DEVELOPMENT OF A GPU ACCELERATED VISUAL TRACKING FRAMEWORK

David Concha Gomez, Raul Cabido Valladolid, Antonio Sanz Montemayor, Juan José Pantrigo
(Universidad Rey Juan Carlos) {david.concha, raul.cabido, juanjose.pantrigo, antonio.sanz}@urjc.es

Agenda

- Introduction
- 2D Articulated Tracking
- CUDA Adaptation
- 3D Tracking
- 3D Articulated Tracking

Agenda

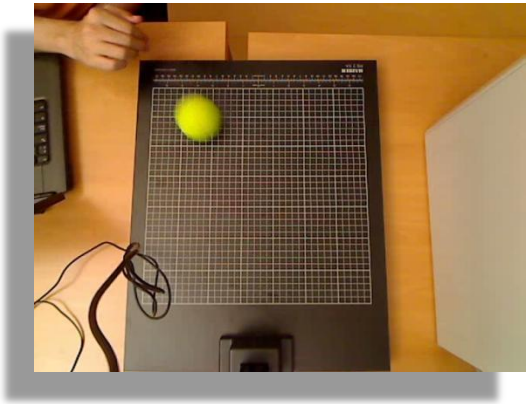
- **Introduction**
- 2D Articulated Tracking
- CUDA Adaptation
- 3D Tracking
- 3D Articulated Tracking

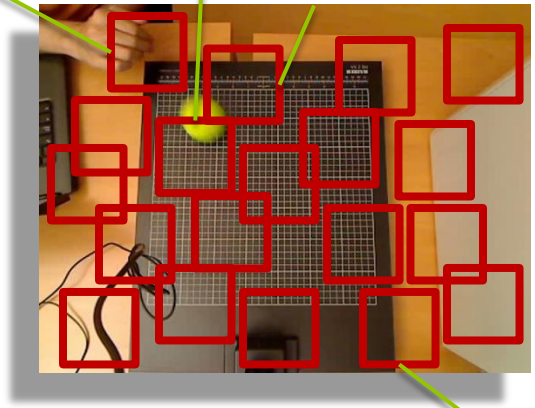
Introduction

- The goal is to build a framework that eases the development of visual tracking systems
- Visual tracking estimates the evolution of a system state through time
- Particle Filter (PF) uses a population of N estimations
  - It can be computationally expensive
  - But it is highly parallelizable

Introduction (particle filter)

- Initialization:
  - Create a new random population





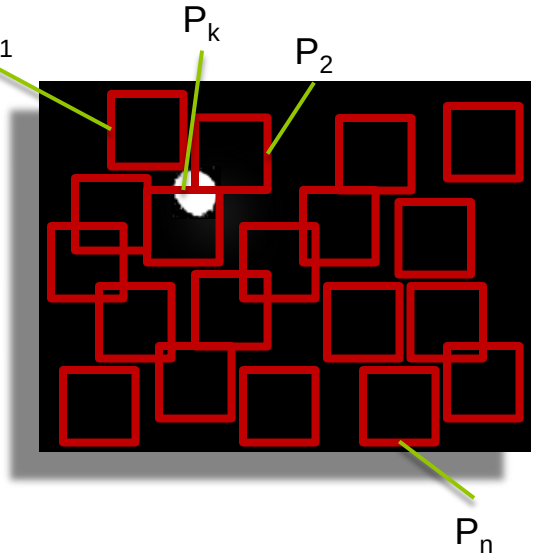
Particle Population

$P_1=(x_1,y_1,\pi_1)$
$P_2=(x_2,y_2,\pi_2)$
…
$P_k=(x_k,y_k,\pi_k)$
…
$P_n=(x_n,y_n,\pi_n)$

π_i=??

Introduction (particle filter)

- Evaluation:
  - Using the measurement model, each particle is weighted

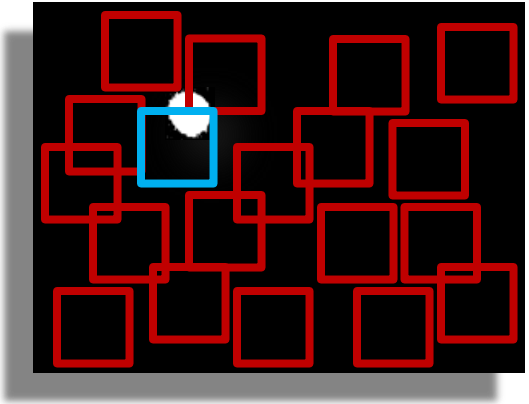


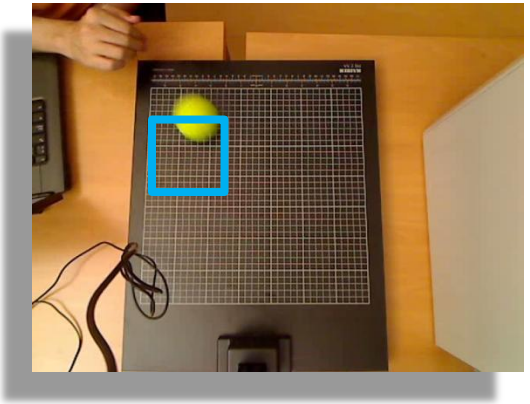
Particle Population

$P_1=(x_1,y_1,3)$
$P_2=(x_2,y_2,27)$
…
$P_k=(x_k,y_k,63)$
…
$P_n=(x_n,y_n,4)$

Introduction (particle filter)

