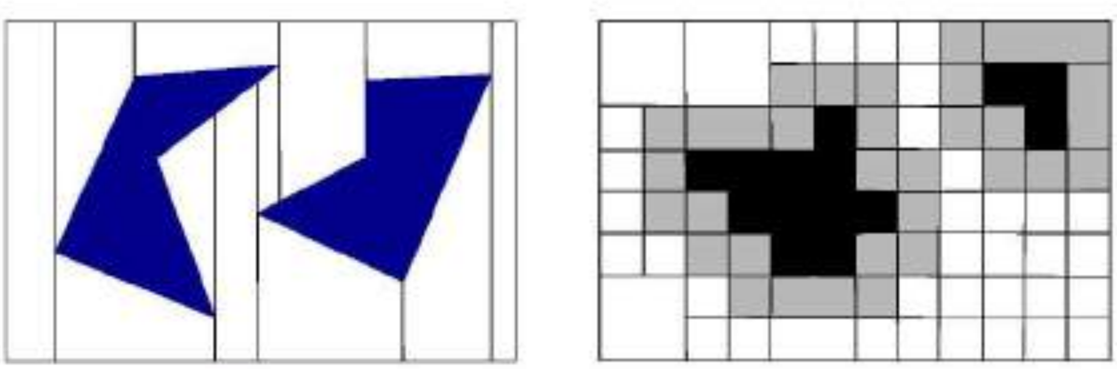


Fig.: Exact and approximate decomposition types

Adjacency graph

An adjacency graph is obtained after the decomposition of the free configuration space. It is comprised of nodes (correspond to cells) and edges, which connects adjacent nodes. Two nodes are supposed to be adjacent if they share a common boundary.

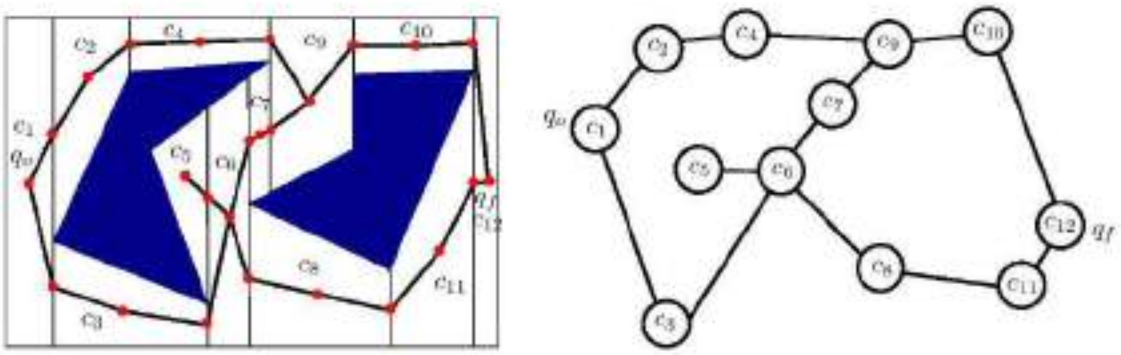


Fig.: Adjacency graph

Trapezoidal Decomposition

Path planning is carried out in two steps. Firstly, cells containing the initial and final configurations should be determined. Secondly, a path within adjacency graph is searched.

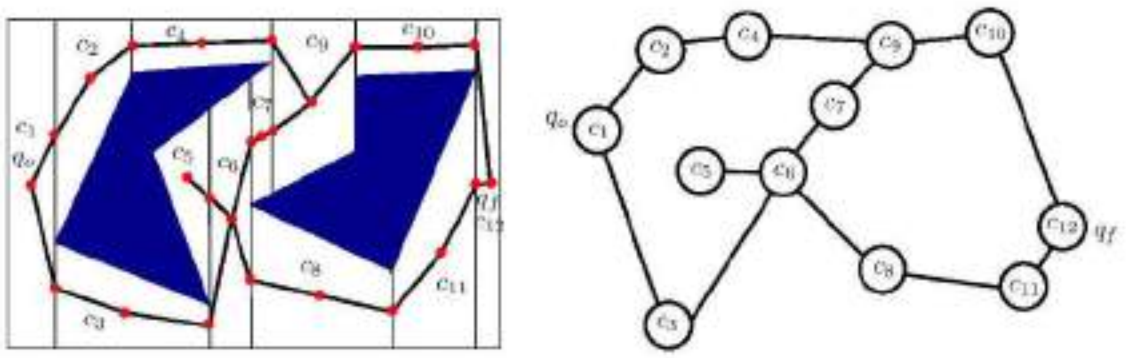


Fig.: Adjacency graph

Trapezoidal Decomposition

Path planning is carried out in two steps. Firstly, cells containing the initial and final configurations should be determined. Secondly, a path within adjacency graph is searched.

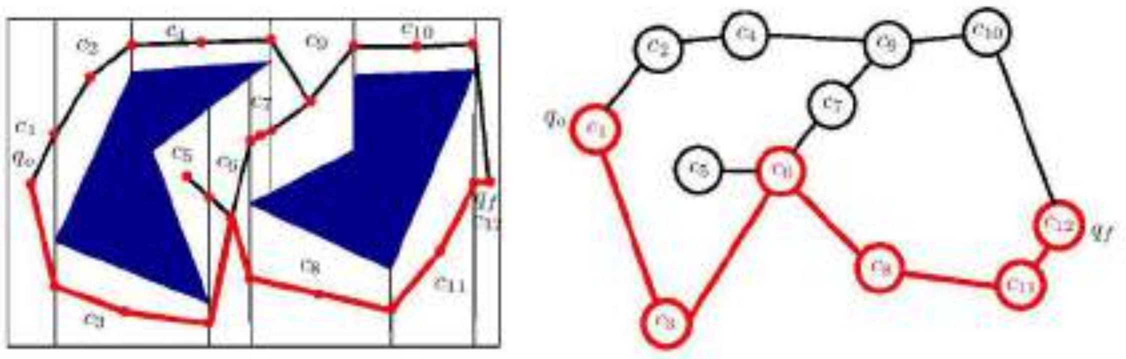
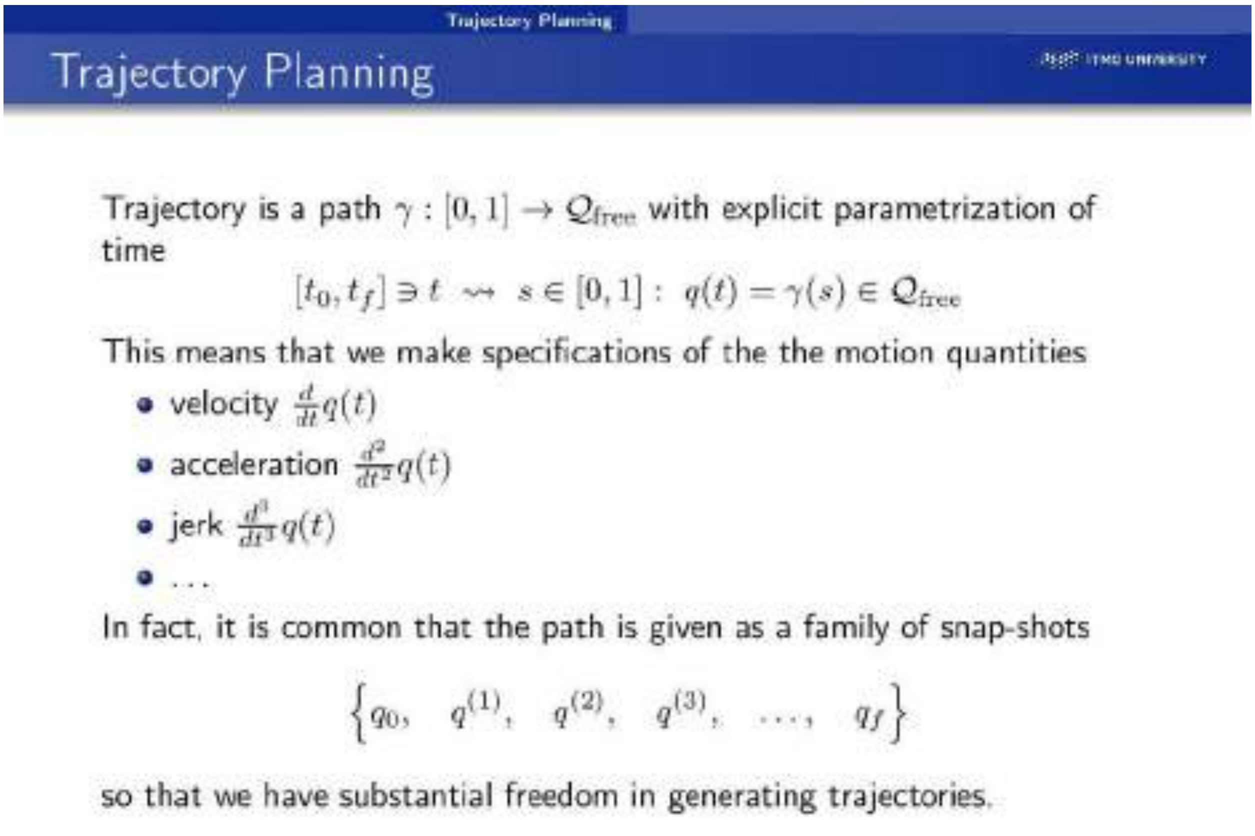


Fig.: Adjacency graph



Trajectory Planning

Trajectory is a path $\gamma : [0,1] \to \mathcal{Q}_{\text{free}}$ with explicit parametrization of time

$$[t_0, t_f] \ni t \rightsquigarrow s \in [0,1] : \ q(t) = \gamma(s) \in \mathcal{Q}_{\text{free}}$$

This means that we make specifications of the the motion quantities

- velocity $\frac{d}{dt}q(t)$
- acceleration $\frac{d^2}{dt^2}q(t)$
- jerk $\frac{d^3}{dt^3}q(t)$
- ...

In fact, it is common that the path is given as a family of snap-shots

$$\left\{ q_0, \quad q^{(1)}, \quad q^{(2)}, \quad q^{(3)}, \quad \dots, \quad q_f \right\}$$

so that we have substantial freedom in generating trajectories.

Decomposition in Fast and Slow Motions

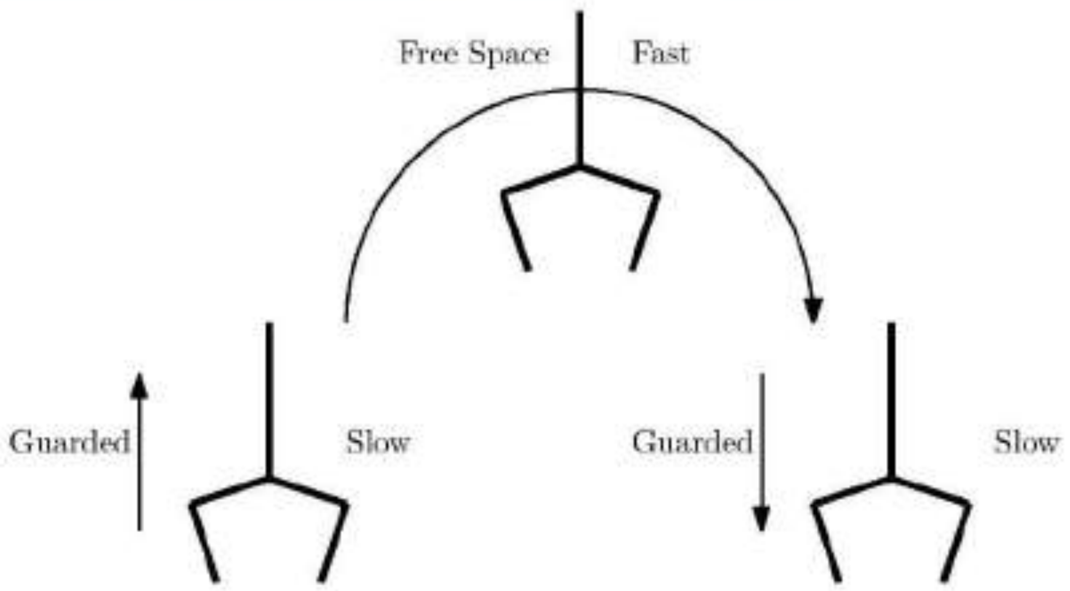
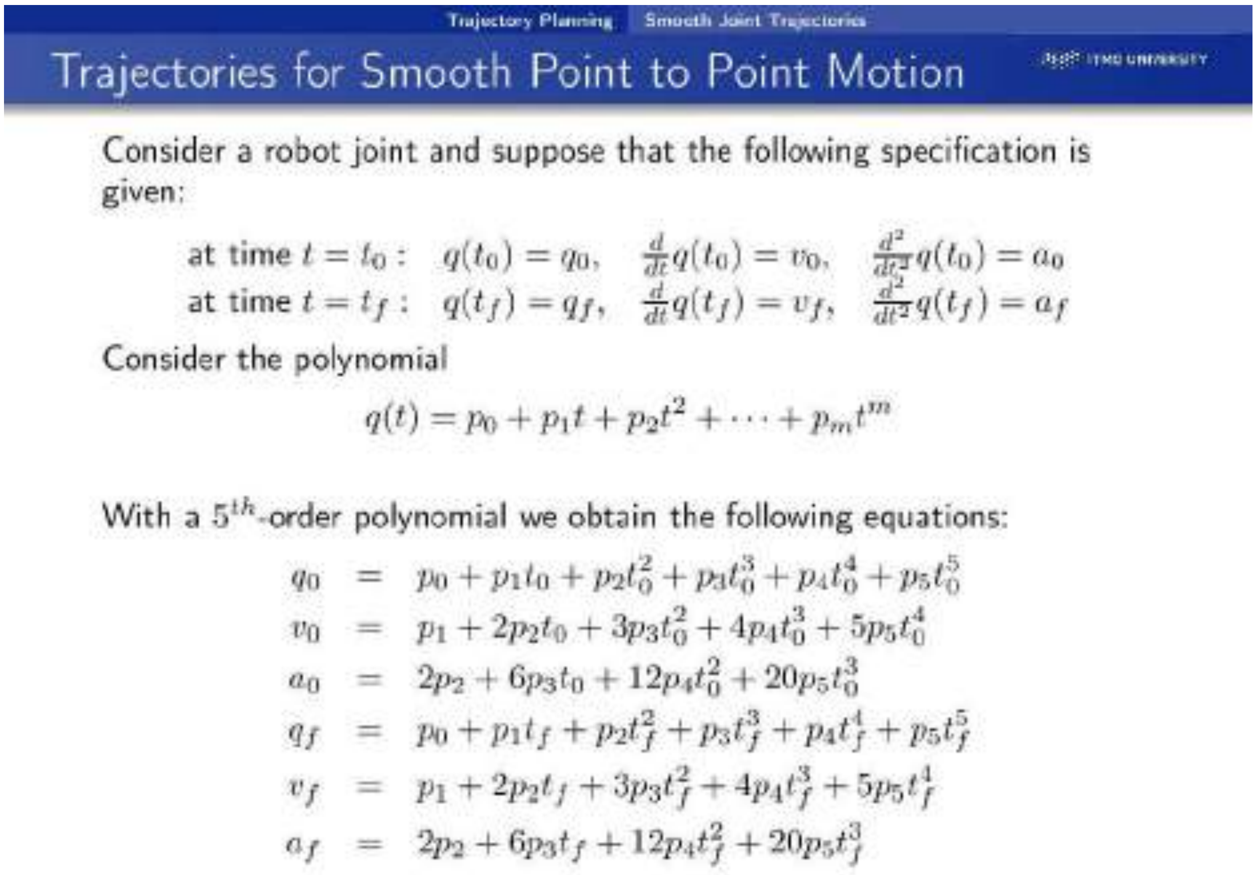


Fig.: Decomposition of the end-effector trajectory in slow guarded motion whenever precision or safety is required and fast free motion whenever no obstacles are nearby



Trajectories for Smooth Point to Point Motion

Consider a robot joint and suppose that the following specification is given:

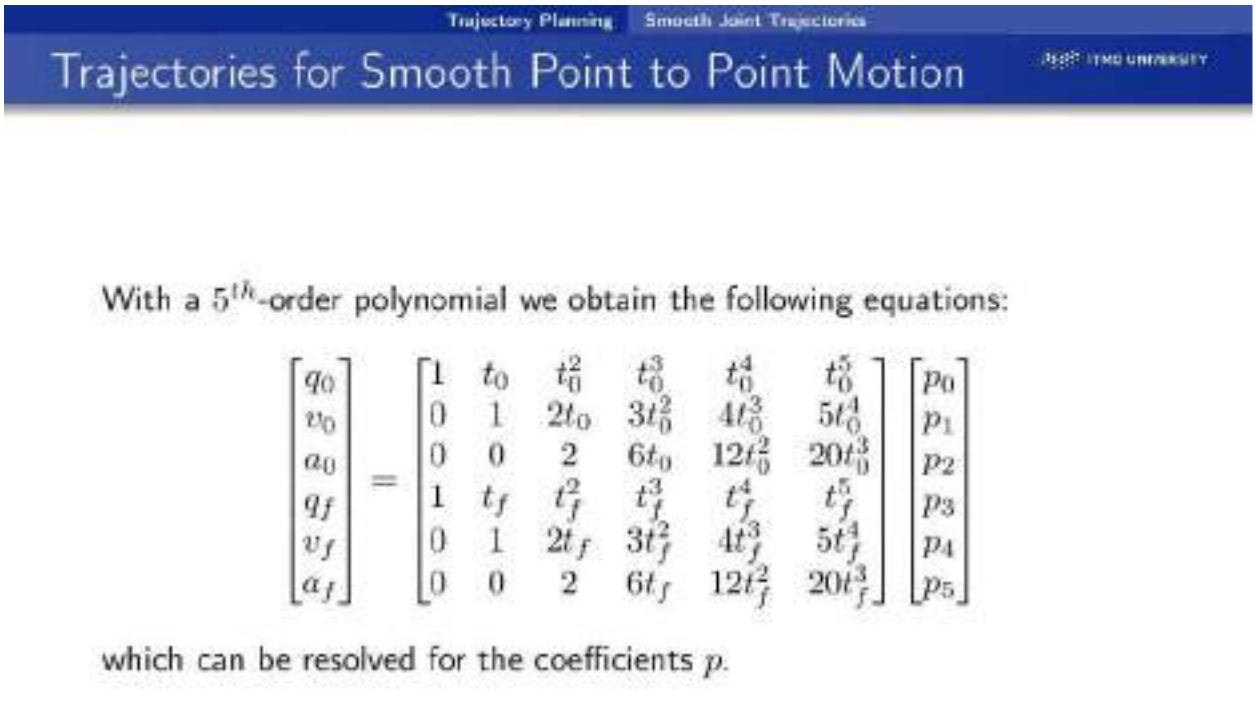
$$\begin{array}{llll} \text{at time } t = t_0: & q(t_0) = q_0, & \frac{d}{dt}q(t_0) = v_0, & \frac{d^2}{dt^2}q(t_0) = a_0 \\ \text{at time } t = t_f: & q(t_f) = q_f, & \frac{d}{dt}q(t_f) = v_f, & \frac{d^2}{dt^2}q(t_f) = a_f \end{array}$$

Consider the polynomial

$$q(t) = p_0 + p_1 t + p_2 t^2 + \cdots + p_m t^m$$

With a 5^{th}-order polynomial we obtain the following equations:

$$\begin{aligned} q_0 &= p_0 + p_1 t_0 + p_2 t_0^2 + p_3 t_0^3 + p_4 t_0^4 + p_5 t_0^5 \\ v_0 &= p_1 + 2p_2 t_0 + 3p_3 t_0^2 + 4p_4 t_0^3 + 5p_5 t_0^4 \\ a_0 &= 2p_2 + 6p_3 t_0 + 12p_4 t_0^2 + 20p_5 t_0^3 \\ q_f &= p_0 + p_1 t_f + p_2 t_f^2 + p_3 t_f^3 + p_4 t_f^4 + p_5 t_f^5 \\ v_f &= p_1 + 2p_2 t_f + 3p_3 t_f^2 + 4p_4 t_f^3 + 5p_5 t_f^4 \\ a_f &= 2p_2 + 6p_3 t_f + 12p_4 t_f^2 + 20p_5 t_f^3 \end{aligned}$$

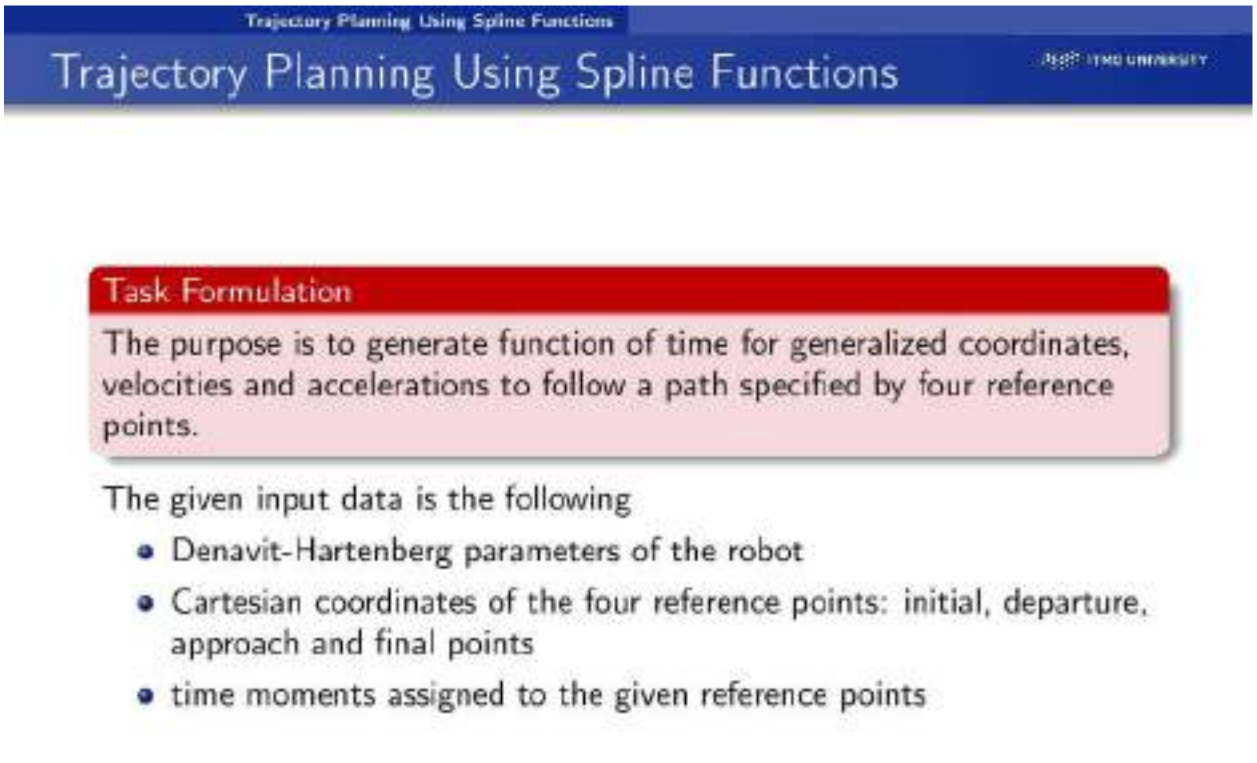


Trajectories for Smooth Point to Point Motion

With a 5^{th}-order polynomial we obtain the following equations:

$$\begin{bmatrix} q_0 \\ v_0 \\ a_0 \\ q_f \\ v_f \\ a_f \end{bmatrix} = \begin{bmatrix} 1 & t_0 & t_0^2 & t_0^3 & t_0^4 & t_0^5 \\ 0 & 1 & 2t_0 & 3t_0^2 & 4t_0^3 & 5t_0^4 \\ 0 & 0 & 2 & 6t_0 & 12t_0^2 & 20t_0^3 \\ 1 & t_f & t_f^2 & t_f^3 & t_f^4 & t_f^5 \\ 0 & 1 & 2t_f & 3t_f^2 & 4t_f^3 & 5t_f^4 \\ 0 & 0 & 2 & 6t_f & 12t_f^2 & 20t_f^3 \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \end{bmatrix}$$

which can be resolved for the coefficients p.



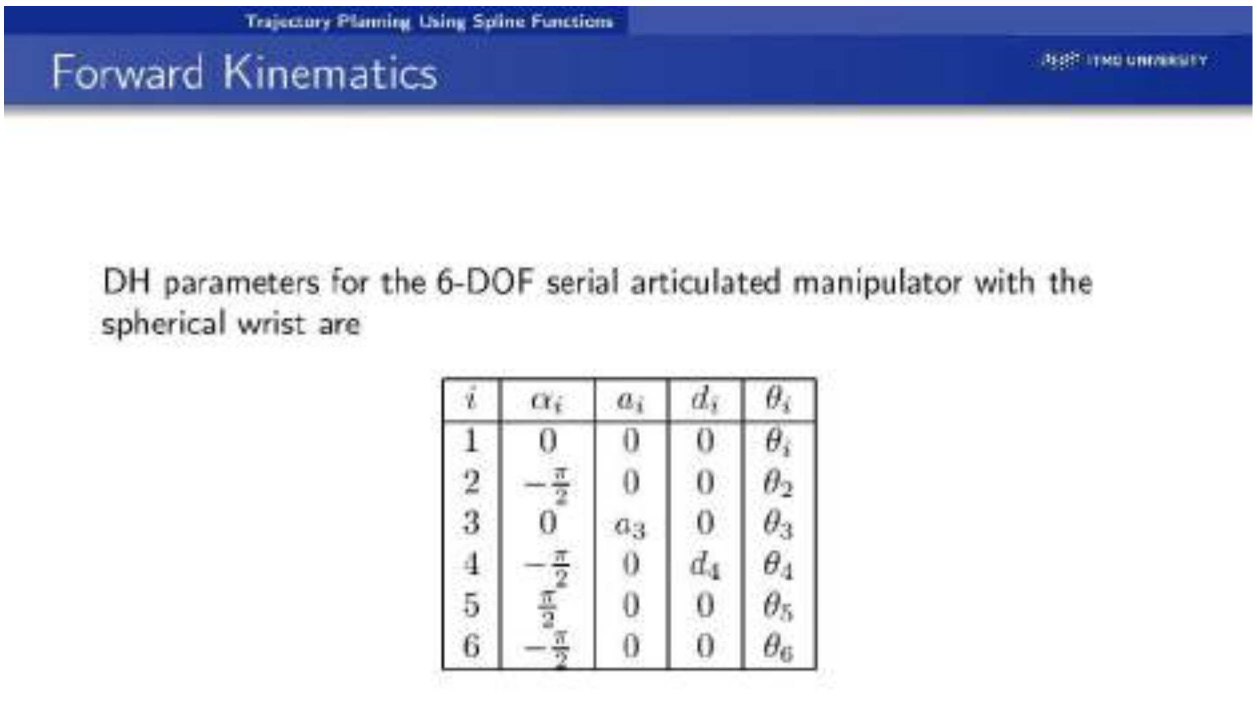
Trajectory Planning Using Spline Functions

Task Formulation

The purpose is to generate function of time for generalized coordinates, velocities and accelerations to follow a path specified by four reference points.

The given input data is the following

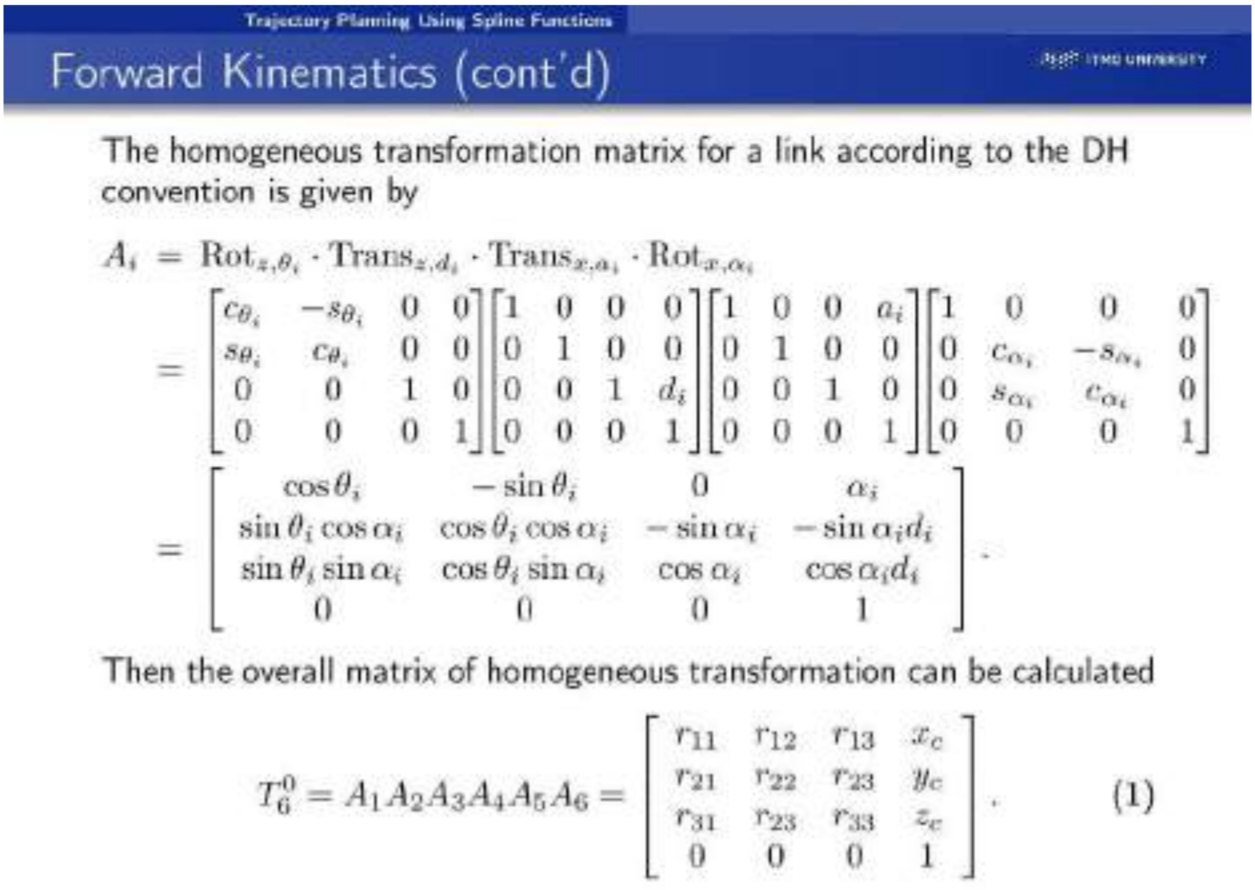
- Denavit-Hartenberg parameters of the robot
- Cartesian coordinates of the four reference points: initial, departure, approach and final points
- time moments assigned to the given reference points



Forward Kinematics

DH parameters for the 6-DOF serial articulated manipulator with the spherical wrist are

i	α_i	a_i	d_i	θ_i
1	0	0	0	θ_1
2	$-\frac{\pi}{2}$	0	0	θ_2
3	0	a_3	0	θ_3
4	$-\frac{\pi}{2}$	0	d_4	θ_4
5	$\frac{\pi}{2}$	0	0	θ_5
6	$-\frac{\pi}{2}$	0	0	θ_6



Trajectory Planning Using Spline Functions

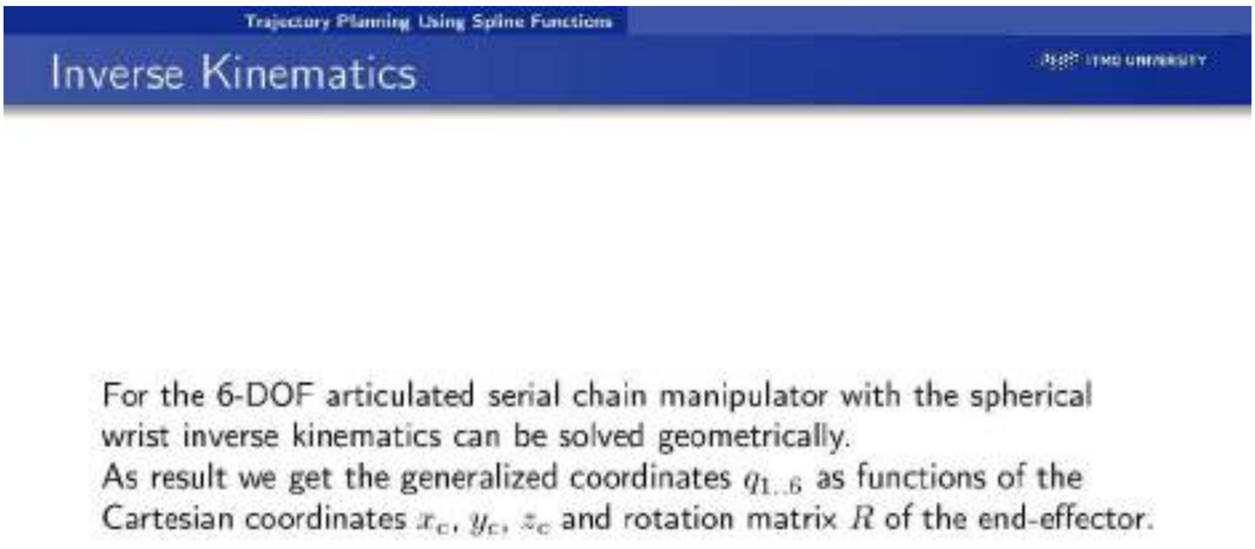
Forward Kinematics (cont'd)

The homogeneous transformation matrix for a link according to the DH convention is given by

$$
\begin{aligned}
A_i &= \mathrm{Rot}_{z,\theta_i} \cdot \mathrm{Trans}_{z,d_i} \cdot \mathrm{Trans}_{x,a_i} \cdot \mathrm{Rot}_{x,\alpha_i} \\
&= \begin{bmatrix} c_{\theta_i} & -s_{\theta_i} & 0 & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}
\begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_{\alpha_i} & -s_{\alpha_i} & 0 \\ 0 & s_{\alpha_i} & c_{\alpha_i} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & \alpha_i \\ \sin\theta_i \cos\alpha_i & \cos\theta_i \cos\alpha_i & -\sin\alpha_i & -\sin\alpha_i d_i \\ \sin\theta_i \sin\alpha_i & \cos\theta_i \sin\alpha_i & \cos\alpha_i & \cos\alpha_i d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}.
\end{aligned}
$$

Then the overall matrix of homogeneous transformation can be calculated

$$
T_6^0 = A_1 A_2 A_3 A_4 A_5 A_6 = \begin{bmatrix} r_{11} & r_{12} & r_{13} & x_c \\ r_{21} & r_{22} & r_{23} & y_c \\ r_{31} & r_{23} & r_{33} & z_c \\ 0 & 0 & 0 & 1 \end{bmatrix}. \qquad (1)
$$



Trajectory Planning Using Spline Functions

Inverse Kinematics

For the 6-DOF articulated serial chain manipulator with the spherical wrist inverse kinematics can be solved geometrically.
As result we get the generalized coordinates $q_{1..6}$ as functions of the Cartesian coordinates x_c, y_c, z_c and rotation matrix R of the end-effector.

Relative Time Parametrization

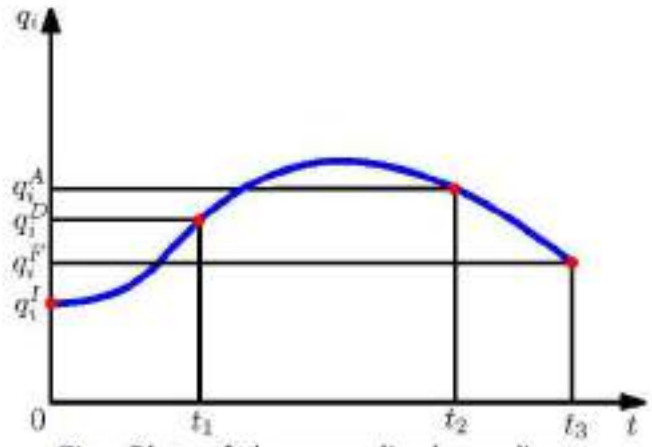


Fig.: Plots of the generalized coordinates

Use relative time for each part of the trajectory instead of absolute time as follows

$$\tau_p = \frac{t - t_{p-1}}{t_p - t_{p-1}}, \quad t_{p-1} \le t \le t_p, \quad 0 \le \tau_p \le 1. \tag{2}$$

Polynomials

Change of generalized coordinates is given by

$$q_i(\tau) = \begin{cases} a_{4i}\tau_1^4 + a_{3i}\tau_1^3 + a_{2i}\tau_1^2 + a_{1i}\tau_1 + a_{0i}, \\ b_{3i}\tau_2^3 + b_{2i}\tau_2^2 + b_{1i}\tau_2 + b_{0i}, \\ c_{4i}\tau_3^4 + c_{3i}\tau_3^3 + c_{2i}\tau_3^2 + c_{1i}\tau_3 + c_{0i}. \end{cases} \tag{3}$$

Change of generalized velocities is given by

$$\dot{q}_i(\tau) = \begin{cases} 4a_{4i}\tau_1^3 + 3a_{3i}\tau_1^2, \\ 3b_{3i}\tau_2^2 + 2b_{2i}\tau_2 + b_{1i}, \\ 4c_{4i}\tau_3^3 + 3c_{3i}\tau_3^2 + 2c_{2i}\tau_3 + c_{1i}. \end{cases} \tag{4}$$

Change of generalized accelerations is given by

$$\ddot{q}_i(\tau) = \begin{cases} 12a_{4i}\tau_1^2 + 6a_{3i}\tau_1, \\ 6b_{3i}\tau_2 + 2b_{2i}, \\ 12c_{4i}\tau_3^2 + 6c_{3i}\tau_3 + 2c_{2i}. \end{cases} \tag{5}$$

Trajectory Planning Using Spline Functions

Constraints

Take into account the following constraints on positions, velocities and accelerations in order to consequently "stick" them

Departure part	Mid-part	Approach part
$q_{1i}(0) = q_i^S$,	$q_{2i}(0) = q_i^D$,	$q_{3i}(0) = q_i^A$,
$\dot{q}_{1i}(0) = 0$,	$\dot{q}_{2i}(0) = \dot{q}_{1i}(1)$,	$\dot{q}_{3i}(0) = \dot{q}_{2i}(1)$,
$\ddot{q}_{1i}(0) = 0$,	$\ddot{q}_{2i}(0) = \ddot{q}_{1i}(1)$,	$\ddot{q}_{3i}(0) = \ddot{q}_{2i}(1)$,
$q_{1i}(1) = q_i^D$.	$q_{2i}(1) = q_i^A$.	$q_{3i}(1) = q_i^F$,
		$\dot{q}_{3i}(1) = 0$,
		$\ddot{q}_{3i}(1) = 0$.

Trajectory Planning Using Spline Functions

Result

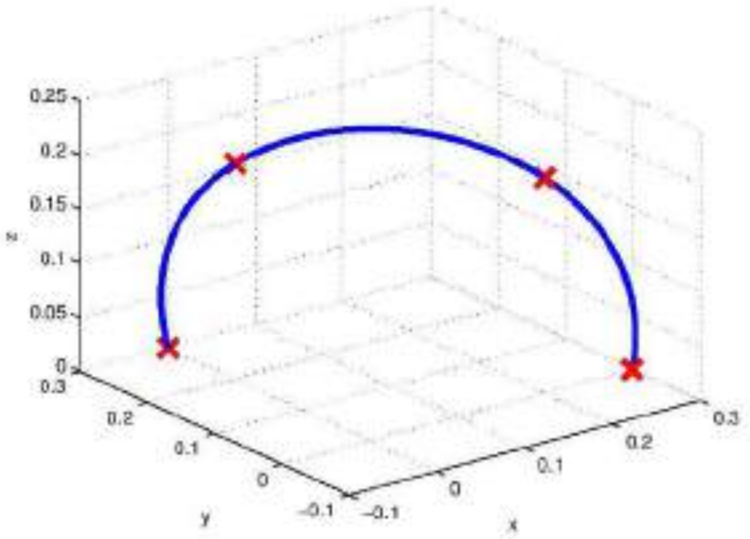
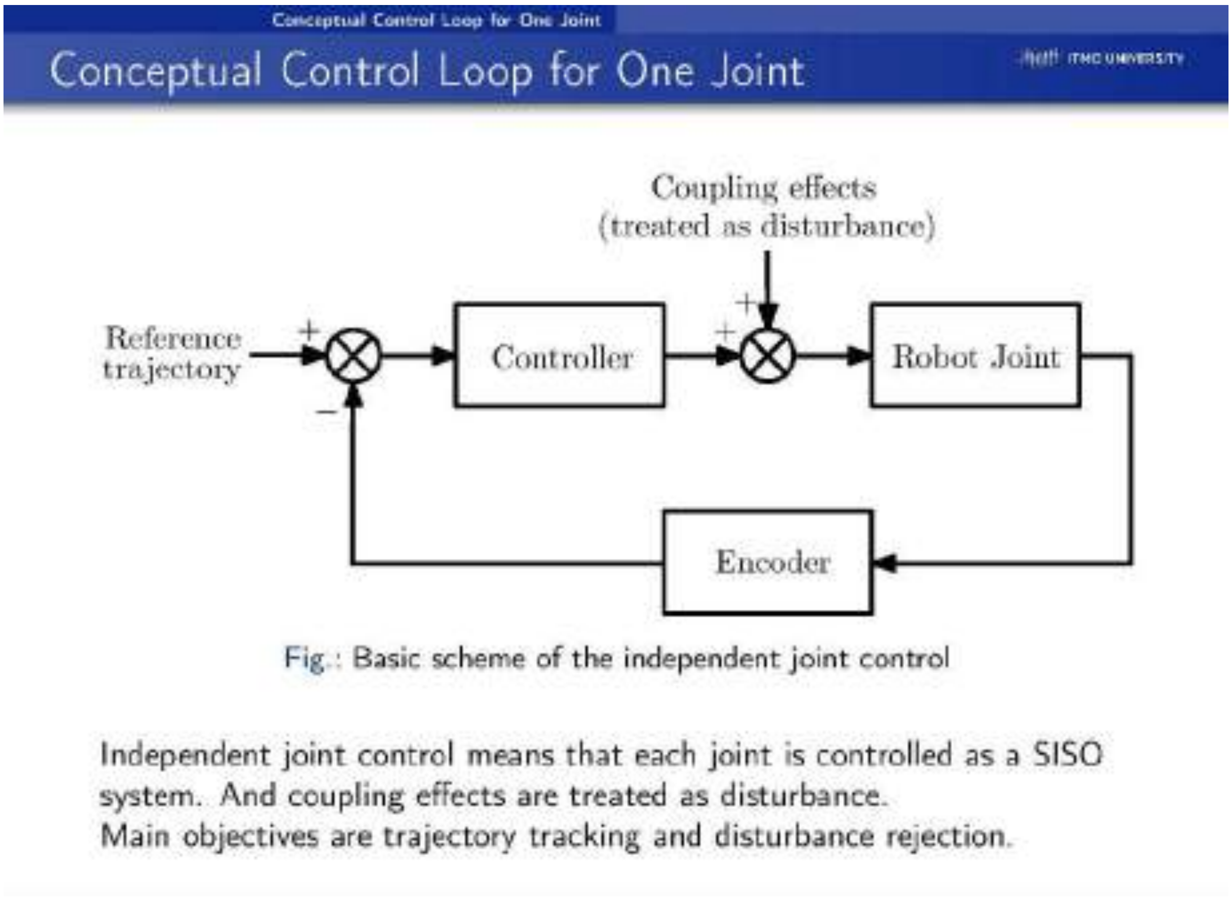


Fig.: The trajectory generated by the spline-functions.

Robot Control Design

Outline



Conceptual Control Loop for One Joint

Conceptual Control Loop for One Joint

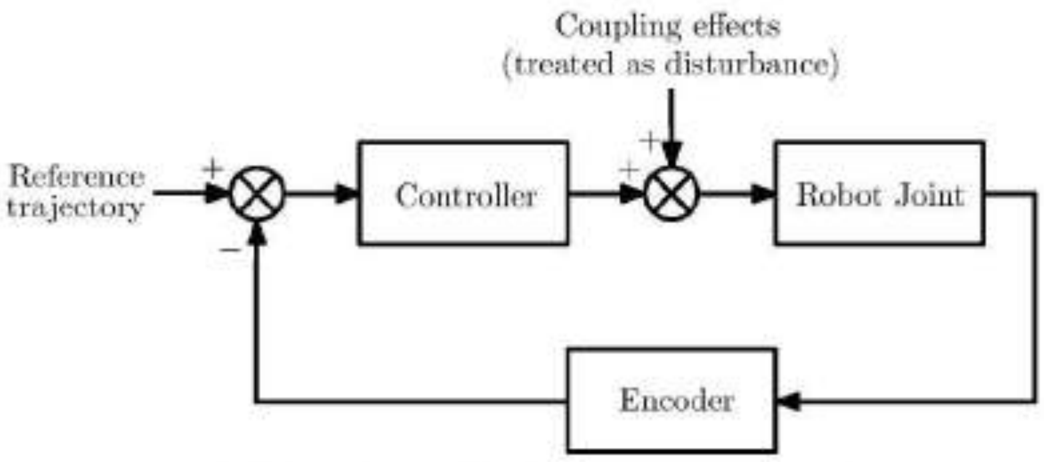
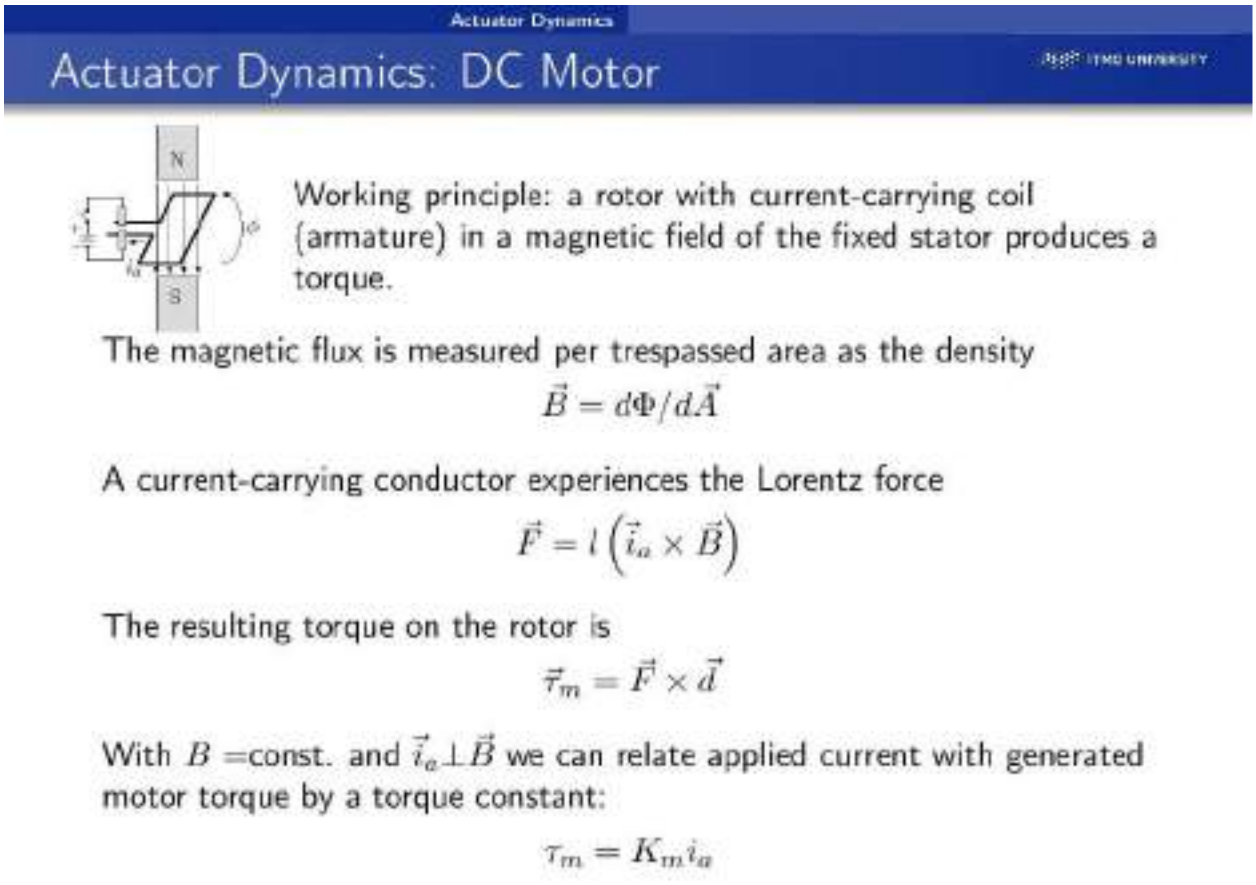


Fig.: Basic scheme of the independent joint control

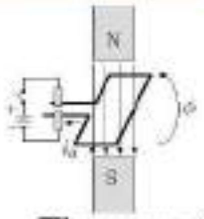
Independent joint control means that each joint is controlled as a SISO system. And coupling effects are treated as disturbance.
Main objectives are trajectory tracking and disturbance rejection.



Actuator Dynamics

Actuator Dynamics: DC Motor

ITMO UNIVERSITY



Working principle: a rotor with current-carrying coil (armature) in a magnetic field of the fixed stator produces a torque.

The magnetic flux is measured per trespassed area as the density

$$\vec{B} = d\Phi / d\vec{A}$$

A current-carrying conductor experiences the Lorentz force

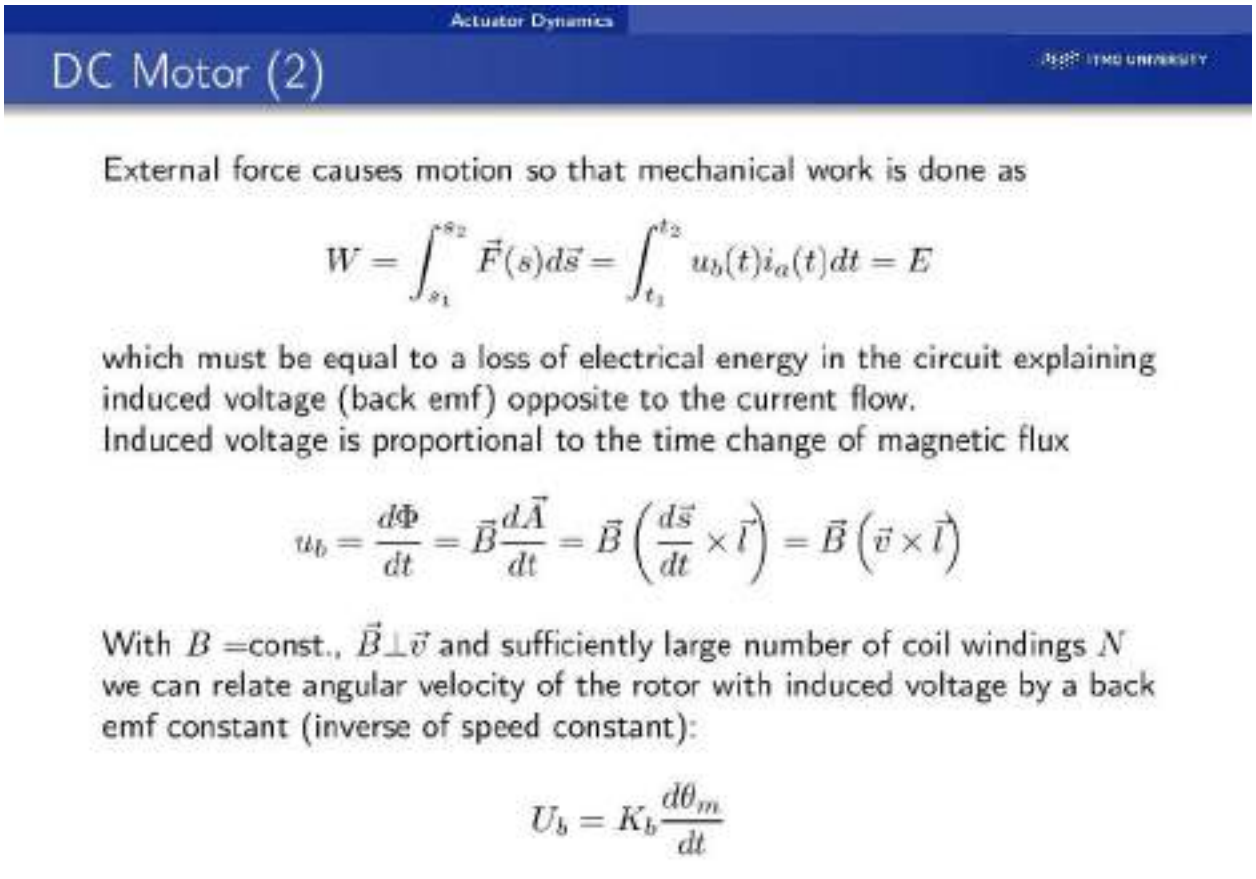
$$\vec{F} = l\left(\vec{i}_a \times \vec{B}\right)$$

The resulting torque on the rotor is

$$\vec{\tau}_m = \vec{F} \times \vec{d}$$

With B =const. and $\vec{i}_a \perp \vec{B}$ we can relate applied current with generated motor torque by a torque constant:

$$\tau_m = K_m i_a$$



Actuator Dynamics

DC Motor (2)

ITMO UNIVERSITY

External force causes motion so that mechanical work is done as

$$W = \int_{s_1}^{s_2} \vec{F}(s) d\vec{s} = \int_{t_1}^{t_2} u_b(t) i_a(t) dt = E$$

which must be equal to a loss of electrical energy in the circuit explaining induced voltage (back emf) opposite to the current flow.
Induced voltage is proportional to the time change of magnetic flux

$$u_b = \frac{d\Phi}{dt} = \vec{B}\frac{d\vec{A}}{dt} = \vec{B}\left(\frac{d\vec{s}}{dt} \times \vec{l}\right) = \vec{B}\left(\vec{v} \times \vec{l}\right)$$

With B =const., $\vec{B} \perp \vec{v}$ and sufficiently large number of coil windings N we can relate angular velocity of the rotor with induced voltage by a back emf constant (inverse of speed constant):

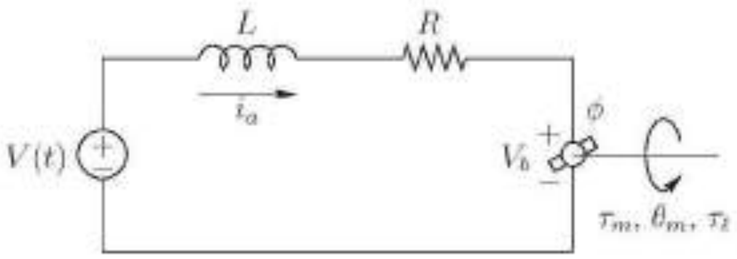
$$U_b = K_b \frac{d\theta_m}{dt}$$

Actuator Dynamics

DC Motor (3)

ITMO UNIVERSITY

Circuit diagram for an armature controlled DC motor with inductance L, resistance R and applied control voltage V:



The differential equation for the armature current is

$$L\frac{di_a}{dt} + Ri_a = V - V_b$$

where the induced voltage $V_b = K_b\frac{d\theta_m}{dt}$ is related to angular velocity of the rotor shaft.

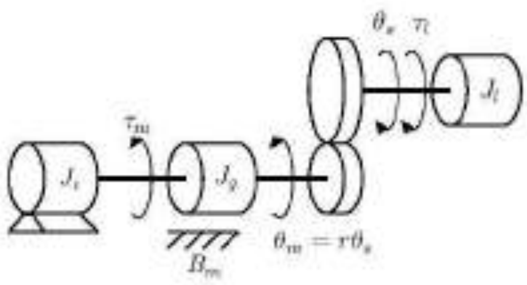
Dynamical Model of a Single-Joint Robot

Dynamical Model of DC-Motor/Gears and Single Link

ITMO UNIVERSITY

Lumped model of a single link with actuator/gear drive train

- DC motor is in series with a gear box
- Gear ratio is $r:1 \Rightarrow \theta_m = r\theta_s$
- Sum of actuator and gear inertias is $J_m = J_a + J_g$
- Inertia of the link is J_l (assumed constant)
- Coefficient of motor friction is B_m



The equation of motion for motor angle θ_m is then

$$\begin{aligned} J_m\frac{d^2\theta_m}{dt^2} &= \tau_m - \tau_l/r - B_m\frac{d\theta_m}{dt} \\ &= K_m i_a - \tau_l/r - B_m\frac{d\theta_m}{dt} \end{aligned}$$

Dynamical Model of DC-Motor/Gears and Single Link

Augmenting the mechanical and electrical models, we obtain:

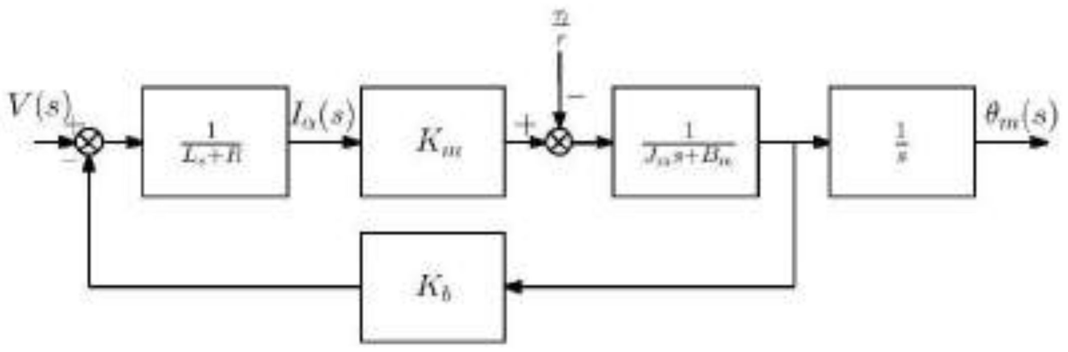
$$\begin{aligned} L\left(\frac{d}{dt}i_a\right) + Ri_a &= V - K_b\left(\frac{d}{dt}\theta_m\right) \\ J_m\left(\frac{d^2}{dt^2}\theta_m\right) + B_m\left(\frac{d}{dt}\theta_m\right) &= K_m i_a - \tau_l/r \end{aligned}$$

LTI system represented by linear ODEs with constant coefficients.
We can use classical methods for controlling it.
The Laplace transform yields

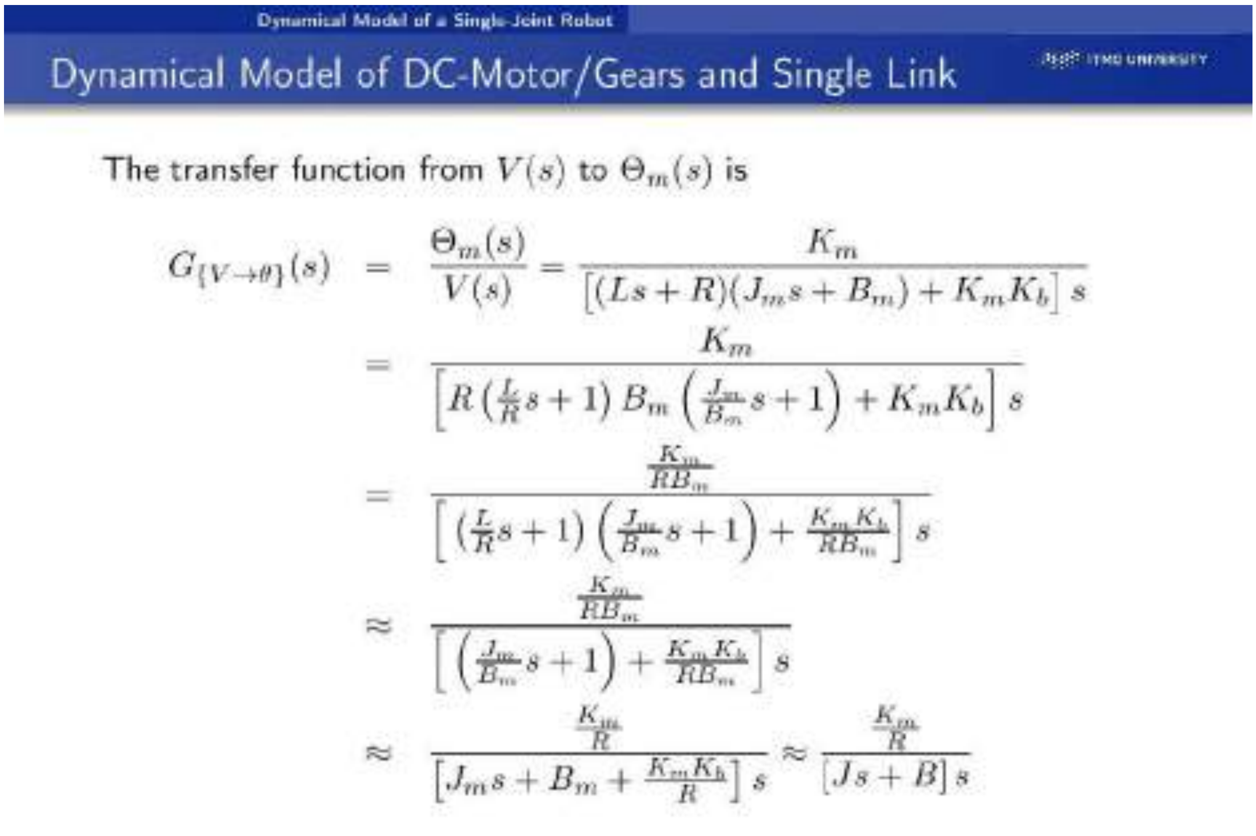
$$\begin{aligned} (Ls+R)I_a(s) &= V(s) - K_b s\Theta_m(s) \\ (J_m s + B_m)s\Theta_m(s) &= K_m I_a(s) - \tau_l(s)/r \end{aligned}$$

Dynamical Model of DC-Motor/Gears and Single Link

Block diagram for a DC motor system:



The electrical subsystem is typically much faster than the mechanical one.
Assuming $\frac{L}{R} \approx 0$ leads to a reduced order actuator dynamics.
This can be justified by e.g. balanced model order reduction.
Let's find the transfer fcts from $V(s)$ to $\Theta_m(s)$ and $\tau_l(s)$ to $\Theta_m(s)$.

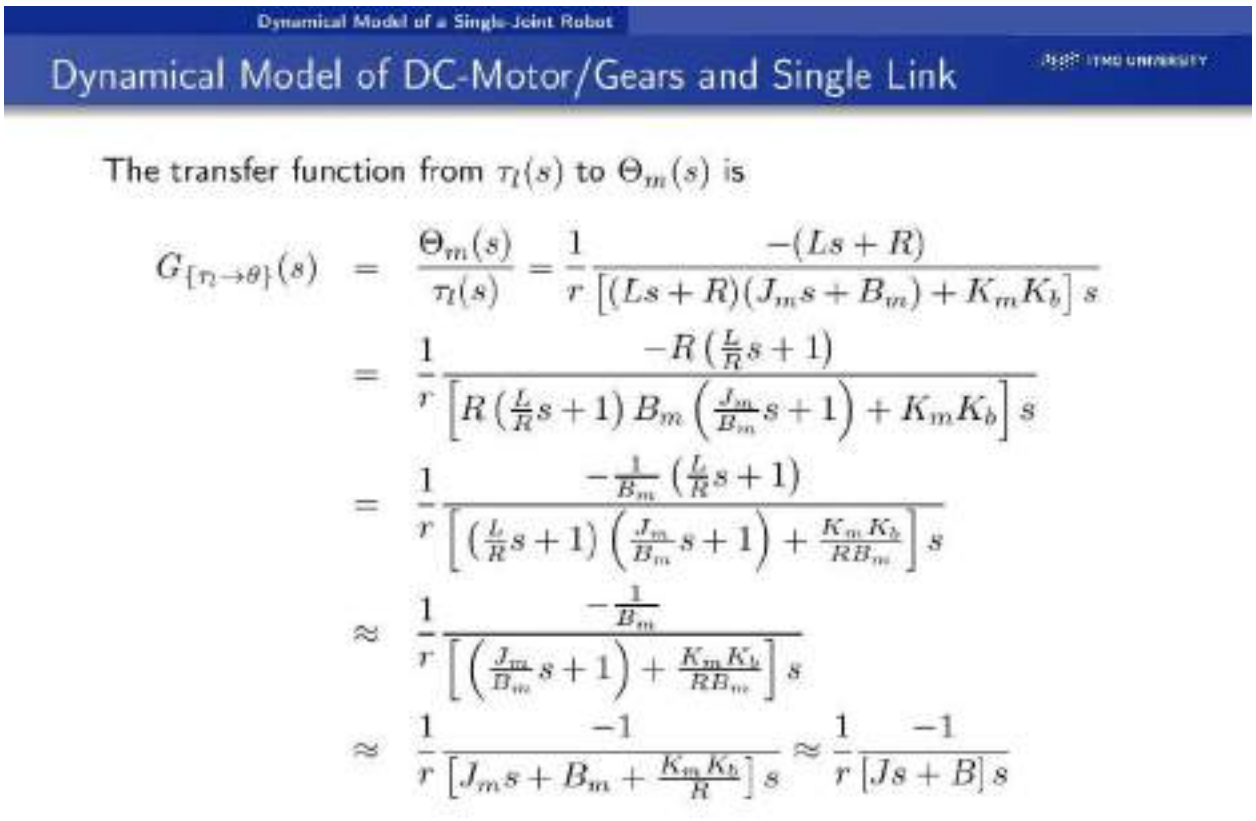


Dynamical Model of DC-Motor/Gears and Single Link

ITMO UNIVERSITY

The transfer function from $V(s)$ to $\Theta_m(s)$ is

$$
\begin{aligned}
G_{\{V\to\theta\}}(s) &= \frac{\Theta_m(s)}{V(s)} = \frac{K_m}{\left[(Ls+R)(J_m s+B_m)+K_m K_b\right] s} \\
&= \frac{K_m}{\left[R\left(\frac{L}{R}s+1\right)B_m\left(\frac{J_m}{B_m}s+1\right)+K_m K_b\right] s} \\
&= \frac{\frac{K_m}{RB_m}}{\left[\left(\frac{L}{R}s+1\right)\left(\frac{J_m}{B_m}s+1\right)+\frac{K_m K_b}{RB_m}\right] s} \\
&\approx \frac{\frac{K_m}{RB_m}}{\left[\left(\frac{J_m}{B_m}s+1\right)+\frac{K_m K_b}{RB_m}\right] s} \\
&\approx \frac{\frac{K_m}{R}}{\left[J_m s+B_m+\frac{K_m K_b}{R}\right] s} \approx \frac{\frac{K_m}{R}}{\left[Js+B\right] s}
\end{aligned}
$$

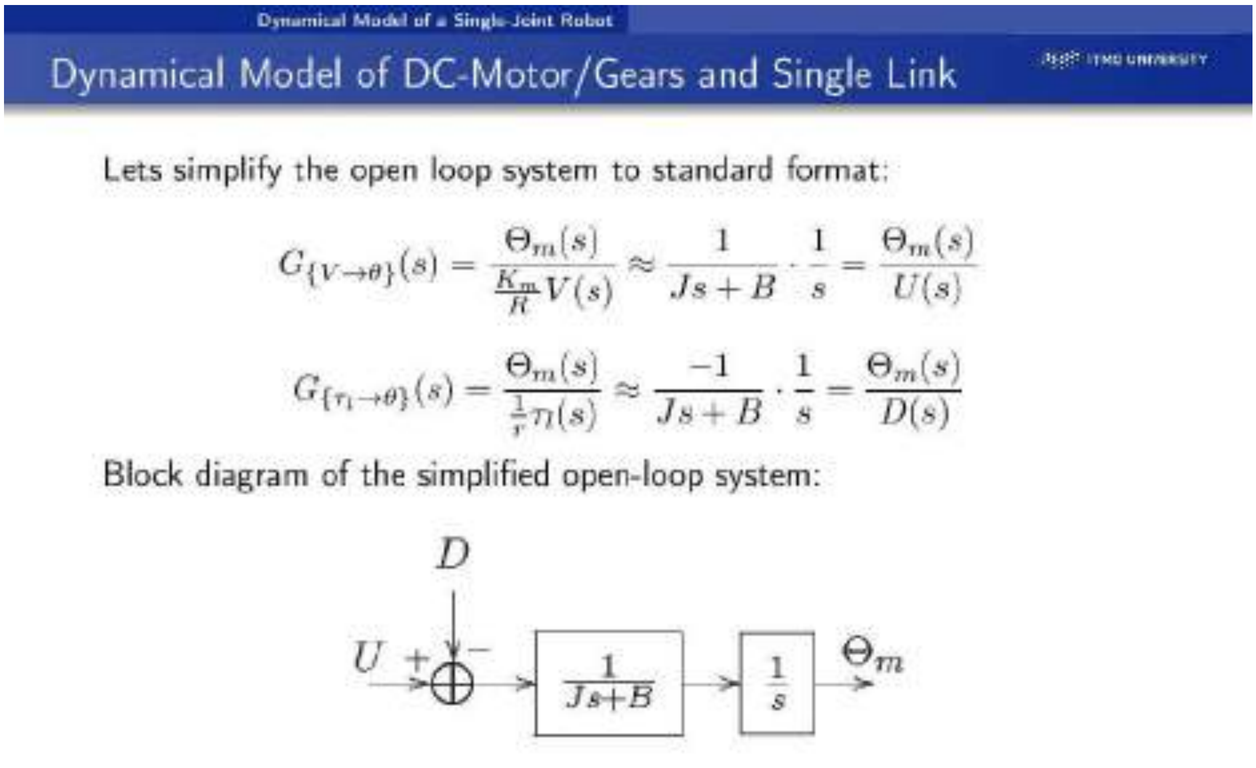


Dynamical Model of DC-Motor/Gears and Single Link

ITMO UNIVERSITY

The transfer function from $\tau_l(s)$ to $\Theta_m(s)$ is

$$
\begin{aligned}
G_{\{\tau_l\to\theta\}}(s) &= \frac{\Theta_m(s)}{\tau_l(s)} = \frac{1}{r}\frac{-(Ls+R)}{\left[(Ls+R)(J_m s+B_m)+K_m K_b\right] s} \\
&= \frac{1}{r}\frac{-R\left(\frac{L}{R}s+1\right)}{\left[R\left(\frac{L}{R}s+1\right)B_m\left(\frac{J_m}{B_m}s+1\right)+K_m K_b\right] s} \\
&= \frac{1}{r}\frac{-\frac{1}{B_m}\left(\frac{L}{R}s+1\right)}{\left[\left(\frac{L}{R}s+1\right)\left(\frac{J_m}{B_m}s+1\right)+\frac{K_m K_b}{RB_m}\right] s} \\
&\approx \frac{1}{r}\frac{-\frac{1}{B_m}}{\left[\left(\frac{J_m}{B_m}s+1\right)+\frac{K_m K_b}{RB_m}\right] s} \\
&\approx \frac{1}{r}\frac{-1}{\left[J_m s+B_m+\frac{K_m K_b}{R}\right] s} \approx \frac{1}{r}\frac{-1}{\left[Js+B\right] s}
\end{aligned}
$$



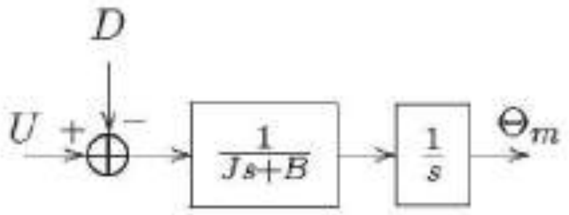
Dynamical Model of DC-Motor/Gears and Single Link

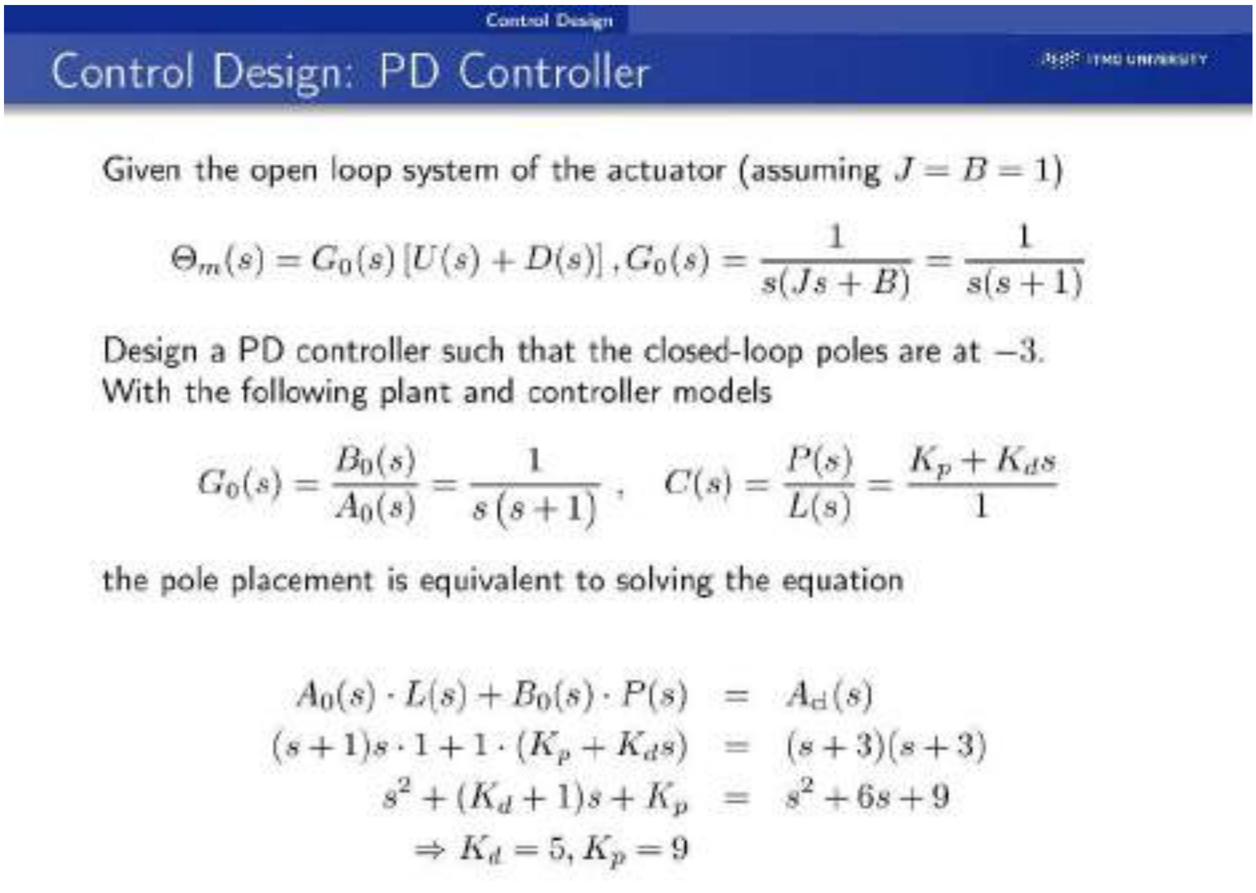
Lets simplify the open loop system to standard format:

$$G_{\{V\to\theta\}}(s) = \frac{\Theta_m(s)}{\frac{K_m}{R}V(s)} \approx \frac{1}{Js+B}\cdot\frac{1}{s} = \frac{\Theta_m(s)}{U(s)}$$

$$G_{\{\tau_l\to\theta\}}(s) = \frac{\Theta_m(s)}{\frac{1}{r}\tau_l(s)} \approx \frac{-1}{Js+B}\cdot\frac{1}{s} = \frac{\Theta_m(s)}{D(s)}$$

Block diagram of the simplified open-loop system:





Control Design: PD Controller

Given the open loop system of the actuator (assuming $J = B = 1$)

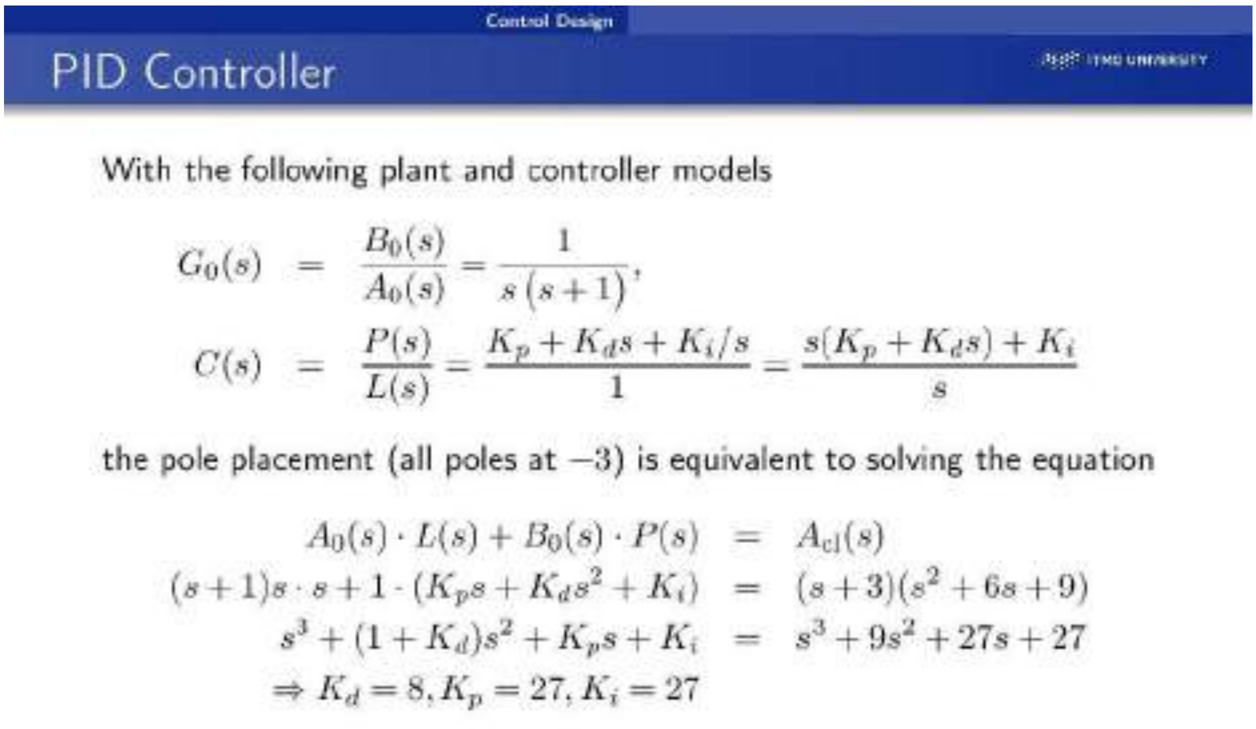
$$\Theta_m(s) = G_0(s)\left[U(s)+D(s)\right], G_0(s) = \frac{1}{s(Js+B)} = \frac{1}{s(s+1)}$$

Design a PD controller such that the closed-loop poles are at -3.
With the following plant and controller models

$$G_0(s) = \frac{B_0(s)}{A_0(s)} = \frac{1}{s\left(s+1\right)}, \quad C(s) = \frac{P(s)}{L(s)} = \frac{K_p + K_d s}{1}$$

the pole placement is equivalent to solving the equation

$$\begin{aligned} A_0(s)\cdot L(s) + B_0(s)\cdot P(s) &= A_d(s) \\ (s+1)s\cdot 1 + 1\cdot(K_p + K_d s) &= (s+3)(s+3) \\ s^2 + (K_d+1)s + K_p &= s^2 + 6s + 9 \\ \Rightarrow K_d = 5, K_p = 9 & \end{aligned}$$



Control Design

PID Controller

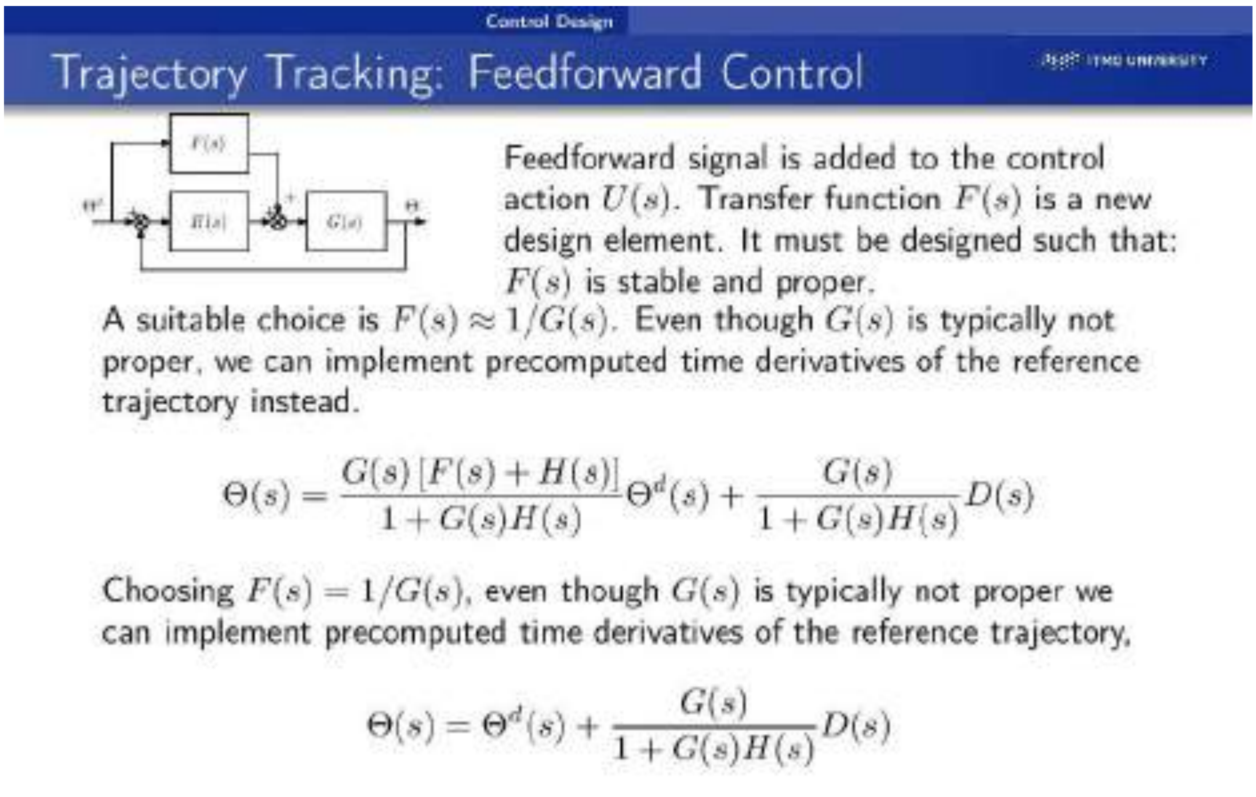
ITMO UNIVERSITY

With the following plant and controller models

$$G_0(s) = \frac{B_0(s)}{A_0(s)} = \frac{1}{s\,(s+1)},$$
$$C(s) = \frac{P(s)}{L(s)} = \frac{K_p + K_d s + K_i/s}{1} = \frac{s(K_p + K_d s) + K_i}{s}$$

the pole placement (all poles at -3) is equivalent to solving the equation

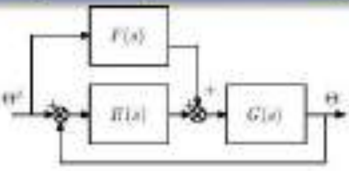
$$\begin{aligned} A_0(s)\cdot L(s) + B_0(s)\cdot P(s) &= A_{\text{cl}}(s) \\ (s+1)s\cdot s + 1\cdot(K_p s + K_d s^2 + K_i) &= (s+3)(s^2+6s+9) \\ s^3 + (1+K_d)s^2 + K_p s + K_i &= s^3 + 9s^2 + 27s + 27 \\ \Rightarrow K_d = 8, K_p = 27, K_i = 27 \end{aligned}$$



Control Design

Trajectory Tracking: Feedforward Control

ITMO UNIVERSITY



Feedforward signal is added to the control action $U(s)$. Transfer function $F(s)$ is a new design element. It must be designed such that: $F(s)$ is stable and proper.
A suitable choice is $F(s) \approx 1/G(s)$. Even though $G(s)$ is typically not proper, we can implement precomputed time derivatives of the reference trajectory instead.

$$\Theta(s) = \frac{G(s)\,[F(s)+H(s)]}{1+G(s)H(s)}\Theta^d(s) + \frac{G(s)}{1+G(s)H(s)}D(s)$$

Choosing $F(s) = 1/G(s)$, even though $G(s)$ is typically not proper we can implement precomputed time derivatives of the reference trajectory,

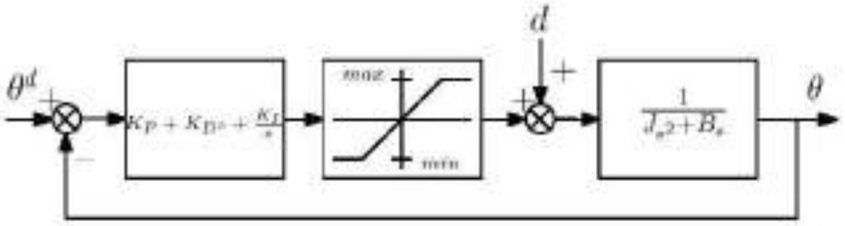
$$\Theta(s) = \Theta^d(s) + \frac{G(s)}{1+G(s)H(s)}D(s)$$

Control Design

Saturation of Input Magnitude and/or Rate of Change

ITMO UNIVERSITY

Second-order system with input saturation limiting the magnitude of the input signal:



The closed-loop system can be drastically affected by input saturation. Increasing the control action beyond saturation limit does not propagate to the plant.
We must check how the performance of the loop is influenced by input saturation

Control Design

Saturation of Input Magnitude and/or Rate of Change

ITMO UNIVERSITY

Typical signal limitations are the following NL functions:

- Saturation

$$u(t) = \mathrm{Sat}\left[\hat{u}(t)\right] = \begin{cases} u_{max}, & \text{if } \hat{u}(t) \geq u_{max} \\ \hat{u}(t), & \text{if } u_{min} \leq \hat{u}(t) \leq u_{max} \\ u_{min}, & \text{if } \hat{u}(t) \leq u_{min} \end{cases}$$

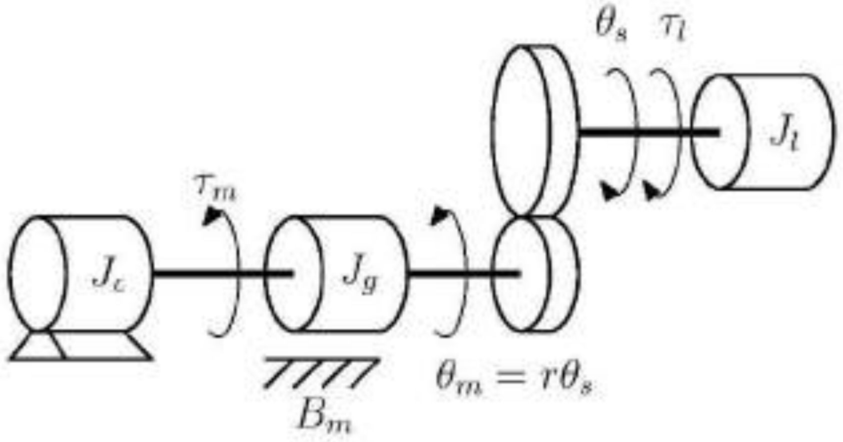
- Slew-Rate Limit

$$\frac{d}{dt}u(t) = \mathrm{Sat}\left[\frac{d}{dt}\hat{u}(t)\right] = \begin{cases} \sigma_{max}, & \text{if } \frac{d}{dt}\hat{u}(t) \geq \sigma_{max} \\ \frac{d}{dt}\hat{u}(t), & \text{if } \sigma_{min} \leq \frac{d}{dt}\hat{u}(t) \leq \sigma_{max} \\ \sigma_{min}, & \text{if } \frac{d}{dt}\hat{u}(t) \leq \sigma_{min} \end{cases}$$

Control Design

Gear Box Dynamics: Problems with Gear Boxes

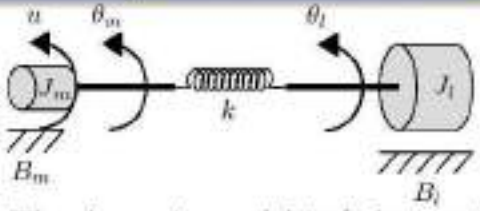
Schematic of the actuator/gear dynamics for a single link:



Lossless transmission, i.e. $\tau_m\dot{\theta}_m = \tau_s\dot{\theta}_s$, is impossible;
Gear box always introduces additional friction and delay;
Gear box can complicate the dynamics and limit the performance.

