Automatic Simplification of Particle System Dynamics

David O'Brien   Susan Fisher   Ming C. Lin

http://gamma.cs.unc.edu/SLOD

Outline

- Introduction and Motivation
- Simulation Level of Detail (SLOD)
  - Definitions and Parameters
  - Creation and Maintenance through Physically-based Spatial Subdivision
- Adding Flexibility
  - Regions of Interest (ROI)
  - Switching ROIs
- Results
- Future Work

Introduction: Motivation

- Real-time VEs & Video Games
  - Increasingly use particle systems and physically based simulations
  - Despite recent advances, still can not simulate complex dynamical systems in real time.
- Goal:
  - Reduce Cost of Dynamics Computations through Simulation Acceleration Techniques
  - Analogous to Model Simplification & Rendering
  - Maintain Consistent Frame Rates for Simulations

Previous Work

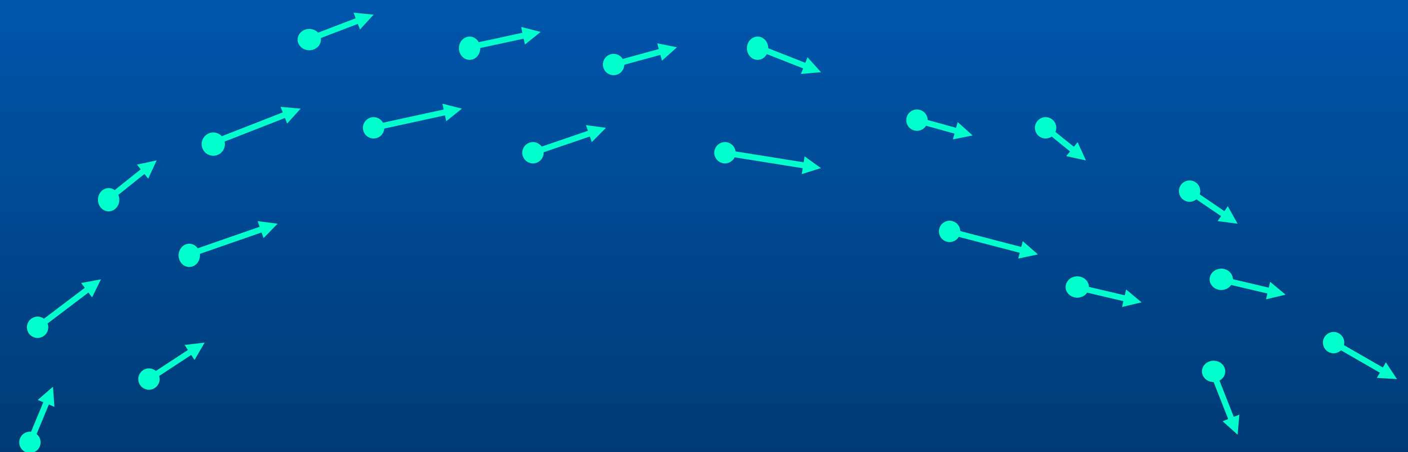
- Model Simplification & Simulation Levels of Detail
  - Fumkhouser et al created a generic framework for LOD and rendering techniques to maintain real-time frame rates
  - Carlson & Hodgins created Simulation Level of Details for groups of legged creatures.
  - Chenney et al proposed view-dependent culling of dynamic systems

Particle Systems

- Why Particle Systems?
  – Natural Phenomena, Modeling, and Group Behavior
- Field System: Force can be applied to a particle in constant time
  – Gravity, Drag, Turbulence
  – Linear with respect to number of particles
- N-Body Simulation:
  – Astronomical Simulation, Potential Calculations
  – n^2 interactions, reduced to *O(n lg n)* with heuristics such as Barnes-Hut Algorithm

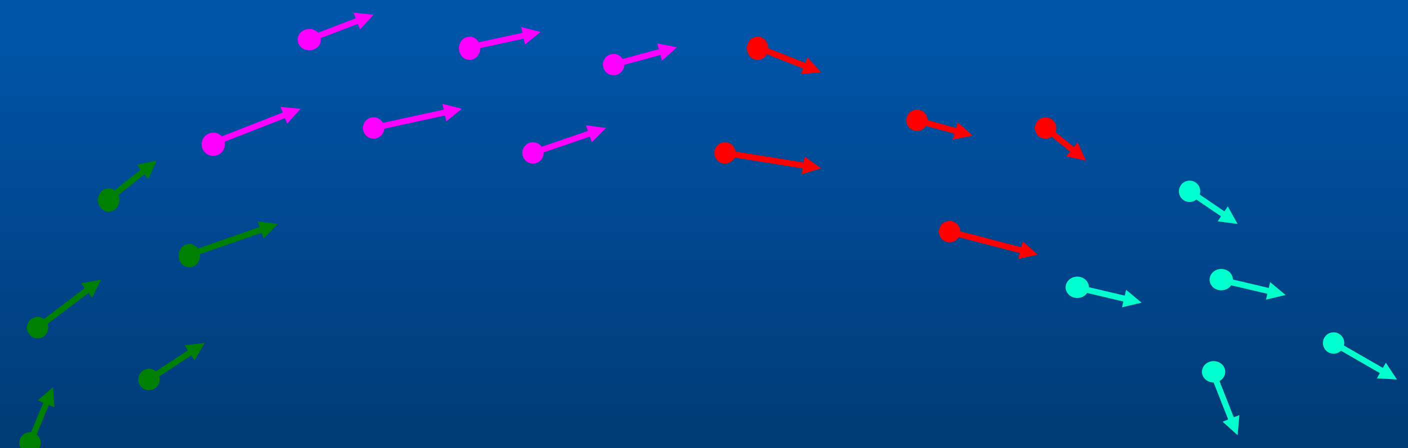
Simulation Level of Detail (SLOD)

Simplifying a Particle System:



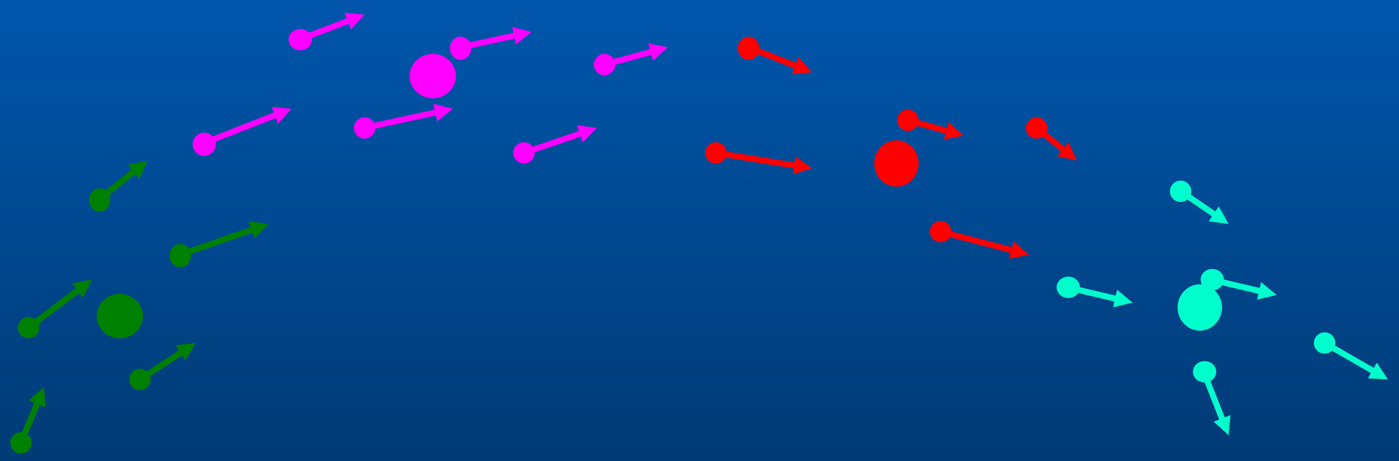
Simulation Level of Detail (SLOD)

Group into Clusters:



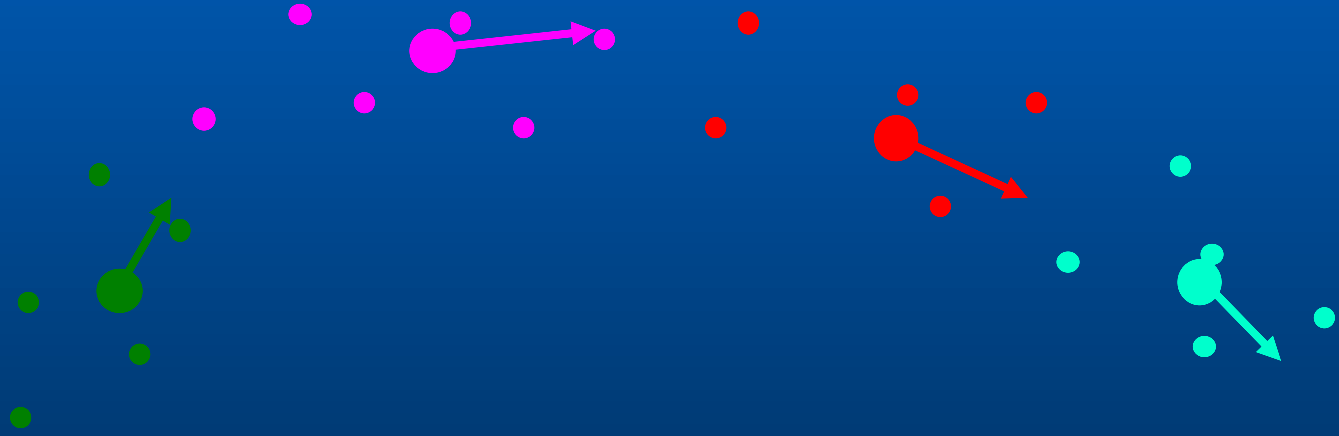
Simulation Level of Detail (SLOD)

Calculated Weighted Center of Mass Position:



