Využití GPGPU pro řešení soustav lineárních rovnic a aplikace ve zpracování obrazu

Vítězslav Žabka

Katedra matematiky, Fakulta jaderná a fyzikálně inženýrská,
České vysoké učení technické v Praze

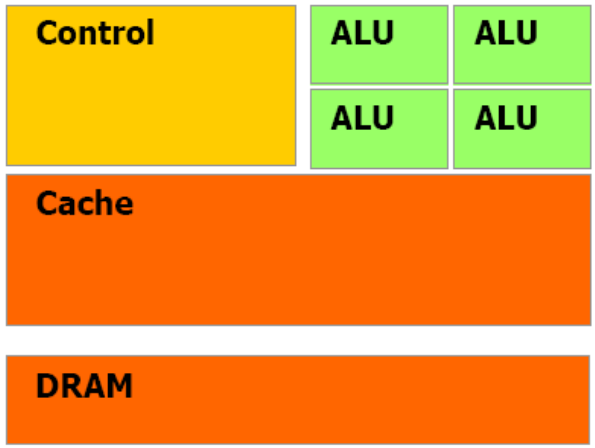
Výzkumný úkol
2008/2009

Cíle práce

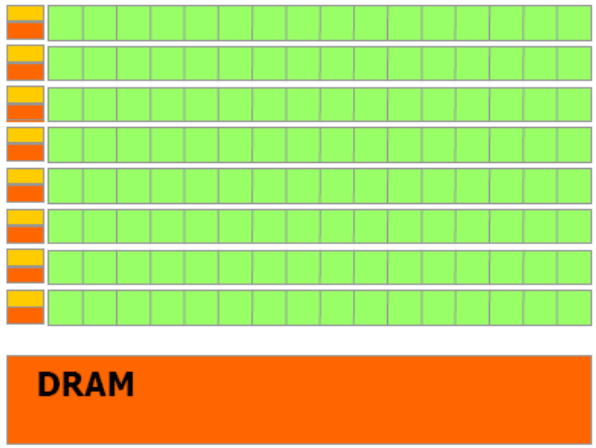
- GPGPU — obecné výpočty na grafických kartách PC (GPU)
  - Urychlení numerických výpočtů
  - Technologie CUDA
  - Implementovat Rungeův-Kuttův řešič a GMRES
- Zpracování obrazu — segmentace
  - Dělení obrazu do částí korespondujících s objekty v obrazu
  - Detekce hran pomocí rovnice Allenova-Cahnova typu

Vlastnosti GPU v porovnání s CPU

- Vysoce paralelní architektura (desítky až stovky jader)
- Přes 20× větší výkon
- 5× větší propustnost paměti
- Obtížnější využít dostupné prostředky



CPU



GPU

NVIDIA CUDA

- Paralelní architektura pro provádění obecných výpočtů na GPU
- Jednodušší na použití než dřívější GPGPU techniky
- Rozšíření jazyku C
- Grafický procesor — vysoce paralelní koprocesor k CPU
- Postup výpočtu:
  1. Kopírování dat do paměti GPU
  2. Zpracování dat na GPU řízené procesorem
  3. Kopírování výsledků z paměti GPU