Control Design

Flexibility in Harmonic Drive Transmission



Model of joint flexibility k in a harmonic drive gear box.

The dynamic model is (where u is a control torque):

$$\begin{aligned} J_l\ddot{\theta}_l + B_l\dot{\theta}_l + k\left(\theta_l - \theta_m\right) &= 0 \\ J_m\ddot{\theta}_m + B_m\dot{\theta}_m - k\left(\theta_l - \theta_m\right) &= u \end{aligned}$$

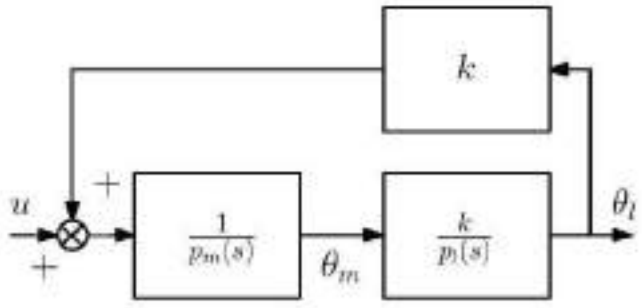
Compute the transfer functions relating the two variables and the control input:

$$\begin{aligned} \Theta_l(s) = \frac{k}{J_ls^2 + B_ls + k}\Theta_m(s) &= \frac{k}{p_l(s)}\Theta_m(s) \\ \Theta_m(s) = \frac{k\Theta_l(s) + U(s)}{J_ms^2 + B_ms + k} &= \frac{1}{p_m(s)}\left(k\Theta_l(s) + U(s)\right) \end{aligned}$$

Control Design

Flexibility in Harmonic Drive Transmission (2)

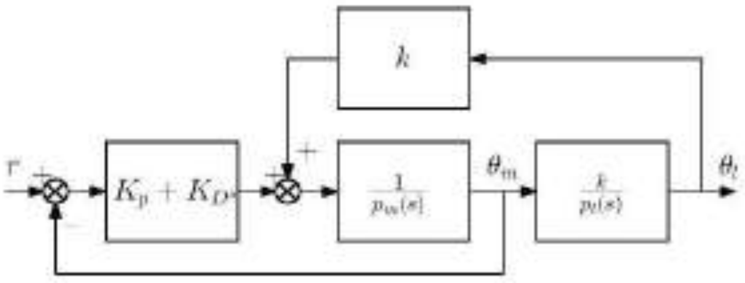
Block diagram of harmonic drive gear box:

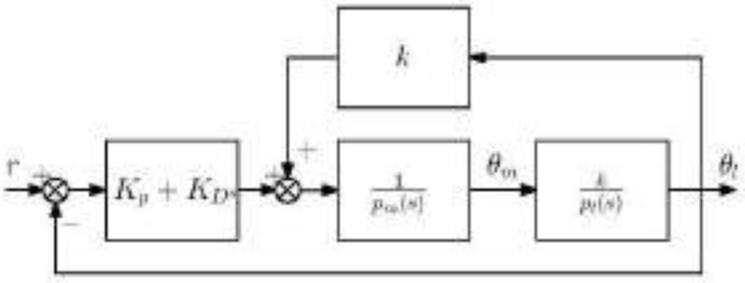


$$
\begin{aligned}
\Theta_l(s) &= \frac{k}{p_l(s)p_m(s) - k^2} U(s) \\
&= \frac{k}{J_m J_l s^4 + (J_l B_m + J_m B_l)s^3 + (k(J_m + J_l) + B_m B_l)s^2 + k(B_m + B_l)s} U(s) \\
&\approx \frac{k}{J_m J_l s^4 + k(J_m + J_l)s^2} U(s)
\end{aligned}
$$

Control Design

PD controller with θ_m vs. θ_l feedbacks



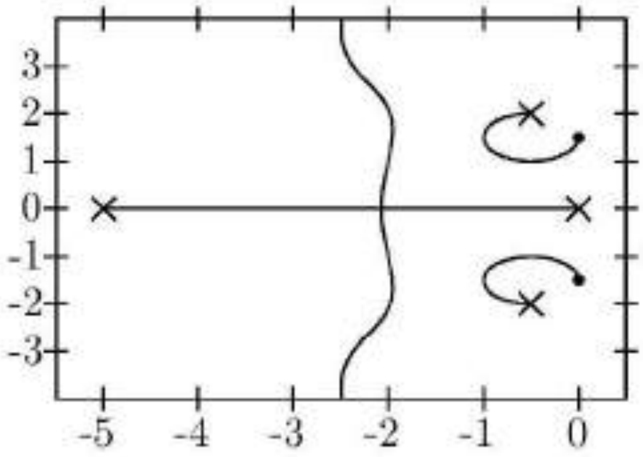


Control Design

PD controller with θ_m or θ_l feedbacks

ITMO UNIVERSITY

Root locus of the closed loop system with θ_m-feedback



The PD controller was chosen as

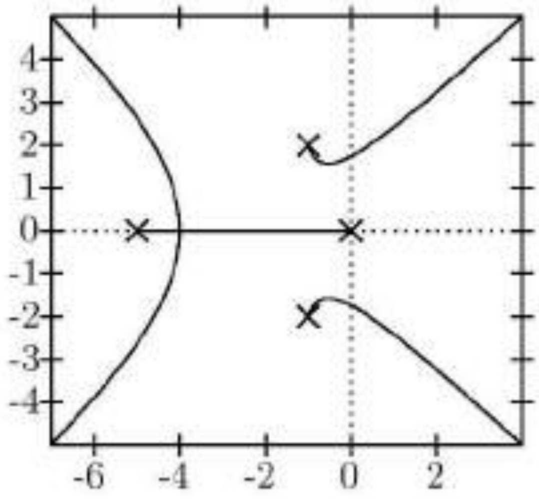
$$PD = K_P + K_D s = K(s+a)$$

Control Design

PD controller with θ_m or θ_l feedbacks

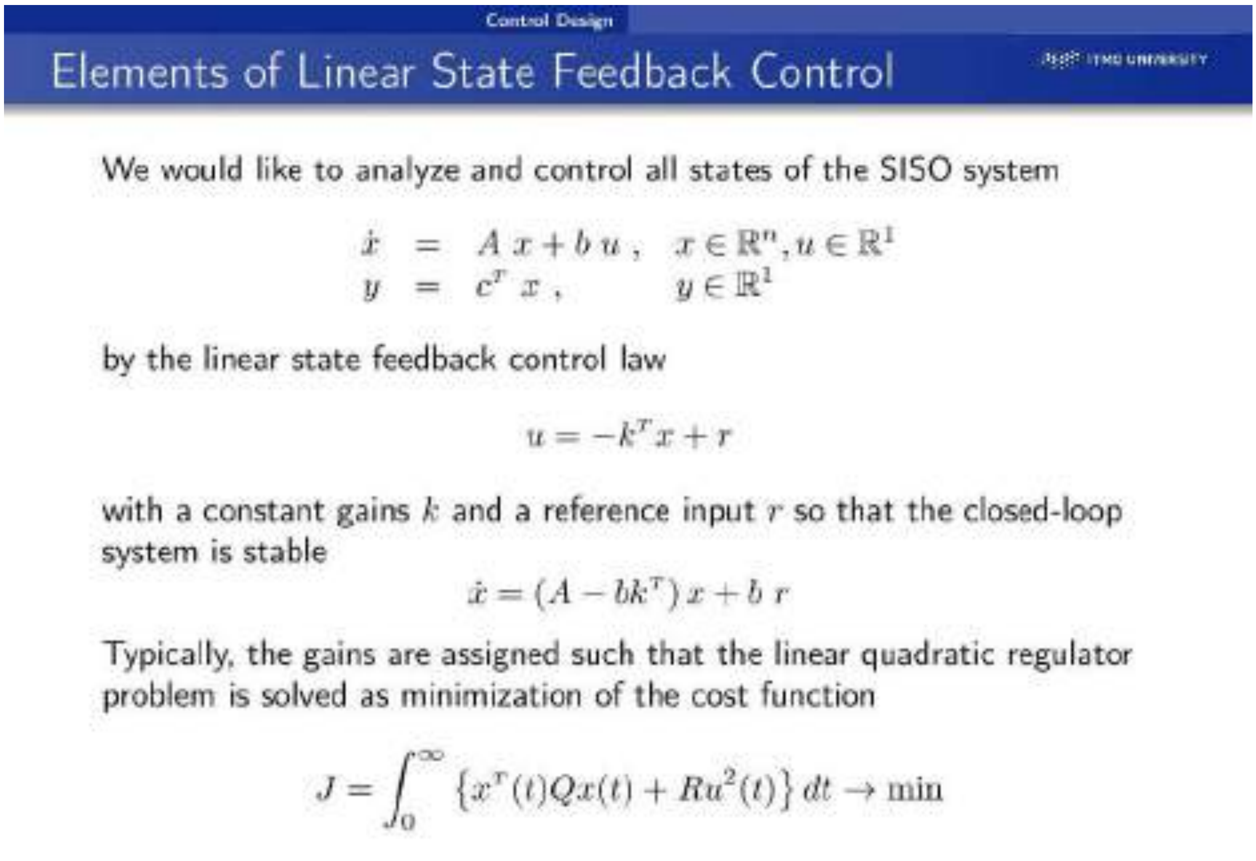
ITMO UNIVERSITY

Root locus of the closed loop system with θ_l-feedback



The PD controller was chosen as

$$PD = K_P + K_D s = K(s+a)$$



Control Design

Elements of Linear State Feedback Control

We would like to analyze and control all states of the SISO system

$$\begin{aligned} \dot{x} &= A\,x + b\,u\,, \quad x \in \mathbb{R}^n, u \in \mathbb{R}^1 \\ y &= c^T\,x\,, \qquad\quad y \in \mathbb{R}^1 \end{aligned}$$

by the linear state feedback control law

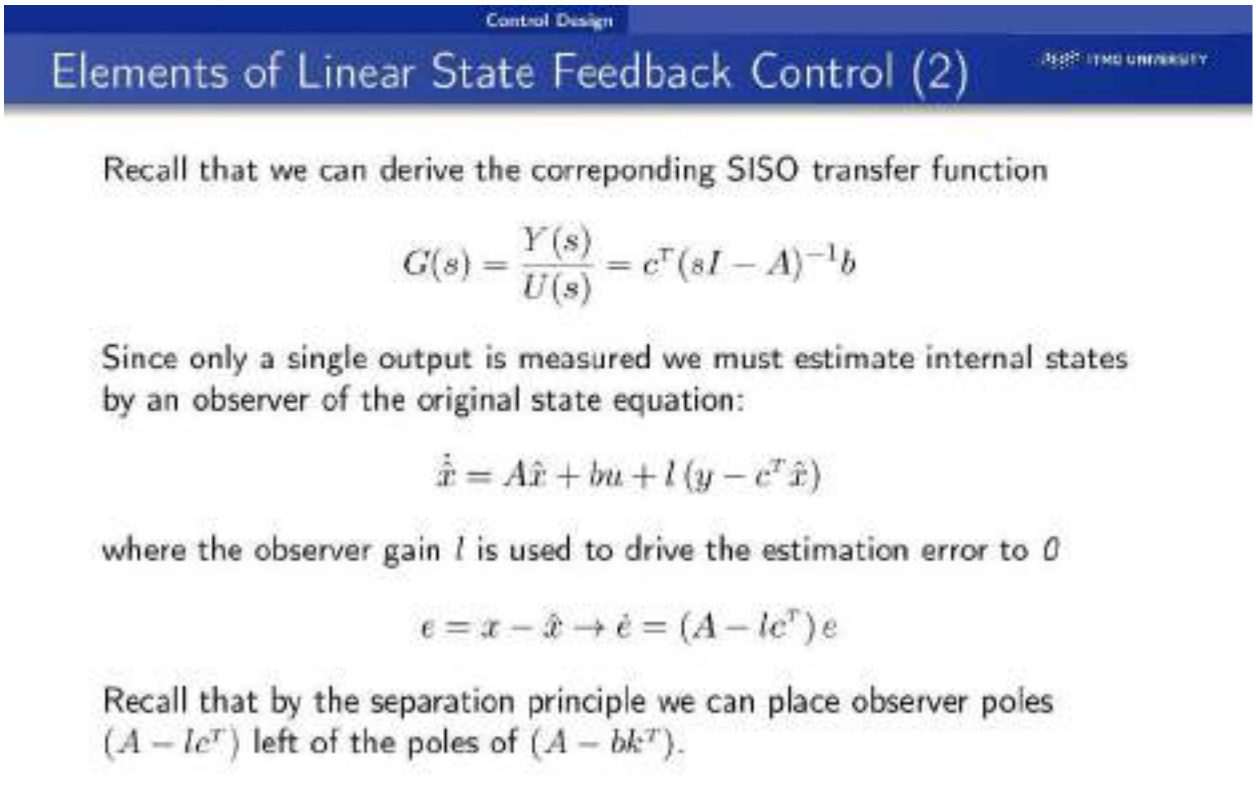
$$u = -k^T x + r$$

with a constant gains k and a reference input r so that the closed-loop system is stable

$$\dot{x} = (A - bk^T)\,x + b\,r$$

Typically, the gains are assigned such that the linear quadratic regulator problem is solved as minimization of the cost function

$$J = \int_0^\infty \left\{x^T(t)Qx(t) + Ru^2(t)\right\} dt \to \min$$



Control Design

Elements of Linear State Feedback Control (2)

Recall that we can derive the correponding SISO transfer function

$$G(s) = \frac{Y(s)}{U(s)} = c^T(sI - A)^{-1}b$$

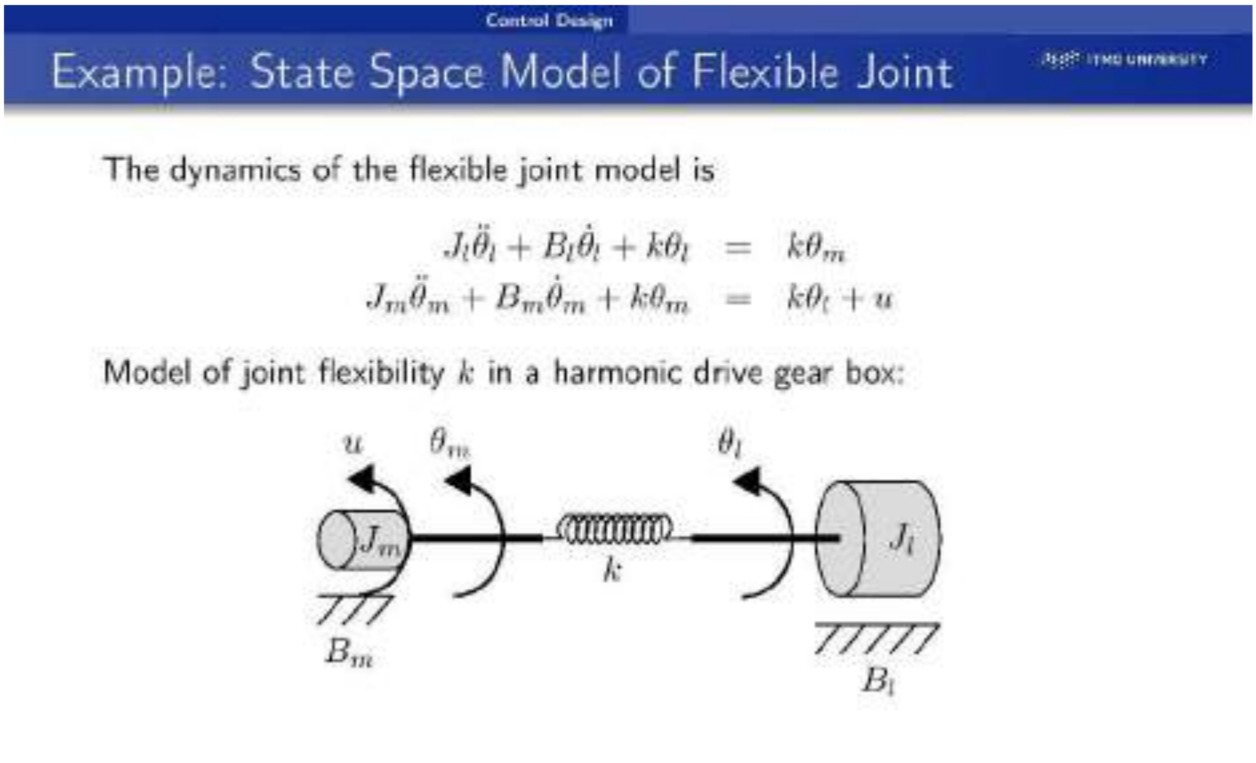
Since only a single output is measured we must estimate internal states by an observer of the original state equation:

$$\dot{\hat{x}} = A\hat{x} + bu + l\,(y - c^T\hat{x})$$

where the observer gain l is used to drive the estimation error to *0*

$$e = x - \hat{x} \to \dot{e} = (A - lc^T)\,e$$

Recall that by the separation principle we can place observer poles $(A - lc^T)$ left of the poles of $(A - bk^T)$.



Control Design

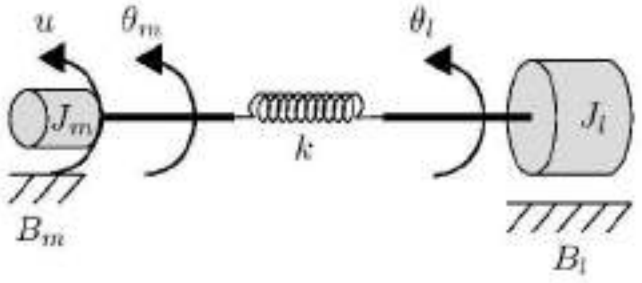
Example: State Space Model of Flexible Joint

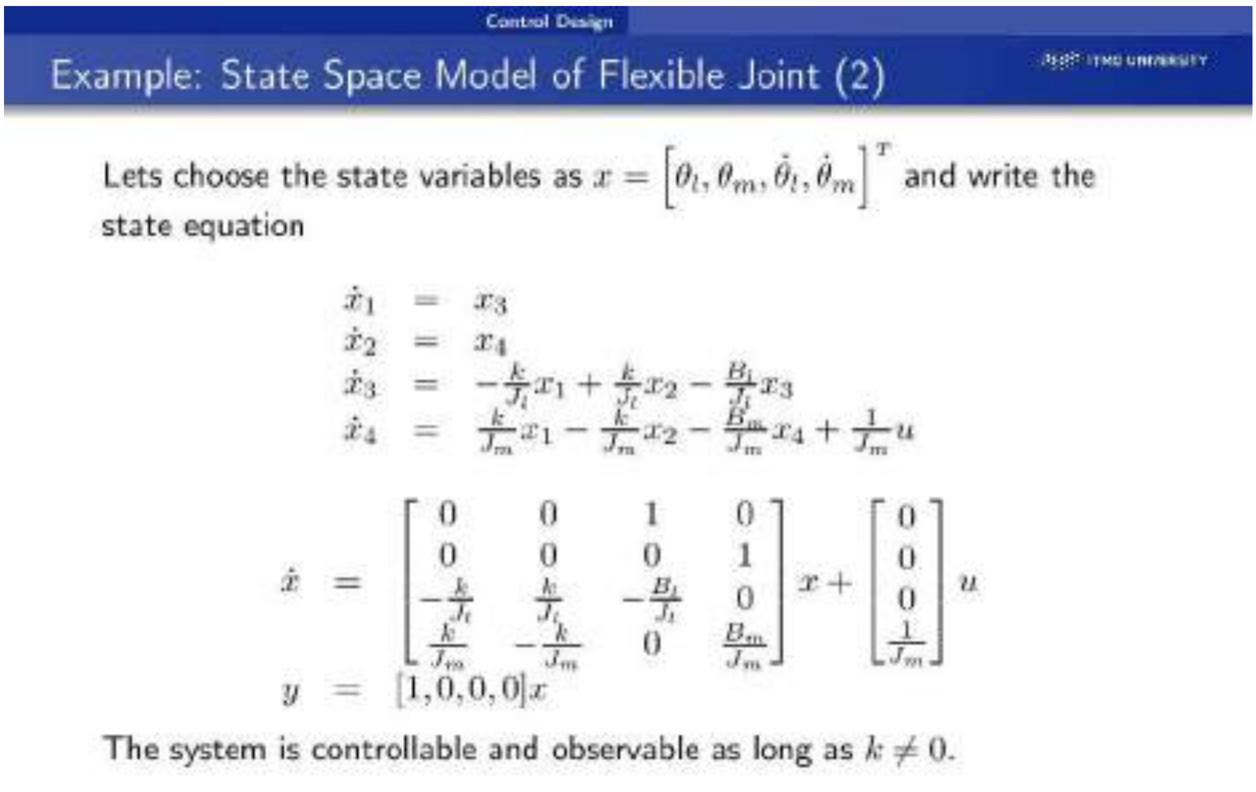
ITMO UNIVERSITY

The dynamics of the flexible joint model is

$$\begin{aligned} J_l\ddot{\theta}_l + B_l\dot{\theta}_l + k\theta_l &= k\theta_m \\ J_m\ddot{\theta}_m + B_m\dot{\theta}_m + k\theta_m &= k\theta_l + u \end{aligned}$$

Model of joint flexibility k in a harmonic drive gear box:





Control Design

Example: State Space Model of Flexible Joint (2)

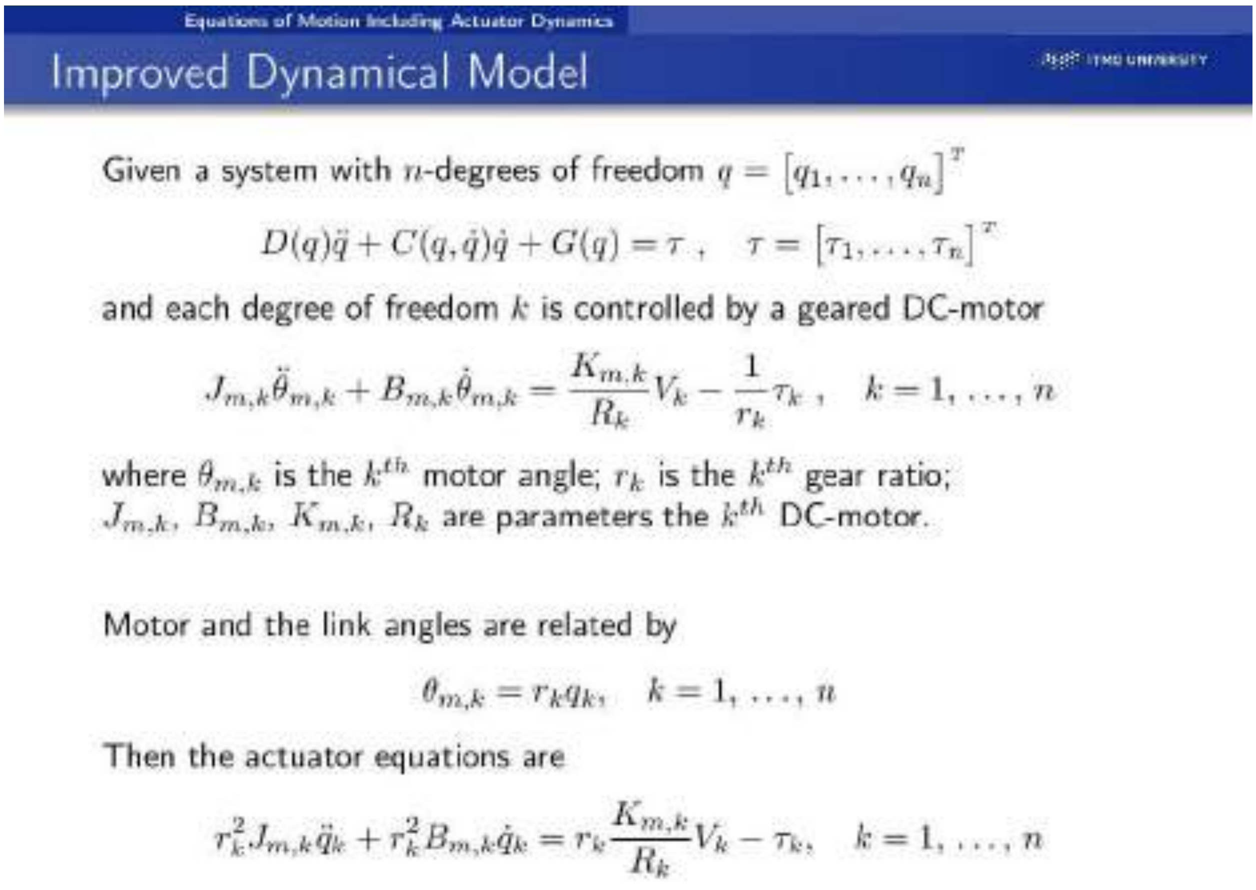
ITMO UNIVERSITY

Lets choose the state variables as $x = \left[\theta_l, \theta_m, \dot{\theta}_l, \dot{\theta}_m\right]^T$ and write the state equation

$$\begin{aligned} \dot{x}_1 &= x_3 \\ \dot{x}_2 &= x_4 \\ \dot{x}_3 &= -\frac{k}{J_l}x_1 + \frac{k}{J_l}x_2 - \frac{B_l}{J_l}x_3 \\ \dot{x}_4 &= \frac{k}{J_m}x_1 - \frac{k}{J_m}x_2 - \frac{B_m}{J_m}x_4 + \frac{1}{J_m}u \end{aligned}$$

$$\begin{aligned} \dot{x} &= \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{k}{J_l} & \frac{k}{J_l} & -\frac{B_l}{J_l} & 0 \\ \frac{k}{J_m} & -\frac{k}{J_m} & 0 & \frac{B_m}{J_m} \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{1}{J_m} \end{bmatrix} u \\ y &= [1, 0, 0, 0]x \end{aligned}$$

The system is controllable and observable as long as $k \neq 0$.



Equations of Motion Including Actuator Dynamics

Improved Dynamical Model

ITMO UNIVERSITY

Given a system with n-degrees of freedom $q = \left[q_1, \ldots, q_n\right]^T$

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau\,, \quad \tau = \left[\tau_1, \ldots, \tau_n\right]^T$$

and each degree of freedom k is controlled by a geared DC-motor

$$J_{m,k}\ddot{\theta}_{m,k} + B_{m,k}\dot{\theta}_{m,k} = \frac{K_{m,k}}{R_k}V_k - \frac{1}{r_k}\tau_k\,, \quad k = 1, \ldots, n$$

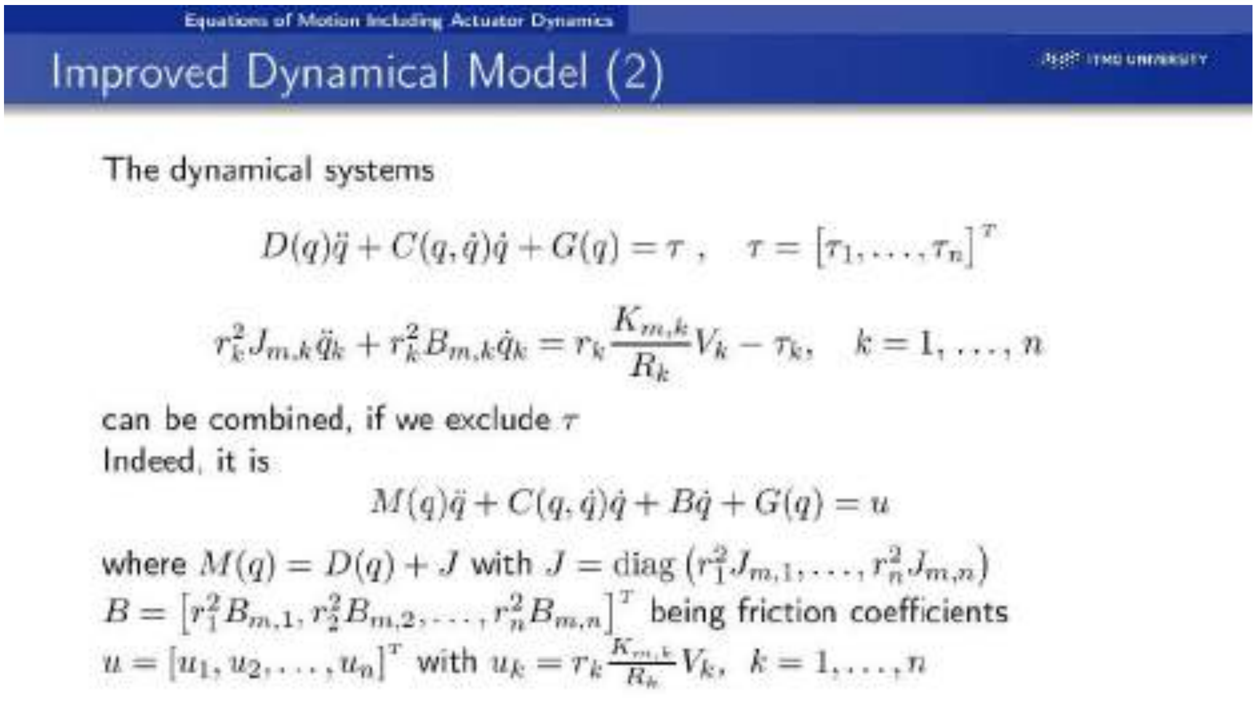
where $\theta_{m,k}$ is the k^{th} motor angle; r_k is the k^{th} gear ratio;
$J_{m,k}$, $B_{m,k}$, $K_{m,k}$, R_k are parameters the k^{th} DC-motor.

Motor and the link angles are related by

$$\theta_{m,k} = r_k q_k, \quad k = 1, \ldots, n$$

Then the actuator equations are

$$r_k^2 J_{m,k}\ddot{q}_k + r_k^2 B_{m,k}\dot{q}_k = r_k\frac{K_{m,k}}{R_k}V_k - \tau_k, \quad k = 1, \ldots, n$$



Equations of Motion Including Actuator Dynamics

Improved Dynamical Model (2)

ITMO UNIVERSITY

The dynamical systems

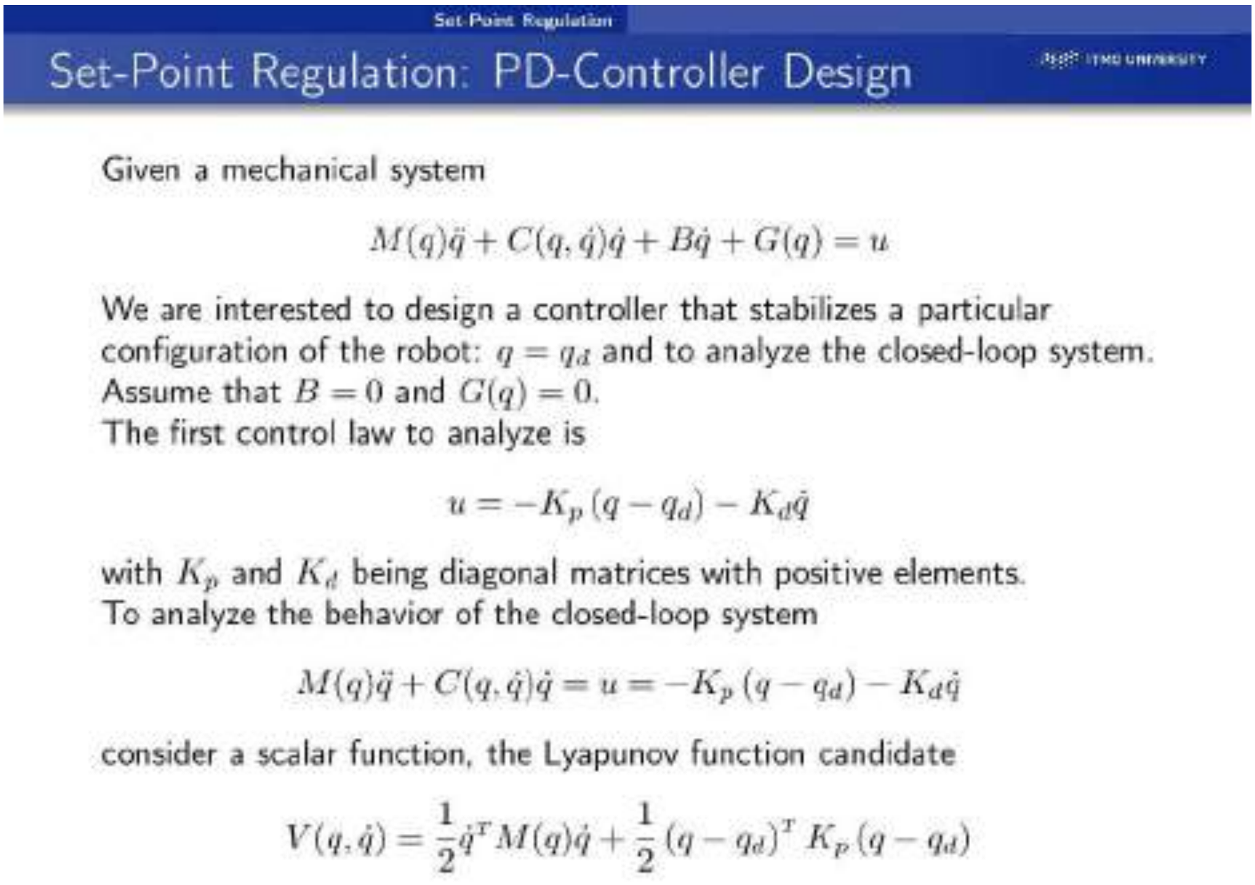
$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau\,, \quad \tau = \left[\tau_1, \ldots, \tau_n\right]^T$$

$$r_k^2 J_{m,k}\ddot{q}_k + r_k^2 B_{m,k}\dot{q}_k = r_k\frac{K_{m,k}}{R_k}V_k - \tau_k, \quad k = 1, \ldots, n$$

can be combined, if we exclude τ
Indeed, it is

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + B\dot{q} + G(q) = u$$

where $M(q) = D(q) + J$ with $J = \operatorname{diag}\left(r_1^2 J_{m,1}, \ldots, r_n^2 J_{m,n}\right)$
$B = \left[r_1^2 B_{m,1}, r_2^2 B_{m,2}, \ldots, r_n^2 B_{m,n}\right]^T$ being friction coefficients
$u = \left[u_1, u_2, \ldots, u_n\right]^T$ with $u_k = r_k\frac{K_{m,k}}{R_k}V_k$, $k = 1, \ldots, n$



Set-Point Regulation

Set-Point Regulation: PD-Controller Design

Given a mechanical system

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + B\dot{q} + G(q) = u$$

We are interested to design a controller that stabilizes a particular configuration of the robot: $q = q_d$ and to analyze the closed-loop system.
Assume that $B = 0$ and $G(q) = 0$.
The first control law to analyze is

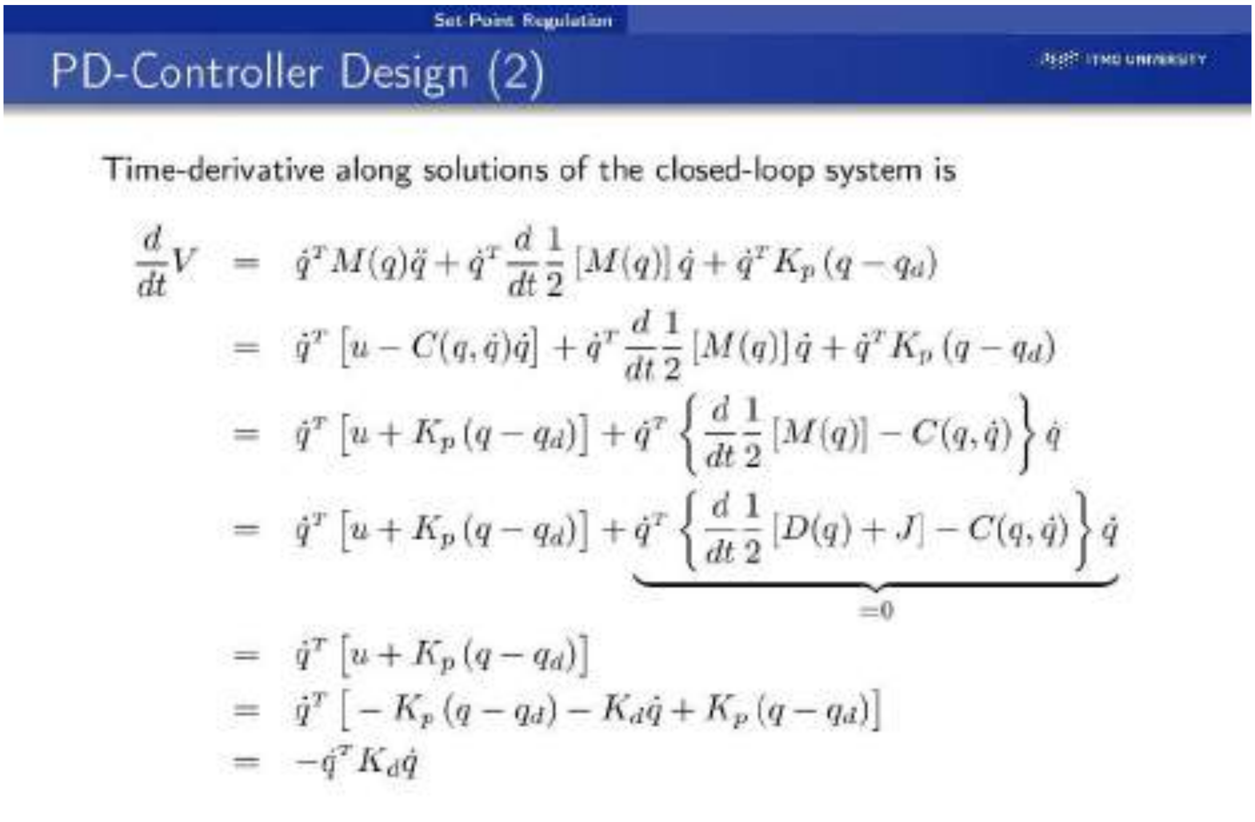
$$u = -K_p\,(q - q_d) - K_d\dot{q}$$

with K_p and K_d being diagonal matrices with positive elements.
To analyze the behavior of the closed-loop system

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} = u = -K_p\,(q - q_d) - K_d\dot{q}$$

consider a scalar function, the Lyapunov function candidate

$$V(q,\dot{q}) = \frac{1}{2}\dot{q}^T M(q)\dot{q} + \frac{1}{2}\,(q - q_d)^T\, K_p\,(q - q_d)$$

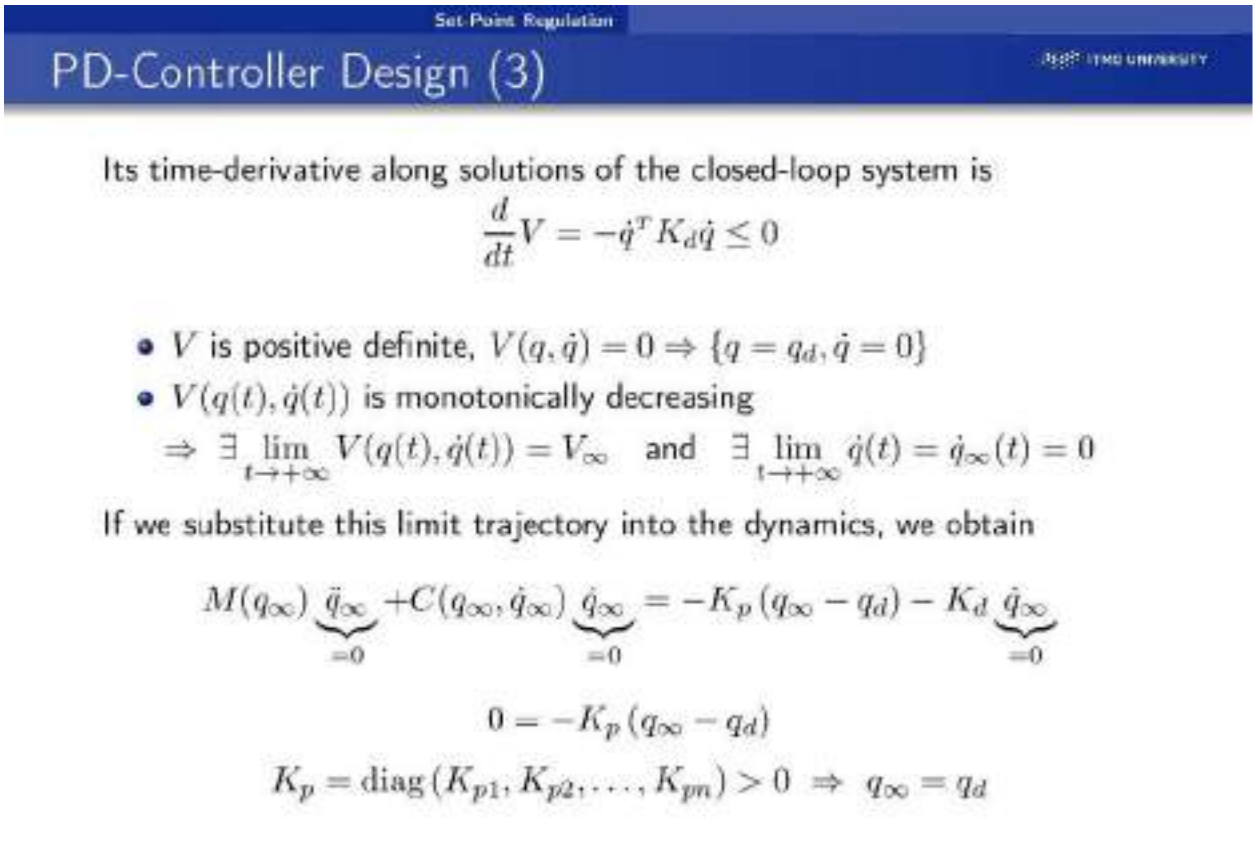


Set-Point Regulation

PD-Controller Design (2)

Time-derivative along solutions of the closed-loop system is

$$\begin{aligned}
\frac{d}{dt}V &= \dot{q}^T M(q)\ddot{q} + \dot{q}^T\frac{d}{dt}\frac{1}{2}\,[M(q)]\,\dot{q} + \dot{q}^T K_p\,(q - q_d) \\
&= \dot{q}^T\left[u - C(q,\dot{q})\dot{q}\right] + \dot{q}^T\frac{d}{dt}\frac{1}{2}\,[M(q)]\,\dot{q} + \dot{q}^T K_p\,(q - q_d) \\
&= \dot{q}^T\left[u + K_p\,(q - q_d)\right] + \dot{q}^T\left\{\frac{d}{dt}\frac{1}{2}\,[M(q)] - C(q,\dot{q})\right\}\dot{q} \\
&= \dot{q}^T\left[u + K_p\,(q - q_d)\right] + \underbrace{\dot{q}^T\left\{\frac{d}{dt}\frac{1}{2}\,[D(q) + J] - C(q,\dot{q})\right\}\dot{q}}_{=0} \\
&= \dot{q}^T\left[u + K_p\,(q - q_d)\right] \\
&= \dot{q}^T\left[-K_p\,(q - q_d) - K_d\dot{q} + K_p\,(q - q_d)\right] \\
&= -\dot{q}^T K_d\dot{q}
\end{aligned}$$



Set-Point Regulation

PD-Controller Design (3)

ITMO UNIVERSITY

Its time-derivative along solutions of the closed-loop system is

$$\frac{d}{dt}V = -\dot{q}^T K_d \dot{q} \leq 0$$

- V is positive definite, $V(q,\dot{q}) = 0 \Rightarrow \{q = q_d, \dot{q} = 0\}$
- $V(q(t),\dot{q}(t))$ is monotonically decreasing

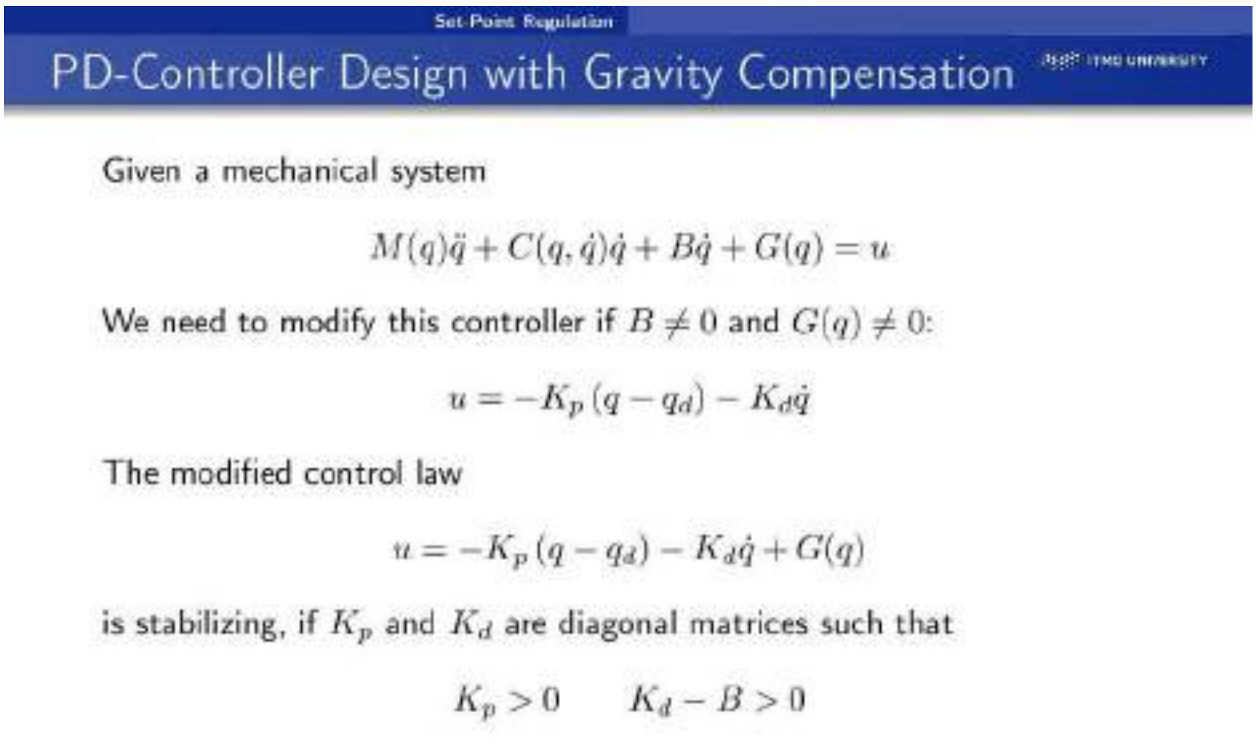
$\Rightarrow \exists \lim\limits_{t\to+\infty} V(q(t),\dot{q}(t)) = V_\infty$ and $\exists \lim\limits_{t\to+\infty} \dot{q}(t) = \dot{q}_\infty(t) = 0$

If we substitute this limit trajectory into the dynamics, we obtain

$$M(q_\infty)\underbrace{\ddot{q}_\infty}_{=0} + C(q_\infty,\dot{q}_\infty)\underbrace{\dot{q}_\infty}_{=0} = -K_p(q_\infty - q_d) - K_d\underbrace{\dot{q}_\infty}_{=0}$$

$$0 = -K_p(q_\infty - q_d)$$

$$K_p = \mathrm{diag}(K_{p1}, K_{p2}, \ldots, K_{pn}) > 0 \;\Rightarrow\; q_\infty = q_d$$



Set-Point Regulation

PD-Controller Design with Gravity Compensation

ITMO UNIVERSITY

Given a mechanical system

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + B\dot{q} + G(q) = u$$

We need to modify this controller if $B \neq 0$ and $G(q) \neq 0$:

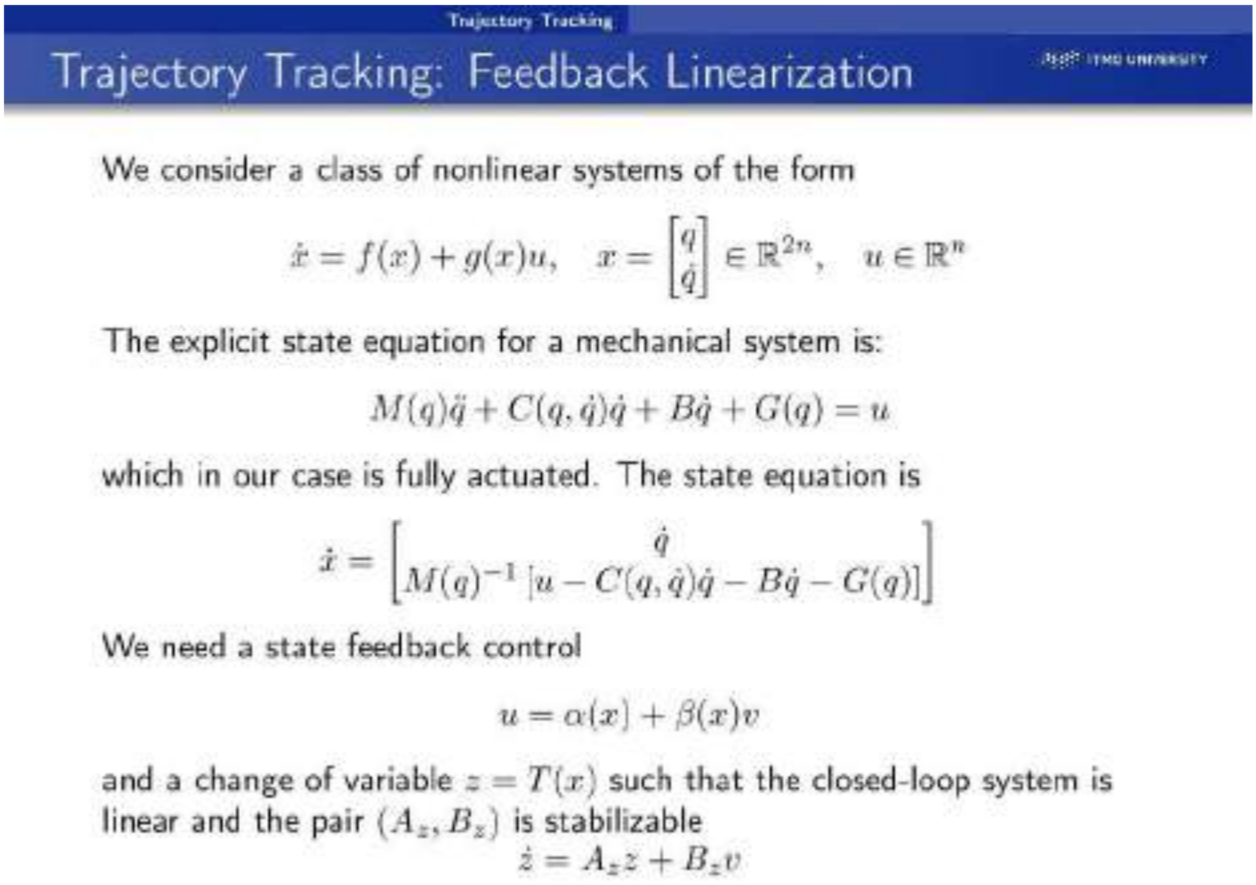
$$u = -K_p(q - q_d) - K_d\dot{q}$$

The modified control law

$$u = -K_p(q - q_d) - K_d\dot{q} + G(q)$$

is stabilizing, if K_p and K_d are diagonal matrices such that

$$K_p > 0 \qquad K_d - B > 0$$



Trajectory Tracking

Trajectory Tracking: Feedback Linearization

We consider a class of nonlinear systems of the form

$$\dot{x} = f(x) + g(x)u, \quad x = \begin{bmatrix} q \\ \dot{q} \end{bmatrix} \in \mathbb{R}^{2n}, \quad u \in \mathbb{R}^n$$

The explicit state equation for a mechanical system is:

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + B\dot{q} + G(q) = u$$

which in our case is fully actuated. The state equation is

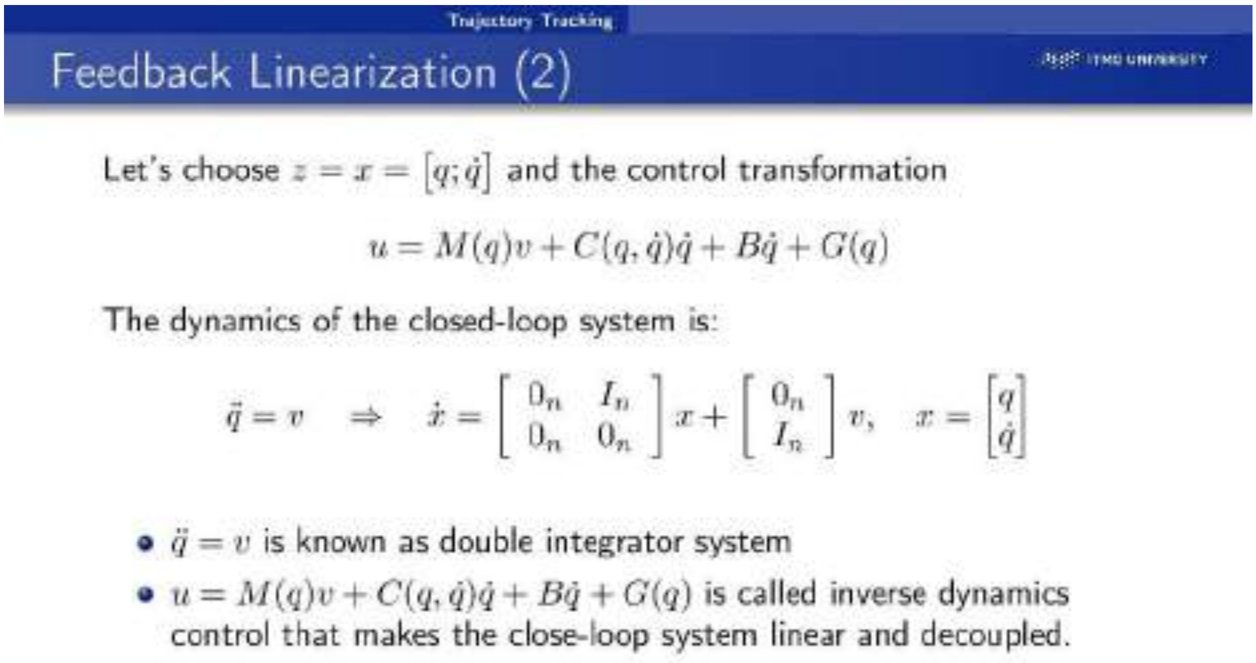
$$\dot{x} = \begin{bmatrix} \dot{q} \\ M(q)^{-1}\left[u - C(q,\dot{q})\dot{q} - B\dot{q} - G(q)\right] \end{bmatrix}$$

We need a state feedback control

$$u = \alpha(x) + \beta(x)v$$

and a change of variable $z = T(x)$ such that the closed-loop system is linear and the pair (A_z, B_z) is stabilizable

$$\dot{z} = A_z z + B_z v$$



Trajectory Tracking

Feedback Linearization (2)

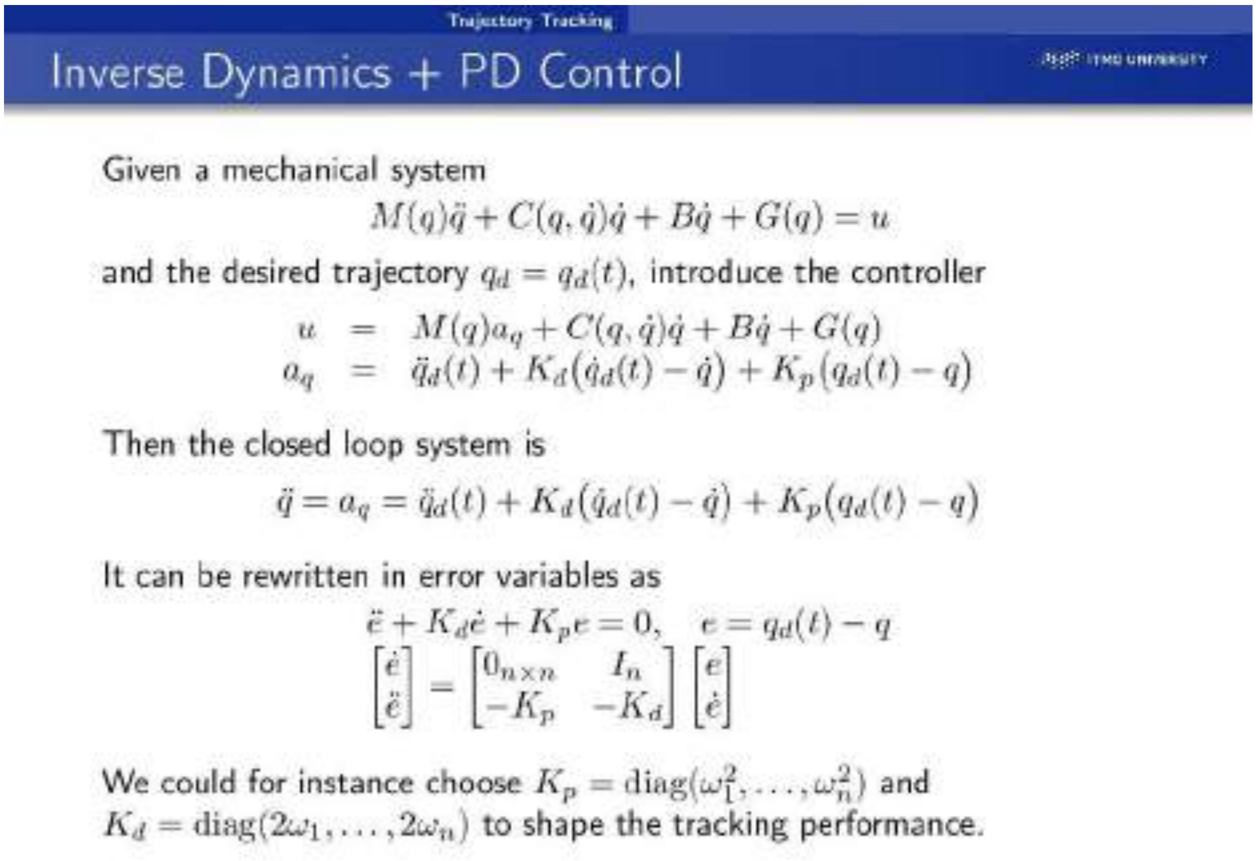
Let's choose $z = x = [q; \dot{q}]$ and the control transformation

$$u = M(q)v + C(q,\dot{q})\dot{q} + B\dot{q} + G(q)$$

The dynamics of the closed-loop system is:

$$\ddot{q} = v \quad \Rightarrow \quad \dot{x} = \begin{bmatrix} 0_n & I_n \\ 0_n & 0_n \end{bmatrix} x + \begin{bmatrix} 0_n \\ I_n \end{bmatrix} v, \quad x = \begin{bmatrix} q \\ \dot{q} \end{bmatrix}$$

- $\ddot{q} = v$ is known as double integrator system
- $u = M(q)v + C(q,\dot{q})\dot{q} + B\dot{q} + G(q)$ is called inverse dynamics control that makes the close-loop system linear and decoupled.



Trajectory Tracking

Inverse Dynamics + PD Control

ITMO UNIVERSITY

Given a mechanical system

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + B\dot{q} + G(q) = u$$

and the desired trajectory $q_d = q_d(t)$, introduce the controller

$$\begin{aligned} u &= M(q)a_q + C(q,\dot{q})\dot{q} + B\dot{q} + G(q) \\ a_q &= \ddot{q}_d(t) + K_d\big(\dot{q}_d(t) - \dot{q}\big) + K_p\big(q_d(t) - q\big) \end{aligned}$$

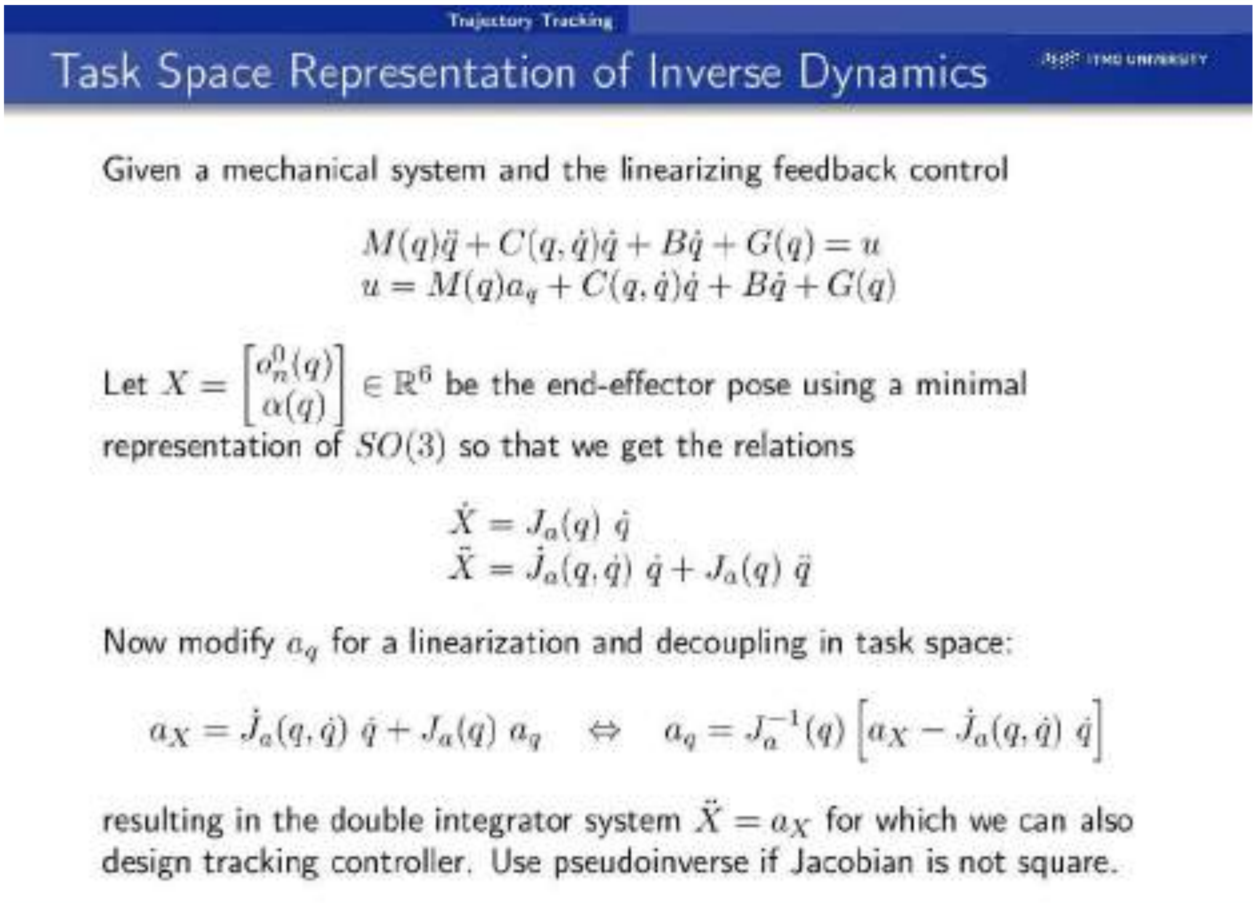
Then the closed loop system is

$$\ddot{q} = a_q = \ddot{q}_d(t) + K_d\big(\dot{q}_d(t) - \dot{q}\big) + K_p\big(q_d(t) - q\big)$$

It can be rewritten in error variables as

$$\ddot{e} + K_d\dot{e} + K_p e = 0, \quad e = q_d(t) - q$$
$$\begin{bmatrix} \dot{e} \\ \ddot{e} \end{bmatrix} = \begin{bmatrix} 0_{n\times n} & I_n \\ -K_p & -K_d \end{bmatrix} \begin{bmatrix} e \\ \dot{e} \end{bmatrix}$$

We could for instance choose $K_p = \mathrm{diag}(\omega_1^2, \ldots, \omega_n^2)$ and $K_d = \mathrm{diag}(2\omega_1, \ldots, 2\omega_n)$ to shape the tracking performance.



Trajectory Tracking

Task Space Representation of Inverse Dynamics

ITMO UNIVERSITY

Given a mechanical system and the linearizing feedback control

$$\begin{aligned} M(q)\ddot{q} + C(q,\dot{q})\dot{q} + B\dot{q} + G(q) = u \\ u = M(q)a_q + C(q,\dot{q})\dot{q} + B\dot{q} + G(q) \end{aligned}$$

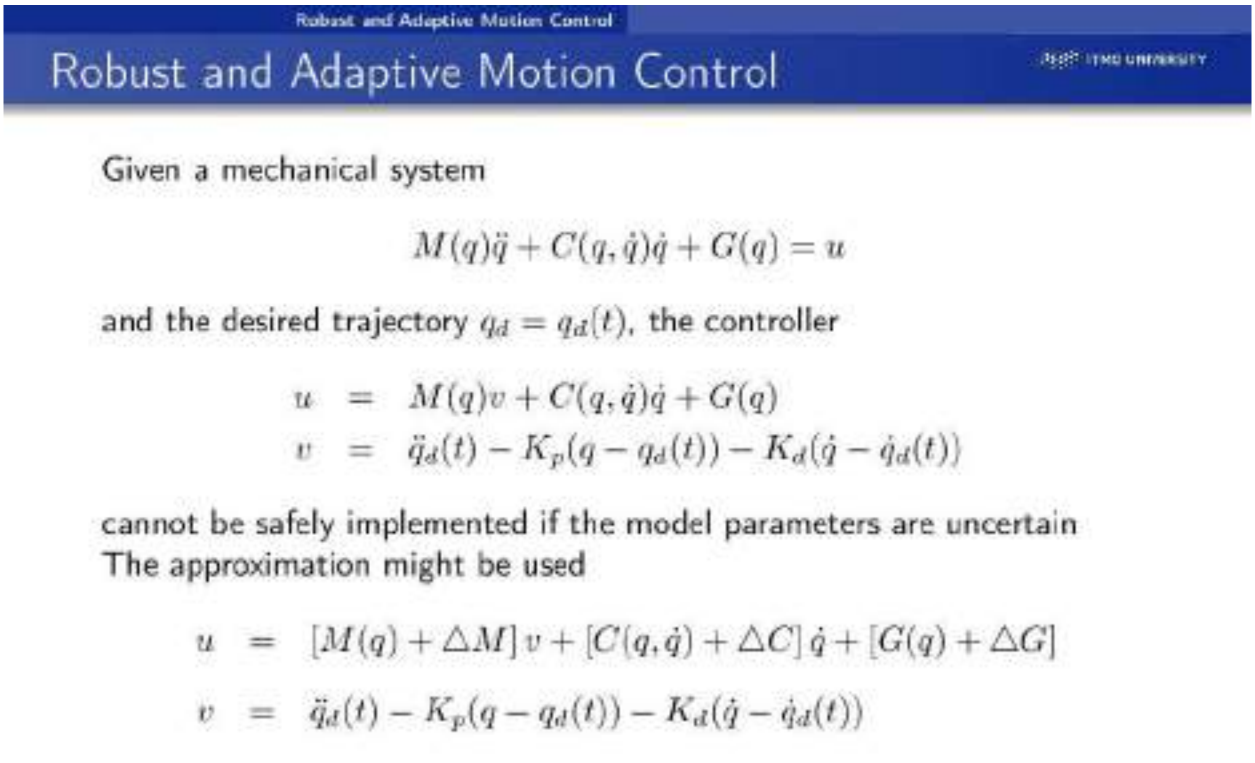
Let $X = \begin{bmatrix} o_n^0(q) \\ \alpha(q) \end{bmatrix} \in \mathbb{R}^6$ be the end-effector pose using a minimal representation of $SO(3)$ so that we get the relations

$$\begin{aligned} \dot{X} &= J_a(q)\ \dot{q} \\ \ddot{X} &= \dot{J}_a(q,\dot{q})\ \dot{q} + J_a(q)\ \ddot{q} \end{aligned}$$

Now modify a_q for a linearization and decoupling in task space:

$$a_X = \dot{J}_a(q,\dot{q})\ \dot{q} + J_a(q)\ a_q \quad \Leftrightarrow \quad a_q = J_a^{-1}(q)\left[a_X - \dot{J}_a(q,\dot{q})\ \dot{q}\right]$$

resulting in the double integrator system $\ddot{X} = a_X$ for which we can also design tracking controller. Use pseudoinverse if Jacobian is not square.