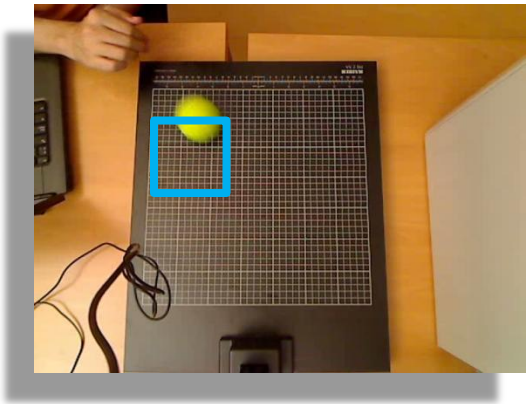
- Estimation:
  - Particle with the highest weight is selected as the best estimator at time step t



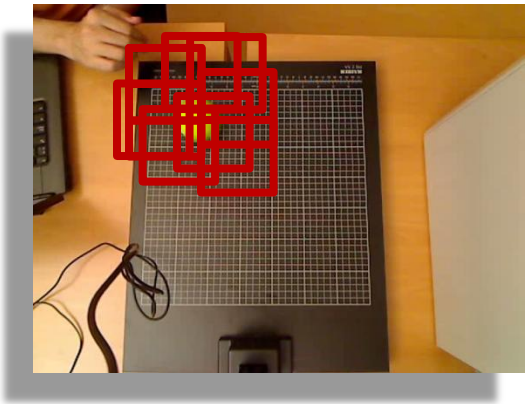


Introduction (particle filter)

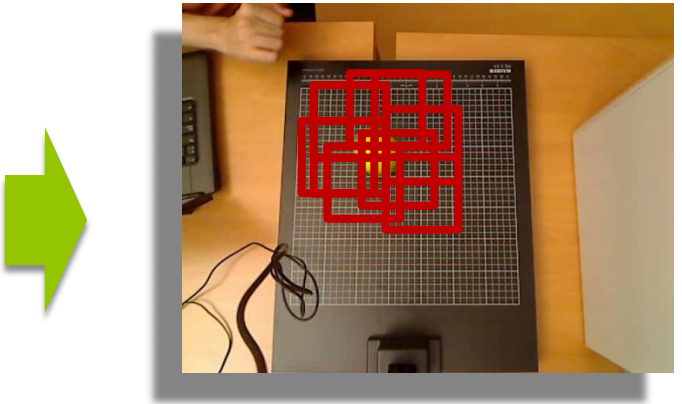
- Selection and diffusion:
  - A new population is generated based on the system estimate, and every particle is diffused to provide diversity



t



t



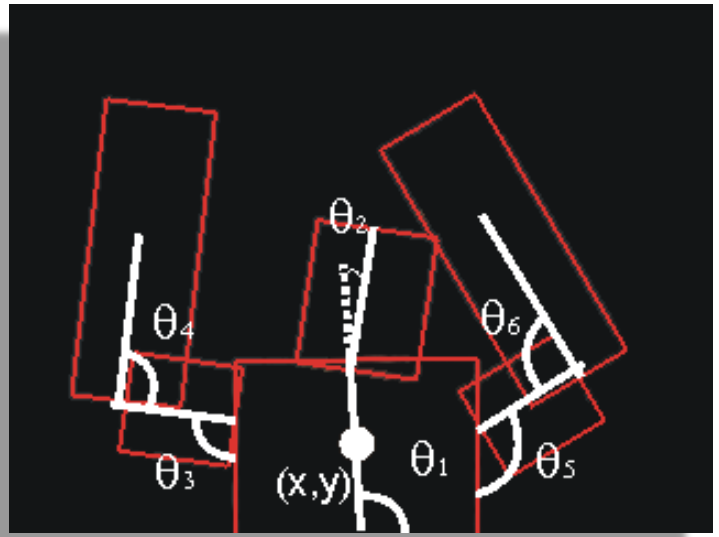
t+1

Agenda

- Introduction
- **2D Articulated Tracking**
- CUDA Adaptation
- 3D Tracking
- 3D Articulated Tracking

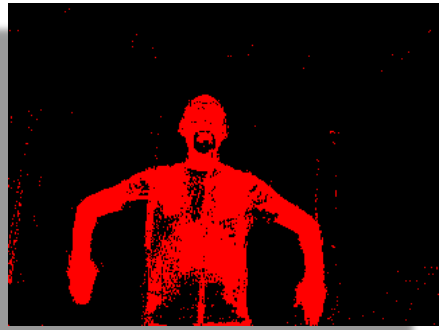
2D Articulated Tracking

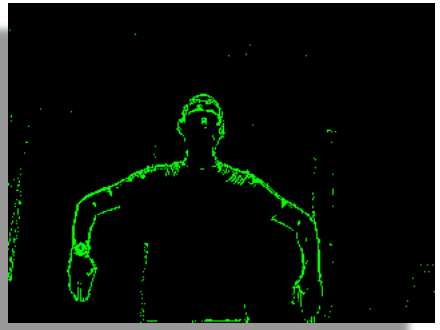
- Our 2D articulated tracking problem in 2009
  - N segments → **X**=(x, y , θ_1, θ_2, θ_3,…, θ_n)



2D Articulated Tracking

- In each frame we know:
  - The previous state of the system
  - The blob information
  - The edge information

