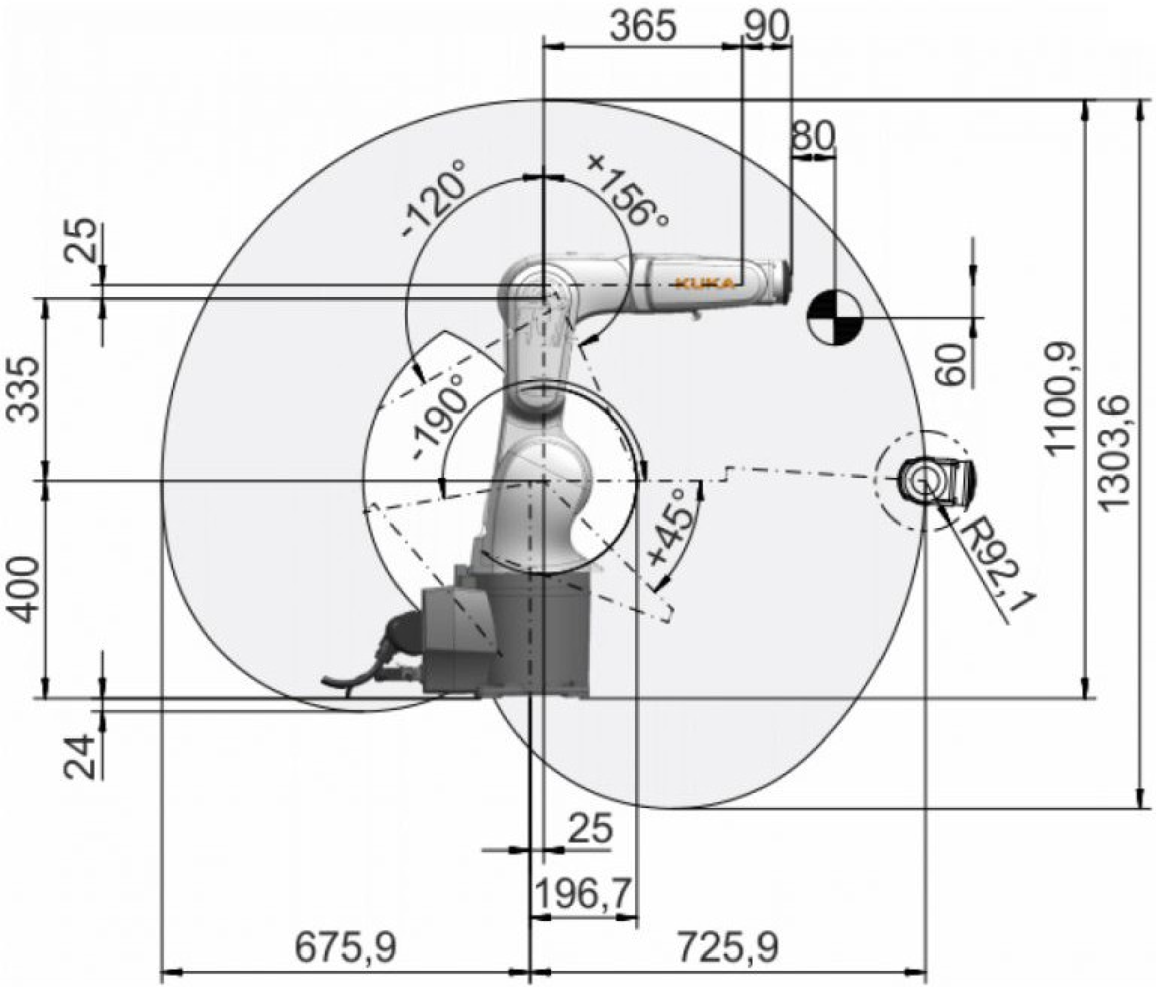
Robotic Arm Kinematics

Joint Parameter Optimization

Presenter: Steven VanCamp



Dimensions: mm

Methodology – Forward Kinematics

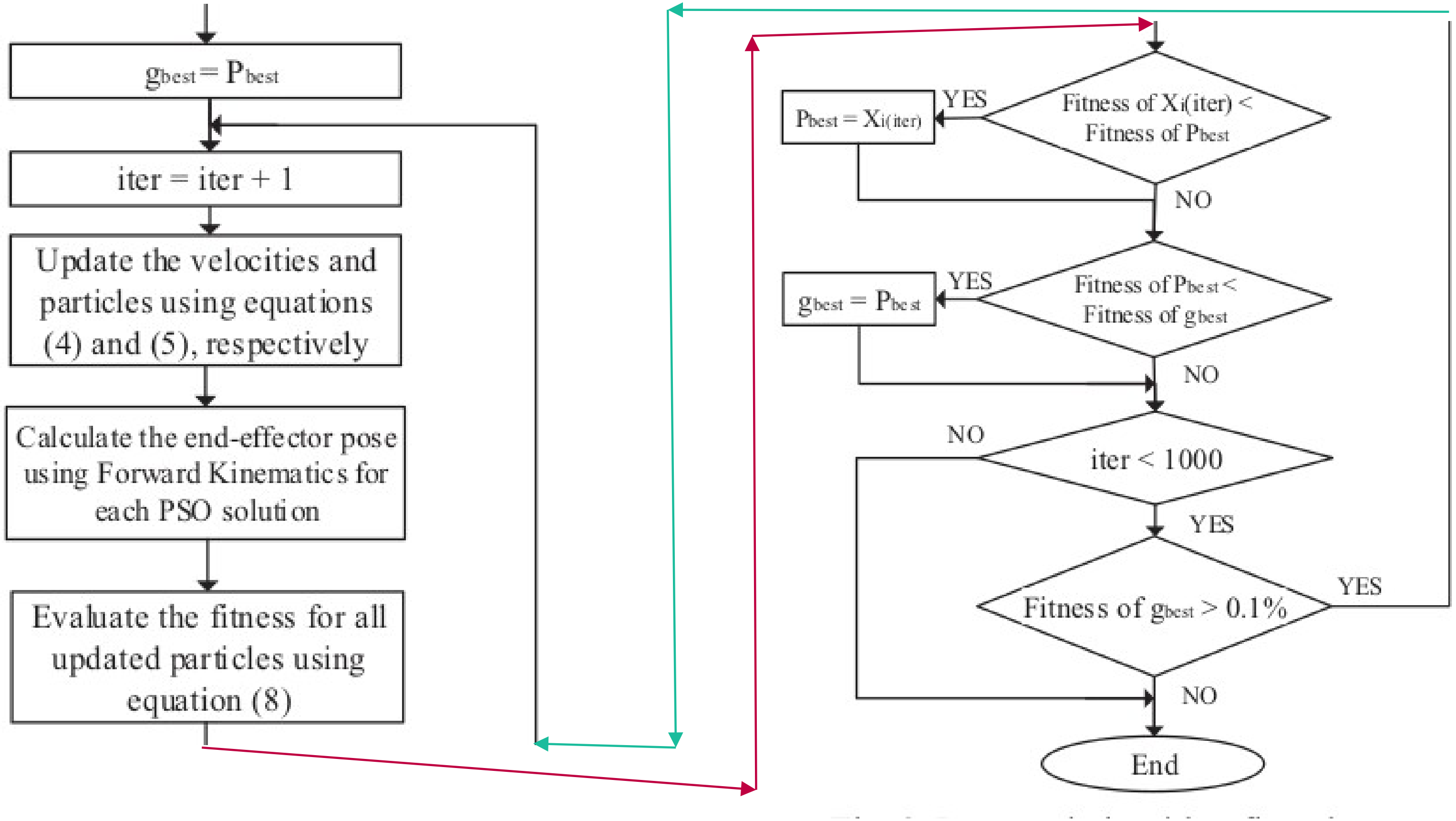
- **Denavit Hartenberg (DH) Representation**
  - Each joint represented as a transformation matrix
  - At least 1 DOF per joint
- **Solve for end effector position from joint parameters**
  - Calculate the DH translation matrix for the final joint

$$\boldsymbol{T}_i^{i-1} = \begin{bmatrix} c_{\theta_i} & -s_{\theta_i}c_{\alpha_i} & s_{\theta_i}s_{\alpha_i} & a_i c_{\theta_i} \\ s_{\theta_i} & c_{\theta_i}c_{\alpha_i} & -c_{\theta_i}s_{\alpha_i} & a_i s_{\theta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