Robust and Adaptive Motion Control

Robust and Adaptive Motion Control

Given a mechanical system

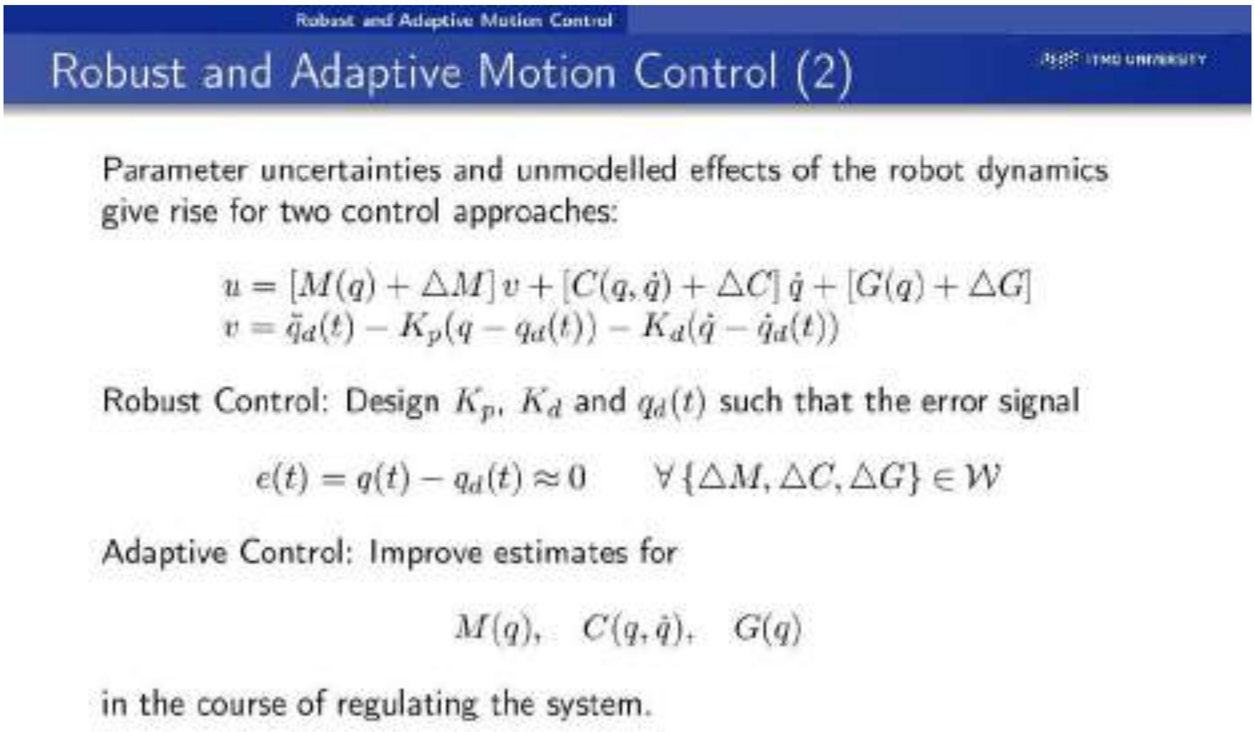
$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = u$$

and the desired trajectory $q_d = q_d(t)$, the controller

$$\begin{aligned} u &= M(q)v + C(q,\dot{q})\dot{q} + G(q) \\ v &= \ddot{q}_d(t) - K_p(q - q_d(t)) - K_d(\dot{q} - \dot{q}_d(t)) \end{aligned}$$

cannot be safely implemented if the model parameters are uncertain
The approximation might be used

$$\begin{aligned} u &= [M(q) + \triangle M]\, v + [C(q,\dot{q}) + \triangle C]\, \dot{q} + [G(q) + \triangle G] \\ v &= \ddot{q}_d(t) - K_p(q - q_d(t)) - K_d(\dot{q} - \dot{q}_d(t)) \end{aligned}$$



Robust and Adaptive Motion Control

Robust and Adaptive Motion Control (2)

Parameter uncertainties and unmodelled effects of the robot dynamics give rise for two control approaches:

$$\begin{aligned} u &= [M(q) + \triangle M]\, v + [C(q,\dot{q}) + \triangle C]\, \dot{q} + [G(q) + \triangle G] \\ v &= \ddot{q}_d(t) - K_p(q - q_d(t)) - K_d(\dot{q} - \dot{q}_d(t)) \end{aligned}$$

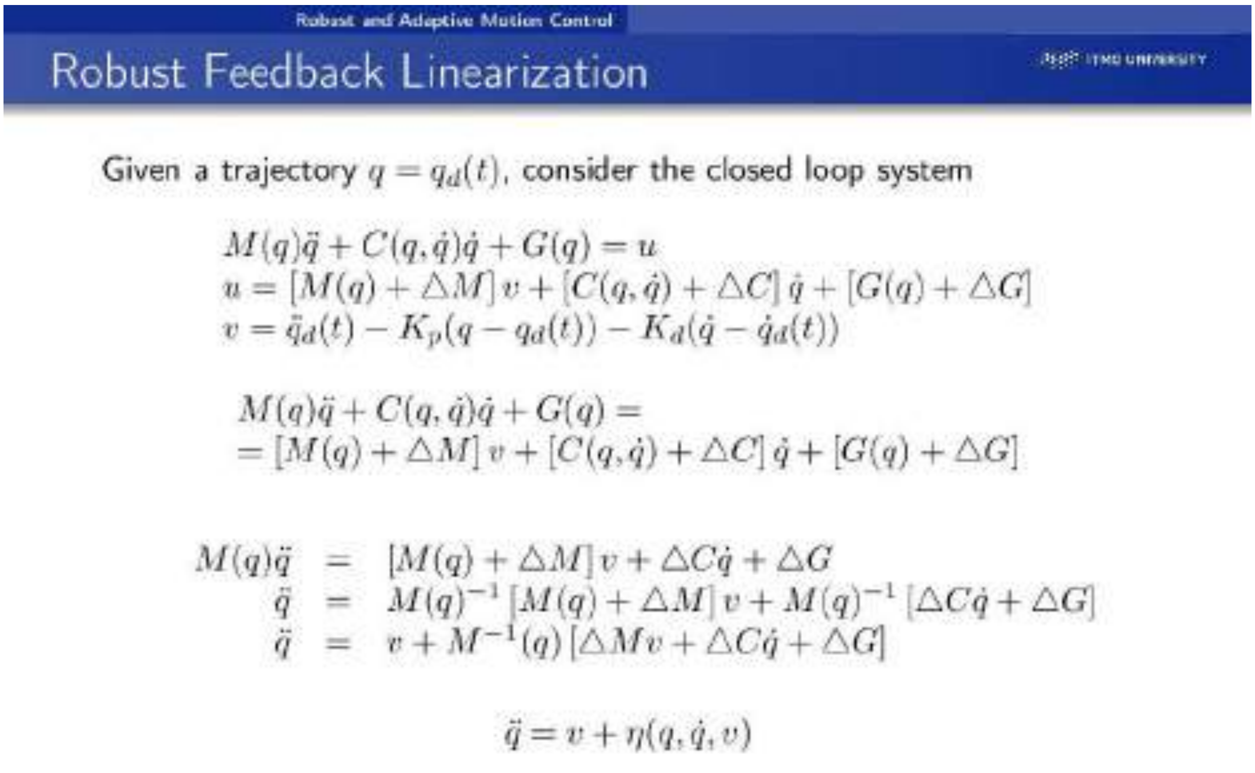
Robust Control: Design K_p, K_d and $q_d(t)$ such that the error signal

$$e(t) = q(t) - q_d(t) \approx 0 \qquad \forall \{\triangle M, \triangle C, \triangle G\} \in \mathcal{W}$$

Adaptive Control: Improve estimates for

$$M(q), \quad C(q,\dot{q}), \quad G(q)$$

in the course of regulating the system.



Robust and Adaptive Motion Control

Robust Feedback Linearization

ITMO UNIVERSITY

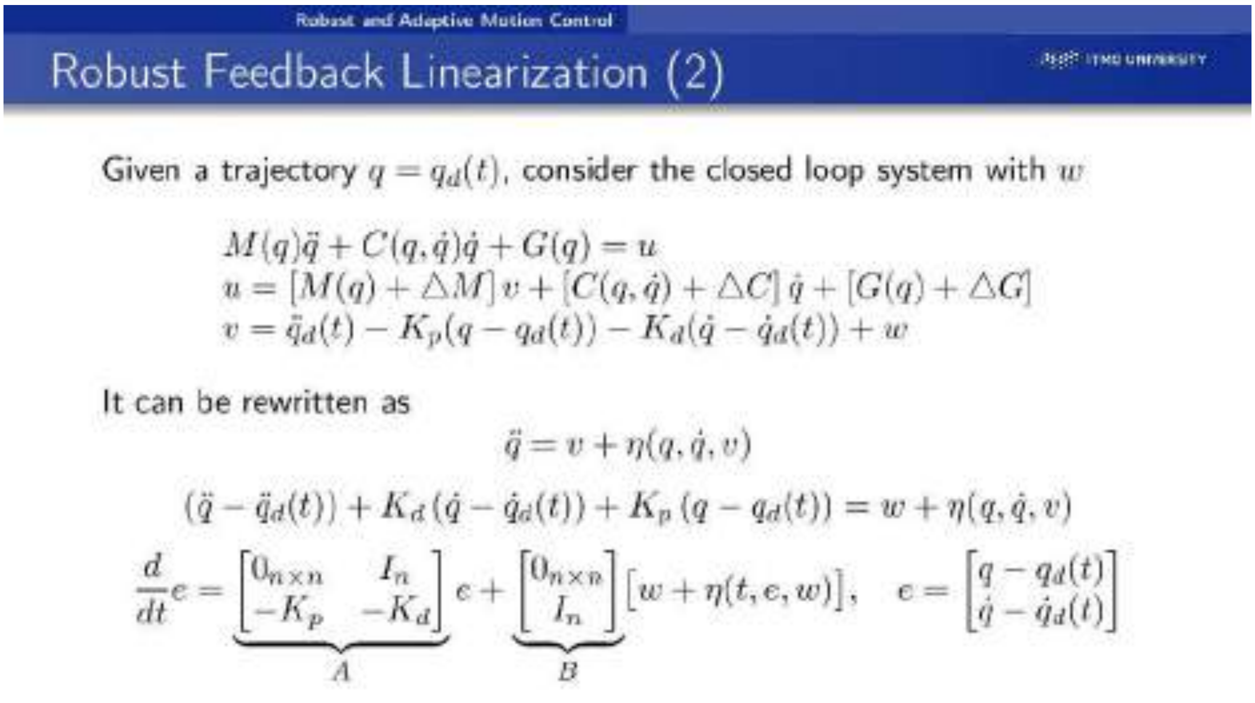
Given a trajectory $q = q_d(t)$, consider the closed loop system

$$\begin{aligned}&M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = u\\&u = [M(q) + \triangle M]\,v + [C(q,\dot{q}) + \triangle C]\,\dot{q} + [G(q) + \triangle G]\\&v = \ddot{q}_d(t) - K_p(q - q_d(t)) - K_d(\dot{q} - \dot{q}_d(t))\end{aligned}$$

$$\begin{aligned}&M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) =\\&= [M(q) + \triangle M]\,v + [C(q,\dot{q}) + \triangle C]\,\dot{q} + [G(q) + \triangle G]\end{aligned}$$

$$\begin{aligned}M(q)\ddot{q} &= [M(q) + \triangle M]\,v + \triangle C\dot{q} + \triangle G\\\ddot{q} &= M(q)^{-1}[M(q) + \triangle M]\,v + M(q)^{-1}[\triangle C\dot{q} + \triangle G]\\\ddot{q} &= v + M^{-1}(q)\,[\triangle M v + \triangle C\dot{q} + \triangle G]\end{aligned}$$

$$\ddot{q} = v + \eta(q,\dot{q},v)$$



Robust and Adaptive Motion Control

Robust Feedback Linearization (2)

ITMO UNIVERSITY

Given a trajectory $q = q_d(t)$, consider the closed loop system with w

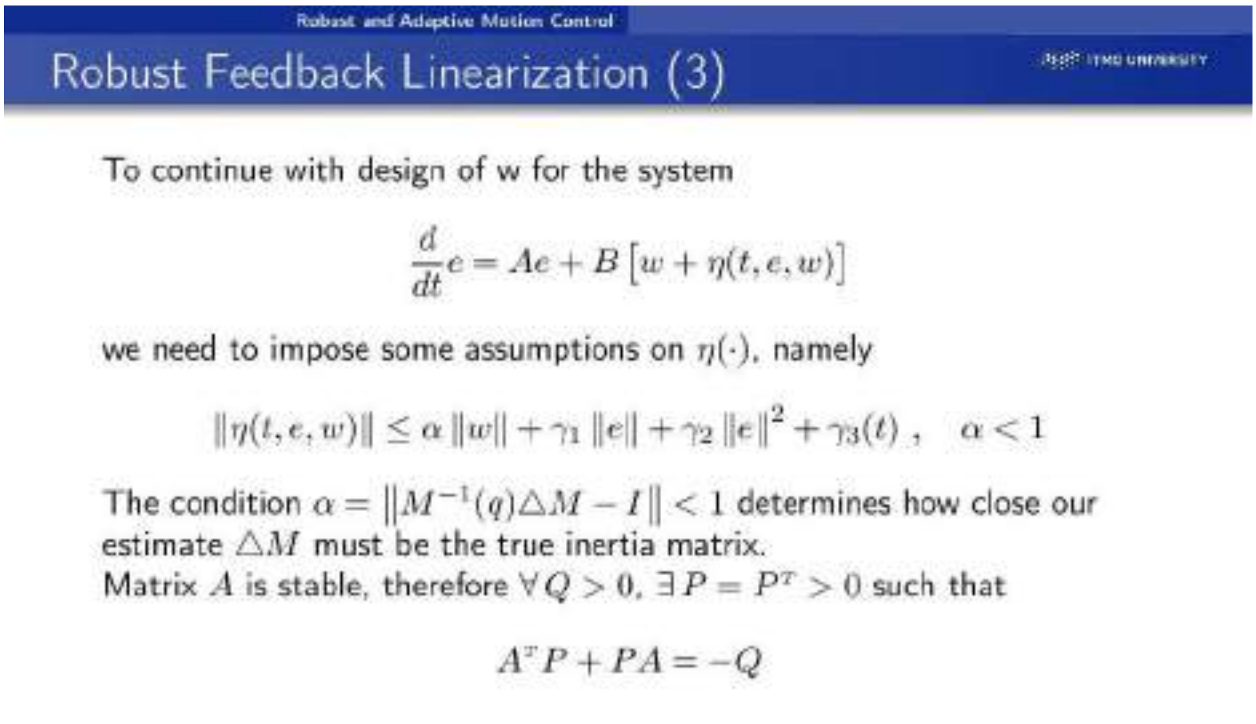
$$\begin{aligned}&M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = u\\&u = [M(q) + \triangle M]\,v + [C(q,\dot{q}) + \triangle C]\,\dot{q} + [G(q) + \triangle G]\\&v = \ddot{q}_d(t) - K_p(q - q_d(t)) - K_d(\dot{q} - \dot{q}_d(t)) + w\end{aligned}$$

It can be rewritten as

$$\ddot{q} = v + \eta(q,\dot{q},v)$$

$$(\ddot{q} - \ddot{q}_d(t)) + K_d\,(\dot{q} - \dot{q}_d(t)) + K_p\,(q - q_d(t)) = w + \eta(q,\dot{q},v)$$

$$\frac{d}{dt}e = \underbrace{\begin{bmatrix}0_{n\times n} & I_n\\ -K_p & -K_d\end{bmatrix}}_{A} e + \underbrace{\begin{bmatrix}0_{n\times n}\\ I_n\end{bmatrix}}_{B} [w + \eta(t,e,w)], \quad e = \begin{bmatrix}q - q_d(t)\\ \dot{q} - \dot{q}_d(t)\end{bmatrix}$$



Robust and Adaptive Motion Control

Robust Feedback Linearization (3)

ITMO UNIVERSITY

To continue with design of w for the system

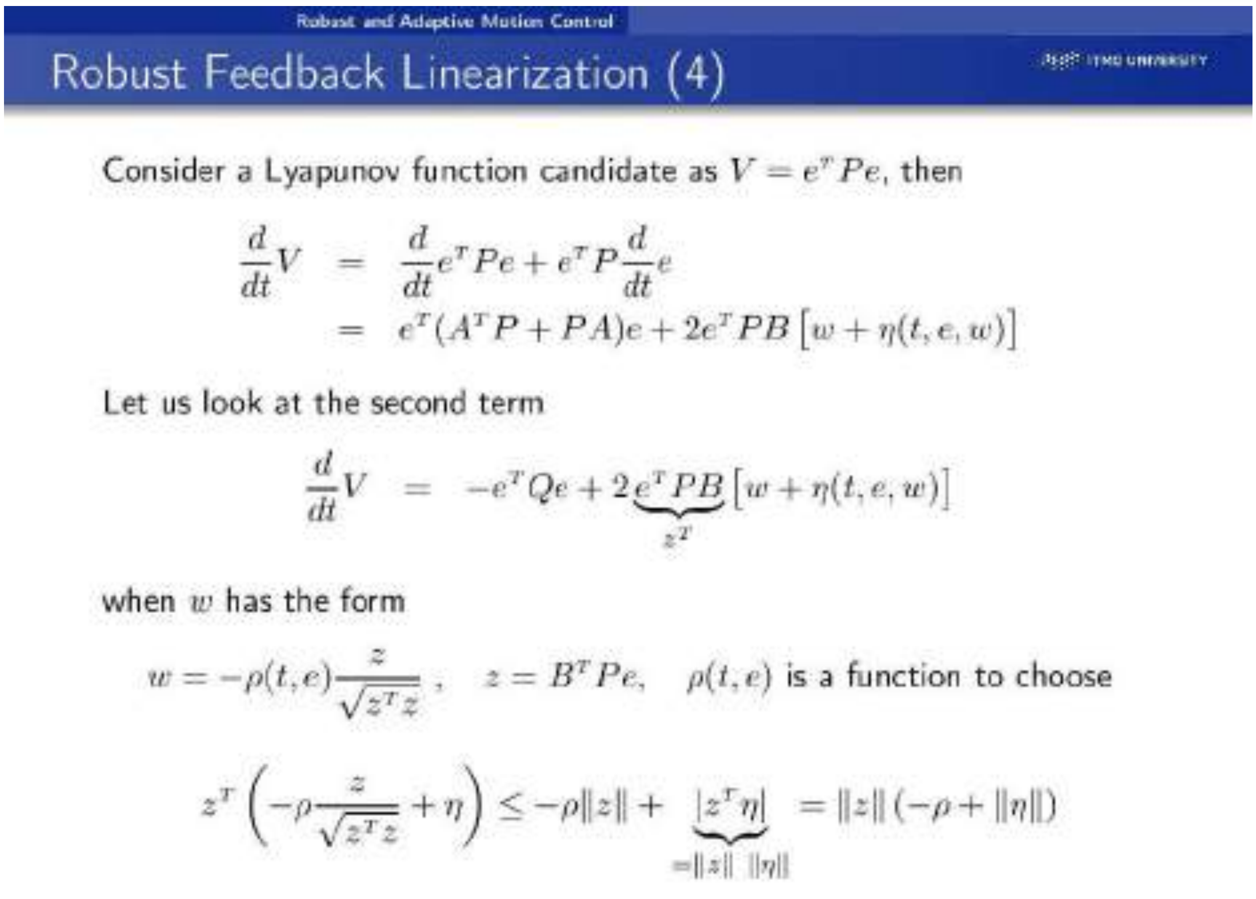
$$\frac{d}{dt}e = Ae + B\left[w + \eta(t,e,w)\right]$$

we need to impose some assumptions on $\eta(\cdot)$, namely

$$\|\eta(t,e,w)\| \le \alpha\,\|w\| + \gamma_1\,\|e\| + \gamma_2\,\|e\|^2 + \gamma_3(t)\ , \quad \alpha < 1$$

The condition $\alpha = \|M^{-1}(q)\triangle M - I\| < 1$ determines how close our estimate $\triangle M$ must be the true inertia matrix.
Matrix A is stable, therefore $\forall\, Q > 0,\ \exists\, P = P^T > 0$ such that

$$A^T P + PA = -Q$$



Robust and Adaptive Motion Control

Robust Feedback Linearization (4)

ITMO UNIVERSITY

Consider a Lyapunov function candidate as $V = e^T Pe$, then

$$\begin{aligned}\frac{d}{dt}V &= \frac{d}{dt}e^T Pe + e^T P\frac{d}{dt}e\\ &= e^T(A^T P + PA)e + 2e^T PB\left[w + \eta(t,e,w)\right]\end{aligned}$$

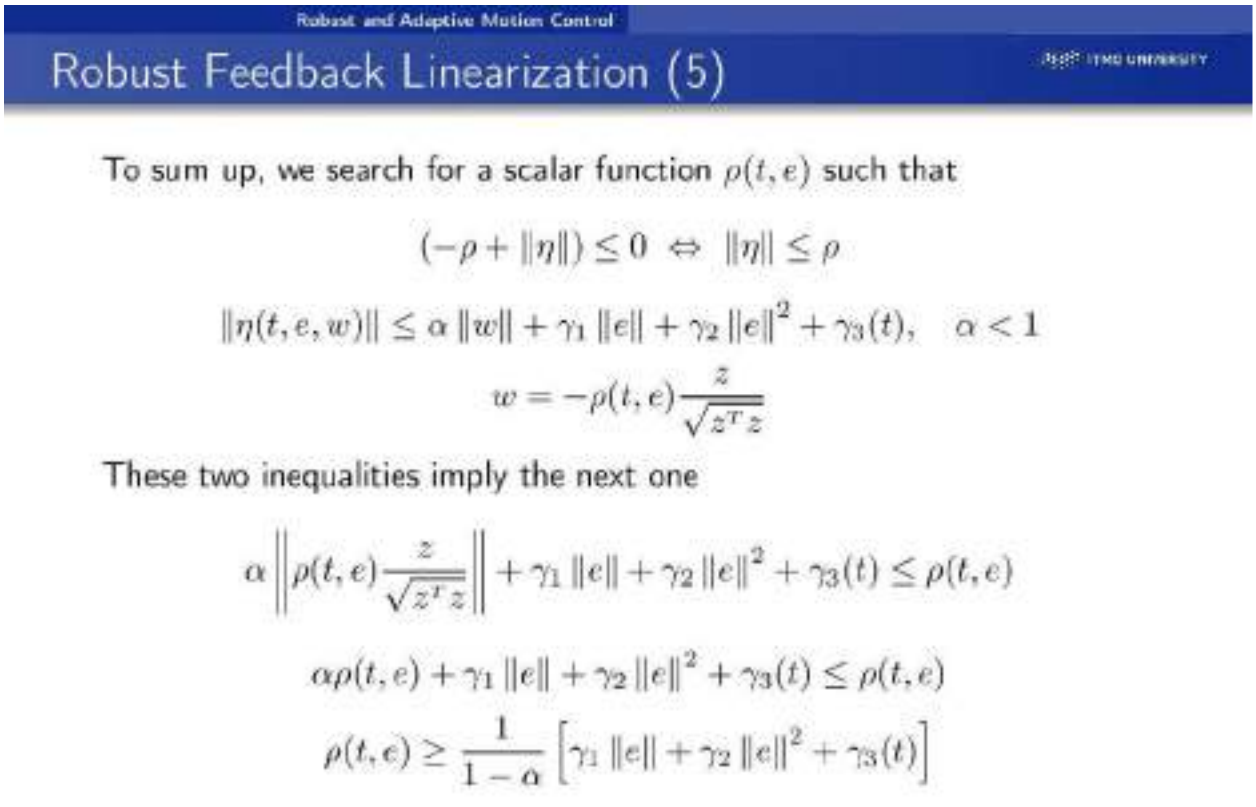
Let us look at the second term

$$\frac{d}{dt}V = -e^T Qe + 2\underbrace{e^T PB}_{z^T}\left[w + \eta(t,e,w)\right]$$

when w has the form

$$w = -\rho(t,e)\frac{z}{\sqrt{z^T z}}\ , \quad z = B^T Pe, \quad \rho(t,e) \text{ is a function to choose}$$

$$z^T\left(-\rho\frac{z}{\sqrt{z^T z}} + \eta\right) \le -\rho\|z\| + \underbrace{|z^T\eta|}_{=\|z\|\ \|\eta\|} = \|z\|\left(-\rho + \|\eta\|\right)$$



Robust and Adaptive Motion Control

Robust Feedback Linearization (5)

ITMO UNIVERSITY

To sum up, we search for a scalar function $\rho(t,e)$ such that

$$(-\rho + \|\eta\|) \leq 0 \;\Leftrightarrow\; \|\eta\| \leq \rho$$

$$\|\eta(t,e,w)\| \leq \alpha\|w\| + \gamma_1\|e\| + \gamma_2\|e\|^2 + \gamma_3(t), \quad \alpha < 1$$

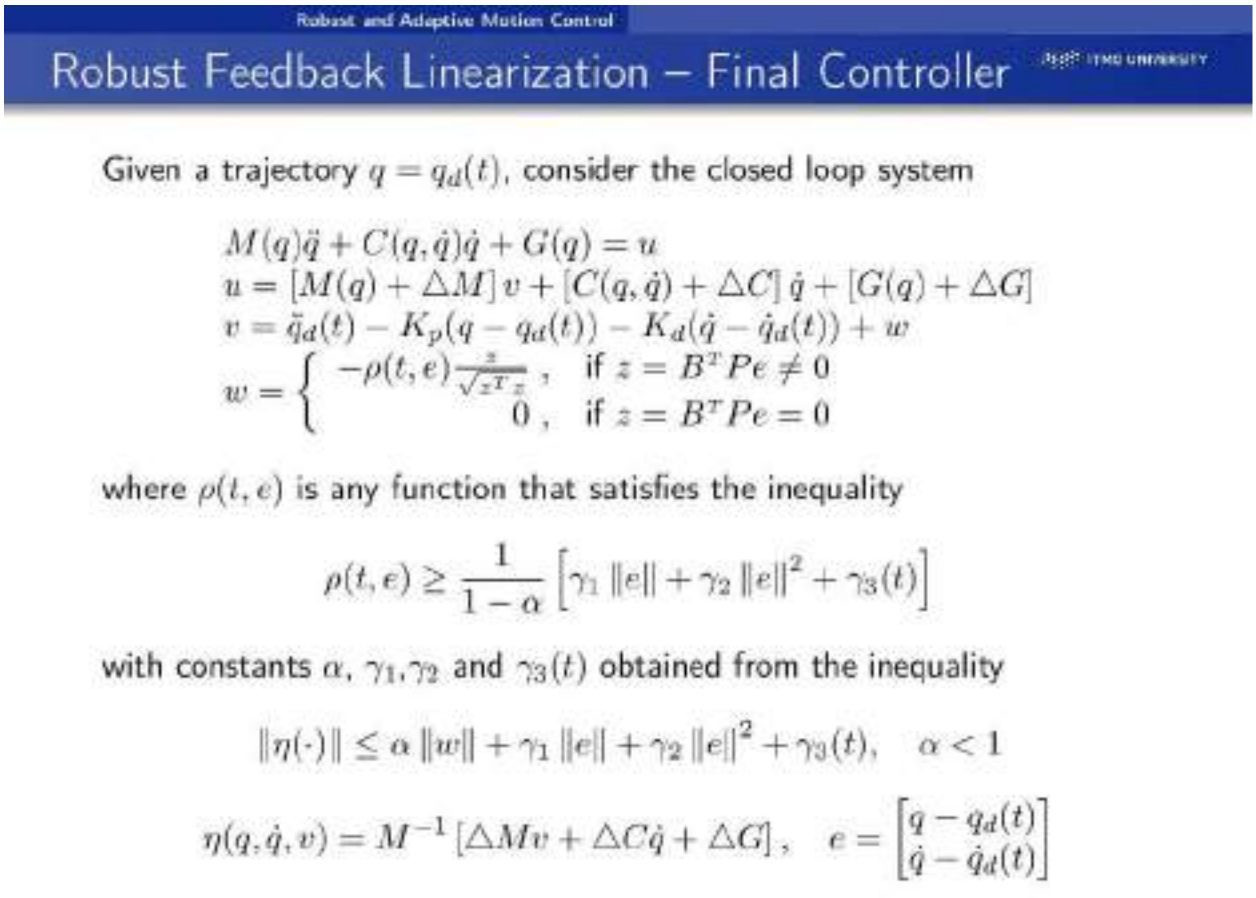
$$w = -\rho(t,e)\frac{z}{\sqrt{z^T z}}$$

These two inequalities imply the next one

$$\alpha\left\|\rho(t,e)\frac{z}{\sqrt{z^T z}}\right\| + \gamma_1\|e\| + \gamma_2\|e\|^2 + \gamma_3(t) \leq \rho(t,e)$$

$$\alpha\rho(t,e) + \gamma_1\|e\| + \gamma_2\|e\|^2 + \gamma_3(t) \leq \rho(t,e)$$

$$\rho(t,e) \geq \frac{1}{1-\alpha}\left[\gamma_1\|e\| + \gamma_2\|e\|^2 + \gamma_3(t)\right]$$



Robust and Adaptive Motion Control

Robust Feedback Linearization – Final Controller

ITMO UNIVERSITY

Given a trajectory $q = q_d(t)$, consider the closed loop system

$$\begin{aligned}
&M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = u \\
&u = [M(q) + \Delta M]\,v + [C(q,\dot{q}) + \Delta C]\,\dot{q} + [G(q) + \Delta G] \\
&v = \ddot{q}_d(t) - K_p(q - q_d(t)) - K_d(\dot{q} - \dot{q}_d(t)) + w \\
&w = \begin{cases} -\rho(t,e)\frac{z}{\sqrt{z^T z}}\,, & \text{if } z = B^T P e \neq 0 \\ 0\,, & \text{if } z = B^T P e = 0 \end{cases}
\end{aligned}$$

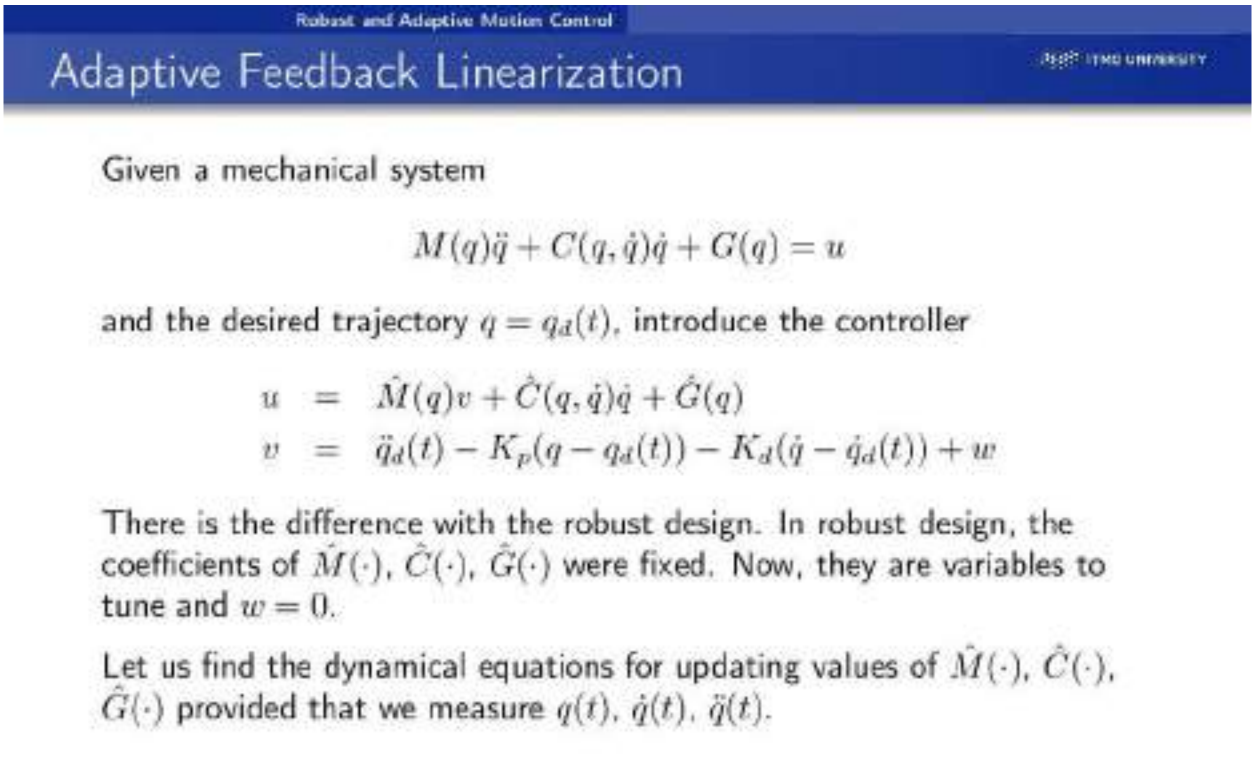
where $\rho(t,e)$ is any function that satisfies the inequality

$$\rho(t,e) \geq \frac{1}{1-\alpha}\left[\gamma_1\|e\| + \gamma_2\|e\|^2 + \gamma_3(t)\right]$$

with constants α, γ_1,γ_2 and $\gamma_3(t)$ obtained from the inequality

$$\|\eta(\cdot)\| \leq \alpha\|w\| + \gamma_1\|e\| + \gamma_2\|e\|^2 + \gamma_3(t), \quad \alpha < 1$$

$$\eta(q,\dot{q},v) = M^{-1}[\Delta M v + \Delta C\dot{q} + \Delta G]\,, \quad e = \begin{bmatrix} q - q_d(t) \\ \dot{q} - \dot{q}_d(t) \end{bmatrix}$$



Robust and Adaptive Motion Control

Adaptive Feedback Linearization

ITMO UNIVERSITY

Given a mechanical system

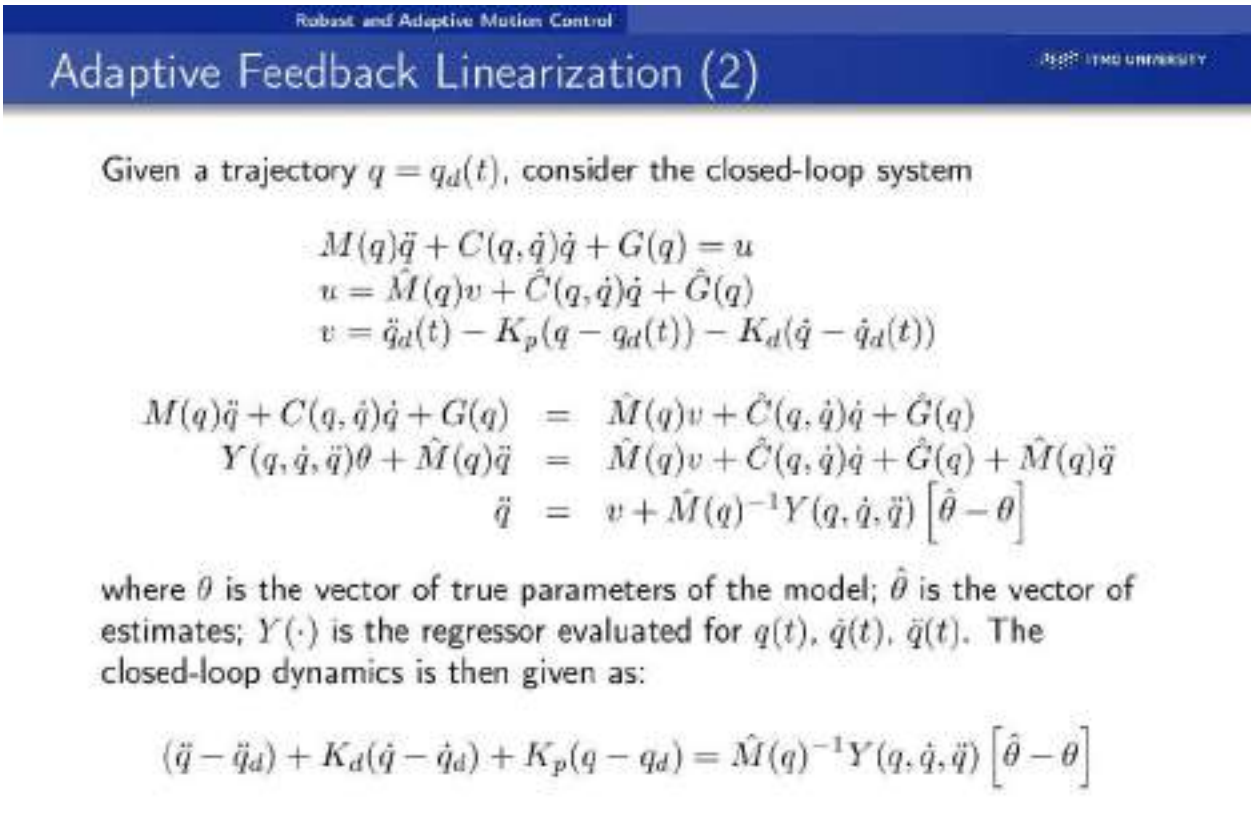
$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = u$$

and the desired trajectory $q = q_d(t)$, introduce the controller

$$\begin{aligned} u &= \hat{M}(q)v + \hat{C}(q,\dot{q})\dot{q} + \hat{G}(q) \\ v &= \ddot{q}_d(t) - K_p(q - q_d(t)) - K_d(\dot{q} - \dot{q}_d(t)) + w \end{aligned}$$

There is the difference with the robust design. In robust design, the coefficients of $\hat{M}(\cdot)$, $\hat{C}(\cdot)$, $\hat{G}(\cdot)$ were fixed. Now, they are variables to tune and $w = 0$.

Let us find the dynamical equations for updating values of $\hat{M}(\cdot)$, $\hat{C}(\cdot)$, $\hat{G}(\cdot)$ provided that we measure $q(t)$, $\dot{q}(t)$, $\ddot{q}(t)$.



Robust and Adaptive Motion Control

Adaptive Feedback Linearization (2)

ITMO UNIVERSITY

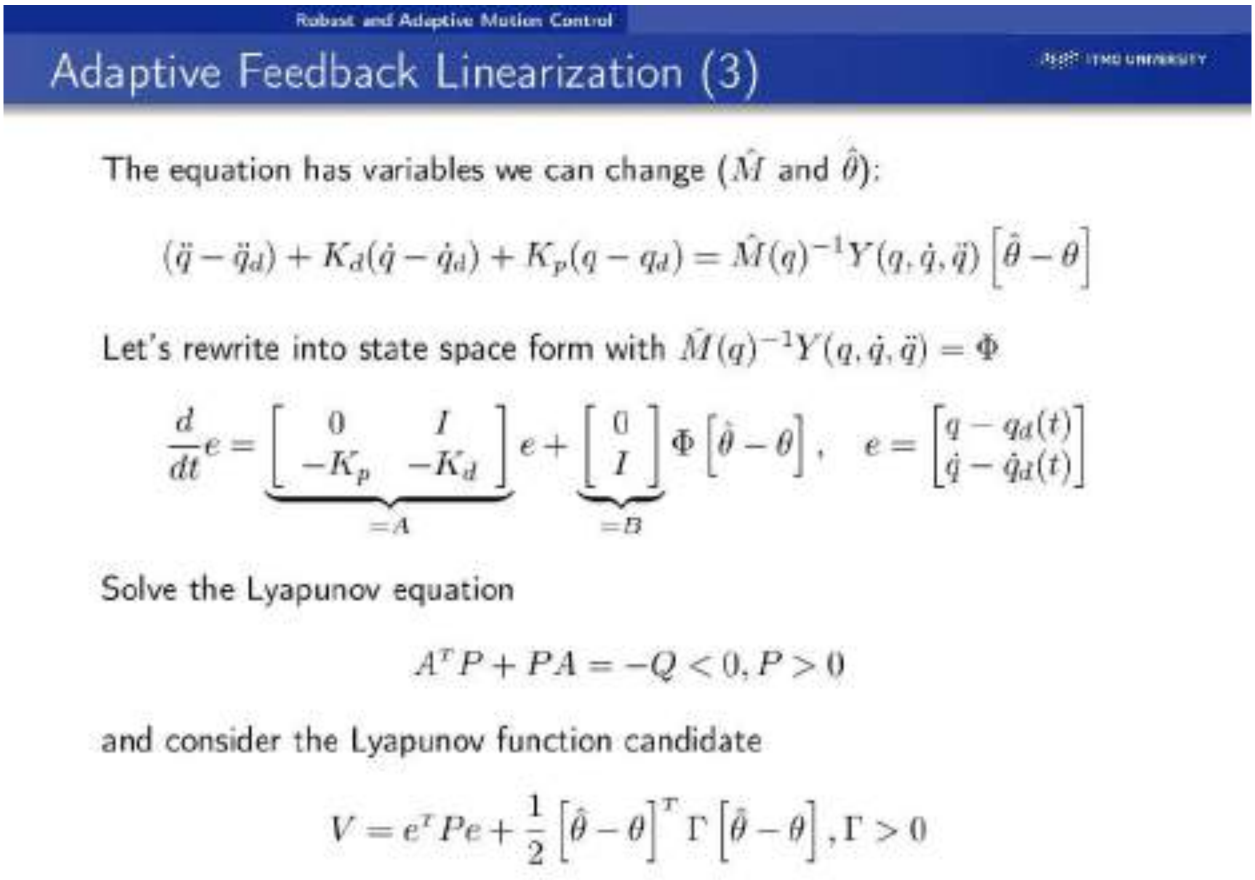
Given a trajectory $q = q_d(t)$, consider the closed-loop system

$$\begin{aligned} & M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = u \\ & u = \hat{M}(q)v + \hat{C}(q,\dot{q})\dot{q} + \hat{G}(q) \\ & v = \ddot{q}_d(t) - K_p(q - q_d(t)) - K_d(\dot{q} - \dot{q}_d(t)) \end{aligned}$$

$$\begin{aligned} M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) &= \hat{M}(q)v + \hat{C}(q,\dot{q})\dot{q} + \hat{G}(q) \\ Y(q,\dot{q},\ddot{q})\theta + \hat{M}(q)\ddot{q} &= \hat{M}(q)v + \hat{C}(q,\dot{q})\dot{q} + \hat{G}(q) + \hat{M}(q)\ddot{q} \\ \ddot{q} &= v + \hat{M}(q)^{-1}Y(q,\dot{q},\ddot{q})\left[\hat{\theta} - \theta\right] \end{aligned}$$

where θ is the vector of true parameters of the model; $\hat{\theta}$ is the vector of estimates; $Y(\cdot)$ is the regressor evaluated for $q(t)$, $\dot{q}(t)$, $\ddot{q}(t)$. The closed-loop dynamics is then given as:

$$(\ddot{q} - \ddot{q}_d) + K_d(\dot{q} - \dot{q}_d) + K_p(q - q_d) = \hat{M}(q)^{-1}Y(q,\dot{q},\ddot{q})\left[\hat{\theta} - \theta\right]$$



Robust and Adaptive Motion Control

Adaptive Feedback Linearization (3)

ITMO UNIVERSITY

The equation has variables we can change ($\hat{M}$ and $\hat{\theta}$):

$$(\ddot{q}-\ddot{q}_d)+K_d(\dot{q}-\dot{q}_d)+K_p(q-q_d)=\hat{M}(q)^{-1}Y(q,\dot{q},\ddot{q})\left[\hat{\theta}-\theta\right]$$

Let's rewrite into state space form with $\hat{M}(q)^{-1}Y(q,\dot{q},\ddot{q})=\Phi$

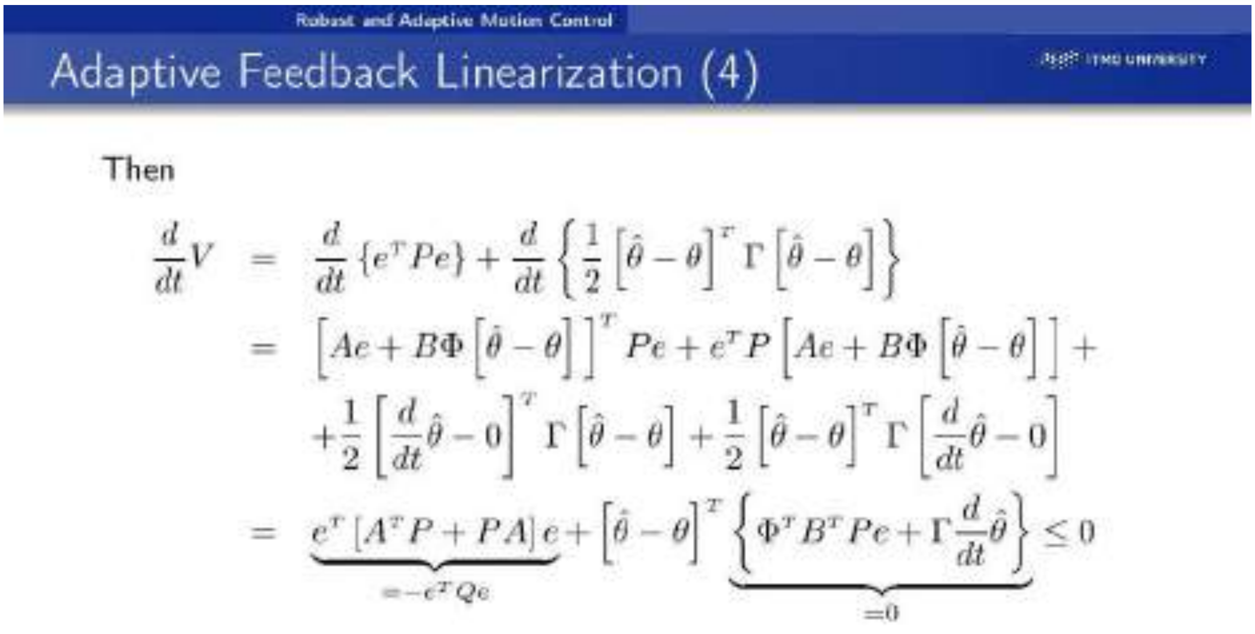
$$\frac{d}{dt}e=\underbrace{\begin{bmatrix}0 & I\\ -K_p & -K_d\end{bmatrix}}_{=A}e+\underbrace{\begin{bmatrix}0\\ I\end{bmatrix}}_{=B}\Phi\left[\hat{\theta}-\theta\right],\quad e=\begin{bmatrix}q-q_d(t)\\ \dot{q}-\dot{q}_d(t)\end{bmatrix}$$

Solve the Lyapunov equation

$$A^TP+PA=-Q<0,P>0$$

and consider the Lyapunov function candidate

$$V=e^TPe+\frac{1}{2}\left[\hat{\theta}-\theta\right]^T\Gamma\left[\hat{\theta}-\theta\right],\Gamma>0$$



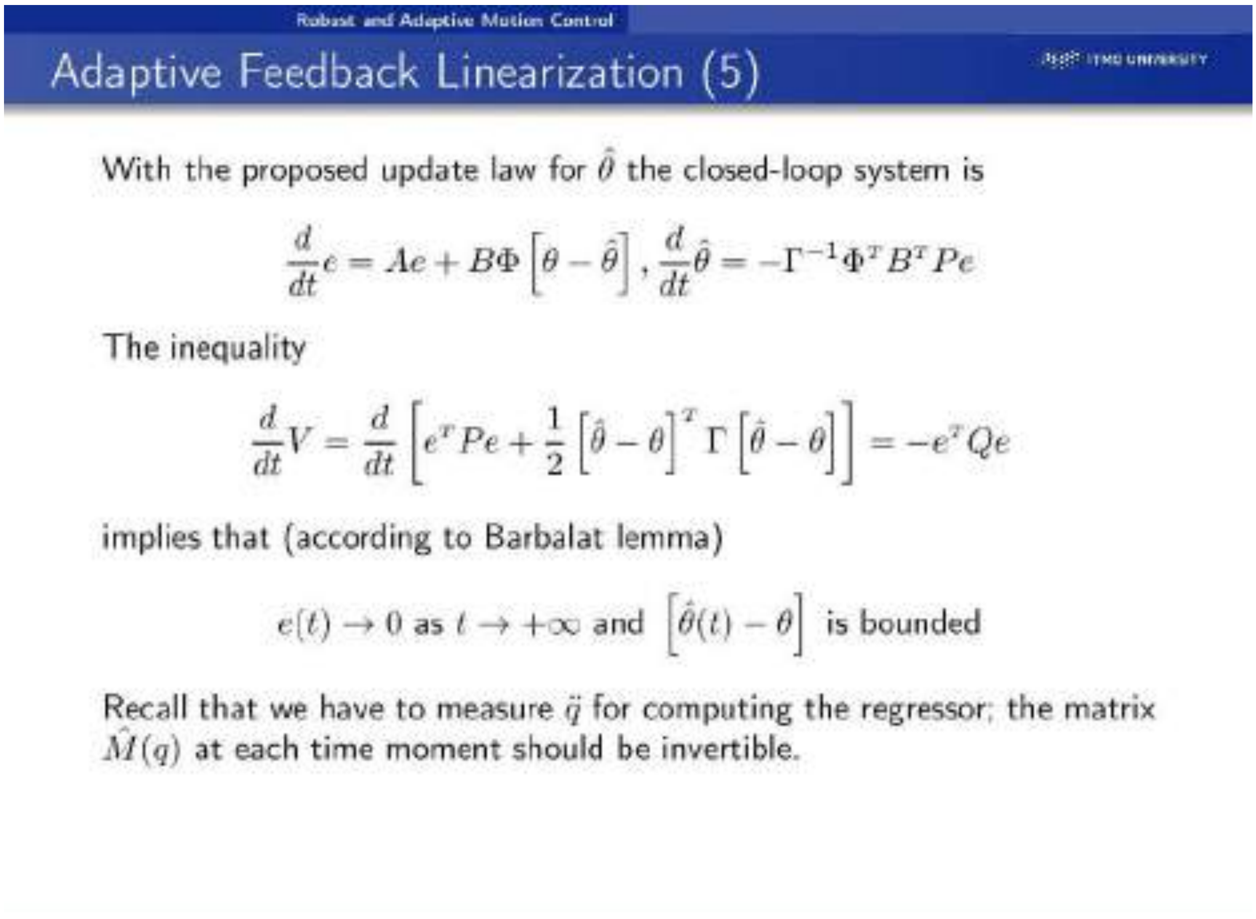
Robust and Adaptive Motion Control

Adaptive Feedback Linearization (4)

ITMO UNIVERSITY

Then

$$\begin{aligned}\frac{d}{dt}V &= \frac{d}{dt}\{e^TPe\}+\frac{d}{dt}\left\{\frac{1}{2}\left[\hat{\theta}-\theta\right]^T\Gamma\left[\hat{\theta}-\theta\right]\right\}\\ &= \left[Ae+B\Phi\left[\hat{\theta}-\theta\right]\right]^TPe+e^TP\left[Ae+B\Phi\left[\hat{\theta}-\theta\right]\right]+\\ &\quad +\frac{1}{2}\left[\frac{d}{dt}\hat{\theta}-0\right]^T\Gamma\left[\hat{\theta}-\theta\right]+\frac{1}{2}\left[\hat{\theta}-\theta\right]^T\Gamma\left[\frac{d}{dt}\hat{\theta}-0\right]\\ &= \underbrace{e^T\left[A^TP+PA\right]e}_{=-e^TQe}+\left[\hat{\theta}-\theta\right]^T\underbrace{\left\{\Phi^TB^TPe+\Gamma\frac{d}{dt}\hat{\theta}\right\}}_{=0}\leq 0\end{aligned}$$



Robust and Adaptive Motion Control

Adaptive Feedback Linearization (5)

ITMO UNIVERSITY

With the proposed update law for $\hat{\theta}$ the closed-loop system is

$$\frac{d}{dt}e = Ae + B\Phi\left[\theta - \hat{\theta}\right], \frac{d}{dt}\hat{\theta} = -\Gamma^{-1}\Phi^T B^T P e$$

The inequality

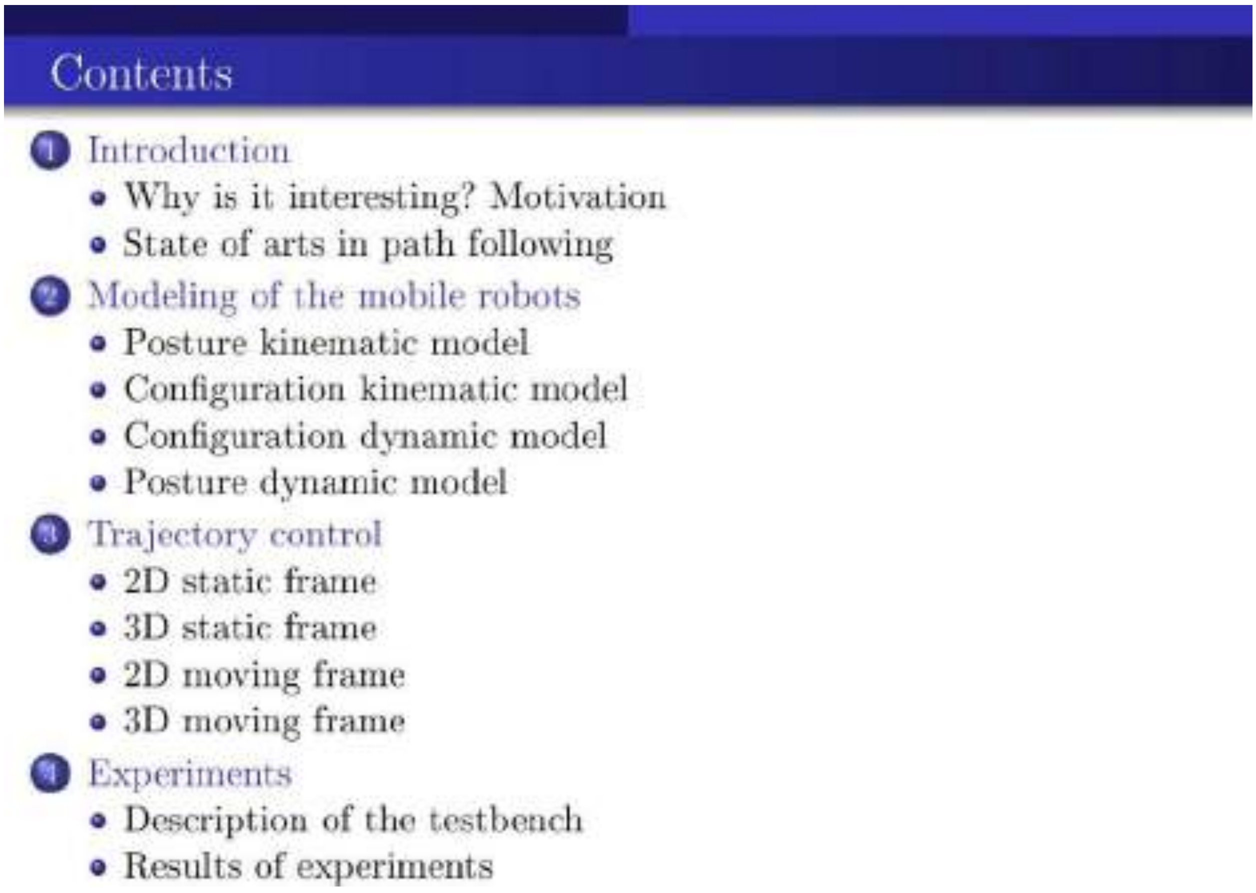
$$\frac{d}{dt}V = \frac{d}{dt}\left[e^T P e + \frac{1}{2}\left[\hat{\theta} - \theta\right]^T \Gamma \left[\hat{\theta} - \theta\right]\right] = -e^T Q e$$

implies that (according to Barbalat lemma)

$$e(t) \to 0 \text{ as } t \to +\infty \text{ and } \left[\hat{\theta}(t) - \theta\right] \text{ is bounded}$$

Recall that we have to measure $\ddot{q}$ for computing the regressor; the matrix $\hat{M}(q)$ at each time moment should be invertible.

Trajectory control of mobile robots



Contents

Introduction Why is it interesting? Motivation

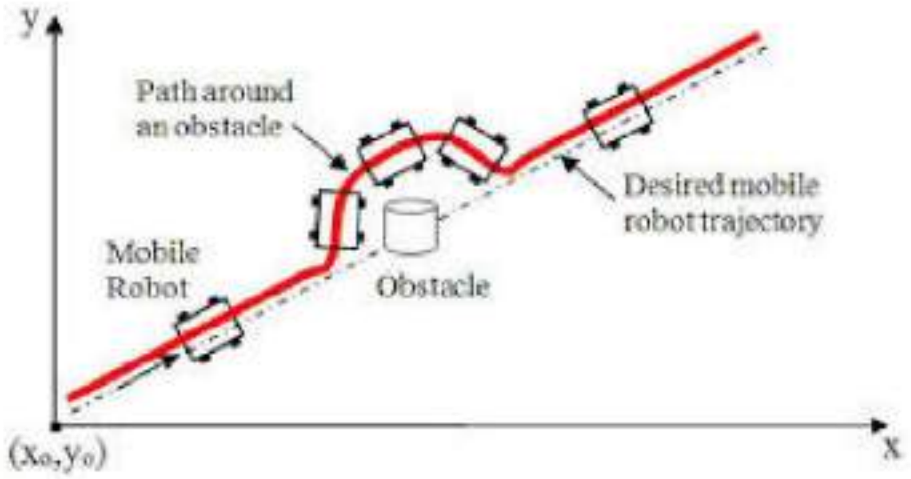
Examples of problems

Autonomous vehicles control (mobile robots, UAV etc.)

ITMO UNIVERSITY

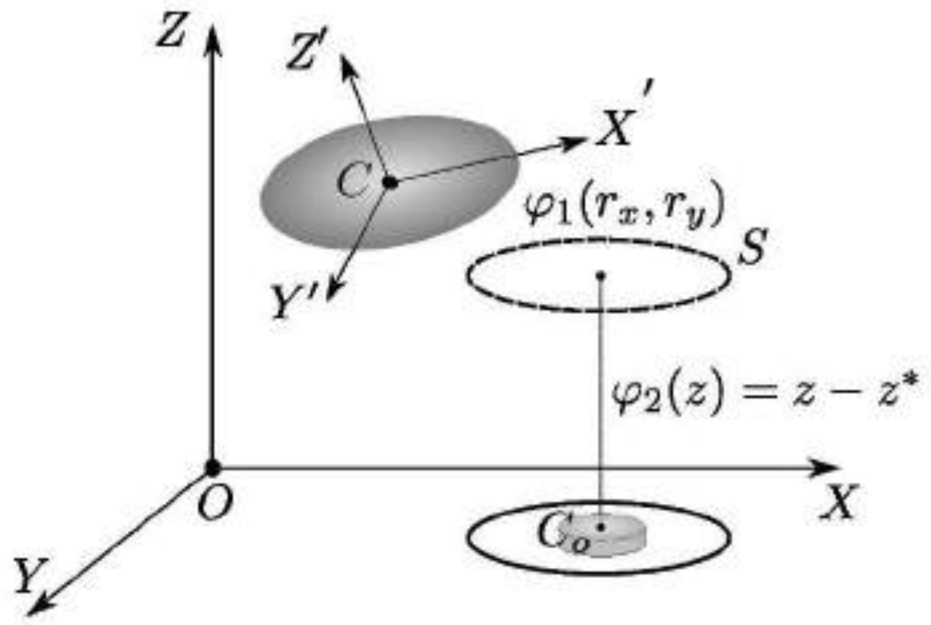
Examples of problems

Collision avoidance:



Examples of problems

Following the moving target:



Introduction | Why is it interesting? Motivation

Examples of problems

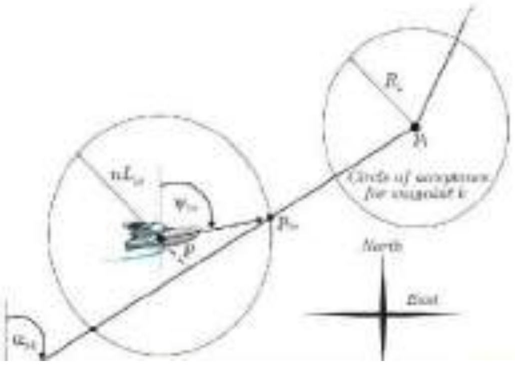
Control of underwater biomimetic robots (Robotic Fish)

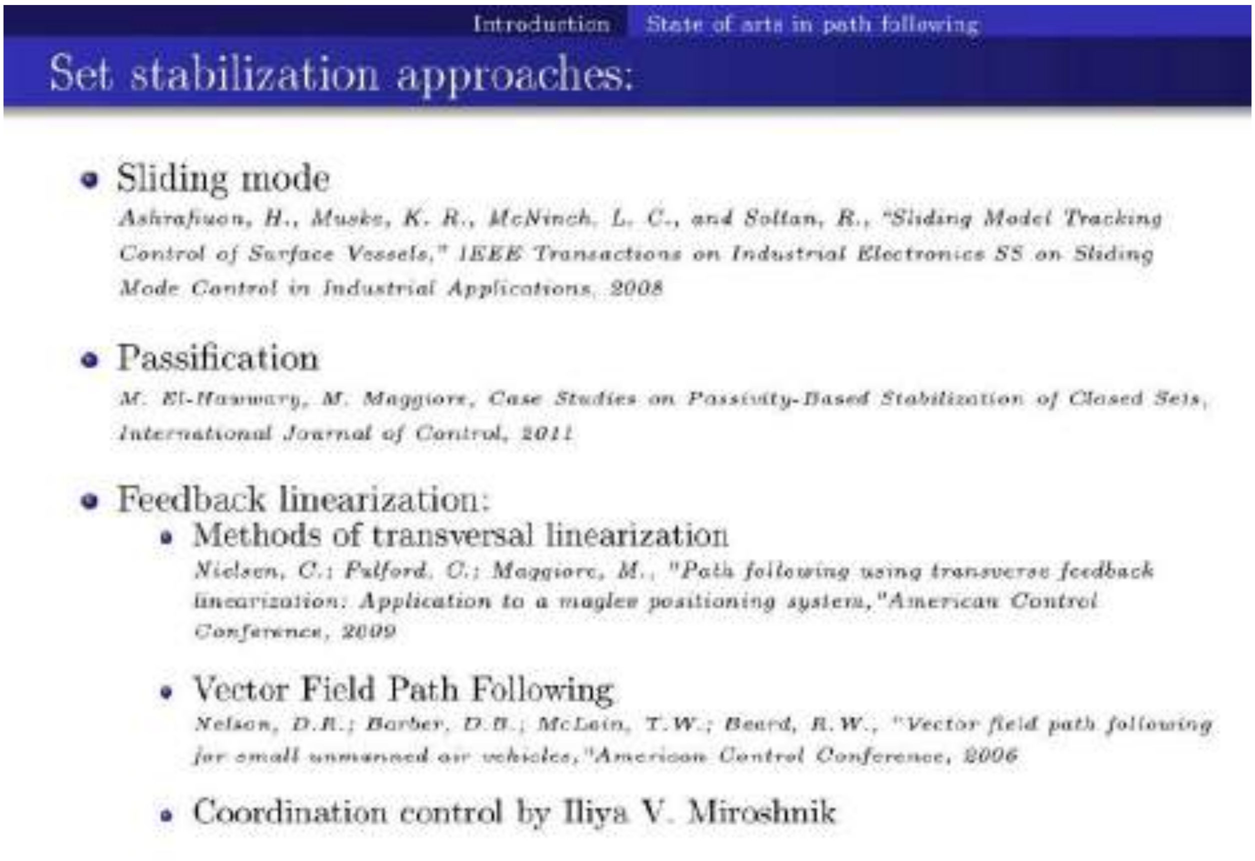
Problem: robot can not stop after reaching a goal.
Possible solution: continue motion along closed curve around the the goal

Introduction | State of arts in path following

Possible solutions:

- 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

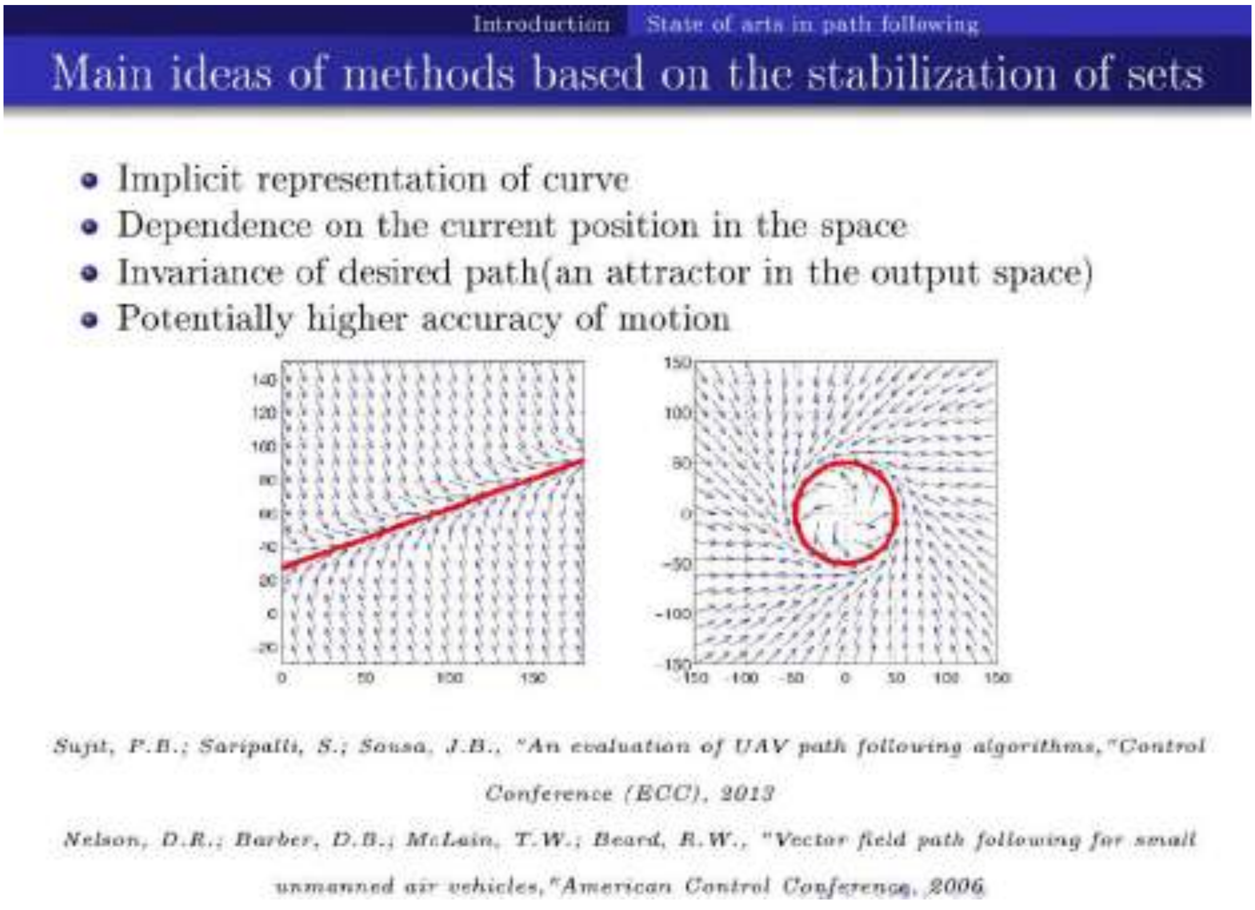




Introduction | State of arts in path following

Set stabilization approaches:

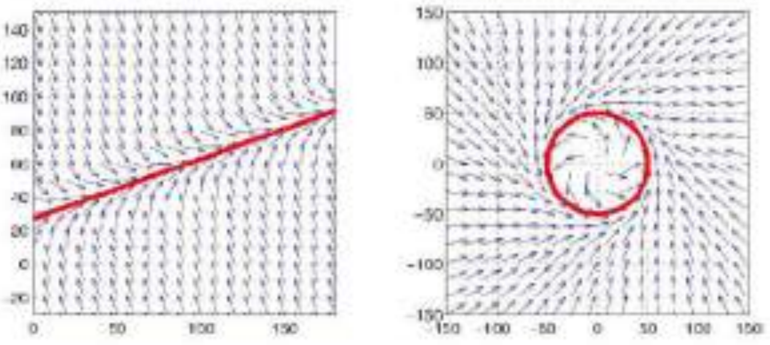
- Sliding mode
  Ashrafiuon, H., Muske, K. R., McNinch, L. C., and Soltan, R., "Sliding Model Tracking Control of Surface Vessels," IEEE Transactions on Industrial Electronics SS on Sliding Mode Control in Industrial Applications, 2008
- Passification
  M. El-Hawwary, M. Maggiore, Case Studies on Passivity-Based 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, 2006
  - Coordination control by Iliya V. Miroshnik



Introduction | State of arts in path following

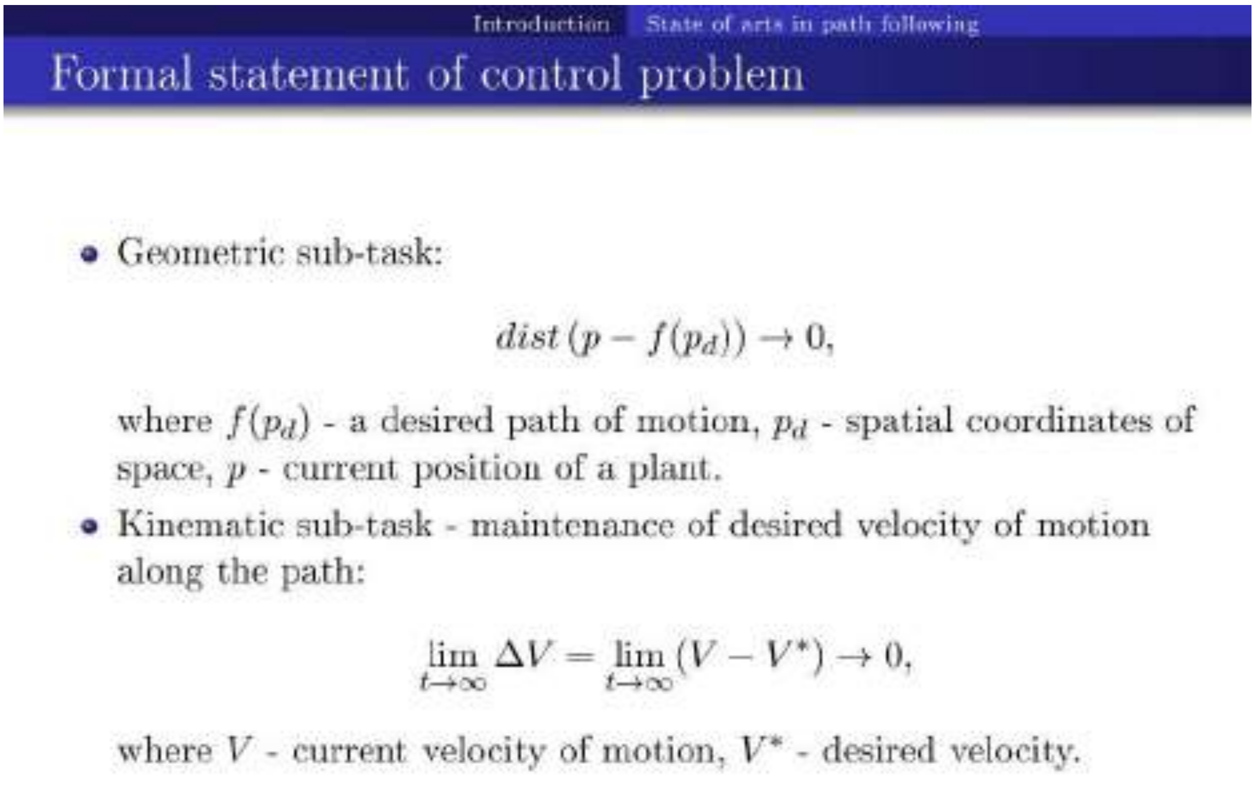
Main ideas of methods based on the stabilization of sets

- 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 vehicles,"American Control Conference, 2006



Formal statement of control problem

- Geometric sub-task:

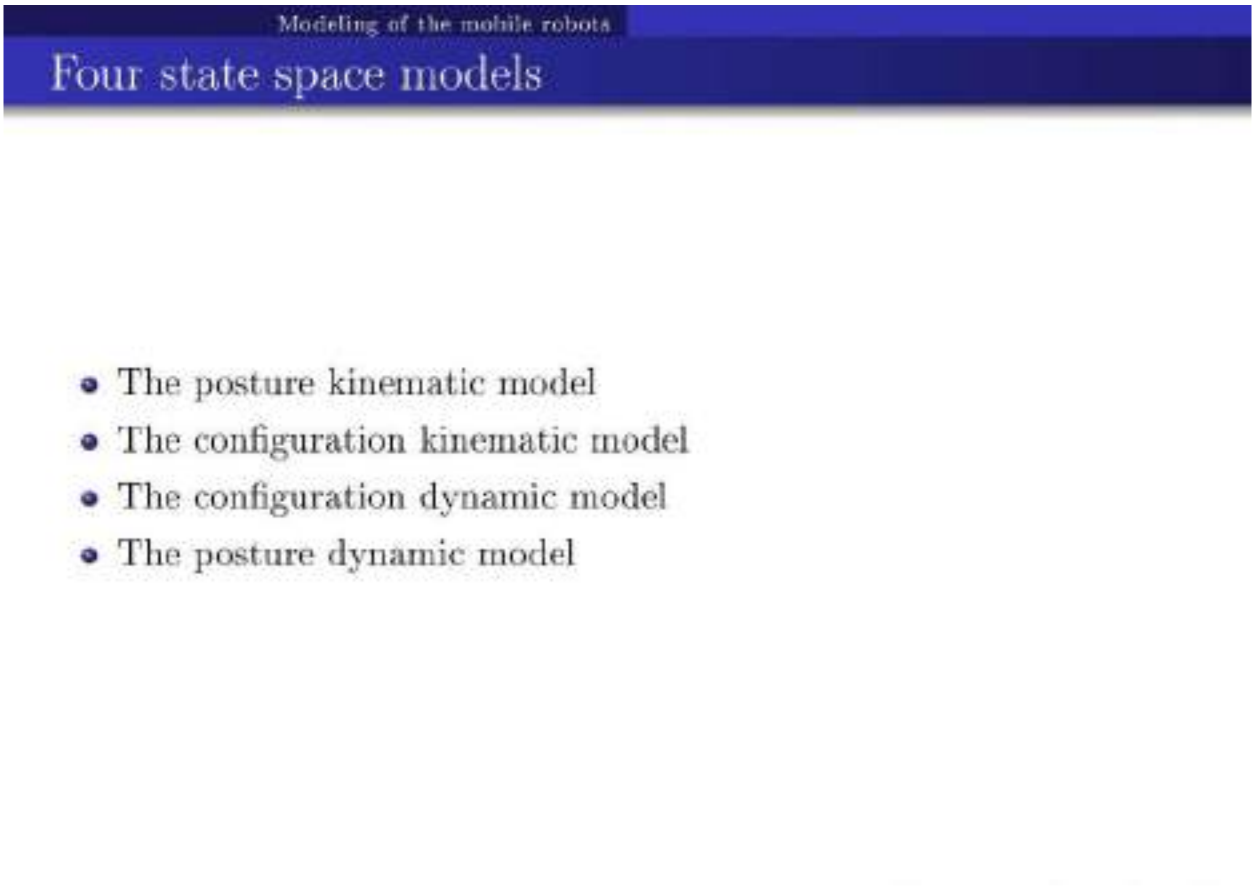
$$dist\,(p - f(p_d)) \to 0,$$

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:

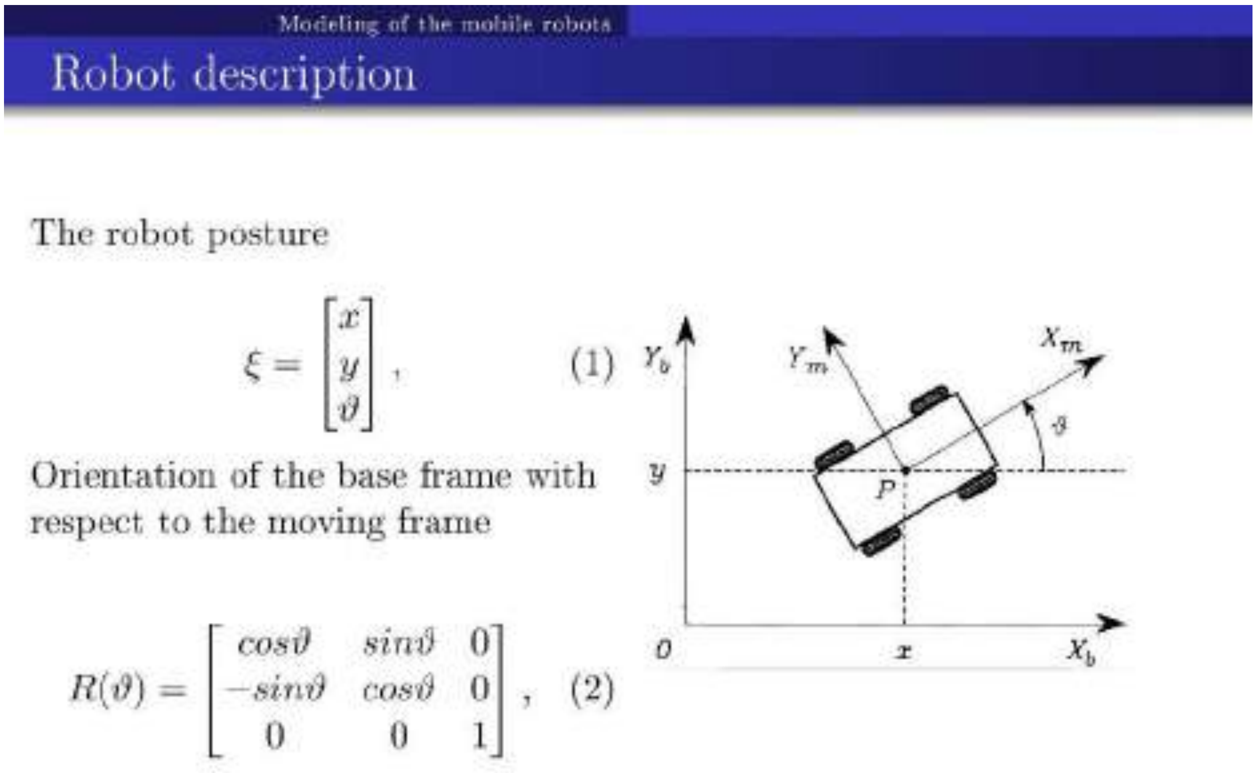
$$\lim_{t\to\infty} \Delta V = \lim_{t\to\infty} (V - V^*) \to 0,$$

where V - current velocity of motion, V^* - desired velocity.



Four state space models

- The posture kinematic model
- The configuration kinematic model
- The configuration dynamic model
- The posture dynamic model

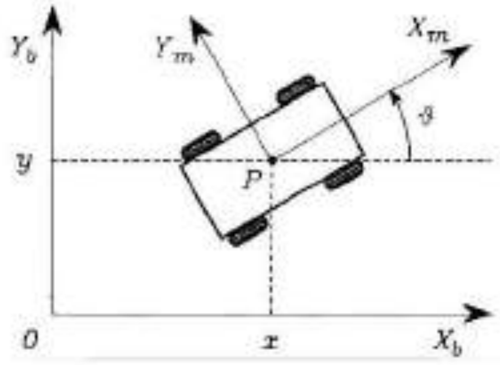


Modeling of the mobile robots

Robot description

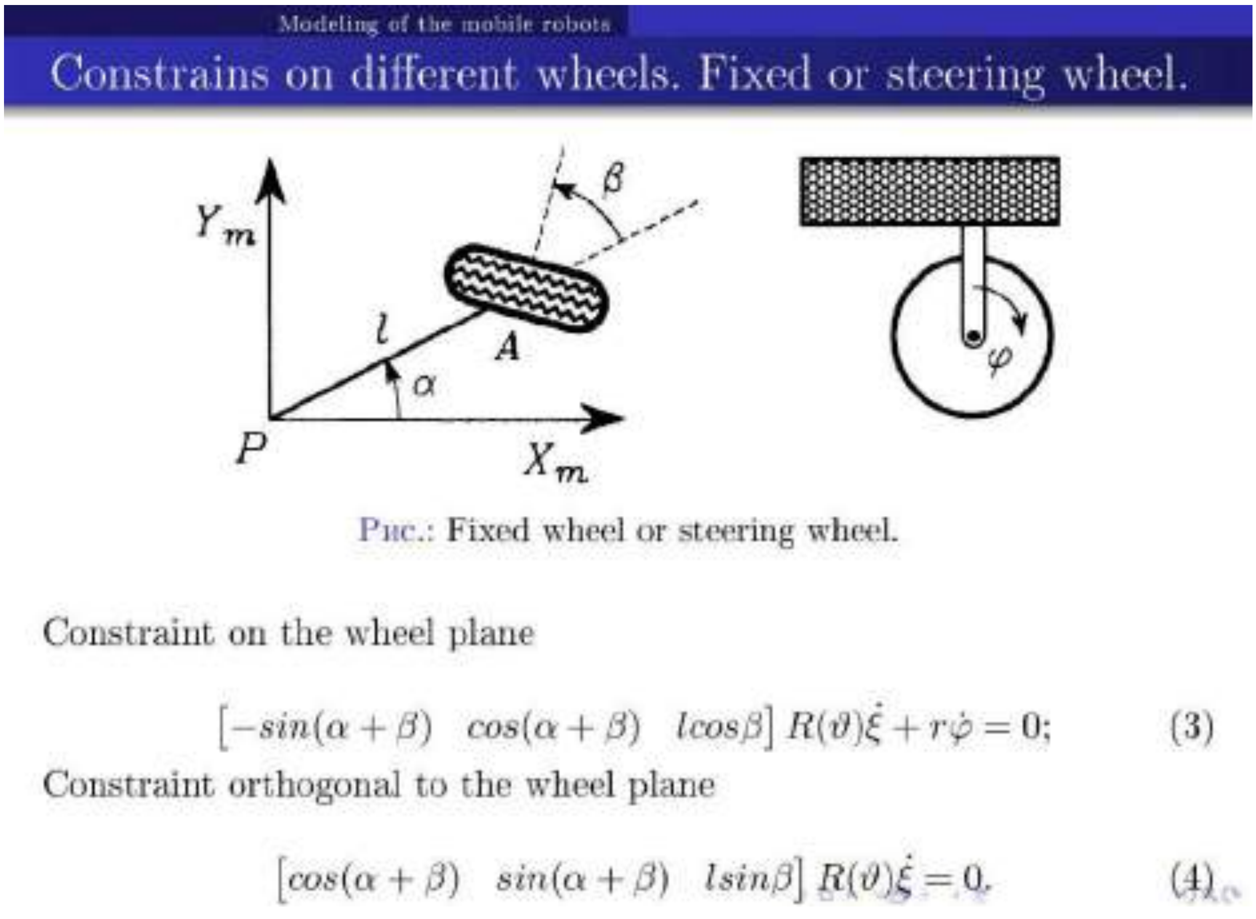
The robot posture

$$\xi = \begin{bmatrix} x \\ y \\ \vartheta \end{bmatrix}, \qquad (1)$$



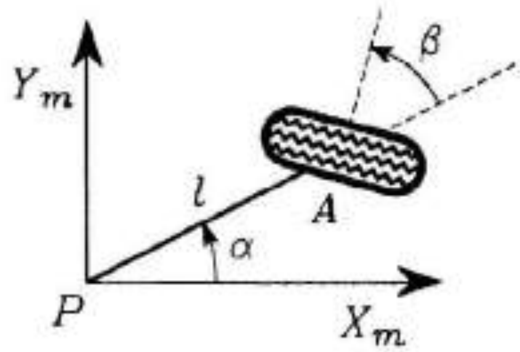
Orientation of the base frame with respect to the moving frame

$$R(\vartheta) = \begin{bmatrix} cos\vartheta & sin\vartheta & 0 \\ -sin\vartheta & cos\vartheta & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad (2)$$



Modeling of the mobile robots

Constrains on different wheels. Fixed or steering wheel.