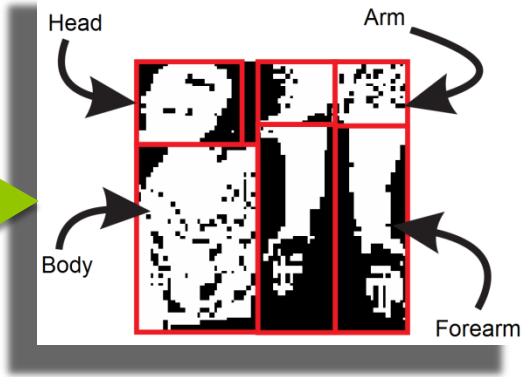




2D Articulated Tracking

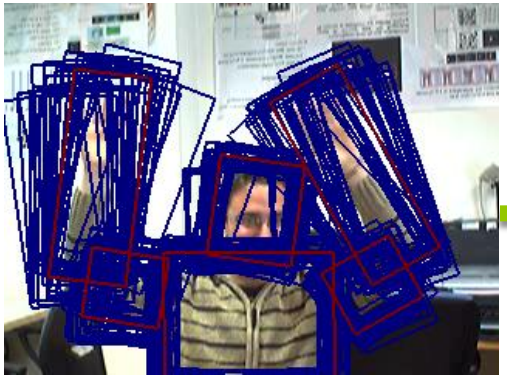
- Each particle defines a image regions that are measured and compacted in tiles (32x32)

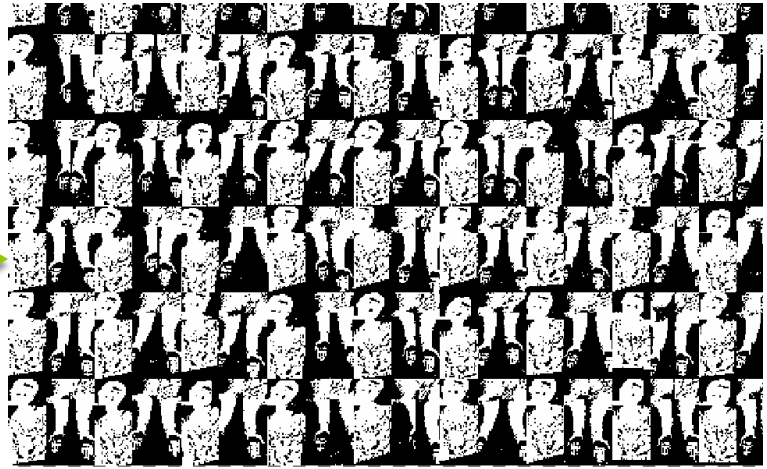
$\mathbf{X}=(x, y, \theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6)$



2D Articulated Tracking

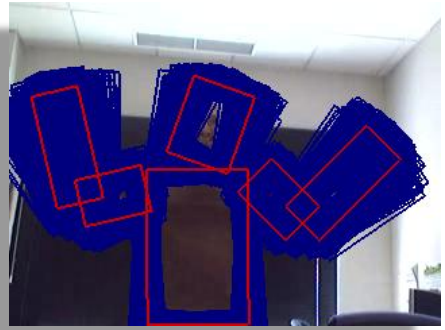
- Because the evaluation of each particle is independent on each other, It is highly parallelizable