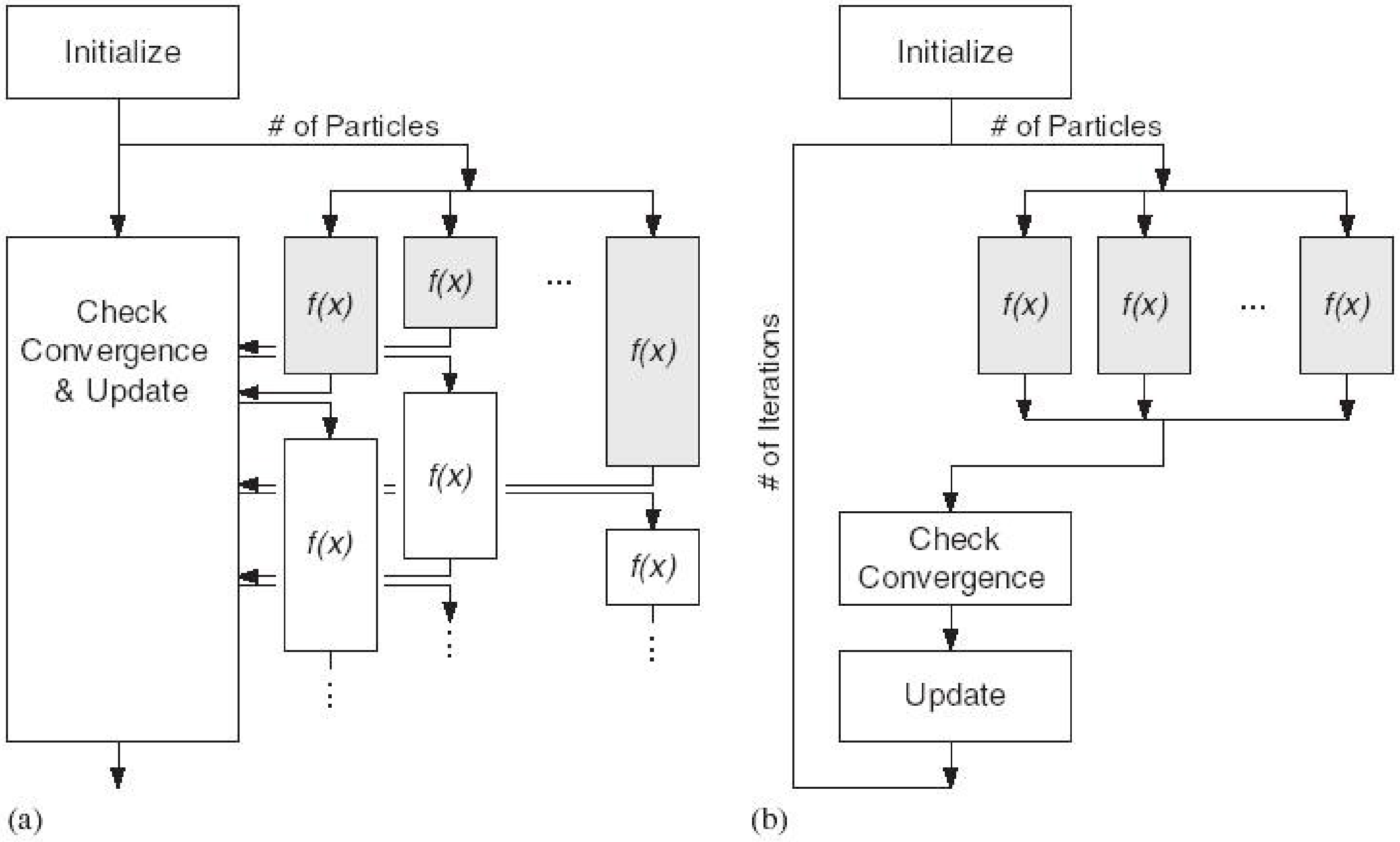
Methodology – Point Swarm Optimization (PSO)

- **Parameter Optimization**
  - Swarm of points each representing a collection of parameter values
  - Searches parameter space for a solution via random variation and through interactions between particles
- **Almost Embarrassingly Parallel**
  - Each thread handles its own particles
  - 1 check for current best solution between iterations