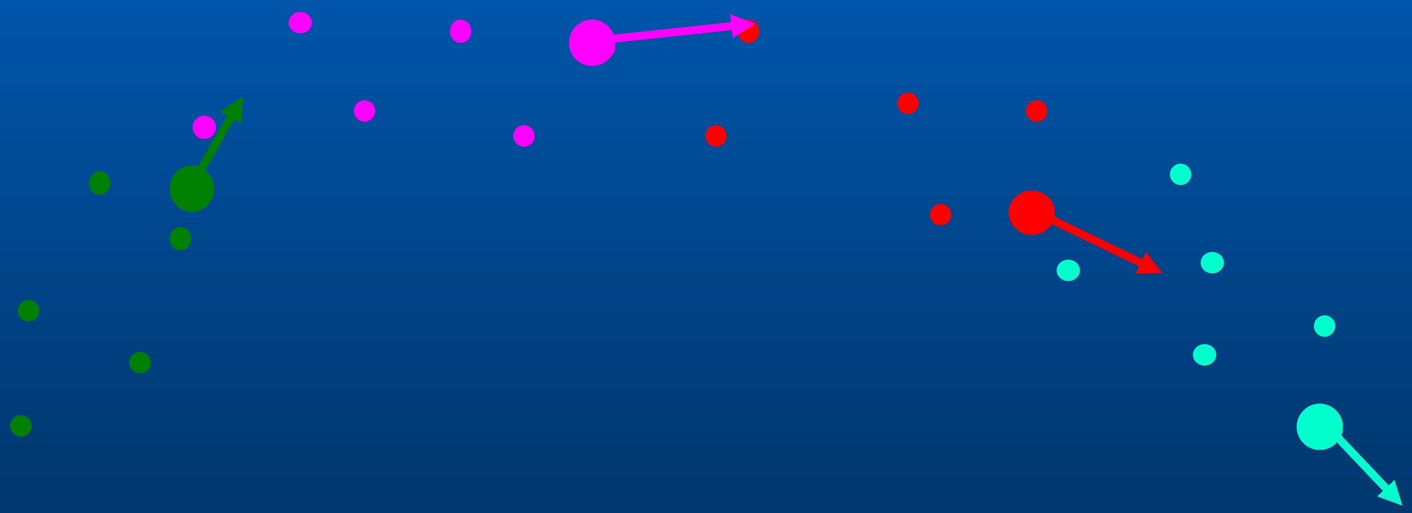
Simulation Level of Detail (SLOD)

- Calculated Weighted Center of Mass Velocity:



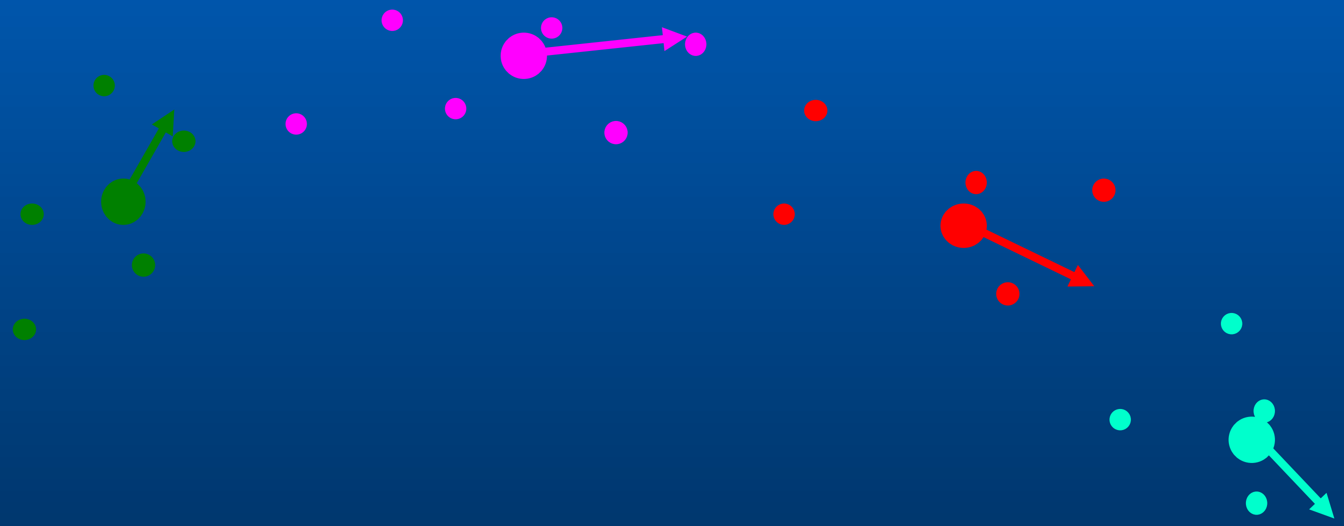
Simulation Level of Detail (SLOD)

- Apply Dynamics Engine to the just the Center of Masses:



Simulation Level of Detail (SLOD)

Update particles to match movement of Center of Masses:



Simulation Level of Detail (SLOD)

Given a Particle Cluster C consisting of n particles:

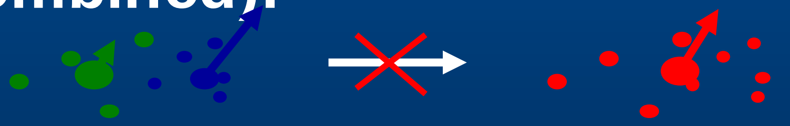
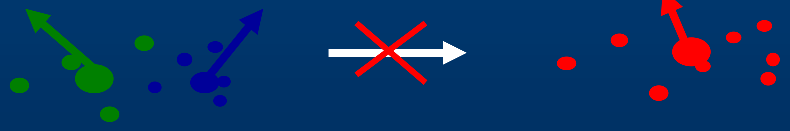
1. Computer Position P_{com} and Velocity V_{com}

$$P_{com} = \frac{\sum_{i=1}^{n} m_i P_i}{\sum_{i=1}^{n} m_i} \qquad V_{com} = \frac{\sum_{i=1}^{n} m_i V_i}{\sum_{i=1}^{n} m_i}$$

where, m_i, P_i and V_i are mass, position and velocity of i^{th} particle

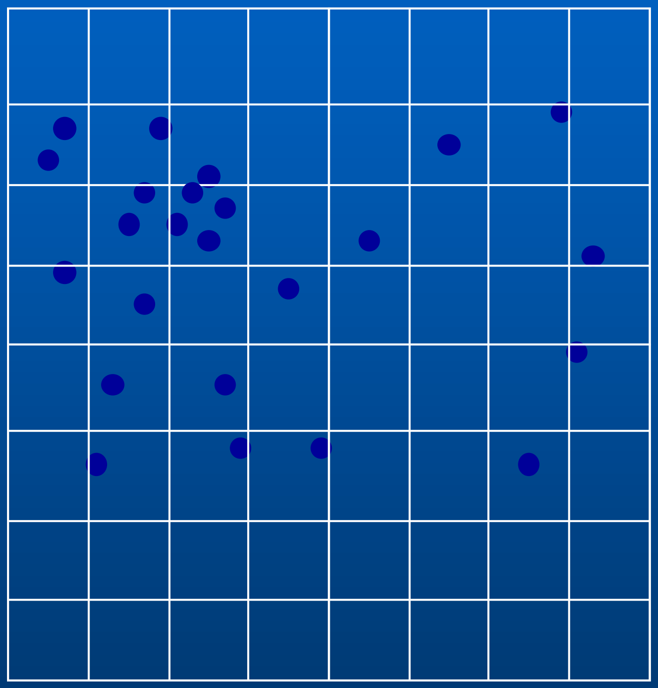
2. Update CoM using standard particle dynamics.
3. Apply change in P_{com} and V_{com} to all particles.

SLOD Parameters

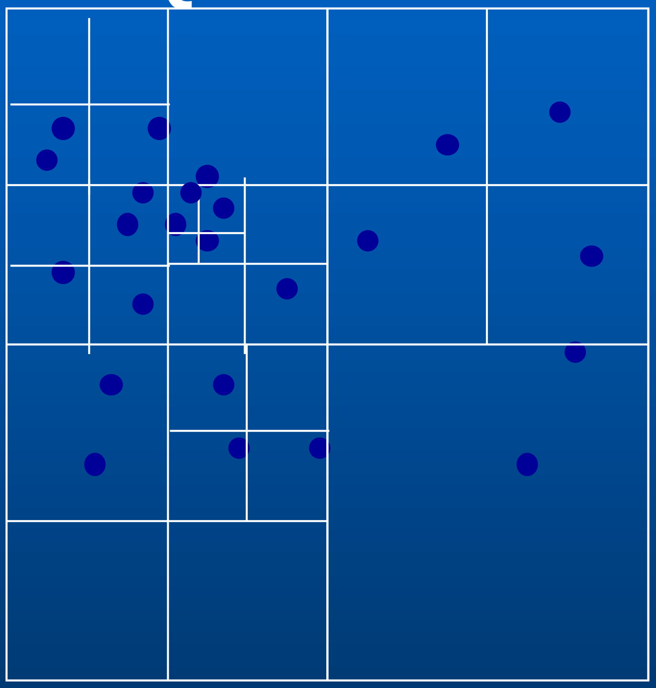
- Main Parameters *(used to create clusters)*:
  - <u>Cluster Size</u>: maximum number of particles per cluster
  - <u>Cluster Breadth</u>: maximum spatial size of a cluster
- Secondary Parameters
  (when clusters can be combined):
  - Velocity Ratio: 
  - Relative Angle: 

How to Cluster?

Uniform Grids:



Quad-Trees:

