MŰEGYETEM 1782

Parallel computing for determining stable parameter domain in mechatronic applications

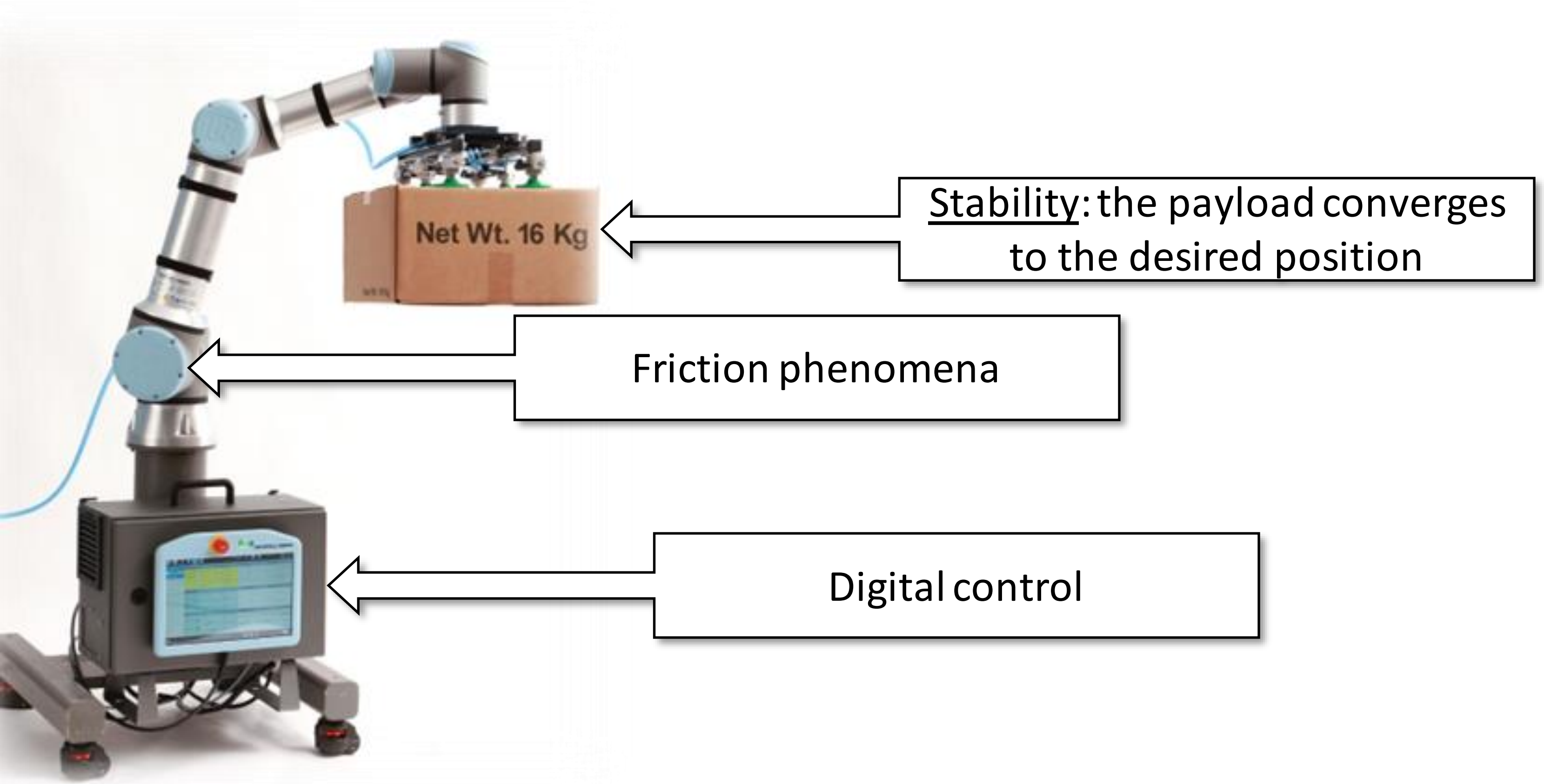
Tamás Haba, Csaba Budai

Budapest University of Technology and Economics

GPU Day 2022

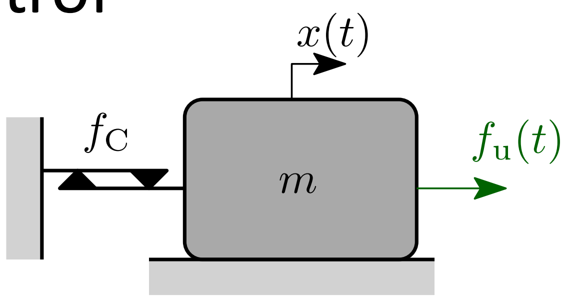
Introduction

- Basic task in mechatronics: positioning
- Design criteria: low settling time, high accuracy, stability

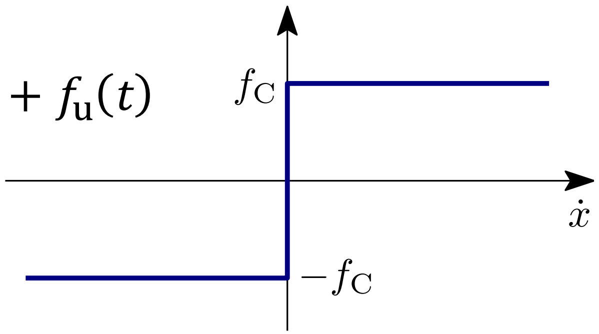


