Chris Rebbelin s0548921

Wellengleichung

Parallel Systems Beleg

06.06.2018

Hochschule für Technik und Wirtschaft Berlin
University of Applied Sciences

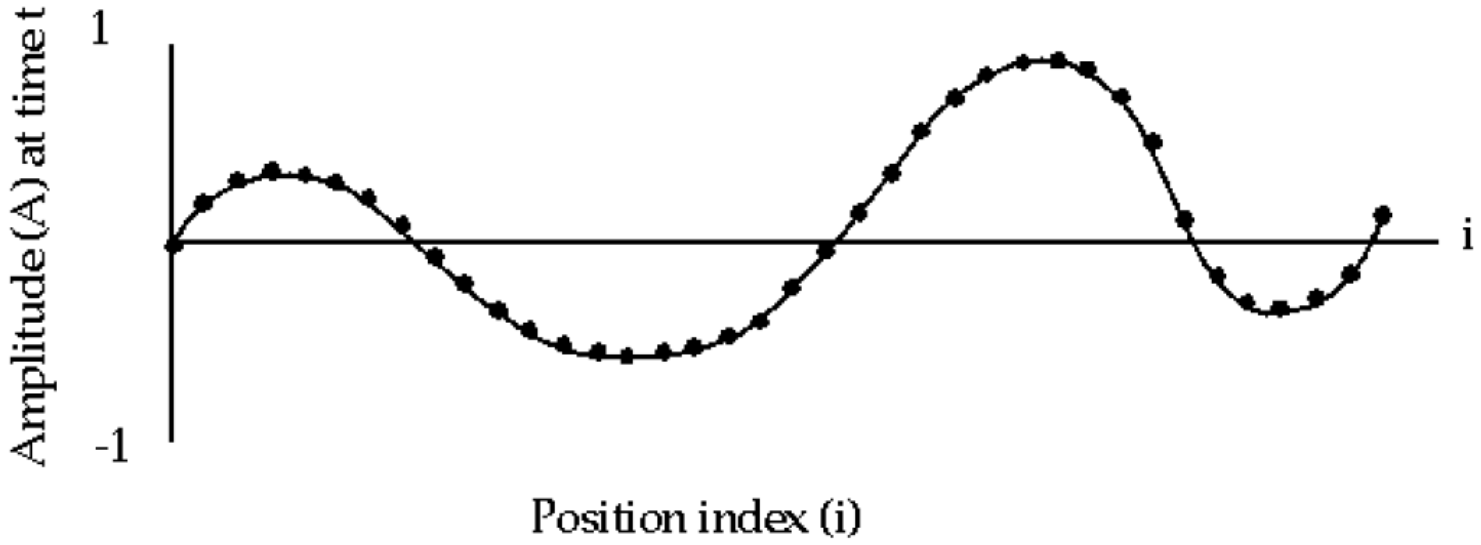
Inhalt

- Aufgabe
- Anforderungen
- Herleitung
- Implementierung
- Stand & Ausblick

Aufgabe

Visualisierung der Amplituden einer vibrierenden Saite auf Grundlage der Wellengleichung im eindimensionalen Fall

Aufgabe



Anforderungen

- Implementierung sequentiell <> parallel
- Konfiguration
- Visualisierung
- Benchmarks / Tests
- Dokumentation

- -optional: Perturbationen

Herleitung

$$u_{tt} = c^2 u_{xx}$$

$$\frac{\partial^2}{\partial t^2}A(i,t) = c^2 * \frac{\partial^2}{\partial i^2}A(i,t)$$

$$\frac{\partial^2}{\partial x^2} f(x) \approx \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}$$

$$\frac{\partial^2}{\partial t^2} A(i,t) = c^2 * \frac{\partial^2}{\partial i^2} A(i,t)$$

$$\frac{\partial^2}{\partial t^2} A(i,t) = c^2 * \frac{\partial^2}{\partial i^2} A(i,t)$$