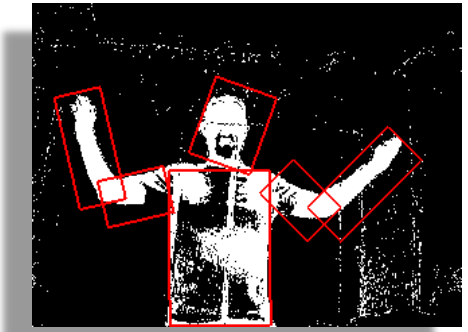




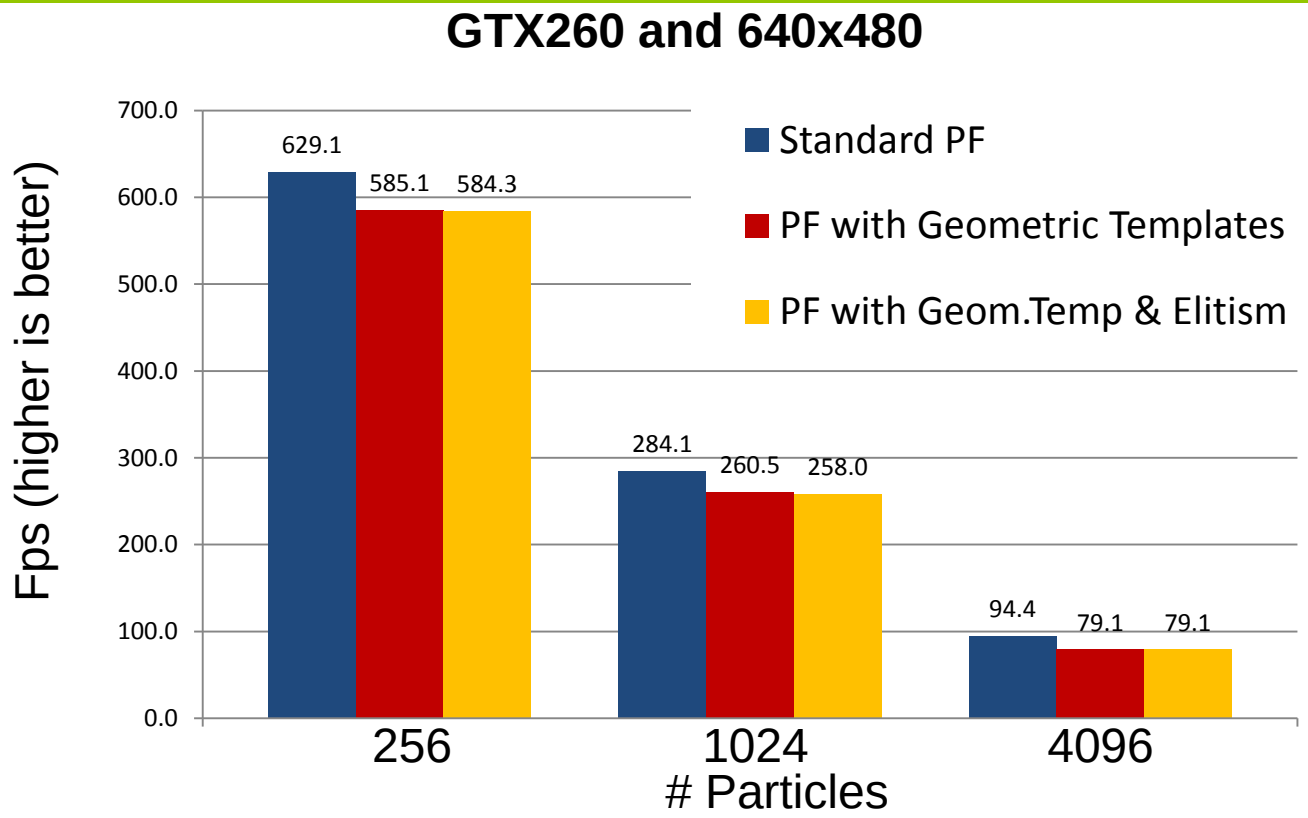
2D Articulated Tracking

- Reducing the compact texture results in the particles weights
- Reducing all the weights give the best particle
- A new population is created by a resampling technique (or any other mechanism)





2D Articulated Tracking



Agenda

- Introduction
- 2D Articulated Tracking
- **CUDA Adaptation**
- 3D Tracking
- 3D Articulated Tracking

CUDA Adaptation

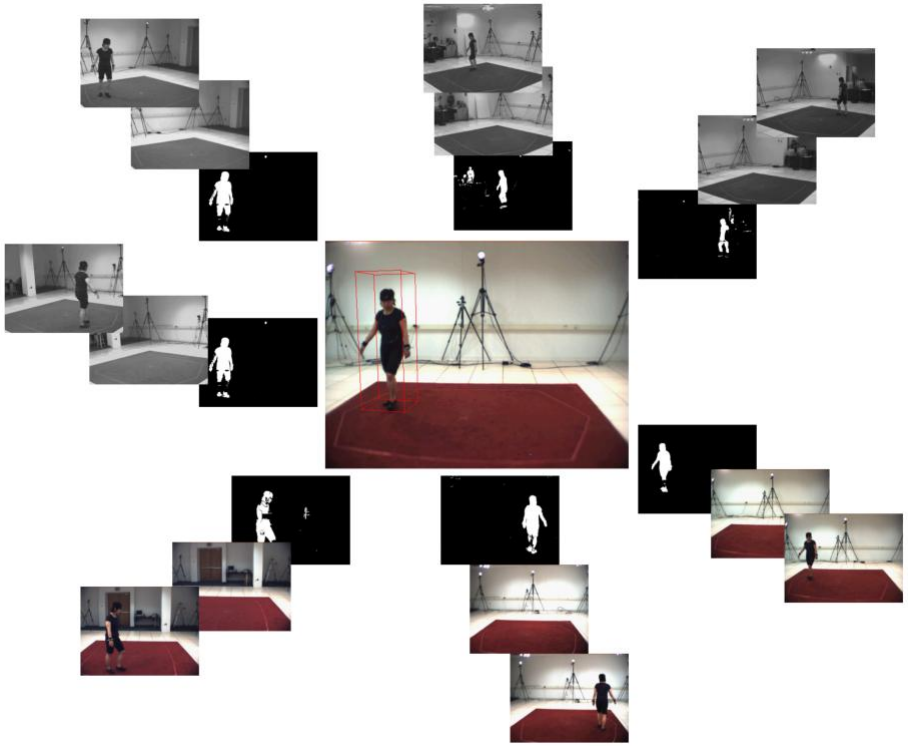
- OpenGL + Shaders:
  - Good performance
  - Reduction: requires power of 2 tiles (i.e. 32x32)
- OpenGL for rasterizer, CUDA for Reduction:
  - Reduction: more efficient for arbitrary tile size
  - Interoperability impact performance
- CUDA Rasterizer and Reduction:
  - For high number of tiles it is as efficient as OpenGL+Shader without its drawbacks

Agenda

- Introduction
- 2D Articulated Tracking
- CUDA Adaptation
- **3D Tracking**
- 3D Articulated Tracking

3D Visual tracking

- Calibrated and synchronized system
- In a multiview system, many images simultaneously
- Increased computational cost of evaluation

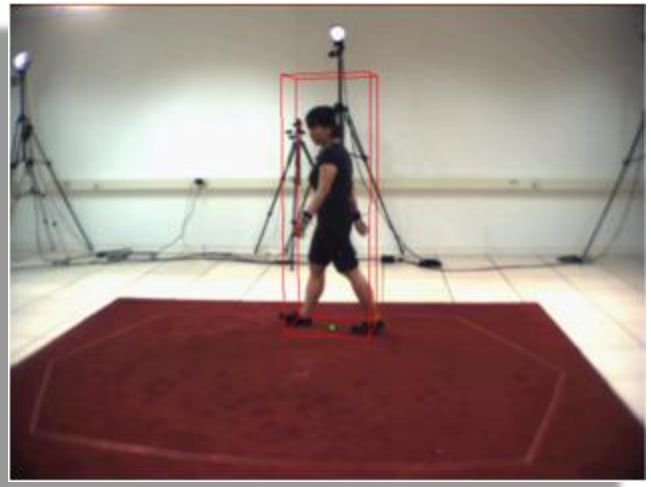


3D Visual tracking

- 3 DOF (x,y,z) to a 3D Volume (8 vertices)