Methodology – Algorithm

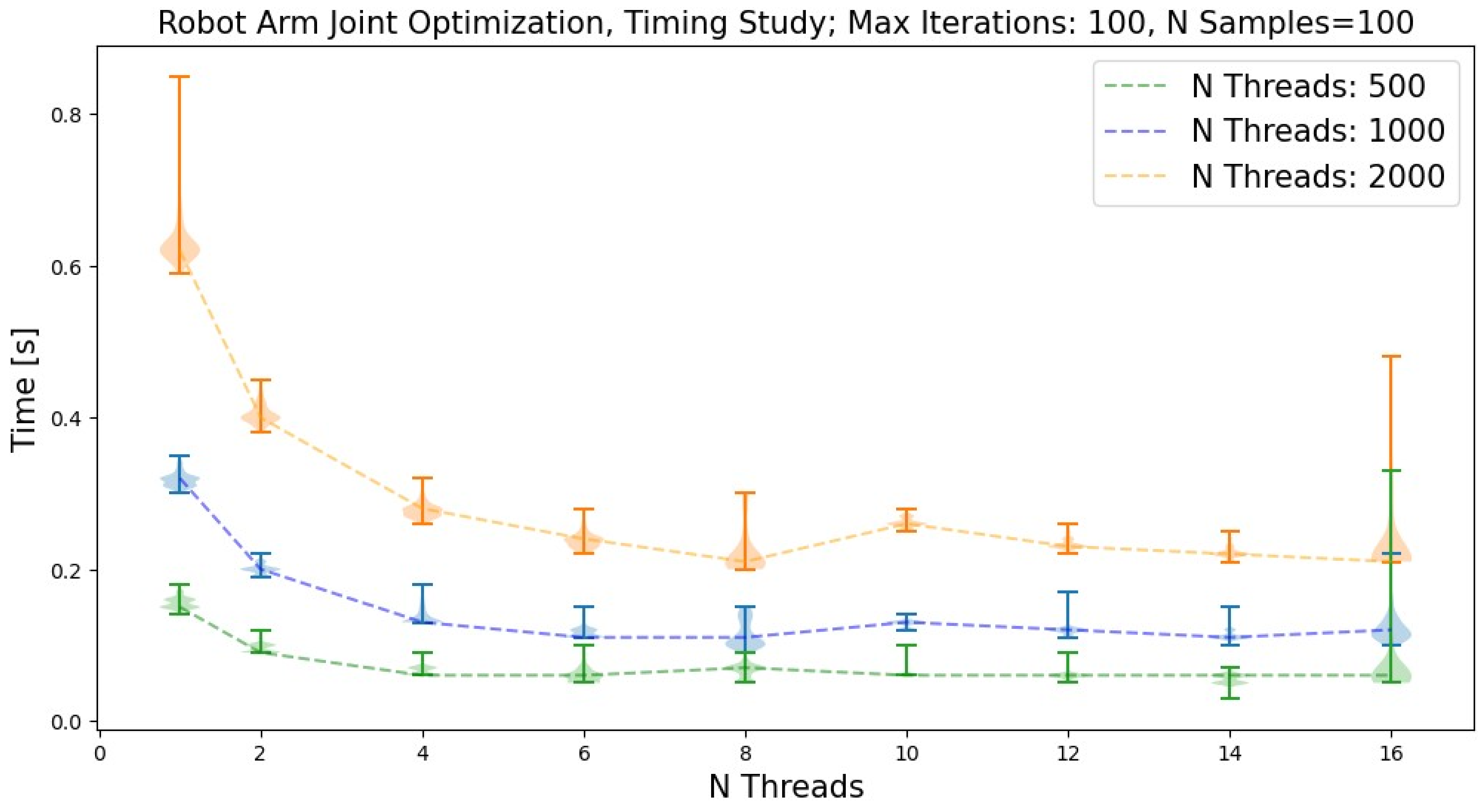


Parallelization – Two Types of (PSO)