Basic model of control

- 1 DoF position control



- Equation of motion: $m \cdot \ddot{x}(t) = -f_{\mathrm{C}} \cdot \mathrm{sgn}\big(\dot{x}(t)\big) + f_{\mathrm{u}}(t)$
  - Coulomb-friction
  - Discontinuous right hand side



- Control law:
  - Full state feedback
  - Digital control with Zero-order hold

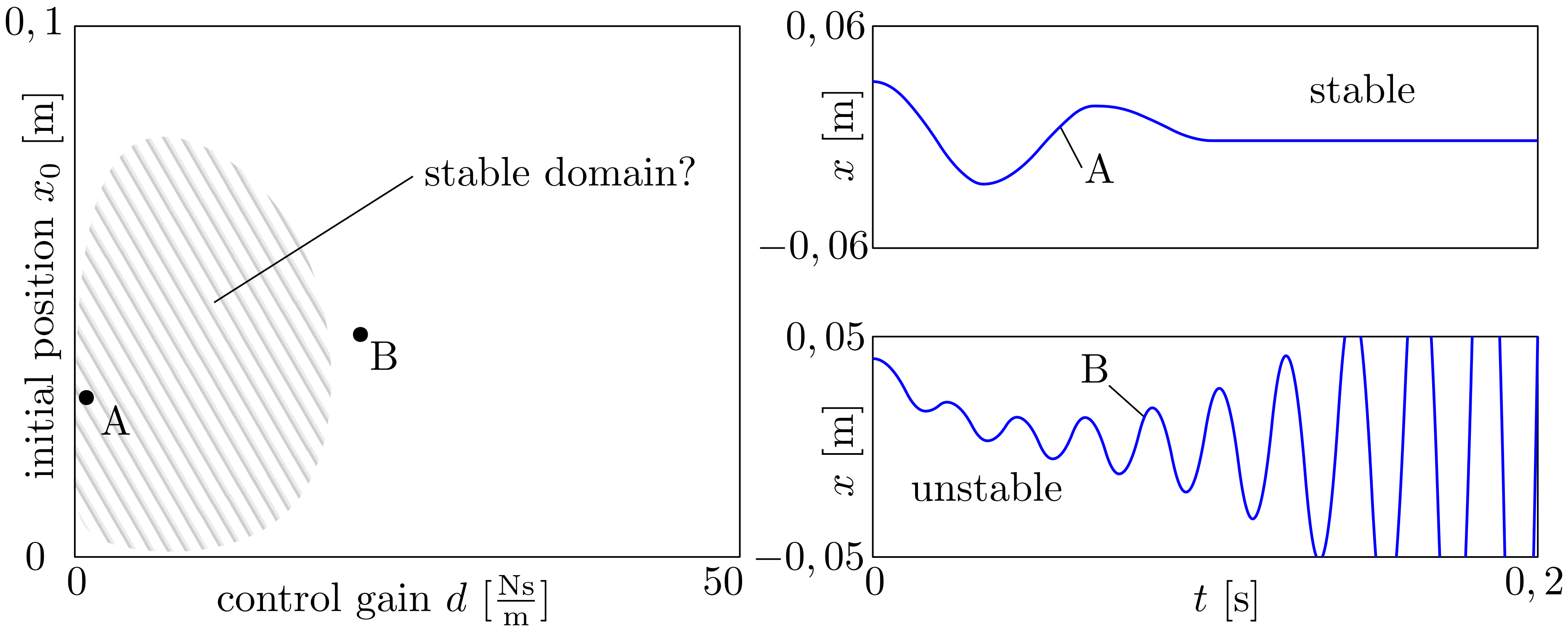
$$f_u(t) = -k_{\mathrm{p}} \cdot x(k\tau) - k_{\mathrm{d}} \cdot \dot{x}(k\tau)$$