(x_k,y_k,z_k)

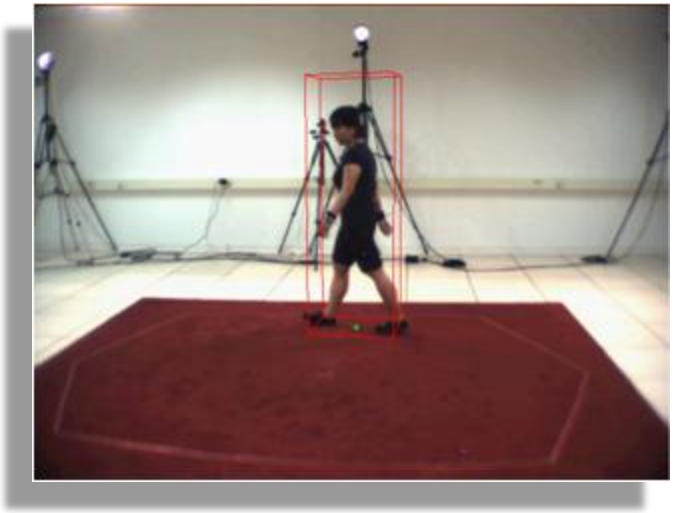
8 vertices



3D Visual tracking

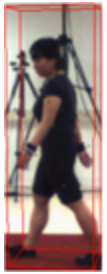
- 3D Volume (8 vertices) to an Axis Aligned Bounding Box