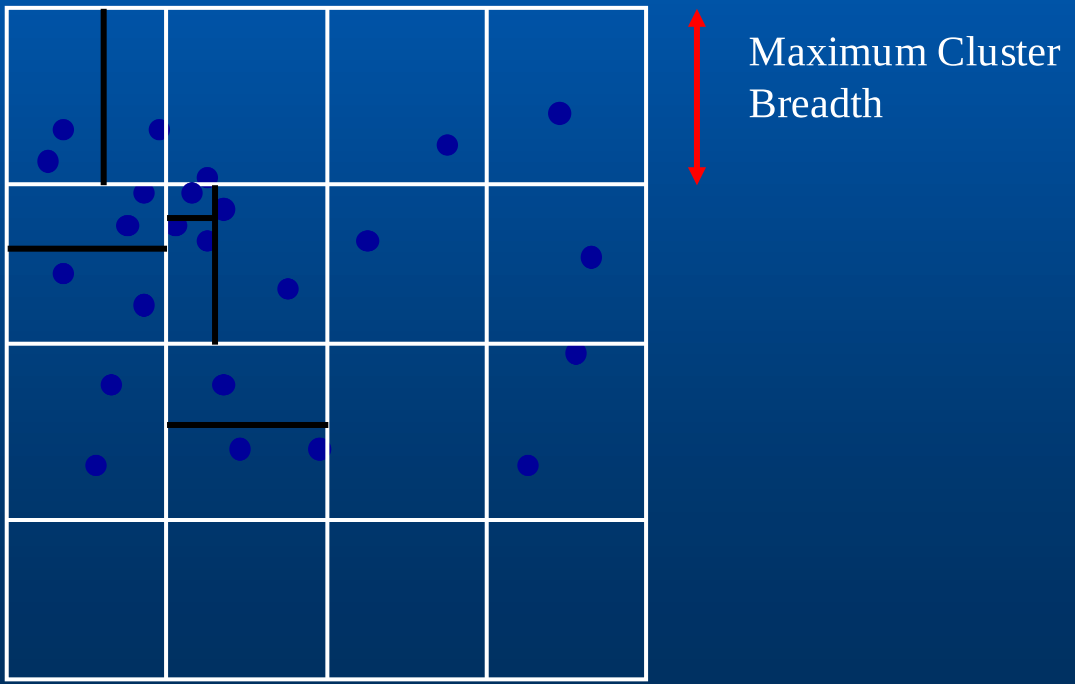


- Fast placement+lookup
- Uneven particle sizes

- Adaptive, good clustering
- O(n lg n) cost too high

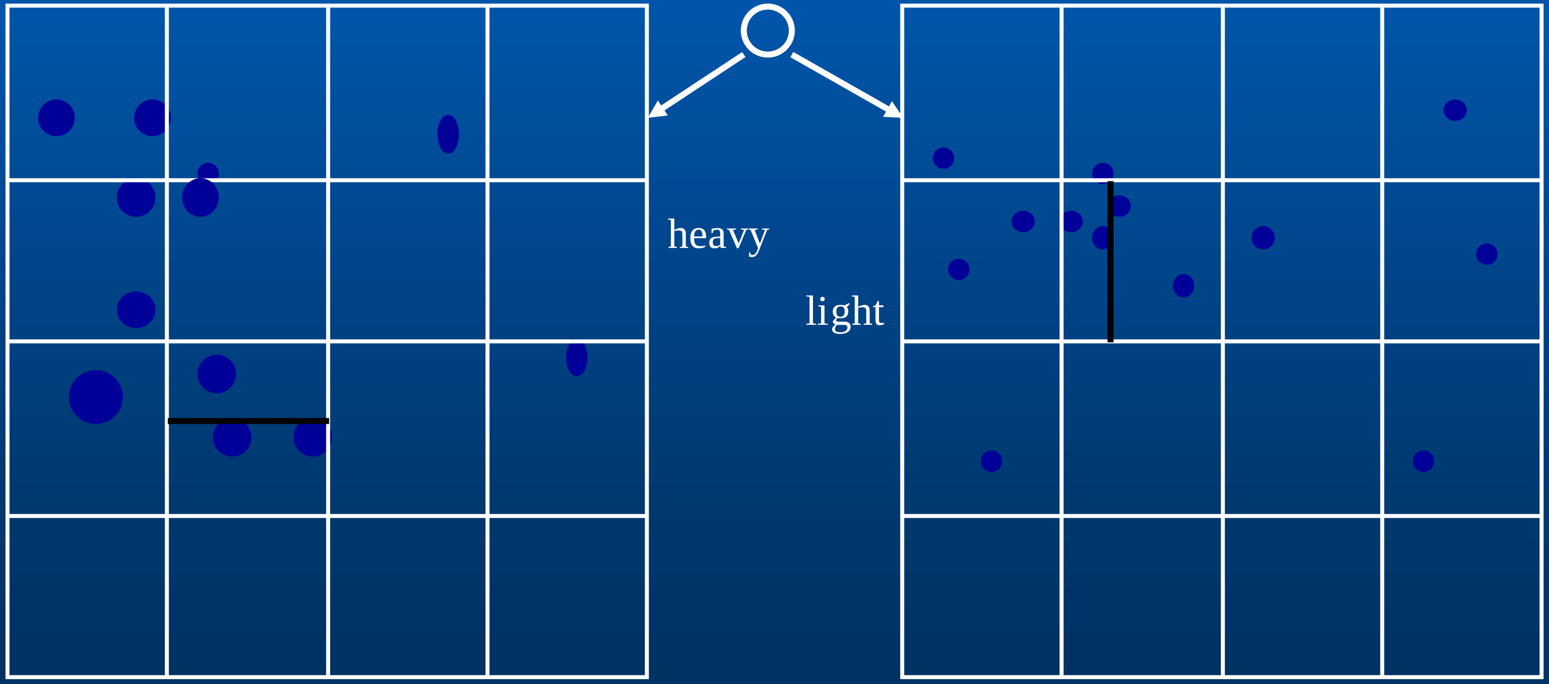
Physically-based Hybrid Subdivision

- Base is a uniform grid
  - Give good insertion and query speed
- When needed, subdivide as a Kd-tree.

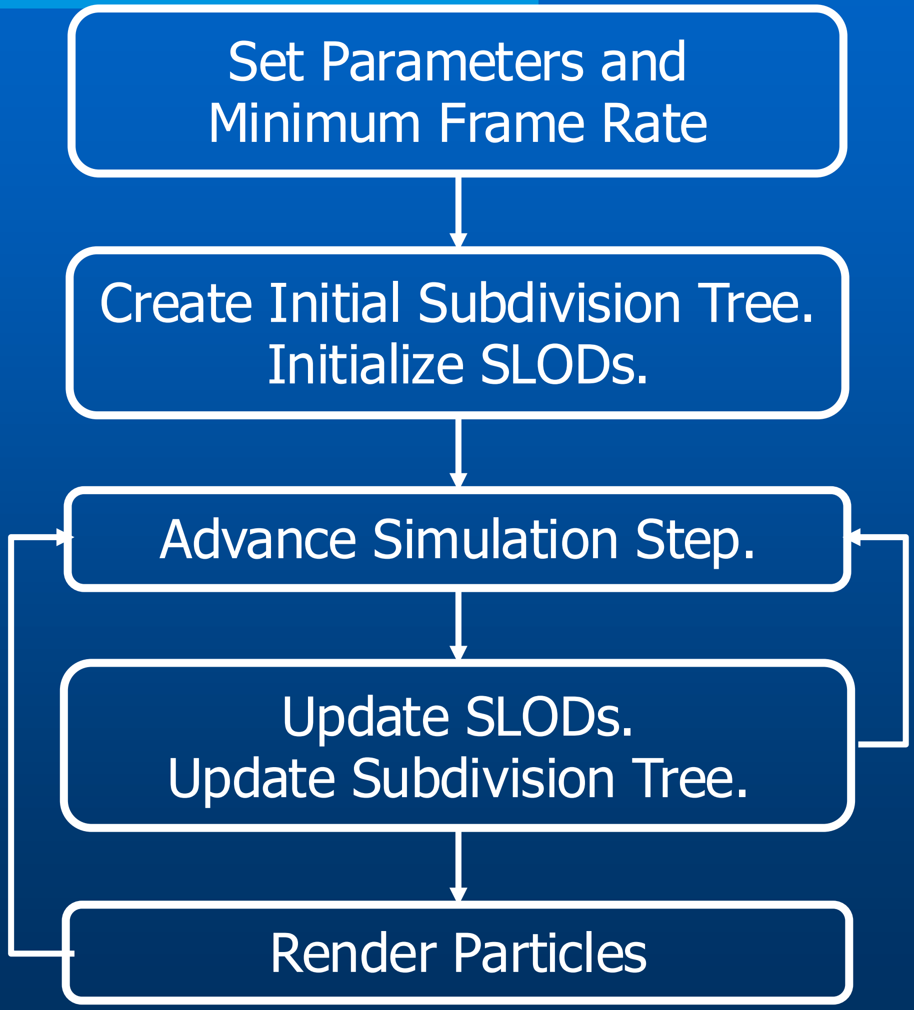


Physically-based Hybrid Subdivision

- **Why Physically-based?**
  - We can subdivide on a physical parameter as well
  - Usually this is done only at the top of the SD hierarchy



SLOD System Architecture

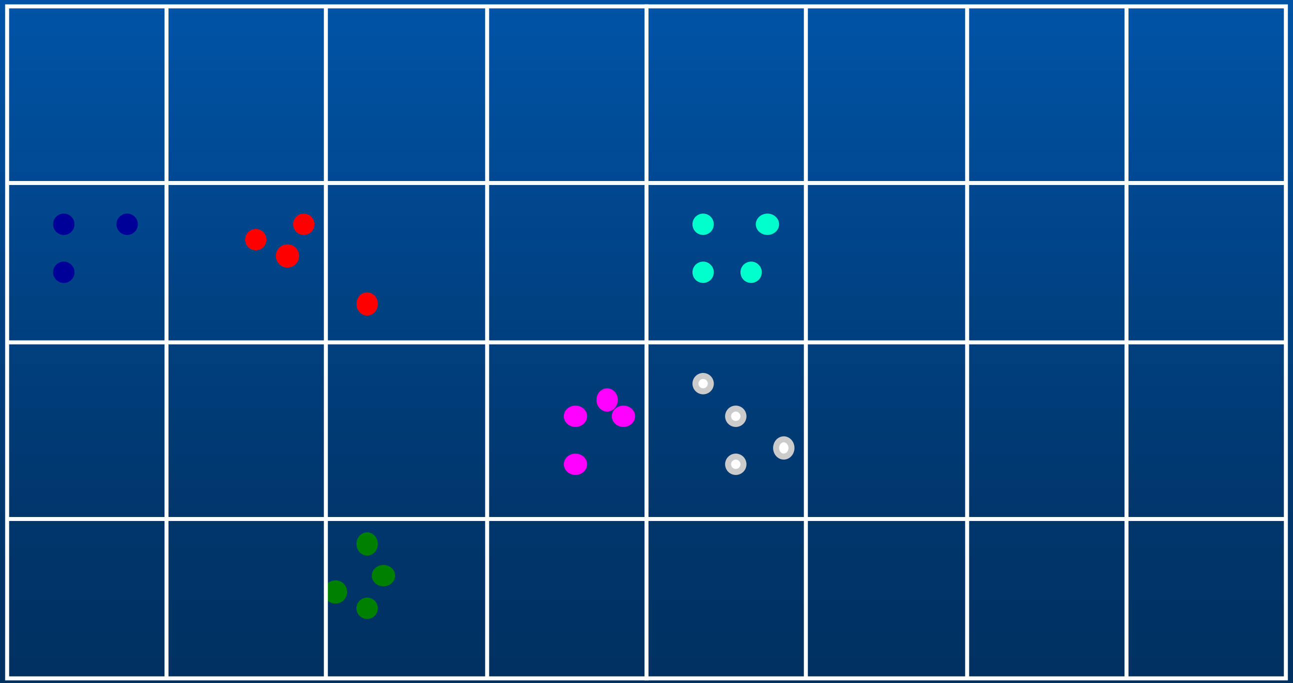


Efficient SLOD Updating

- Can not rebuild the Subdivision Tree on each Simulation Step.
- After each simulation step:
  - Remove and Reinsert each cluster that has moved out of its cell.
  - When inserting, merge nearby clusters when possible.
  - Prune the tree of unnecessary empty cells
  - Insert new particles into nearby clusters.
    - If clusters become too large, split them.
- Always taking into account changes in SLOD parameters