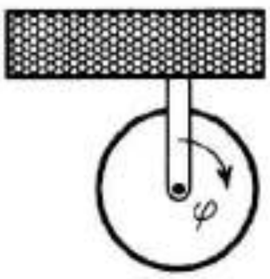


Рис.: Fixed wheel or steering wheel.

Constraint on the wheel plane

$$\begin{bmatrix} -sin(\alpha+\beta) & cos(\alpha+\beta) & lcos\beta \end{bmatrix} R(\vartheta)\dot{\xi} + r\dot{\varphi} = 0; \qquad (3)$$

Constraint orthogonal to the wheel plane

$$\begin{bmatrix} cos(\alpha+\beta) & sin(\alpha+\beta) & lsin\beta \end{bmatrix} R(\vartheta)\dot{\xi} = 0. \qquad (4)$$

Modeling of the mobile robots

Constrains on different wheels. Castor wheel.

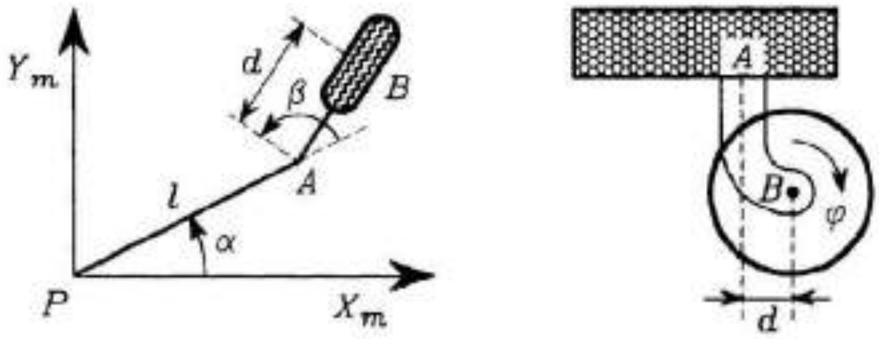


Рис.: Castor wheel.

Constraint on the wheel plane

$$\begin{bmatrix} -sin(\alpha+\beta) & cos(\alpha+\beta) & lcos\beta \end{bmatrix} R(\vartheta)\dot{\xi} + r\dot{\varphi} = 0; \tag{5}$$

Constraint orthogonal to the wheel plane

$$\begin{bmatrix} cos(\alpha+\beta) & sin(\alpha+\beta) & d + lsin\beta \end{bmatrix} R(\vartheta)\dot{\xi} + d\dot{\beta} = 0. \tag{6}$$

Modeling of the mobile robots

Constrains on different wheels. Swedish wheel.

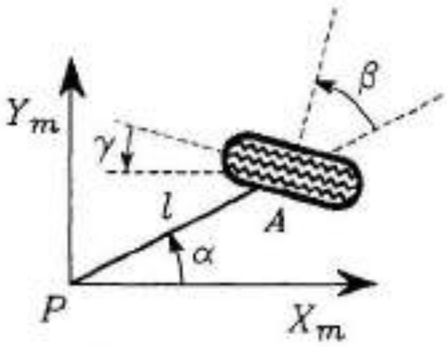
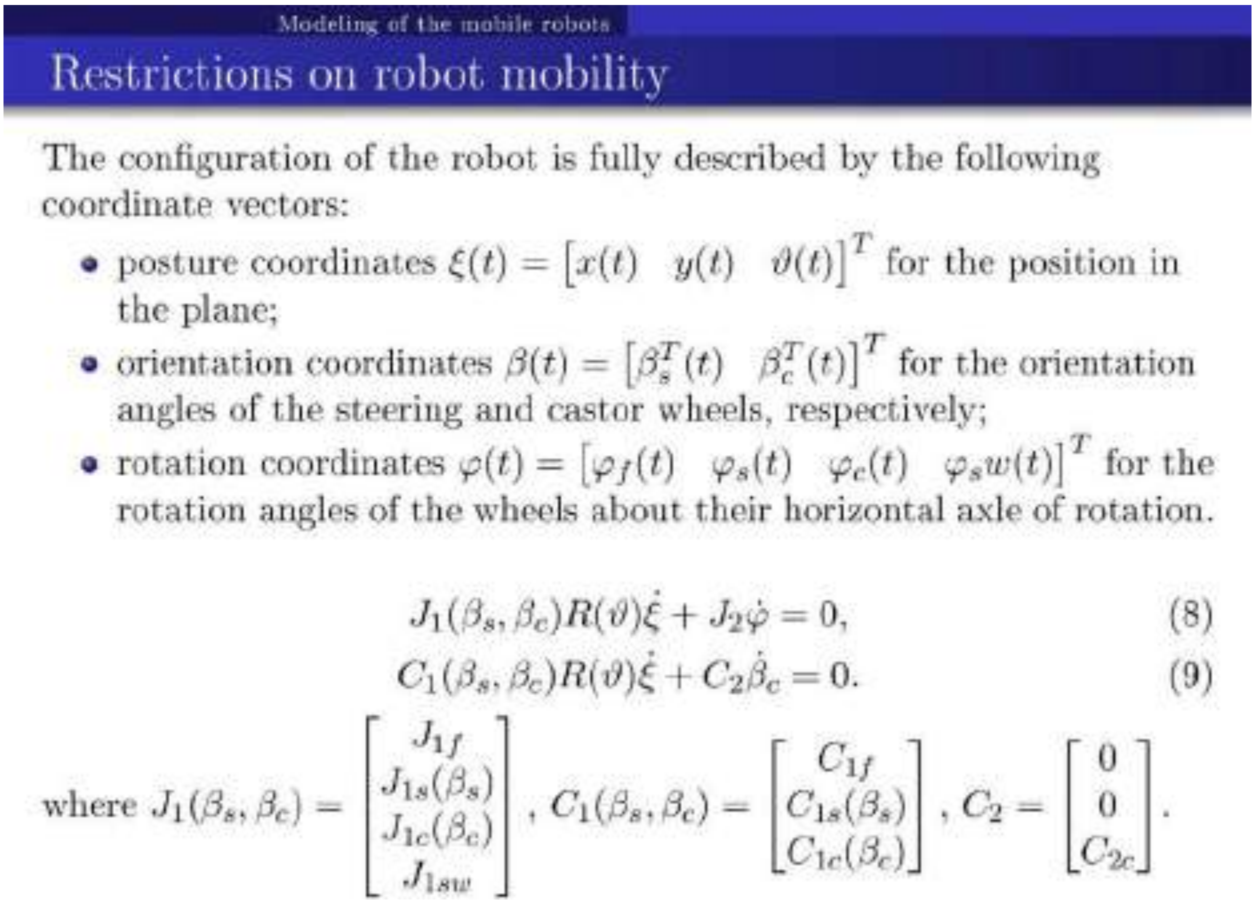


Рис.: Swedish wheel.

The motion constraint

$$\begin{bmatrix} -sin(\alpha+\beta+\gamma) & cos(\alpha+\beta+\gamma) & lcos(\beta+\gamma) \end{bmatrix} R(\vartheta)\dot{\xi} + rcos\gamma\dot{\varphi} = 0. \tag{7}$$



Modeling of the mobile robots

Restrictions on robot mobility

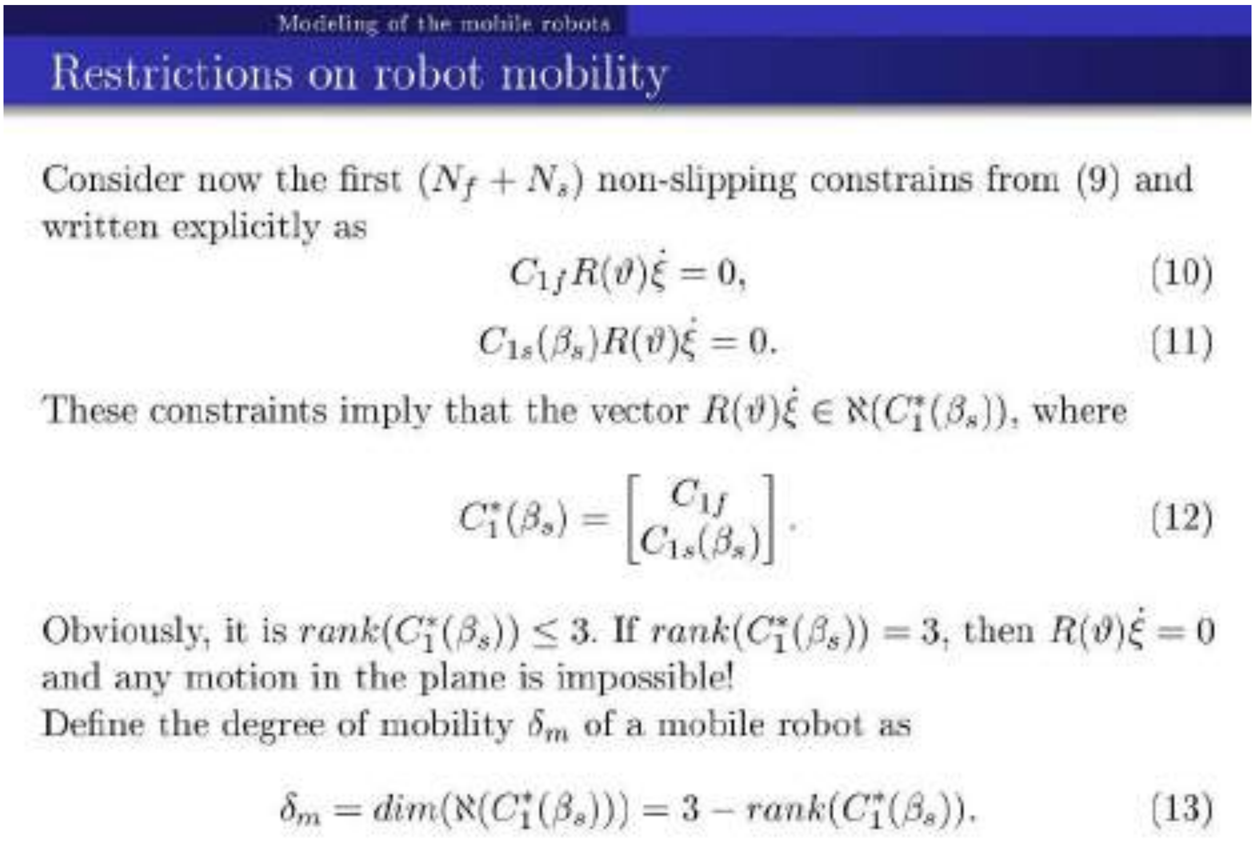
The configuration of the robot is fully described by the following coordinate vectors:

- posture coordinates $\xi(t) = \begin{bmatrix} x(t) & y(t) & \vartheta(t) \end{bmatrix}^T$ for the position in the plane;
- orientation coordinates $\beta(t) = \begin{bmatrix} \beta_s^T(t) & \beta_c^T(t) \end{bmatrix}^T$ for the orientation angles of the steering and castor wheels, respectively;
- rotation coordinates $\varphi(t) = \begin{bmatrix} \varphi_f(t) & \varphi_s(t) & \varphi_c(t) & \varphi_s w(t) \end{bmatrix}^T$ for the rotation angles of the wheels about their horizontal axle of rotation.

$$J_1(\beta_s, \beta_c)R(\vartheta)\dot{\xi} + J_2\dot{\varphi} = 0, \tag{8}$$

$$C_1(\beta_s, \beta_c)R(\vartheta)\dot{\xi} + C_2\dot{\beta}_c = 0. \tag{9}$$

where $J_1(\beta_s, \beta_c) = \begin{bmatrix} J_{1f} \\ J_{1s}(\beta_s) \\ J_{1c}(\beta_c) \\ J_{1sw} \end{bmatrix}$, $C_1(\beta_s, \beta_c) = \begin{bmatrix} C_{1f} \\ C_{1s}(\beta_s) \\ C_{1c}(\beta_c) \end{bmatrix}$, $C_2 = \begin{bmatrix} 0 \\ 0 \\ C_{2c} \end{bmatrix}$.



Modeling of the mobile robots

Restrictions on robot mobility

Consider now the first $(N_f + N_s)$ non-slipping constrains from (9) and written explicitly as

$$C_{1f}R(\vartheta)\dot{\xi} = 0, \tag{10}$$

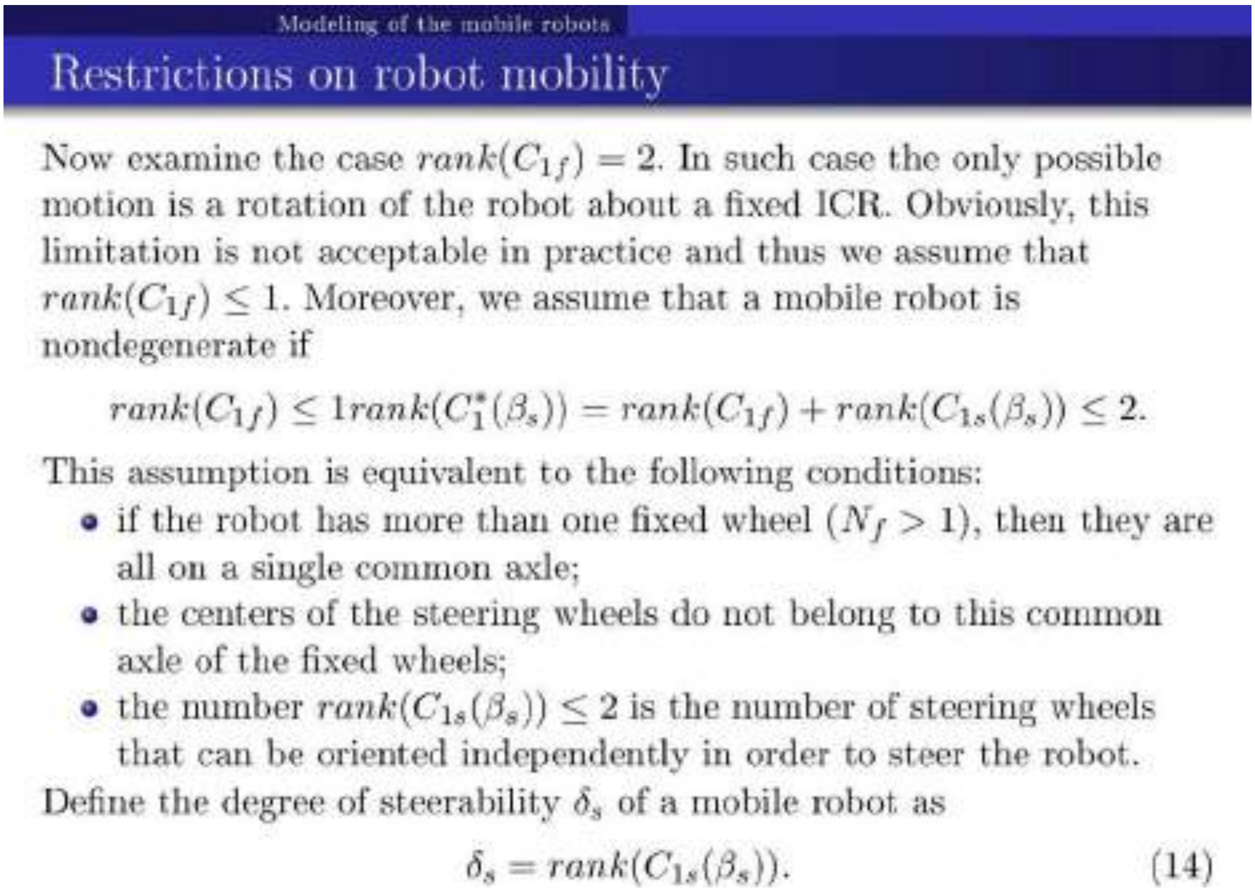
$$C_{1s}(\beta_s)R(\vartheta)\dot{\xi} = 0. \tag{11}$$

These constraints imply that the vector $R(\vartheta)\dot{\xi} \in \aleph(C_1^*(\beta_s))$, where

$$C_1^*(\beta_s) = \begin{bmatrix} C_{1f} \\ C_{1s}(\beta_s) \end{bmatrix}. \tag{12}$$

Obviously, it is $rank(C_1^*(\beta_s)) \leq 3$. If $rank(C_1^*(\beta_s)) = 3$, then $R(\vartheta)\dot{\xi} = 0$ and any motion in the plane is impossible!
Define the degree of mobility δ_m of a mobile robot as

$$\delta_m = dim(\aleph(C_1^*(\beta_s))) = 3 - rank(C_1^*(\beta_s)). \tag{13}$$



Modeling of the mobile robots

Restrictions on robot mobility

Now examine the case $rank(C_{1f}) = 2$. In such case the only possible motion is a rotation of the robot about a fixed ICR. Obviously, this limitation is not acceptable in practice and thus we assume that $rank(C_{1f}) \leq 1$. Moreover, we assume that a mobile robot is nondegenerate if

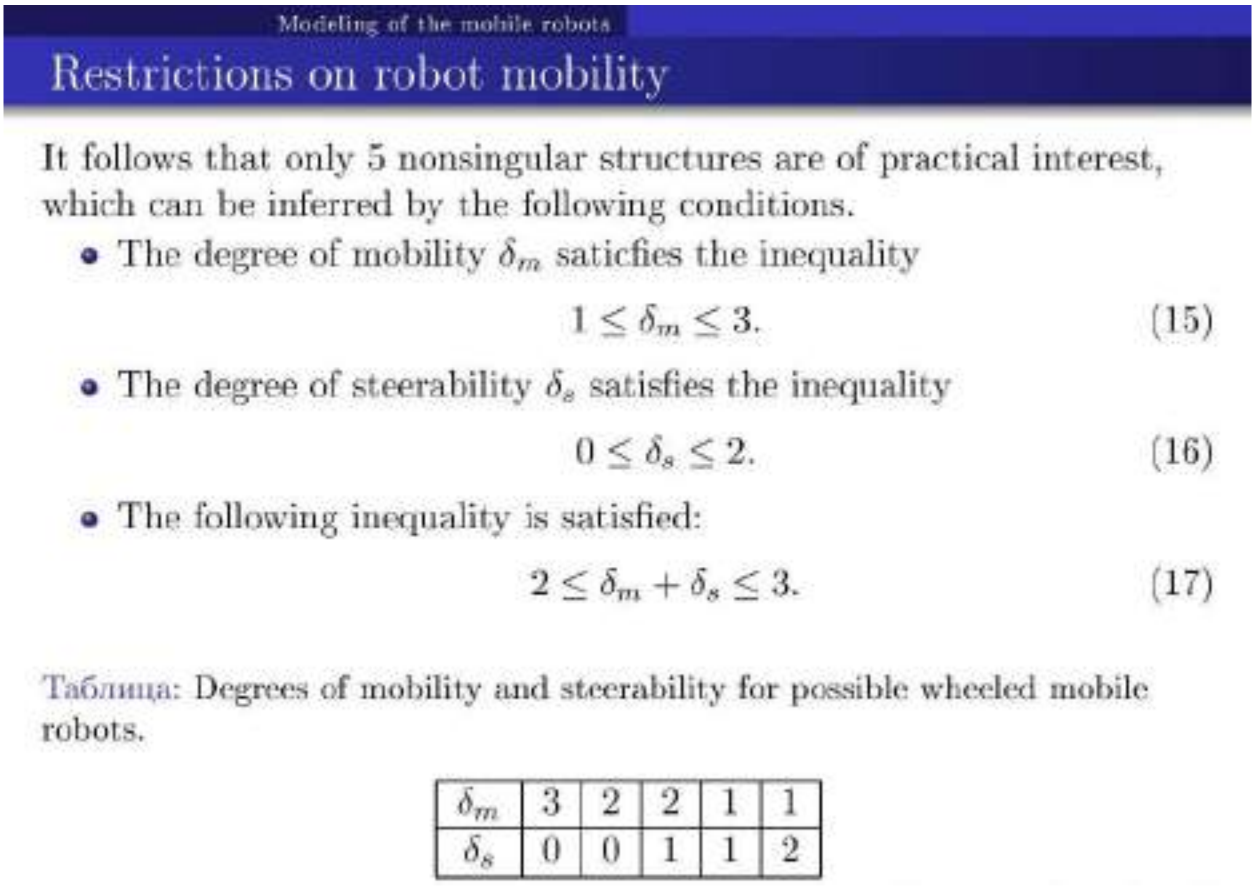
$$rank(C_{1f}) \leq 1 rank(C_1^*(\beta_s)) = rank(C_{1f}) + rank(C_{1s}(\beta_s)) \leq 2.$$

This assumption is equivalent to the following conditions:

- if the robot has more than one fixed wheel ($N_f > 1$), then they are all on a single common axle;
- the centers of the steering wheels do not belong to this common axle of the fixed wheels;
- the number $rank(C_{1s}(\beta_s)) \leq 2$ is the number of steering wheels that can be oriented independently in order to steer the robot.

Define the degree of steerability δ_s of a mobile robot as

$$\delta_s = rank(C_{1s}(\beta_s)). \tag{14}$$



Modeling of the mobile robots

Restrictions on robot mobility

It follows that only 5 nonsingular structures are of practical interest, which can be inferred by the following conditions.

- The degree of mobility δ_m saticfies the inequality

$$1 \leq \delta_m \leq 3. \tag{15}$$

- The degree of steerability δ_s satisfies the inequality

$$0 \leq \delta_s \leq 2. \tag{16}$$

- The following inequality is satisfied:

$$2 \leq \delta_m + \delta_s \leq 3. \tag{17}$$

Таблица: Degrees of mobility and steerability for possible wheeled mobile robots.

δ_m	3	2	2	1	1
δ_s	0	0	1	1	2

Type 3, 0 robot with swedish wheels

The constrains have from (8) where

$$J_1 = J_{1sw} = \begin{bmatrix} -\sqrt{3}/2 & 1/2 & L \\ 0 & -1 & L \\ \sqrt{3}/2 & 1/2 & L \end{bmatrix},$$

$$J_2 = \begin{bmatrix} r & 0 & 0 \\ 0 & r & 0 \\ 0 & 0 & r \end{bmatrix}.$$

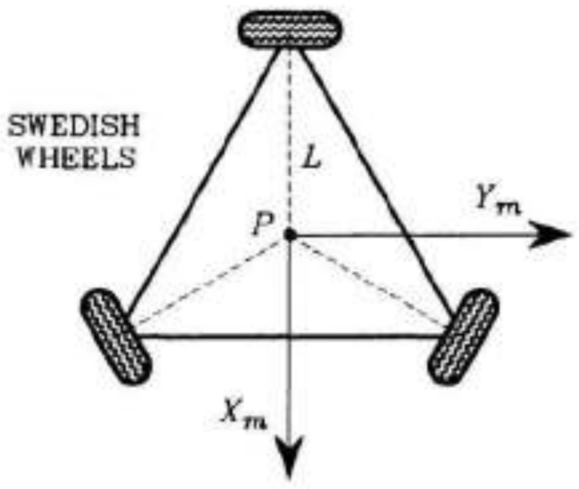


Таблица: Characteristic constants of type 3, 0 robot with swedish wheels.

Wheels	α	β	γ	l
$1sw$	$\pi/3$	0	0	L
$2sw$	π	0	0	L
$3sw$	$5\pi/3$	0	0	L

Types of possible wheeled mobile robots

- Type (3, 0) robot. In this case it is

$$\delta_m = dim(\aleph(C_1^*(\beta_s))) = 3 \qquad \delta_s = 0.$$

- Type (2, 0) robot. In this case it is

$$\delta_m = dim(\aleph(C_1^*(\beta_s))) = dim(\aleph(C_{1f})) = 2 \qquad \delta_s = 0.$$

- Type (2, 1) robot. In this case it is

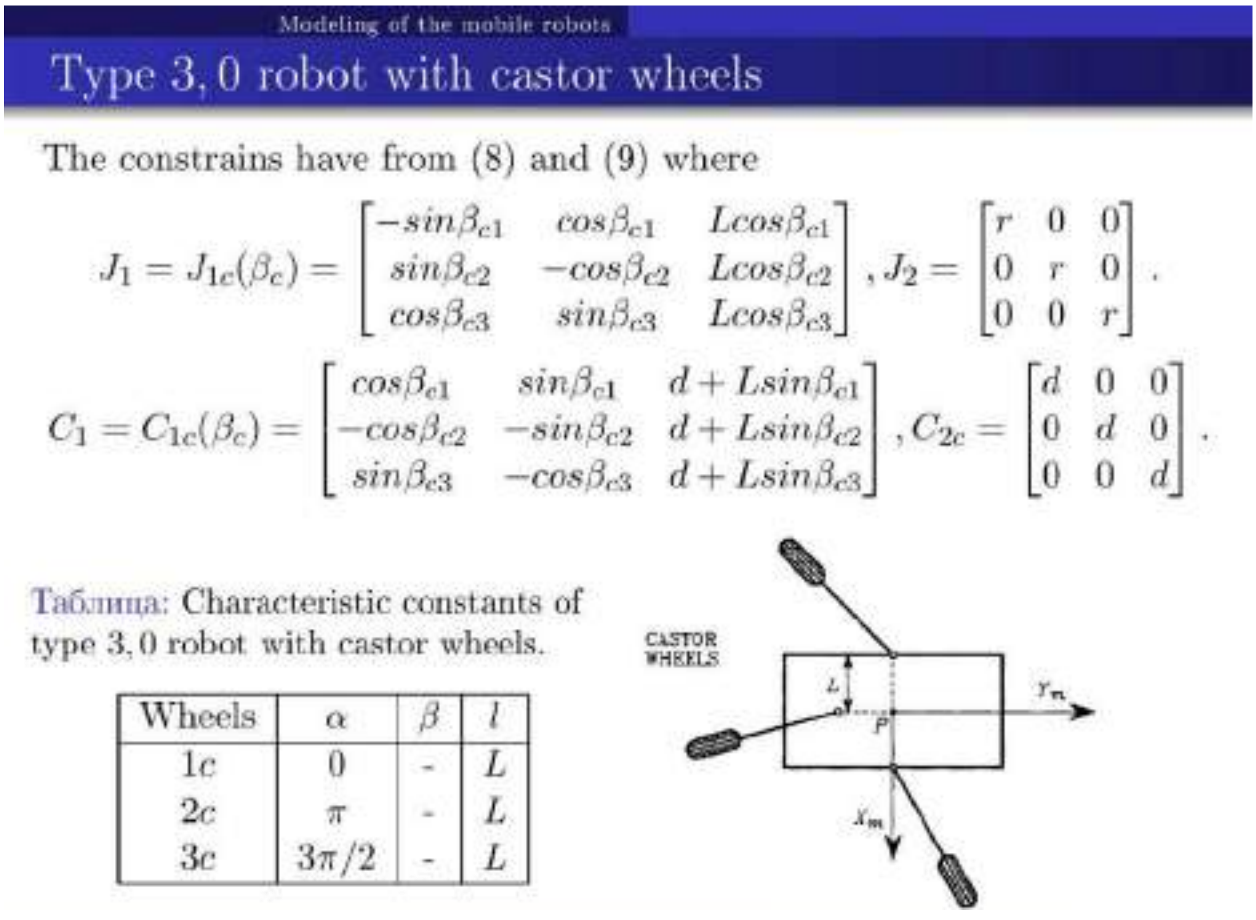
$$\delta_m = dim(\aleph(C_1^*(\beta_s))) = dim(\aleph(C_{1s}(\beta_s))) = 3 \qquad \delta_s = 1.$$

- Type (1, 1) robot. In this case it is

$$\delta_m = dim(\aleph(C_1^*(\beta_s))) = 1 \qquad \delta_s = 1.$$

- Type (1, 2) robot. In this case it is

$$\delta_m = dim(\aleph(C_1^*(\beta_s))) = dim(\aleph(C_{1s}(\beta_s))) = 1 \qquad \delta_s = 2.$$



Modeling of the mobile robots

Type 3, 0 robot with castor wheels

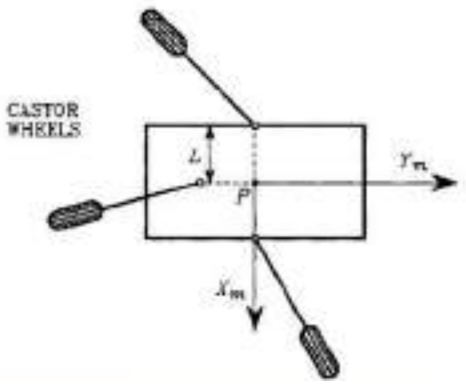
The constrains have from (8) and (9) where

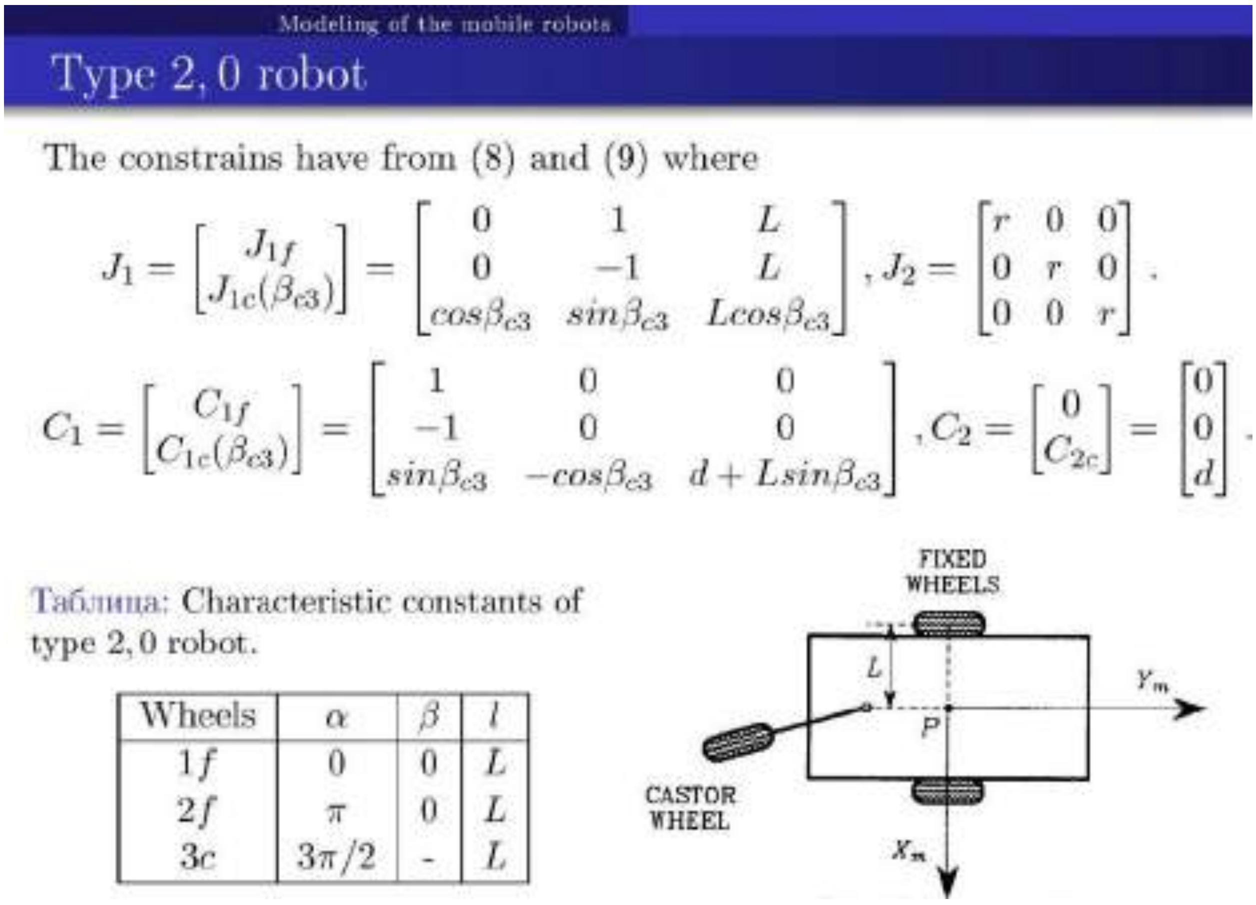
$$J_1 = J_{1c}(\beta_c) = \begin{bmatrix} -sin\beta_{c1} & cos\beta_{c1} & Lcos\beta_{c1} \\ sin\beta_{c2} & -cos\beta_{c2} & Lcos\beta_{c2} \\ cos\beta_{c3} & sin\beta_{c3} & Lcos\beta_{c3} \end{bmatrix}, J_2 = \begin{bmatrix} r & 0 & 0 \\ 0 & r & 0 \\ 0 & 0 & r \end{bmatrix}.$$

$$C_1 = C_{1c}(\beta_c) = \begin{bmatrix} cos\beta_{c1} & sin\beta_{c1} & d + Lsin\beta_{c1} \\ -cos\beta_{c2} & -sin\beta_{c2} & d + Lsin\beta_{c2} \\ sin\beta_{c3} & -cos\beta_{c3} & d + Lsin\beta_{c3} \end{bmatrix}, C_{2c} = \begin{bmatrix} d & 0 & 0 \\ 0 & d & 0 \\ 0 & 0 & d \end{bmatrix}.$$

Таблица: Characteristic constants of type 3, 0 robot with castor wheels.

Wheels	α	β	l
$1c$	0	-	L
$2c$	π	-	L
$3c$	$3\pi/2$	-	L





Modeling of the mobile robots

Type 2, 0 robot

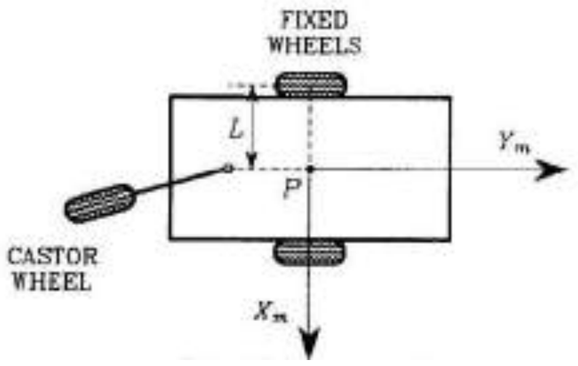
The constrains have from (8) and (9) where

$$J_1 = \begin{bmatrix} J_{1f} \\ J_{1c}(\beta_{c3}) \end{bmatrix} = \begin{bmatrix} 0 & 1 & L \\ 0 & -1 & L \\ cos\beta_{c3} & sin\beta_{c3} & Lcos\beta_{c3} \end{bmatrix}, J_2 = \begin{bmatrix} r & 0 & 0 \\ 0 & r & 0 \\ 0 & 0 & r \end{bmatrix}.$$

$$C_1 = \begin{bmatrix} C_{1f} \\ C_{1c}(\beta_{c3}) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ sin\beta_{c3} & -cos\beta_{c3} & d + Lsin\beta_{c3} \end{bmatrix}, C_2 = \begin{bmatrix} 0 \\ C_{2c} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ d \end{bmatrix}.$$

Таблица: Characteristic constants of type 2, 0 robot.

Wheels	α	β	l
$1f$	0	0	L
$2f$	π	0	L
$3c$	$3\pi/2$	-	L



Modeling of the mobile robots

Type 2, 1 robot

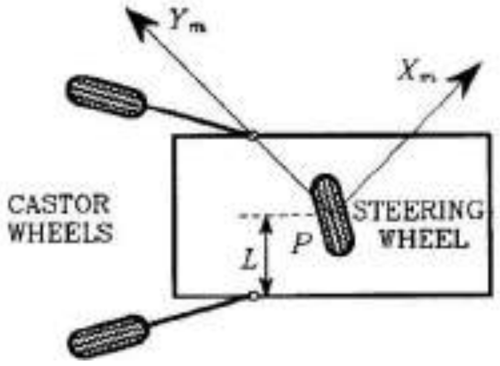
The constrains have from (8) and (9) where

$$J_1 = \begin{bmatrix} J_{1s}(\beta_{s1}) \\ J_{1c}(\beta_{c2}, \beta_{c3}) \end{bmatrix} = \begin{bmatrix} -sin\beta_{s1} & cos\beta_{s1} & 0 \\ -cos\beta_{c2} & -sin\beta_{c2} & \sqrt{2}Lcos\beta_{c2} \\ sin\beta_{c3} & cos\beta_{c3} & \sqrt{2}Lcos\beta_{c3} \end{bmatrix}, J_2 = \begin{bmatrix} r & 0 & 0 \\ 0 & r & 0 \\ 0 & 0 & r \end{bmatrix}$$

$$C_1 = \begin{bmatrix} C_{1s}(\beta_{s1}) \\ C_{1c}(\beta_{c2}, \beta_{c3}) \end{bmatrix} = \begin{bmatrix} cos\beta_{s1} & sin\beta_{s1} & 0 \\ -sin\beta_{c2} & cos\beta_{c2} & d+\sqrt{2}Lsin\beta_{c2} \\ -cos\beta_{c3} & -sin\beta_{c3} & d+\sqrt{2}Lsin\beta_{c3} \end{bmatrix}, C_2 = \begin{bmatrix} 0 \\ C_{2c} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ d & 0 \\ 0 & d \end{bmatrix}$$

Таблица: Characteristic constants of type 2, 1 robot.

Wheels	α	β	l
1s	0	-	0
2c	$\pi/2$	-	$L\sqrt{2}$
3c	π	-	$L\sqrt{2}$



Modeling of the mobile robots

Type 1, 1 robot

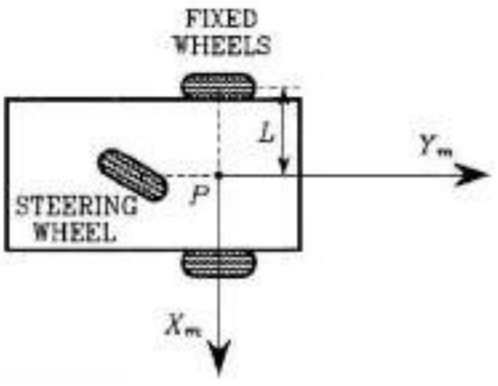
The constrains have from (8) and (9) where

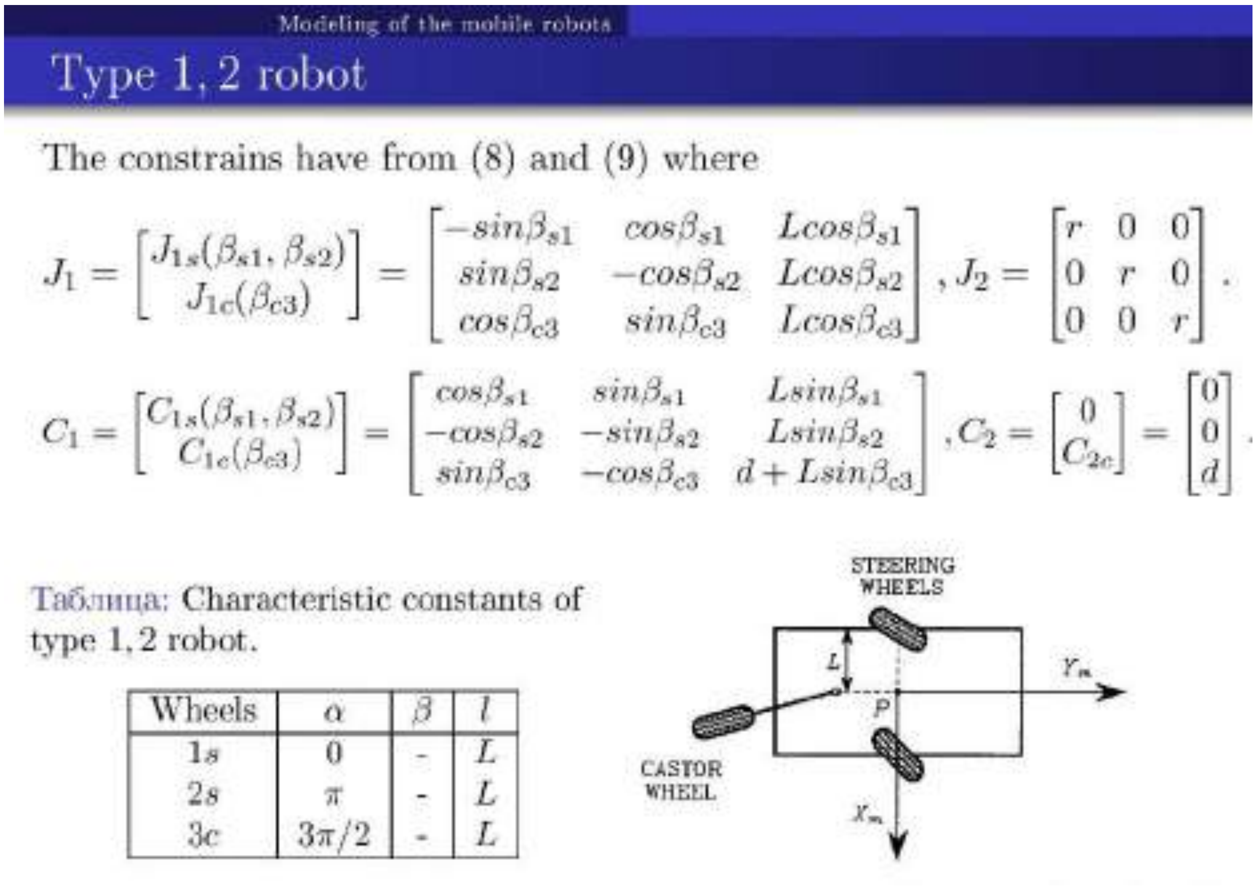
$$J_1 = \begin{bmatrix} J_{1f} \\ J_{1s}(\beta_{s3}) \end{bmatrix} = \begin{bmatrix} 0 & 1 & L \\ 0 & -1 & L \\ cos\beta_{s3} & sin\beta_{s3} & Lcos\beta_{s3} \end{bmatrix}, J_2 = \begin{bmatrix} r & 0 & 0 \\ 0 & r & 0 \\ 0 & 0 & r \end{bmatrix}.$$

$$C_1 = \begin{bmatrix} C_{1f} \\ C_{1s}(\beta_{c3}) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ sin\beta_{s3} & -cos\beta_{s3} & Lsin\beta_{s3} \end{bmatrix}, C_2 = 0.$$

Таблица: Characteristic constants of type 1, 1 robot.

Wheels	α	β	l
$1f$	0	0	L
$2f$	π	0	L
$3s$	$3\pi/2$	-	L





Type 1, 2 robot

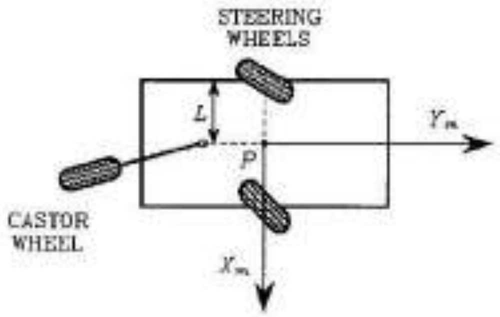
The constrains have from (8) and (9) where

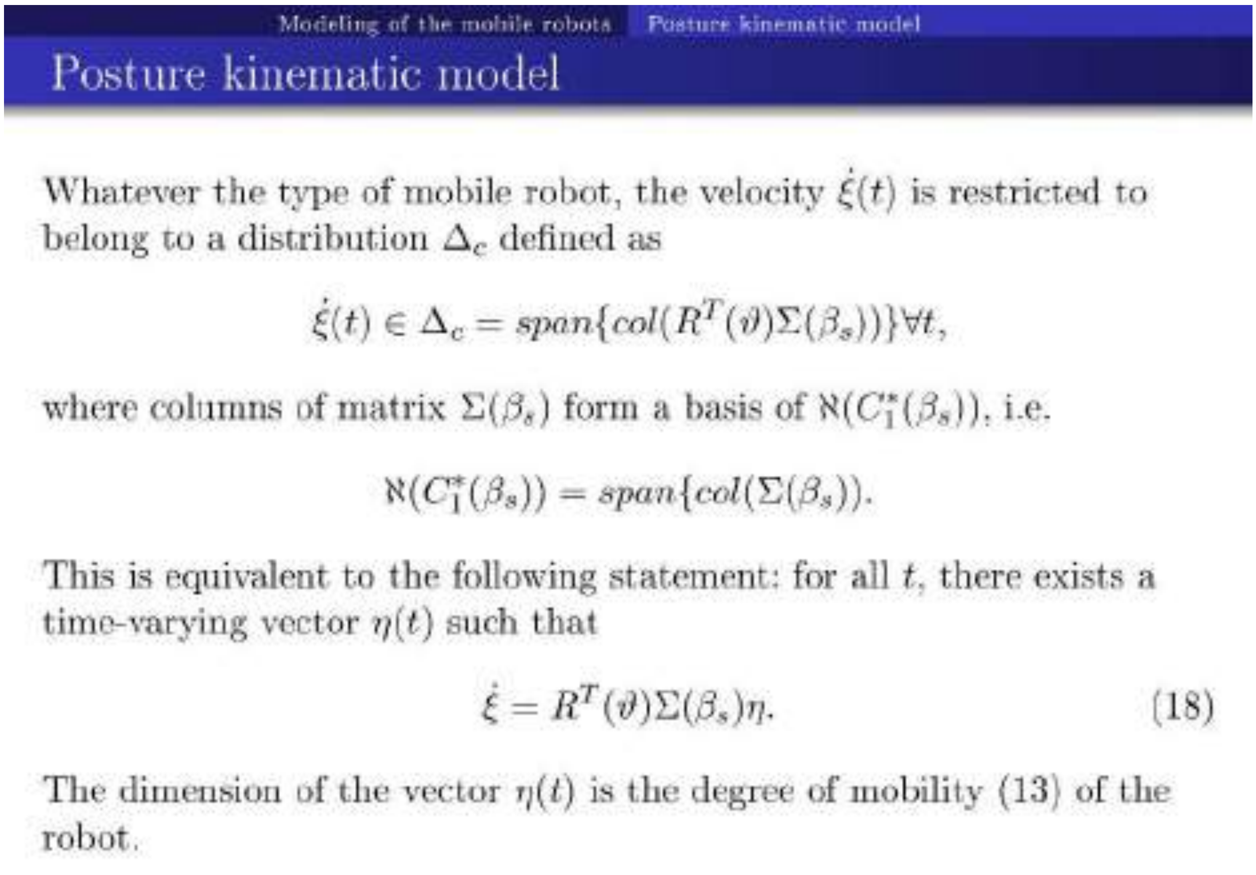
$$J_1 = \begin{bmatrix} J_{1s}(\beta_{s1}, \beta_{s2}) \\ J_{1c}(\beta_{c3}) \end{bmatrix} = \begin{bmatrix} -sin\beta_{s1} & cos\beta_{s1} & Lcos\beta_{s1} \\ sin\beta_{s2} & -cos\beta_{s2} & Lcos\beta_{s2} \\ cos\beta_{c3} & sin\beta_{c3} & Lcos\beta_{c3} \end{bmatrix}, J_2 = \begin{bmatrix} r & 0 & 0 \\ 0 & r & 0 \\ 0 & 0 & r \end{bmatrix}.$$

$$C_1 = \begin{bmatrix} C_{1s}(\beta_{s1}, \beta_{s2}) \\ C_{1c}(\beta_{c3}) \end{bmatrix} = \begin{bmatrix} cos\beta_{s1} & sin\beta_{s1} & Lsin\beta_{s1} \\ -cos\beta_{s2} & -sin\beta_{s2} & Lsin\beta_{s2} \\ sin\beta_{c3} & -cos\beta_{c3} & d + Lsin\beta_{c3} \end{bmatrix}, C_2 = \begin{bmatrix} 0 \\ C_{2c} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ d \end{bmatrix}.$$

Таблица: Characteristic constants of type 1, 2 robot.

Wheels	α	β	l
$1s$	0	-	L
$2s$	π	-	L
$3c$	$3\pi/2$	-	L





Posture kinematic model

Whatever the type of mobile robot, the velocity $\dot{\xi}(t)$ is restricted to belong to a distribution Δ_c defined as

$$\dot{\xi}(t) \in \Delta_c = span\{col(R^T(\vartheta)\Sigma(\beta_s))\} \forall t,$$

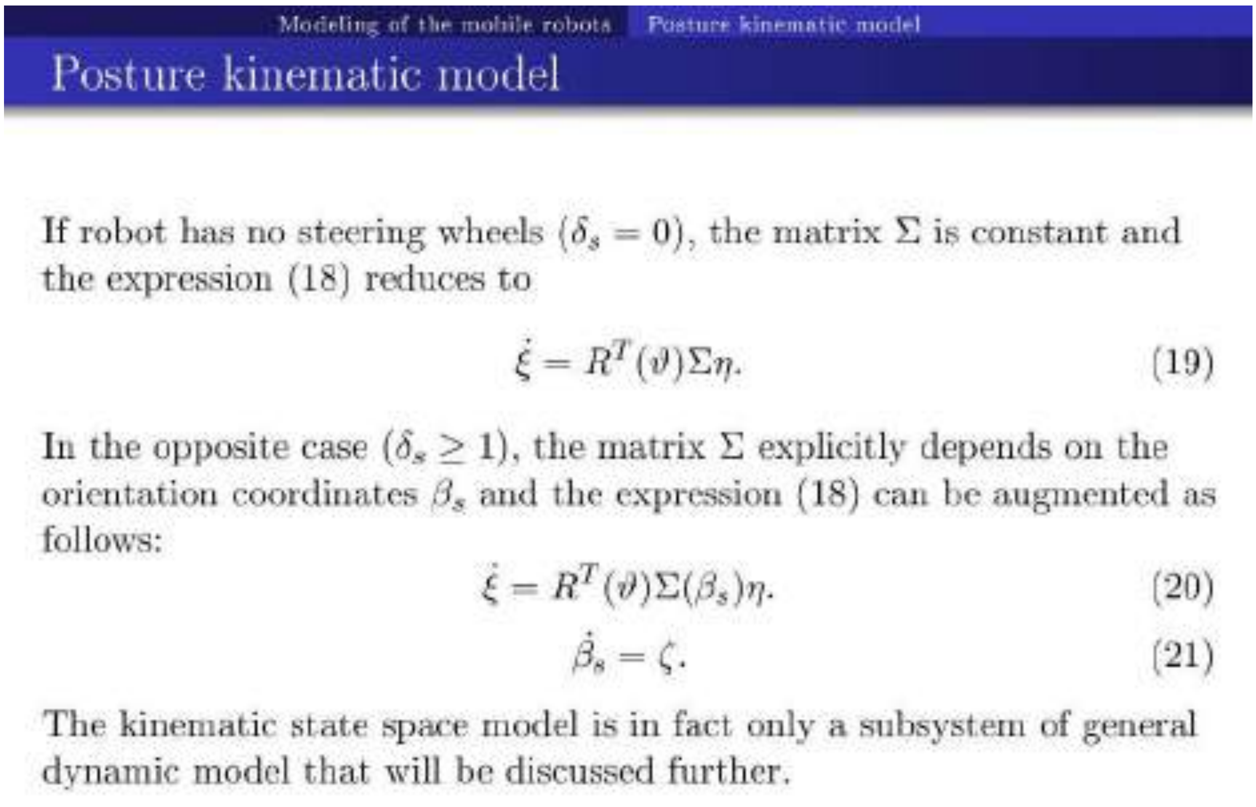
where columns of matrix $\Sigma(\beta_s)$ form a basis of $\aleph(C_1^*(\beta_s))$, i.e.

$$\aleph(C_1^*(\beta_s)) = span\{col(\Sigma(\beta_s)).$$

This is equivalent to the following statement: for all t, there exists a time-varying vector $\eta(t)$ such that

$$\dot{\xi} = R^T(\vartheta)\Sigma(\beta_s)\eta. \tag{18}$$

The dimension of the vector $\eta(t)$ is the degree of mobility (13) of the robot.



Posture kinematic model

If robot has no steering wheels ($\delta_s = 0$), the matrix Σ is constant and the expression (18) reduces to

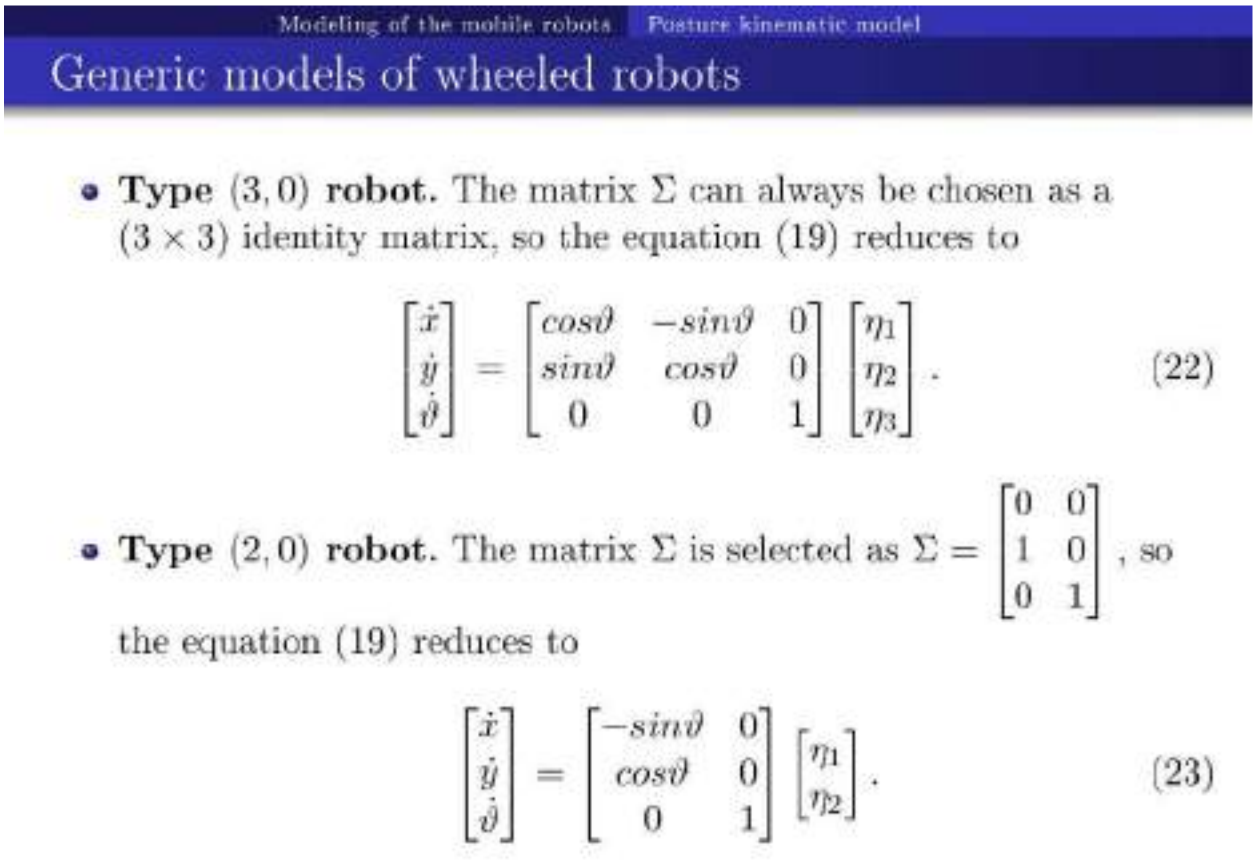
$$\dot{\xi} = R^T(\vartheta)\Sigma\eta. \tag{19}$$

In the opposite case ($\delta_s \geq 1$), the matrix Σ explicitly depends on the orientation coordinates β_s and the expression (18) can be augmented as follows:

$$\dot{\xi} = R^T(\vartheta)\Sigma(\beta_s)\eta. \tag{20}$$

$$\dot{\beta}_s = \zeta. \tag{21}$$

The kinematic state space model is in fact only a subsystem of general dynamic model that will be discussed further.



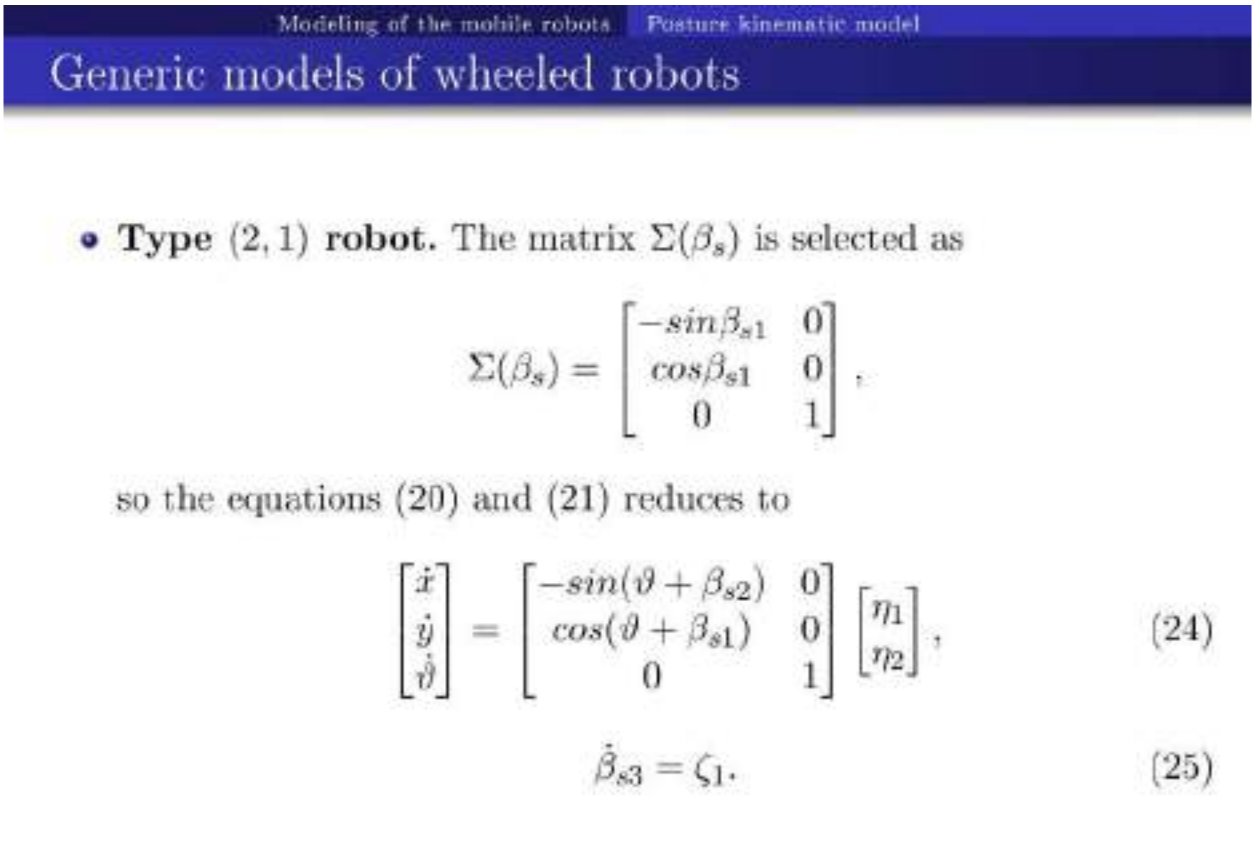
Generic models of wheeled robots

- **Type** $(3,0)$ **robot.** The matrix Σ can always be chosen as a (3×3) identity matrix, so the equation (19) reduces to

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\vartheta} \end{bmatrix} = \begin{bmatrix} cos\vartheta & -sin\vartheta & 0 \\ sin\vartheta & cos\vartheta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \eta_1 \\ \eta_2 \\ \eta_3 \end{bmatrix}. \tag{22}$$

- **Type** $(2,0)$ **robot.** The matrix Σ is selected as $\Sigma = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$, so the equation (19) reduces to

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\vartheta} \end{bmatrix} = \begin{bmatrix} -sin\vartheta & 0 \\ cos\vartheta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \eta_1 \\ \eta_2 \end{bmatrix}. \tag{23}$$



Modeling of the mobile robots | Posture kinematic model

Generic models of wheeled robots

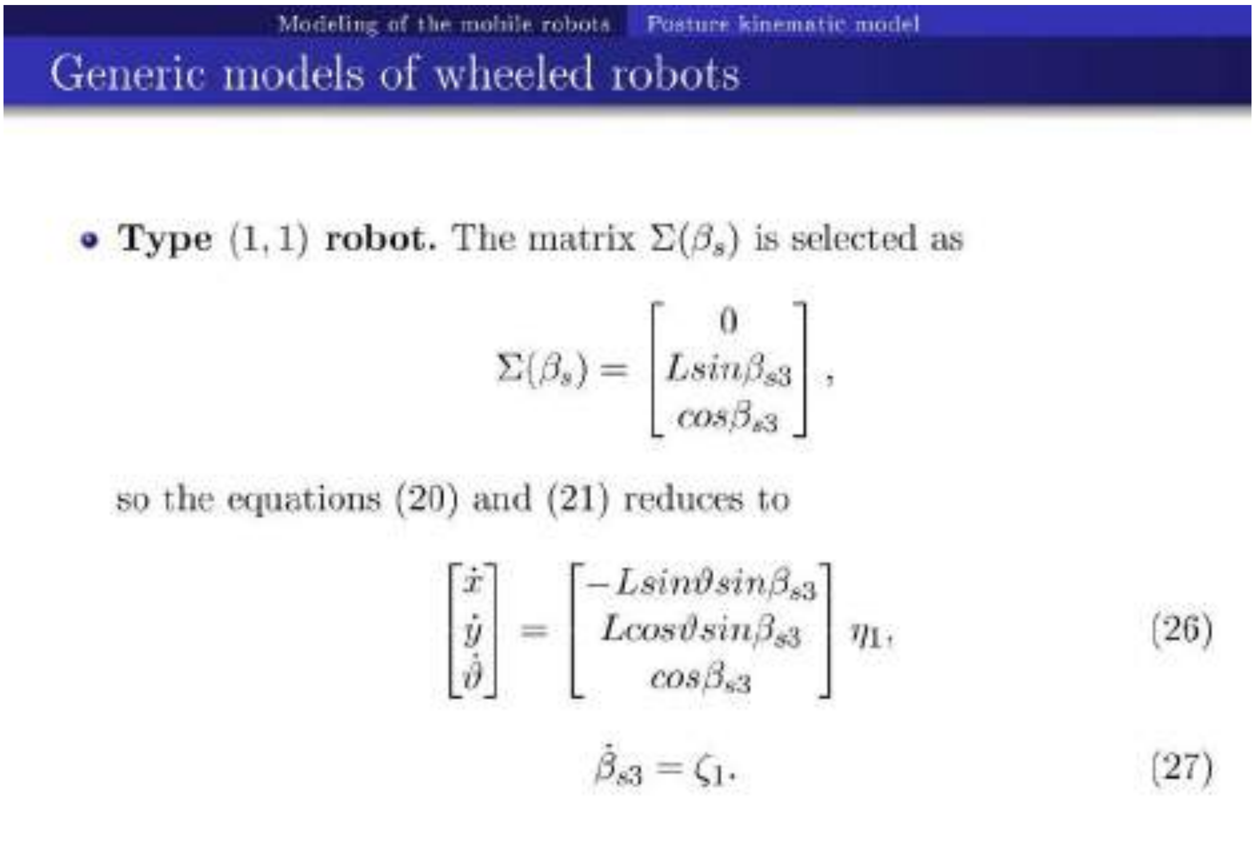
- **Type** (2, 1) **robot.** The matrix $\Sigma(\beta_s)$ is selected as

$$\Sigma(\beta_s) = \begin{bmatrix} -sin\beta_{s1} & 0 \\ cos\beta_{s1} & 0 \\ 0 & 1 \end{bmatrix},$$

so the equations (20) and (21) reduces to

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\vartheta} \end{bmatrix} = \begin{bmatrix} -sin(\vartheta + \beta_{s2}) & 0 \\ cos(\vartheta + \beta_{s1}) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \eta_1 \\ \eta_2 \end{bmatrix}, \tag{24}$$

$$\dot{\beta}_{s3} = \zeta_1. \tag{25}$$



Modeling of the mobile robots | Posture kinematic model

Generic models of wheeled robots

- **Type** (1, 1) **robot.** The matrix $\Sigma(\beta_s)$ is selected as

$$\Sigma(\beta_s) = \begin{bmatrix} 0 \\ Lsin\beta_{s3} \\ cos\beta_{s3} \end{bmatrix},$$

so the equations (20) and (21) reduces to

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\vartheta} \end{bmatrix} = \begin{bmatrix} -Lsin\vartheta sin\beta_{s3} \\ Lcos\vartheta sin\beta_{s3} \\ cos\beta_{s3} \end{bmatrix} \eta_1, \tag{26}$$

$$\dot{\beta}_{s3} = \zeta_1. \tag{27}$$

Generic models of wheeled robots

- **Type** $(1, 2)$ **robot.** The matrix $\Sigma(\beta_s)$ is selected as

$$\Sigma(\beta_s) = \begin{bmatrix} -2L sin\beta_{s1} sin\beta_{s2} \\ L sin(\beta_{s1} + \beta_{s2}) \\ sin(\beta_{s2} - \beta_{s1}) \end{bmatrix},$$

so the equations (20) and (21) reduces to

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\vartheta} \end{bmatrix} = \begin{bmatrix} -L(sin\beta_{s1} sin(\vartheta + \beta_{s2}) + sin\beta_{s2} sin(\vartheta + \beta_{s1})) \\ L(sin\beta_{s1} cos(\vartheta + \beta_{s2}) + sin\beta_{s2} cos(\vartheta + \beta_{s1})) \\ sin(\beta_{s2} - \beta_{s1}) \end{bmatrix} \eta_1, \quad (28)$$

$$\dot{\beta}_{s1} = \zeta_1. \quad (29)$$

$$\dot{\beta}_{s2} = \zeta_2. \quad (30)$$

Mobility, steerability and manoeuvrability

Rewrite the posture kinematic model in the compact form

$$\dot{z} = B(z)u, \quad (31)$$

where either (δ_s)

$$z = \xi, \quad B(z) = R^T(\vartheta)\Sigma, \quad u = \eta$$

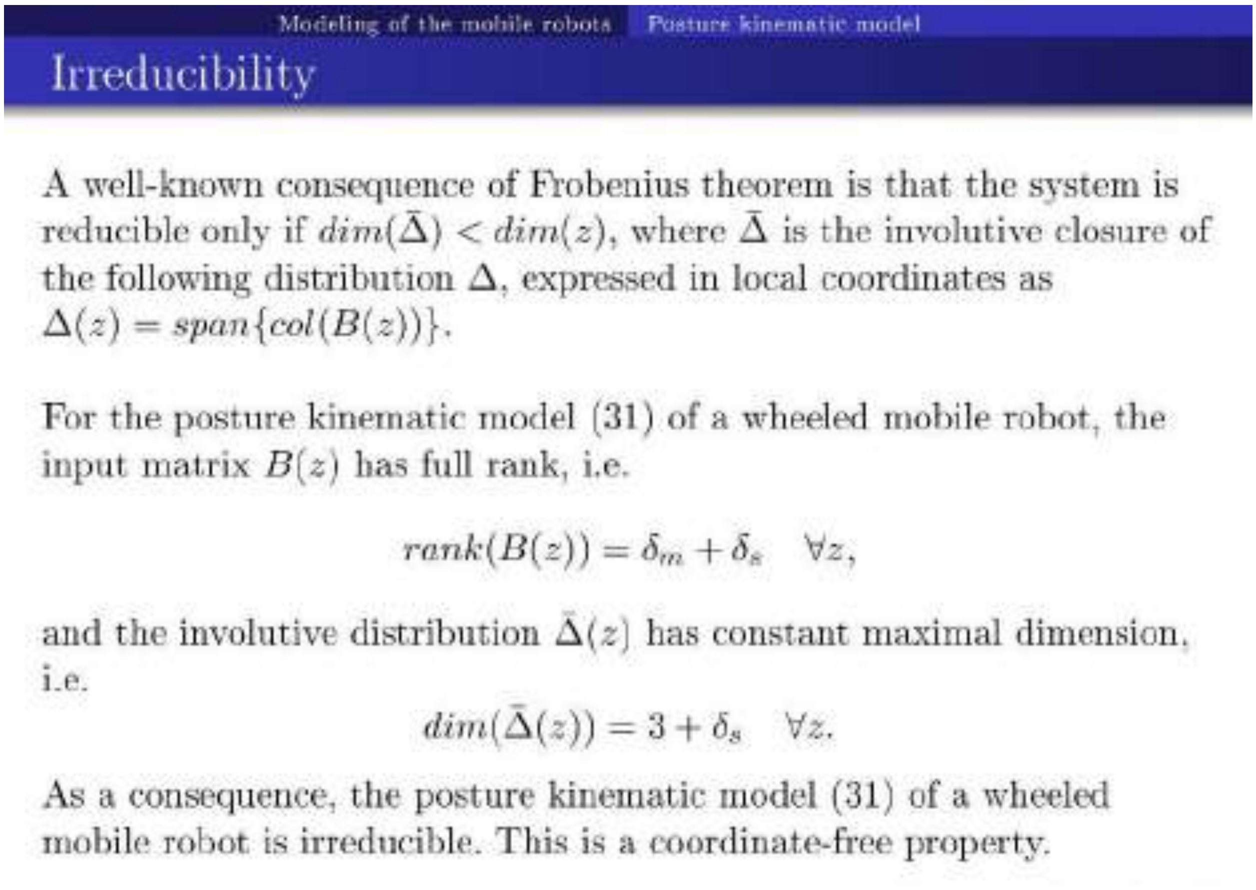
or $(\delta_s \geq 1)$

$$z = \begin{bmatrix} \xi \\ \beta_s \end{bmatrix}, \quad B(z) = \begin{bmatrix} R^T(\vartheta)\Sigma(\beta_s) & 0 \\ 0 & I \end{bmatrix}, \quad u = \begin{bmatrix} \eta \\ \zeta \end{bmatrix}.$$

Consider degree of manoeuvrability

$$\delta_M = \delta_m + \delta_s.$$

The ideal situation is that of omnidirectional mobile robots where $\delta_m = \delta_M = 3$.



Irreducibility

A well-known consequence of Frobenius theorem is that the system is reducible only if $dim(\bar{\Delta}) < dim(z)$, where $\bar{\Delta}$ is the involutive closure of the following distribution Δ, expressed in local coordinates as $\Delta(z) = span\{col(B(z))\}$.

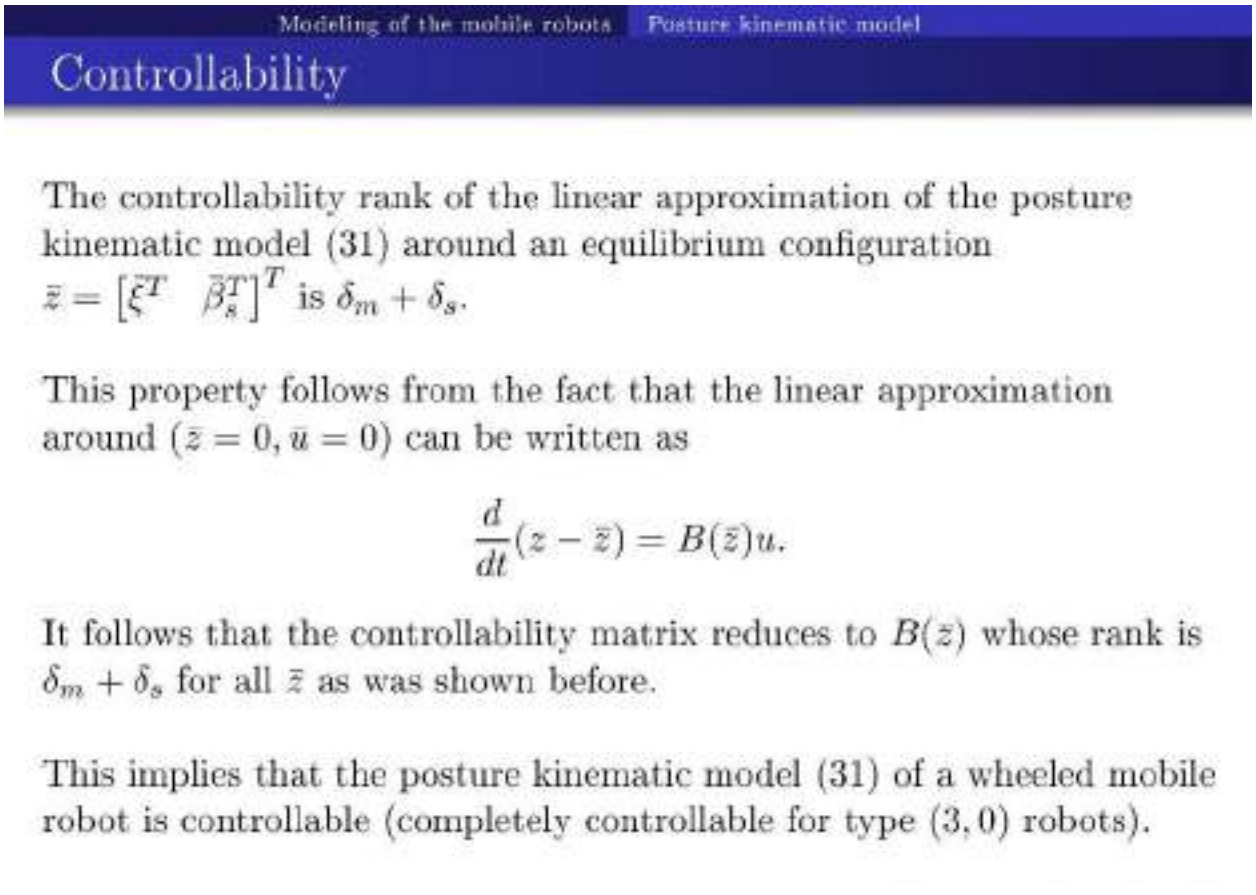
For the posture kinematic model (31) of a wheeled mobile robot, the input matrix $B(z)$ has full rank, i.e.

$$rank(B(z)) = \delta_m + \delta_s \quad \forall z,$$

and the involutive distribution $\bar{\Delta}(z)$ has constant maximal dimension, i.e.

$$dim(\bar{\Delta}(z)) = 3 + \delta_s \quad \forall z.$$

As a consequence, the posture kinematic model (31) of a wheeled mobile robot is irreducible. This is a coordinate-free property.



Controllability

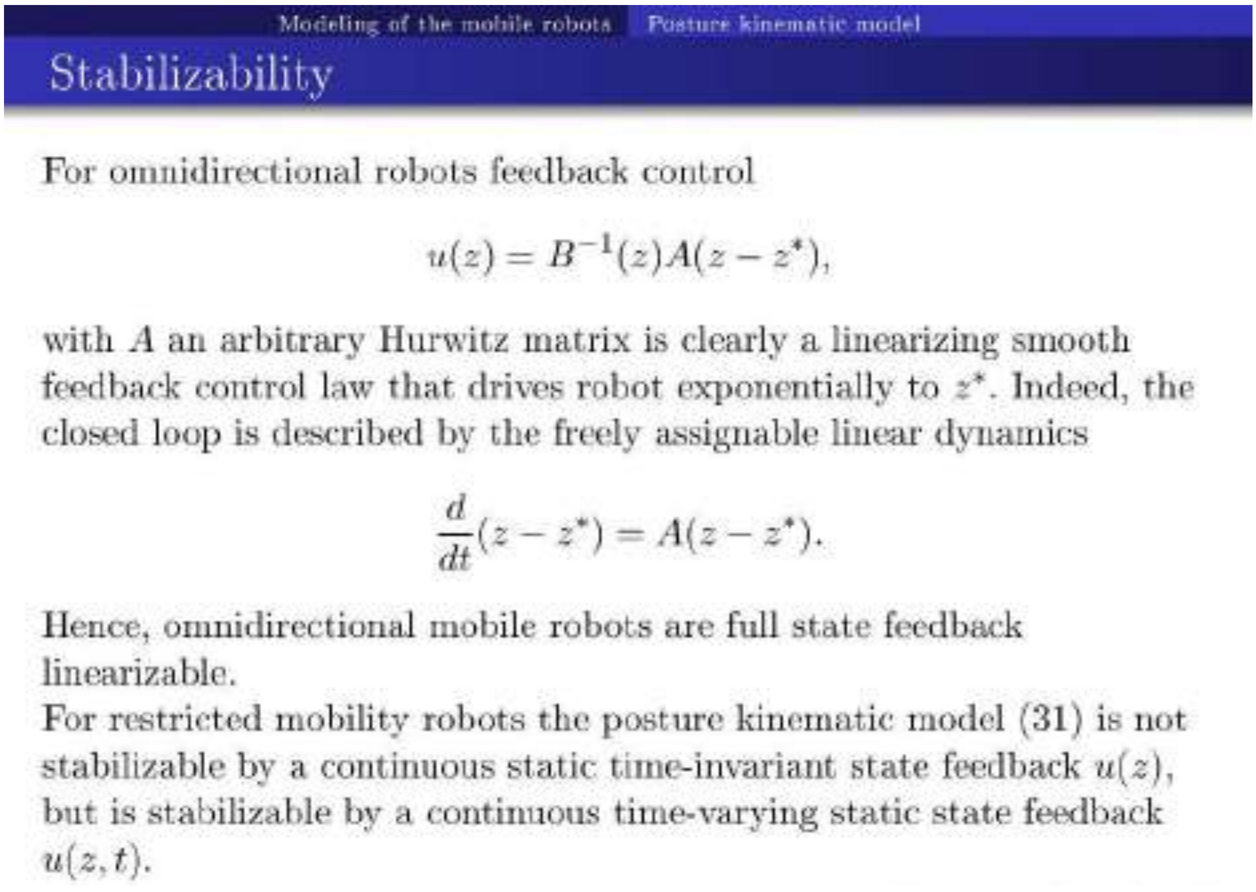
The controllability rank of the linear approximation of the posture kinematic model (31) around an equilibrium configuration $\bar{z} = \begin{bmatrix}\bar{\xi}^T & \bar{\beta}_s^T\end{bmatrix}^T$ is $\delta_m + \delta_s$.

This property follows from the fact that the linear approximation around $(\bar{z} = 0, \bar{u} = 0)$ can be written as

$$\frac{d}{dt}(z - \bar{z}) = B(\bar{z})u.$$

It follows that the controllability matrix reduces to $B(\bar{z})$ whose rank is $\delta_m + \delta_s$ for all $\bar{z}$ as was shown before.

This implies that the posture kinematic model (31) of a wheeled mobile robot is controllable (completely controllable for type (3, 0) robots).