$$k = 0,1,2,\ldots$$

$$t \in [k\tau, (k+1)\tau)$$

Problem statement

- How do we determine the stable control gains?



Problem statement

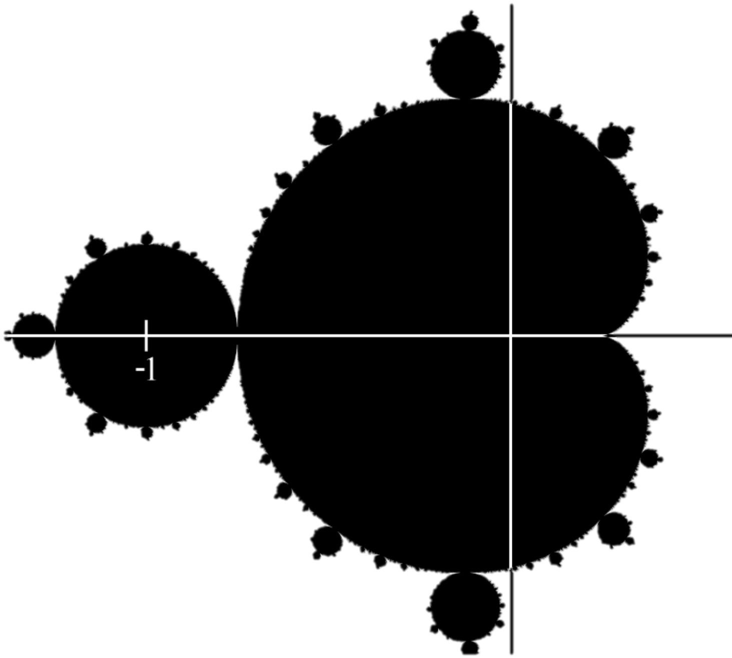
- How do we determine the stable control gains?
- Method 1: Analytically
  - Very limited use case
  - Friction effects must be neglected
- Method 2: By simulations
  - Nonlinear models can be used
  - High computational costs
  - Can be parallelized