$$\frac{A(i,t+1) - 2A(i,t) + A(i,t-1)}{t^2} =$$

$$\frac{\partial^2}{\partial t^2} A(i,t) = c^2 * \frac{\partial^2}{\partial i^2} A(i,t)$$

$$\frac{A(i,t+1) - 2A(i,t) + A(i,t-1)}{t^2} = c^2 \left(\frac{A(i+1,t) - 2A(i,t) + A(i-1,t)}{i^2} \right)$$

$$\frac{A(i,t+1)-2A(i,t)+A(i,t-1)}{t^2}=c^2\left(\frac{A(i+1,t)-2A(i,t)+A(i-1,t)}{i^2}\right.$$

$$\frac{A(i,t+1)-2A(i,t)+A(i,t-1)}{t^2} = c^2\left(\frac{A(i+1,t)-2A(i,t)+A(i,t)}{i^2}\right)$$

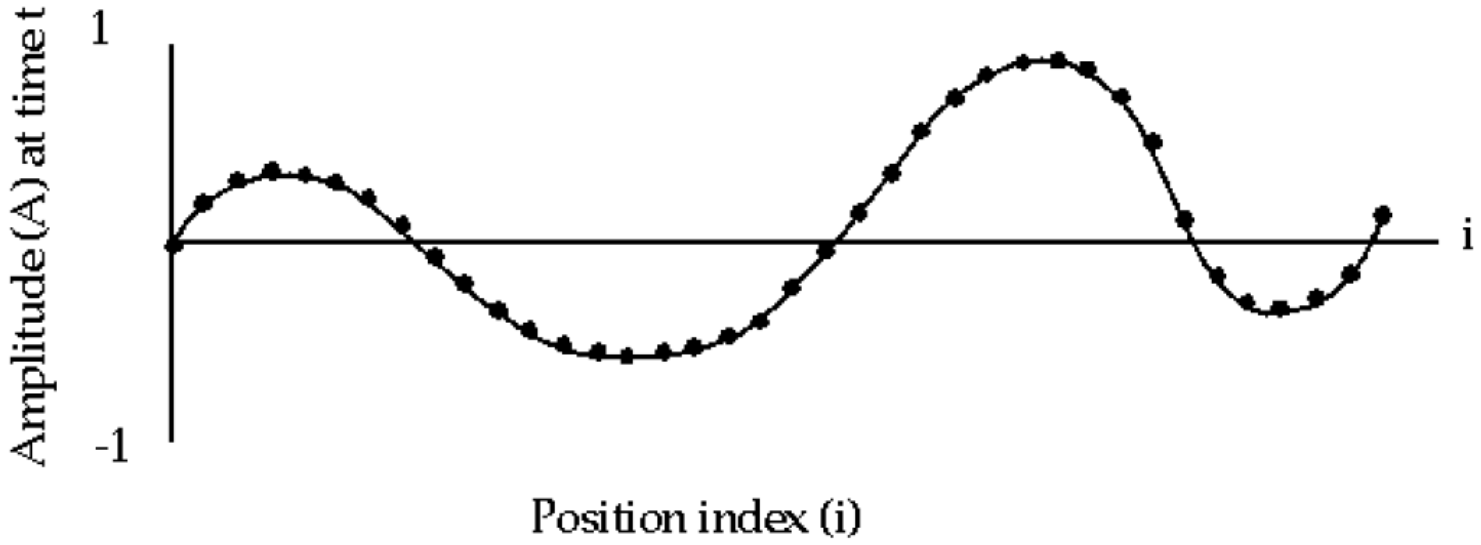
$$A(i,t+1)-2A(i,t)+A(i,t-1) = \left(\frac{c^2t^2}{i^2}\right)(A(i+1,t)-2A(i,t)+A(i-1,t))$$

$$\frac{A(i,t+1)-2A(i,t)+A(i,t-1)}{t^2} = c^2\left(\frac{A(i+1,t)-2A(i,t)+A(i-1,t)}{i^2}\right)$$