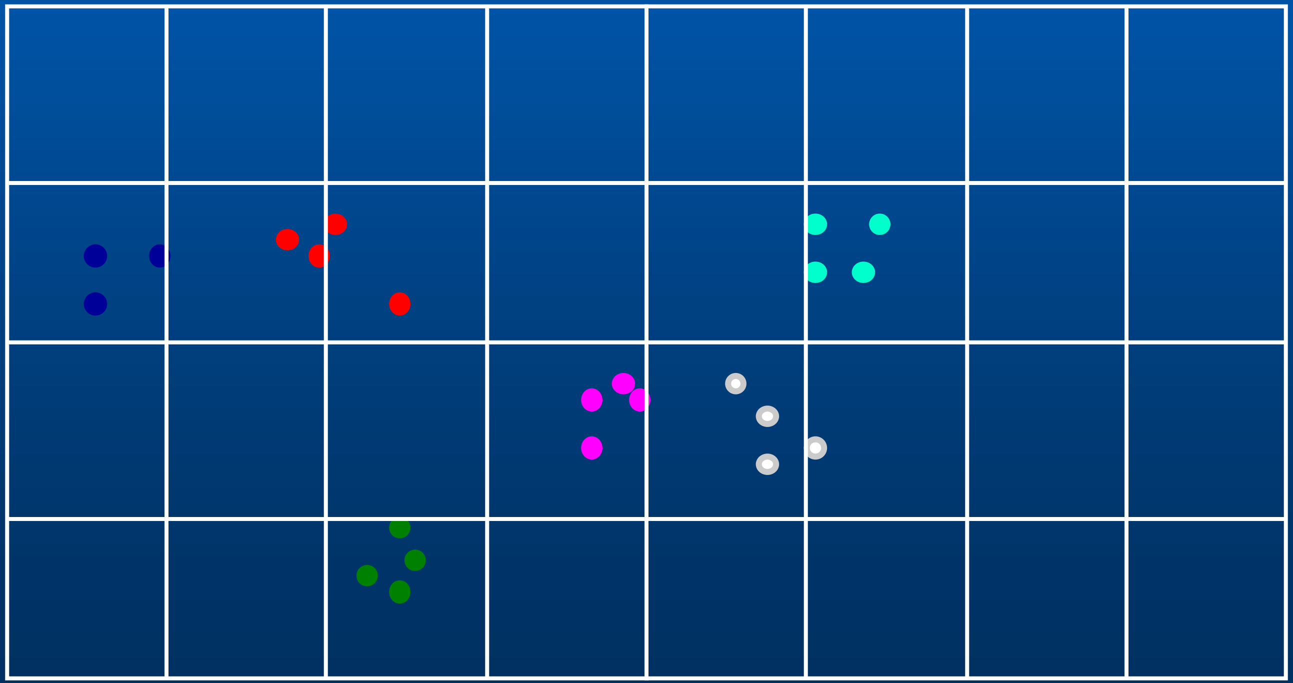
Smooth SLOD Switching

- Only update clusters when they move out of their cell.
  - Max Cluster Size switches from 4 to 2



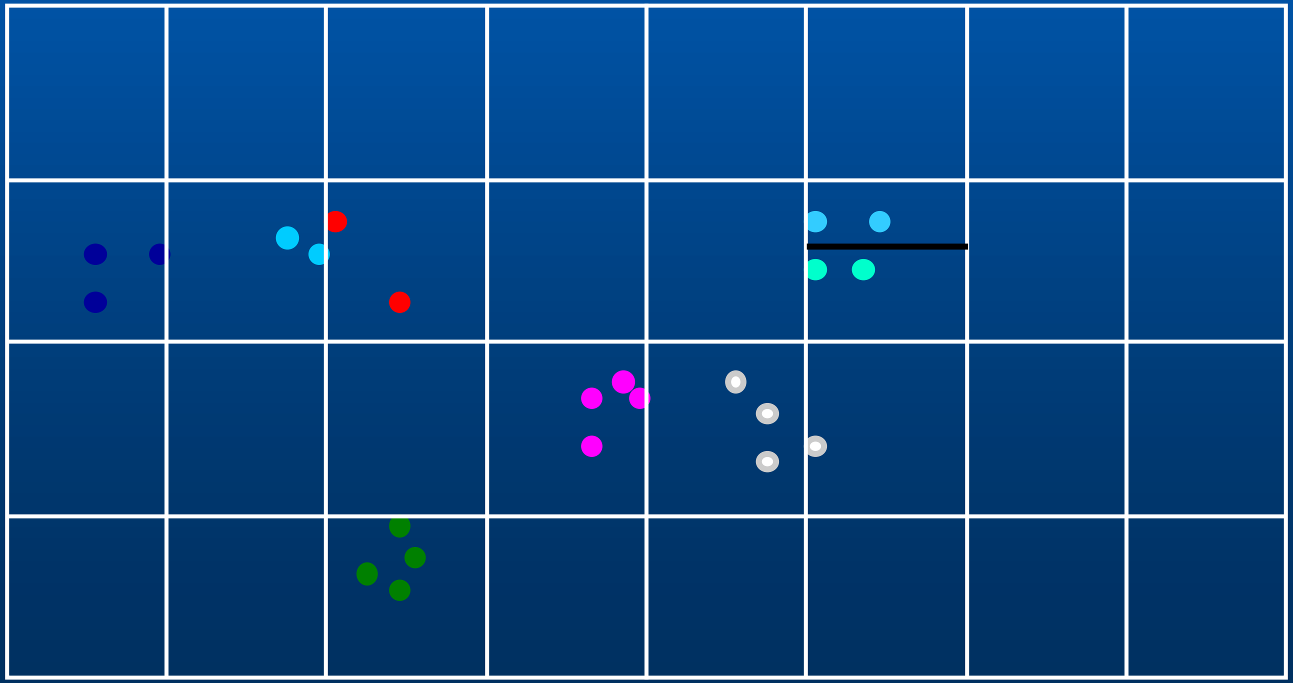
Smooth SLOD Switching

- **Only update clusters when they move out of their cell.**
  - **Max Cluster Size switches from 4 to 2**



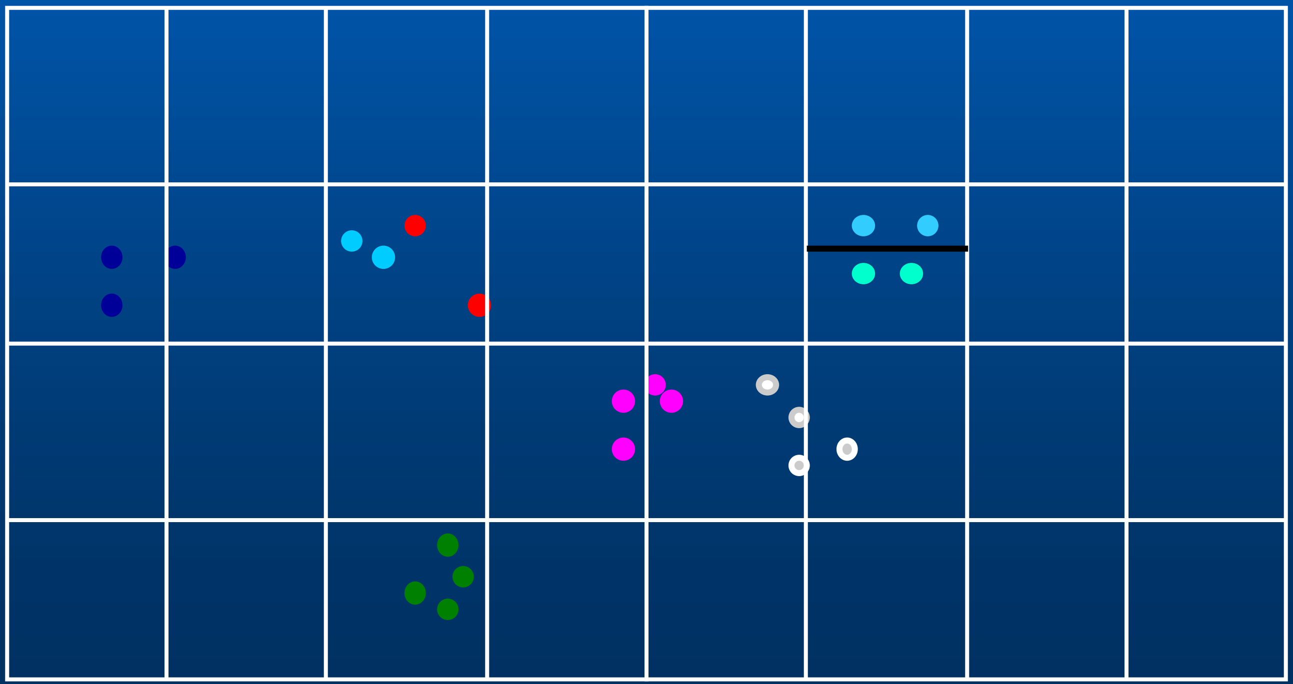
Smooth SLOD Switching

- Only update clusters when they move out of their cell.
  - Max Cluster Size switches from 4 to 2



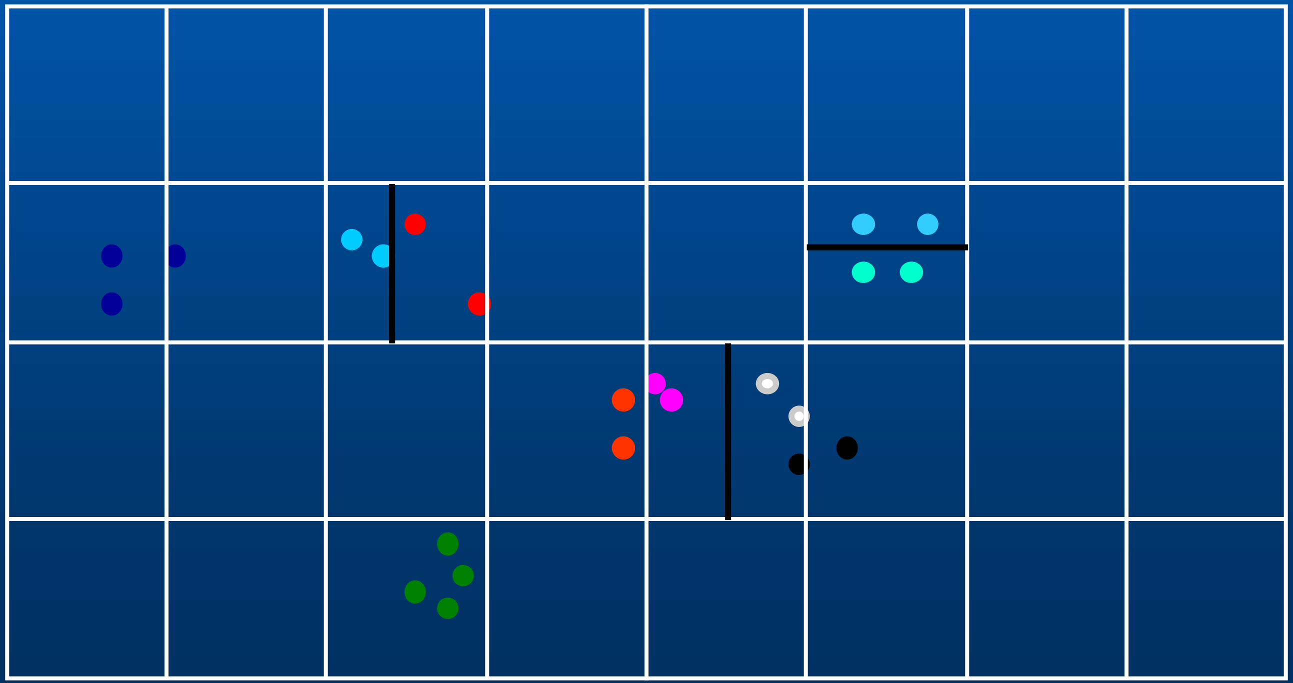
Smooth SLOD Switching

- Only update clusters when they move out of their cell.
  – Max Cluster Size switches from 4 to 2