8 vertices



AAB
B

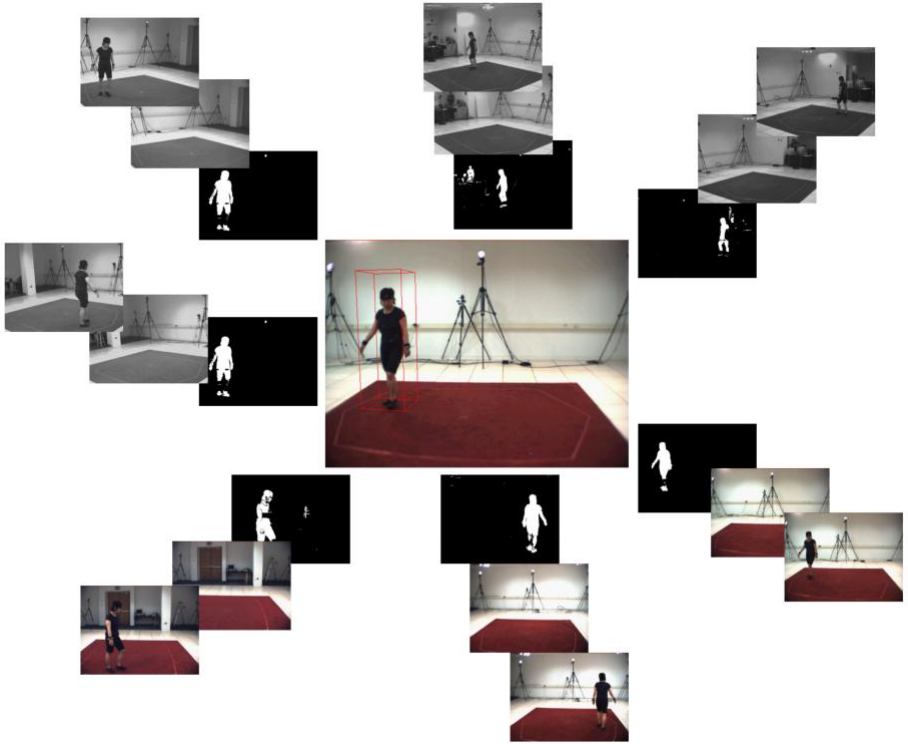
For each camera



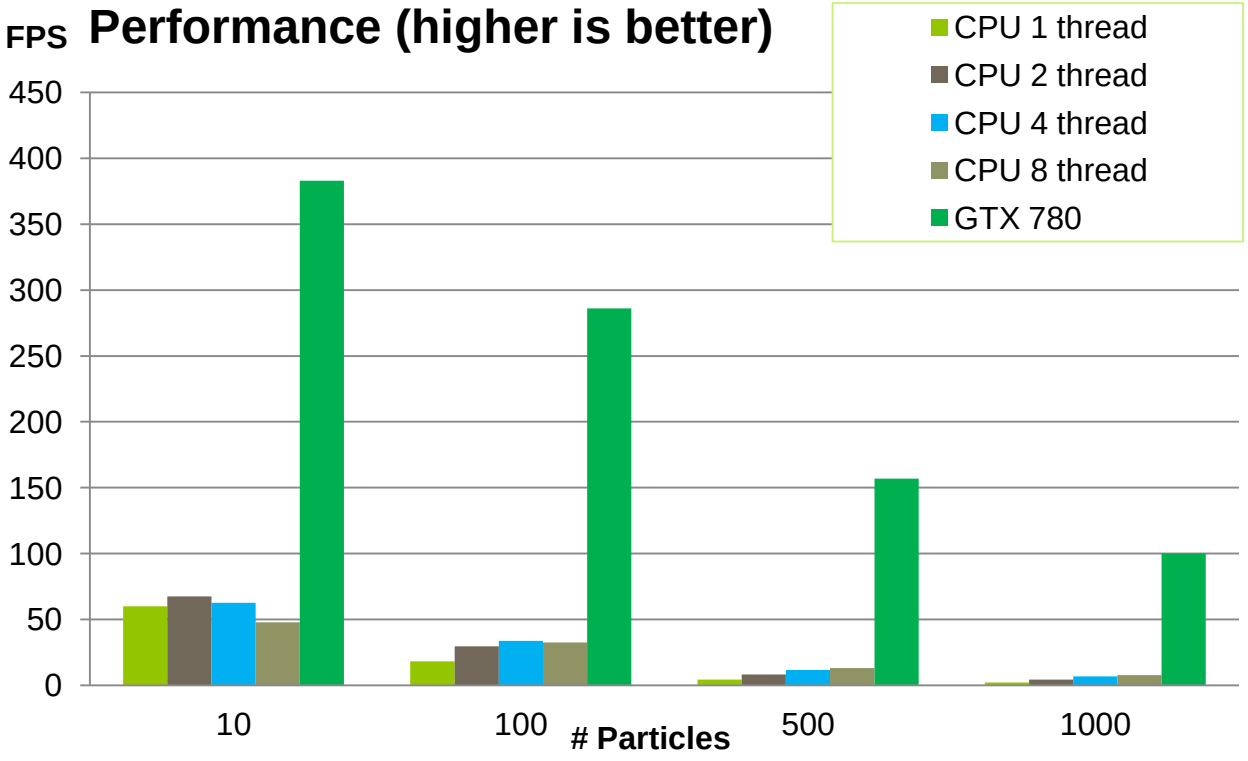
$$\pi_{v,i}^{t} = \sum_{k,l=x_0,y_0}^{x_m,y_m} I^{t}(k,l)$$

3D Visual tracking

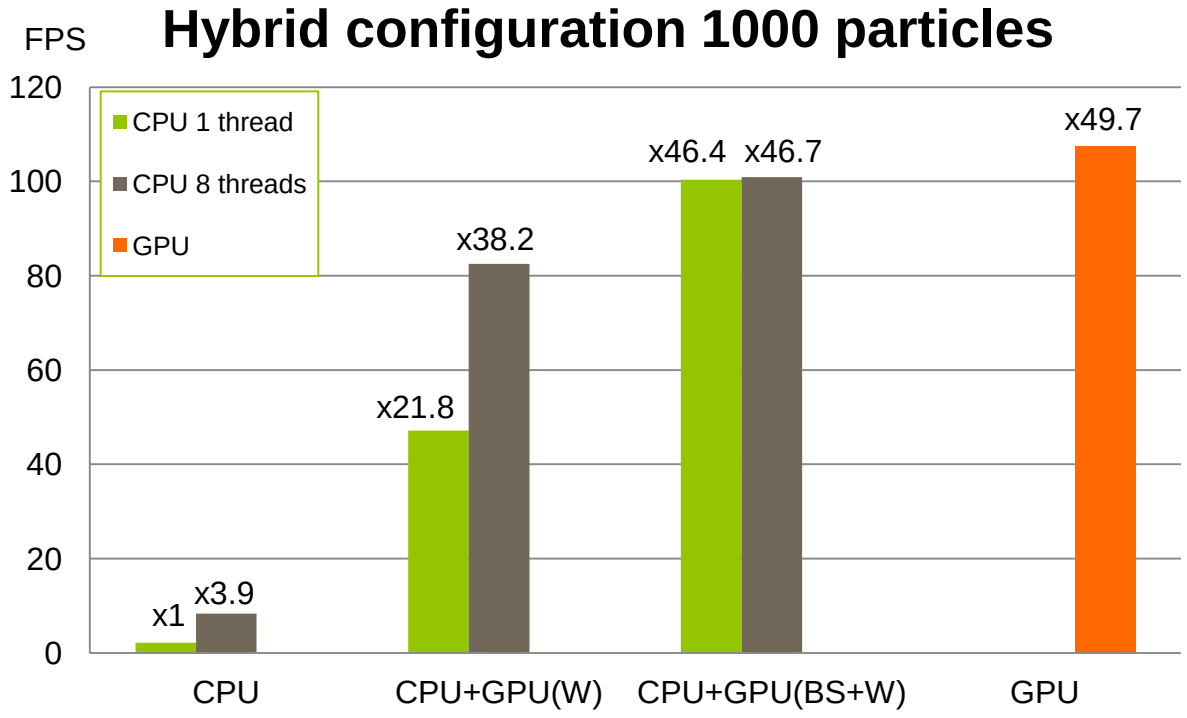
- One thread block obtain the weight of one particle
- The AABB approach simplified computation but induce some error
- The remaining PF stages are similar