Stability analysis by simulation

- Inspiration: Mandelbrot-set
- For which $c \in \mathbb{C}$ does the sequence

$$z_0 = 0$$
$$z_{k+1} = z_k^2 + c$$

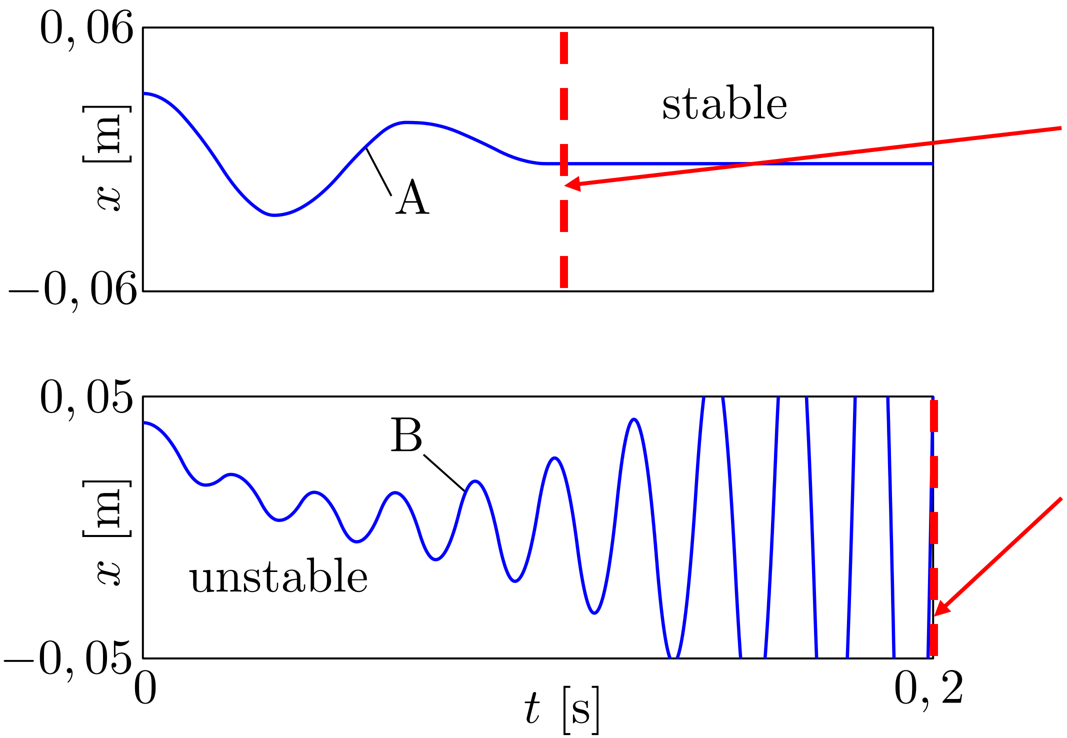
  converge?



- Escape-time algorithm
  - run the iteration in a loop
  - Stop conditions:
    - $|z_k| > 2$: the sequence diverges
    - A maximum number of iterations is reached: the sequence is considered convergent