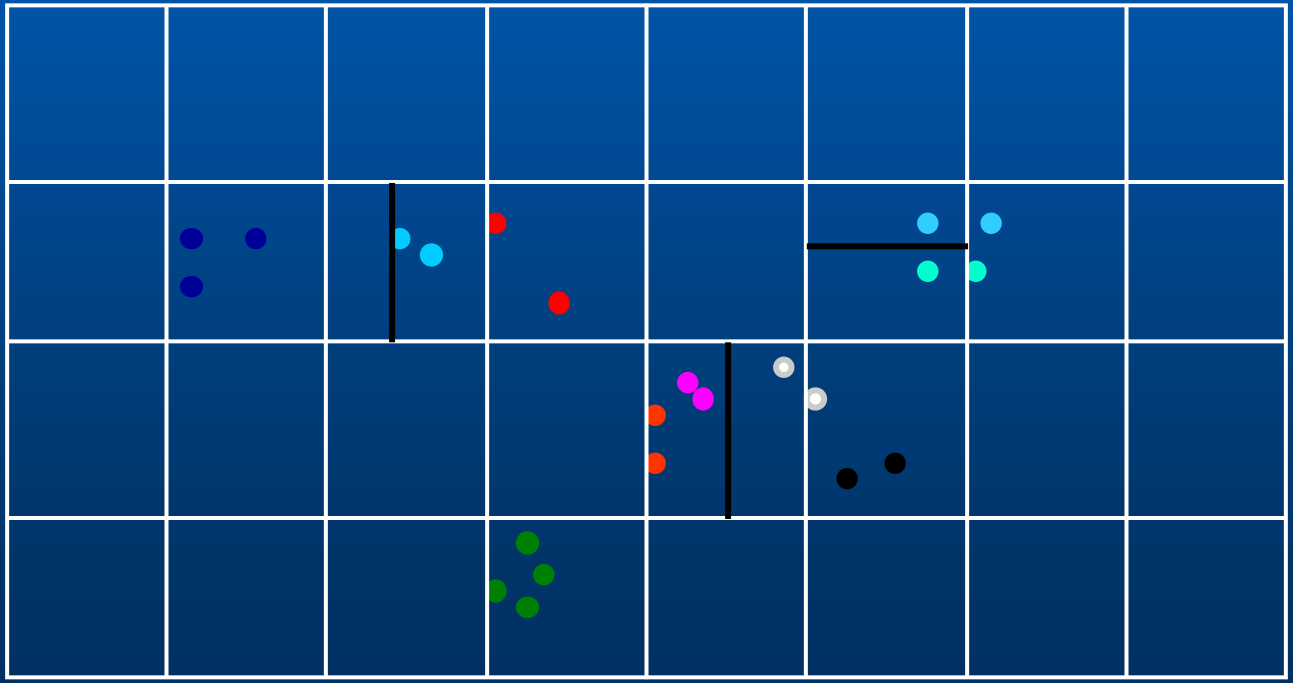
Smooth SLOD Switching

- Only update clusters when they move out of their cell.
  - Max Cluster Size switches from 4 to 2



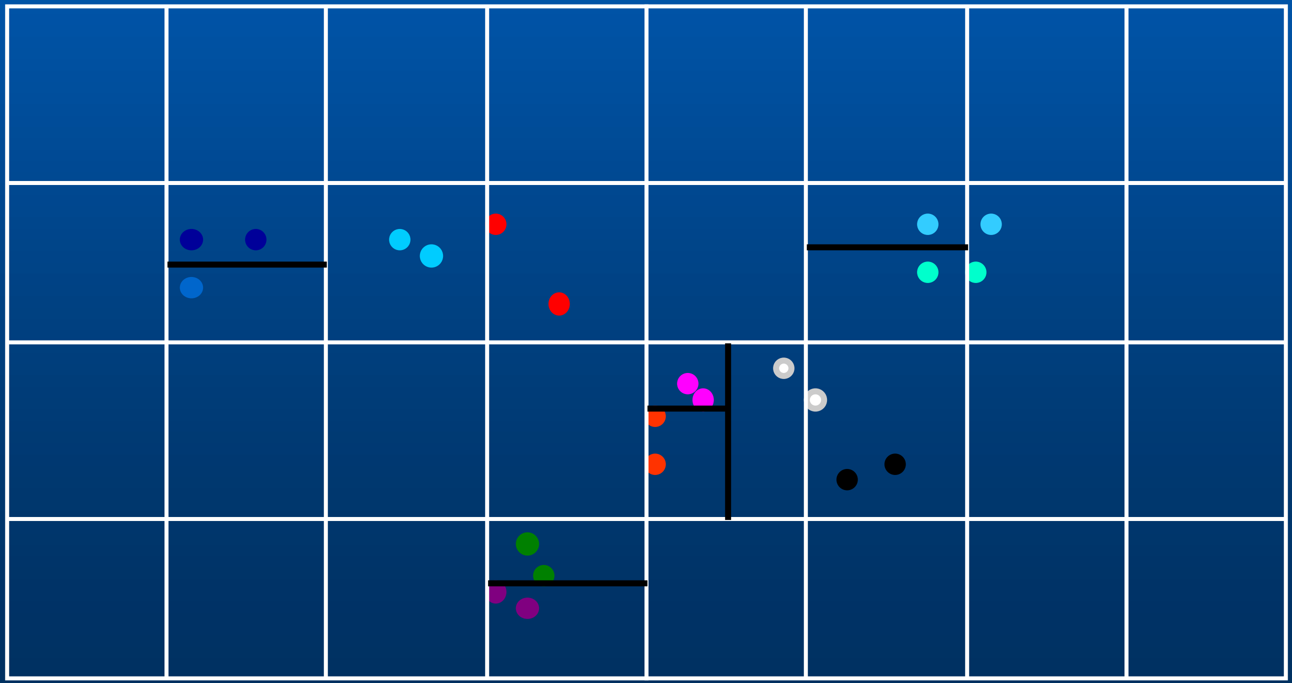
Smooth SLOD Switching

- Only update clusters when they move out of their cell.
  - Max Cluster Size switches from 4 to 2



Smooth SLOD Switching

- Only update clusters when they move out of their cell.
  - Only changed the SLOD parameters

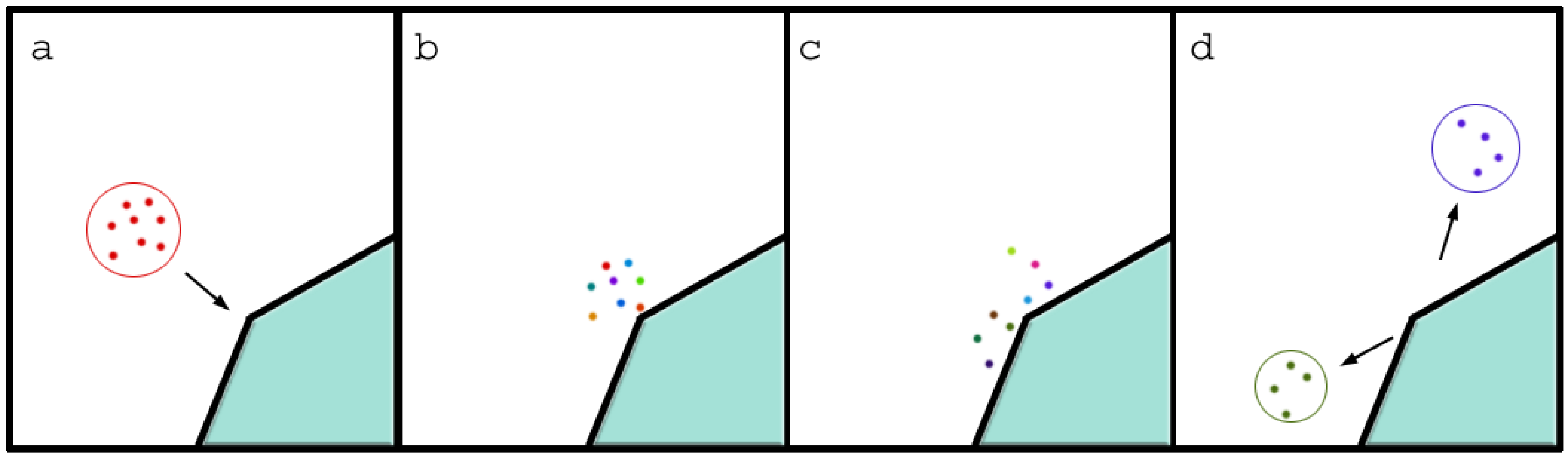


Regions of Interest (ROI)

- **System is still not flexible enough**
  - Real particle systems are dynamic
  - Different Areas need Different SLOD levels
- **Solution:**
  - Divide Simulation Space into many ROIs.
  - ROIs can independently:
    - have separate SLOD settings
    - size and reshape
    - move around the simulation space
    - be added or deleted dynamically