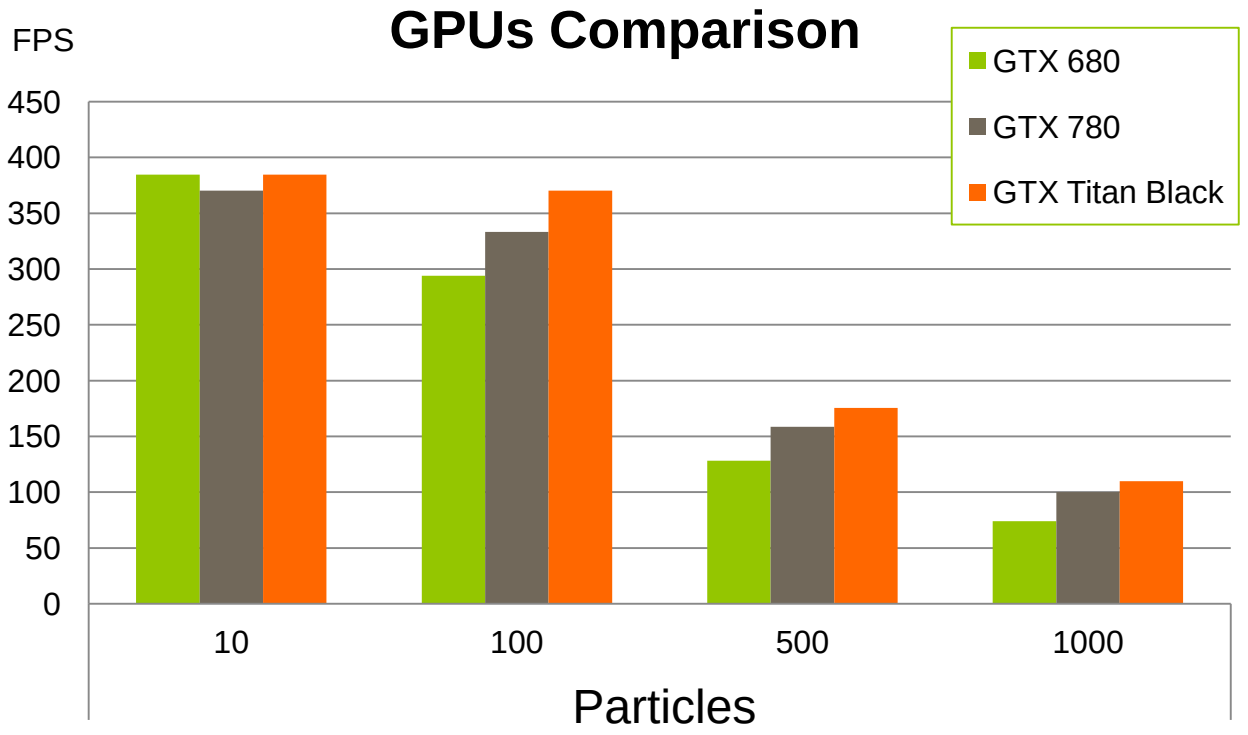
3D Visual tracking



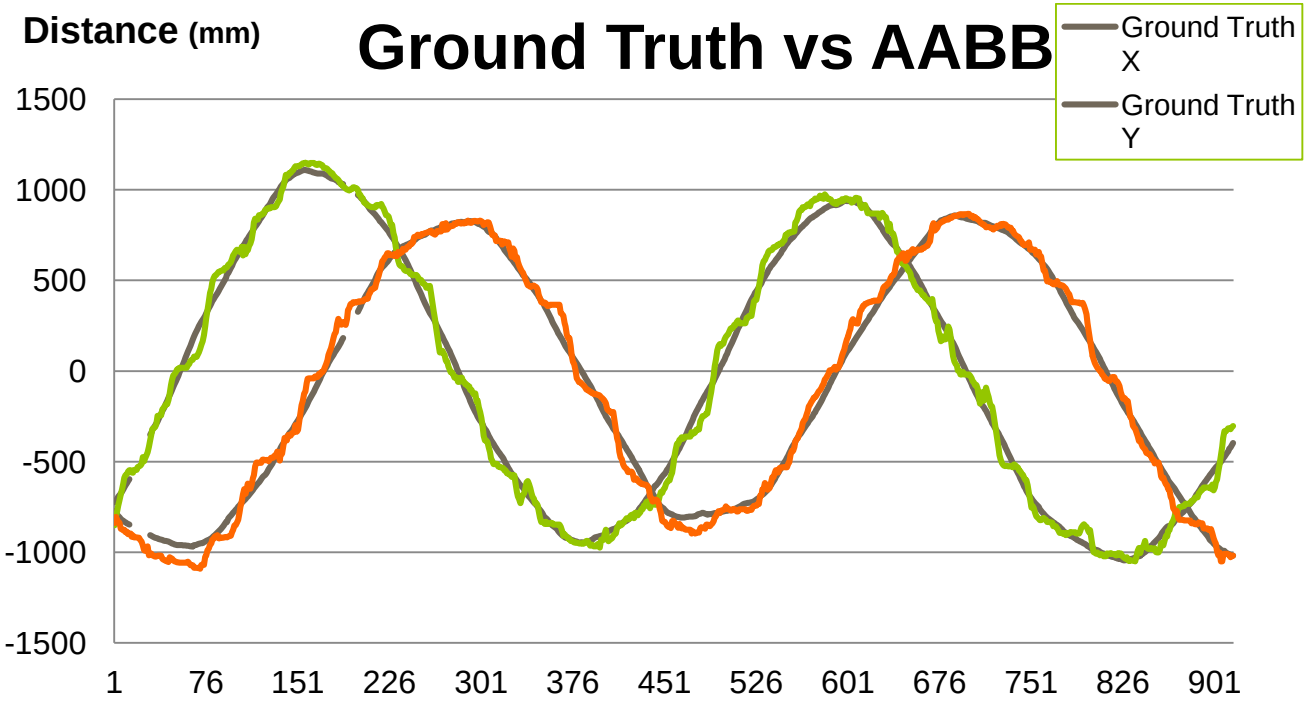
3D Visual tracking



3D Visual tracking



3D Visual tracking

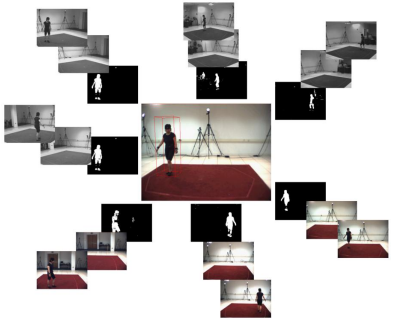


Agenda

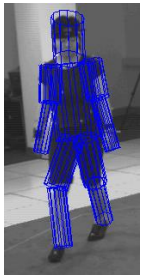
- Introduction
- 2D Articulated Tracking
- CUDA Adaptation
- 3D Object Tracking
- **3D Articulated Tracking (work in progress)**