Stabilizability

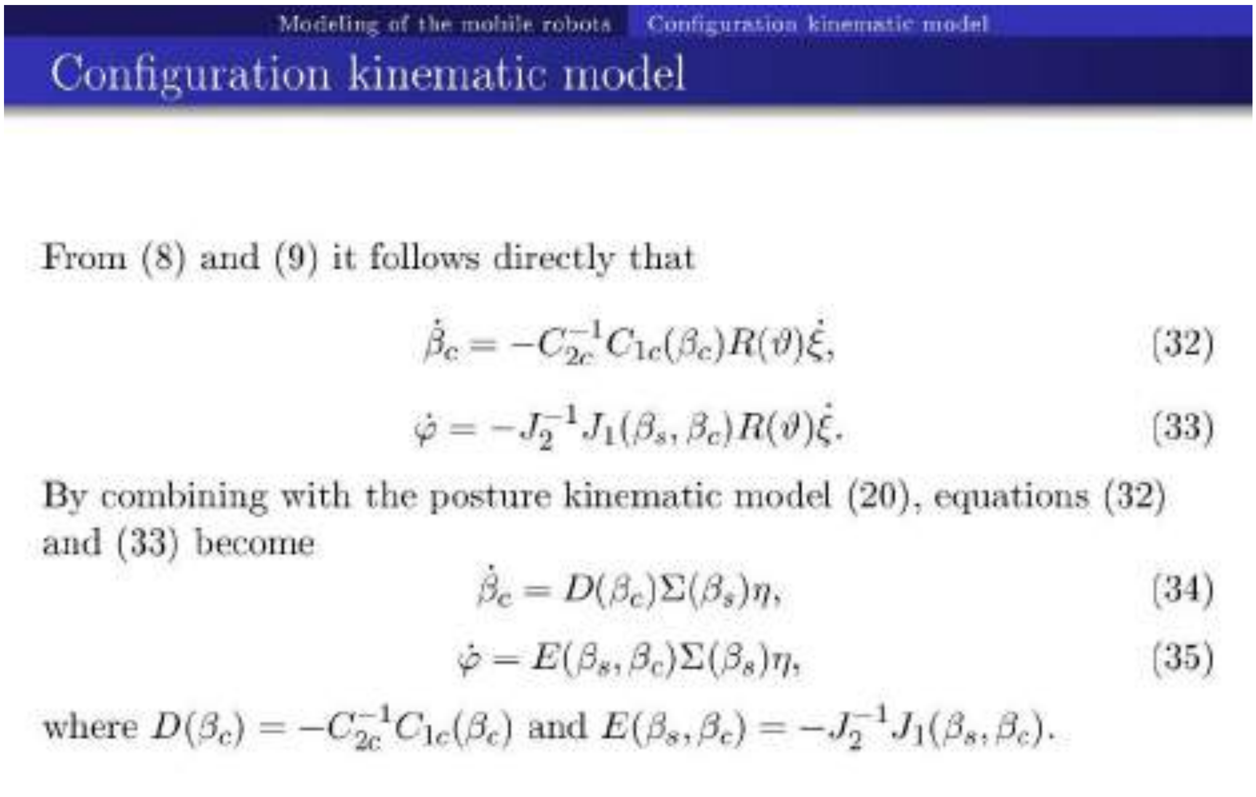
For omnidirectional robots feedback control

$$u(z) = B^{-1}(z)A(z - z^*),$$

with A an arbitrary Hurwitz matrix is clearly a linearizing smooth feedback control law that drives robot exponentially to z^*. Indeed, the closed loop is described by the freely assignable linear dynamics

$$\frac{d}{dt}(z - z^*) = A(z - z^*).$$

Hence, omnidirectional mobile robots are full state feedback linearizable.
For restricted mobility robots the posture kinematic model (31) is not stabilizable by a continuous static time-invariant state feedback $u(z)$, but is stabilizable by a continuous time-varying static state feedback $u(z, t)$.



Configuration kinematic model

From (8) and (9) it follows directly that

$$\dot{\beta}_c = -C_{2c}^{-1}C_{1c}(\beta_c)R(\vartheta)\dot{\xi}, \tag{32}$$

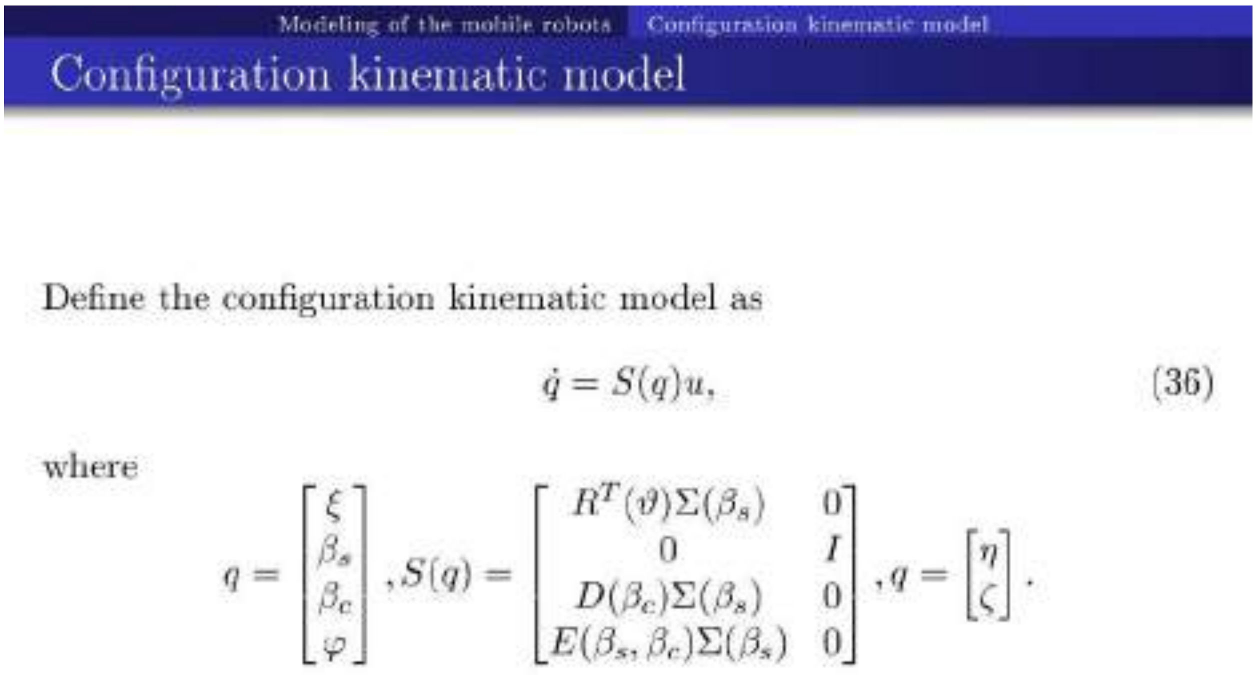
$$\dot{\varphi} = -J_2^{-1}J_1(\beta_s, \beta_c)R(\vartheta)\dot{\xi}. \tag{33}$$

By combining with the posture kinematic model (20), equations (32) and (33) become

$$\dot{\beta}_c = D(\beta_c)\Sigma(\beta_s)\eta, \tag{34}$$

$$\dot{\varphi} = E(\beta_s, \beta_c)\Sigma(\beta_s)\eta, \tag{35}$$

where $D(\beta_c) = -C_{2c}^{-1}C_{1c}(\beta_c)$ and $E(\beta_s, \beta_c) = -J_2^{-1}J_1(\beta_s, \beta_c)$.



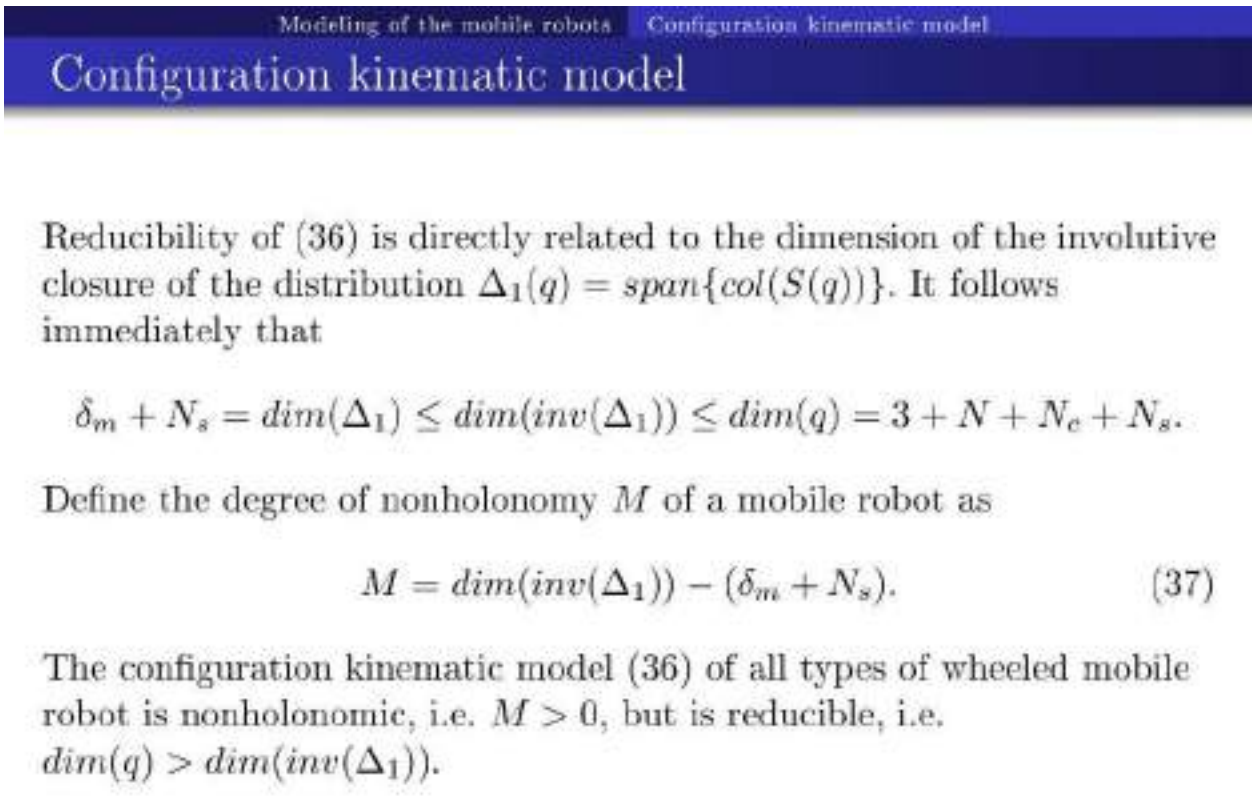
Configuration kinematic model

Define the configuration kinematic model as

$$\dot{q} = S(q)u, \tag{36}$$

where

$$q = \begin{bmatrix} \xi \\ \beta_s \\ \beta_c \\ \varphi \end{bmatrix}, S(q) = \begin{bmatrix} R^T(\vartheta)\Sigma(\beta_s) & 0 \\ 0 & I \\ D(\beta_c)\Sigma(\beta_s) & 0 \\ E(\beta_s, \beta_c)\Sigma(\beta_s) & 0 \end{bmatrix}, q = \begin{bmatrix} \eta \\ \zeta \end{bmatrix}.$$



Configuration kinematic model

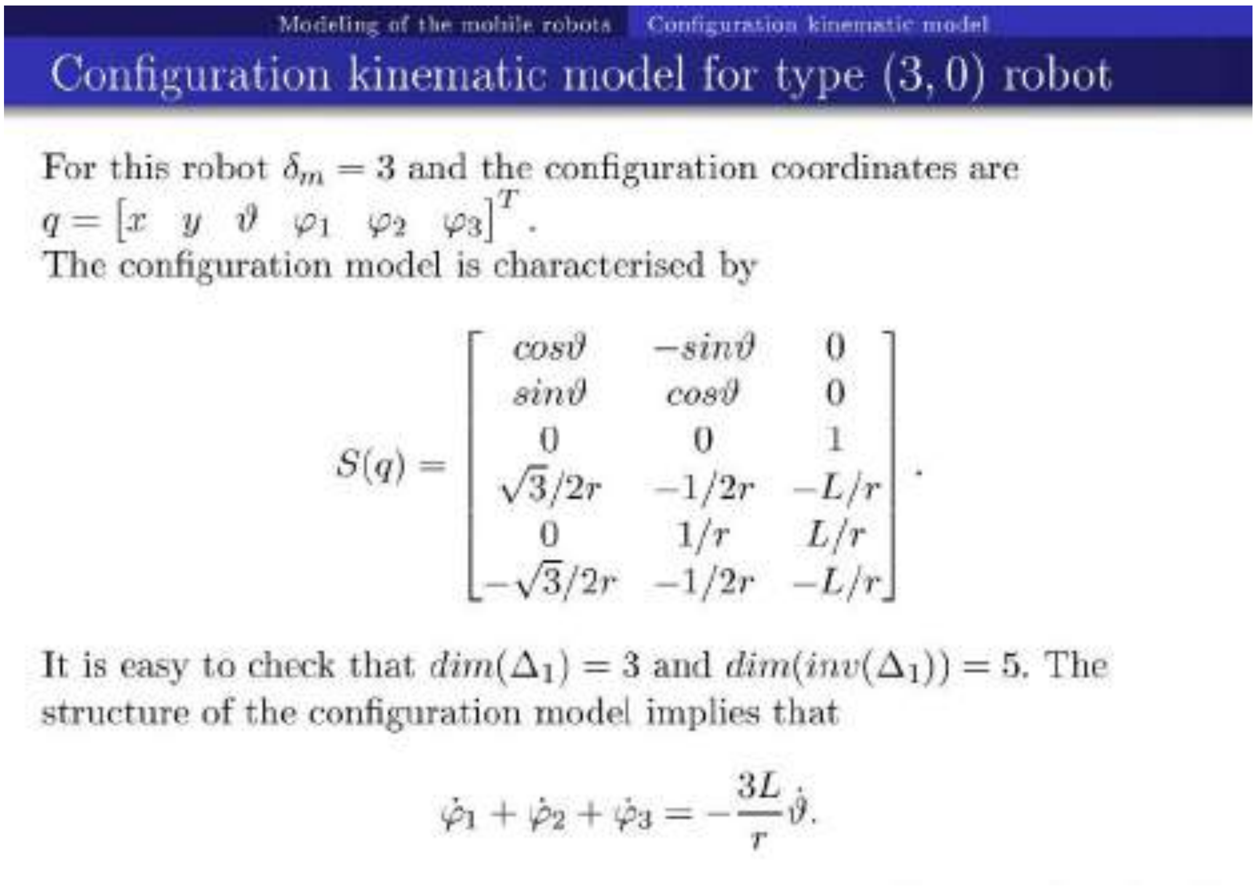
Reducibility of (36) is directly related to the dimension of the involutive closure of the distribution $\Delta_1(q) = span\{col(S(q))\}$. It follows immediately that

$$\delta_m + N_s = dim(\Delta_1) \leq dim(inv(\Delta_1)) \leq dim(q) = 3 + N + N_c + N_s.$$

Define the degree of nonholonomy M of a mobile robot as

$$M = dim(inv(\Delta_1)) - (\delta_m + N_s). \tag{37}$$

The configuration kinematic model (36) of all types of wheeled mobile robot is nonholonomic, i.e. $M > 0$, but is reducible, i.e. $dim(q) > dim(inv(\Delta_1))$.



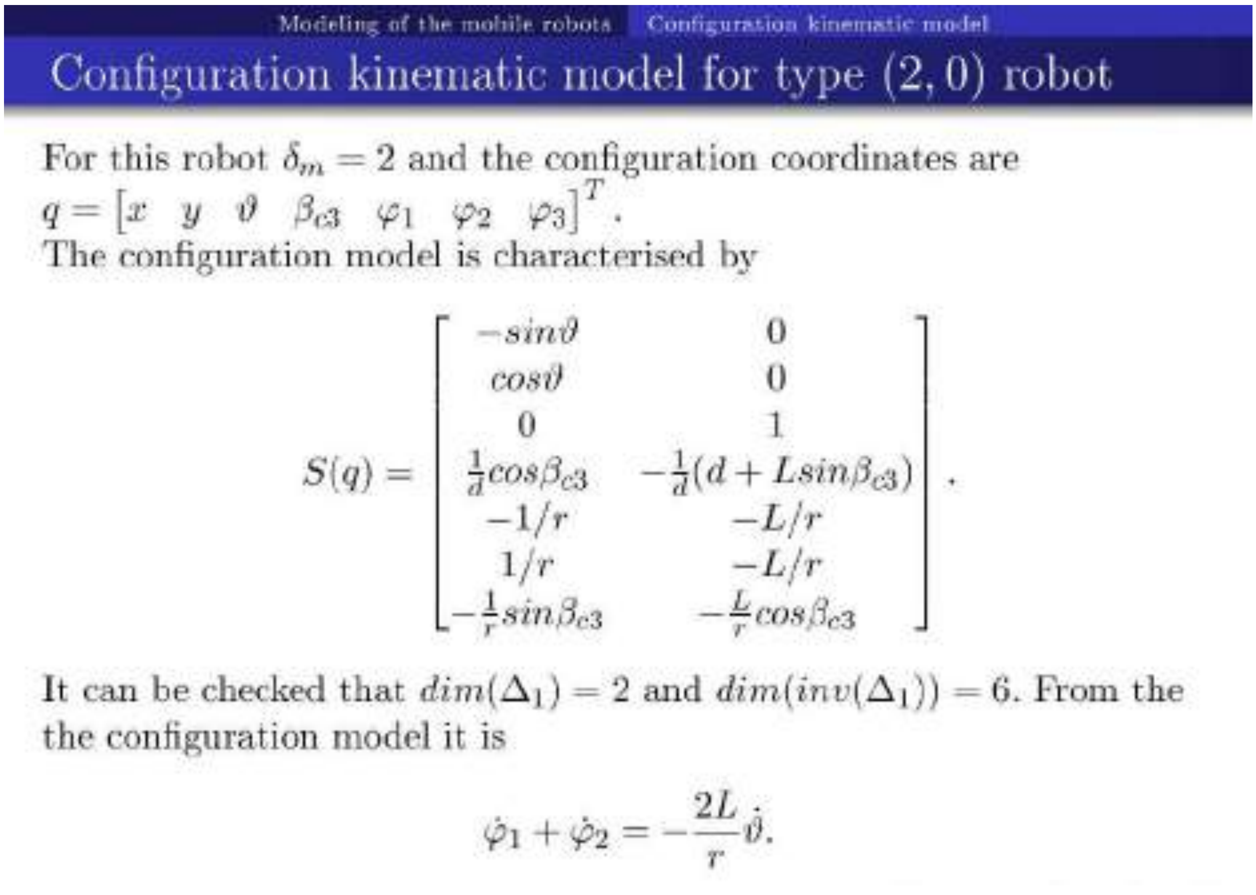
Configuration kinematic model for type (3, 0) robot

For this robot $\delta_m = 3$ and the configuration coordinates are
$q = \begin{bmatrix} x & y & \vartheta & \varphi_1 & \varphi_2 & \varphi_3 \end{bmatrix}^T$.
The configuration model is characterised by

$$S(q) = \begin{bmatrix} cos\vartheta & -sin\vartheta & 0 \\ sin\vartheta & cos\vartheta & 0 \\ 0 & 0 & 1 \\ \sqrt{3}/2r & -1/2r & -L/r \\ 0 & 1/r & L/r \\ -\sqrt{3}/2r & -1/2r & -L/r \end{bmatrix}.$$

It is easy to check that $dim(\Delta_1) = 3$ and $dim(inv(\Delta_1)) = 5$. The structure of the configuration model implies that

$$\dot{\varphi}_1 + \dot{\varphi}_2 + \dot{\varphi}_3 = -\frac{3L}{r}\dot{\vartheta}.$$



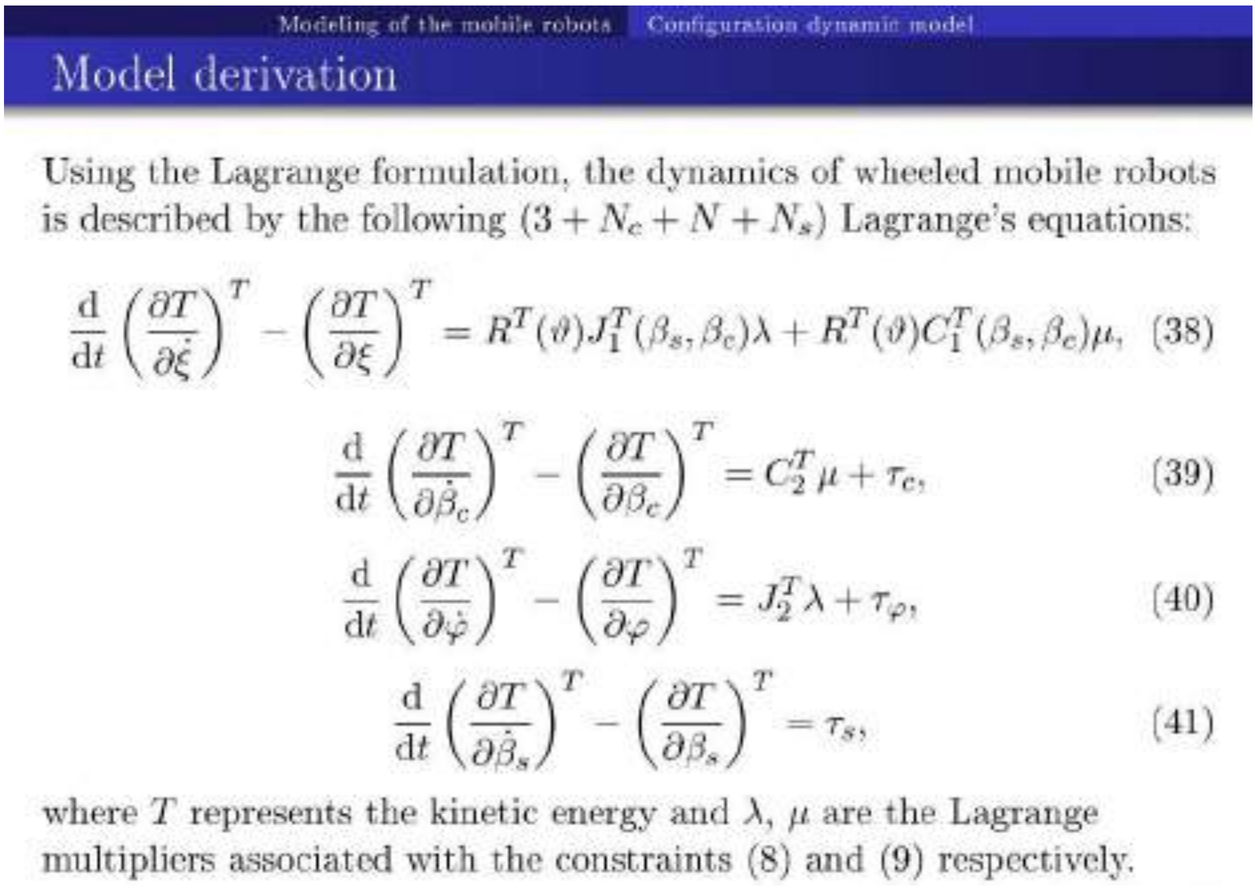
Configuration kinematic model for type (2, 0) robot

For this robot $\delta_m = 2$ and the configuration coordinates are
$q = \begin{bmatrix} x & y & \vartheta & \beta_{c3} & \varphi_1 & \varphi_2 & \varphi_3 \end{bmatrix}^T$.
The configuration model is characterised by

$$S(q) = \begin{bmatrix} -sin\vartheta & 0 \\ cos\vartheta & 0 \\ 0 & 1 \\ \frac{1}{d}cos\beta_{c3} & -\frac{1}{d}(d + Lsin\beta_{c3}) \\ -1/r & -L/r \\ 1/r & -L/r \\ -\frac{1}{r}sin\beta_{c3} & -\frac{L}{r}cos\beta_{c3} \end{bmatrix}.$$

It can be checked that $dim(\Delta_1) = 2$ and $dim(inv(\Delta_1)) = 6$. From the the configuration model it is

$$\dot{\varphi}_1 + \dot{\varphi}_2 = -\frac{2L}{r}\dot{\vartheta}.$$



Modeling of the mobile robots | Configuration dynamic model

Model derivation

Using the Lagrange formulation, the dynamics of wheeled mobile robots is described by the following $(3 + N_c + N + N_s)$ Lagrange's equations:

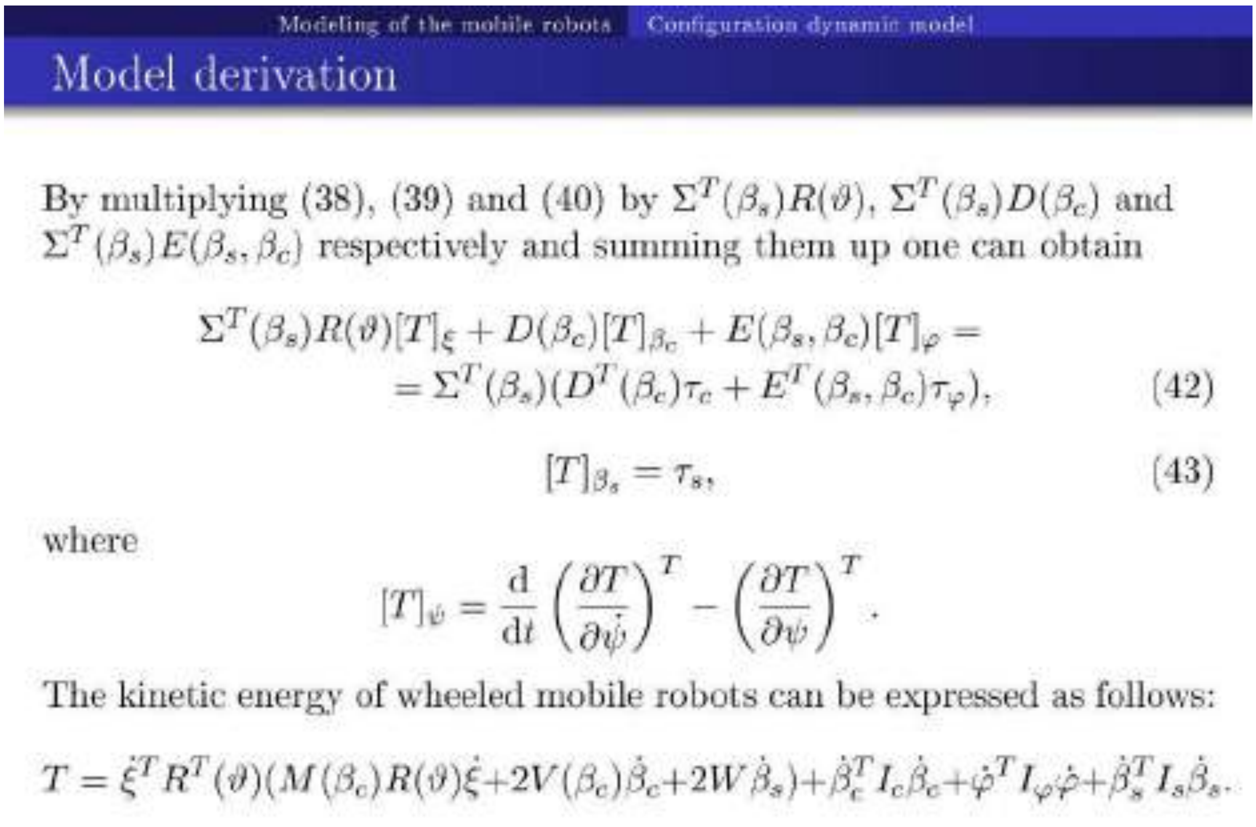
$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial T}{\partial \dot\xi}\right)^T - \left(\frac{\partial T}{\partial \xi}\right)^T = R^T(\vartheta)J_1^T(\beta_s,\beta_c)\lambda + R^T(\vartheta)C_1^T(\beta_s,\beta_c)\mu, \quad (38)$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial T}{\partial \dot\beta_c}\right)^T - \left(\frac{\partial T}{\partial \beta_c}\right)^T = C_2^T\mu + \tau_c, \quad (39)$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial T}{\partial \dot\varphi}\right)^T - \left(\frac{\partial T}{\partial \varphi}\right)^T = J_2^T\lambda + \tau_\varphi, \quad (40)$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial T}{\partial \dot\beta_s}\right)^T - \left(\frac{\partial T}{\partial \beta_s}\right)^T = \tau_s, \quad (41)$$

where T represents the kinetic energy and λ, μ are the Lagrange multipliers associated with the constraints (8) and (9) respectively.



Modeling of the mobile robots | Configuration dynamic model

Model derivation

By multiplying (38), (39) and (40) by $\Sigma^T(\beta_s)R(\vartheta)$, $\Sigma^T(\beta_s)D(\beta_c)$ and $\Sigma^T(\beta_s)E(\beta_s,\beta_c)$ respectively and summing them up one can obtain

$$\begin{aligned}\Sigma^T(\beta_s)R(\vartheta)[T]_\xi + D(\beta_c)[T]_{\beta_c} + E(\beta_s,\beta_c)[T]_\varphi = \\ = \Sigma^T(\beta_s)(D^T(\beta_c)\tau_c + E^T(\beta_s,\beta_c)\tau_\varphi),\end{aligned} \quad (42)$$

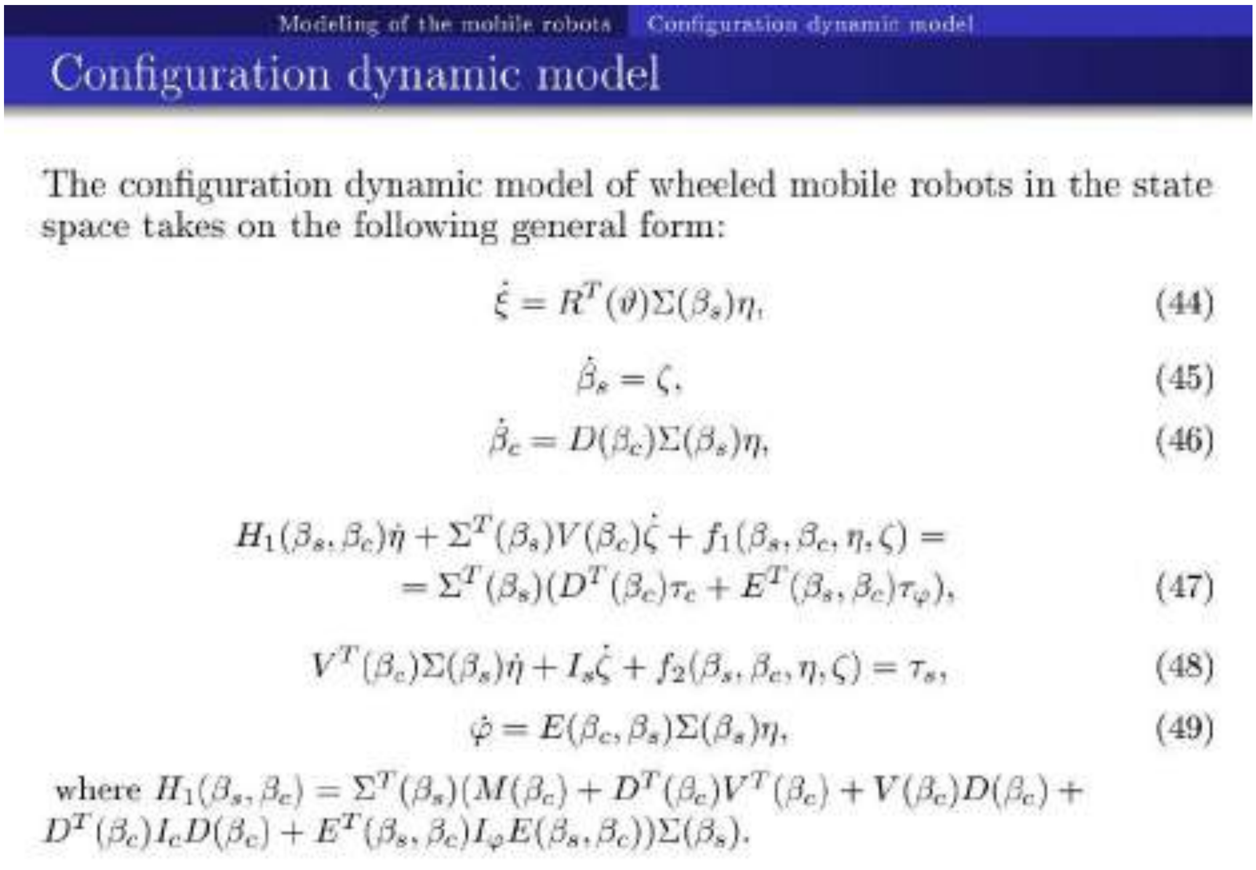
$$[T]_{\beta_s} = \tau_s, \quad (43)$$

where

$$[T]_\psi = \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial T}{\partial \dot\psi}\right)^T - \left(\frac{\partial T}{\partial \psi}\right)^T.$$

The kinetic energy of wheeled mobile robots can be expressed as follows:

$$T = \dot\xi^T R^T(\vartheta)(M(\beta_c)R(\vartheta)\dot\xi + 2V(\beta_c)\dot\beta_c + 2W\dot\beta_s) + \dot\beta_c^T I_c\dot\beta_c + \dot\varphi^T I_\varphi\dot\varphi + \dot\beta_s^T I_s\dot\beta_s.$$



Configuration dynamic model

The configuration dynamic model of wheeled mobile robots in the state space takes on the following general form:

$$\dot{\xi} = R^T(\vartheta)\Sigma(\beta_s)\eta, \tag{44}$$

$$\dot{\beta}_s = \zeta, \tag{45}$$

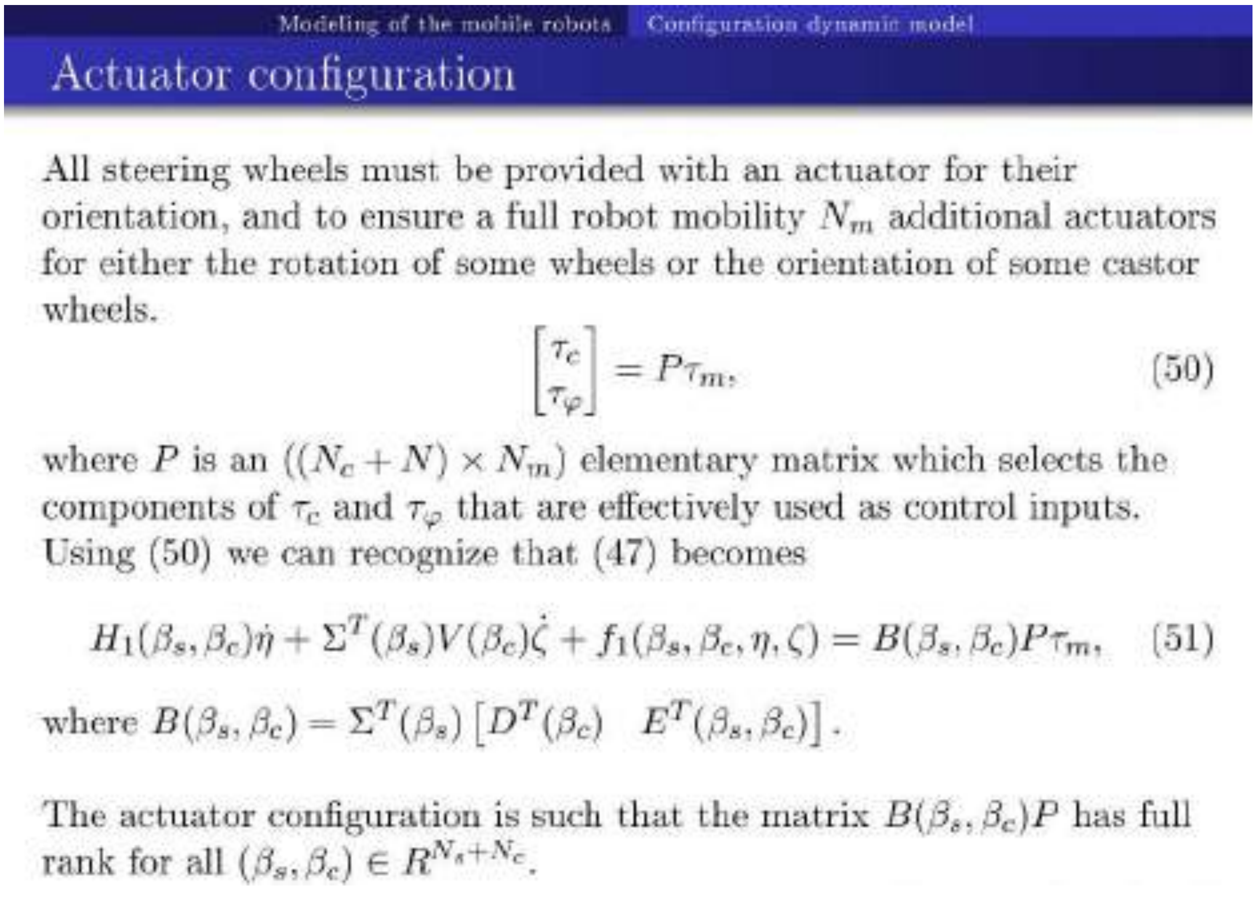
$$\dot{\beta}_c = D(\beta_c)\Sigma(\beta_s)\eta, \tag{46}$$

$$\begin{aligned} H_1(\beta_s, \beta_c)\dot{\eta} + \Sigma^T(\beta_s)V(\beta_c)\dot{\zeta} + f_1(\beta_s, \beta_c, \eta, \zeta) = \\ = \Sigma^T(\beta_s)(D^T(\beta_c)\tau_c + E^T(\beta_s, \beta_c)\tau_\varphi), \end{aligned} \tag{47}$$

$$V^T(\beta_c)\Sigma(\beta_s)\dot{\eta} + I_s\dot{\zeta} + f_2(\beta_s, \beta_c, \eta, \zeta) = \tau_s, \tag{48}$$

$$\dot{\varphi} = E(\beta_c, \beta_s)\Sigma(\beta_s)\eta, \tag{49}$$

where $H_1(\beta_s, \beta_c) = \Sigma^T(\beta_s)(M(\beta_c) + D^T(\beta_c)V^T(\beta_c) + V(\beta_c)D(\beta_c) + D^T(\beta_c)I_cD(\beta_c) + E^T(\beta_s, \beta_c)I_\varphi E(\beta_s, \beta_c))\Sigma(\beta_s)$.



Actuator configuration

All steering wheels must be provided with an actuator for their orientation, and to ensure a full robot mobility N_m additional actuators for either the rotation of some wheels or the orientation of some castor wheels.

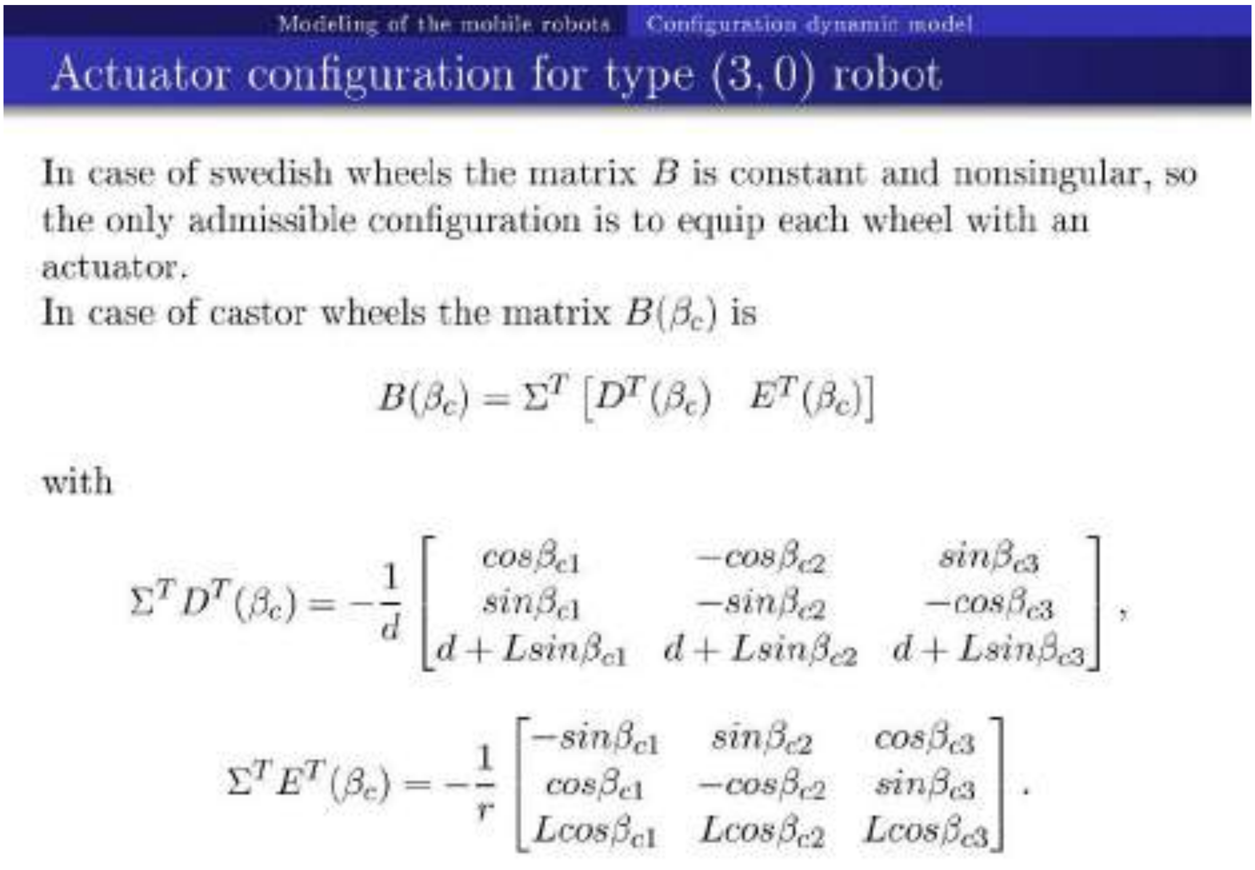
$$\begin{bmatrix} \tau_c \\ \tau_\varphi \end{bmatrix} = P\tau_m, \tag{50}$$

where P is an $((N_c + N) \times N_m)$ elementary matrix which selects the components of τ_c and τ_φ that are effectively used as control inputs. Using (50) we can recognize that (47) becomes

$$H_1(\beta_s, \beta_c)\dot{\eta} + \Sigma^T(\beta_s)V(\beta_c)\dot{\zeta} + f_1(\beta_s, \beta_c, \eta, \zeta) = B(\beta_s, \beta_c)P\tau_m, \tag{51}$$

where $B(\beta_s, \beta_c) = \Sigma^T(\beta_s)\begin{bmatrix} D^T(\beta_c) & E^T(\beta_s, \beta_c) \end{bmatrix}$.

The actuator configuration is such that the matrix $B(\beta_s, \beta_c)P$ has full rank for all $(\beta_s, \beta_c) \in R^{N_s+N_c}$.


Modeling of the mobile robots | Configuration dynamic model

Actuator configuration for type (3, 0) robot

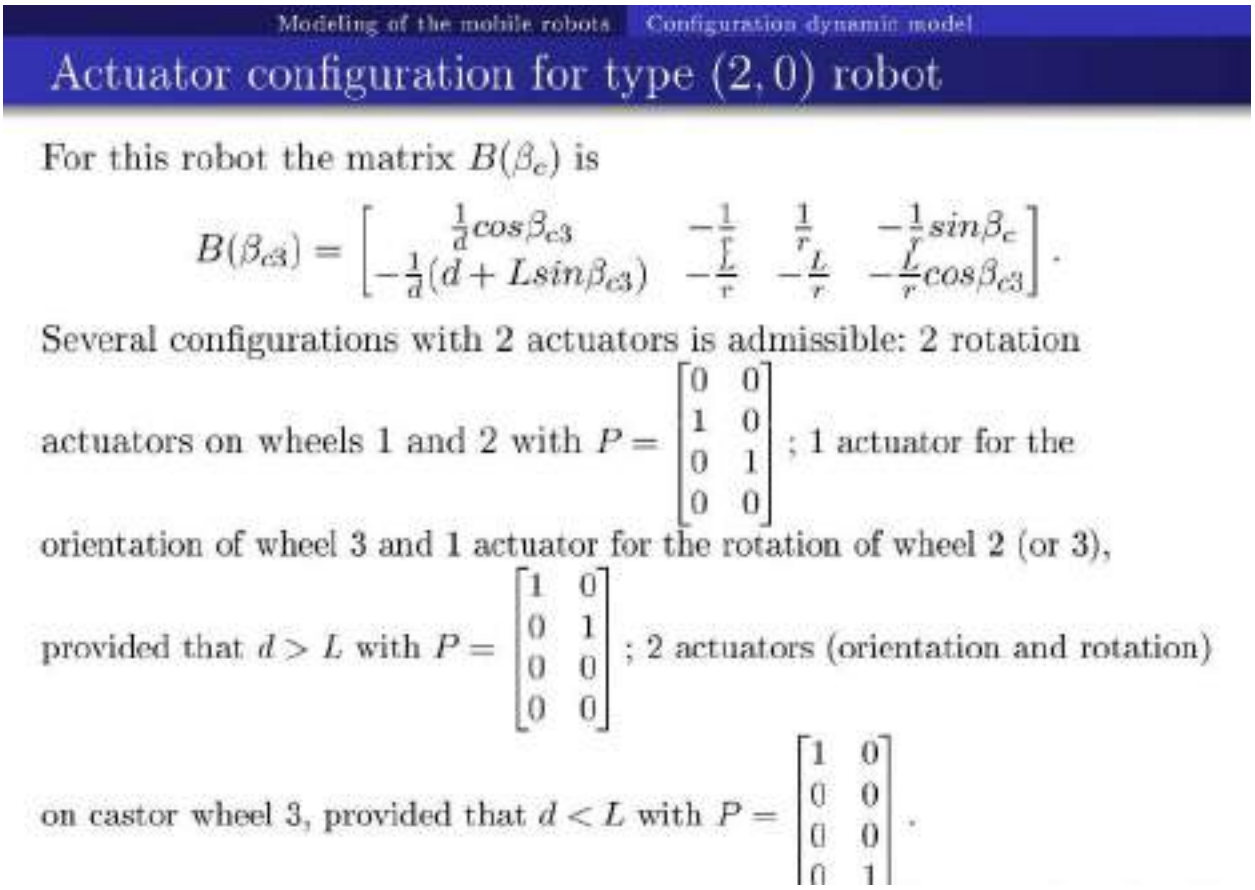
In case of swedish wheels the matrix B is constant and nonsingular, so the only admissible configuration is to equip each wheel with an actuator.
In case of castor wheels the matrix $B(\beta_c)$ is

$$B(\beta_c) = \Sigma^T \begin{bmatrix} D^T(\beta_c) & E^T(\beta_c) \end{bmatrix}$$

with

$$\Sigma^T D^T(\beta_c) = -\frac{1}{d}\begin{bmatrix} cos\beta_{c1} & -cos\beta_{c2} & sin\beta_{c3} \\ sin\beta_{c1} & -sin\beta_{c2} & -cos\beta_{c3} \\ d+Lsin\beta_{c1} & d+Lsin\beta_{c2} & d+Lsin\beta_{c3} \end{bmatrix},$$

$$\Sigma^T E^T(\beta_c) = -\frac{1}{r}\begin{bmatrix} -sin\beta_{c1} & sin\beta_{c2} & cos\beta_{c3} \\ cos\beta_{c1} & -cos\beta_{c2} & sin\beta_{c3} \\ Lcos\beta_{c1} & Lcos\beta_{c2} & Lcos\beta_{c3} \end{bmatrix}.$$

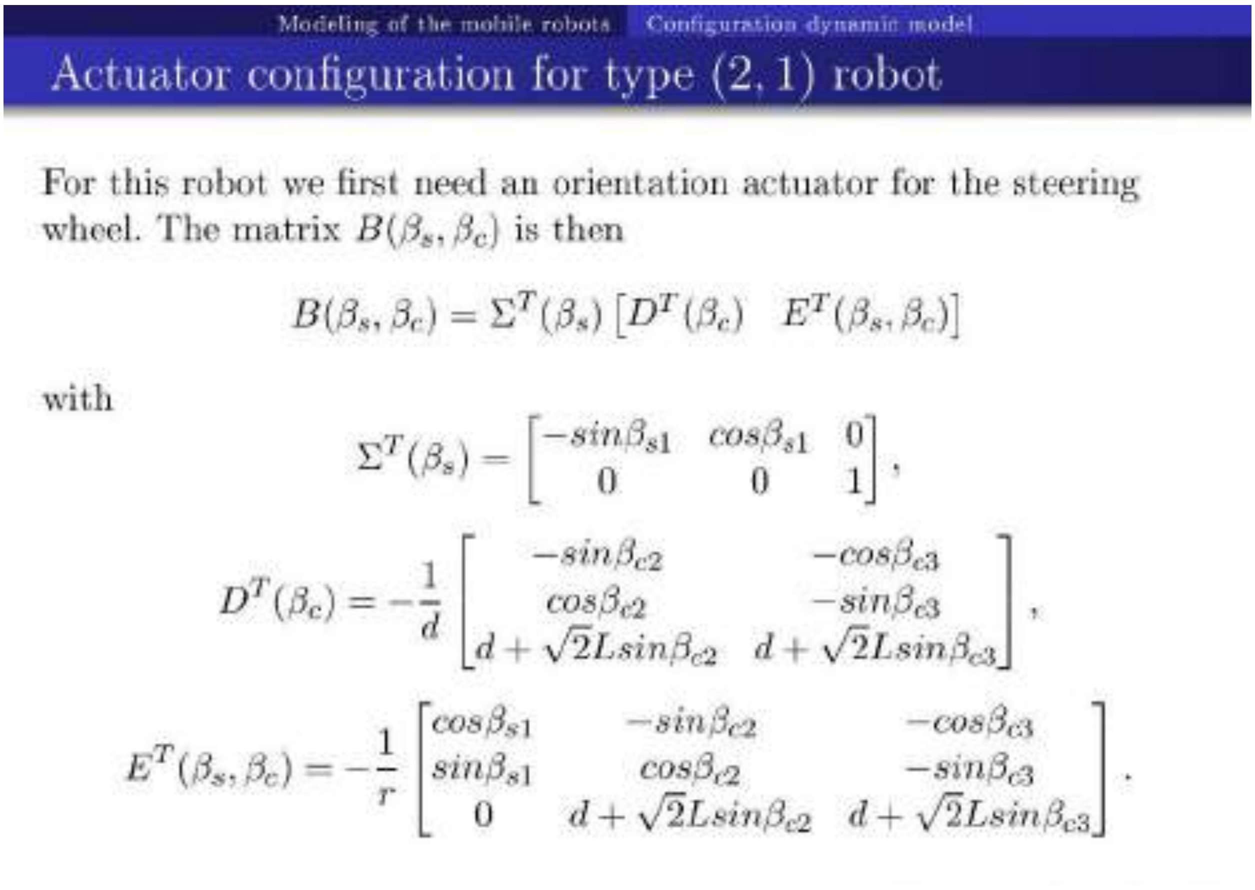

Modeling of the mobile robots | Configuration dynamic model

Actuator configuration for type (2, 0) robot

For this robot the matrix $B(\beta_c)$ is

$$B(\beta_{c3}) = \begin{bmatrix} \frac{1}{d}cos\beta_{c3} & -\frac{1}{r} & \frac{1}{r} & -\frac{1}{r}sin\beta_c \\ -\frac{1}{d}(d+Lsin\beta_{c3}) & -\frac{L}{r} & -\frac{L}{r} & -\frac{L}{r}cos\beta_{c3} \end{bmatrix}.$$

Several configurations with 2 actuators is admissible: 2 rotation actuators on wheels 1 and 2 with $P = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}$; 1 actuator for the orientation of wheel 3 and 1 actuator for the rotation of wheel 2 (or 3), provided that $d > L$ with $P = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$; 2 actuators (orientation and rotation) on castor wheel 3, provided that $d < L$ with $P = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}$.



Actuator configuration for type (2, 1) robot

For this robot we first need an orientation actuator for the steering wheel. The matrix $B(\beta_s, \beta_c)$ is then

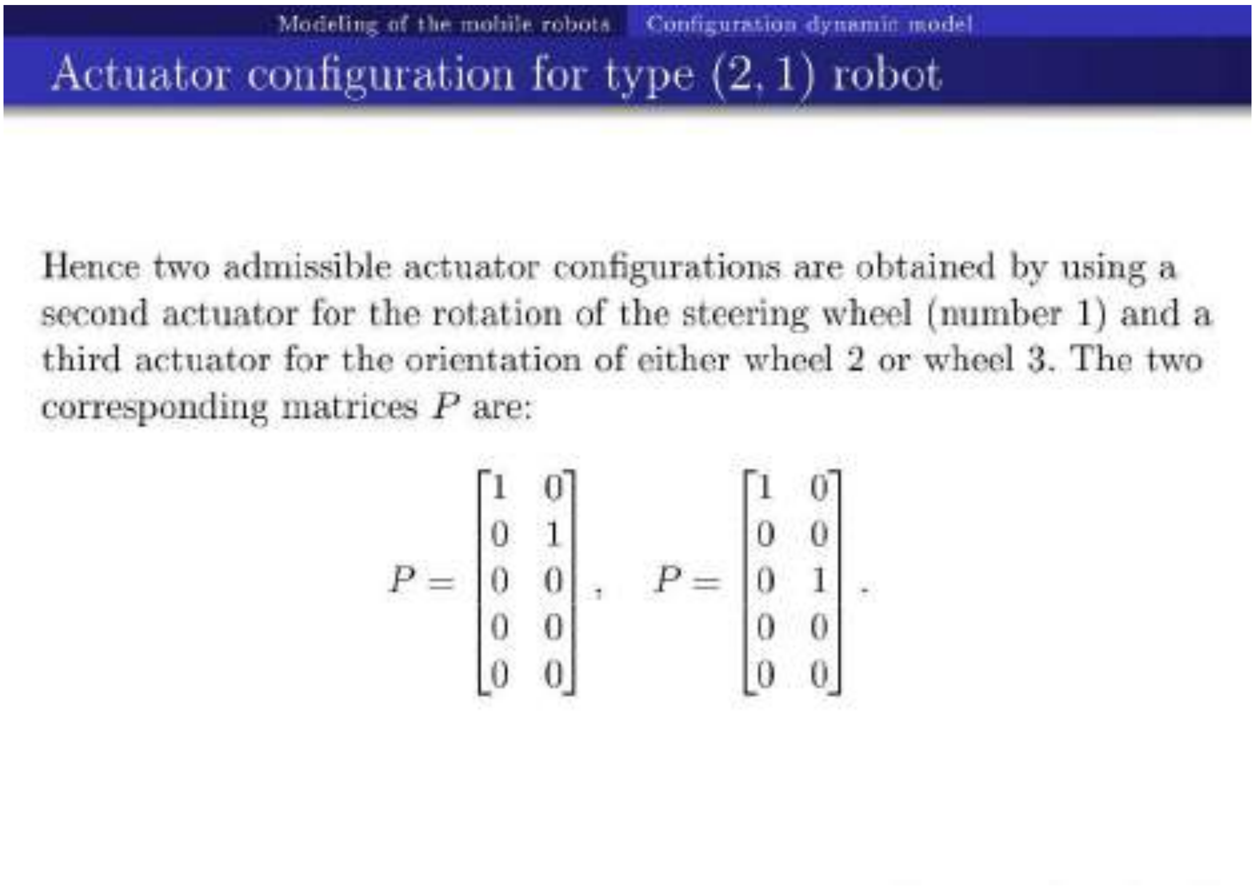
$$B(\beta_s, \beta_c) = \Sigma^T(\beta_s)\begin{bmatrix} D^T(\beta_c) & E^T(\beta_s, \beta_c) \end{bmatrix}$$

with

$$\Sigma^T(\beta_s) = \begin{bmatrix} -sin\beta_{s1} & cos\beta_{s1} & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

$$D^T(\beta_c) = -\frac{1}{d}\begin{bmatrix} -sin\beta_{c2} & -cos\beta_{c3} \\ cos\beta_{c2} & -sin\beta_{c3} \\ d+\sqrt{2}Lsin\beta_{c2} & d+\sqrt{2}Lsin\beta_{c3} \end{bmatrix},$$

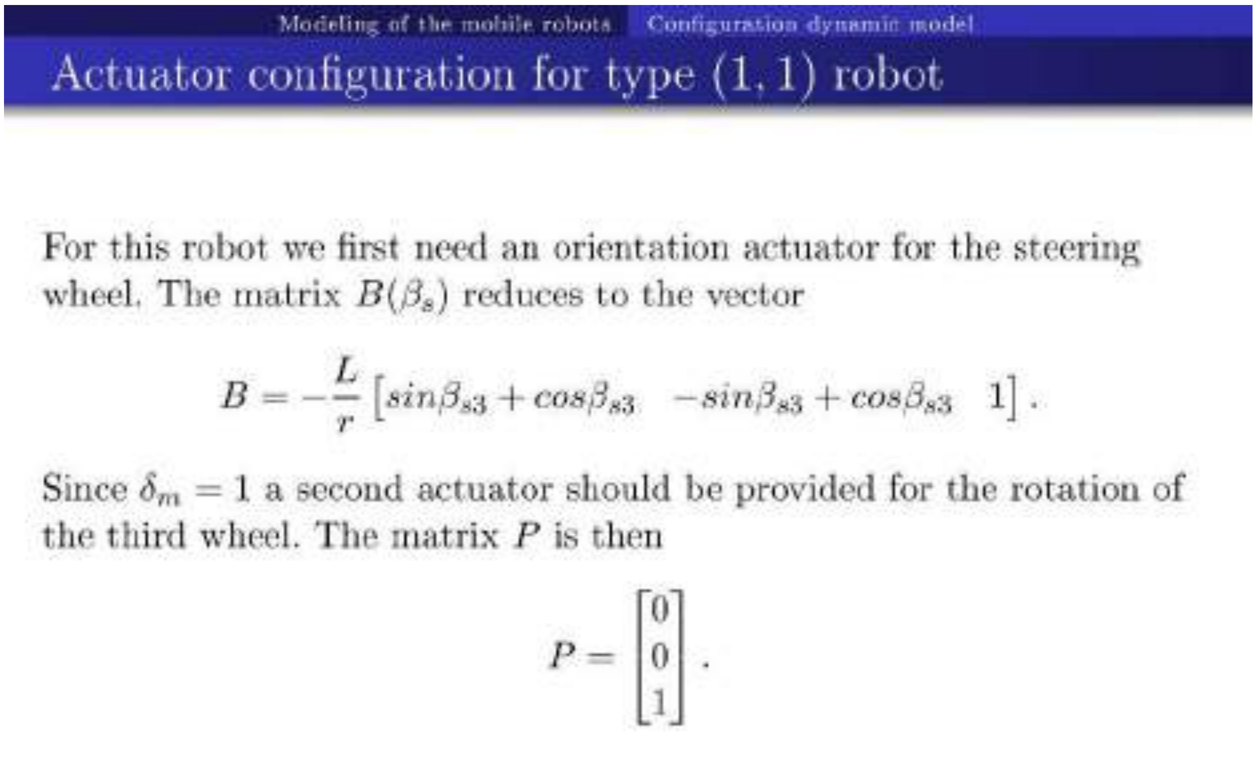
$$E^T(\beta_s, \beta_c) = -\frac{1}{r}\begin{bmatrix} cos\beta_{s1} & -sin\beta_{c2} & -cos\beta_{c3} \\ sin\beta_{s1} & cos\beta_{c2} & -sin\beta_{c3} \\ 0 & d+\sqrt{2}Lsin\beta_{c2} & d+\sqrt{2}Lsin\beta_{c3} \end{bmatrix}.$$



Actuator configuration for type (2, 1) robot

Hence two admissible actuator configurations are obtained by using a second actuator for the rotation of the steering wheel (number 1) and a third actuator for the orientation of either wheel 2 or wheel 3. The two corresponding matrices P are:

$$P = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad P = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}.$$



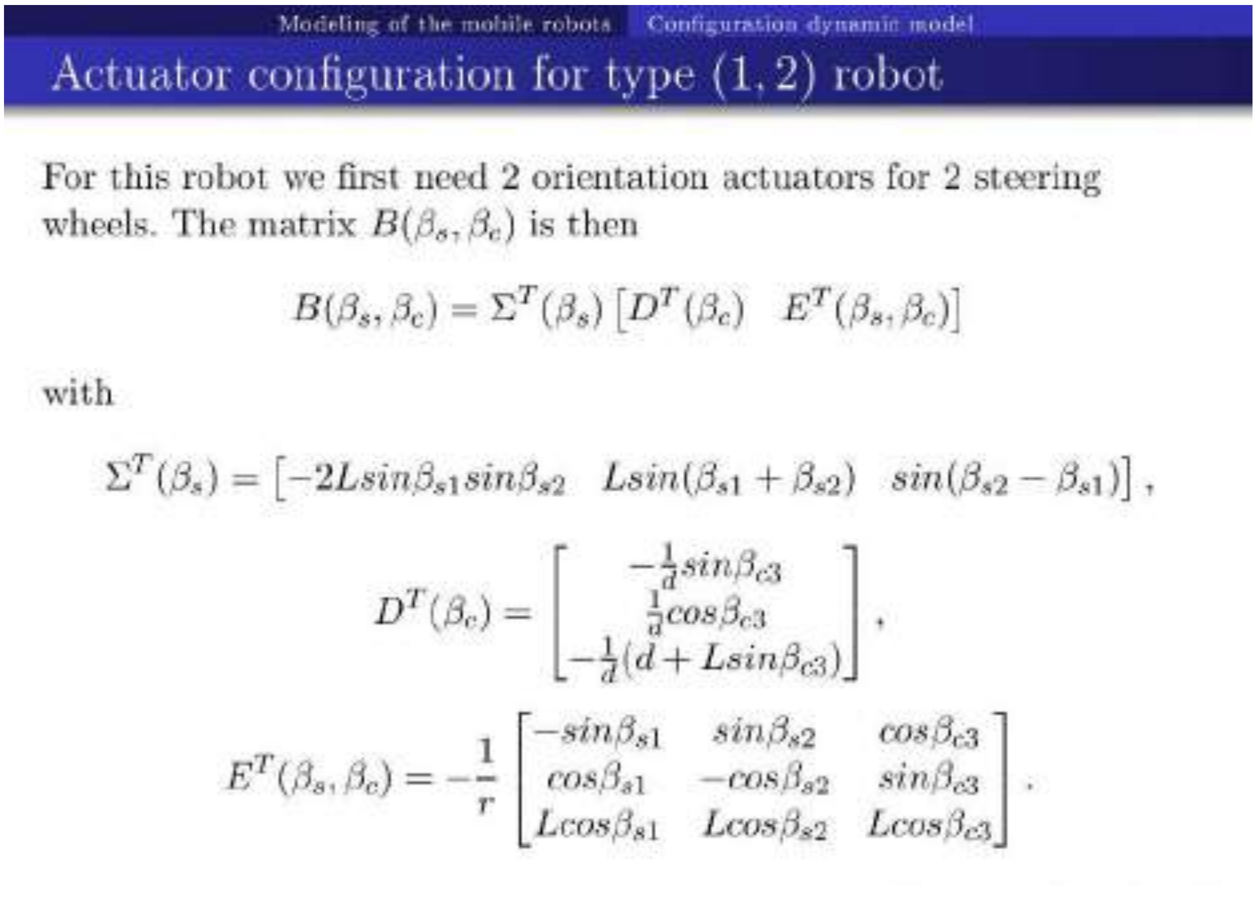
Actuator configuration for type (1, 1) robot

For this robot we first need an orientation actuator for the steering wheel. The matrix $B(\beta_s)$ reduces to the vector

$$B = -\frac{L}{r}\begin{bmatrix} sin\beta_{s3} + cos\beta_{s3} & -sin\beta_{s3} + cos\beta_{s3} & 1 \end{bmatrix}.$$

Since $\delta_m = 1$ a second actuator should be provided for the rotation of the third wheel. The matrix P is then

$$P = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.$$



Actuator configuration for type (1, 2) robot

For this robot we first need 2 orientation actuators for 2 steering wheels. The matrix $B(\beta_s, \beta_c)$ is then

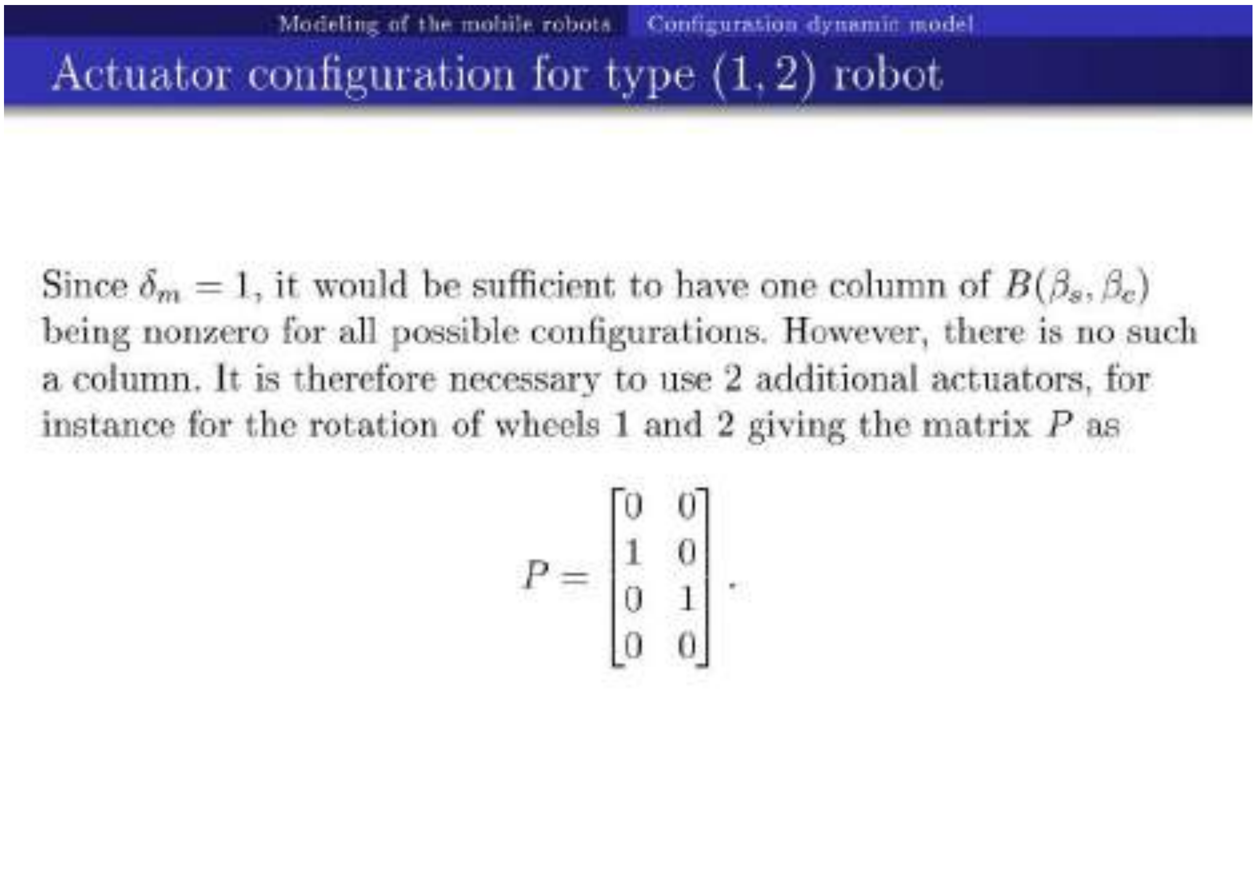
$$B(\beta_s, \beta_c) = \Sigma^T(\beta_s)\begin{bmatrix} D^T(\beta_c) & E^T(\beta_s, \beta_c) \end{bmatrix}$$

with

$$\Sigma^T(\beta_s) = \begin{bmatrix} -2Lsin\beta_{s1}sin\beta_{s2} & Lsin(\beta_{s1} + \beta_{s2}) & sin(\beta_{s2} - \beta_{s1}) \end{bmatrix},$$

$$D^T(\beta_c) = \begin{bmatrix} -\frac{1}{d}sin\beta_{c3} \\ \frac{1}{d}cos\beta_{c3} \\ -\frac{1}{d}(d + Lsin\beta_{c3}) \end{bmatrix},$$

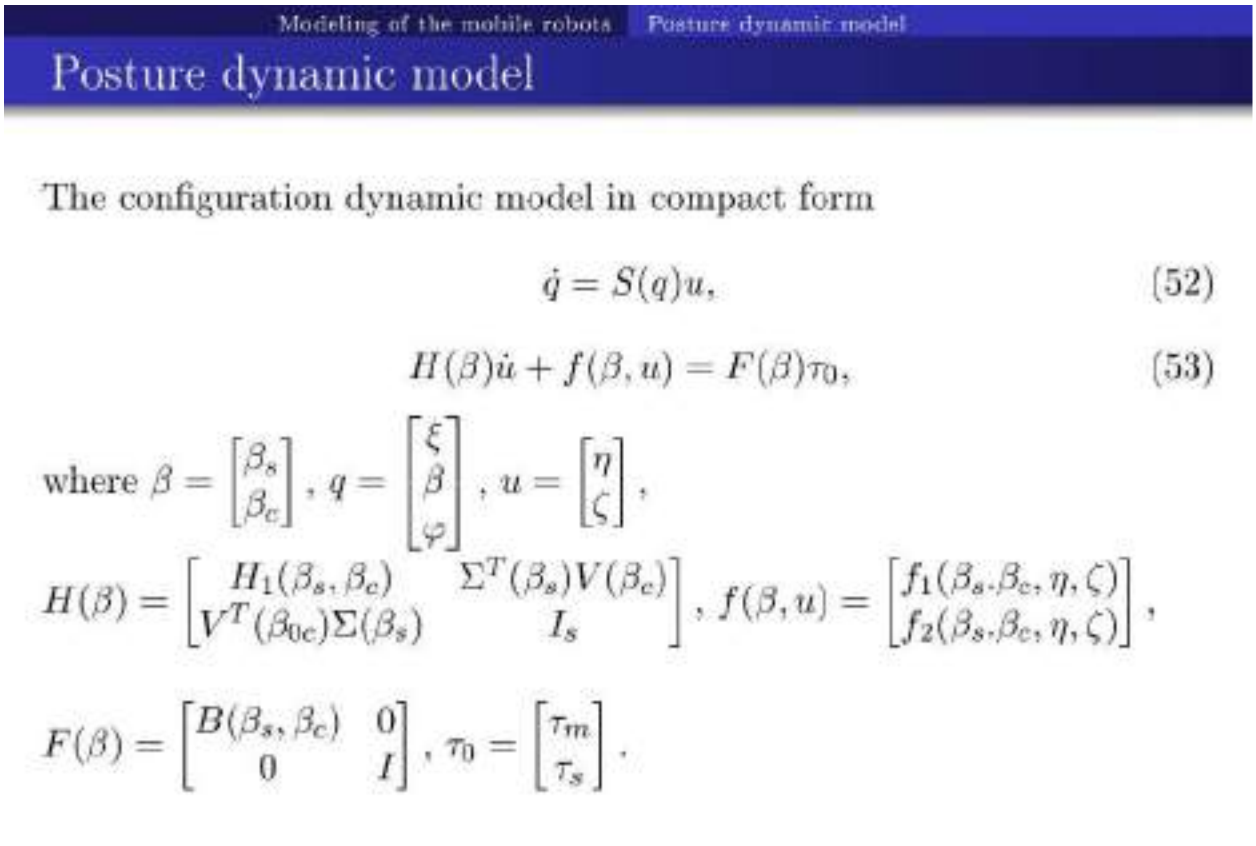
$$E^T(\beta_s, \beta_c) = -\frac{1}{r}\begin{bmatrix} -sin\beta_{s1} & sin\beta_{s2} & cos\beta_{c3} \\ cos\beta_{s1} & -cos\beta_{s2} & sin\beta_{c3} \\ Lcos\beta_{s1} & Lcos\beta_{s2} & Lcos\beta_{c3} \end{bmatrix}.$$



Actuator configuration for type (1, 2) robot

Since $\delta_m = 1$, it would be sufficient to have one column of $B(\beta_s, \beta_c)$ being nonzero for all possible configurations. However, there is no such a column. It is therefore necessary to use 2 additional actuators, for instance for the rotation of wheels 1 and 2 giving the matrix P as

$$P = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}.$$



Posture dynamic model

The configuration dynamic model in compact form

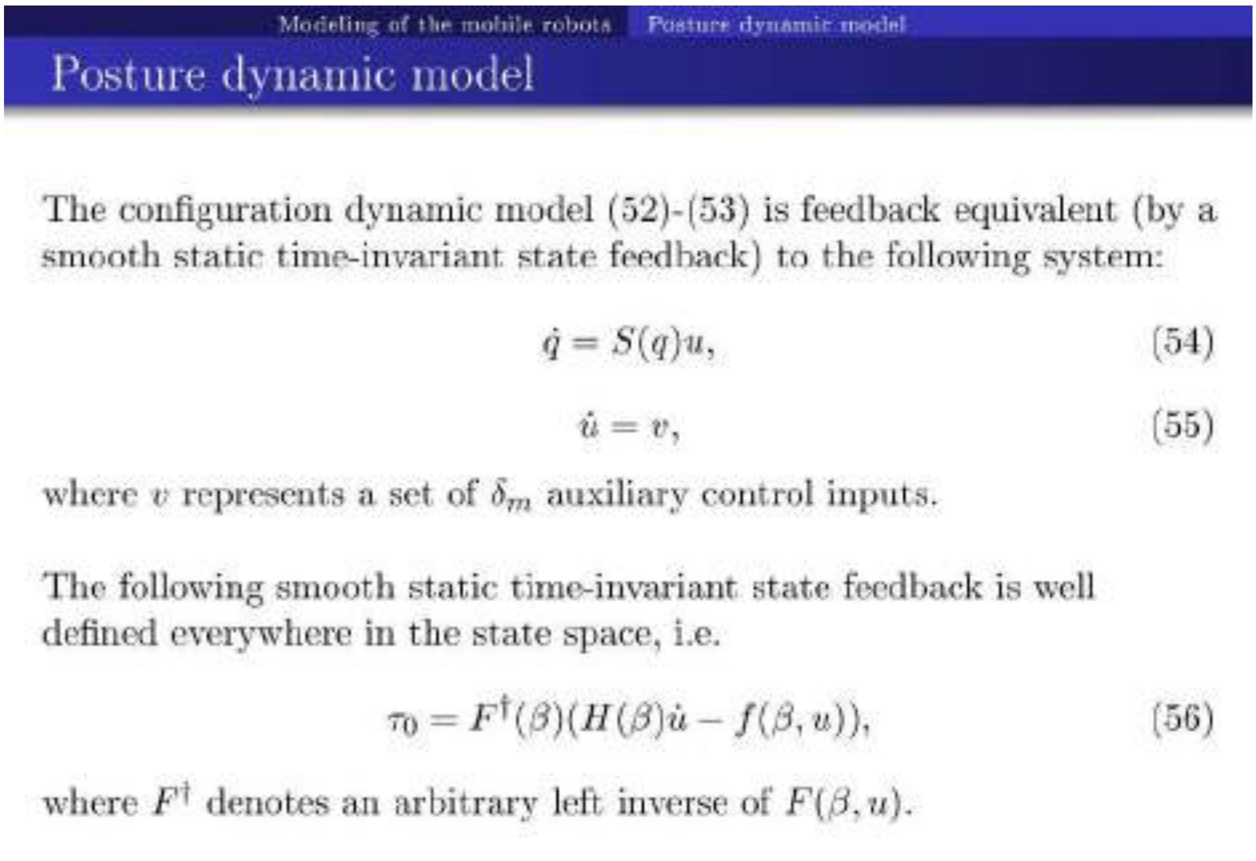
$$\dot{q} = S(q)u, \tag{52}$$

$$H(\beta)\dot{u} + f(\beta, u) = F(\beta)\tau_0, \tag{53}$$

where $\beta = \begin{bmatrix} \beta_s \\ \beta_c \end{bmatrix}$, $q = \begin{bmatrix} \xi \\ \beta \\ \varphi \end{bmatrix}$, $u = \begin{bmatrix} \eta \\ \zeta \end{bmatrix}$,

$H(\beta) = \begin{bmatrix} H_1(\beta_s, \beta_c) & \Sigma^T(\beta_s)V(\beta_c) \\ V^T(\beta_{0c})\Sigma(\beta_s) & I_s \end{bmatrix}$, $f(\beta, u) = \begin{bmatrix} f_1(\beta_s, \beta_c, \eta, \zeta) \\ f_2(\beta_s, \beta_c, \eta, \zeta) \end{bmatrix}$,

$F(\beta) = \begin{bmatrix} B(\beta_s, \beta_c) & 0 \\ 0 & I \end{bmatrix}$, $\tau_0 = \begin{bmatrix} \tau_m \\ \tau_s \end{bmatrix}$.



Posture dynamic model

The configuration dynamic model (52)-(53) is feedback equivalent (by a smooth static time-invariant state feedback) to the following system:

$$\dot{q} = S(q)u, \tag{54}$$

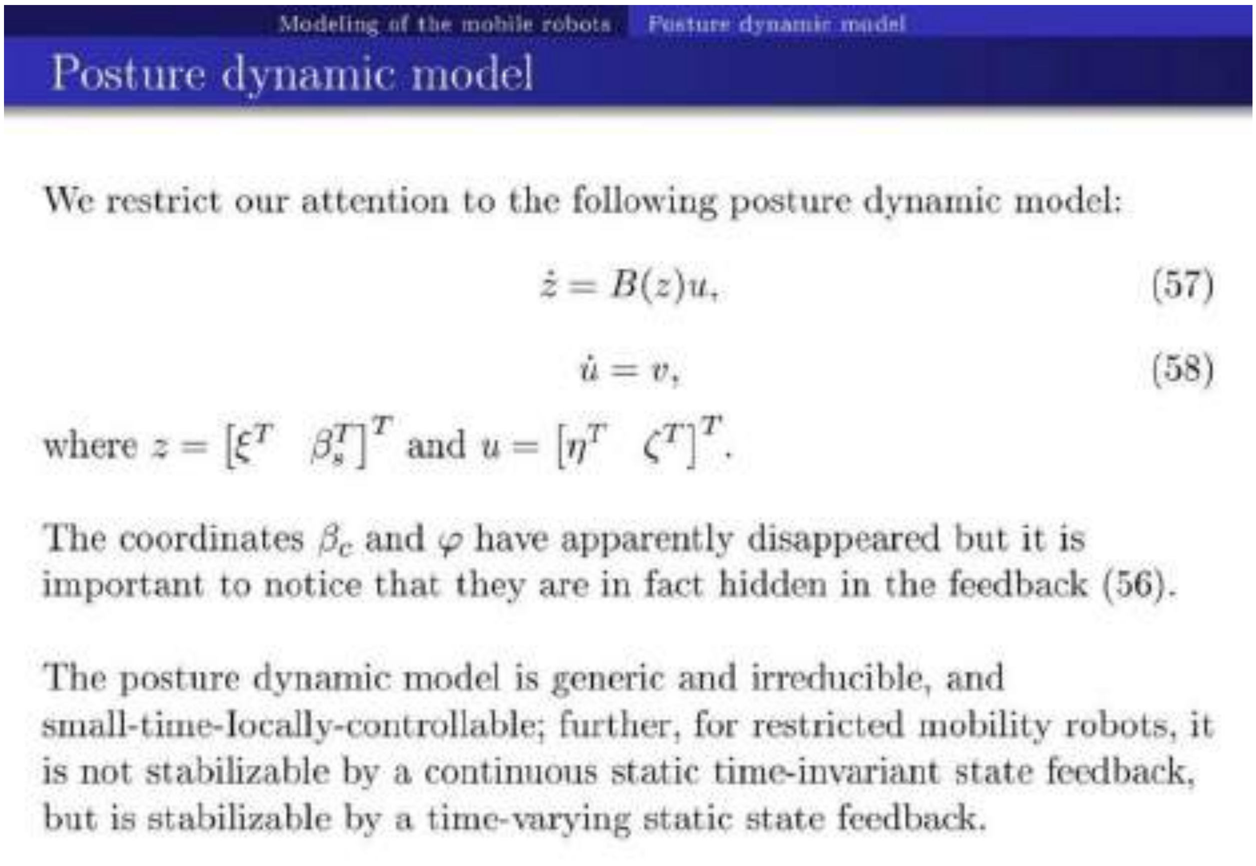
$$\dot{u} = v, \tag{55}$$

where v represents a set of δ_m auxiliary control inputs.

The following smooth static time-invariant state feedback is well defined everywhere in the state space, i.e.

$$\tau_0 = F^{\dagger}(\beta)(H(\beta)\dot{u} - f(\beta, u)), \tag{56}$$

where $F^{\dagger}$ denotes an arbitrary left inverse of $F(\beta, u)$.