Segmentace obrazu — model fázového pole

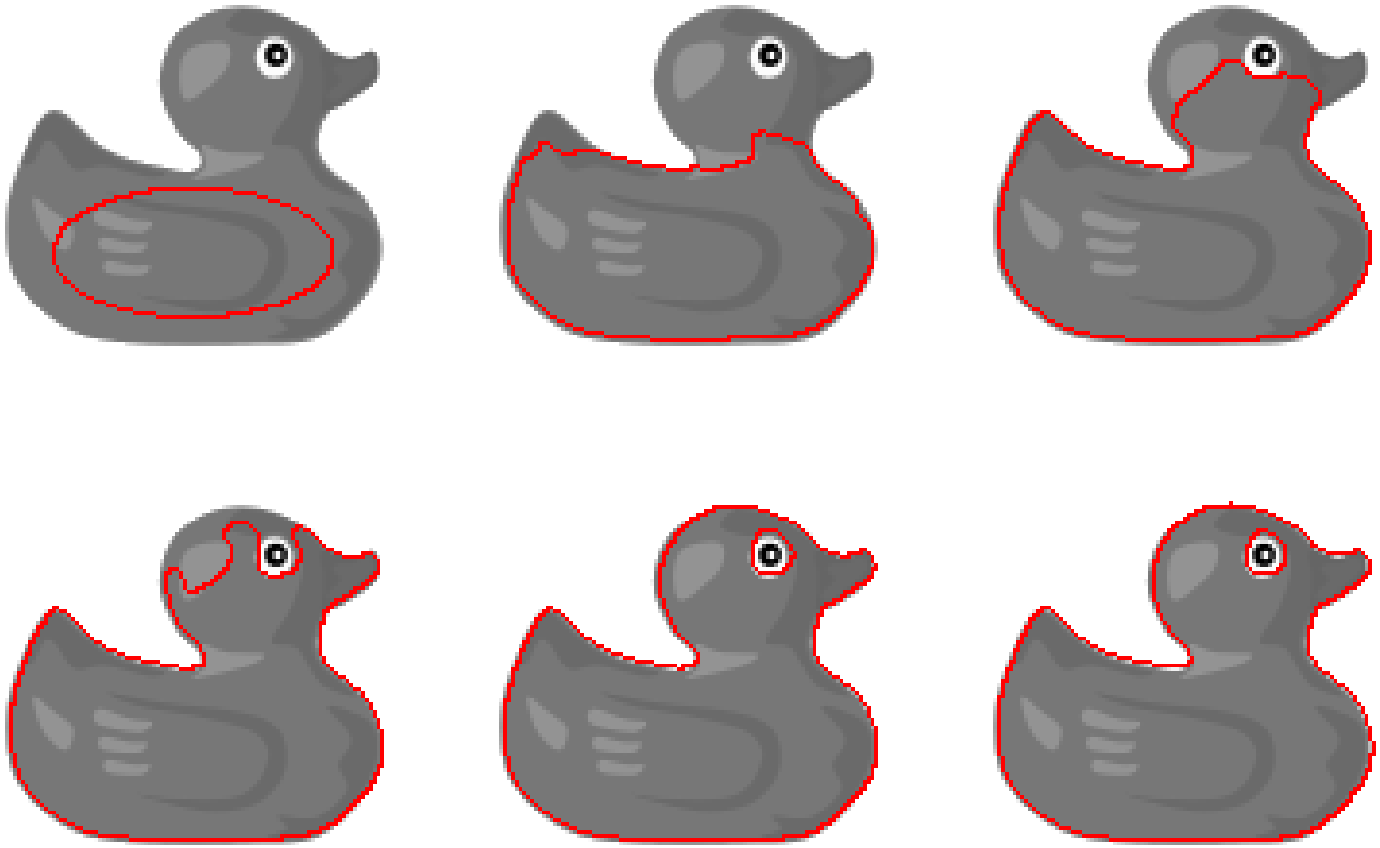
- Metoda založená na detekci hran
- Rovnice Allenova-Cahnova typu pro funkci $p = p(t, x)$ na $(0, T) \times \Omega$:

$$\begin{aligned} \xi \frac{\partial p}{\partial t} &= \xi \nabla \cdot (g \nabla p) + \frac{1}{\xi} g f_0(p) + g \xi F |\nabla p|, \\ p|_{\partial\Omega} &= 0 \quad \text{na } (0, T) \times \partial\Omega, \\ p|_{t=0} &= p_{\text{ini}} \quad \text{na } \overline{\Omega} \end{aligned}$$

- $\Omega \subset \mathbb{R}^2$ oblast
- $f_0(p) = p(1-p)(p-0{,}5)$
- $g = g(x)$ obsahuje informace o segmentovaném obrazu
- $\xi > 0$, $F = F(x)$ parametry

segmentační křivka: $\Gamma(t) \equiv p(t) = \frac{1}{2}$

Průběh segmentace



Numerické řešení — metoda přímek

- Diskretizace v prostorových proměnných
  $\longrightarrow$ semidiskrétní numerické schéma

$$\frac{\mathrm{d}x}{\mathrm{d}t} = f(t, x), \quad x(t_0) = x_0$$

- Mersonova varianta Rungeovy-Kuttovy metody
  - Adaptivní volba časového kroku
  - Implementace v CUDA jednoduchá
  - Předpoklad výrazného urychlení výpočtu pomocí GPU

Numerické řešení — GMRES

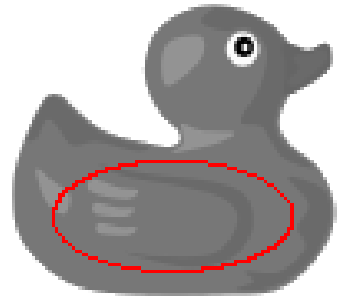
- Časová diskretizace semidiskrétního schématu
  $\longrightarrow$ semi-implictní numerické schéma