Stability analysis by simulation

- Determining stability for one parameter combination



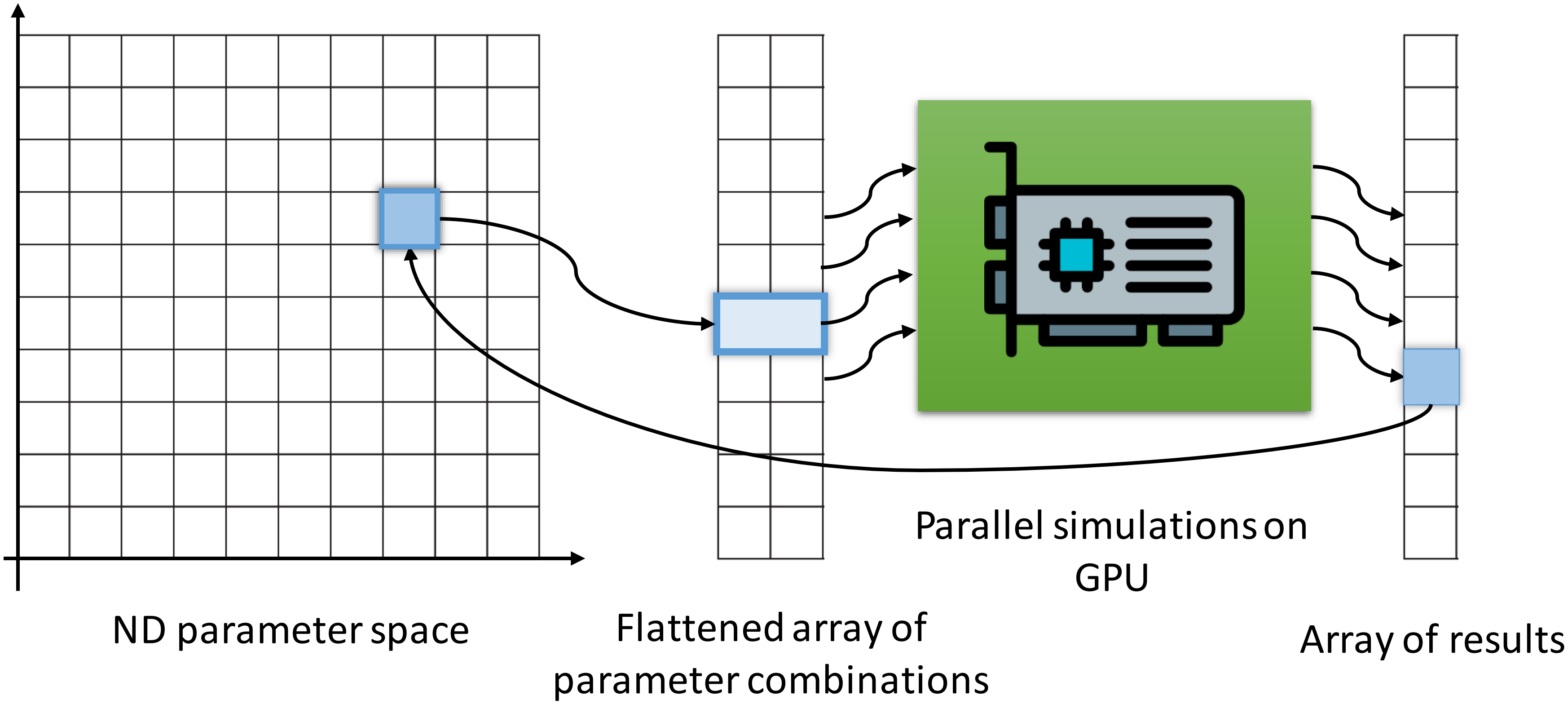
Stop condition A:
motion stops and remains in rest
→ definitely stable

Stop condition B:
time limit exceeded
→ considered unstable

Discretizing the problem

- The original system is an ODE: $m \cdot \ddot{x}(t) = -f_{\mathrm{C}} \cdot \operatorname{sgn}\big(\dot{x}(t)\big) + f_{\mathrm{u}}(t)$
- Escape-time algorithm works with sequences
- Method 1: discretizing derivative
  - General approach
  - Common methods: Euler, Runge-Kutta
  - Handling discontinuities is problematic
- Method 2: piecewise analytic solution
  - Timestep fixed to sampling time
  - Specific to the system
  - Effective computation

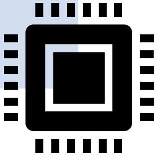
Parallelizing the simulations



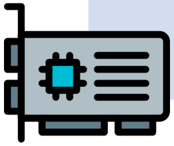
Implementation

- Method implemented in OpenCL™

Host code		Kernel code
• Written in C/C++ • Allocates buffers • Sets initial values • Requests kernel execution • Reads back results	Parameters, initial conditions → ← Stability analysis results	• Written in OpenCL C • Implements escape time algorithm • Determines stability for one parameter combination