$$A(i,t+1)-2A(i,t)+A(i,t-1) = \left(\frac{c^2t^2}{i^2}\right)(A(i+1,t)-2A(i,t)+A(i-1,t))$$

$$A(i,t+1)-2A(i,t)+A(i,t-1) = c\,(A(i+1,t)-2A(i,t)+A(i-1,t))$$

Aufgabe



$$A(i,t+1) = 2A(i,t) - A(i,t-1) + c(A(i-1,t) - 2.0A(i,t) + A(i+1,t))$$

Implementierung

```
//#pragma omp parallel for \
shared(nextStep, currentStep, previousStep, C_SPEED, NPOINTS) \
private(i)
for (i = 1; i < NPOINTS; i++) {
    nextStep[i] = 2.0 * currentStep[i] - previousStep[i] +
    C_SPEED * (currentStep[i - 1] - (2.0 * currentStep[i]) + currentStep[i + 1]);
}

nextStep[0] = 0.0;
nextStep[NPOINTS] = 0.0;
```

Implementierung

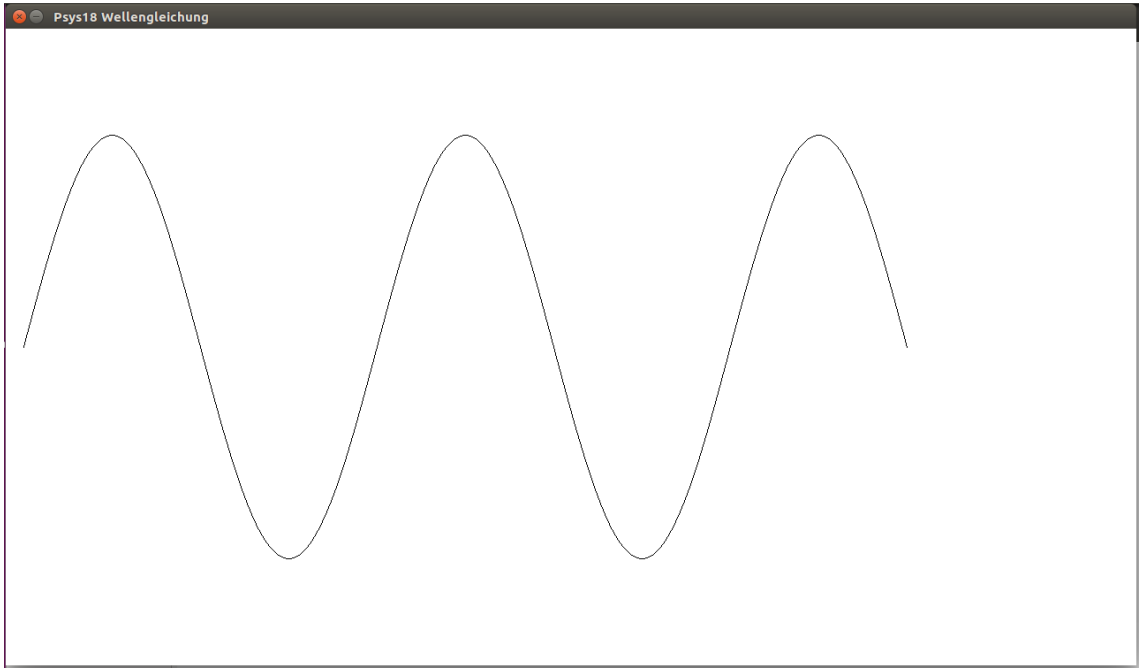
```
/*for (int k = 0; k < NPOINTS+1; k++) {
    previousStep[k] = currentStep[k];
    currentStep[k] = nextStep[k];
}*/

 double *tempStep = previousStep;
 previousStep = currentStep;
 currentStep = nextStep;
 nextStep = tempStep;
```