Posture dynamic model

We restrict our attention to the following posture dynamic model:

$$\dot{z} = B(z)u, \tag{57}$$

$$\dot{u} = v, \tag{58}$$

where $z = \begin{bmatrix}\xi^T & \beta_s^T\end{bmatrix}^T$ and $u = \begin{bmatrix}\eta^T & \zeta^T\end{bmatrix}^T$.

The coordinates β_c and φ have apparently disappeared but it is important to notice that they are in fact hidden in the feedback (56).

The posture dynamic model is generic and irreducible, and small-time-locally-controllable; further, for restricted mobility robots, it is not stabilizable by a continuous static time-invariant state feedback, but is stabilizable by a time-varying static state feedback.

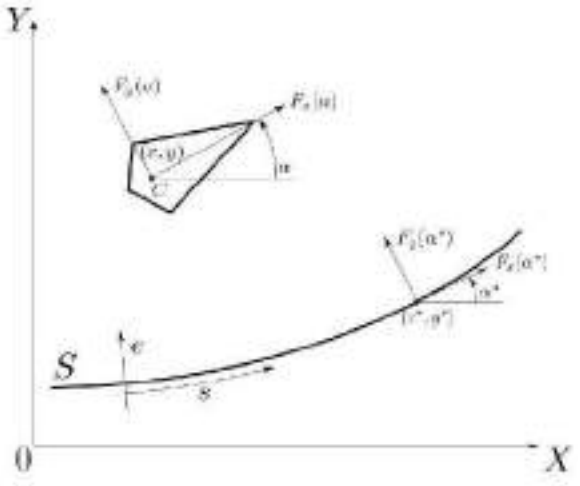
Motion on the plane

Dynamic model of robot motion:

$$\dot{v}_x = v_y\omega + \frac{1}{m}F_x, \qquad (59)$$

$$\dot{v}_y = -v_x\omega + \frac{1}{m}F_y, \qquad (60)$$

$$\dot{\omega} = \frac{1}{J}M_c, \qquad (61)$$



Motion on the plane

Relation of linear velocities in the fixed and absolute frames:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = T^T(\alpha)\begin{bmatrix} v_x \\ v_y \end{bmatrix}, \qquad (62)$$

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

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix} = \frac{1}{m}T^T(\alpha)\begin{bmatrix} F_x \\ F_y \end{bmatrix}. \qquad (63)$$

Motion on the plane

The desired path is an implicitly described smooth segment of curve S:

$$\varphi(x, y) = 0, \tag{64}$$

and relevant local coordinate s (path length) is defined as

$$s = \psi(x, y) \tag{65}$$

Selection of functions (91) and (92) is mostly limited by regularity condition implying that Jacobian matrix

$$\Upsilon(x, y) = \begin{bmatrix} \frac{\partial\psi}{\partial x} & \frac{\partial\psi}{\partial y} \\ \frac{\partial\varphi}{\partial x} & \frac{\partial\varphi}{\partial y} \end{bmatrix} \tag{66}$$

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

$$\dot{T^*}(\alpha^*) = \dot{s}\xi(s)ET^*(\alpha^*), \tag{67}$$

where $E = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$ and $\xi(s)$ is the path curvative.

From (67) also follows

$$\dot{\alpha^*} = \dot{s}\xi(s). \tag{68}$$

Angular orientation

Robot angular orientation with respect to curve S is defined as

$$\alpha = \alpha^*(s) + \Delta\alpha, \tag{69}$$

where $\Delta\alpha = const$ is the desired robot orientation with respest to the path. In matrix notation, (69) takes the form

$$T(\alpha) = T(\Delta\alpha)T(\alpha^*). \tag{70}$$

Introducing errors and problem statement

Violation of condition (91) is characterised by orthogonal deviation

$$e = \varphi(x, y). \tag{71}$$

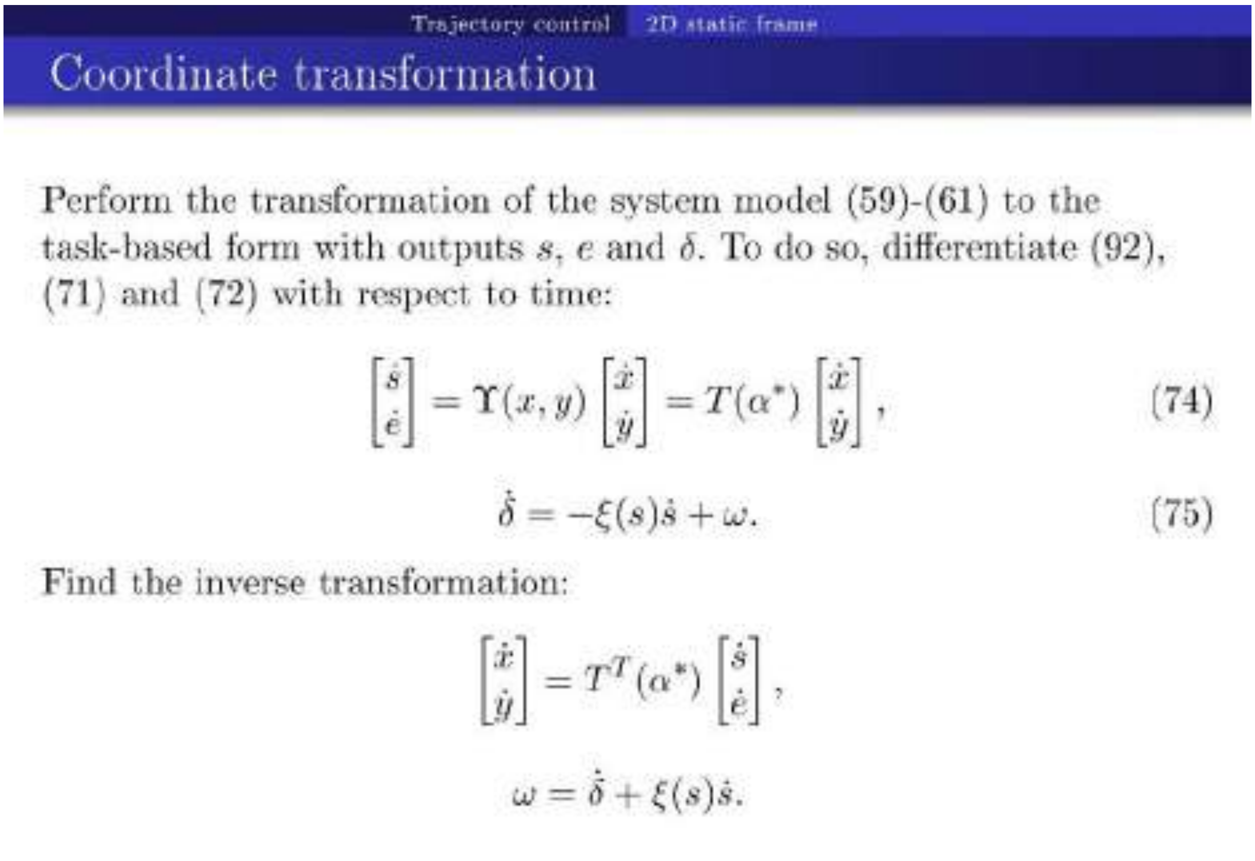
Violation of condition (68) is characterised by angular deviation

$$\delta = \alpha - \alpha^* + \Delta\alpha. \tag{72}$$

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

$$\Delta V_s = V_s^* - \dot{s}. \tag{73}$$



Trajectory control | 2D static frame

Coordinate transformation

Perform the transformation of the system model (59)-(61) to the task-based form with outputs s, e and δ. To do so, differentiate (92), (71) and (72) with respect to time:

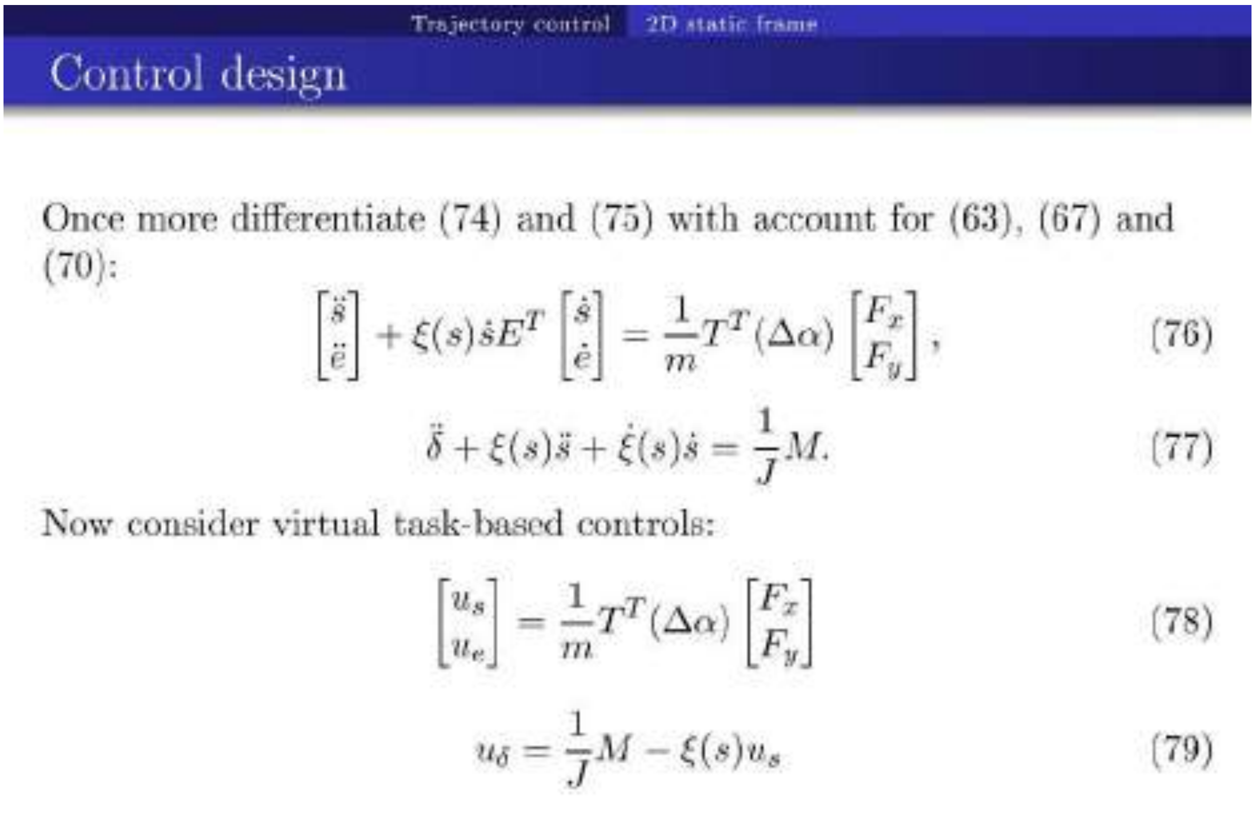
$$\begin{bmatrix}\dot s\\ \dot e\end{bmatrix} = \Upsilon(x,y)\begin{bmatrix}\dot x\\ \dot y\end{bmatrix} = T(\alpha^*)\begin{bmatrix}\dot x\\ \dot y\end{bmatrix}, \qquad (74)$$

$$\dot\delta = -\xi(s)\dot s + \omega. \qquad (75)$$

Find the inverse transformation:

$$\begin{bmatrix}\dot x\\ \dot y\end{bmatrix} = T^T(\alpha^*)\begin{bmatrix}\dot s\\ \dot e\end{bmatrix},$$

$$\omega = \dot\delta + \xi(s)\dot s.$$



Trajectory control | 2D static frame

Control design

Once more differentiate (74) and (75) with account for (63), (67) and (70):

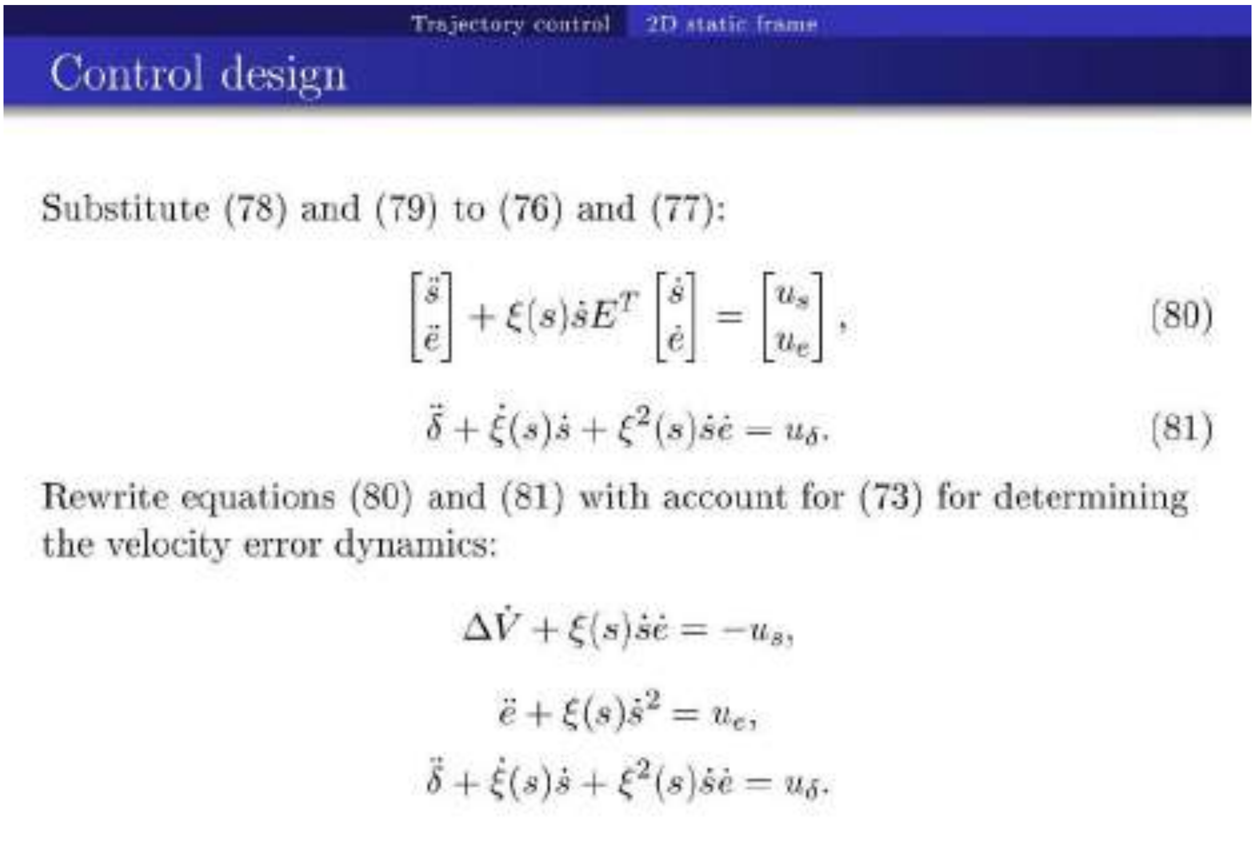
$$\begin{bmatrix}\ddot s\\ \ddot e\end{bmatrix} + \xi(s)\dot s E^T\begin{bmatrix}\dot s\\ \dot e\end{bmatrix} = \frac{1}{m}T^T(\Delta\alpha)\begin{bmatrix}F_x\\ F_y\end{bmatrix}, \qquad (76)$$

$$\ddot\delta + \xi(s)\ddot s + \dot\xi(s)\dot s = \frac{1}{J}M. \qquad (77)$$

Now consider virtual task-based controls:

$$\begin{bmatrix}u_s\\ u_e\end{bmatrix} = \frac{1}{m}T^T(\Delta\alpha)\begin{bmatrix}F_x\\ F_y\end{bmatrix} \qquad (78)$$

$$u_\delta = \frac{1}{J}M - \xi(s)u_s \qquad (79)$$



Control design

Substitute (78) and (79) to (76) and (77):

$$\begin{bmatrix} \ddot{s} \\ \ddot{e} \end{bmatrix} + \xi(s)\dot{s}E^T \begin{bmatrix} \dot{s} \\ \dot{e} \end{bmatrix} = \begin{bmatrix} u_s \\ u_e \end{bmatrix}, \tag{80}$$

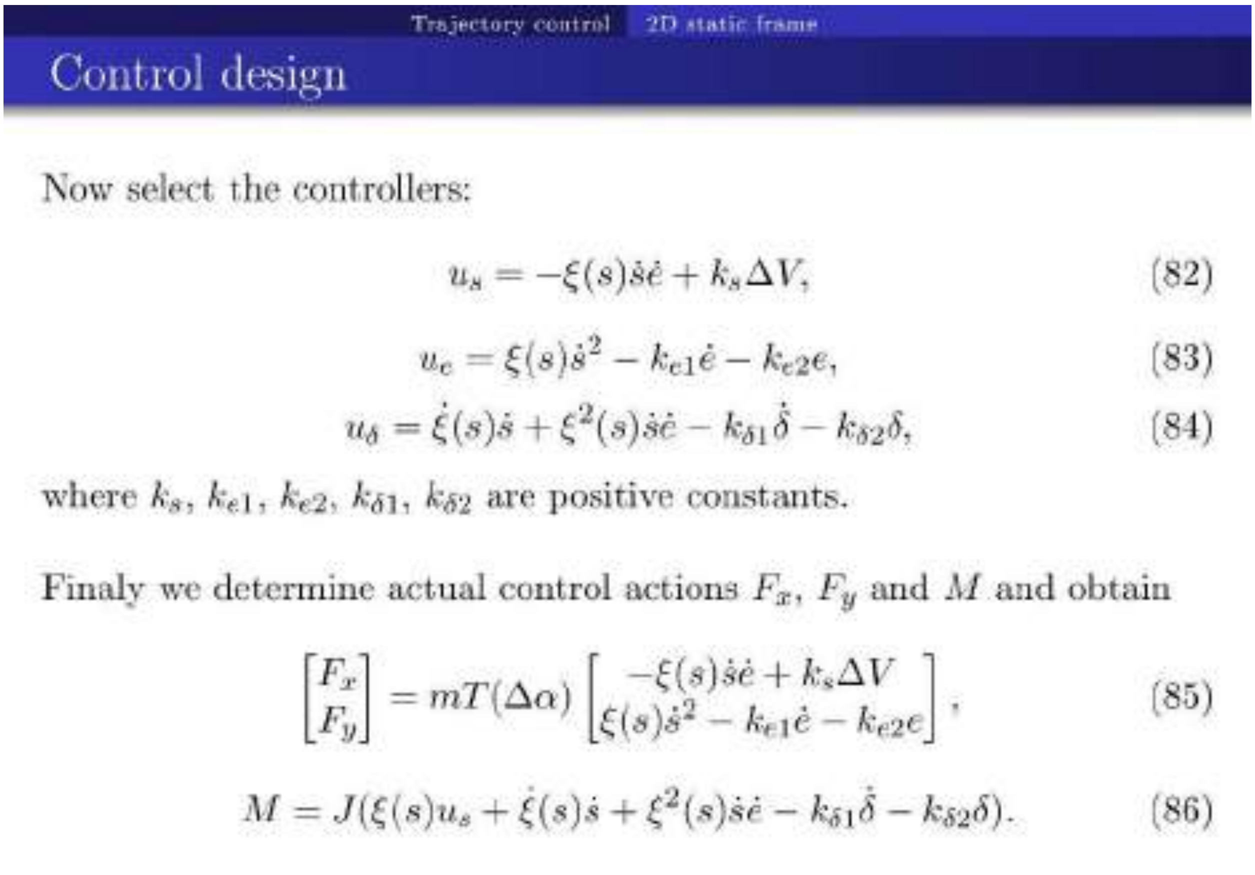
$$\ddot{\delta} + \dot{\xi}(s)\dot{s} + \xi^2(s)\dot{s}\dot{e} = u_\delta. \tag{81}$$

Rewrite equations (80) and (81) with account for (73) 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:

$$u_s = -\xi(s)\dot{s}\dot{e} + k_s\Delta V, \tag{82}$$

$$u_e = \xi(s)\dot{s}^2 - k_{e1}\dot{e} - k_{e2}e, \tag{83}$$

$$u_\delta = \dot{\xi}(s)\dot{s} + \xi^2(s)\dot{s}\dot{e} - k_{\delta 1}\dot{\delta} - k_{\delta 2}\delta, \tag{84}$$

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

$$\begin{bmatrix} F_x \\ F_y \end{bmatrix} = mT(\Delta\alpha) \begin{bmatrix} -\xi(s)\dot{s}\dot{e} + k_s\Delta V \\ \xi(s)\dot{s}^2 - k_{e1}\dot{e} - k_{e2}e \end{bmatrix}, \tag{85}$$

$$M = J(\xi(s)u_s + \dot{\xi}(s)\dot{s} + \xi^2(s)\dot{s}\dot{e} - k_{\delta 1}\dot{\delta} - k_{\delta 2}\delta). \tag{86}$$

Example. Straight line segment

The normalized equation of the straight line

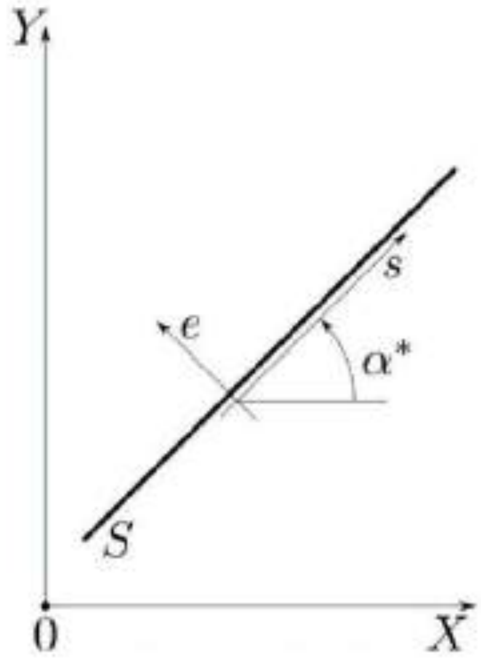
$$\varphi(q) = -\sin\alpha^* x + \cos\alpha^* y + \varphi_0 = 0,$$

$$\psi(q) = \cos\alpha^* x + \sin\alpha^* y + \psi_0,$$

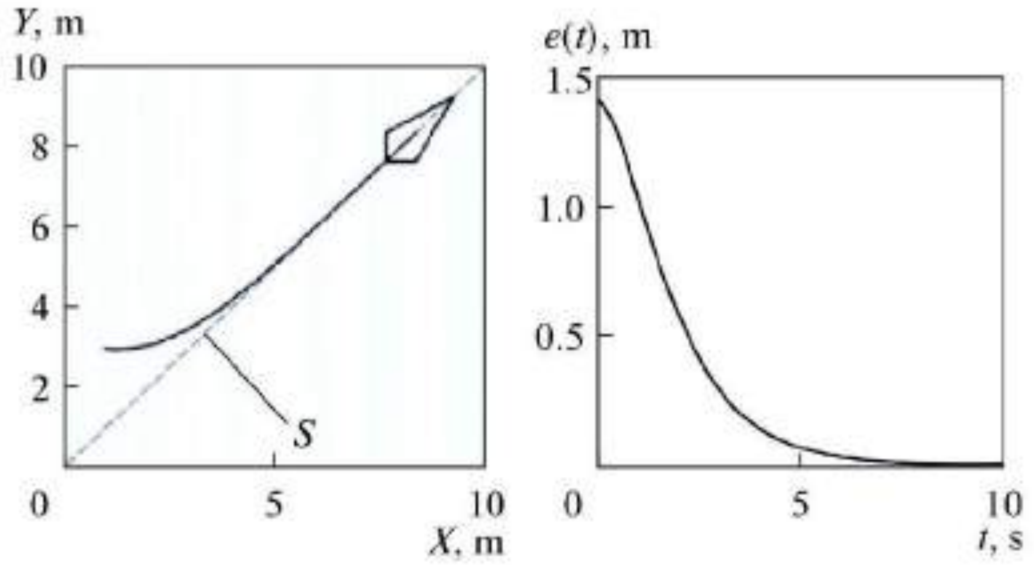
where α^* is the line inclination, $\varphi_0 = const$, $\psi_0 = const$.
Orthogonal Jacobian matrix takes the form

$$\Upsilon(q) = \begin{bmatrix} \cos\alpha^* & \sin\alpha^* \\ -\sin\alpha^* & \cos\alpha^* \end{bmatrix} \in SO(2).$$

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

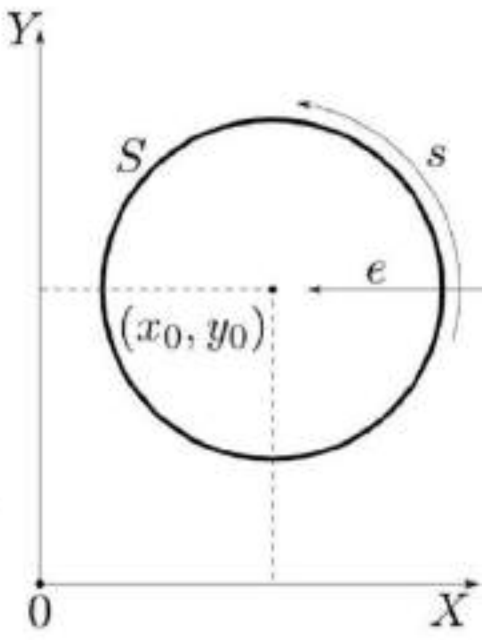
$$\varphi(q) = \frac{1}{2R}(R^2-(x-x_0)^2-(y-y_0)^2) = 0,$$

$$\psi(q) = R\arctan\frac{(y-y_0)}{(x-x_0)}.$$

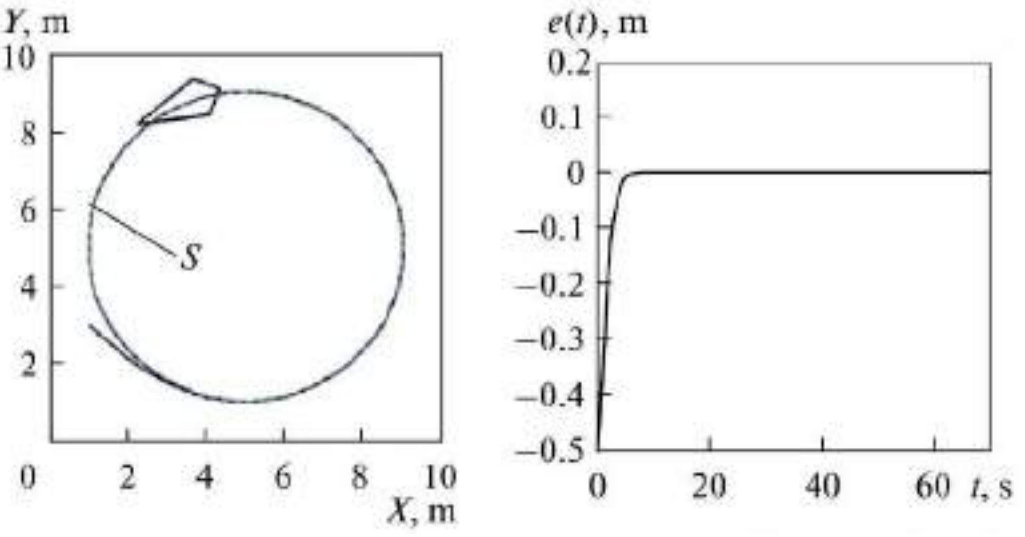
Orthogonal Jacobian matrix takes the form

$$\Upsilon(q) = \frac{1}{R}\begin{bmatrix} -(y-y_0) & (x-x_0) \\ -(x-x_0) & -(y-y_0) \end{bmatrix} \in SO(2).$$

The path curvature is $\xi(s) = \frac{1}{R}$



Example. Simulation of the motion along the arc of a circle.



Trajectory control | 2D static frame

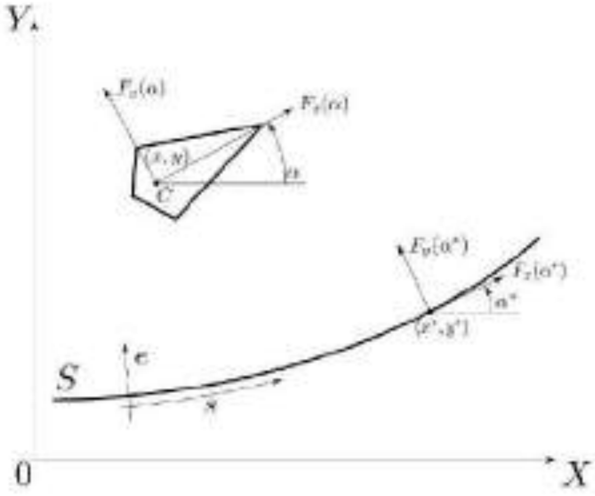
Dynamic model

$$m\ddot{q} = F, \tag{87}$$

$$\dot{q} = R_O^I(\alpha)v, \tag{88}$$

$$R_O^I(\alpha) = \begin{bmatrix} cos\alpha & sin\alpha \\ -sin\alpha & cos\alpha \end{bmatrix}, \tag{89}$$

$$\dot{\alpha} = \omega, \tag{90}$$



Trajectory control | 2D static frame

Motion on the plane

The desired path is an implicitly described smooth segment of curve S:

$$\varphi(q) = 0, \tag{91}$$

and relevant local coordinate s (path length) is defined as

$$s = \psi(q) \tag{92}$$

Selection of functions (91) and (92) is mostly limited by regularity condition implying that Jacobian matrix

$$\Upsilon(q) = \begin{bmatrix} \frac{\partial\psi(q)}{\partial x} & \frac{\partial\psi(q)}{\partial y} \\ \frac{\partial\varphi(q)}{\partial x} & \frac{\partial\varphi(q)}{\partial y} \end{bmatrix} \tag{93}$$

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:

$$V_1 = \frac{1}{2}(\dot{q} - \bar{v})^T(\dot{q} - \bar{v}), \tag{94}$$

where $\bar{v}$ - a vector of desired velocities.

Find the derivation of the Lyapunov function V_1:

$$\dot{V}_1 = (\dot{q} - \bar{v})^T(\ddot{q} - \dot{\bar{v}}) = (\dot{q} - \bar{v})^T(\frac{F}{m} - \dot{\bar{v}}). \tag{95}$$

Control design

Define the control signal as:

$$\frac{1}{m}F = \dot{\bar{v}} - k_q(\dot{q} - \bar{v}), \tag{96}$$

where k_q is a positive constant. Then the derivation of the Lyapunov function V_1 is

$$\dot{V}_1 = -k_q(\dot{q} - \bar{v})^T(\dot{q} - \bar{v}) \leq 0, \tag{97}$$

which means asymptotic stability of the point $\dot{q} - \bar{v} = 0$.

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 (87)-(90) to the task-based form with outputs s, e_1 and e_2, using Jacobian matrix (93):

$$\begin{bmatrix} \dot{s} \\ \dot{e}_1 \end{bmatrix} = \Upsilon(q)\dot{q} = \Upsilon(q)R_I^O(\alpha)v. \tag{98}$$

We can choose the control signal u_s in the form

$$u_s = R_O^I \Upsilon^{-1}(r) \begin{bmatrix} V^* \\ 0 \end{bmatrix} \tag{99}$$

Now design stabilization control u_e. Consider Lyapunov Function:

$$V_2 = \frac{k_e}{2}\varphi^2(q), \tag{100}$$

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

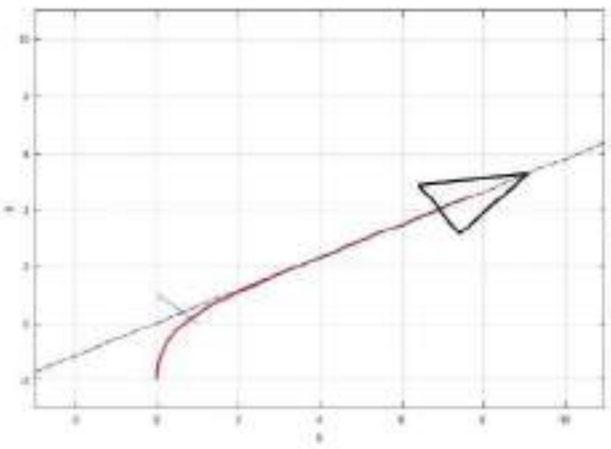
$$u_e = -k_e\varphi(q)\frac{\partial}{\partial q}\varphi(q), \tag{101}$$

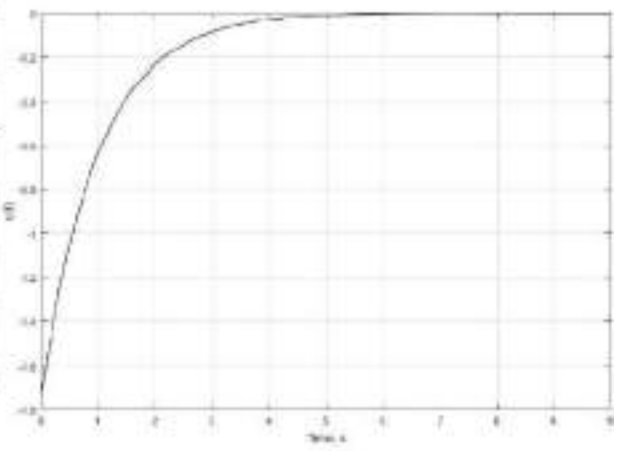
where k_e is positive constant.
Then the derivation of the Lyapunov function V_2 is

$$\dot{V}_2 = -u_e^2 \leq 0$$

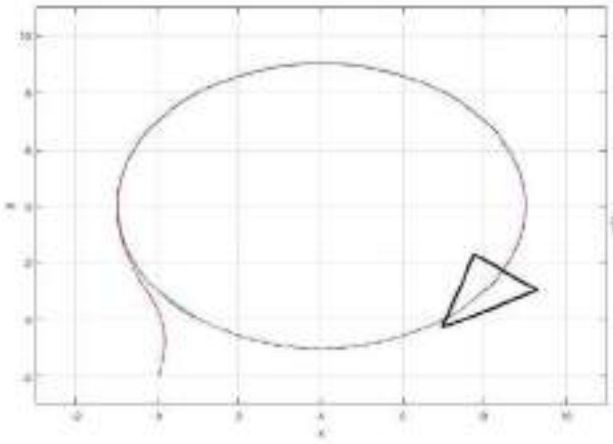
It proves the asymptotic stability of the initial system at the point $e(q) = 0$.

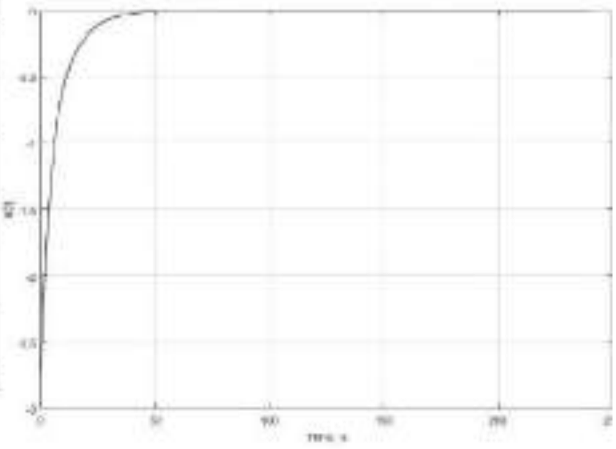
Example. Simulation of the motion along the straight line.





Example. Simulation of the motion along the arc of a circle.





Spatial motion

Dynamic model:

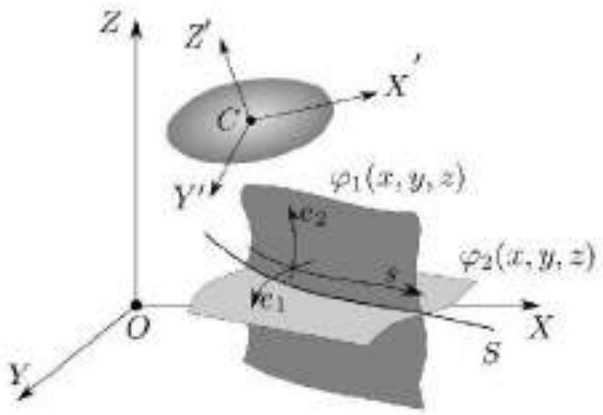
$$\dot{x} = v, \qquad (102)$$

$$\dot{v} = \frac{1}{m}F_c, \qquad (103)$$

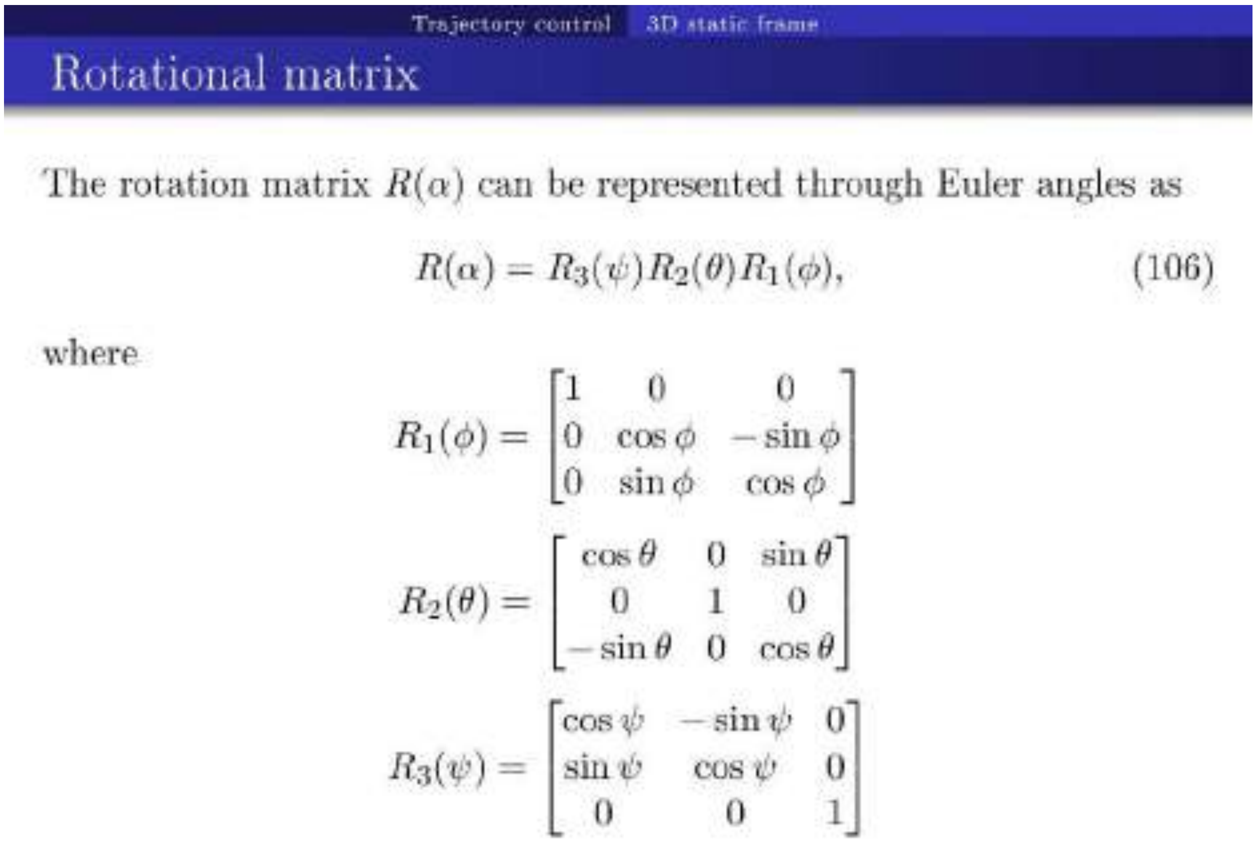
$$\dot{R}(\alpha) = S(\omega)R(\alpha), \qquad (104)$$

$$J\dot{\omega} + \omega \times J\omega = M_c, \qquad (105)$$

$$S(\omega) = \begin{bmatrix} 0 & \omega_3 & -\omega_2 \\ -\omega_3 & 0 & \omega_1 \\ \omega_2 & -\omega_1 & 0 \end{bmatrix}.$$



ITMO UNIVERSITY



Trajectory control | 3D static frame

Rotational matrix

The rotation matrix $R(\alpha)$ can be represented through Euler angles as

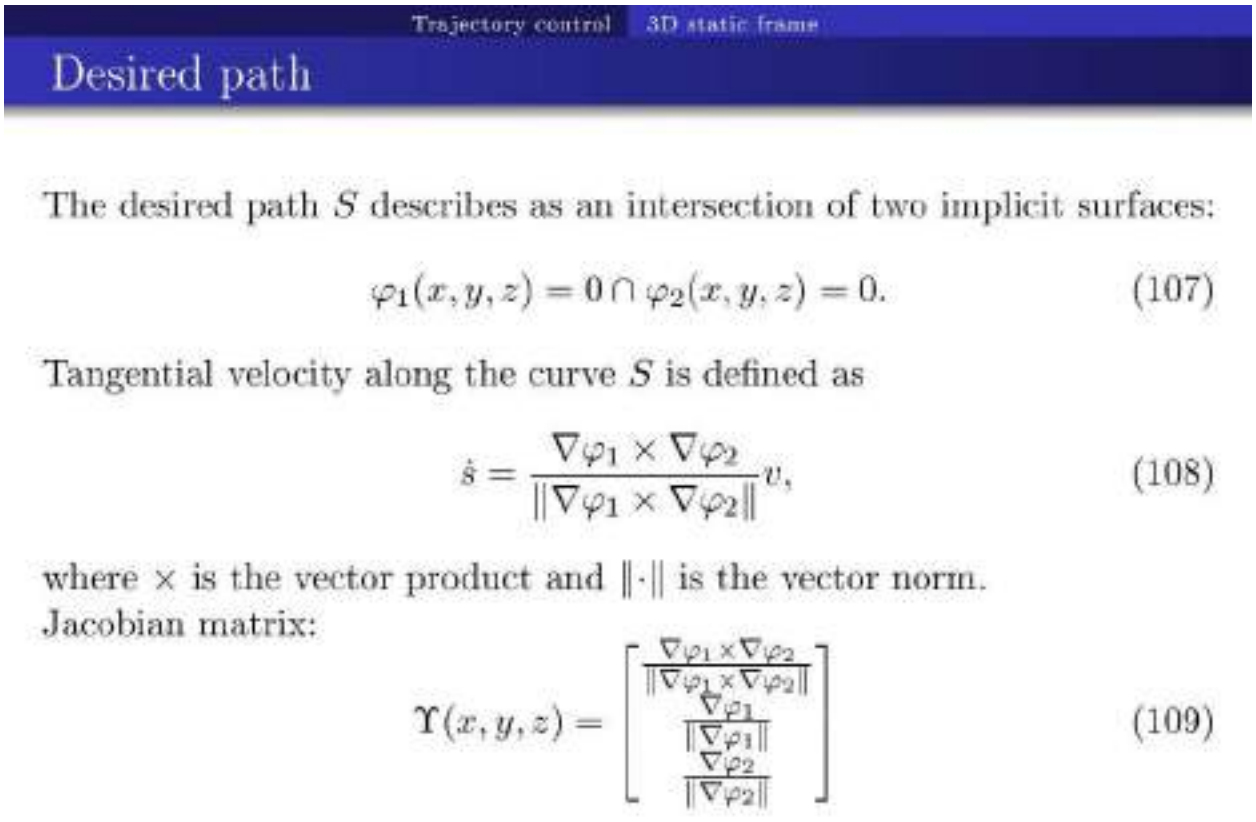
$$R(\alpha) = R_3(\psi)R_2(\theta)R_1(\phi), \tag{106}$$

where

$$R_1(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}$$

$$R_2(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

$$R_3(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



Trajectory control | 3D static frame

Desired path

The desired path S describes as an intersection of two implicit surfaces:

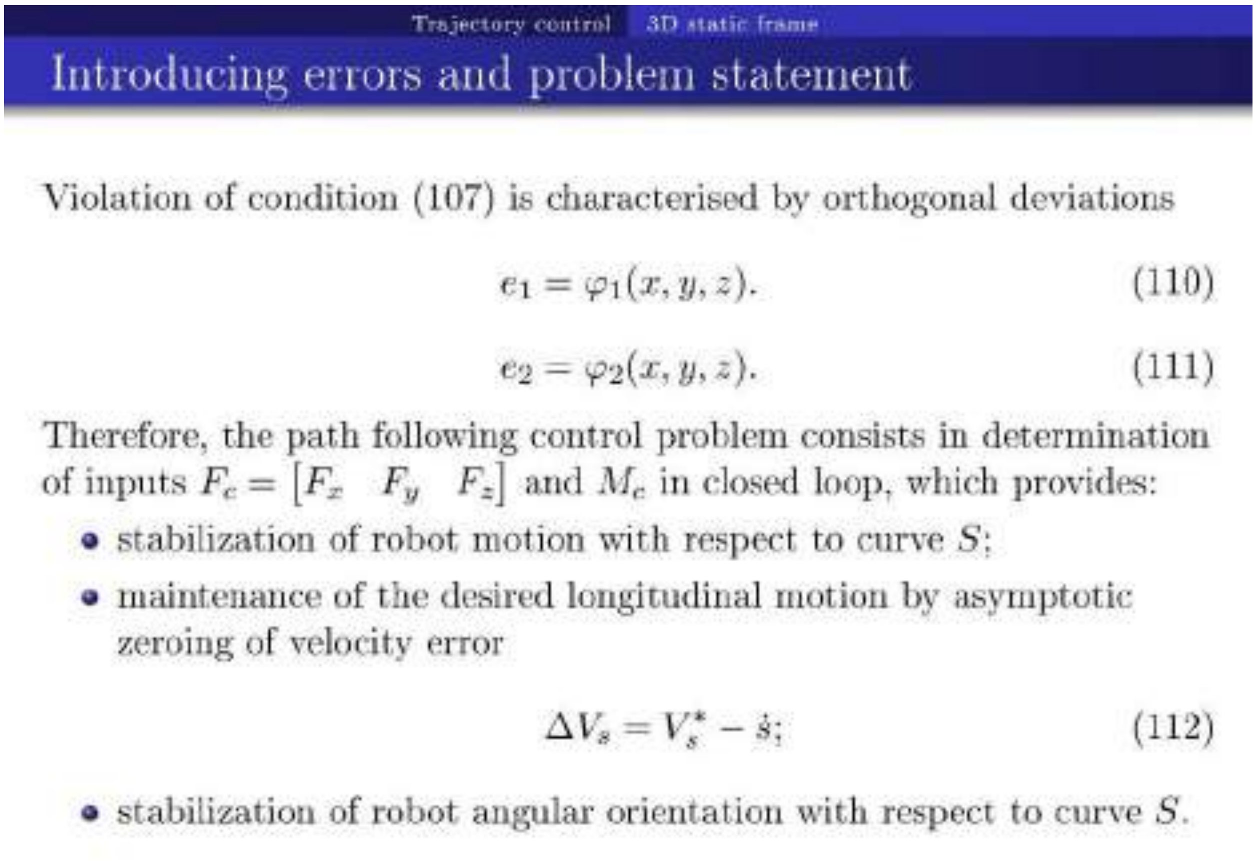
$$\varphi_1(x, y, z) = 0 \cap \varphi_2(x, y, z) = 0. \tag{107}$$

Tangential velocity along the curve S is defined as

$$\dot{s} = \frac{\nabla\varphi_1 \times \nabla\varphi_2}{\|\nabla\varphi_1 \times \nabla\varphi_2\|} v, \tag{108}$$

where $\times$ is the vector product and $\|\cdot\|$ is the vector norm.
Jacobian matrix:

$$\Upsilon(x, y, z) = \begin{bmatrix} \frac{\nabla\varphi_1 \times \nabla\varphi_2}{\|\nabla\varphi_1 \times \nabla\varphi_2\|} \\ \frac{\nabla\varphi_1}{\|\nabla\varphi_1\|} \\ \frac{\nabla\varphi_2}{\|\nabla\varphi_2\|} \end{bmatrix} \tag{109}$$



Trajectory control | 3D static frame

Introducing errors and problem statement

Violation of condition (107) is characterised by orthogonal deviations

$$e_1 = \varphi_1(x, y, z). \tag{110}$$

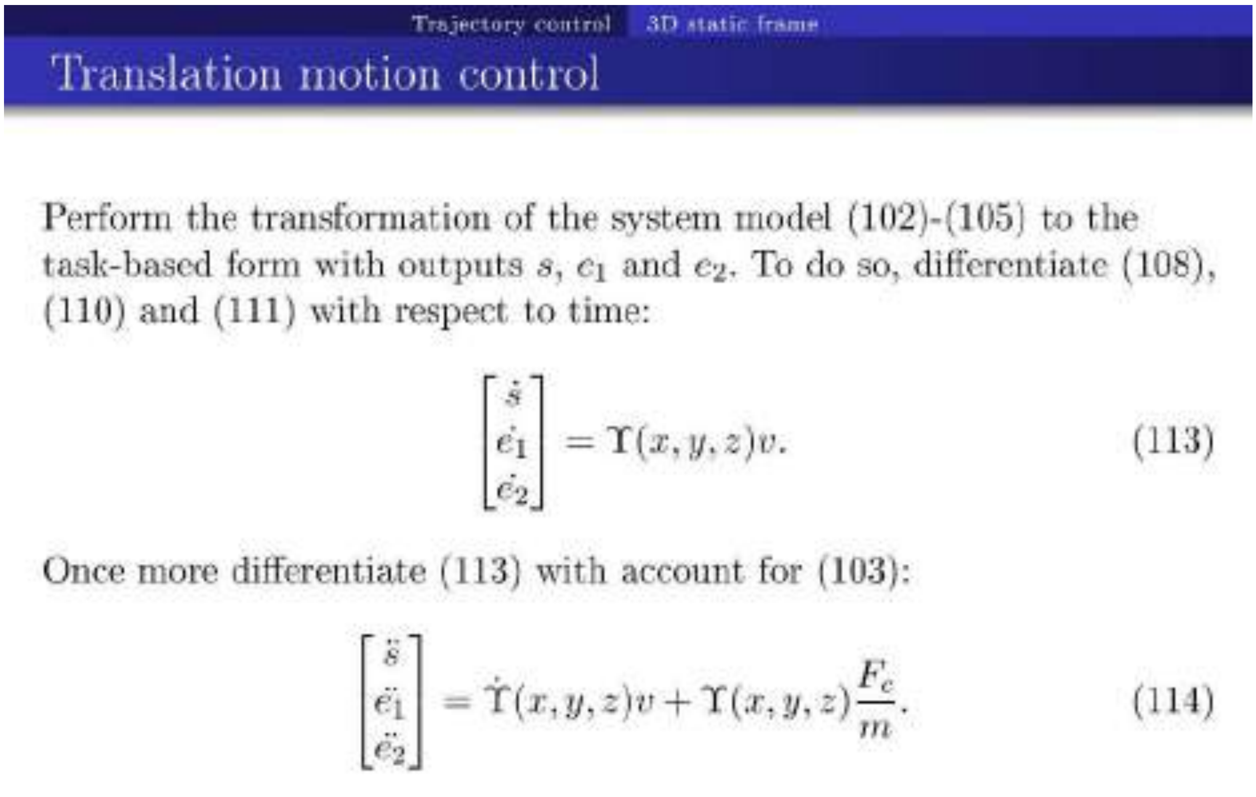
$$e_2 = \varphi_2(x, y, z). \tag{111}$$

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

$$\Delta V_s = V_s^* - \dot{s}; \tag{112}$$

- stabilization of robot angular orientation with respect to curve S.



Trajectory control | 3D static frame

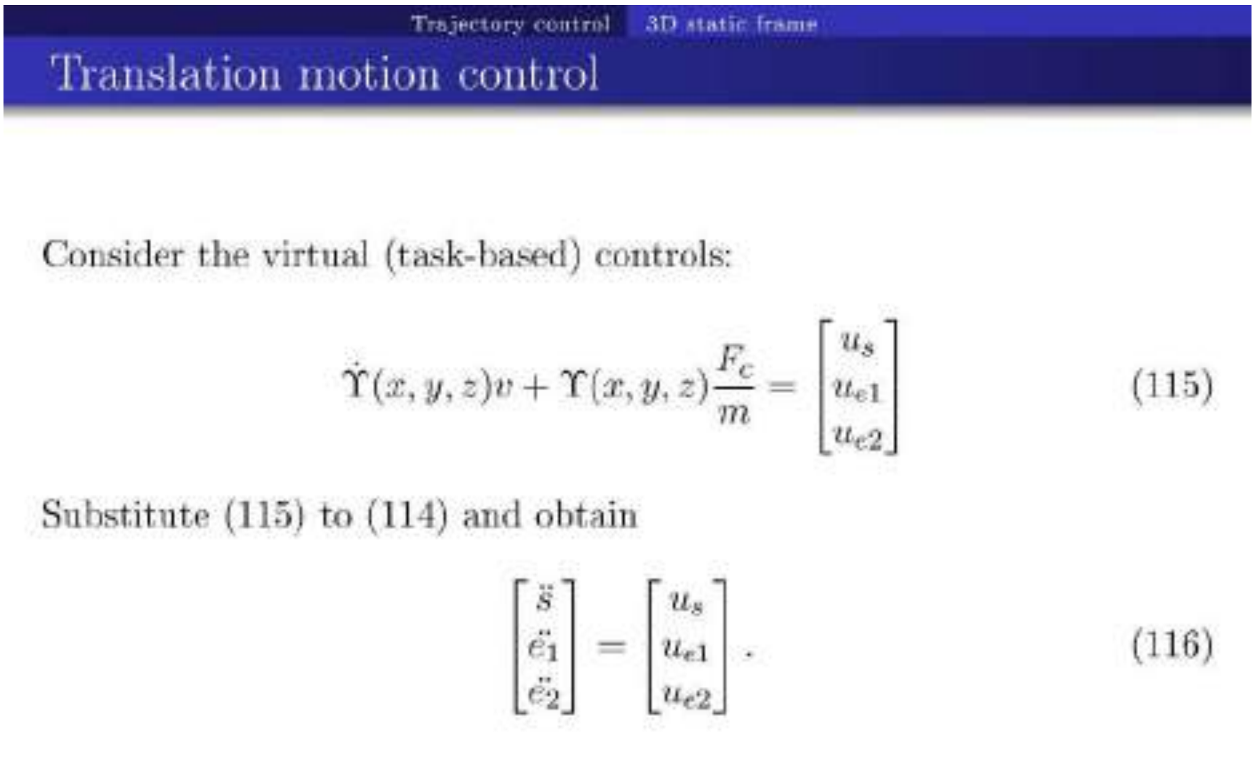
Translation motion control

Perform the transformation of the system model (102)-(105) to the task-based form with outputs s, e_1 and e_2. To do so, differentiate (108), (110) and (111) with respect to time:

$$\begin{bmatrix} \dot{s} \\ \dot{e}_1 \\ \dot{e}_2 \end{bmatrix} = \Upsilon(x, y, z)v. \tag{113}$$

Once more differentiate (113) with account for (103):

$$\begin{bmatrix} \ddot{s} \\ \ddot{e}_1 \\ \ddot{e}_2 \end{bmatrix} = \dot{\Upsilon}(x, y, z)v + \Upsilon(x, y, z)\frac{F_c}{m}. \tag{114}$$



Trajectory control | 3D static frame

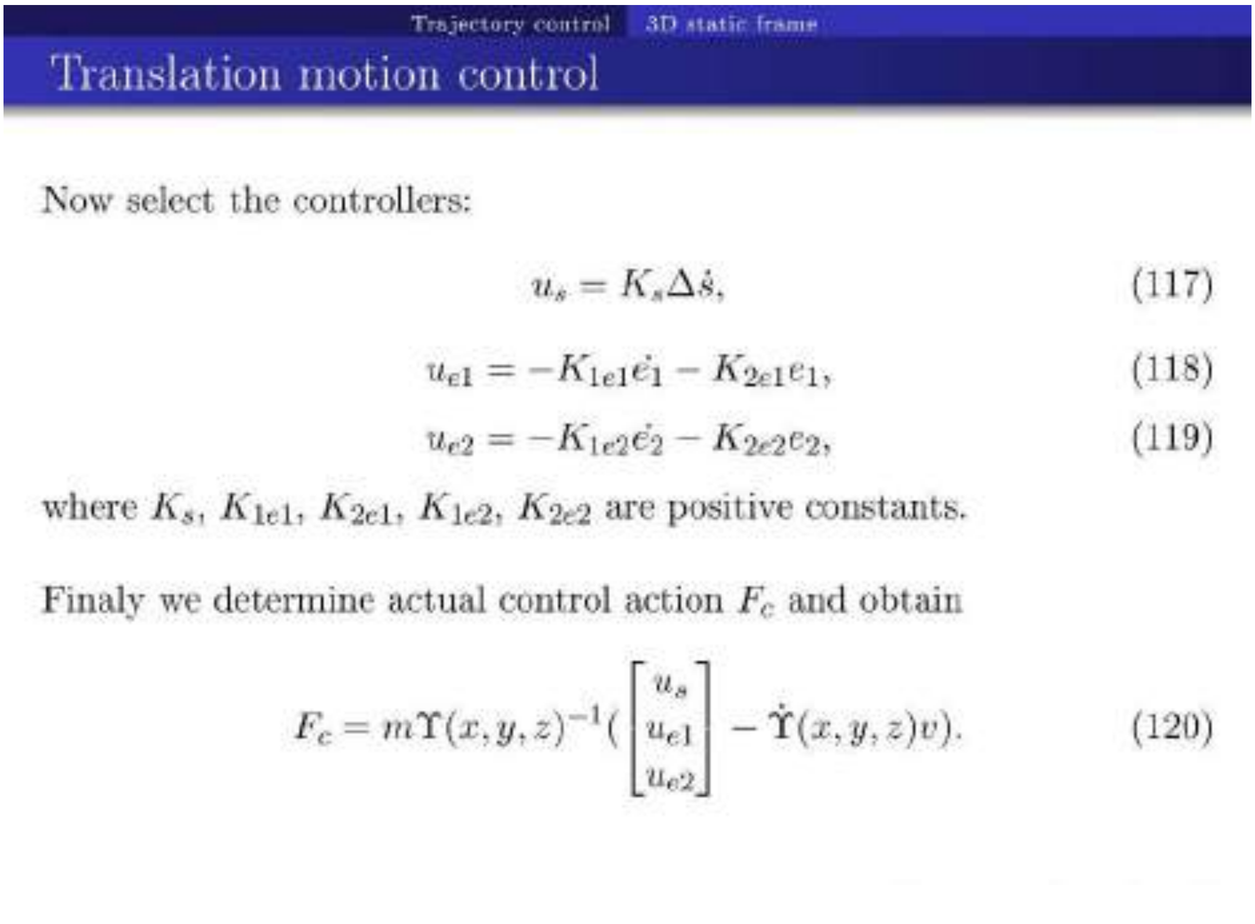
Translation motion control

Consider the virtual (task-based) controls:

$$\dot{\Upsilon}(x,y,z)v + \Upsilon(x,y,z)\frac{F_c}{m} = \begin{bmatrix} u_s \\ u_{e1} \\ u_{e2} \end{bmatrix} \tag{115}$$

Substitute (115) to (114) and obtain

$$\begin{bmatrix} \ddot{s} \\ \ddot{e}_1 \\ \ddot{e}_2 \end{bmatrix} = \begin{bmatrix} u_s \\ u_{e1} \\ u_{e2} \end{bmatrix}. \tag{116}$$



Trajectory control | 3D static frame

Translation motion control

Now select the controllers:

$$u_s = K_s \Delta \dot{s}, \tag{117}$$

$$u_{e1} = -K_{1e1}\dot{e}_1 - K_{2e1}e_1, \tag{118}$$

$$u_{e2} = -K_{1e2}\dot{e}_2 - K_{2e2}e_2, \tag{119}$$

where K_s, K_{1e1}, K_{2e1}, K_{1e2}, K_{2e2} are positive constants.

Finaly we determine actual control action F_c and obtain

$$F_c = m\Upsilon(x,y,z)^{-1}(\begin{bmatrix} u_s \\ u_{e1} \\ u_{e2} \end{bmatrix} - \dot{\Upsilon}(x,y,z)v). \tag{120}$$

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

$$R(\delta) = R(\alpha)R^T(\alpha^*)R^T(\Delta), \tag{121}$$

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

$$e_r = \frac{1}{2}(R(\delta) - R(\delta)^T)^\vee, \tag{122}$$

where $\vee$ it the transformation $SO(3) \to R^3$.

Rotation motion control

Define the angular speed error e_ω. Differentiate (121) with account for (104) and obtain the equation

$$\frac{d}{dt}R(\delta) = S(\dot{\delta})R(\delta) = e_\omega R(\delta), \tag{123}$$

$$\frac{d}{dt}R(\delta) = S(\omega)R(\delta) - R(\alpha)R^T(\alpha^*)S(\omega^*)R^T(\Delta), \tag{124}$$

Use the property of skew symmetric matrix $RS(\omega)R^T = S(R\omega)$ and obtain final expression

$$\frac{d}{dt}R(\delta) = (S(\omega) - S(R(\alpha)R^T(\alpha^*)\omega^*))R(\delta), \tag{125}$$

and

$$e_\omega = \omega - R(\alpha)R^T(\alpha^*)\omega^*. \tag{126}$$

Rotation motion control

Differentiating (126) with account for (104)

$$\dot{e_\omega} = \frac{1}{J}(M - \omega \times J\omega) + a_d, \tag{127}$$

where $a_d = -S(\omega)R(\alpha)R^T(\alpha^*)\omega^* + R(\alpha)R^T(\alpha^*)\dot{\omega}^*$. Resulting attitude controller has form

$$M_c = \omega \times J\omega - Ja_d - K_R e_r - K_\omega e_\omega. \tag{128}$$

where K_R, K_ω, are positive constants.

Numerical example

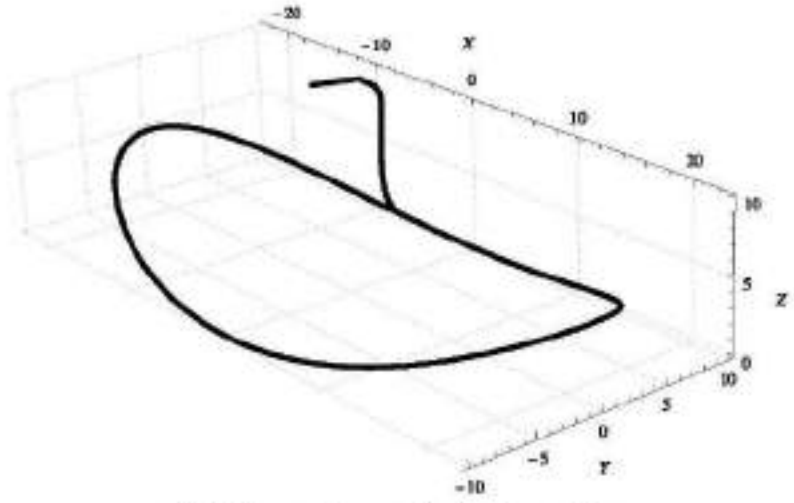
Consider the plant as a rigid body described by model (102)-(105) with $m = 1$, $J = 1$.

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 $\dot{s} = 1$.

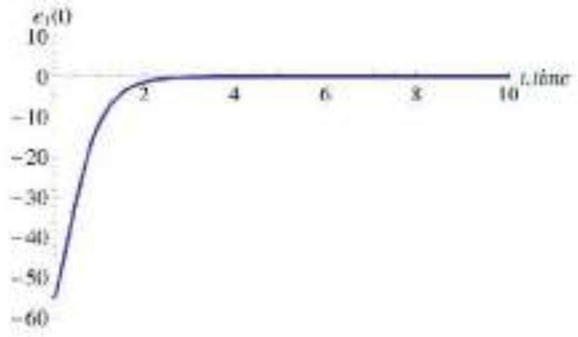
Numerical example

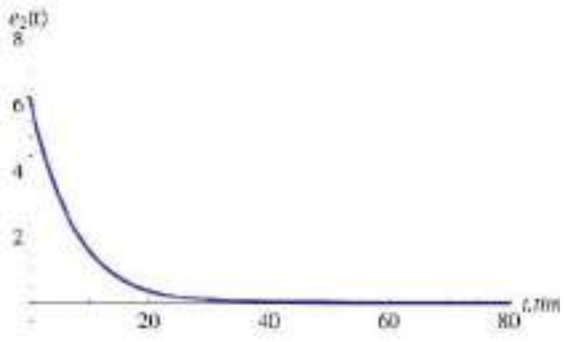


Motion along desired path:

$$\varphi_1(x,y,z) = 0.2x^2 + y^2 - R^2 = 0 \cap \varphi_2(x,y,z) = z + 0.05y^2 - 5 = 0$$

Numerical example

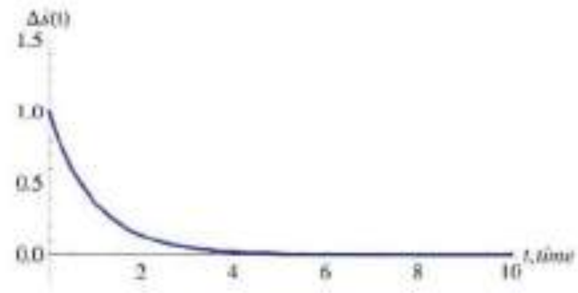




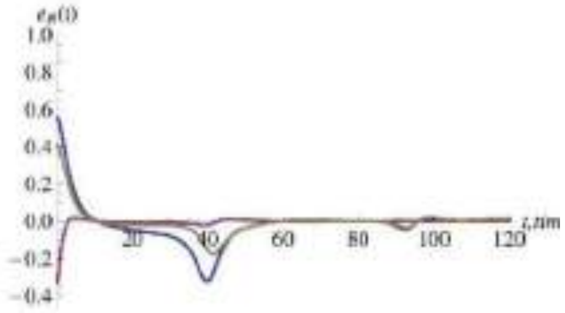
Position error $e_1 = \varphi_1(x,y,z)$

Position error $e_2 = \varphi_2(x,y,z)$

Numerical example



Speed error $\Delta V = \dot{s}^* - \dot{s}$



Angular error e_r

2D moving frame

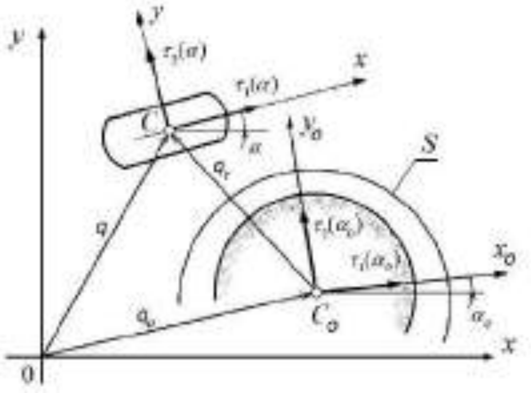
Dynamic model of the plant:

$$\begin{bmatrix} \dot{y} \\ \dot{\alpha} \end{bmatrix} = R^T(\alpha) \begin{bmatrix} V_z \\ \omega \end{bmatrix}, \quad (129)$$

$$A \begin{bmatrix} \dot{V}_z \\ \dot{\omega} \end{bmatrix} = R^T(\alpha) \begin{bmatrix} F_z \\ M \end{bmatrix}, \quad (130)$$

$$R(\alpha) = \begin{bmatrix} T(\alpha) & 0 \\ 0 & 1 \end{bmatrix},$$

$$A = \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & J \end{bmatrix},$$



External moving object

Dynamic model of the external moving object:

$$\dot{y}^o = V^o, \tag{131}$$

$$\dot{T}(\alpha^o) = \omega^o E T(\alpha^o), \tag{132}$$

Desired trajectory in relative coordinates

$$\varphi(x) = 0, \tag{133}$$

Local coordinate

$$s = \psi(x), \tag{134}$$

Relative coordinates

Position, velocity and acceleration of the plant in moving frame:

$$x = T(\alpha^o)(y - y^o), \tag{135}$$

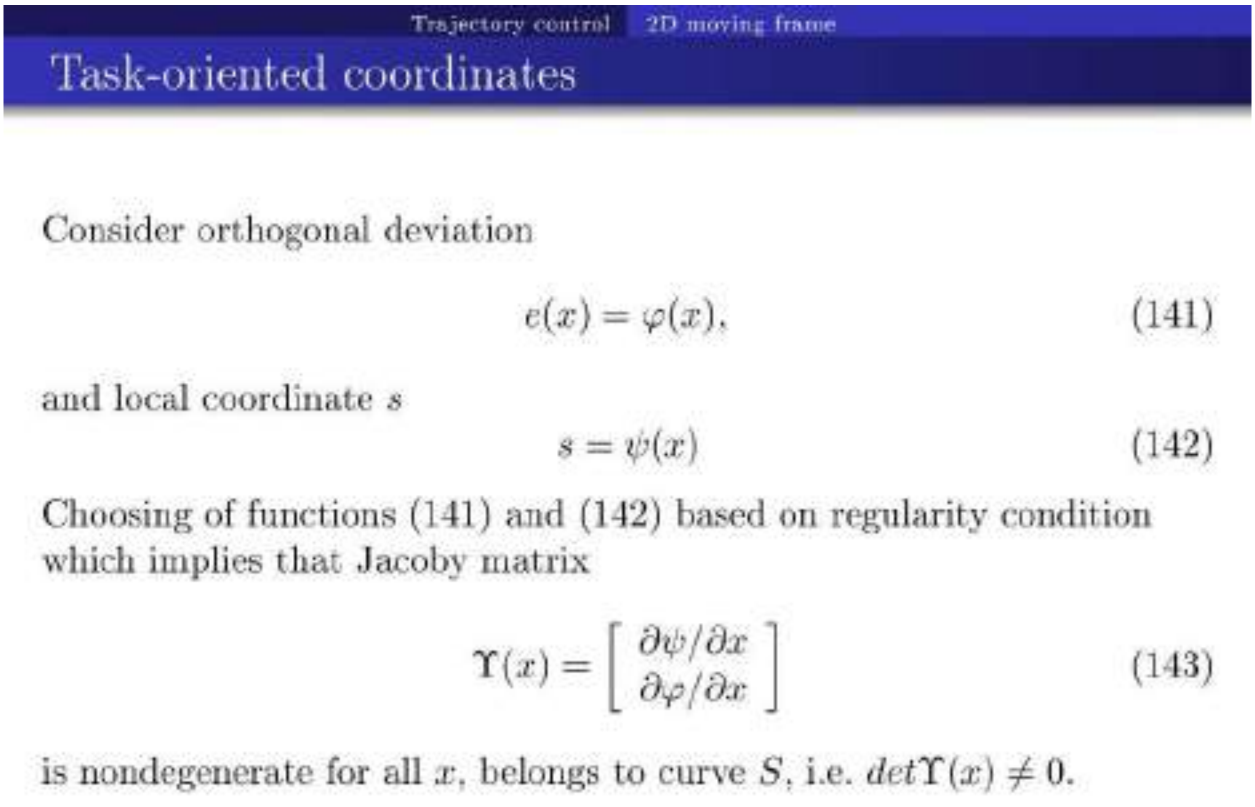
$$\alpha_x = \alpha - \alpha^o. \tag{136}$$

$$\dot{x} = \omega^o E x + T(\alpha^o)\left(\dot{y} - \dot{y}^o\right), \tag{137}$$

$$\dot{\alpha}_x = \omega - \omega^o, \tag{138}$$

$$\begin{aligned} \ddot{x} &= (\omega^o)^2 x + 2\omega^o E T(\alpha^o)\left(\dot{y} - \dot{y}^o\right) + \\ &+ \frac{1}{m} T(\alpha^o) T^T(\alpha) F_z, \end{aligned} \tag{139}$$

$$\ddot{\alpha}_x = \frac{1}{J} M. \tag{140}$$



Task-oriented coordinates

Consider orthogonal deviation

$$e(x) = \varphi(x), \tag{141}$$

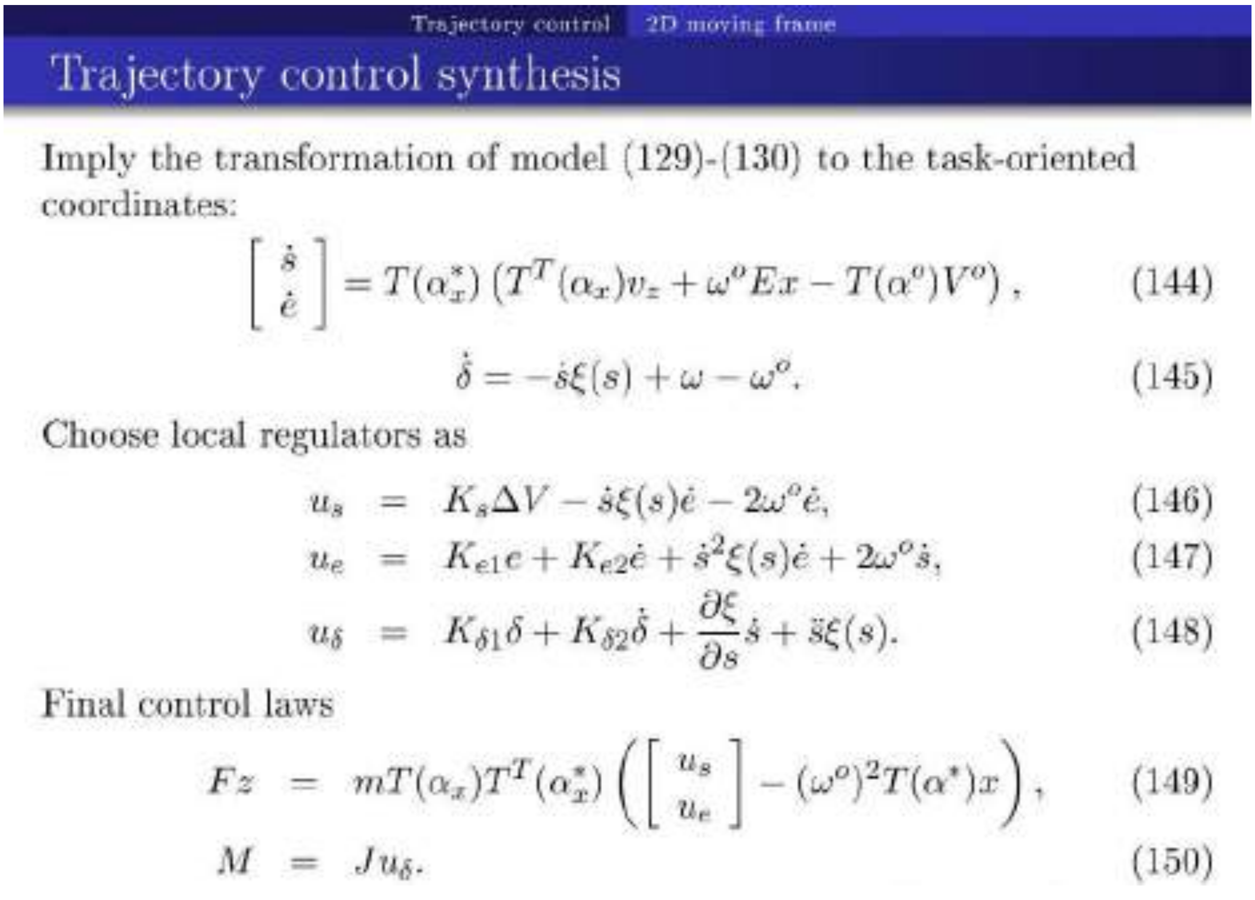
and local coordinate s

$$s = \psi(x) \tag{142}$$

Choosing of functions (141) and (142) based on regularity condition which implies that Jacoby matrix

$$\Upsilon(x) = \begin{bmatrix} \partial\psi/\partial x \\ \partial\varphi/\partial x \end{bmatrix} \tag{143}$$

is nondegenerate for all x, belongs to curve S, i.e. $det\Upsilon(x) \neq 0$.



Trajectory control synthesis

Imply the transformation of model (129)-(130) to the task-oriented coordinates:

$$\begin{bmatrix} \dot s \\ \dot e \end{bmatrix} = T(\alpha_x^*)\left(T^T(\alpha_x)v_z + \omega^o E x - T(\alpha^o)V^o\right), \tag{144}$$

$$\dot\delta = -\dot s\xi(s) + \omega - \omega^o. \tag{145}$$

Choose local regulators as

$$\begin{aligned} u_s &= K_s\Delta V - \dot s\xi(s)\dot e - 2\omega^o\dot e, && (146)\\ u_e &= K_{e1}e + K_{e2}\dot e + \dot s^2\xi(s)\dot e + 2\omega^o\dot s, && (147)\\ u_\delta &= K_{\delta 1}\delta + K_{\delta 2}\dot\delta + \frac{\partial\xi}{\partial s}\dot s + \ddot s\xi(s). && (148)\end{aligned}$$

Final control laws

$$\begin{aligned} Fz &= mT(\alpha_x)T^T(\alpha_x^*)\left(\begin{bmatrix} u_s \\ u_e \end{bmatrix} - (\omega^o)^2T(\alpha^*)x\right), && (149)\\ M &= Ju_\delta. && (150)\end{aligned}$$

Collision avoidance strategies

Some collision avoidance strategies

- Bypass
- Detour

Equidistant border around the obstacle

$$\varphi^*(q) = x^2 + y^2 - R^2 = 0, \tag{151}$$

Trajectory control in presence of moving external object.

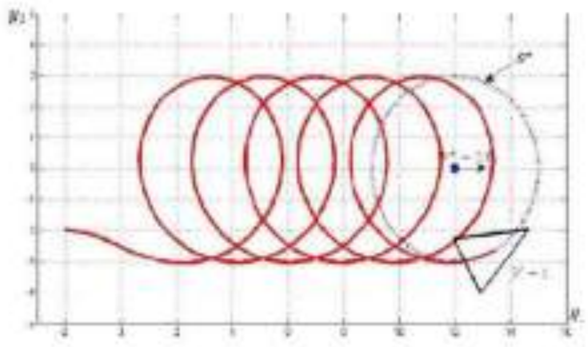


Рис.: The results of modeling the motion relative to a moving external object.

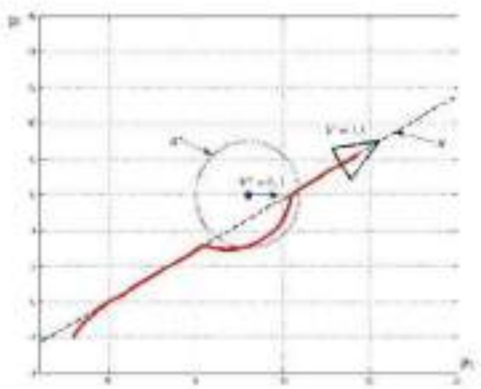


Рис.: The results of modeling the detour of a moving obstacle.

