Optimization, Robotics & Biomechanics (ORB)
Institute of Computer Engineering (ziti)
Heidelberg University

Prof. Dr. Katja Mombaur,
Benjamin Reh,
Silvan Lindner

# **Robotics 1** (WS 2018/2019)

Exercise Sheet 2

Presentation during exercises in calendar week 46

## Initial Value Problems and Numerical Integration

A first order initial value problem (IVP) is an ordinary differential equation (ODE)

$$y'(t) = f(t, y(t)) \tag{1}$$

together with an initial value $y(t_0) = y_0$. $f(t, y(t))$ is called the *right hand side* of the ODE. The solution to this IVP is obviously given by

$$y(t) = y_0 + \int_{t_0}^{t} f(s, y(s))ds,$$

however, a closed form solution is often difficult to obtain. In that case numerical integration is used to approximate a solution for $y(t)$.

A simple yet powerful class for the solutions of IVP's are the *single step methods*. Let $t_0 < t_1 < t_2 < \ldots < t_n$ be a subdivision of the interval $[t_0, t_n]$. Then $y(t_{i+1})$ is approximated by

$$y_{i+1} = y_i + h_i \cdot \Phi(t_i, y_i, y_{i+1}, h_i), \quad \forall i = 0, \ldots, n-1 \tag{2}$$

with the *step length* $h_i = t_{i+1} - t_i$. The simplest single step methods are

$$\begin{aligned} \textbf{Explicit Euler :} \quad & y_{i+1} = y_i + h/2\,[f(t_i, y_i) + f(t_{i+1}, y_{i+1})] \\ \textbf{Heuns Method:} \quad & y_{i+1} = y_i + h/2\,[f(t_i, y_i) + f(t_{i+1}, y_{i+1} + hf(t_i, y_i))] \end{aligned}$$

which simply approximate the integral using box or trapez-rule approximations.

In most programming exercises an integrator will be provided

```
VectorXd integrator (double t, VectorXd y, double h, RHS rhs)
```

that performs a single integration step as in equation (2), e.g. calculates $y_{i+1}$ from $y_i$, $t_i$ and $h$. The value `rhs` has to be a function handle

```
VectorXd rhs (double t, VectorXd y)
```

which is the *right hand side* of an ordinary differential equation (see equation (1)).

### Exercise 2.1 – Order Reduction for Ordinary Differential Equations

Numerical integration schemes mostly handle first order ODEs. However, many systems come in the form of higher order ODEs. A m-th order ODE is given by the general form

$$y^{(m)}(t) = f\left(t,\, y^{(0)}(t),\, y^{(1)}(t),\, \ldots,\, y^{(m-1)}(t)\right), \tag{3}$$

where $y^{(i)}(t)$ is the i-th time derivative of $y$. In order to solve ODE (3) using numerical integration an order reduction needs to be performed. By substituting

$$z_i(t) := y^{(i)}(t), \quad i = 0, \ldots, m-1 \tag{4}$$

system (3) can be re-arranged into a system of 1st order ODEs

$$z'(t) = \begin{bmatrix} z_0' \\ \vdots \\ z_{m-2}' \\ z_{m-1}' \end{bmatrix} = \begin{bmatrix} z_1 \\ \vdots \\ z_{m-1} \\ f(t, z_0, z_1, \ldots, z_{m-1}) \end{bmatrix}. \tag{5}$$

The following ODEs are to be converted into an equivalent system of 1st order ODEs:

- $u^{(2)}(t) = u(t)$
- The equation of motions of a rigid multi-body systems: $M(q)\ddot{q} = N(q, \dot{q}) + T\tau$
  **Additional information:** $M(q)$ represents the positive definite and symmetric *joint space inertia matrix*. The term $N(q; \dot{q})$ holds nonlinear dynamic effects (e.g. Coriolis effects) and the effects of gravity on the rigid multi body system. $\tau$ are called the generalized forces on the system (e.g. joint torques) which are projected on the minimal coordinates by the projector $T$.