$$Ax = b, \quad A \in \mathbb{R}^{n \times n} \text{ řídká, nesymetrická}$$

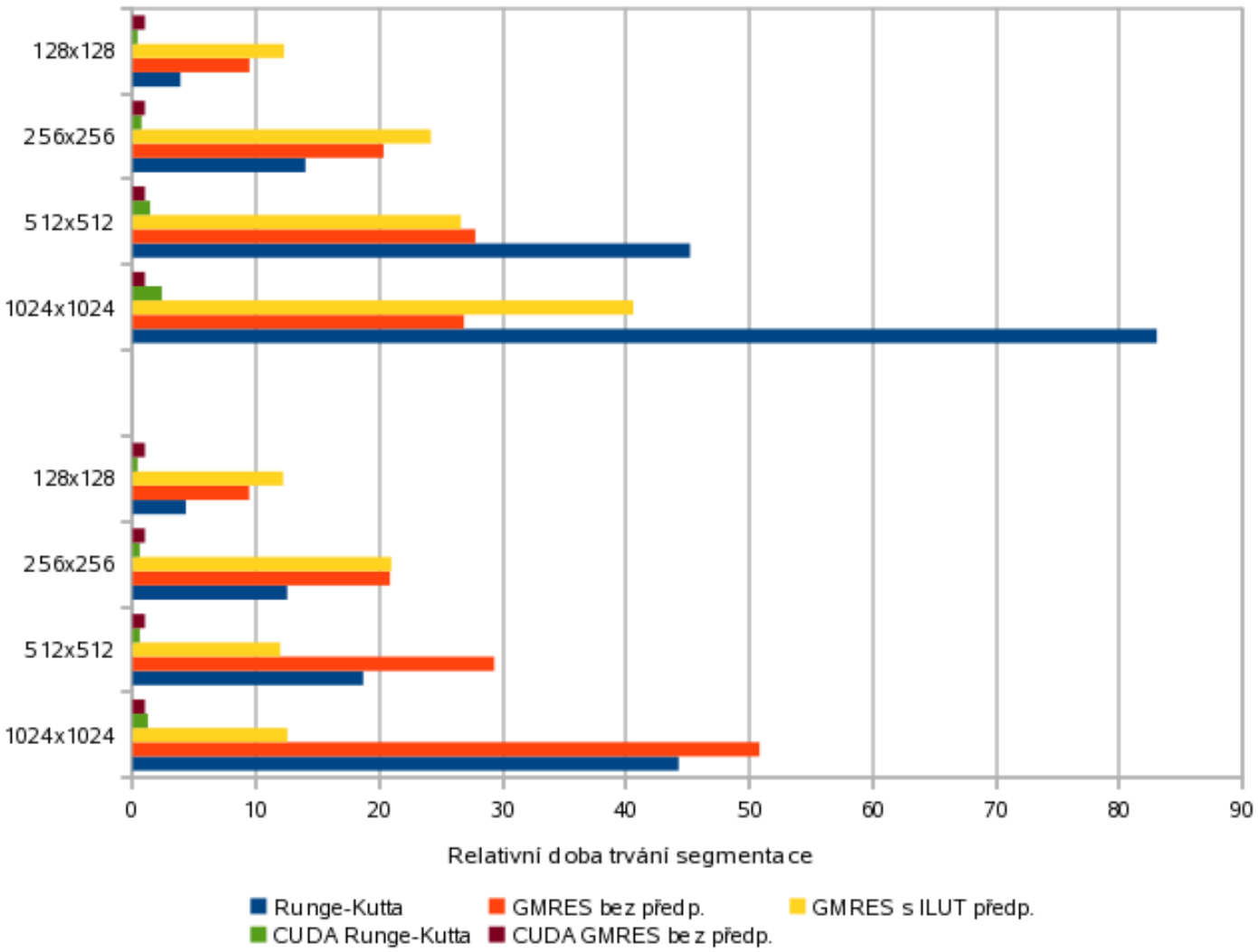
- Metoda zobecněných minimálních reziduí (GMRES)
  - Implementace v CUDA složitější
  - Možnost předpodmínění

Porovnání způsobů segmentace

- CPU — Intel Core 2 Duo E6550
  - Frekvence: 2333 MHz
  - Velikost L2 cache: 4 MB
  - Propustnost paměti RAM: 12,8 GB/s
  - Teoretický výkon: 9,32 GFLOPS/jádro
  - Výpočty s dvojitou přesností
- GPU — NVIDIA GeForce 8800 GT
  - Počet jader: 112
  - Frekvence: 1620 MHz
  - Propustnost paměti: 60,8 GB/s
  - Teoretický výkon: 544 GFLOPS
  - Výpočty s jednoduchou přesností
- Segmentace se zaměřením na rychlost
- Stejné parametry pro všechny velikosti obrazu



Výsledky


128x128
256x256
512x512
1024x1024
128x128
256x256
512x512
1024x1024
0
10
20
30
40
50
60
70
80
90
Relativní doba trvání segmentace
Runge-Kutta
GMRES bez předp.
GMRES s ILUT předp.
CUDA Runge-Kutta
CUDA GMRES bez předp.

Výsledky

- Kvalita segmentace stejná pro CPU i GPU verze řešičů
- Urychlení metody Runge-Kutta 10–35×
- Urychlení GMRES 10–50×
- U větších obrazů nastává větší urychlení
- Pro malé obrazy výhodnější Runge-Kutta, pro větší GMRES

Závěr

- Metody Runge-Kutta i GMRES vhodné pro implementaci v architektuře CUDA
- Přesnost dostatečná pro segmentaci obrazu
- Urychlení 10–50× oproti jednovláknové CPU implementaci
- Možnosti dalšího vývoje:
  - Dvojitá přesnost
  - Předpodmínění GMRES na GPU pomocí CUDA
  - Využití více GPU najednou