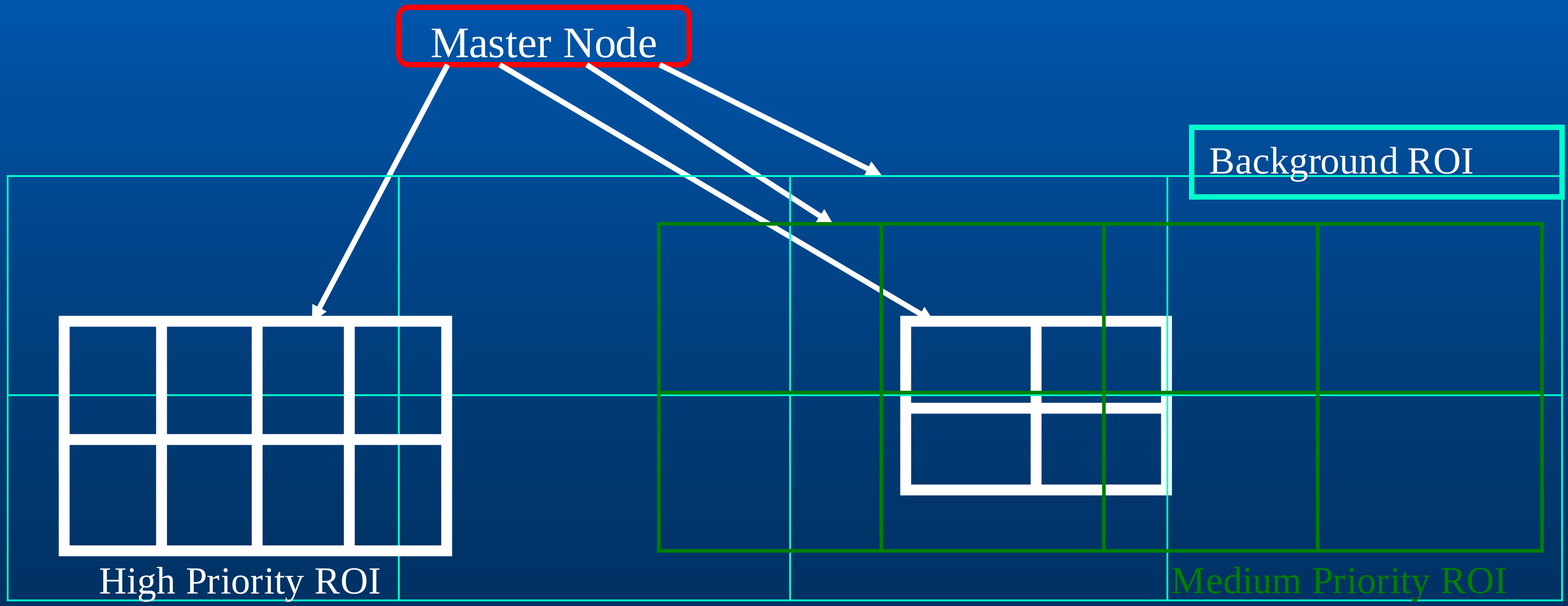
Regions of Interest (ROI)

- We need higher resolution SLOD for important events
  - Collisions on uneven surfaces
  - “Emitter Problem”

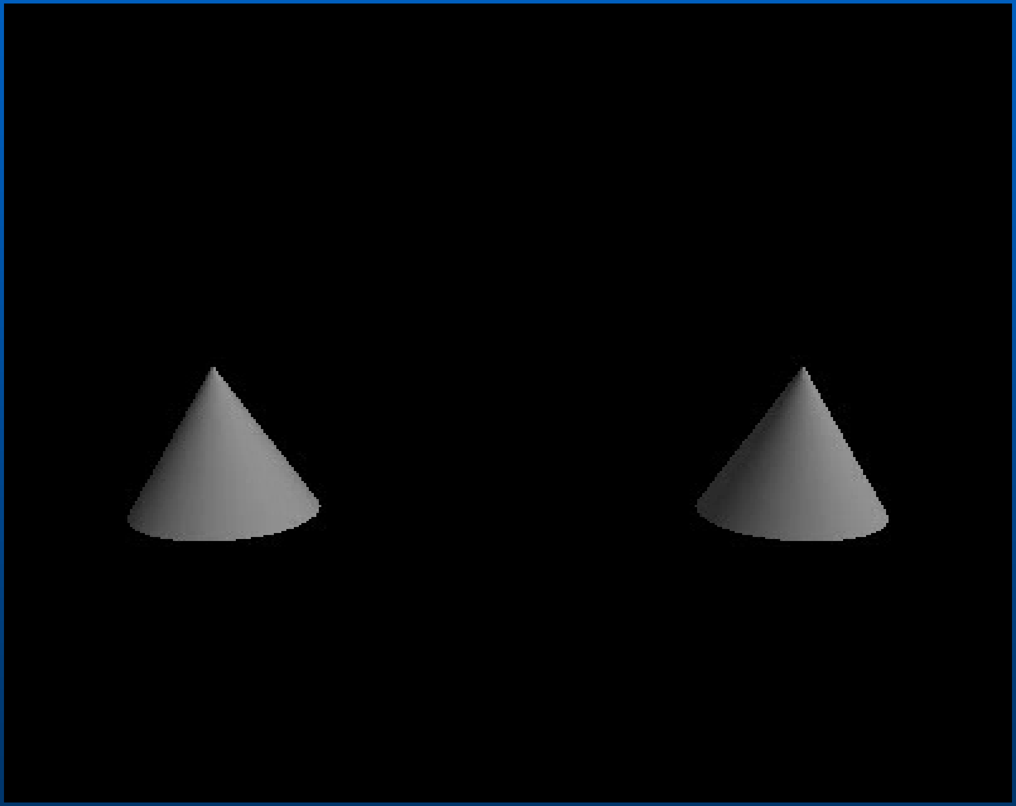


Regions of Interest (ROI)

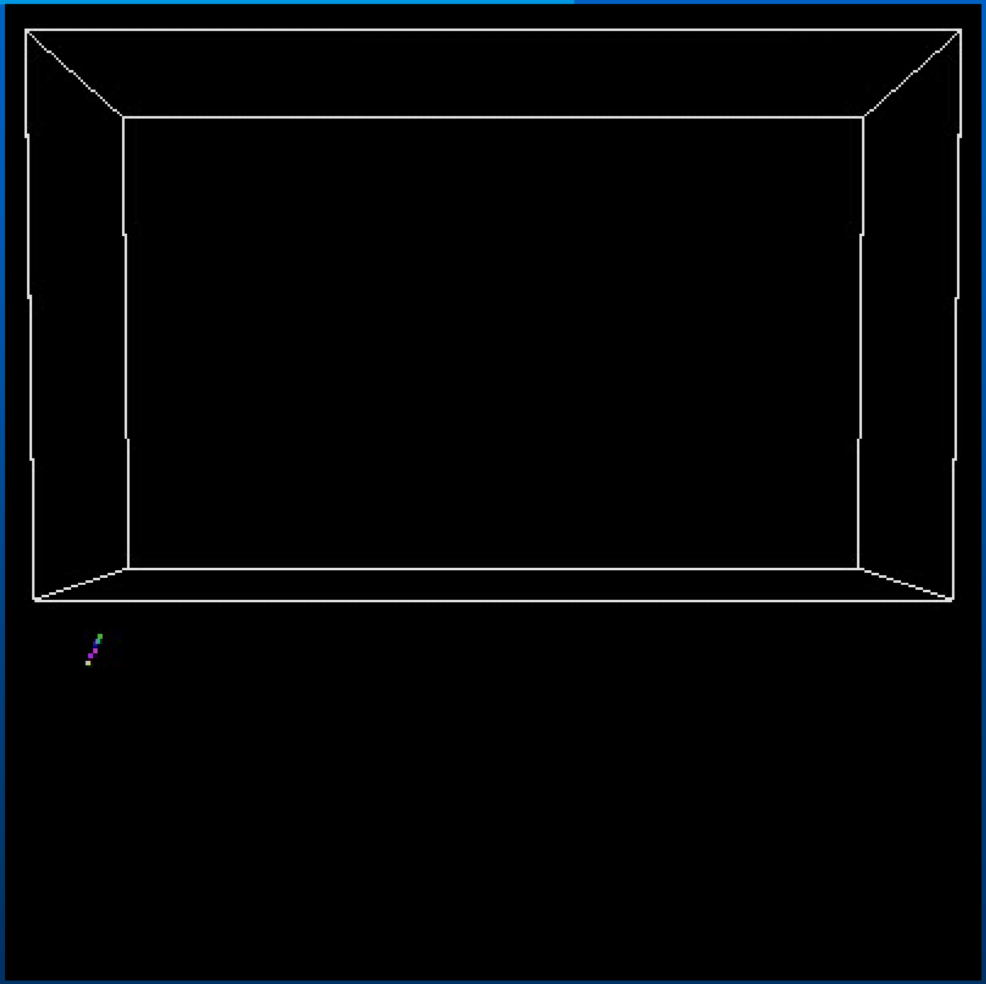
- **Hybrid subdivision tree with ROIs:**
  - Several hybrid SD trees grouped under a Master Node.



Results: Regions of Interest



Results: Changing ROIs



Maintaining Constant Frame Rate

- **After each Step of the Simulation, the frame rate average over the last few frames is checked.**
  - **If too low, adjust SLOD parameters to a coarser level**
  - **Larger clusters begin to form and frame rate increases**

Maintaining Constant Frame Rate

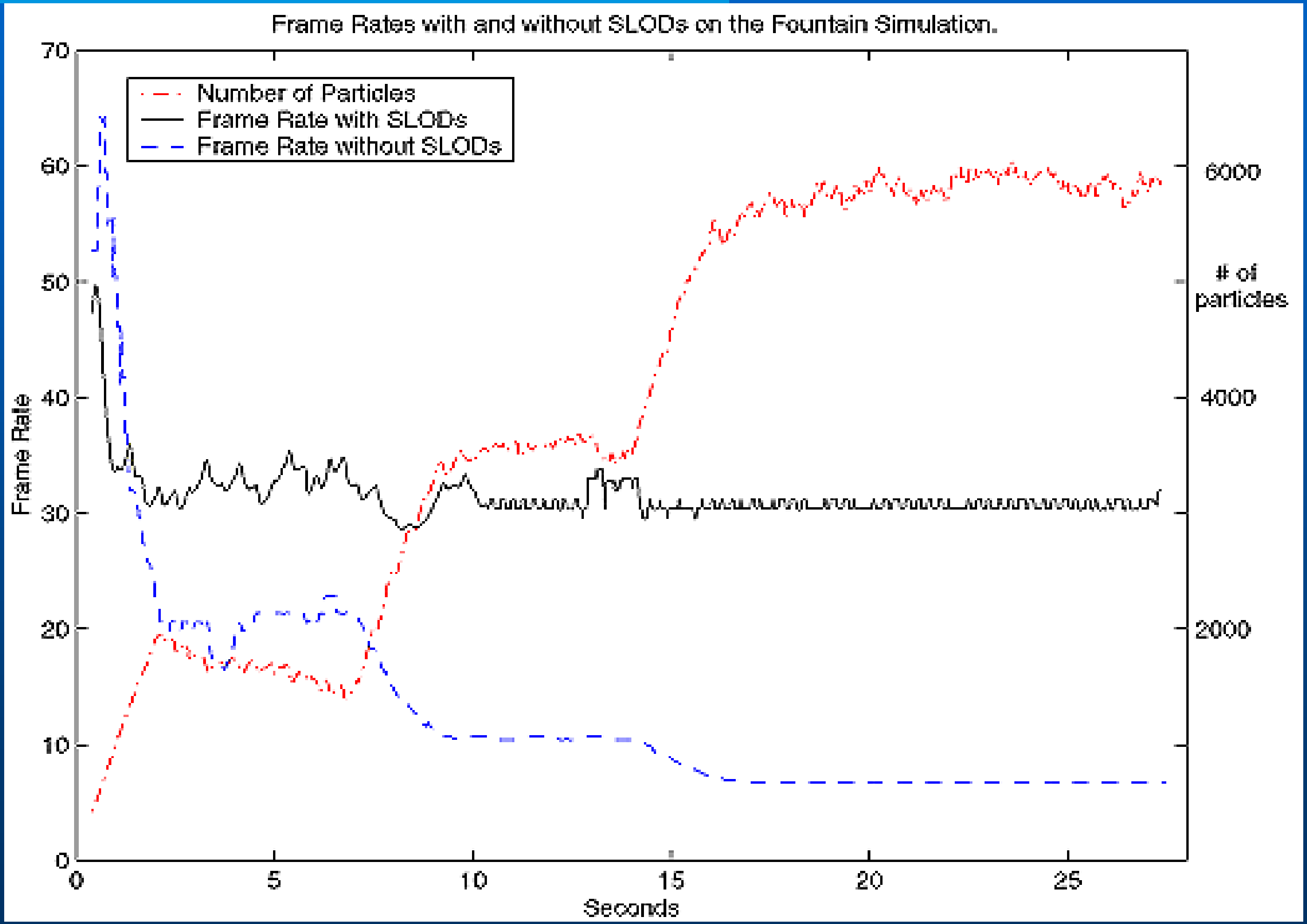
- How to adjust the parameters?
  - First adjust the Max. Cluster Size.
    - Increment/Decrement by 1
    - Increment/Decrement by a percentage relative to how close we are to achieving frame rate goal
    - Both give similar results.
  - Adjust Cluster Breadth only if necessary.