Stand

- sequentielle Implementierung
- OpenMP Implementierung

- (rudimentäre) Visualisierung mit SDL

Stand

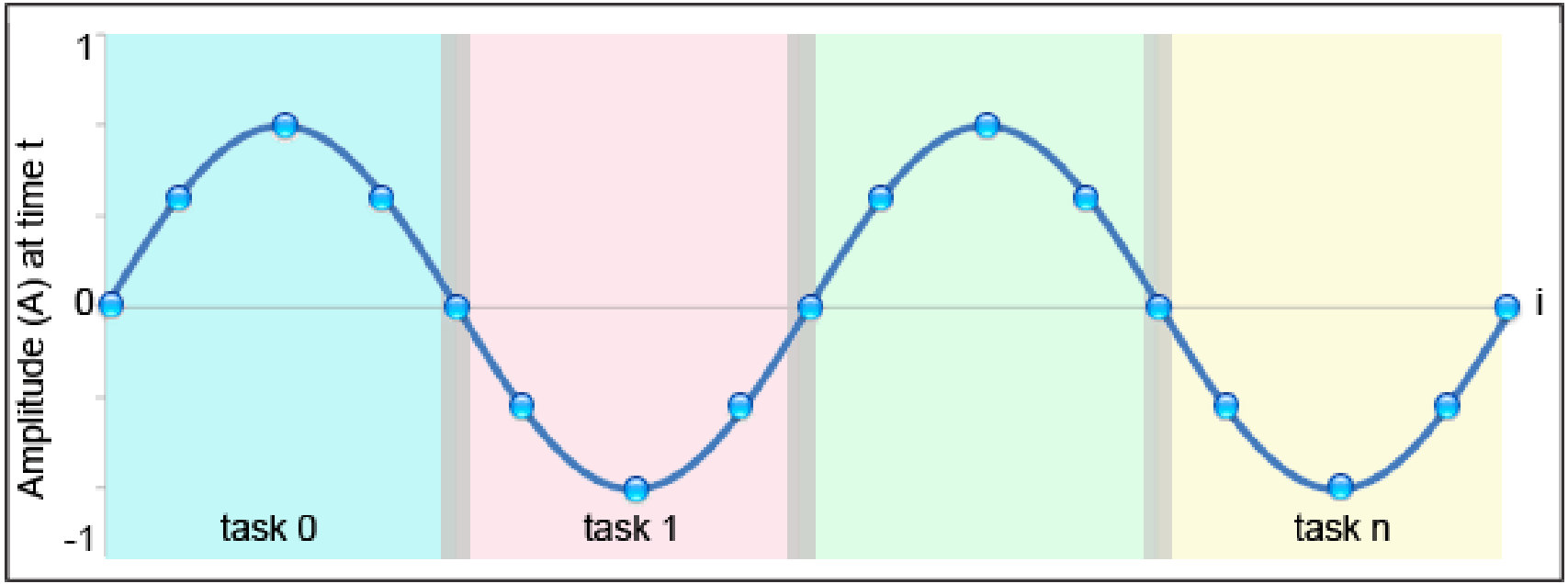


Demo

Ausblick

- zweite Schnittstelle implementieren (OpenMPI, OpenCL…?)

OpenMPI



Ausblick

- zweite Schnittstelle implementieren
- Performance testen / Benchmark
- Dokumentation

- evtl. Perturbationen?

Quellen

- *Lecture 8: Solving the Heat, Laplace and Wave equations using finite difference methods* (https://www.math.ubc.ca/~peirce/M257_316_2012_Lecture_8.pdf)

- *Parallel Examples 1-D Wave Equation* (https://computing.llnl.gov/tutorials/parallel_comp/#ExamplesWave)

- *Finite difference methods for wave motion* (http://hplgit.github.io/num-methods-for-PDEs/doc/pub/wave/pdf/wave-4print-A4-2up.pdf)

Danke für die Aufmerksamkeit!

www.htw-berlin.de