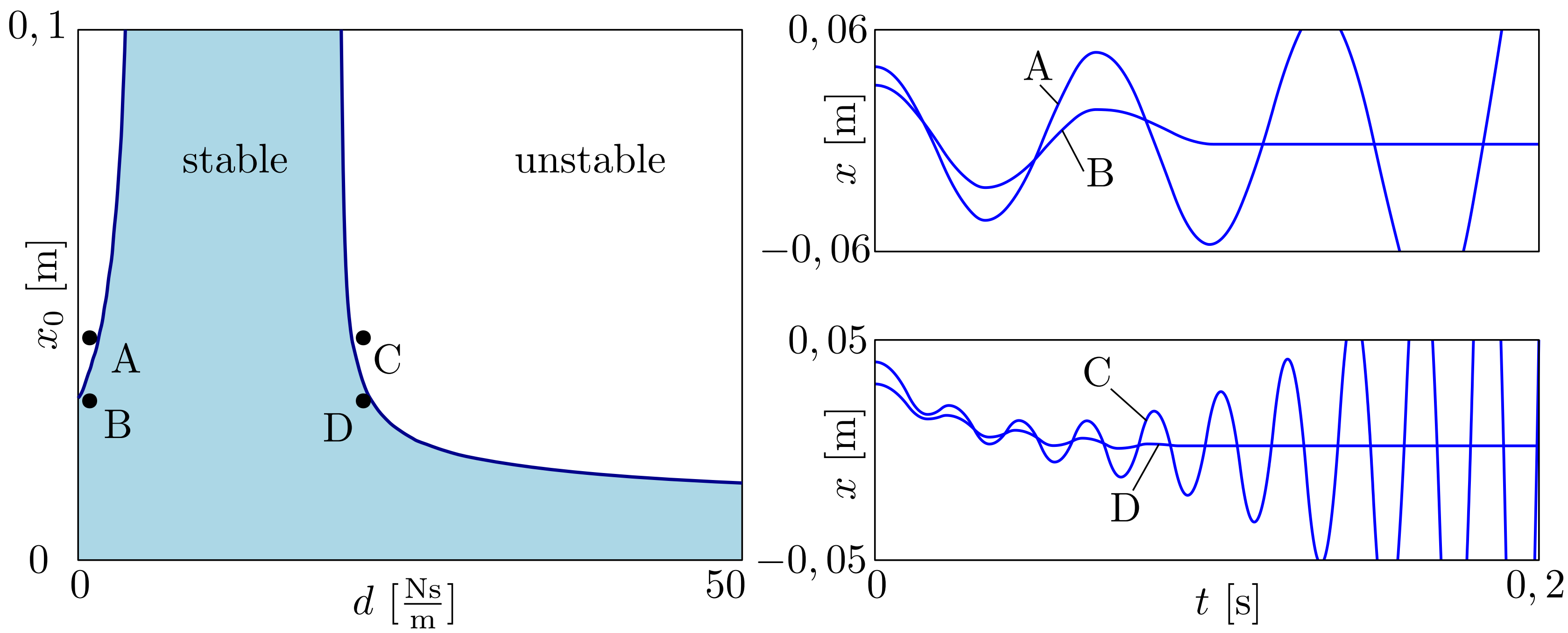
Runs on CPU



Runs on GPU

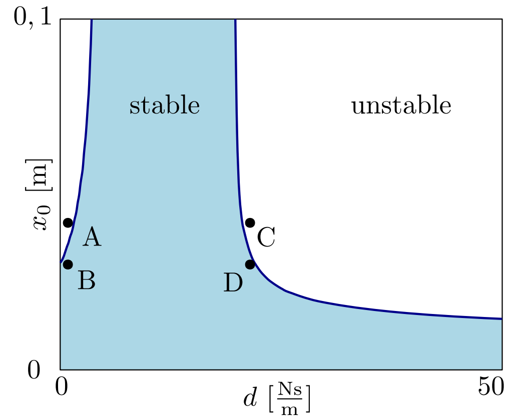
Results

- Simulations implemented in OpenCL
- 1000 by 1000 parameter grid
- 1000 iterations for each parameter combination



Wrap-up

1. Initial problem in continuous-time $m \cdot \ddot{x}(t) = -f_C \cdot \mathrm{sgn}(\dot{x}(t)) + f_u(t)$
2. Discretize solution $x_{k+1} = f(x_k)$
3. Implement simulation and escape-time algorithm in kernel code OpenCL™
4. Run simulations on GPU parallel
5. Result: high resolution stable parameter domain



Thank you for your attention!