Error Analysis

- **Error in such system not well defined**
- **Concerned with global appearance and *macroscopic* behavior, not local errors on individual particles**
- **Merges:**
  - Weighted averaging in merges affects lighter particles/clusters heavier ones
  - Maximum shift in velocity is reduced when merging clusters have similar mass
  - This provides an argument in favor of subdividing on mass, as well as spatial coordinates.

Results: Frame Rates

Results: Frame Rates



Future Work

- Can this approach or similar ideas be generalized to other dynamical systems?
- Automatic Determination of ROIs
- Particle based Cloth Simulations
- Can we ensure that the Macro Integrity and results of the simulation are correct?

Prairie Grass

- Animate prairie grass in real-time
- 3 LODs: Near, Medium, and Far
- Pre-compute physically-based wind effects, and implement with procedural wind effects

“Animating Prairies in Real-Time”
By F. Perbert and M.-P. Cani,
Proc. of I3D 2001.

LODs

- Near
  - Geometric 3D model
- Medium
  - Volumetric texture mapped onto vertical polygon strips (“2.5D”)
- Far
  - Static 2D Texture

Transition: Near to Medium

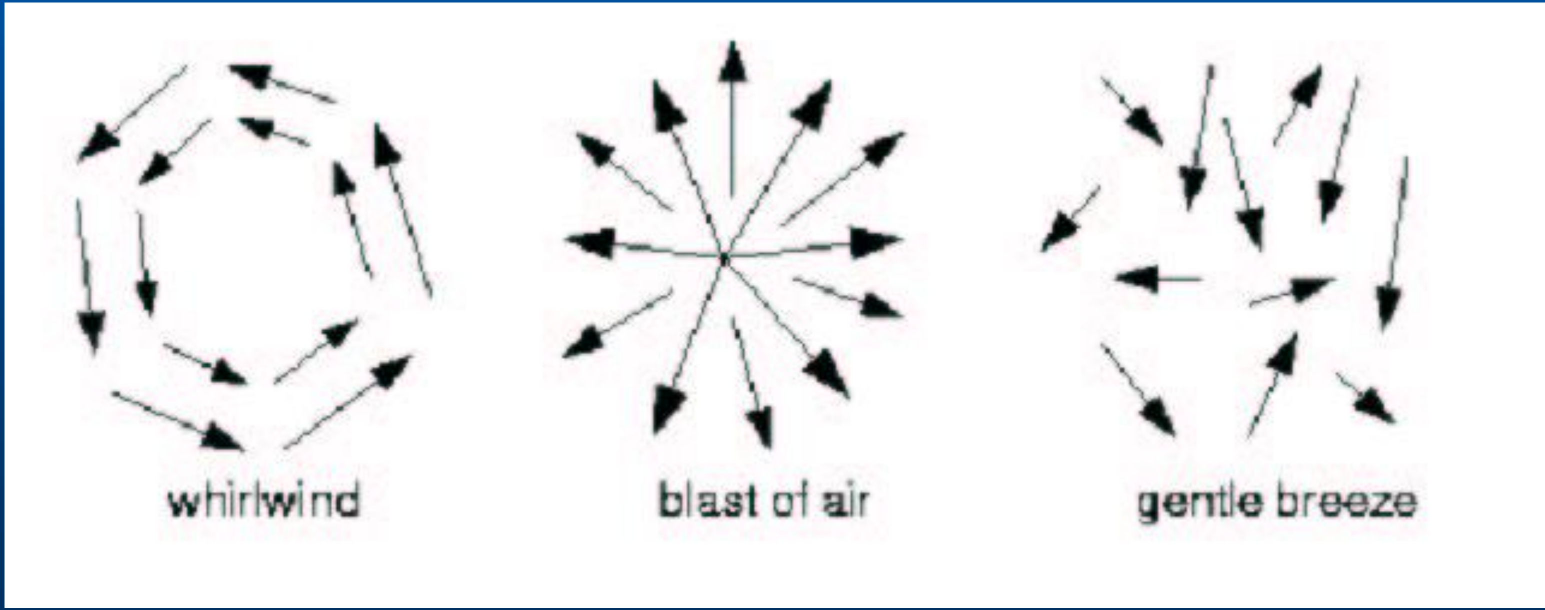
- Volumetric texture for a patch of grass generated from the 3D model
- Linearly interpolate each blade of grass to its corresponding position on the texture map

Transition: Medium to Far

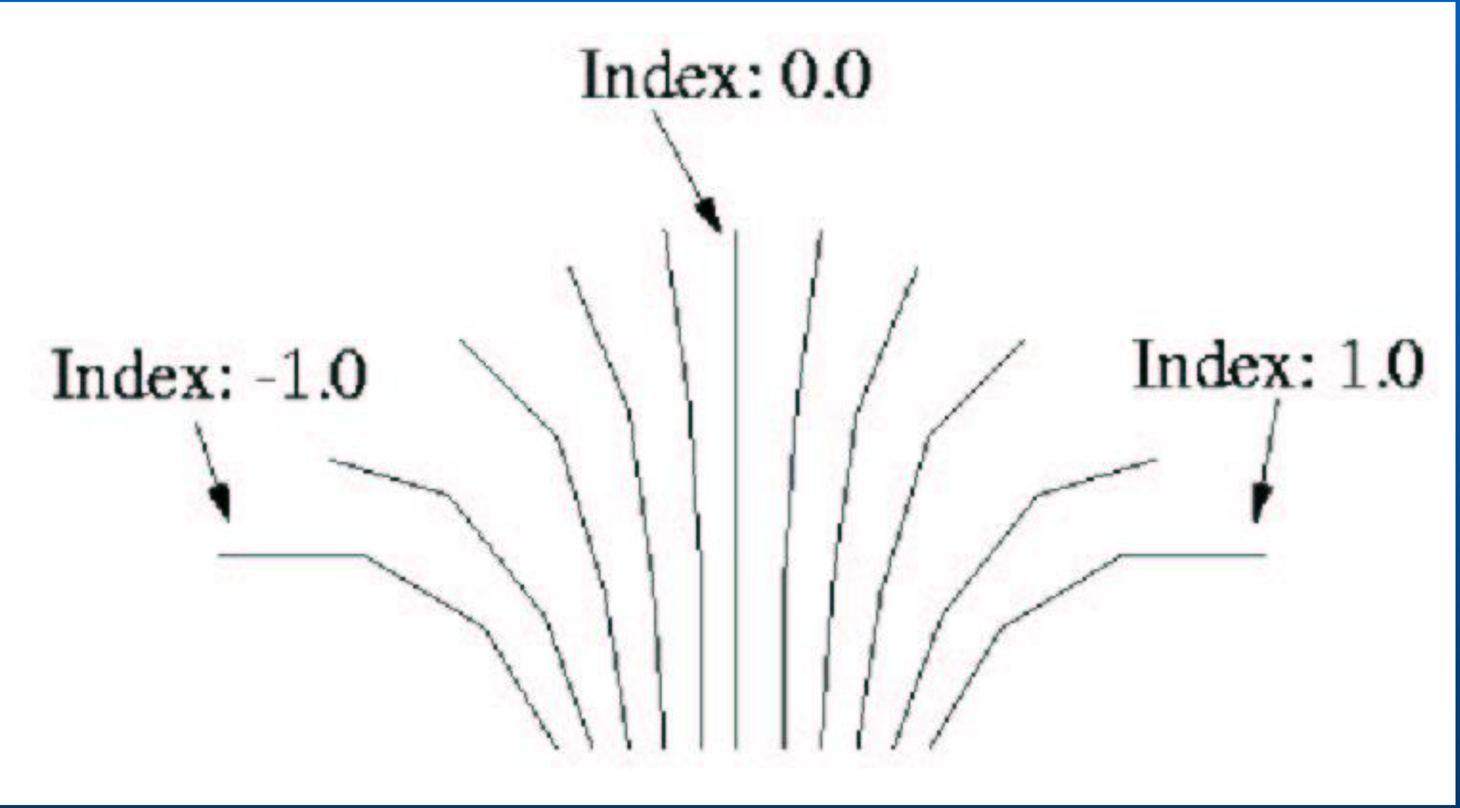
- Without hills, simple cross dissolve is sufficient, since the 2D texture is far away
- To improve appearance with hills, make 2.5 D texture polygons grow (vanish) from (into) the ground, while making the 2D texture vanish (appear)