### Exercise 2.2 – Single Pendulum

The dynamic equation of the single pendulum has been discussed in lecture – it resolves to the expression below:

$$ml\ddot{q} = -m \cdot g \cdot \sin(q)$$

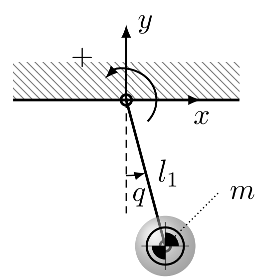



Figure 1: Single pendulum.

- This 2nd order ODE should be transformed into its equivalent 1st order ODE system and implemented inside the code snippet from `single_pendulum/main.cpp`. Integration of the motion should then be computed using the `rk4_integrator` over a time horizon of 5 seconds. The implementation should be based on the following values for $m = 0.2$ and $l = 0.4$. Run your code to generate the `animation.csv` file. Visualize the results with MeshUp using the provided `singlePendulum.lua` and `motion-singlePendulum.txt`.
- Change the integrator to `euler_integrator` and increase the time horizon to 20 seconds. What happens?

**Exercise 2.3 – Configuration and Work Spaces**

The tip coordinate for the two-link planar 2R robot in figure 2 are given by

$$x = 2\cos(\theta_1) + \cos(\theta_1 + \theta_2)$$
$$y = 2\sin(\theta_1) + \sin(\theta_1 + \theta_2)$$

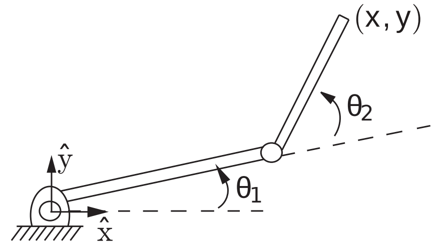



Figure 2: Two-link planar 2R open chain.

- What is the robot's configuration space?
- What is the robot's workspace (i.e., the set of all points reachable by the tip)?
- Suppose infinitely long vertical barriers are placed at $x = 1$ and $x = -1$. What is the free configuration space of the robot (i.e. the portion of the configuration space that does not result in any collision with the vertical barriers)?

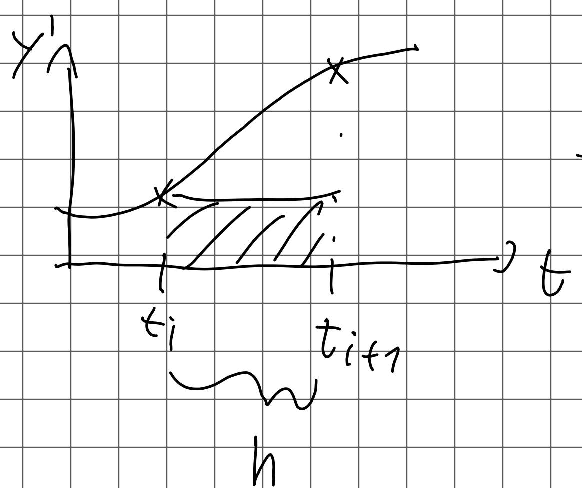
## Initial Value Problem:

$y' = f(t, y(t))$

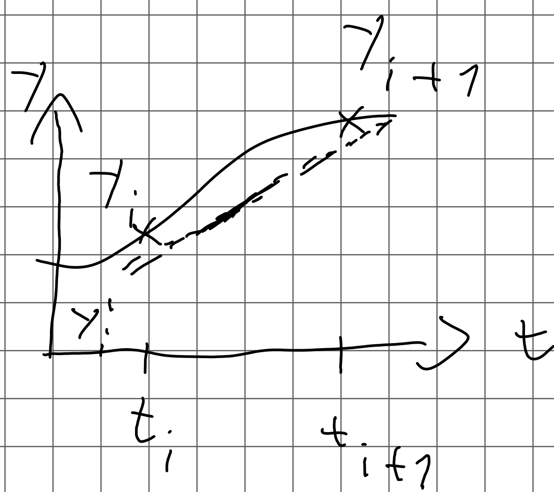
$y = \int f(s, y(s))\, ds + y_0$

$y_{i+1} = y_i + h\phi \rightarrow$ Integrator

## explicit Eeuler




→




..... egal schau in Rasendts Script

→ Last slide: good book: (2ond)
↳ "..... of Robotics"
Modern Robotics
↳ youtube - Kurs ?