Moving frame description

Model of the plant motion:

$$\ddot{x}(t) = g - \frac{f(t)}{m}\vec{n}(t), \qquad (152)$$

$$\dot{R}(t) = R(t)S(\omega(t)), \qquad (153)$$

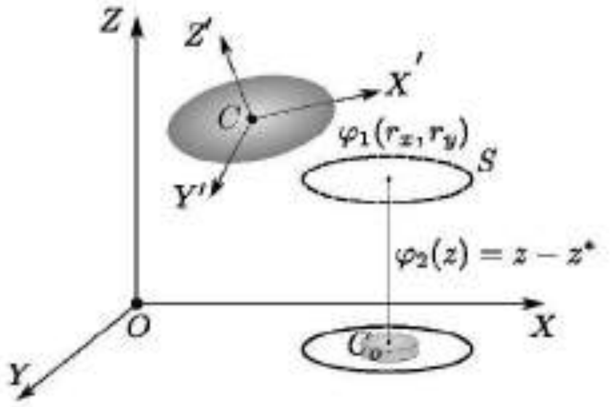
$$M_c(t) = J\dot{\omega}(t) + \omega(t) \times J\omega(t). \qquad (154)$$

Description of the moving frame:

$$\dot{x}_i = R_T(\alpha^*)v_i, \qquad (155)$$

$$\dot{\alpha}^* = \omega_i, \qquad (156)$$

$$\dot{R}_T(\alpha^*) = R_T S(\omega_i). \qquad (157)$$



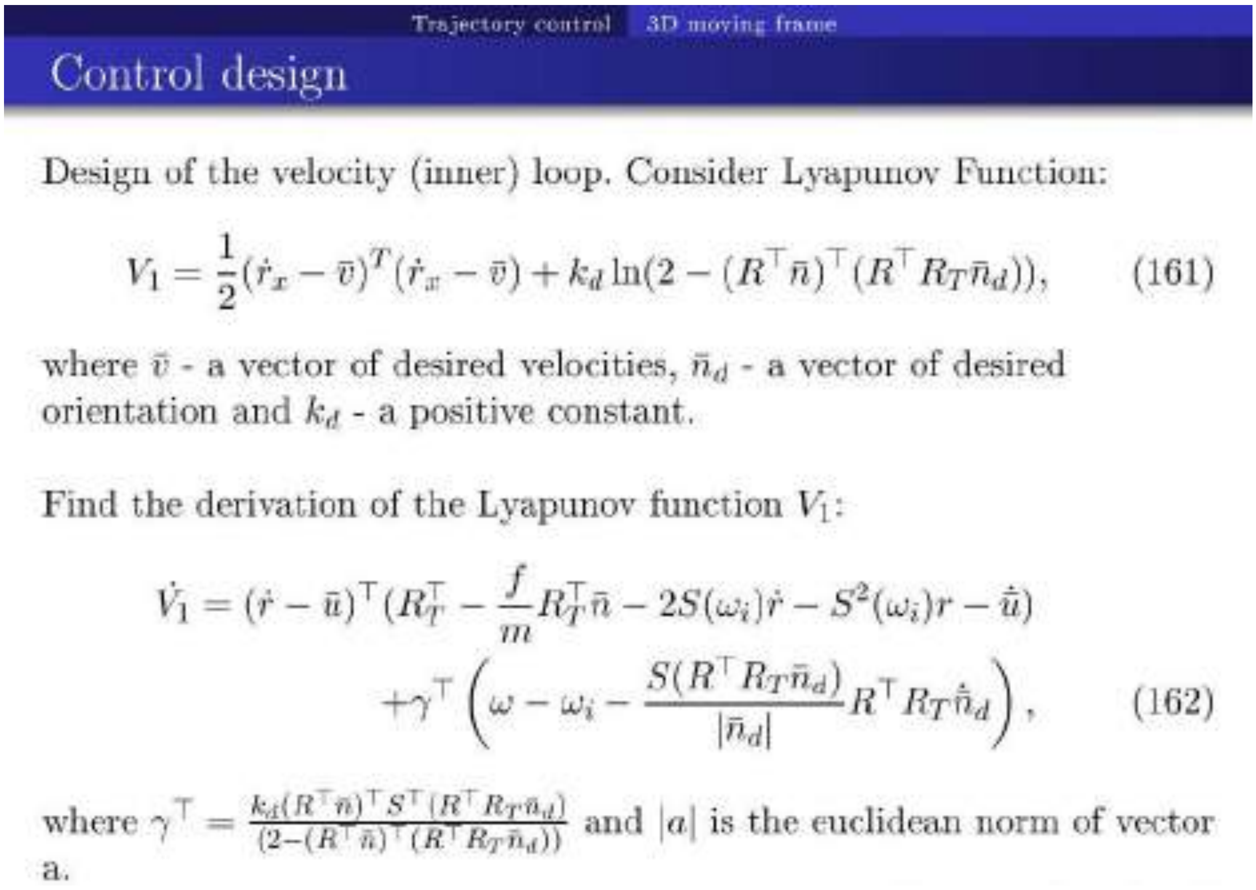
Moving frame description

Relative position:

$$r = R_T^\top(\alpha^*)(x - x_i), \qquad (158)$$

$$\dot{r} = R_T^\top(\alpha^*)\dot{x} - S(\omega_i)r, \qquad (159)$$

$$\ddot{r} = R_T^\top(\alpha^*)\ddot{x} - 2S(\omega_i)\dot{r} - S^2(\omega_i)r. \qquad (160)$$



Trajectory control 3D moving frame

Control design

Design of the velocity (inner) loop. Consider Lyapunov Function:

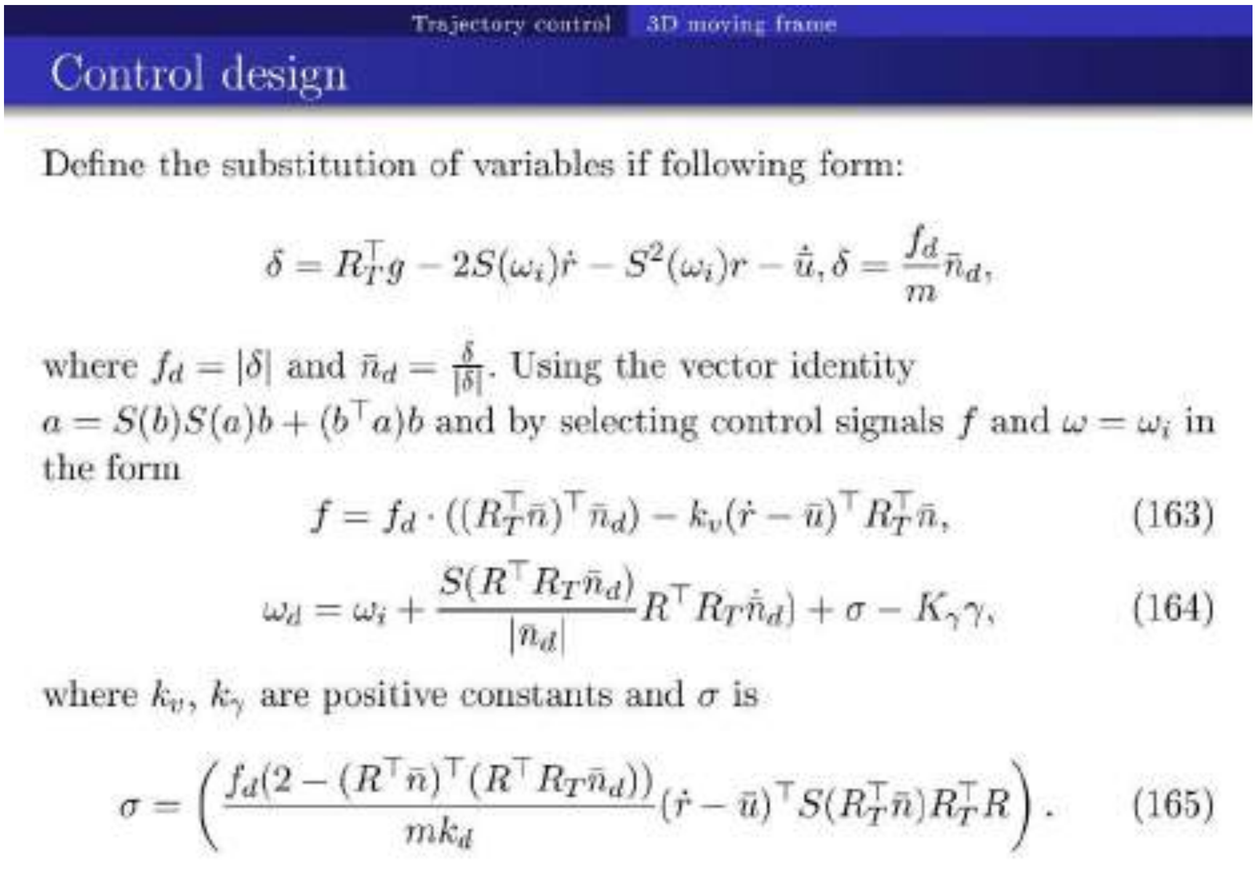
$$V_1 = \frac{1}{2}(\dot r_x - \bar v)^T(\dot r_x - \bar v) + k_d \ln(2 - (R^\top \bar n)^\top (R^\top R_T \bar n_d)), \qquad (161)$$

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:

$$\dot V_1 = (\dot r - \bar u)^\top (R_T^\top - \frac{f}{m} R_T^\top \bar n - 2S(\omega_i)\dot r - S^2(\omega_i) r - \dot{\bar u})$$
$$+\gamma^\top \left(\omega - \omega_i - \frac{S(R^\top R_T \bar n_d)}{|\bar n_d|} R^\top R_T \dot{\bar n}_d\right), \qquad (162)$$

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.



Trajectory control 3D moving frame

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 $\bar n_d = \frac{\delta}{|\delta|}$. Using the vector identity $a = S(b)S(a)b + (b^\top a)b$ and by selecting control signals f and $\omega = \omega_i$ in the form

$$f = f_d \cdot ((R_T^\top \bar n)^\top \bar n_d) - k_v (\dot r - \bar u)^\top R_T^\top \bar n, \qquad (163)$$

$$\omega_d = \omega_i + \frac{S(R^\top R_T \bar n_d)}{|\bar n_d|} R^\top R_T \dot{\bar n}_d) + \sigma - K_\gamma \gamma, \qquad (164)$$

where k_v, k_γ are positive constants and σ is

$$\sigma = \left(\frac{f_d(2 - (R^\top \bar n)^\top (R^\top R_T \bar n_d))}{m k_d} (\dot r - \bar u)^\top S(R_T^\top \bar n) R_T^\top R\right). \qquad (165)$$

ITMO UNIVERSITY

Then the derivation of the Lyapunov function V_1 is

$$\dot{V}_1 = -k_v((\dot{r} - \bar{u})^\top R_T^\top \bar{n})^2 - k_\gamma \gamma^\top \gamma \leq, \tag{166}$$

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 (158)-(160) to the task-based form with outputs s, e_1 and e_2, using Jacobian matrix (109):

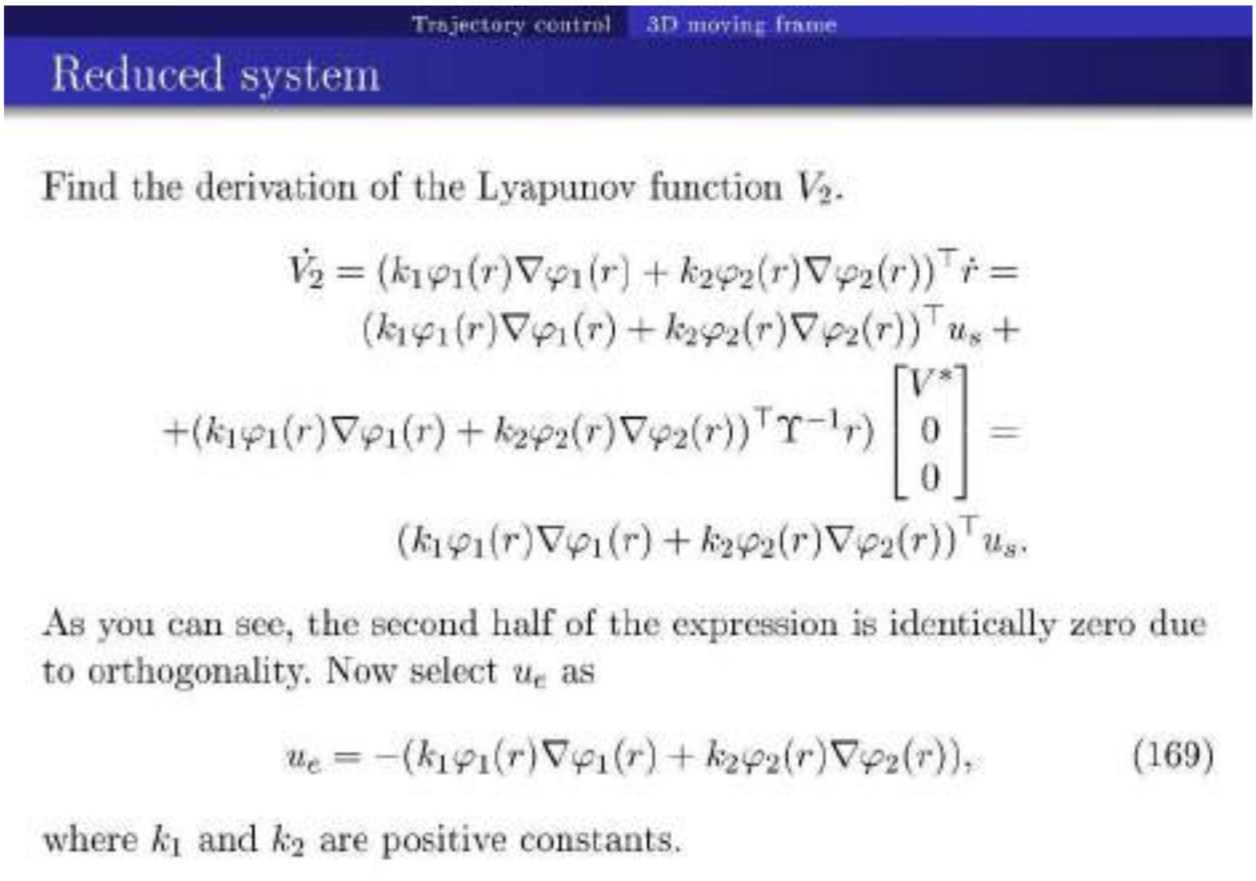
$$\begin{bmatrix} \dot{s} \\ \dot{e}_1 \\ \dot{e}_2 \end{bmatrix} = \Upsilon(r)\dot{r}$$

We can choose the control signal u_s in the form

$$u_s = \Upsilon^{-1}(r) \begin{bmatrix} V_s^* \\ 0 \\ 0 \end{bmatrix} \tag{167}$$

Now design stabilization control u_e. Consider Lyapunov Function:

$$V_2 = \frac{k_1}{2}\varphi_1^2(r) + \frac{k_2}{2}\varphi_2^2(r), \tag{168}$$



Trajectory control 3D moving frame

Reduced system

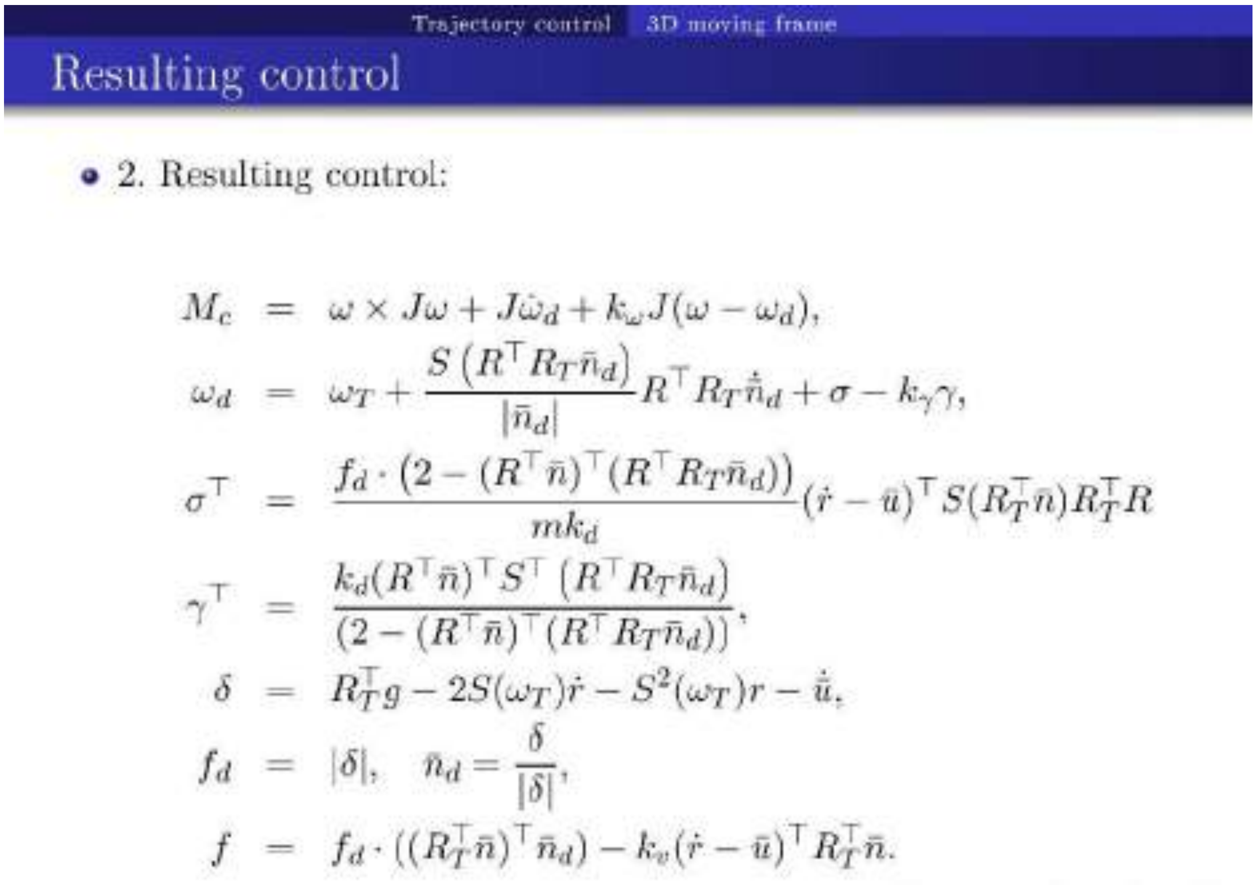
Find the derivation of the Lyapunov function V_2.

$$\dot{V_2} = (k_1\varphi_1(r)\nabla\varphi_1(r) + k_2\varphi_2(r)\nabla\varphi_2(r))^\top \dot{r} = \\ (k_1\varphi_1(r)\nabla\varphi_1(r) + k_2\varphi_2(r)\nabla\varphi_2(r))^\top u_s + \\ +(k_1\varphi_1(r)\nabla\varphi_1(r) + k_2\varphi_2(r)\nabla\varphi_2(r))^\top \Upsilon^{-1} r) \begin{bmatrix} V^* \\ 0 \\ 0 \end{bmatrix} = \\ (k_1\varphi_1(r)\nabla\varphi_1(r) + k_2\varphi_2(r)\nabla\varphi_2(r))^\top u_s.$$

As you can see, the second half of the expression is identically zero due to orthogonality. Now select u_e as

$$u_e = -(k_1\varphi_1(r)\nabla\varphi_1(r) + k_2\varphi_2(r)\nabla\varphi_2(r)), \tag{169}$$

where k_1 and k_2 are positive constants.



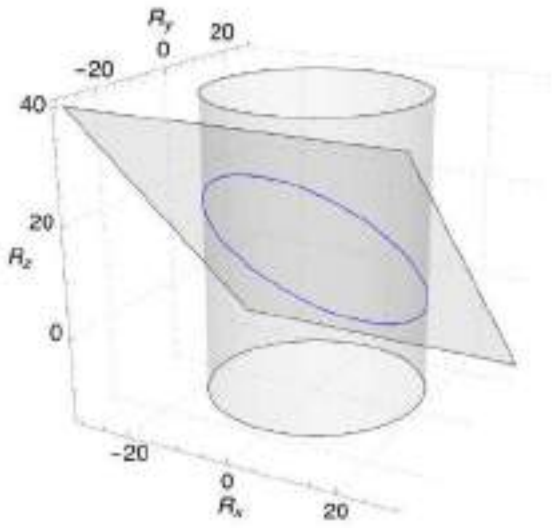
Trajectory control 3D moving frame

Resulting control

- 2. Resulting control:

$$\begin{aligned} M_c &= \omega \times J\omega + J\dot{\omega}_d + k_\omega J(\omega - \omega_d), \\ \omega_d &= \omega_T + \frac{S\left(R^\top R_T \bar{n}_d\right)}{|\bar{n}_d|} R^\top R_T \dot{\bar{n}}_d + \sigma - k_\gamma \gamma, \\ \sigma^\top &= \frac{f_d \cdot (2 - (R^\top \bar{n})^\top (R^\top R_T \bar{n}_d))}{m k_d} (\dot{r} - \bar{u})^\top S(R_T^\top \bar{n}) R_T^\top R \\ \gamma^\top &= \frac{k_d (R^\top \bar{n})^\top S^\top \left(R^\top R_T \bar{n}_d\right)}{(2 - (R^\top \bar{n})^\top (R^\top R_T \bar{n}_d))}, \\ \delta &= R_T^\top g - 2S(\omega_T)\dot{r} - S^2(\omega_T) r - \dot{\bar{u}}, \\ f_d &= |\delta|, \quad \bar{n}_d = \frac{\delta}{|\delta|}, \\ f &= f_d \cdot ((R_T^\top \bar{n})^\top \bar{n}_d) - k_v (\dot{r} - \bar{u})^\top R_T^\top \bar{n}. \end{aligned}$$

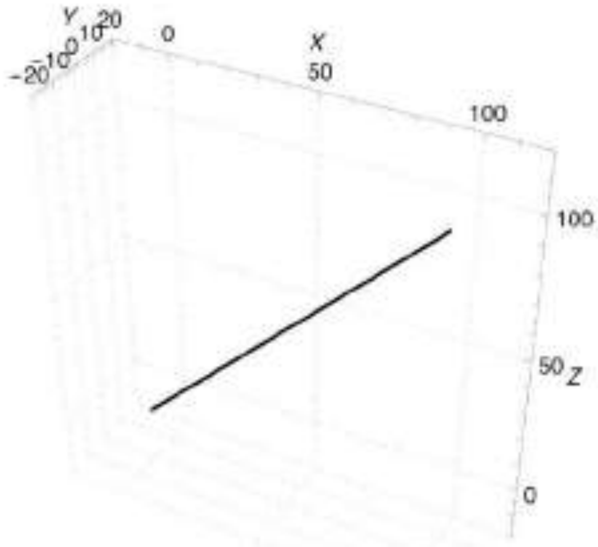
Example



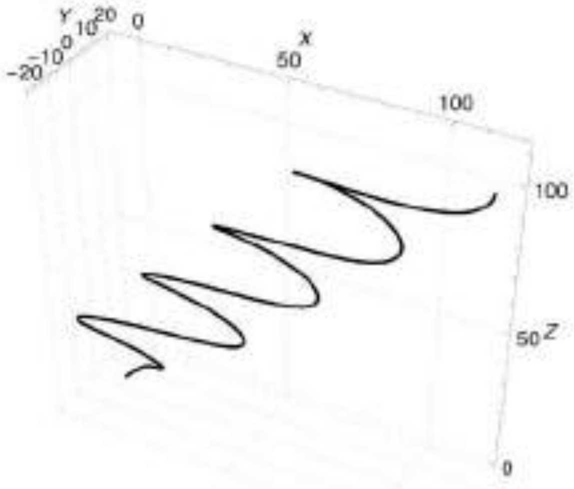
$$\varphi_1(r) = r_x^2 + r_y^2 - 400 = 0 \cap \varphi_2(r) = r_z + r_y - 10 = 0$$

The desired speed along the given path $\dot{s}^* = 30$.

Example

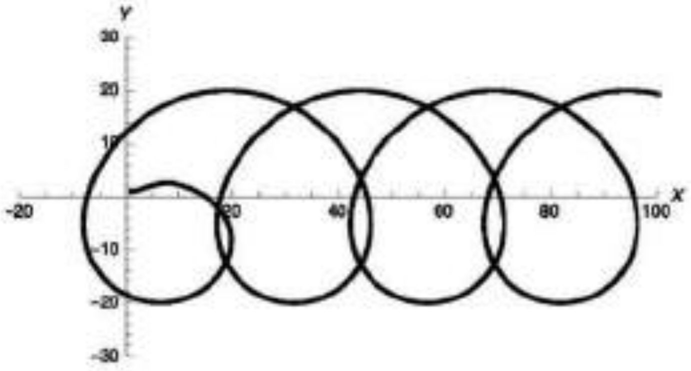


Moving frame spatial motion

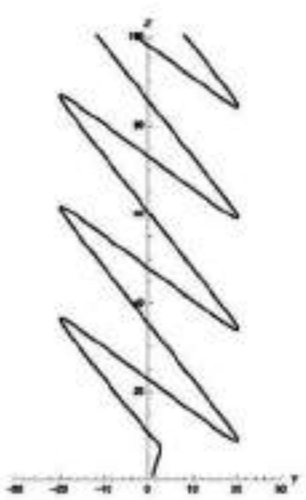


Plant spatial motion

Example

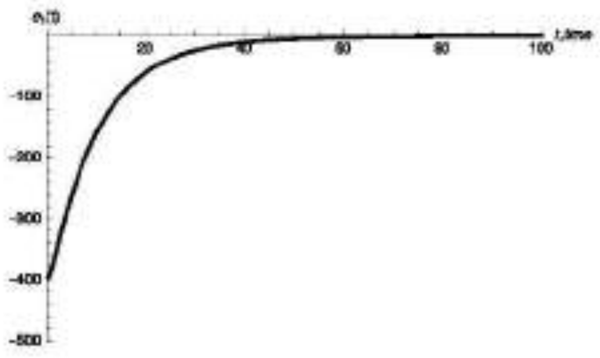


Projection of the plant motion on XY plane

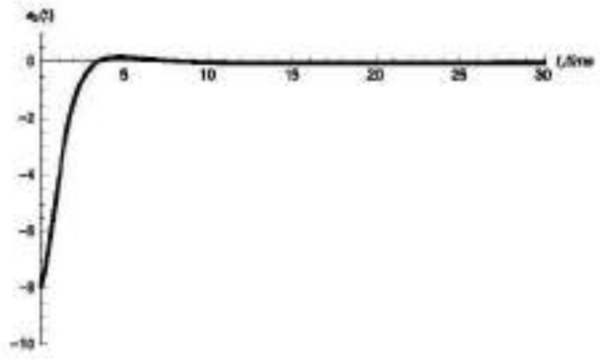


Projection of the plant motion on YZ plane

Example

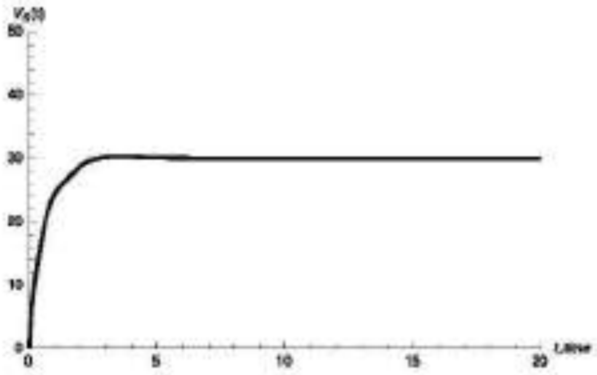


$e_1 = \varphi_1(r)$

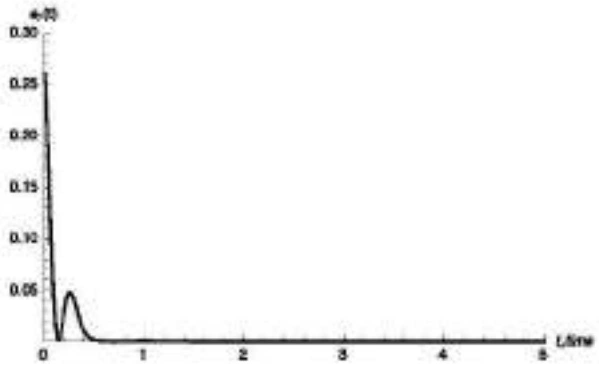


$e_2 = \varphi_2(r)$

Example



The velocity along the path
$V^*(t) = 30$



Angular error
$e_r = 1 - (R^\top \bar{n})^\top (R^\top R_T \bar{n}_d)$

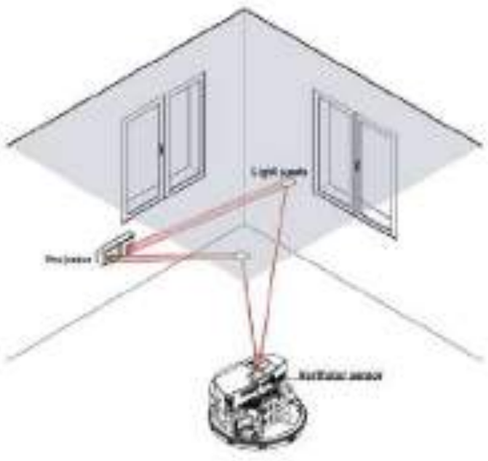
Omnidirectional mobile robot "Robotino" by Festo Didactics

Geometric dimensions:

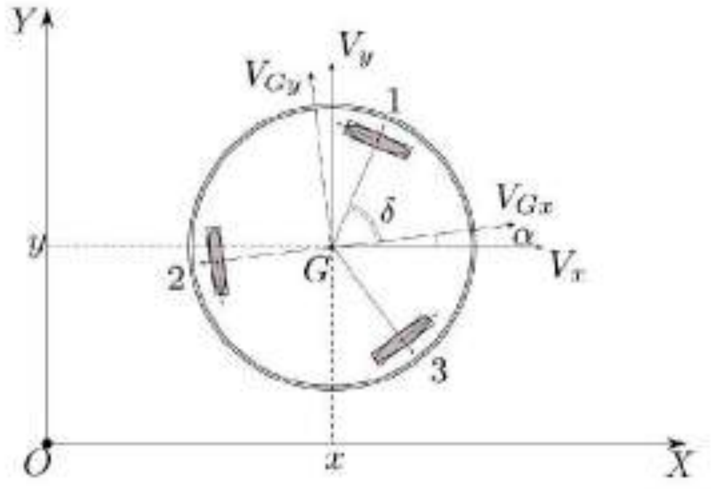
- Diameter: 370 mm
- Height: 210 mm
- Weight: 11 kg

Omni wheels "Robotino"

Local Navigation "Northstar"

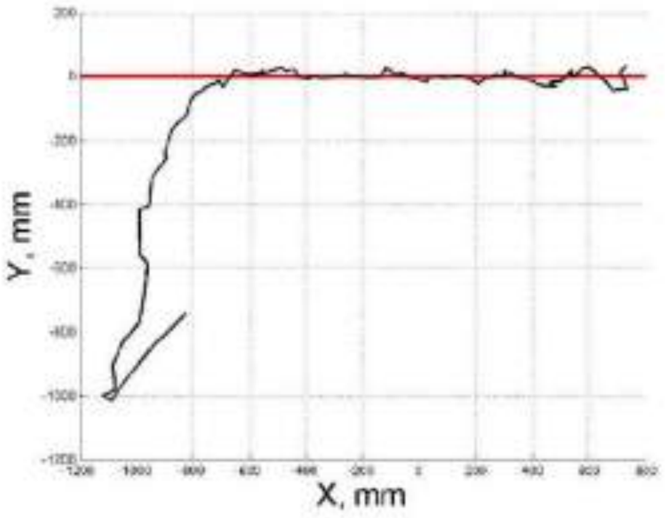


Mathematical model



$$\begin{bmatrix} \dot\theta_1 \\ \dot\theta_2 \\ \dot\theta_3 \end{bmatrix} = \begin{bmatrix} -\sin\frac{\pi}{3} & \cos\frac{\pi}{3} & L \\ 0 & -1 & L \\ \sin\frac{\pi}{3} & \cos\frac{\pi}{3} & L \end{bmatrix} \begin{bmatrix} V_{Gx} \\ V_{Gy} \\ \dot\alpha \end{bmatrix}$$

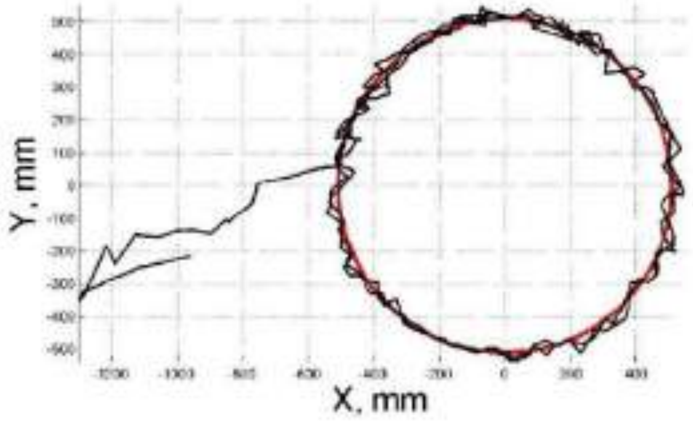
Motion along a straight line



$$\varphi(x, y) = -\sin\alpha x + \cos\alpha y = 0,$$

Experiments Results of experiments

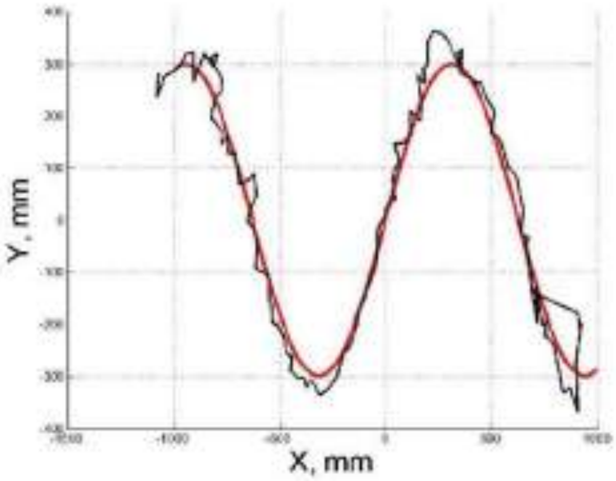
Motion along the circle



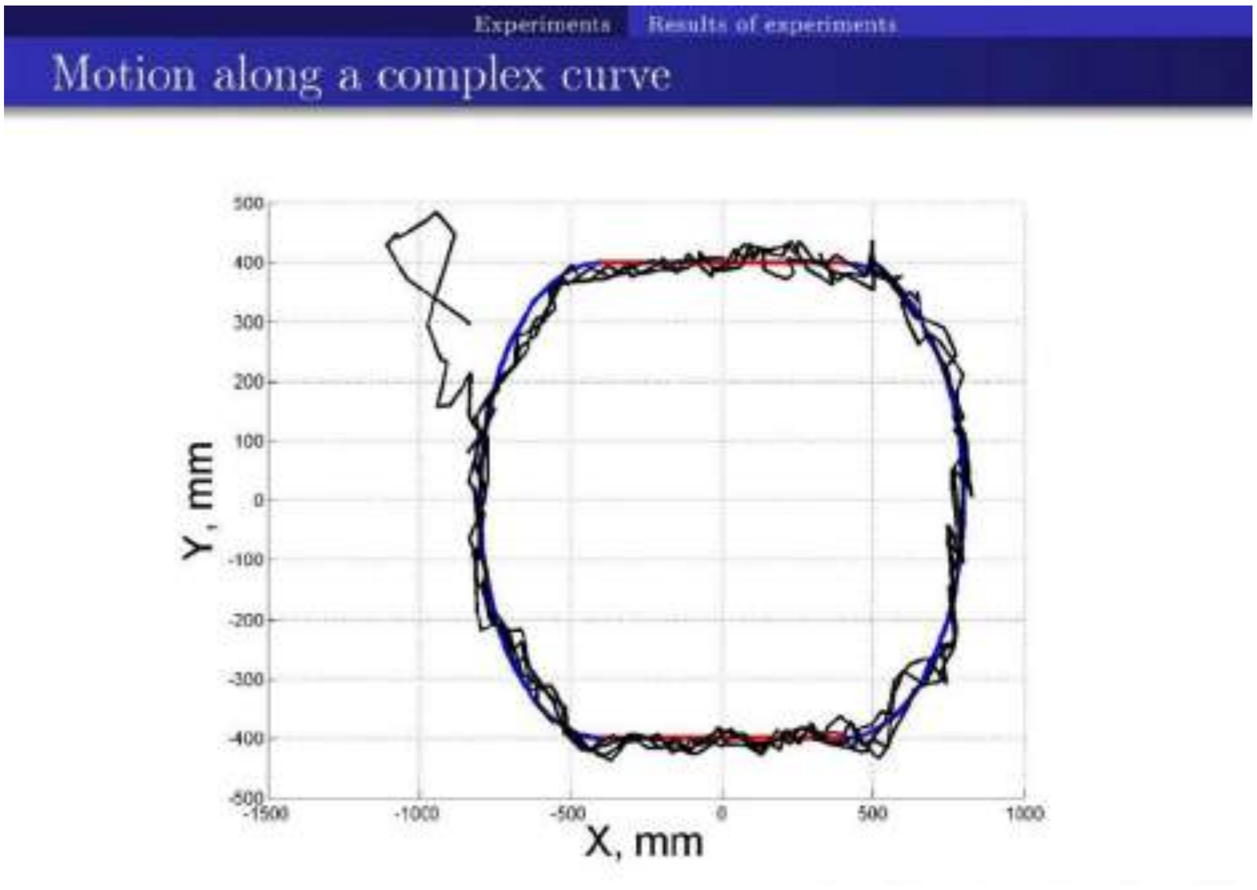
$$\varphi(x, y) = x^2 + y^2 - 2500 = 0$$

Experiments Results of experiments

Motion along the sinusoid



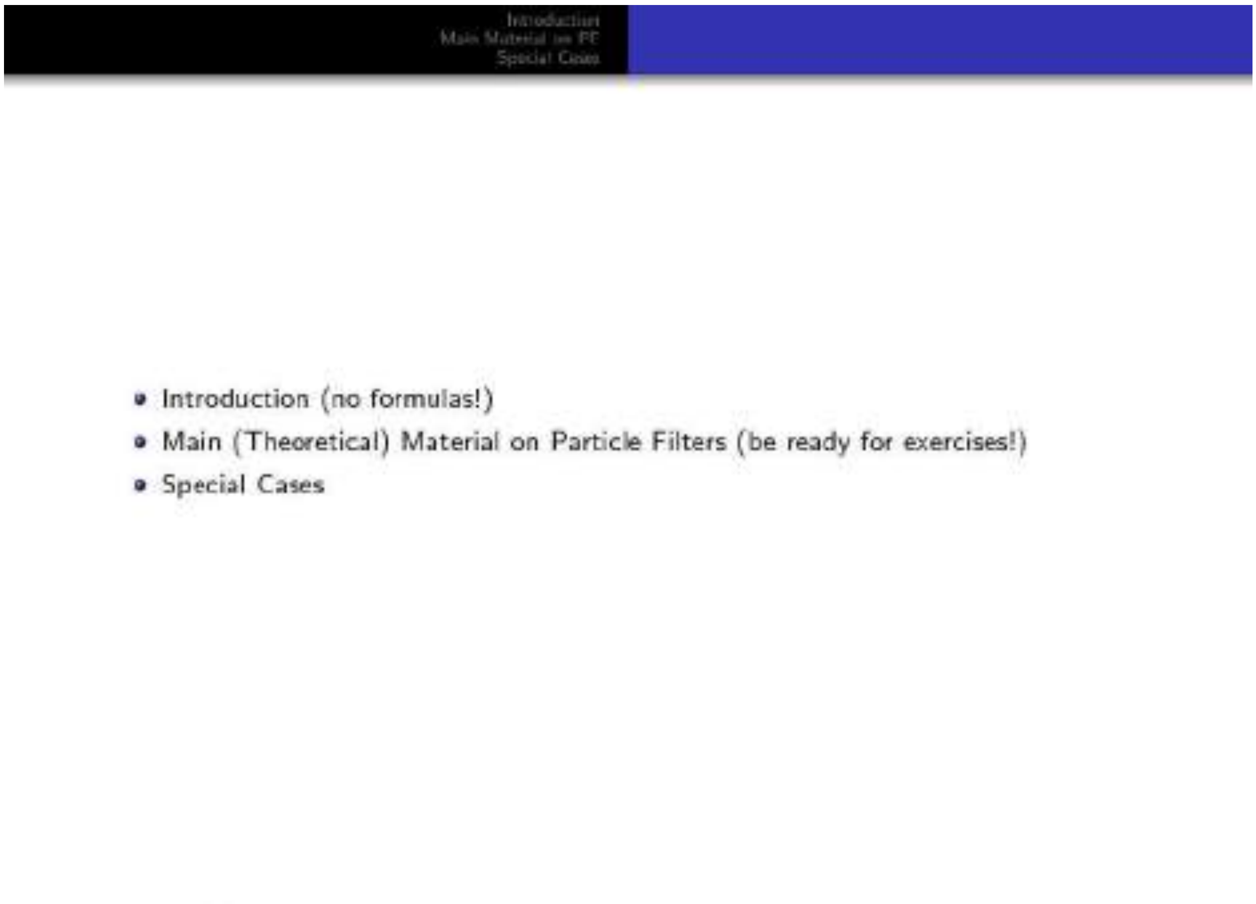
$$\varphi(x, y) = -300 \sin 0.005x + y = 0$$



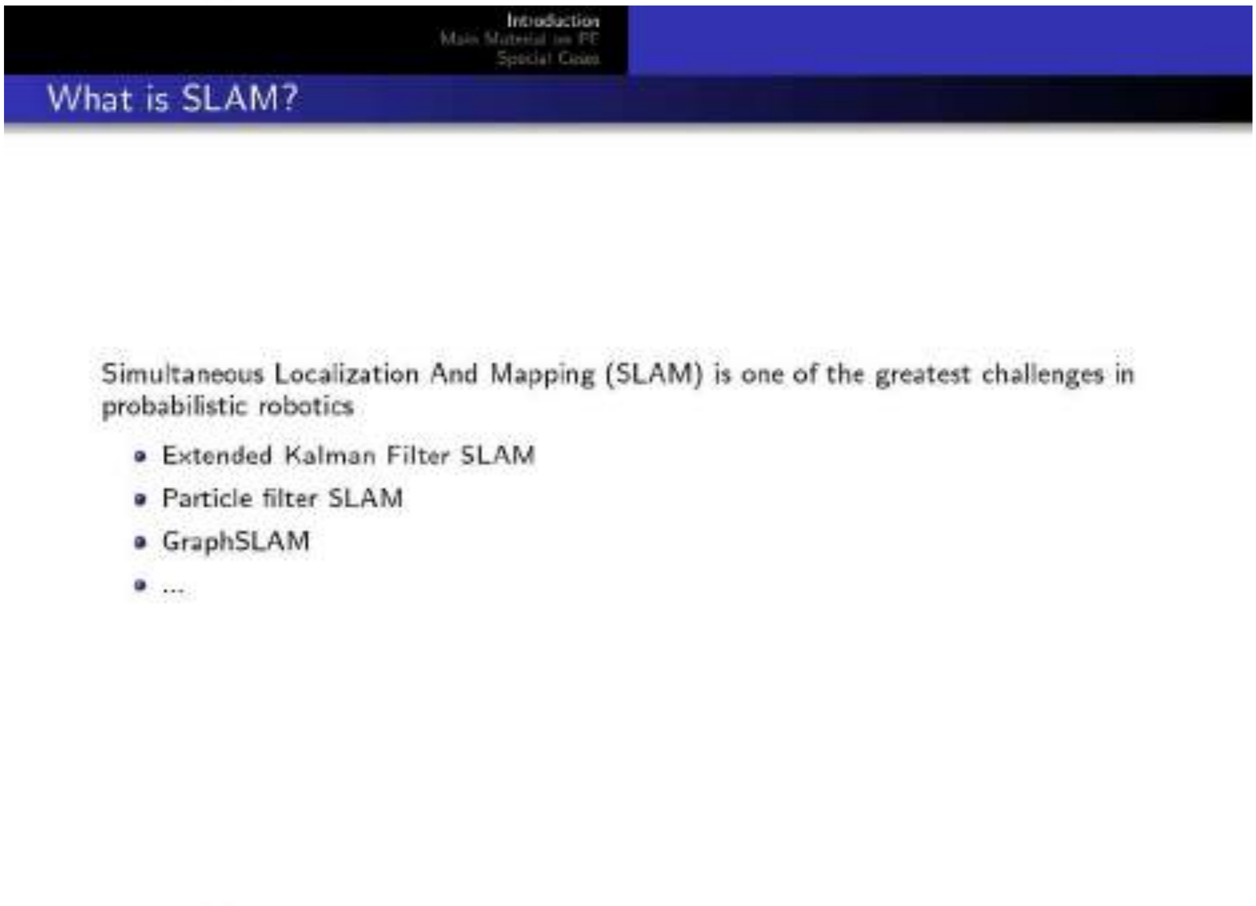
Experiments Results of experiments

Motion along a complex curve

Particle Filters and Its Applications in SLAM



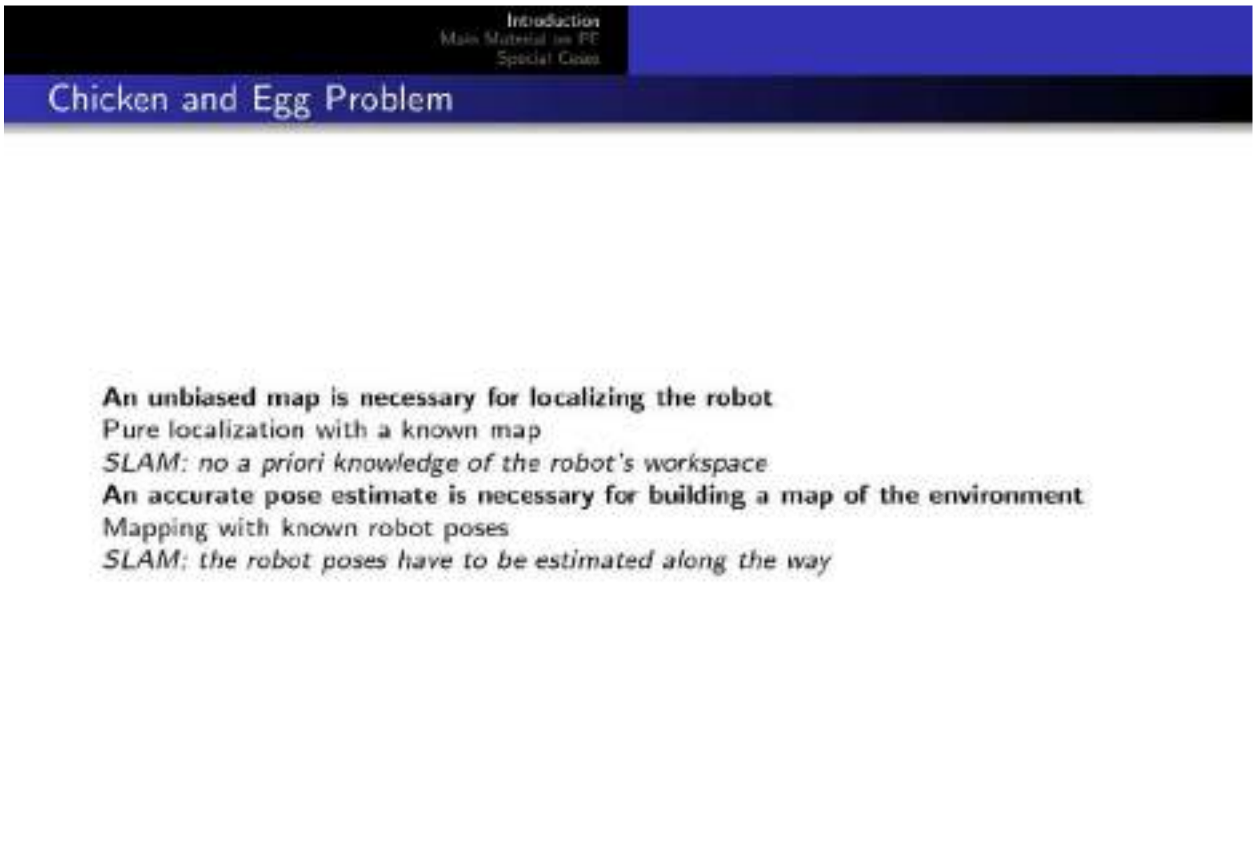
- Introduction (no formulas!)
- Main (Theoretical) Material on Particle Filters (be ready for exercises!)
- Special Cases



What is SLAM?

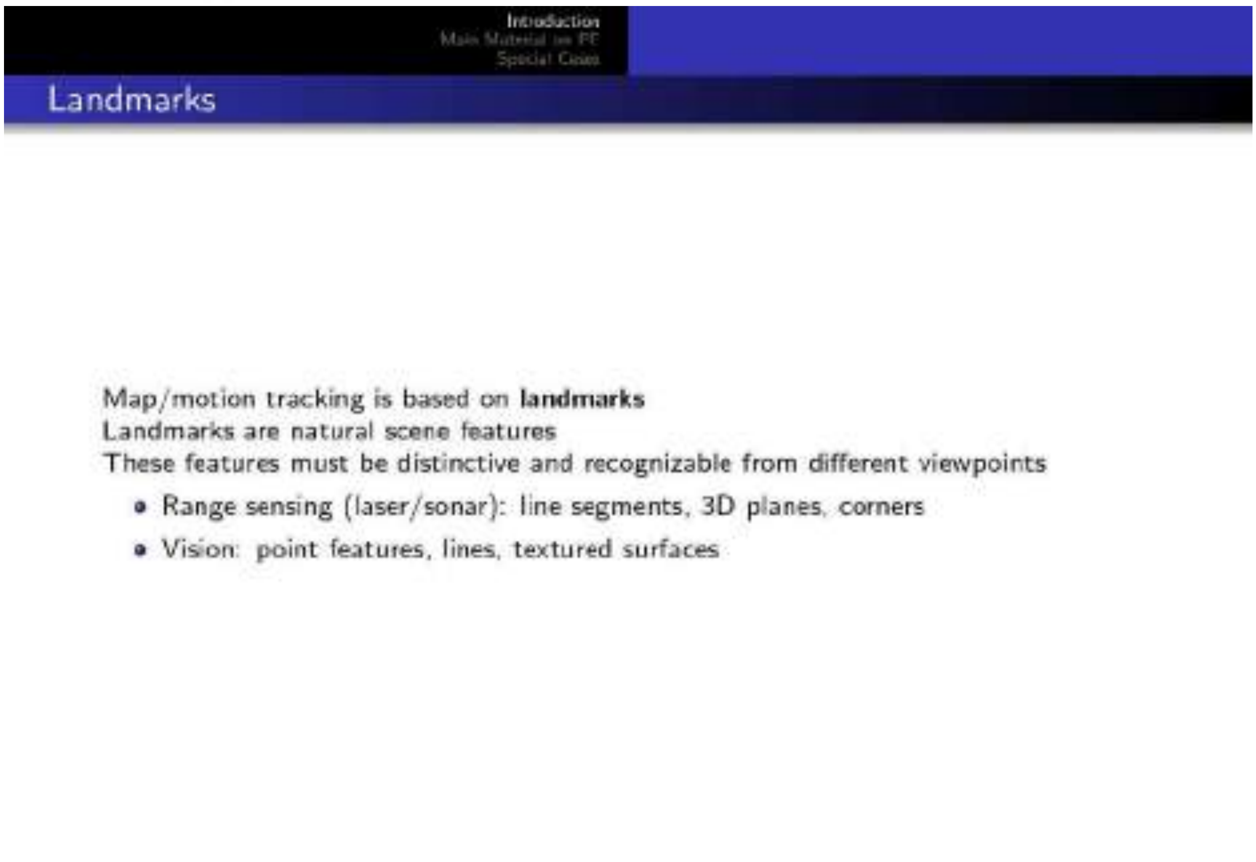
Simultaneous Localization And Mapping (SLAM) is one of the greatest challenges in probabilistic robotics

- Extended Kalman Filter SLAM
- Particle filter SLAM
- GraphSLAM
- ...



Chicken and Egg Problem

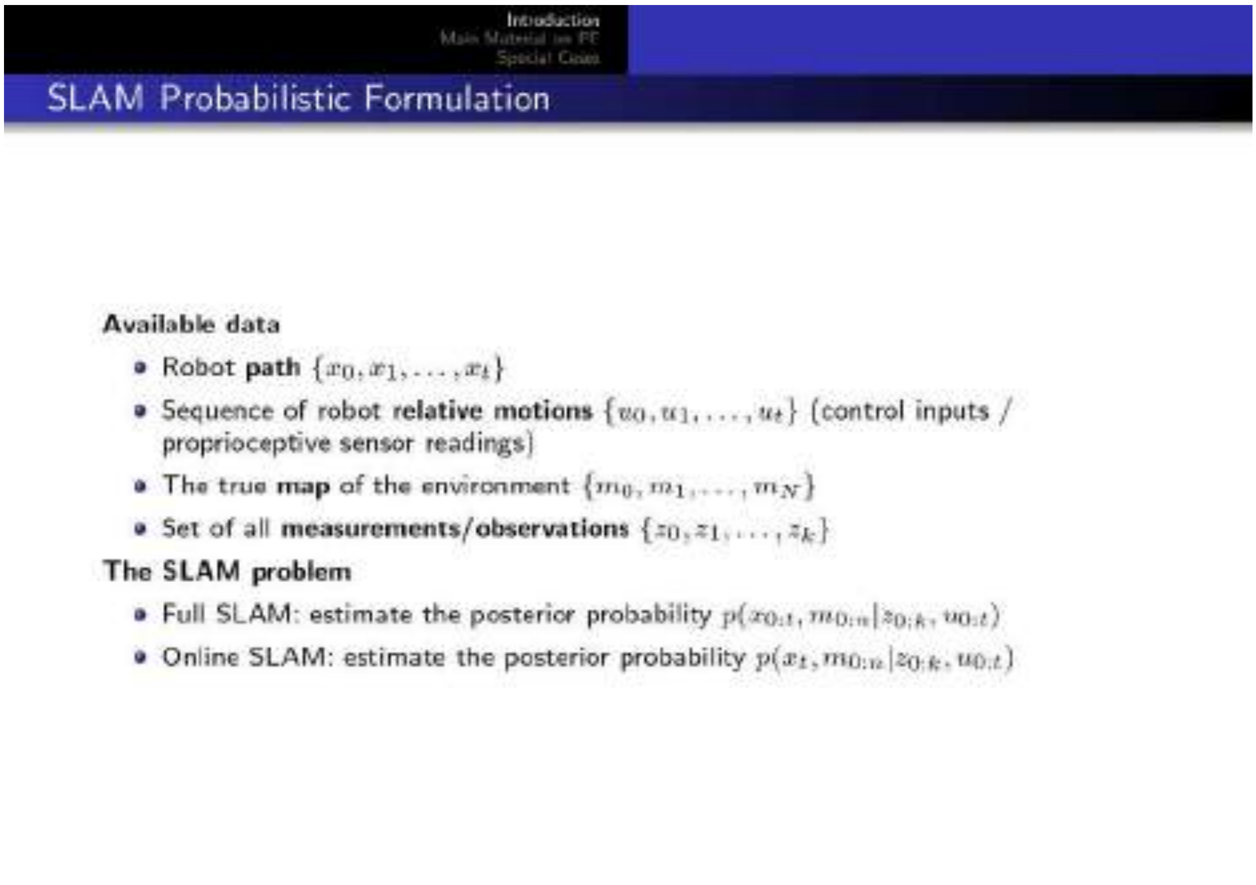
An unbiased map is necessary for localizing the robot
Pure localization with a known map
SLAM: no a priori knowledge of the robot's workspace
An accurate pose estimate is necessary for building a map of the environment
Mapping with known robot poses
SLAM: the robot poses have to be estimated along the way



Landmarks

Map/motion tracking is based on **landmarks**
Landmarks are natural scene features
These features must be distinctive and recognizable from different viewpoints

- Range sensing (laser/sonar): line segments, 3D planes, corners
- Vision: point features, lines, textured surfaces



SLAM Probabilistic Formulation

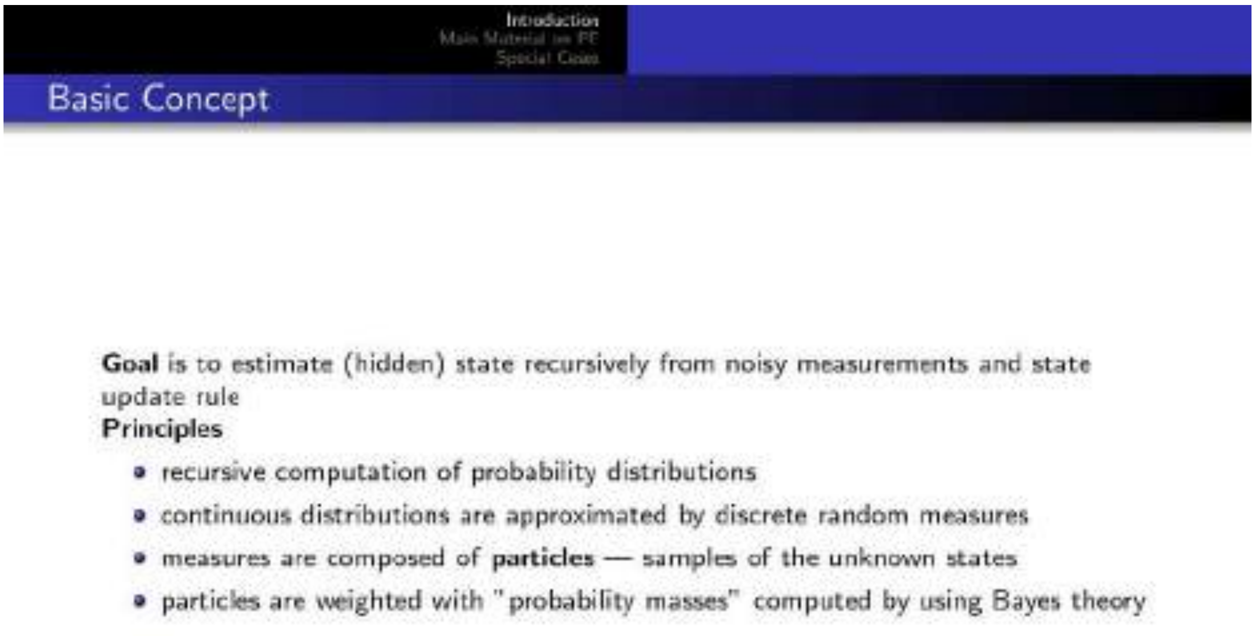
Available data

- Robot **path** $\{x_0, x_1, \ldots, x_t\}$
- Sequence of robot **relative motions** $\{u_0, u_1, \ldots, u_t\}$ (control inputs / proprioceptive sensor readings)
- The true **map** of the environment $\{m_0, m_1, \ldots, m_N\}$
- Set of all **measurements/observations** $\{z_0, z_1, \ldots, z_k\}$

The SLAM problem

- Full SLAM: estimate the posterior probability $p(x_{0:t}, m_{0:n} | z_{0:k}, u_{0:t})$
- Online SLAM: estimate the posterior probability $p(x_t, m_{0:n} | z_{0:k}, u_{0:t})$

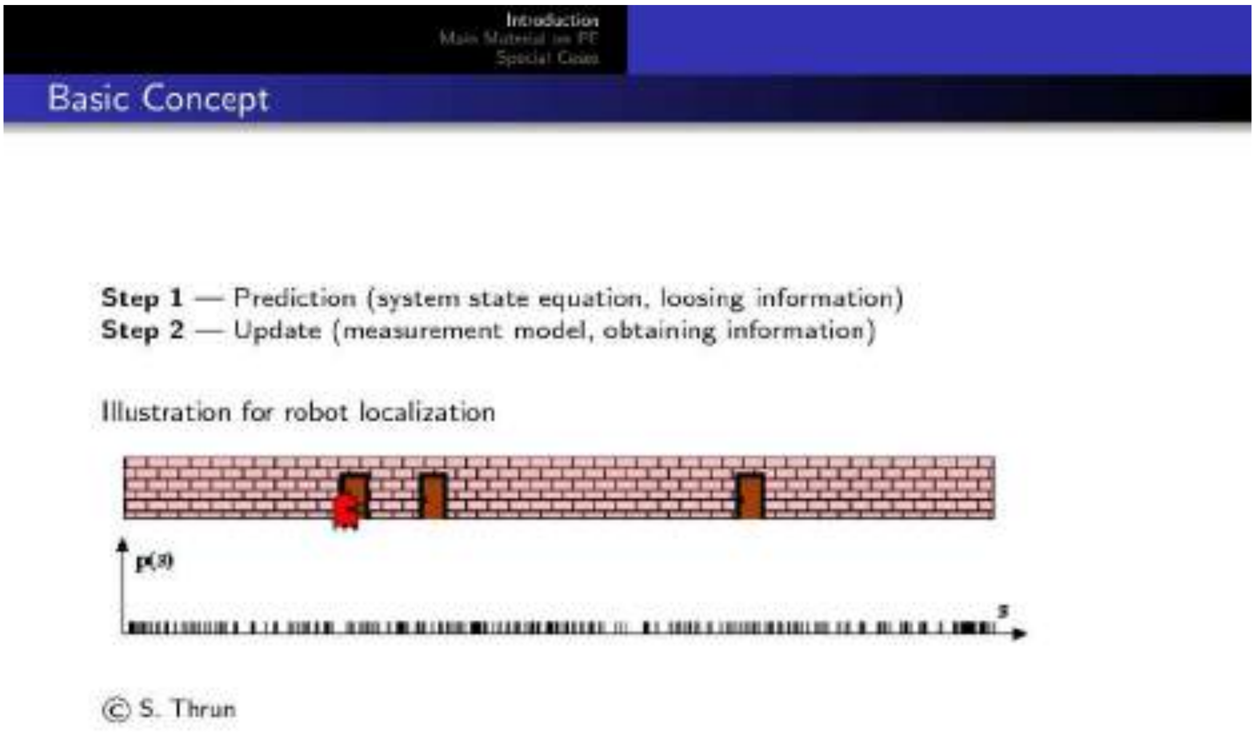
Particle Filters



Basic Concept

Goal is to estimate (hidden) state recursively from noisy measurements and state update rule
Principles

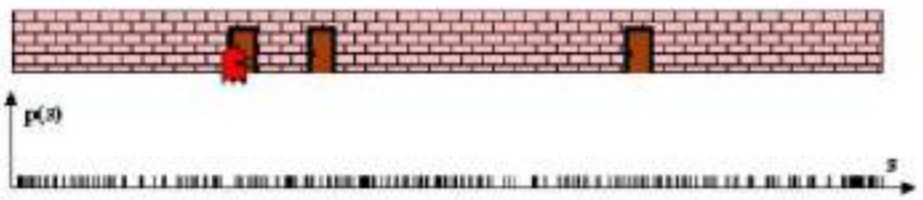
- recursive computation of probability distributions
- continuous distributions are approximated by discrete random measures
- measures are composed of **particles** — samples of the unknown states
- particles are weighted with "probability masses" computed by using Bayes theory



Basic Concept

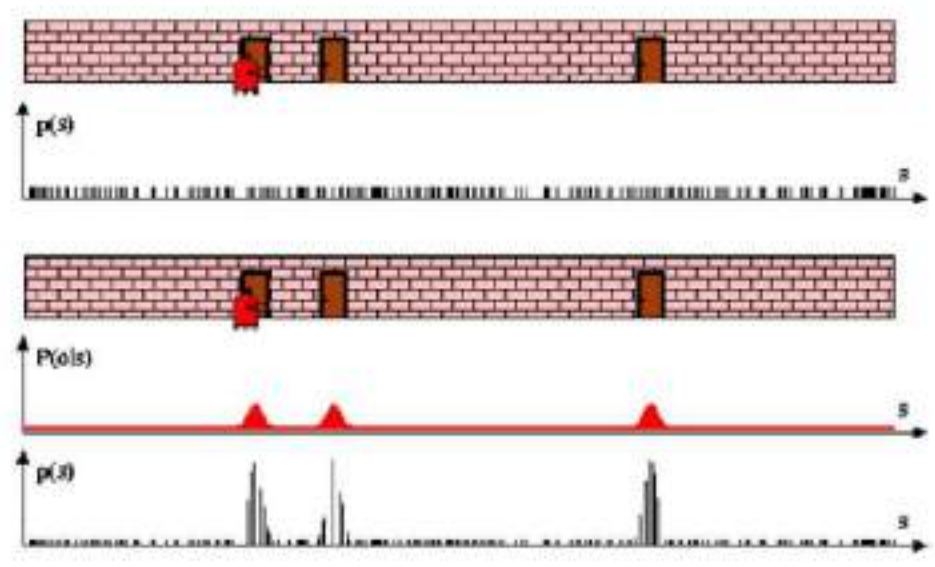
Step 1 — Prediction (system state equation, loosing information)
Step 2 — Update (measurement model, obtaining information)

Illustration for robot localization



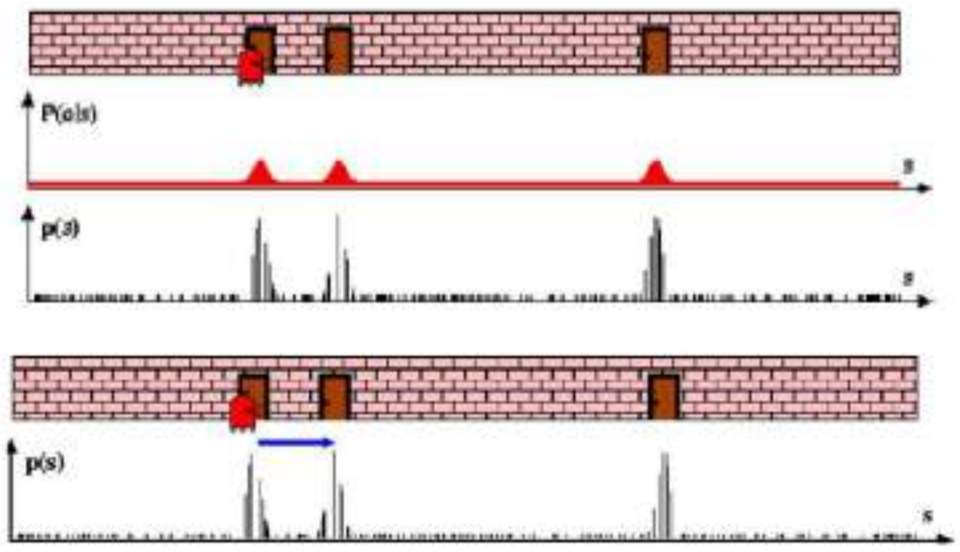
© S. Thrun

Basic Concept

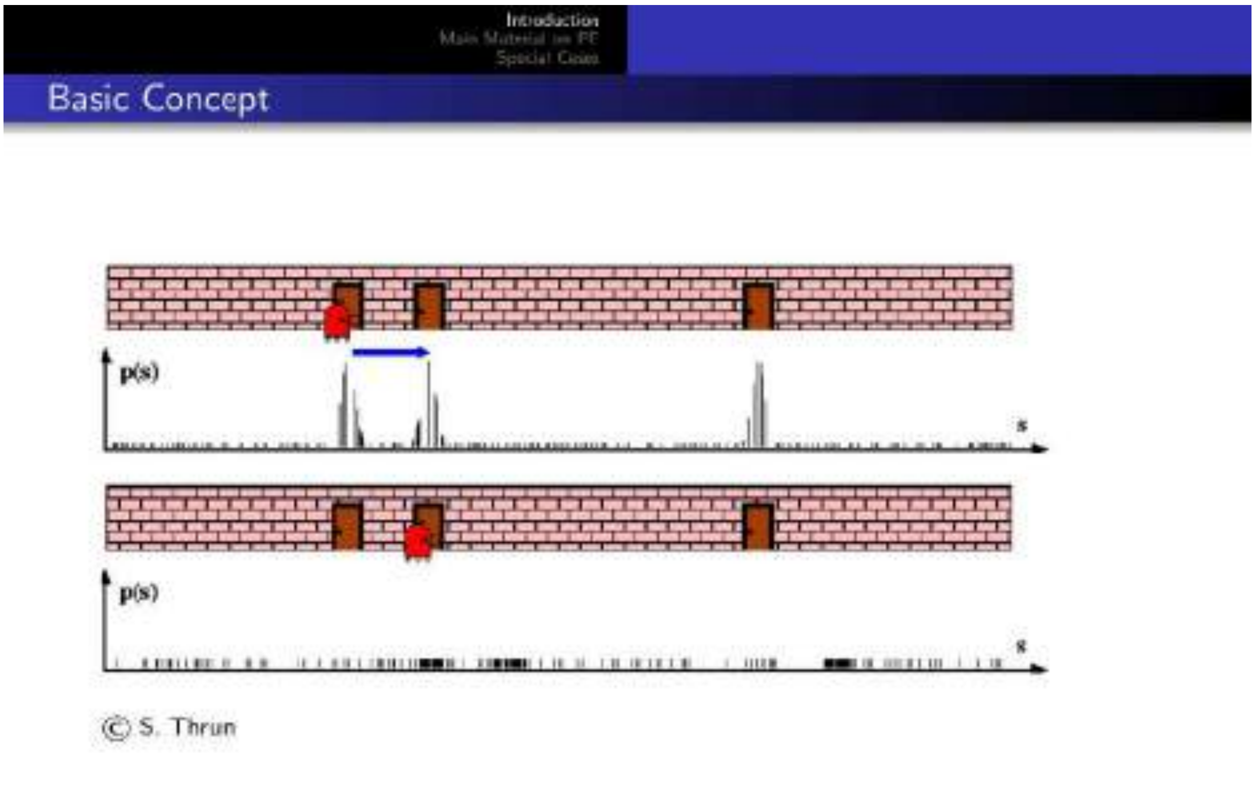


© S. Thrun

Basic Concept

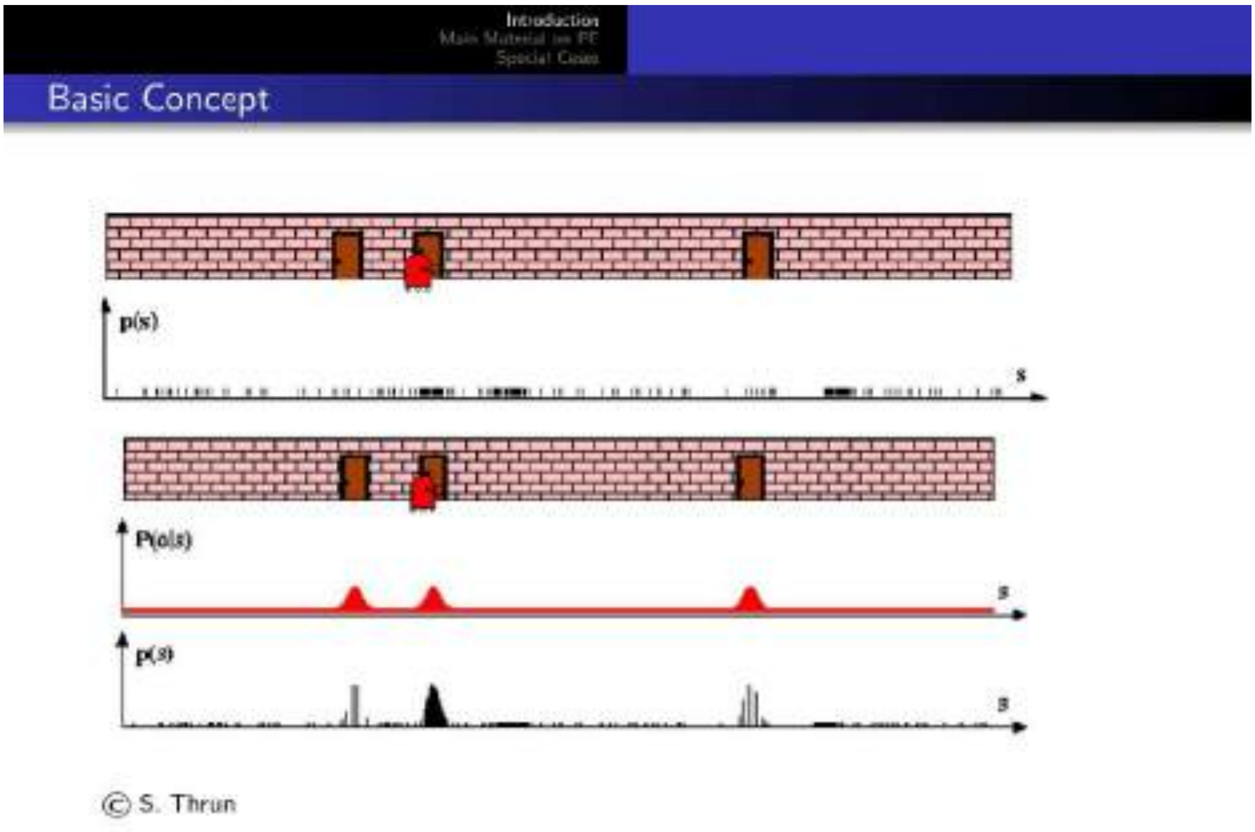


© S. Thrun



Basic Concept

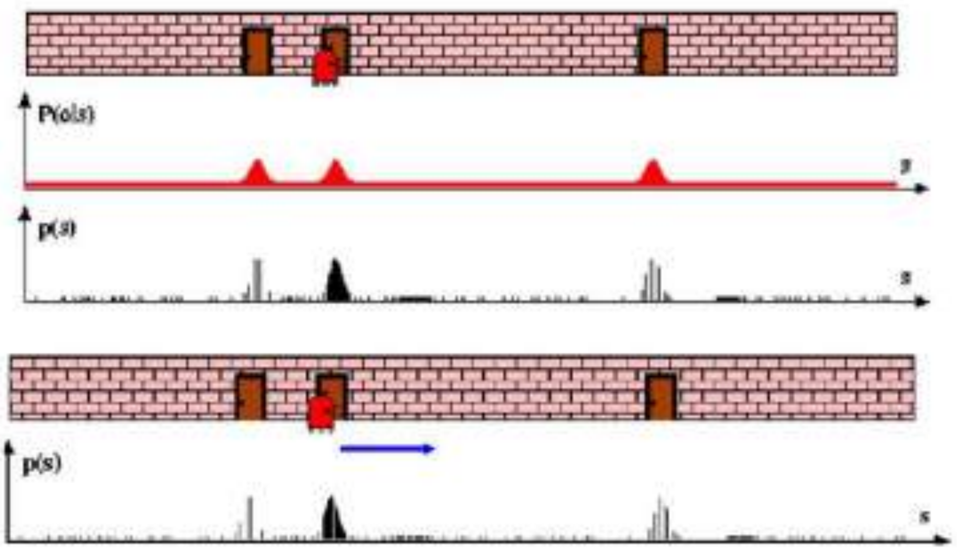
© S. Thrun



Basic Concept

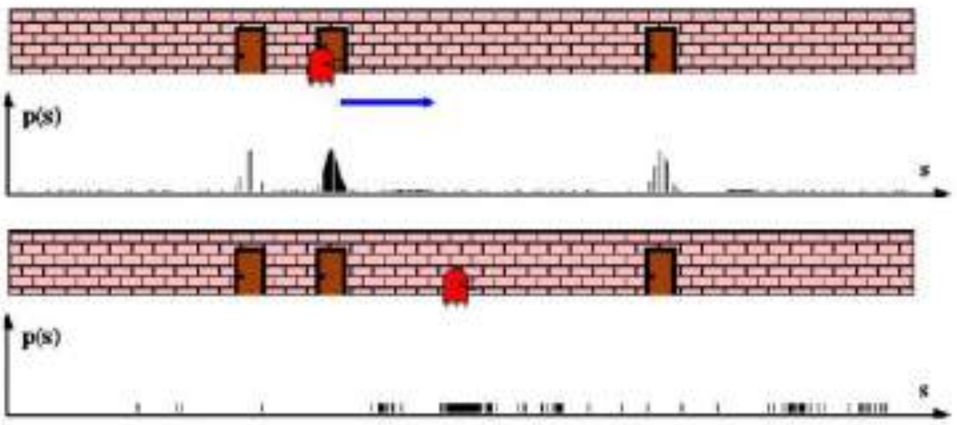
© S. Thrun

Basic Concept

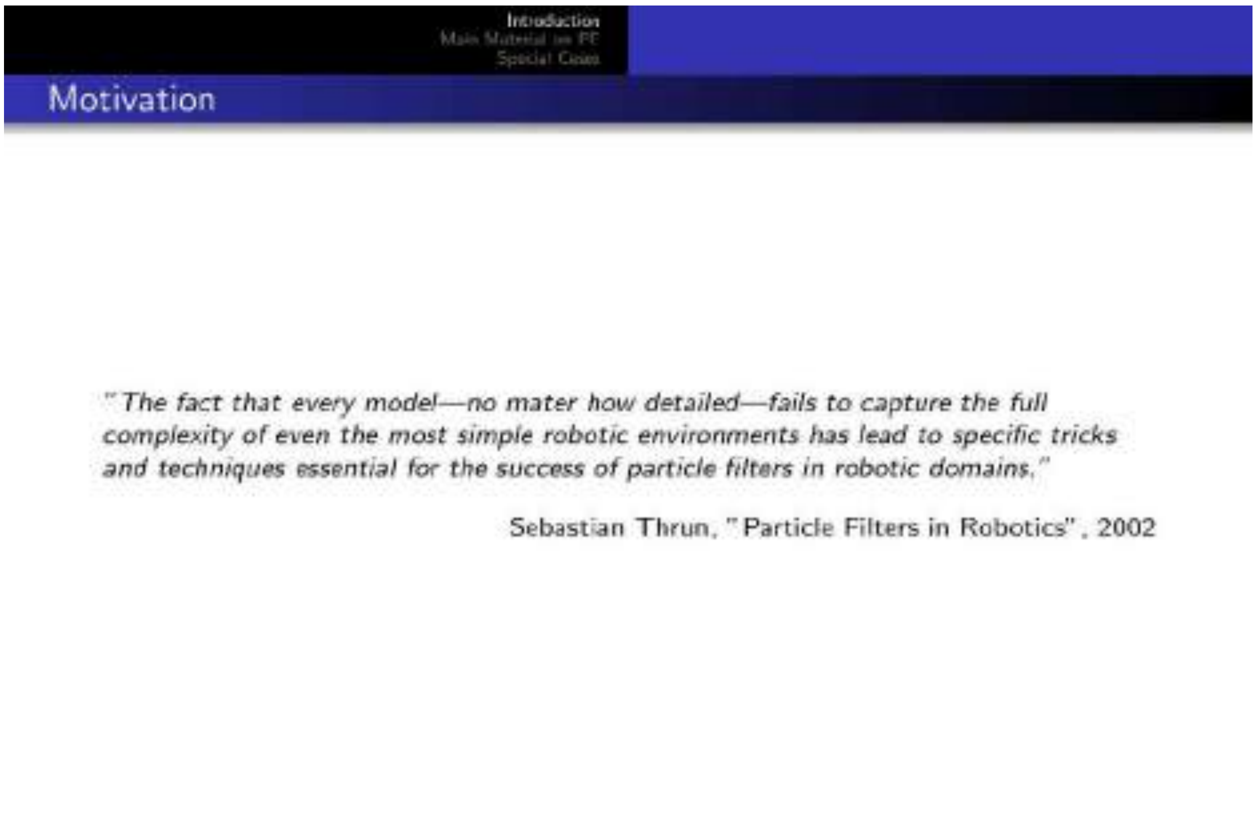


© S. Thrun

Basic Concept



© S. Thrun

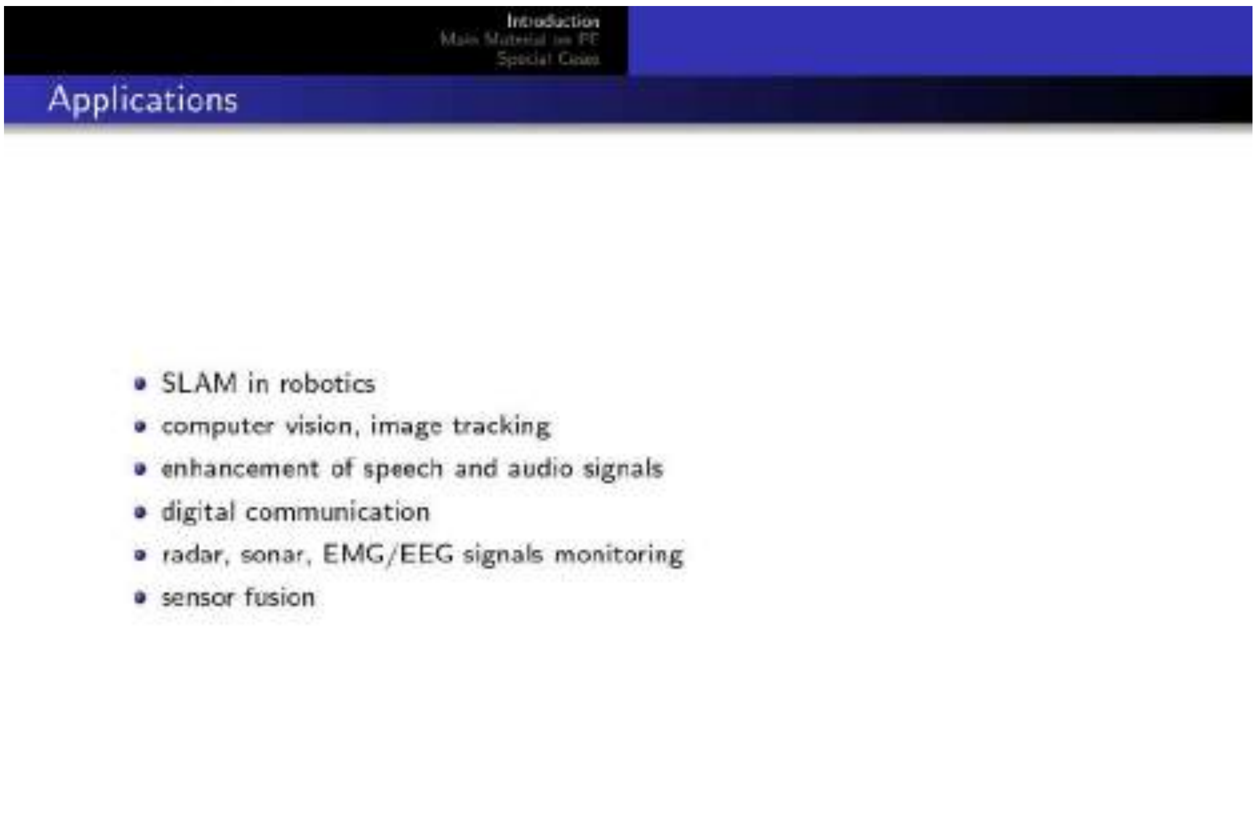


Introduction
Main Material on PF
Special Cases

Motivation

"The fact that every model—no mater how detailed—fails to capture the full complexity of even the most simple robotic environments has lead to specific tricks and techniques essential for the success of particle filters in robotic domains."