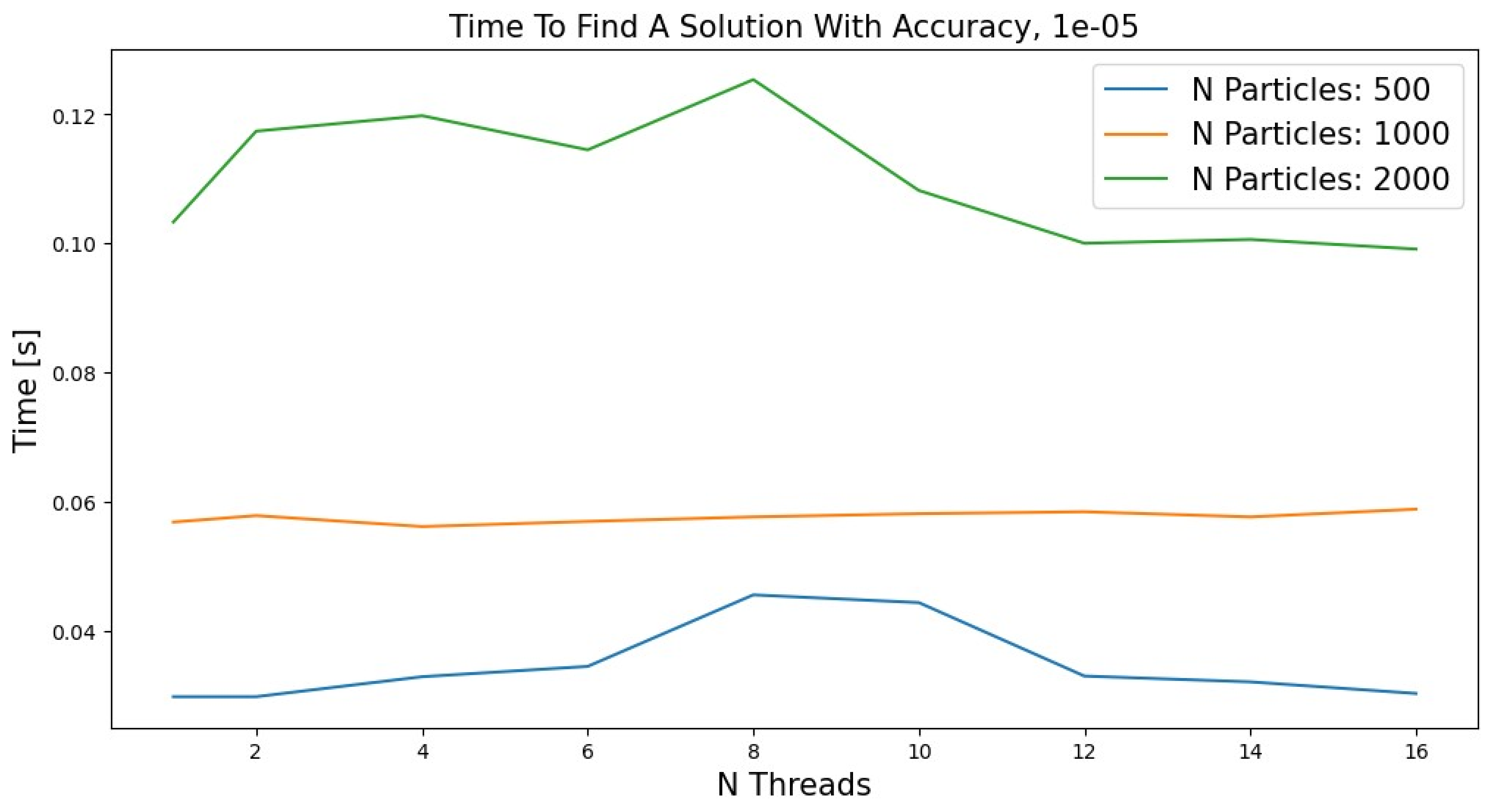


Results – Overview

- **The program was already very fast in serial**
  - Runtime < 0.1 [s] for 1000 particles over 100 iterations
- **Parallelization was fairly straight forward**
  - Almost no change to the algorithm was needed
- **Parallelization did have a noticeable effect**
  - The time to complete n iterations was decreased
  - The solution accuracy was not effected



Robot Arm Joint Optimization, Timing Study; Max Iterations: 100, N Samples=100
N Threads: 500
N Threads: 1000
N Threads: 2000
Time [s]
0.0
0.2
0.4
0.6
0.8
0
2
4
6
8
10
12
14
16
N Threads


Time To Find A Solution With Accuracy, 1e-05
N Particles: 500
N Particles: 1000
N Particles: 2000
Time [s]
0.04
0.06
0.08
0.10
0.12
2
4
6
8
10
12
14
16
N Threads