3D Articulated Tracking

- Articulated body model with more than 30 DOF
  - larger particle population
  - very computationally expensive evaluation
- Each particle is represented by multiple regions (articulated) in many images (3D)
- Regions no longer axis aligned and cylindrical

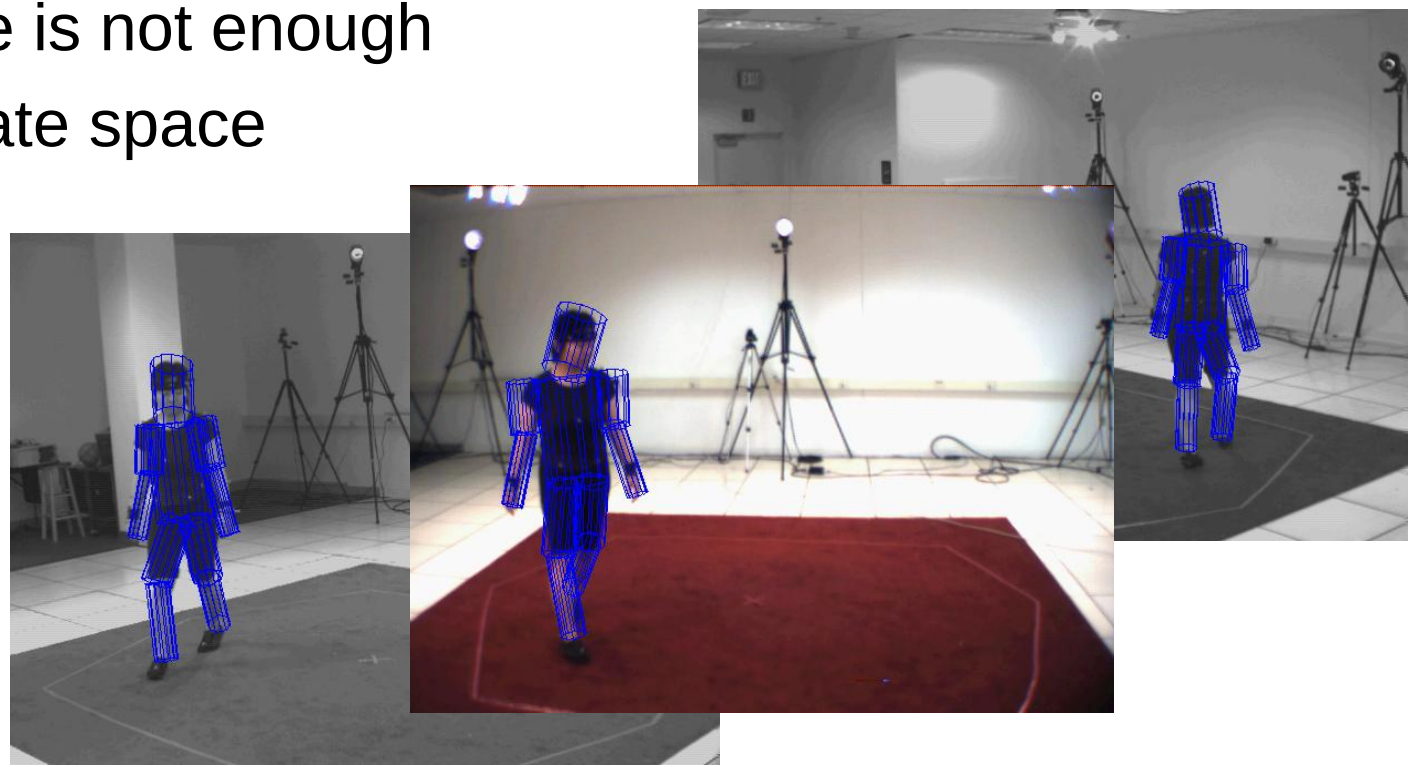


+



3D Articulated Tracking

- PF alone is not enough
- Huge state space



Conclusions

- Particle filter is a scalable and parallel friendly method.
- Its weight computation is the most demanding stage
- 2D articulated tracking
  - Up to 630 fps on GPU using shaders (256 particles)
- 3D object tracking
  - Improvement of almost x50 against CPU
  - Good accuracy with 100 particles running at 300 fps.
- Now combining both ideas for the 3D articulated problem

Questions

2004
SIGGRAPH poster
First PF weighting on GPU

2006
SIGGRAPH poster
Complete PF on GPU

2009
GPUTA 2009
2D articulated tracking

2012
GTC
CUDA Rasterizer

2013
GTC
3D Object Tracking
(Kinect+GPU)

Thank you for your attention

2014
GTC
High-Performance Multiview 3D Tracking
Using Particle Filters

2014
JRTIP
CPU vs GPU
Performance study of PF

2015-16 ??
3D articulated