Sebastian Thrun, "Particle Filters in Robotics", 2002



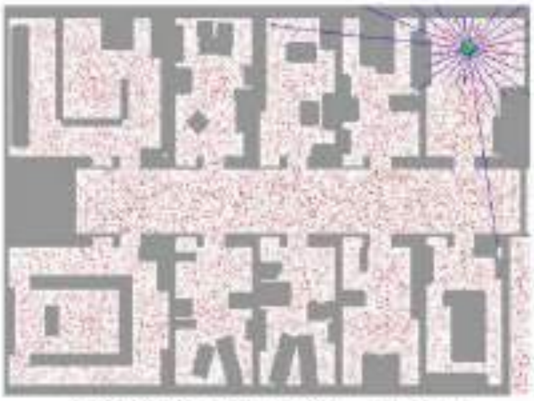
Introduction
Main Material on PF
Special Cases

Applications

- SLAM in robotics
- computer vision, image tracking
- enhancement of speech and audio signals
- digital communication
- radar, sonar, EMG/EEG signals monitoring
- sensor fusion

Applications

Localization in 2D using PF



© D. Fox, University Washington Robotics State Estimation Lab

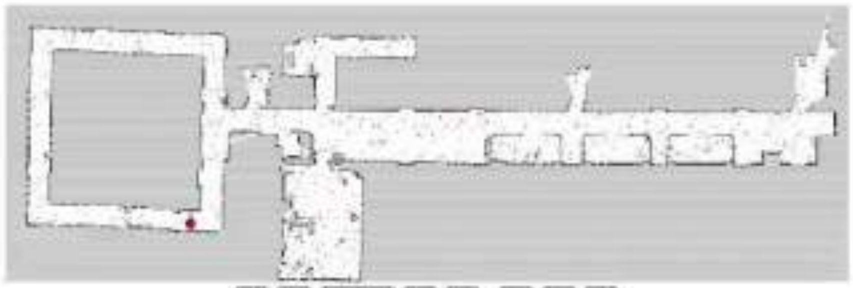
Applications

PF with adaptively adjusting particle set size

© D. Fox, University Washington Robotics State Estimation Lab

Applications

Real-time implementation of PF (mixtures of sample sets)



© D. Fox, University Washington Robotics State Estimation Lab

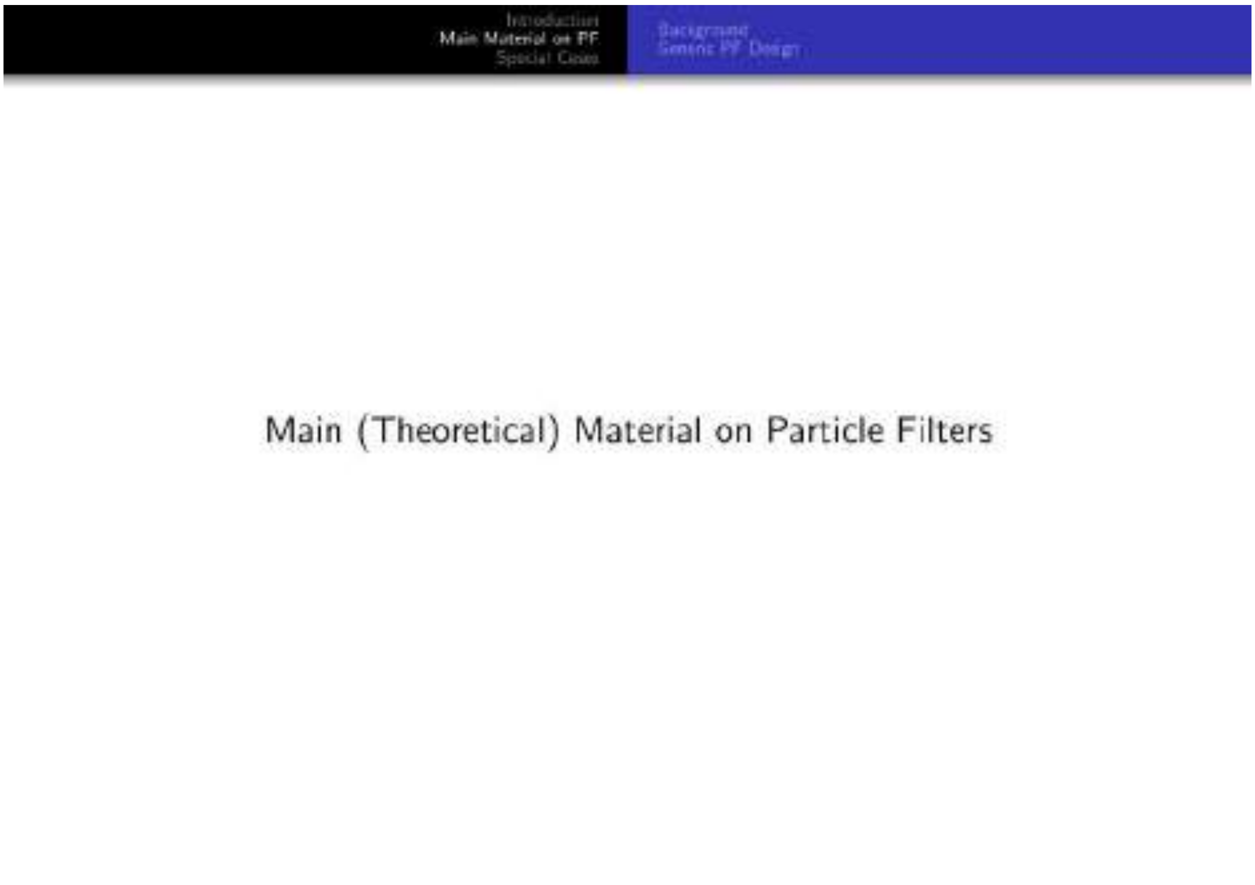
Features

Cons

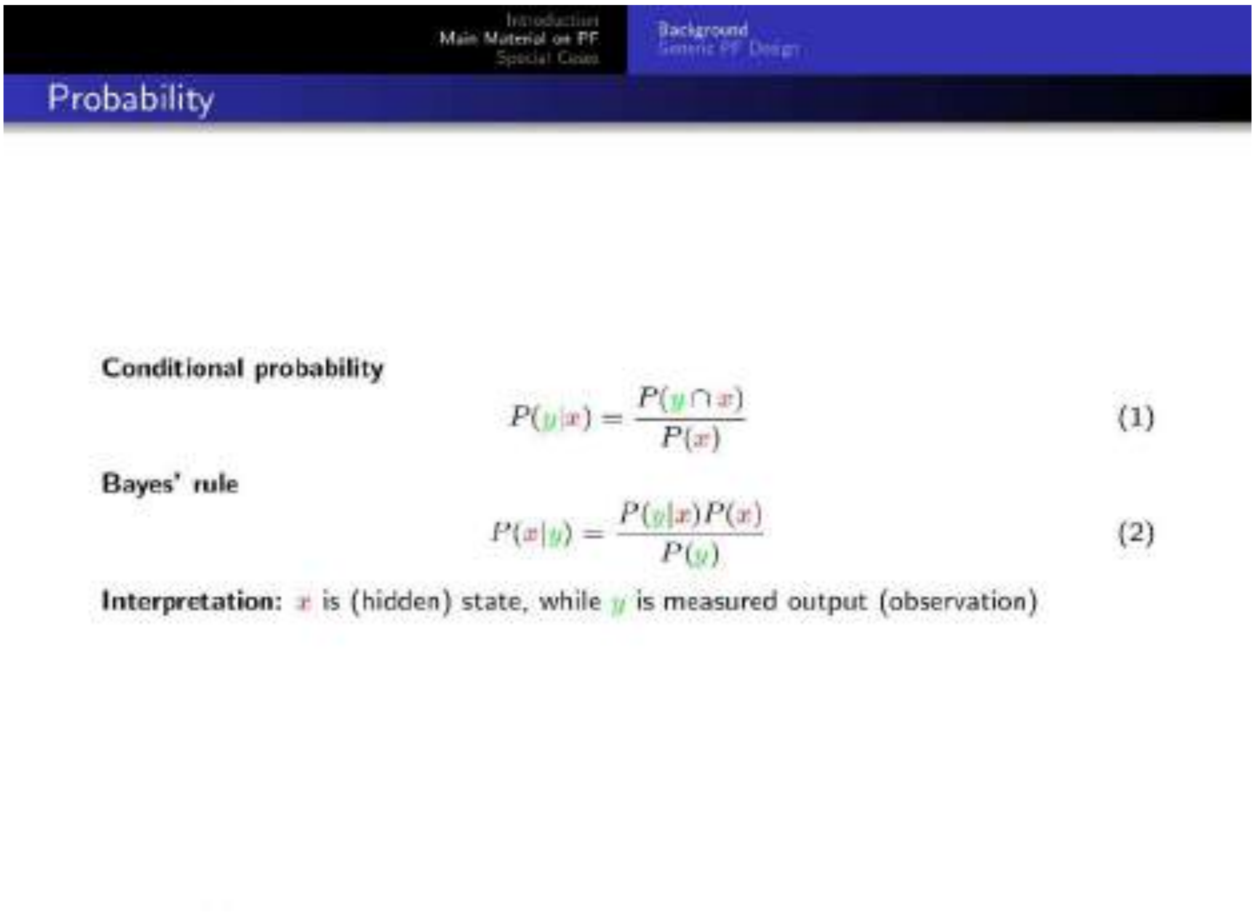
- approximate, probabilistic
- high computational complexity

Pros

- can cope with nonlinear, non-Gaussian problems
- easy to program
- potential for parallel implementation



Main (Theoretical) Material on Particle Filters



Probability

Conditional probability

$$P(y|x) = \frac{P(y \cap x)}{P(x)} \tag{1}$$

Bayes' rule

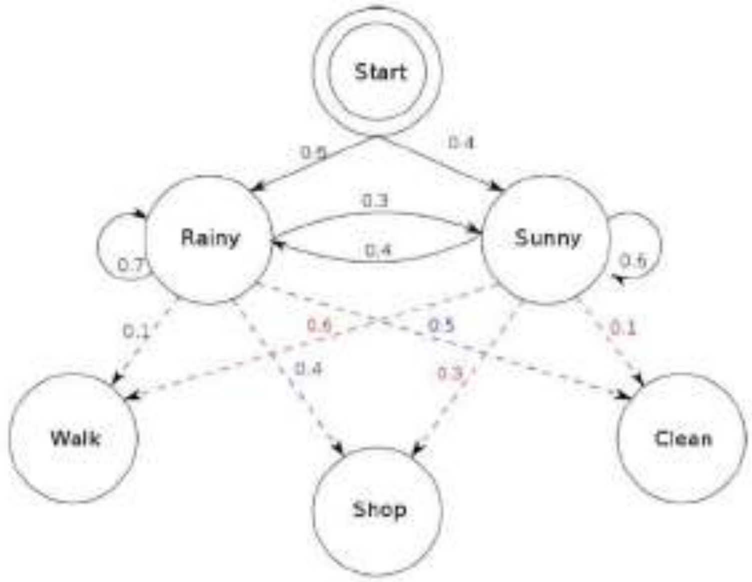
$$P(x|y) = \frac{P(y|x)P(x)}{P(y)} \tag{2}$$

Interpretation: x is (hidden) state, while y is measured output (observation)

ITMO UNIVERSITY

Model

Hidden Markov model

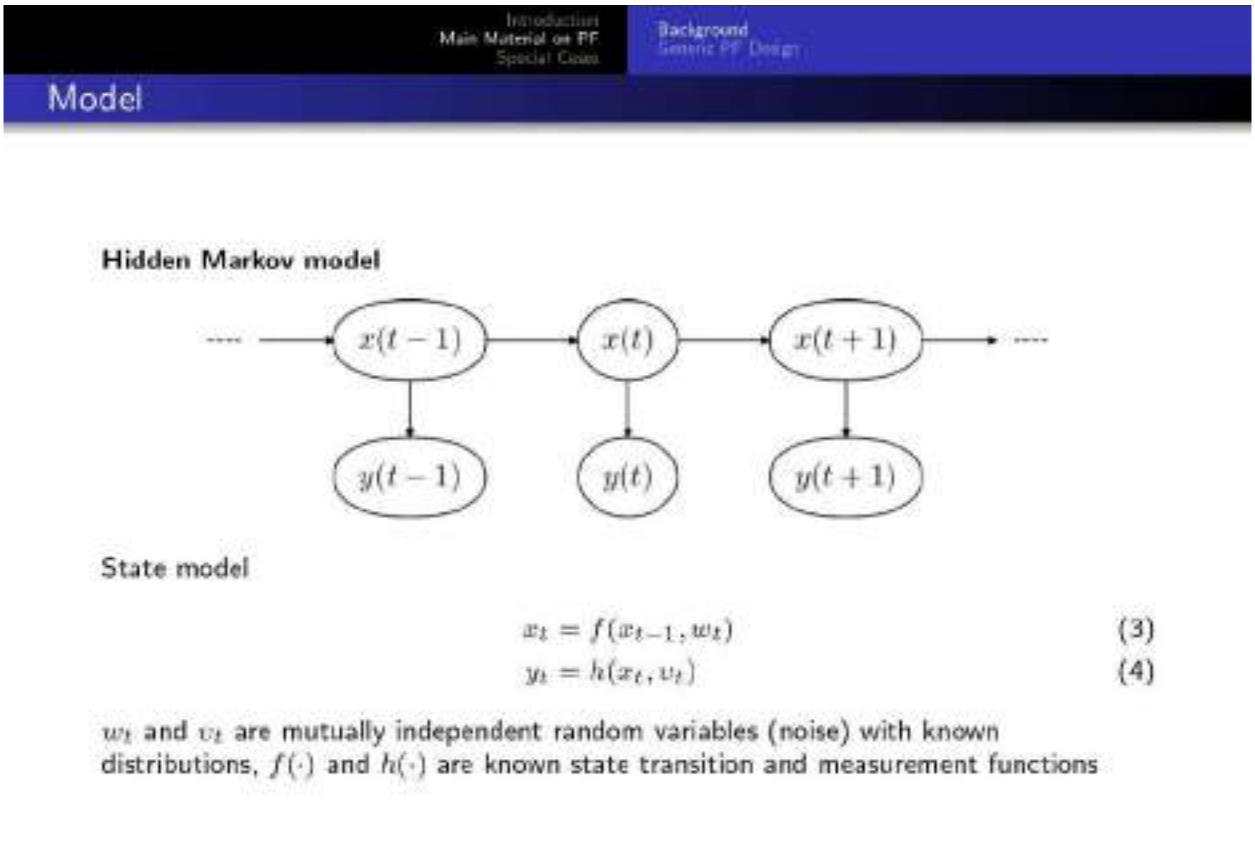


Model

Hidden Markov model

$$\begin{tikzcd}
\cdots \arrow[r] & x(t-1) \arrow[r] \arrow[d] & x(t) \arrow[r] \arrow[d] & x(t+1) \arrow[r] \arrow[d] & \cdots \\
 & y(t-1) & y(t) & y(t+1) &
\end{tikzcd}$$

$$x_t | x_{t-1} \sim P(x | x_{t-1}) \qquad (3)$$
$$y_t | x_t \sim P(y | x_t) \qquad (4)$$



Model

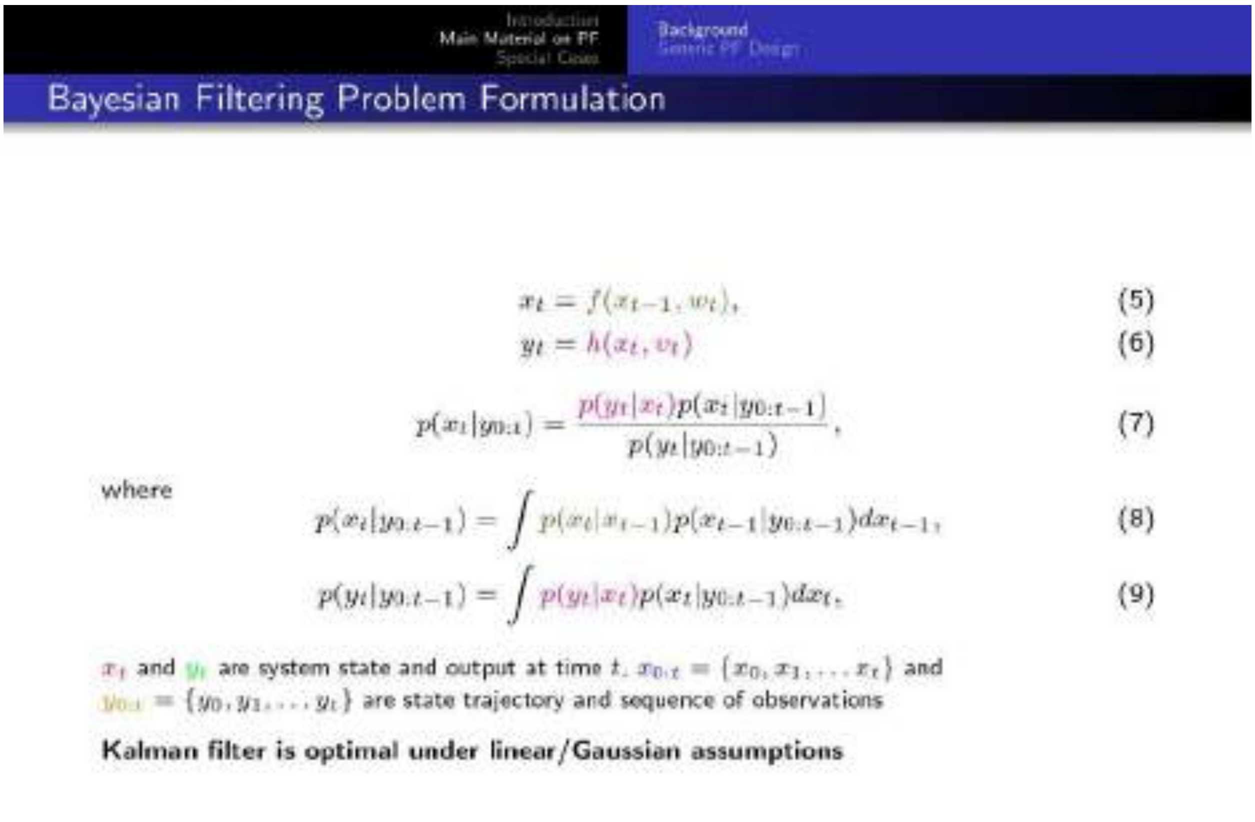
Hidden Markov model

$$\begin{tikzcd} \cdots \arrow[r] & x(t-1) \arrow[r] \arrow[d] & x(t) \arrow[r] \arrow[d] & x(t+1) \arrow[r] \arrow[d] & \cdots \\ & y(t-1) & y(t) & y(t+1) & \end{tikzcd}$$

State model

$$x_t = f(x_{t-1}, w_t) \quad (3)$$
$$y_t = h(x_t, v_t) \quad (4)$$

w_t and v_t are mutually independent random variables (noise) with known distributions, $f(\cdot)$ and $h(\cdot)$ are known state transition and measurement functions



Bayesian Filtering Problem Formulation

$$x_t = f(x_{t-1}, w_t), \quad (5)$$
$$y_t = h(x_t, v_t) \quad (6)$$

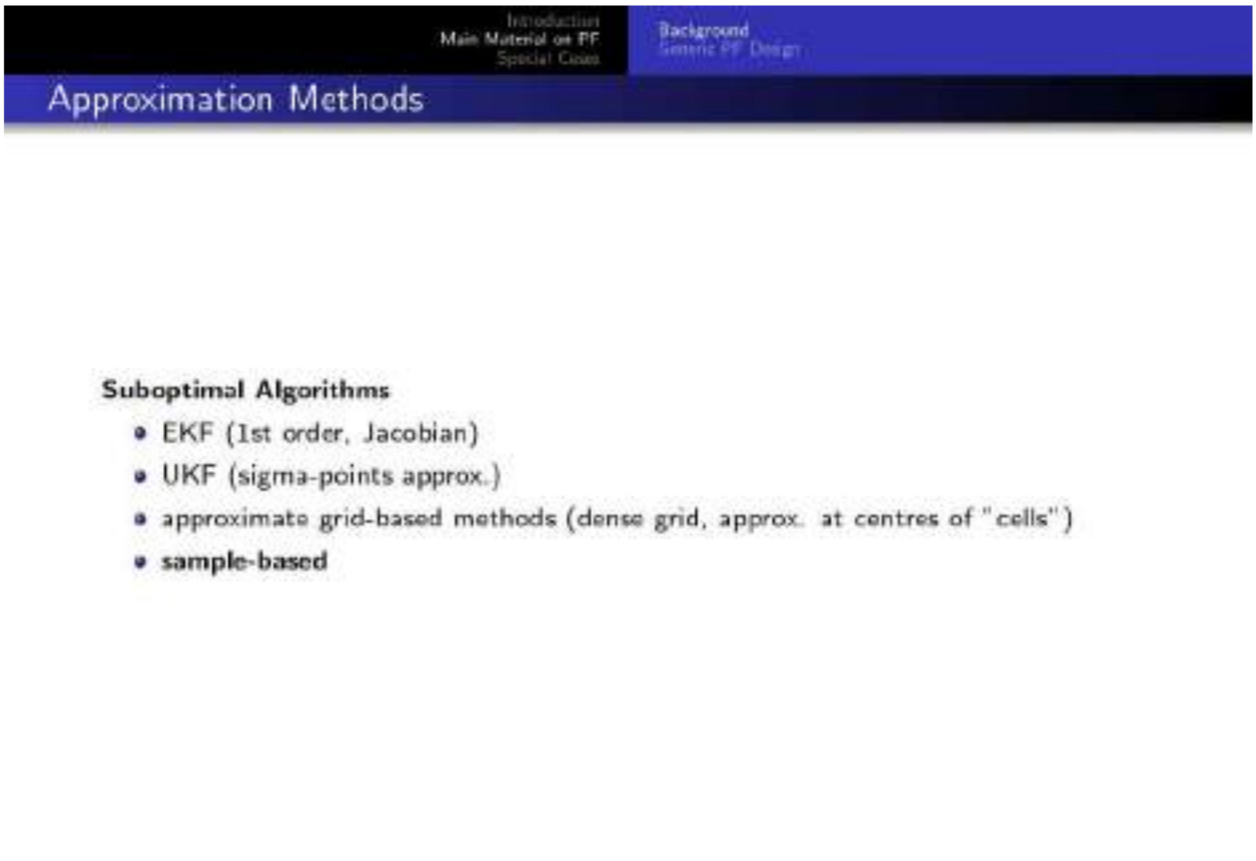
$$p(x_t|y_{0:t}) = \frac{p(y_t|x_t)p(x_t|y_{0:t-1})}{p(y_t|y_{0:t-1})}, \quad (7)$$

where

$$p(x_t|y_{0:t-1}) = \int p(x_t|x_{t-1})p(x_{t-1}|y_{0:t-1})dx_{t-1}, \quad (8)$$
$$p(y_t|y_{0:t-1}) = \int p(y_t|x_t)p(x_t|y_{0:t-1})dx_t, \quad (9)$$

x_t and y_t are system state and output at time t, $x_{0:t} = \{x_0, x_1, \ldots x_t\}$ and $y_{0:t} = \{y_0, y_1, \ldots, y_t\}$ are state trajectory and sequence of observations

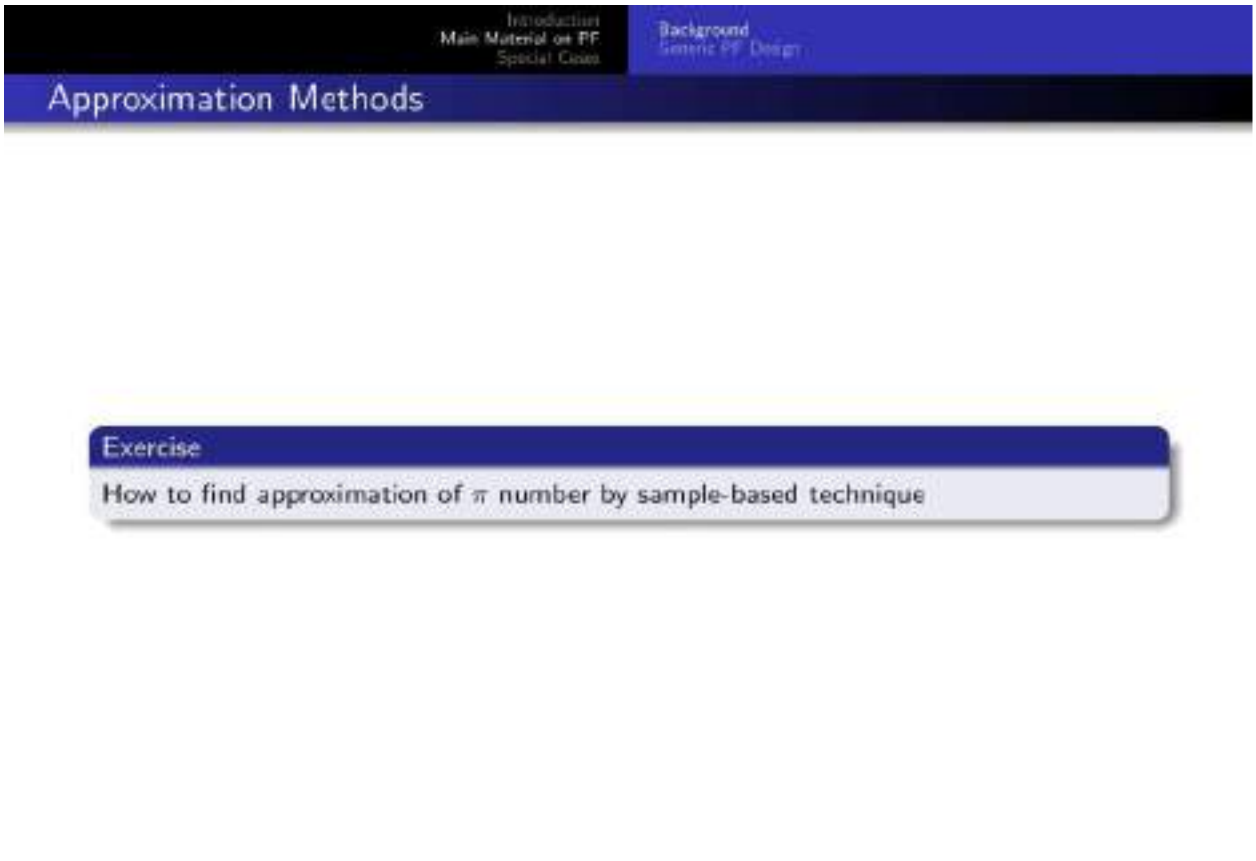
Kalman filter is optimal under linear/Gaussian assumptions



Approximation Methods

Suboptimal Algorithms

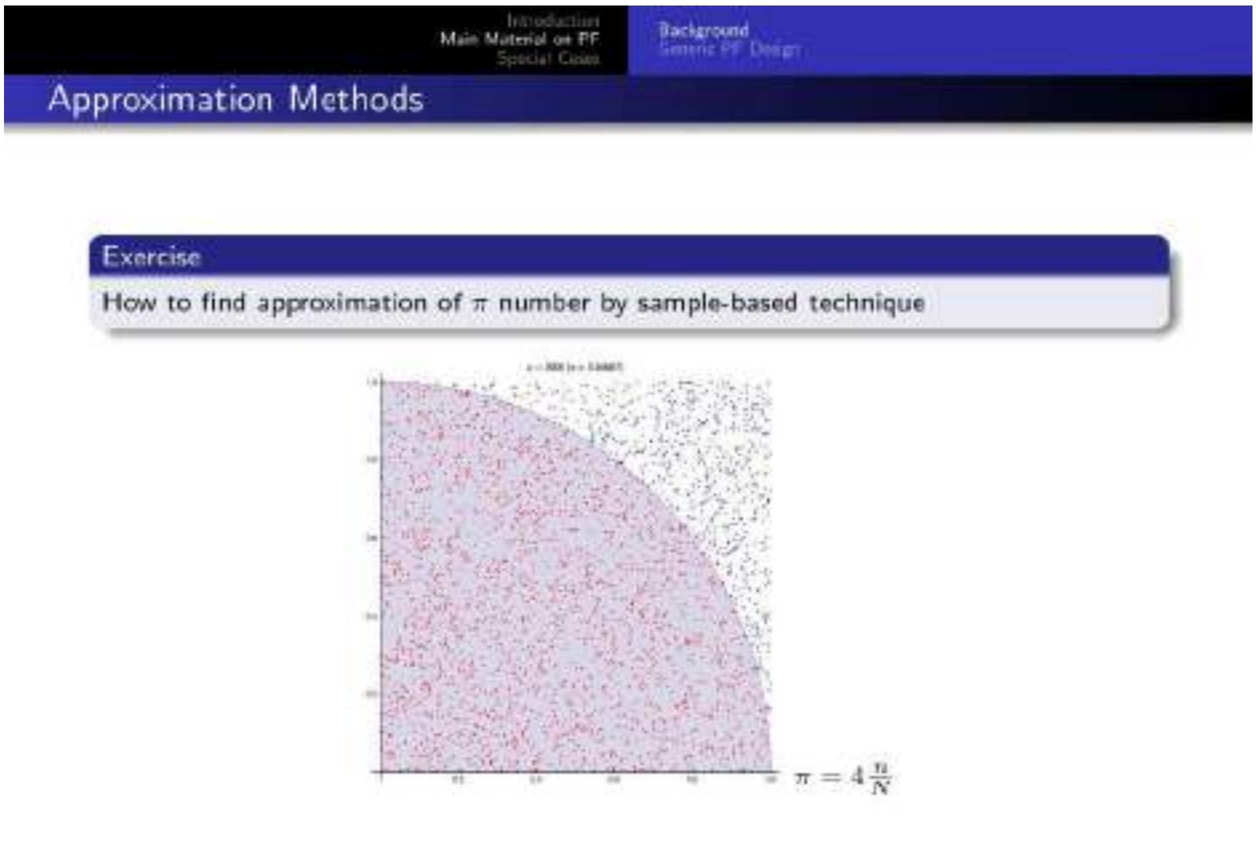
- EKF (1st order, Jacobian)
- UKF (sigma-points approx.)
- approximate grid-based methods (dense grid, approx. at centres of "cells")
- **sample-based**



Approximation Methods

Exercise

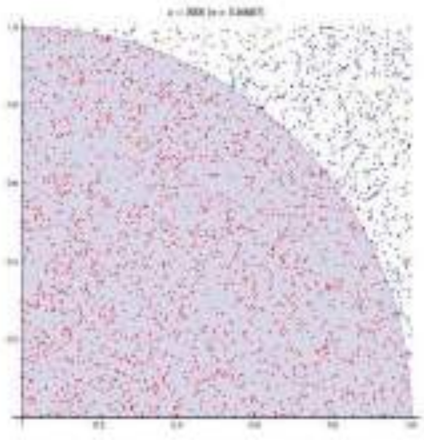
How to find approximation of π number by sample-based technique



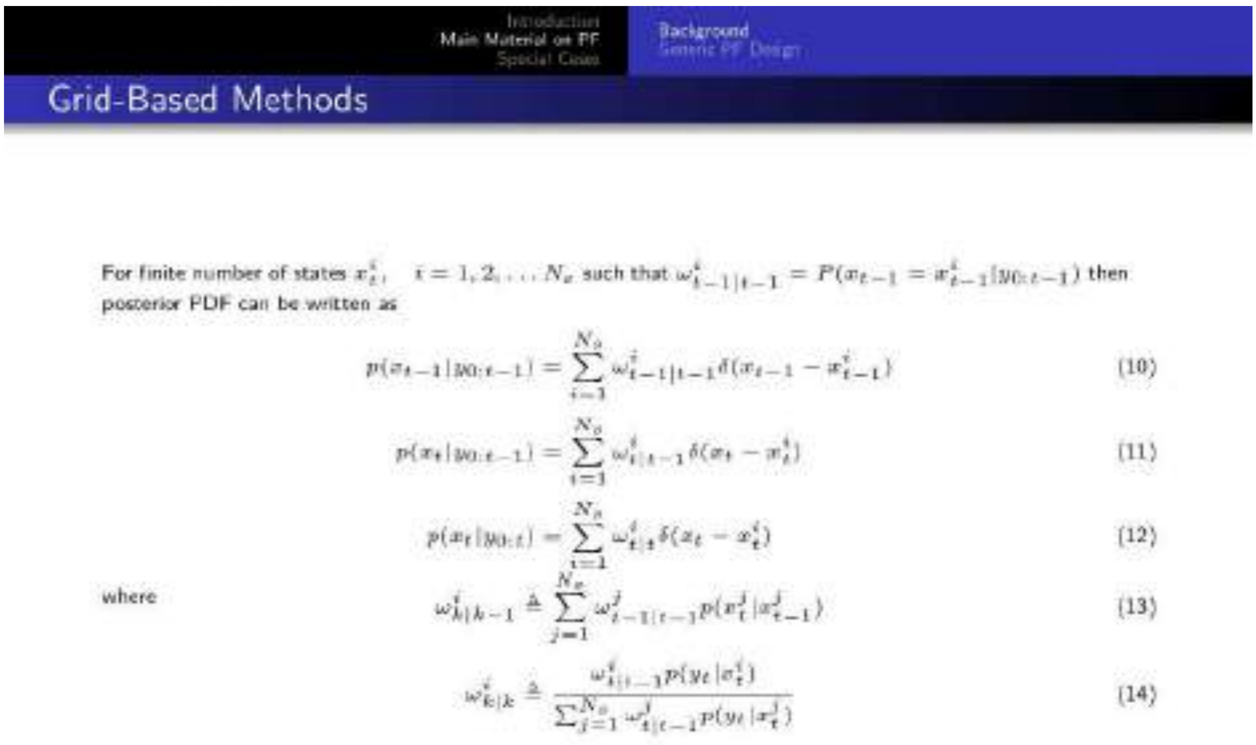
Approximation Methods

Exercise

How to find approximation of π number by sample-based technique



$\pi = 4\frac{n}{N}$



Grid-Based Methods

For finite number of states x^i_t, $i = 1, 2, \ldots N_s$ such that $\omega^i_{t-1|t-1} = P(x_{t-1} = x^i_{t-1}|y_{0:t-1})$ then posterior PDF can be written as

$$p(x_{t-1}|y_{0:t-1}) = \sum_{i=1}^{N_s} \omega^i_{t-1|t-1}\delta(x_{t-1} - x^i_{t-1}) \tag{10}$$

$$p(x_t|y_{0:t-1}) = \sum_{i=1}^{N_s} \omega^i_{t|t-1}\delta(x_t - x^i_t) \tag{11}$$

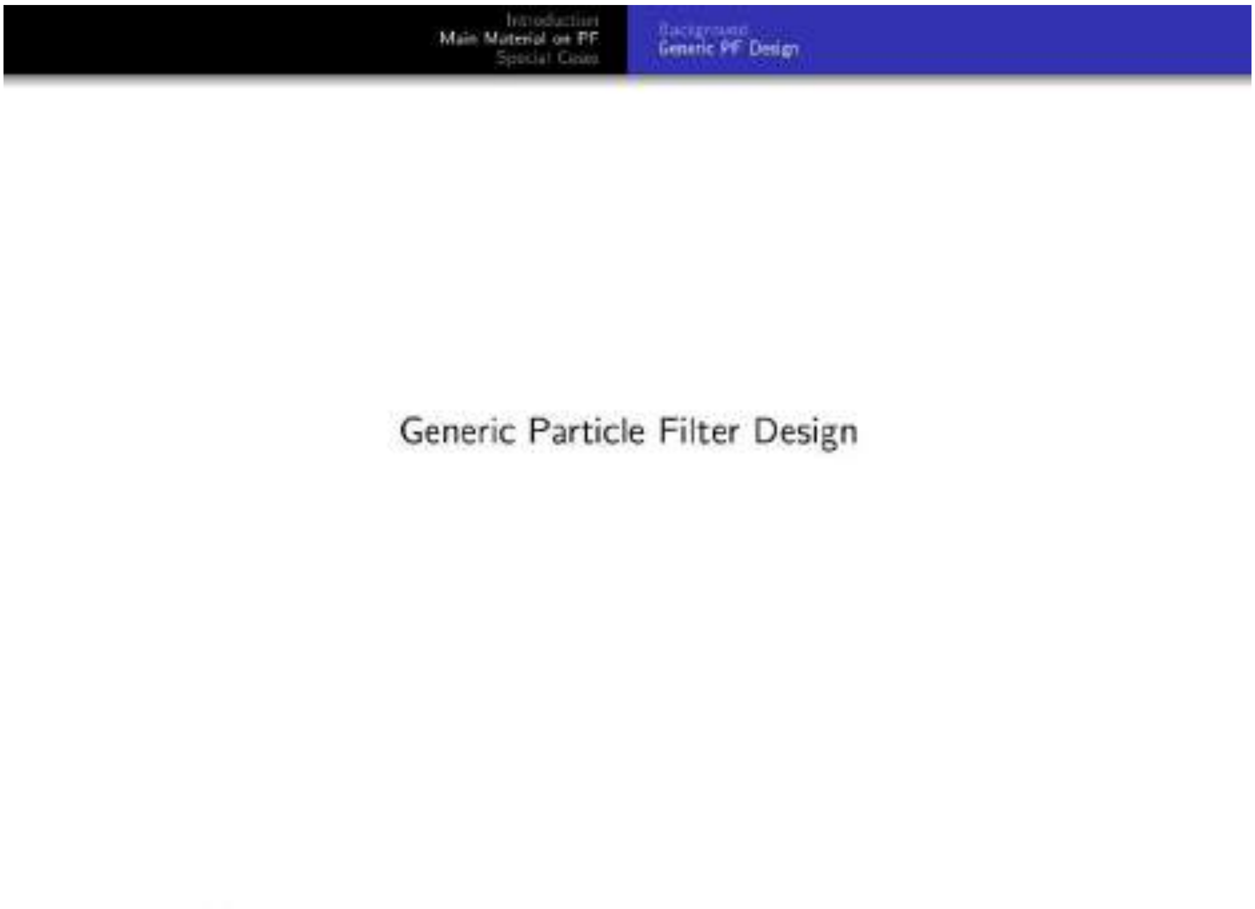
$$p(x_t|y_{0:t}) = \sum_{i=1}^{N_s} \omega^i_{t|t}\delta(x_t - x^i_t) \tag{12}$$

where

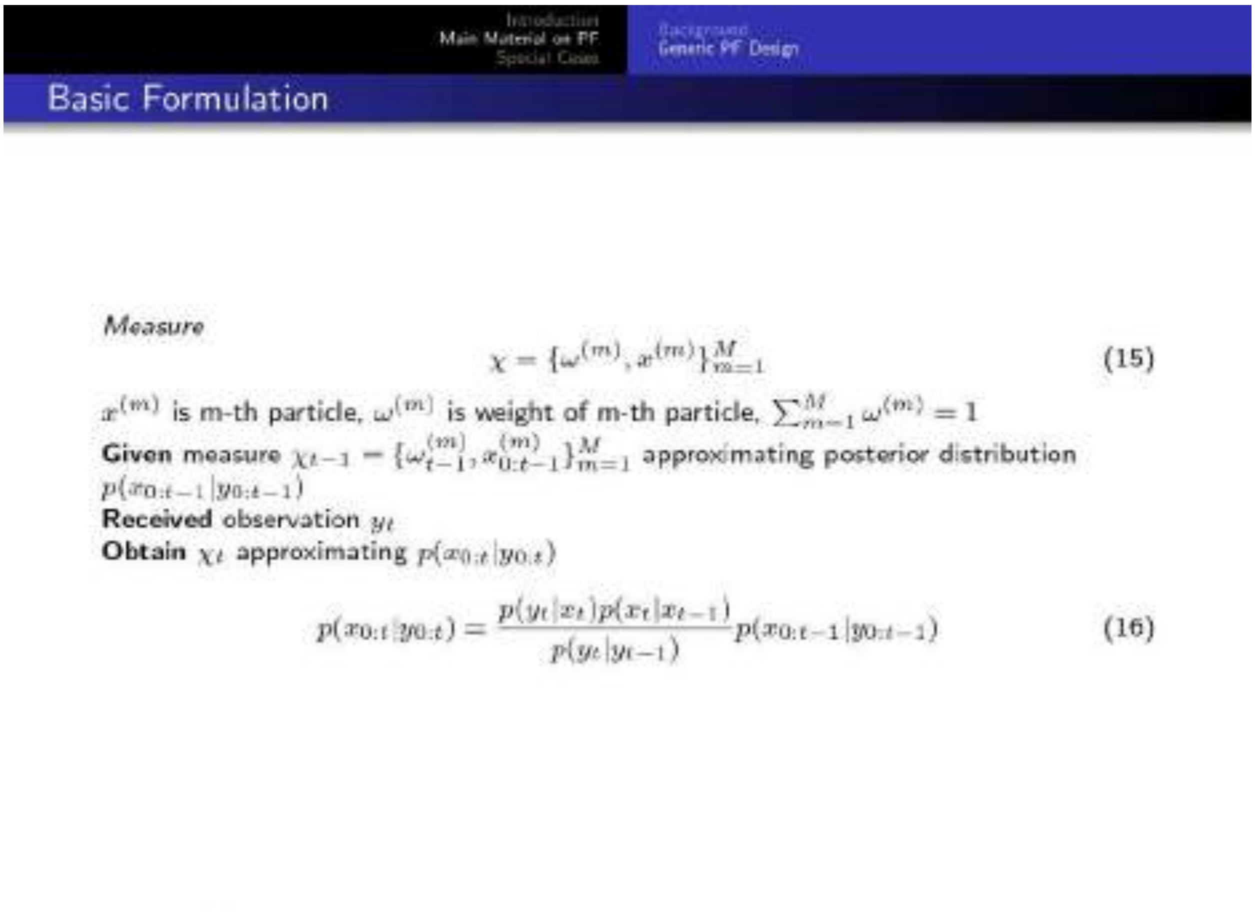
$$\omega^i_{k|k-1} \triangleq \sum_{j=1}^{N_s} \omega^j_{t-1|t-1} p(x^i_t|x^j_{t-1}) \tag{13}$$

$$\omega^i_{k|k} \triangleq \frac{\omega^i_{t|t-1} p(y_t|x^i_t)}{\sum_{j=1}^{N_s} \omega^j_{t|t-1} p(y_t|x^j_t)} \tag{14}$$

ITMO UNIVERSITY



Generic Particle Filter Design



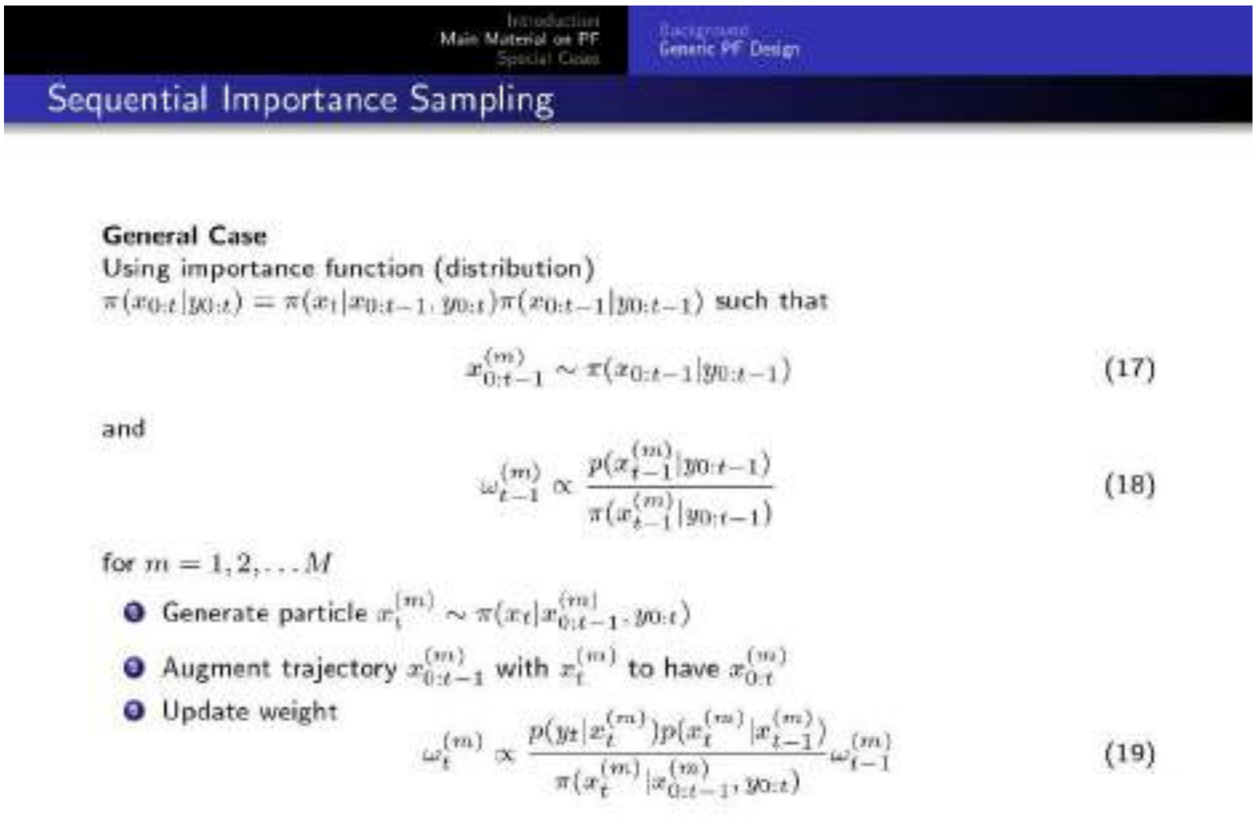
Basic Formulation

Measure

$$\chi = \{\omega^{(m)}, x^{(m)}\}_{m=1}^{M} \tag{15}$$

$x^{(m)}$ is m-th particle, $\omega^{(m)}$ is weight of m-th particle, $\sum_{m=1}^{M} \omega^{(m)} = 1$
Given measure $\chi_{t-1} = \{\omega_{t-1}^{(m)}, x_{0:t-1}^{(m)}\}_{m=1}^{M}$ approximating posterior distribution $p(x_{0:t-1}|y_{0:t-1})$
Received observation y_t
Obtain χ_t approximating $p(x_{0:t}|y_{0:t})$

$$p(x_{0:t}|y_{0:t}) = \frac{p(y_t|x_t)p(x_t|x_{t-1})}{p(y_t|y_{t-1})} p(x_{0:t-1}|y_{0:t-1}) \tag{16}$$



Sequential Importance Sampling

General Case
Using importance function (distribution)
$\pi(x_{0:t}|y_{0:t}) = \pi(x_t|x_{0:t-1}, y_{0:t})\pi(x_{0:t-1}|y_{0:t-1})$ such that

$$x_{0:t-1}^{(m)} \sim \pi(x_{0:t-1}|y_{0:t-1}) \tag{17}$$

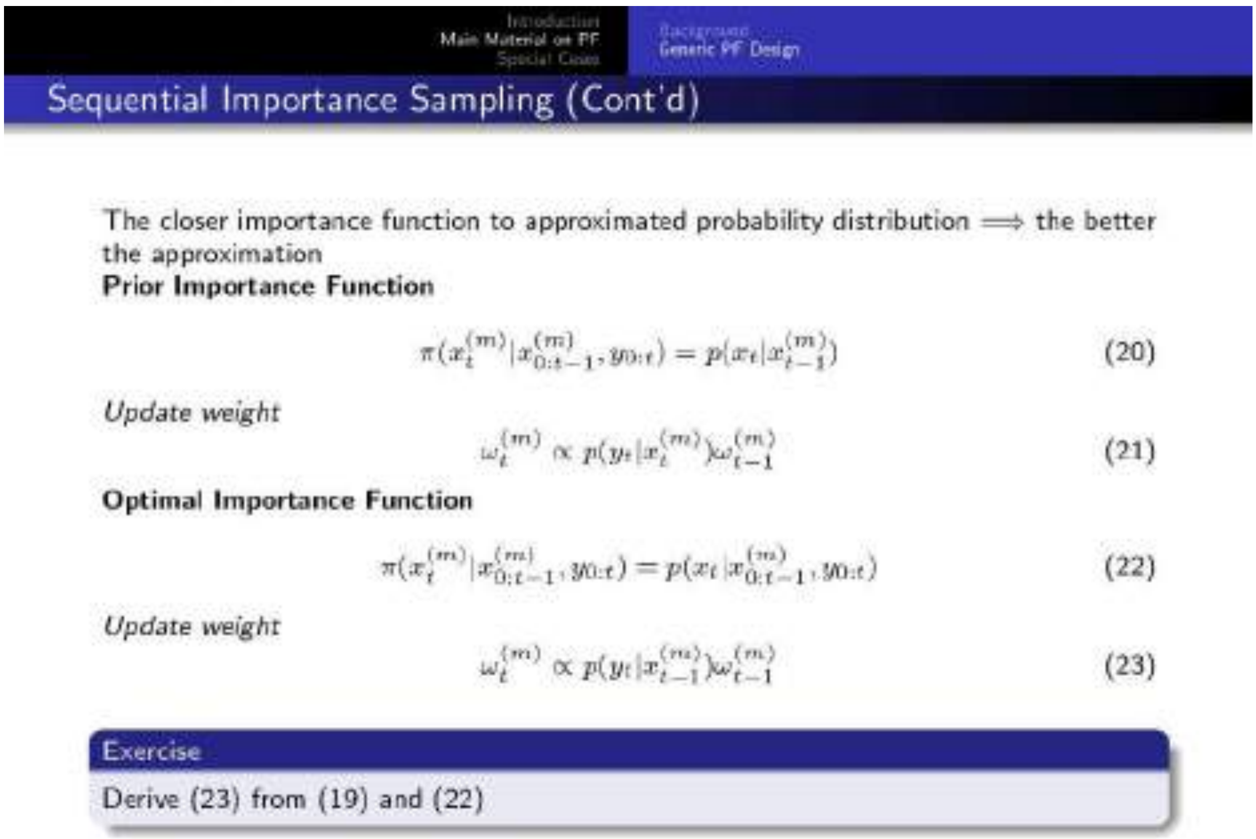
and

$$\omega_{t-1}^{(m)} \propto \frac{p(x_{t-1}^{(m)}|y_{0:t-1})}{\pi(x_{t-1}^{(m)}|y_{0:t-1})} \tag{18}$$

for $m = 1, 2, \dots M$

1. Generate particle $x_t^{(m)} \sim \pi(x_t|x_{0:t-1}^{(m)}, y_{0:t})$
2. Augment trajectory $x_{0:t-1}^{(m)}$ with $x_t^{(m)}$ to have $x_{0:t}^{(m)}$
3. Update weight

$$\omega_t^{(m)} \propto \frac{p(y_t|x_t^{(m)})p(x_t^{(m)}|x_{t-1}^{(m)})}{\pi(x_t^{(m)}|x_{0:t-1}^{(m)}, y_{0:t})}\omega_{t-1}^{(m)} \tag{19}$$



Sequential Importance Sampling (Cont'd)

The closer importance function to approximated probability distribution $\Longrightarrow$ the better the approximation
Prior Importance Function

$$\pi(x_t^{(m)}|x_{0:t-1}^{(m)}, y_{0:t}) = p(x_t|x_{t-1}^{(m)}) \tag{20}$$

Update weight

$$\omega_t^{(m)} \propto p(y_t|x_t^{(m)})\omega_{t-1}^{(m)} \tag{21}$$

Optimal Importance Function

$$\pi(x_t^{(m)}|x_{0:t-1}^{(m)}, y_{0:t}) = p(x_t|x_{0:t-1}^{(m)}, y_{0:t}) \tag{22}$$

Update weight

$$\omega_t^{(m)} \propto p(y_t|x_{t-1}^{(m)})\omega_{t-1}^{(m)} \tag{23}$$

Exercise
Derive (23) from (19) and (22)

Resampling

Degeneration problem

After several steps most particles will have negligible weights $\Longrightarrow$ filter degenerates, i.e. its performance deteriorates

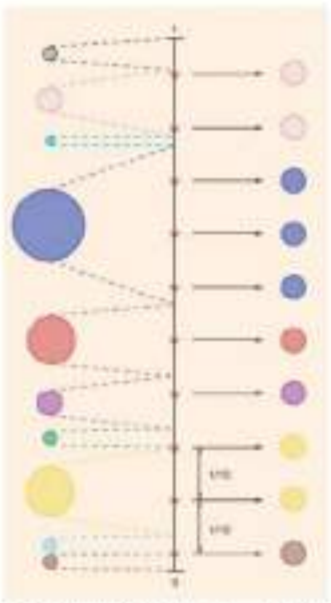
Effective sample size

$$M_E = \frac{M}{1 + Var(\omega_k^{*(m)})} \approx \frac{1}{\sum_{m=1}^{M} (\omega_k^{(m)})^2} \tag{24}$$

Resampling solution

Resampling is a scheme that eliminates particles with small weights and replicates particles with large weights in probabilistic manner

Resampling (Cont'd)



© IEEE (Djuric et. al, 2003)

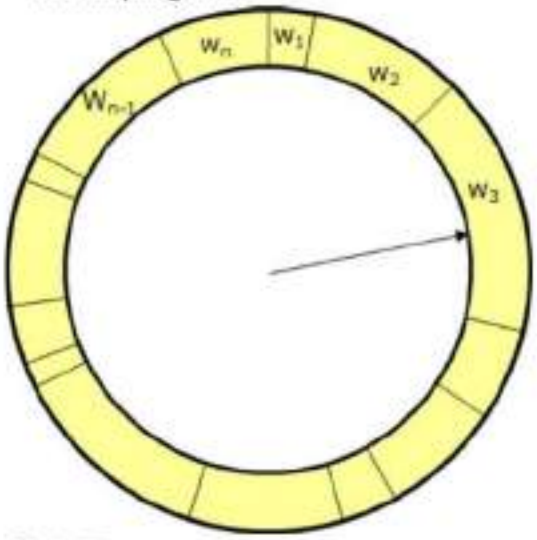
Systematic resampling

Pseudo code

```
$c_0 = 0$
for $m = 1 \ldots M$
    $c_m = c_{m-1} + \omega_t^{(m)}$
end for
$m = 1$
$u_1 = U[0, 1/M]$
for $j = 1 \ldots M$
    $u_j = u_1 + \frac{j-1}{M}$
    while $u_j > c_m$
        $m = m + 1$
    end while
    $x_t^{(j)} = x_t^{(m)}$
    $\omega_t^{(j)} = 1/M$
    $m^j = m$ #parent particle
end for
```

Resampling (Cont'd)

Resampling wheel



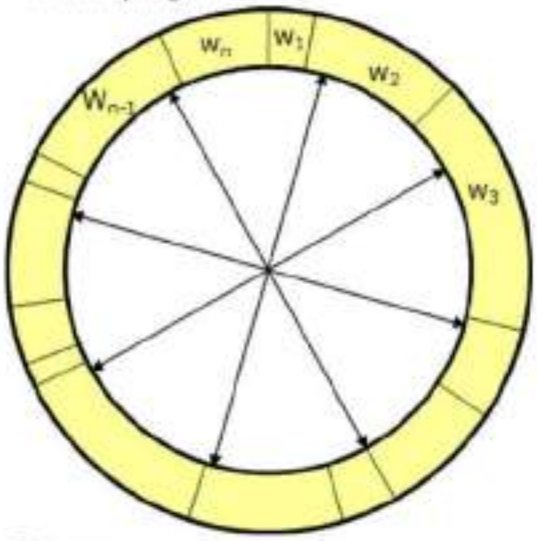
© S. Thrun

Pseudo code

```
$m = U[1 \dots M]$
$\beta = 0$
for $i = 1 \dots M$
    $\beta = \beta + U[0 \dots 2\omega^{max}]$
    while $\omega^{(m)} < \beta$
        $\beta = \beta - \omega^{(m)}$
        $m = m + 1$
    end while
    $x^{(i)} = x^{(m)}$
end for
```

Resampling (Cont'd)

Resampling wheel



© S. Thrun

Stochastic universal resampling

Resampling (Cont'd)

Exercise

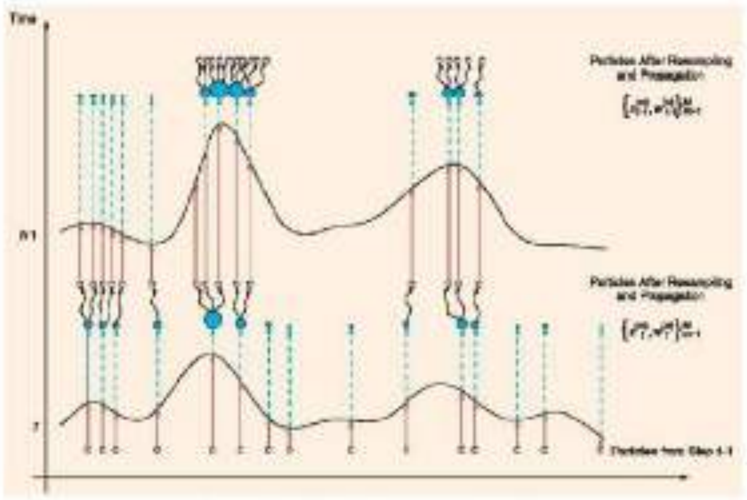
Given

$$
\begin{aligned}
\omega^{(1)} &= 0.5\\
\omega^{(2)} &= 1.2\\
\omega^{(3)} &= 1.5\\
\omega^{(4)} &= 1.0\\
\omega^{(5)} &= 0.8
\end{aligned}
$$

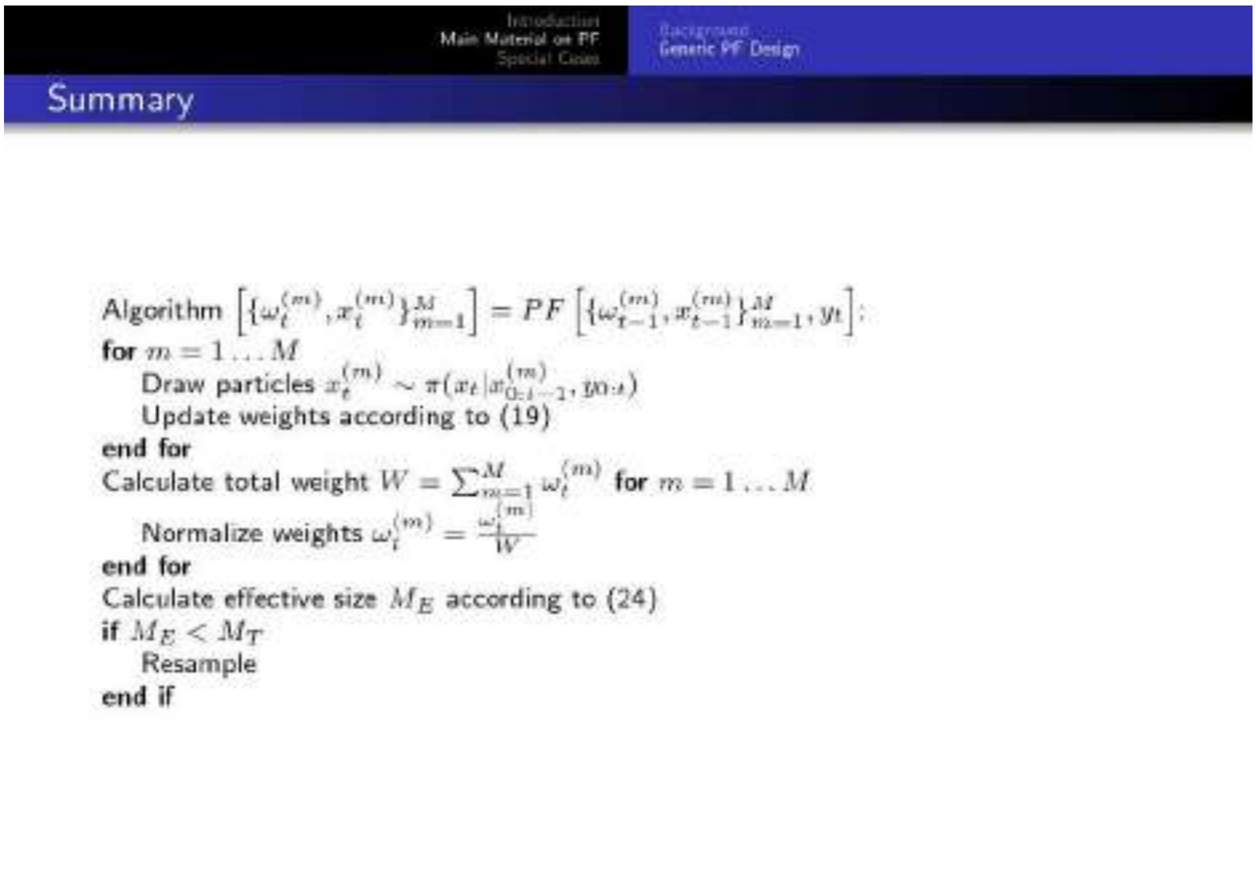
- Do we need resampling if $M_T = 0.2$?
- Calculate probabilities of **not picking** particles with the highest and lowest weights for $O(M)$ resampling algorithm

Summary

Particle evolution diagram

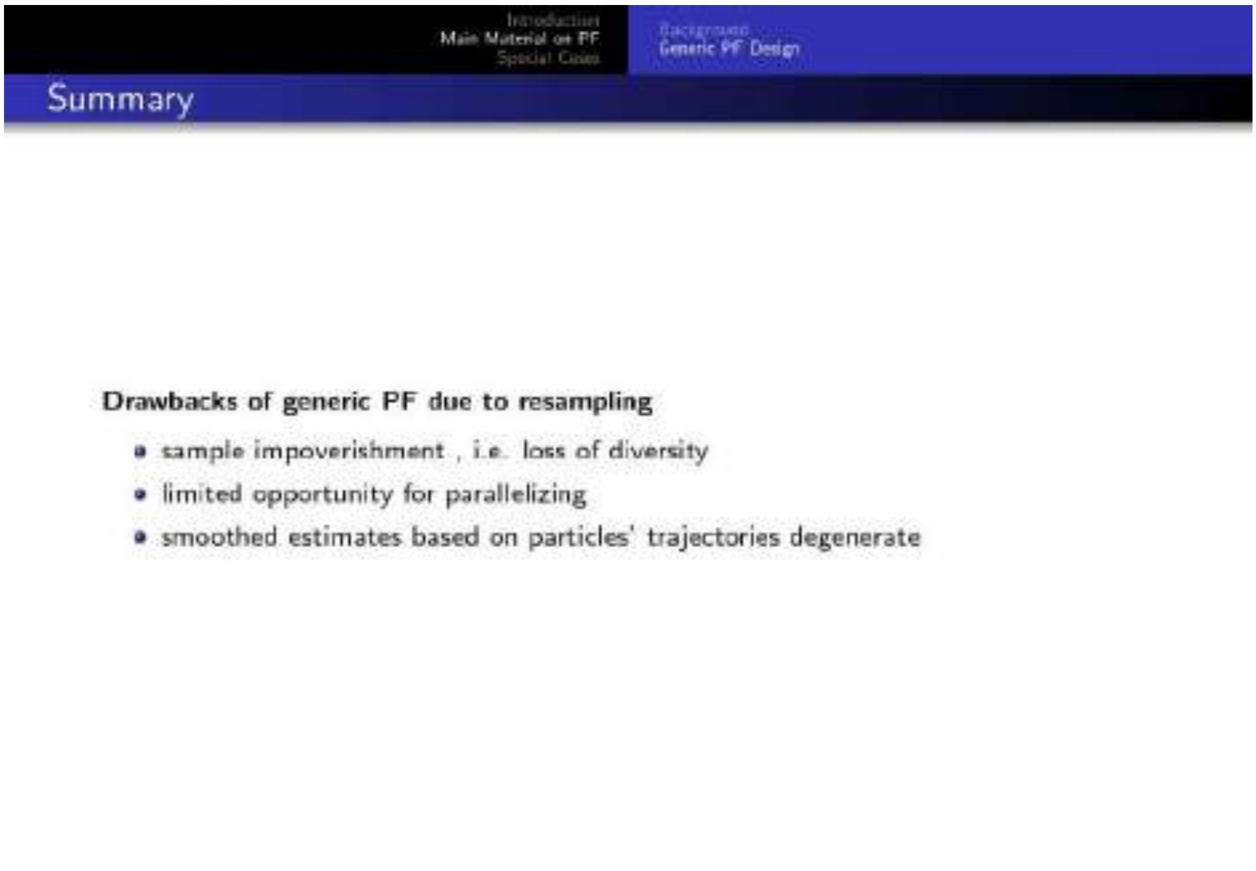

Time
t+1
t
Particles After Resampling and Propagation
Particles After Resampling and Propagation
Particles from Step t-1

© IEEE (Djuric et. al, 2003)



Summary

```
Algorithm $\left[\{\omega_t^{(m)}, x_t^{(m)}\}_{m=1}^{M}\right] = PF\left[\{\omega_{t-1}^{(m)}, x_{t-1}^{(m)}\}_{m=1}^{M}, y_t\right]$;
for $m = 1 \ldots M$
    Draw particles $x_t^{(m)} \sim \pi(x_t | x_{0:t-1}^{(m)}, y_{0:t})$
    Update weights according to (19)
end for
Calculate total weight $W = \sum_{m=1}^{M} \omega_t^{(m)}$ for $m = 1 \ldots M$
    Normalize weights $\omega_t^{(m)} = \frac{\omega_t^{(m)}}{W}$
end for
Calculate effective size $M_E$ according to (24)
if $M_E < M_T$
    Resample
end if
```

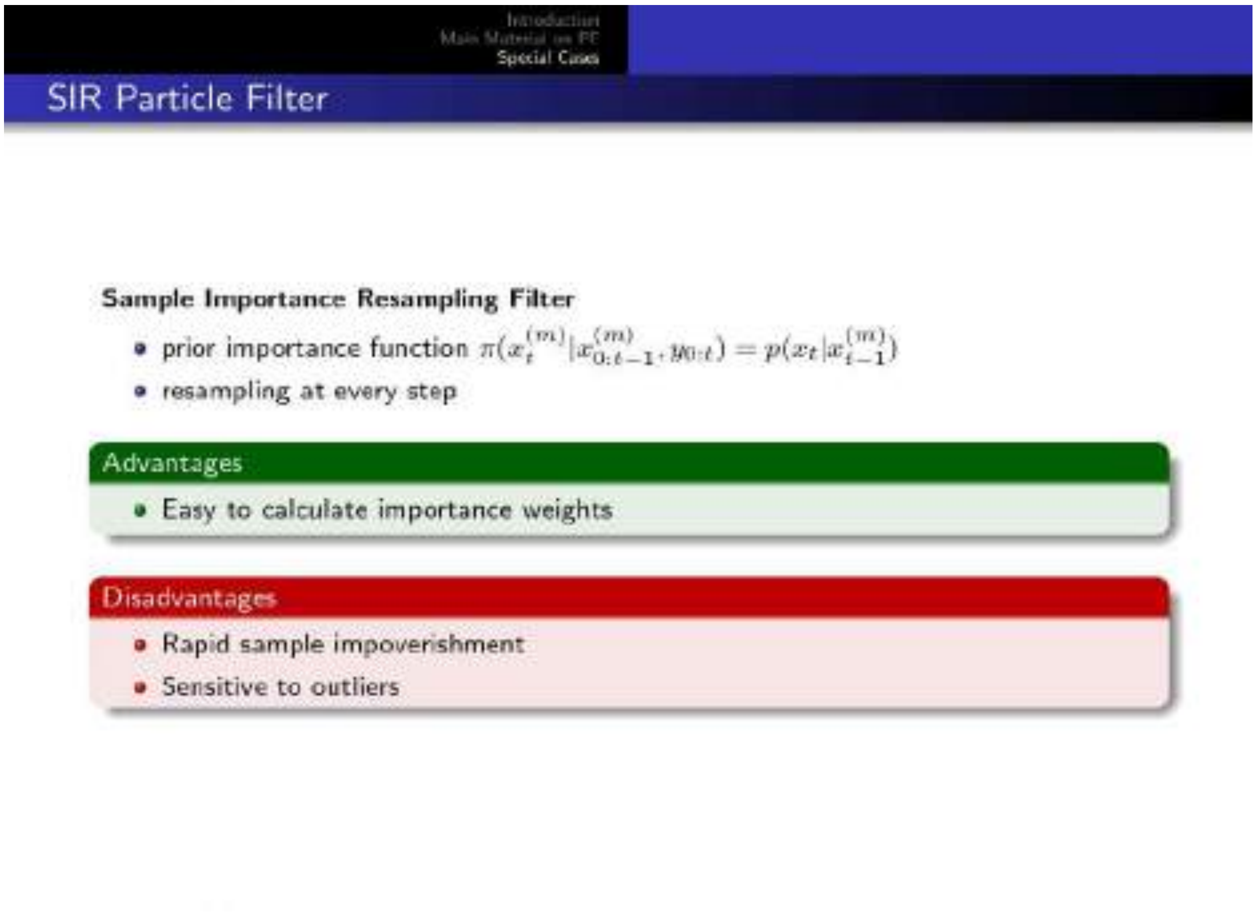


Summary

Drawbacks of generic PF due to resampling

- sample impoverishment , i.e. loss of diversity
- limited opportunity for parallelizing
- smoothed estimates based on particles' trajectories degenerate

Special Cases



SIR Particle Filter

Sample Importance Resampling Filter

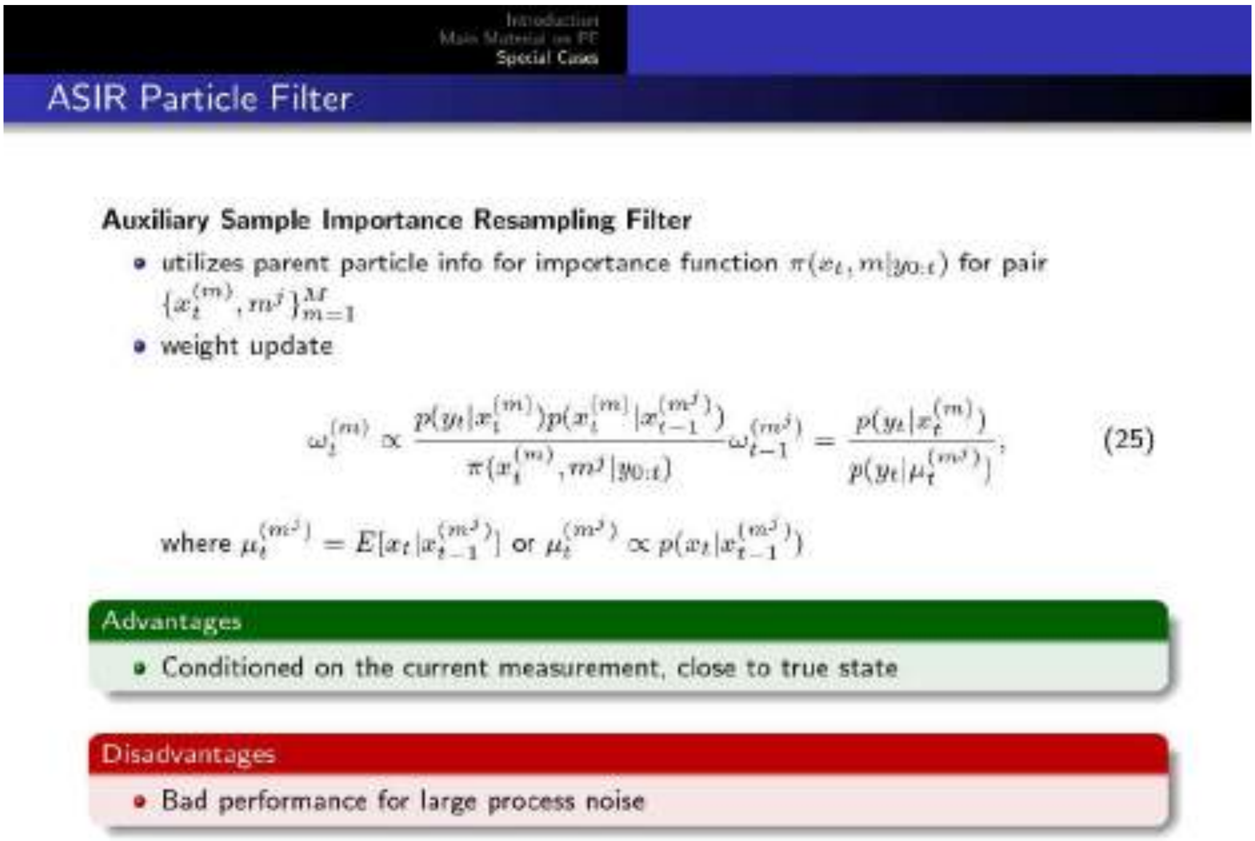
- prior importance function $\pi(x_t^{(m)}|x_{0:t-1}^{(m)}, y_{0:t}) = p(x_t|x_{t-1}^{(m)})$
- resampling at every step

Advantages

- Easy to calculate importance weights

Disadvantages

- Rapid sample impoverishment
- Sensitive to outliers



ASIR Particle Filter

Auxiliary Sample Importance Resampling Filter

- utilizes parent particle info for importance function $\pi(x_t, m|y_{0:t})$ for pair $\{x_t^{(m)}, m^j\}_{m=1}^{M}$
- weight update

$$\omega_t^{(m)} \propto \frac{p(y_t|x_t^{(m)})p(x_t^{(m)}|x_{t-1}^{(m^j)})}{\pi(x_t^{(m)}, m^j|y_{0:t})}\omega_{t-1}^{(m^j)} = \frac{p(y_t|x_t^{(m)})}{p(y_t|\mu_t^{(m^j)})}; \tag{25}$$

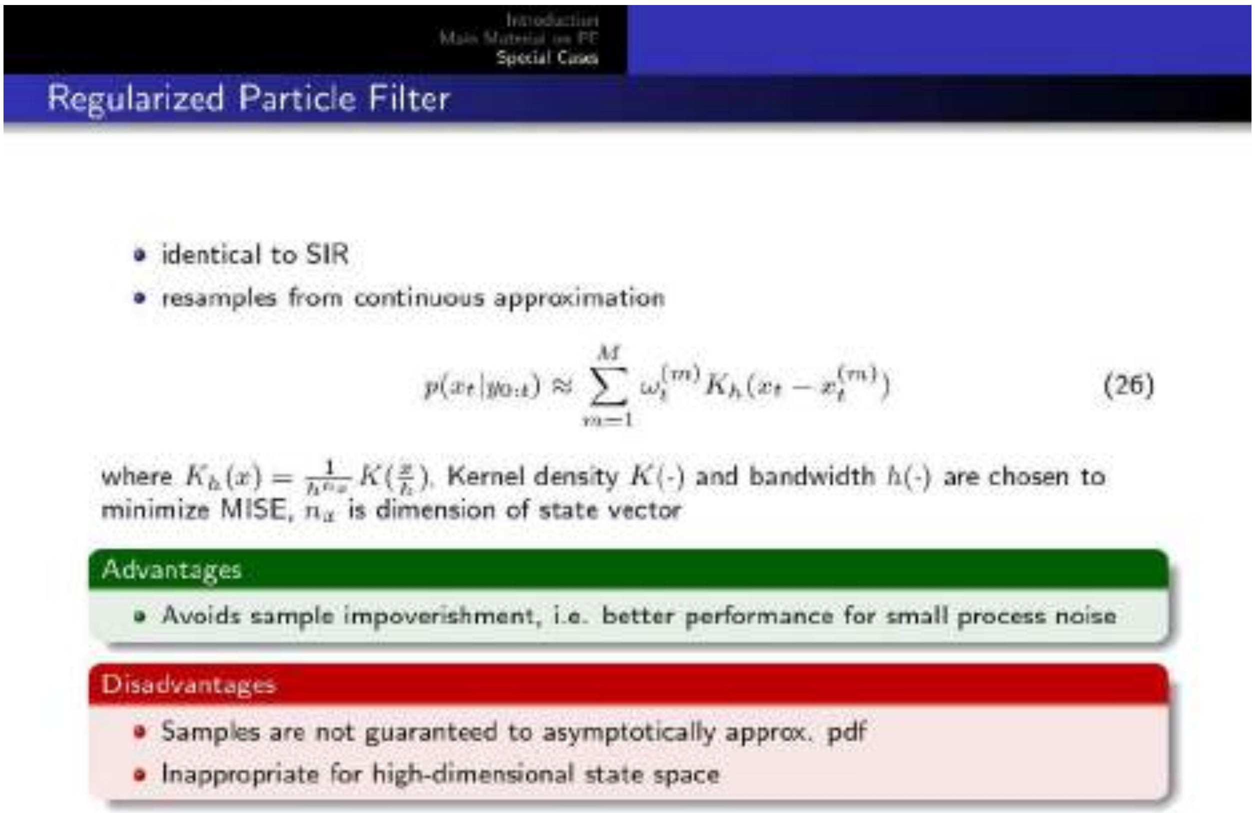
where $\mu_t^{(m^j)} = E[x_t|x_{t-1}^{(m^j)}]$ or $\mu_t^{(m^j)} \propto p(x_t|x_{t-1}^{(m^j)})$

Advantages

- Conditioned on the current measurement, close to true state

Disadvantages

- Bad performance for large process noise



Regularized Particle Filter

- identical to SIR
- resamples from continuous approximation

$$p(x_t|y_{0:t}) \approx \sum_{m=1}^{M} \omega_t^{(m)} K_h(x_t - x_t^{(m)}) \tag{26}$$

where $K_h(x) = \frac{1}{h^{n_x}}K(\frac{x}{h})$. Kernel density $K(\cdot)$ and bandwidth $h(\cdot)$ are chosen to minimize MISE, n_x is dimension of state vector

Advantages

- Avoids sample impoverishment, i.e. better performance for small process noise

Disadvantages

- Samples are not guaranteed to asymptotically approx. pdf
- Inappropriate for high-dimensional state space

ITMO UNIVERSITY
Department of Control systems and Informatics

- per. Grivtsova, 14,
  Saint Petersburg, Russia, 190000

  +7 (812) 595-41-28
  csi@corp.ifmo.ru
  csi.ifmo.ru/en/