Wind Primitives

- Types
  - Gentle Breeze
  - Gust
  - Whirlwind
  - Blast

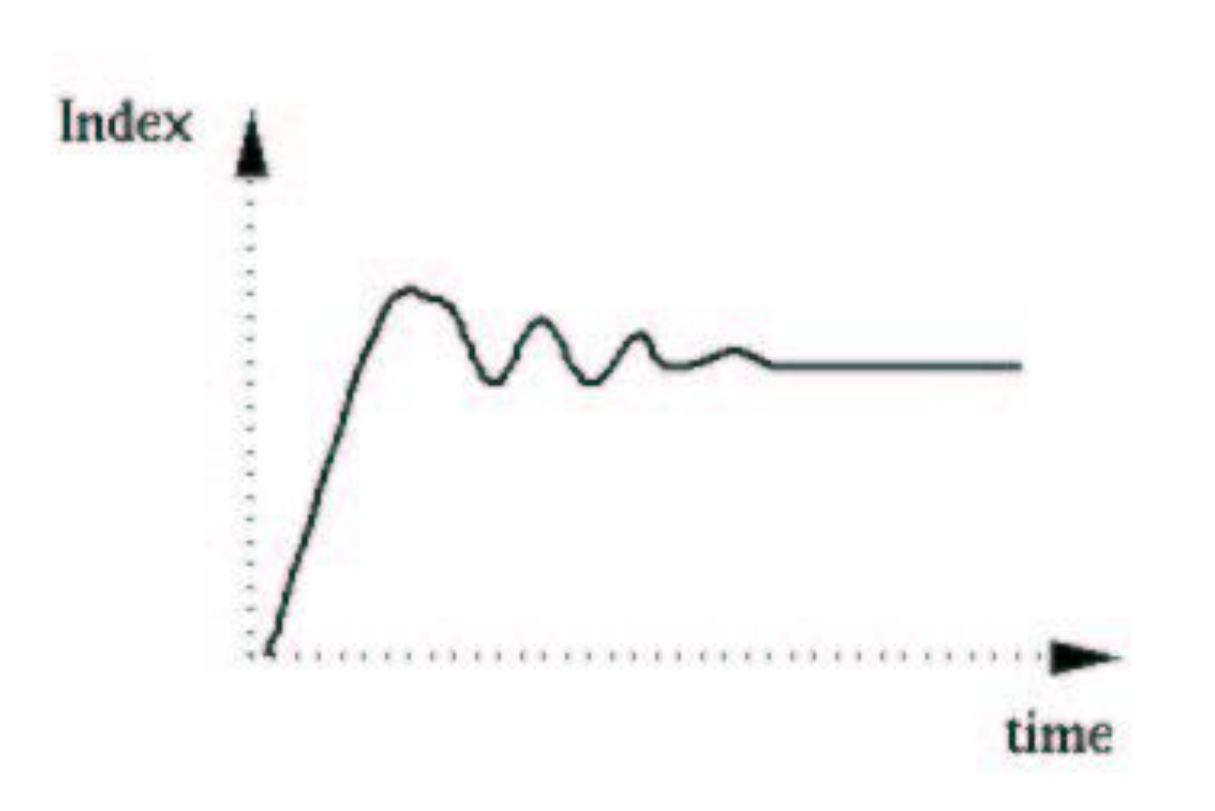


Posture



Range of motion for a blade of grass is computed using a physically-based model

Variation in Posture Index



A constant wind starts blowing

Samples

- http://www-imagis.imag.fr/Membres/Frank.Perbet/prairie_dea/

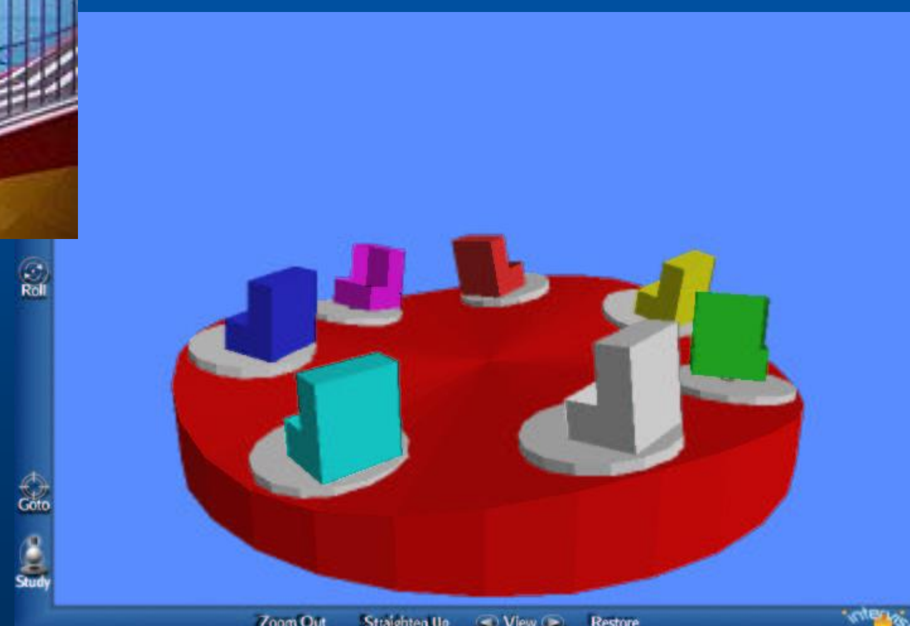
Speed

Quality	Low	Medium	High
No. of blades per patch	160	320	500
2.5D distance range	3m - 8m	2m –12m	3m – 20m
No. Seg. Per grass blade	3	4	8
Approx. nb. Blades per image	100,000	500,000	1,000,000
Frame rate on SGI O2	5 Hz	4 Hz	2 Hz
Frame rate on ONYX	25 Hz	12.5 Hz	8 Hz

View-dependent Culling of Dynamic Systems in VEs

By S. Chenney and D. Forsyth

Proc. of I3D 1997



Influence of Initial Conditions

- Strong – Viewer can predict state based on initial conditions accurately, so must simulate
- Medium – Viewer can make some qualitative predictions
- Weak – Viewer can make no predictions, but can have expectations of state, based on physical principles

Parameters

- θ - Angular position of platform on the track
- ϕ - Angular position of the car on the platform

Start State

Platforms accelerate

$$\theta_0 = \text{constant}$$

$$\dot{\theta}_0 = 0$$

$$\ddot{\theta} = \frac{\tau(t)}{I_{tilt-a-whirl}} \approx \alpha_{start}(t)$$

$$\phi_0 = \text{constant}$$

$$\dot{\phi}_0 = 0$$

$$\ddot{\phi}_0 = f(\theta, \dot{\theta}, \phi, \dot{\phi}) + \text{user impact}$$

Run State

Motion in a steady state

$$\theta = \dot{\theta} t + \theta_0$$

$$\dot{\theta} = 6.5\text{rpm}$$

$$\ddot{\theta} = 0$$

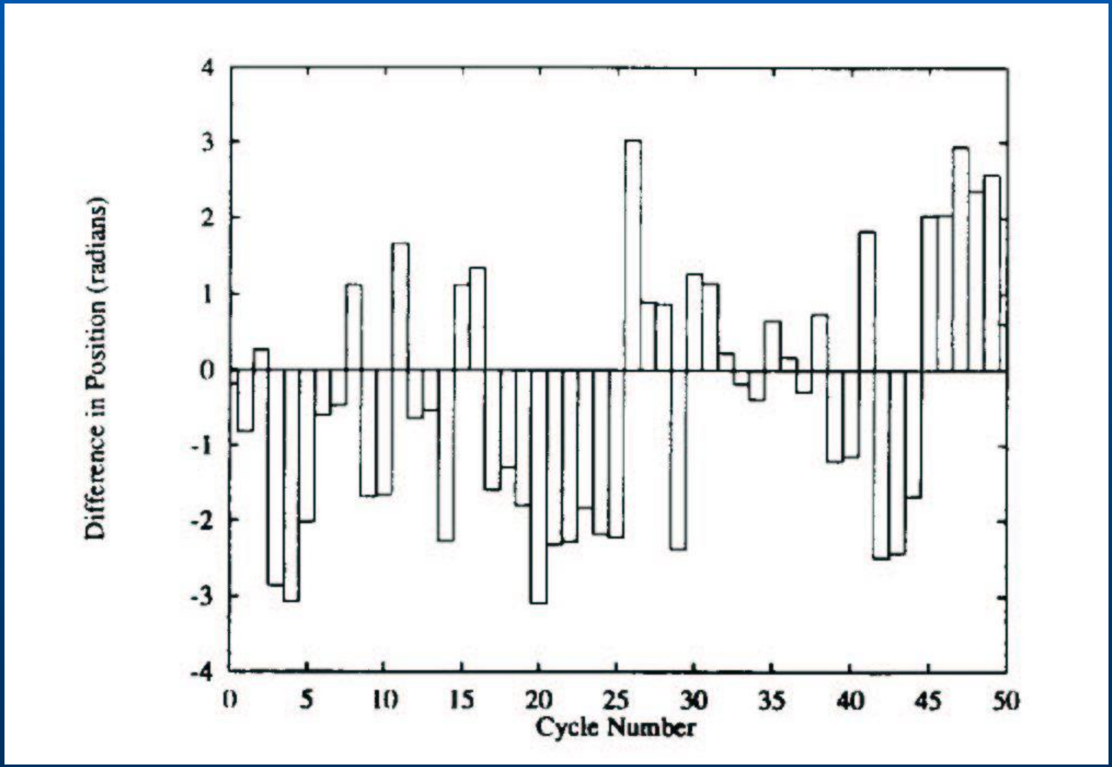
$$\phi = \int_0^t \dot{\phi}\, dt$$

$$\dot{\phi} = \int_0^t \ddot{\phi}\, dt$$

$$\ddot{\phi} = f(\theta, \dot{\theta}, \phi, \dot{\phi}) + \text{user impact}$$

Chaotic Behavior in Run State

- ϕ For 2 cars whose initial conditions vary by 10° and 10°/s
- Very hard to predict state and, after 9 seconds of virtual time, state can be sampled from a probability density



Stop State

Platforms slow down and stop

$$\theta = \int_0^t \dot{\theta}\, dt$$

$$\dot{\theta} = \int_0^t \ddot{\theta}\, dt$$

$$\dot{\theta}_f = 0$$

$$\ddot{\theta} \approx \alpha_{stop}(t)$$

$$\phi = \int_0^t \dot{\phi}\, dt$$

$$\dot{\phi} = \int_0^t \ddot{\phi}\, dt$$

$$\ddot{\phi} = f(\theta, \dot{\theta}, \phi, \dot{\phi}) + \text{user impact}$$

Decay State

- **Cars are still in motion, and energy is decaying as a damped harmonic oscillator**

$$\theta = \text{constant}$$

$$\dot{\theta} = 0$$

$$\ddot{\theta} = 0$$

$$\phi = \int_0^t \dot{\phi}\,dt$$

$$\dot{\phi} = \int_0^t \ddot{\phi}\,dt$$

$$\ddot{\phi} \approx -k_1\dot{\phi}$$

Once the car's angular velocity has dropped far enough, we can use a linear model

$$\ddot{\phi} \approx -k_2$$

Stationary State

Everything is stationary

$$\theta = \text{constant}$$

$$\dot{\theta} = 0$$

$$\ddot{\theta} = 0$$

$$\phi = \text{constant}$$

$$\dot{\phi} = 0$$

$$\ddot{\phi} = 0$$

Re-entering View

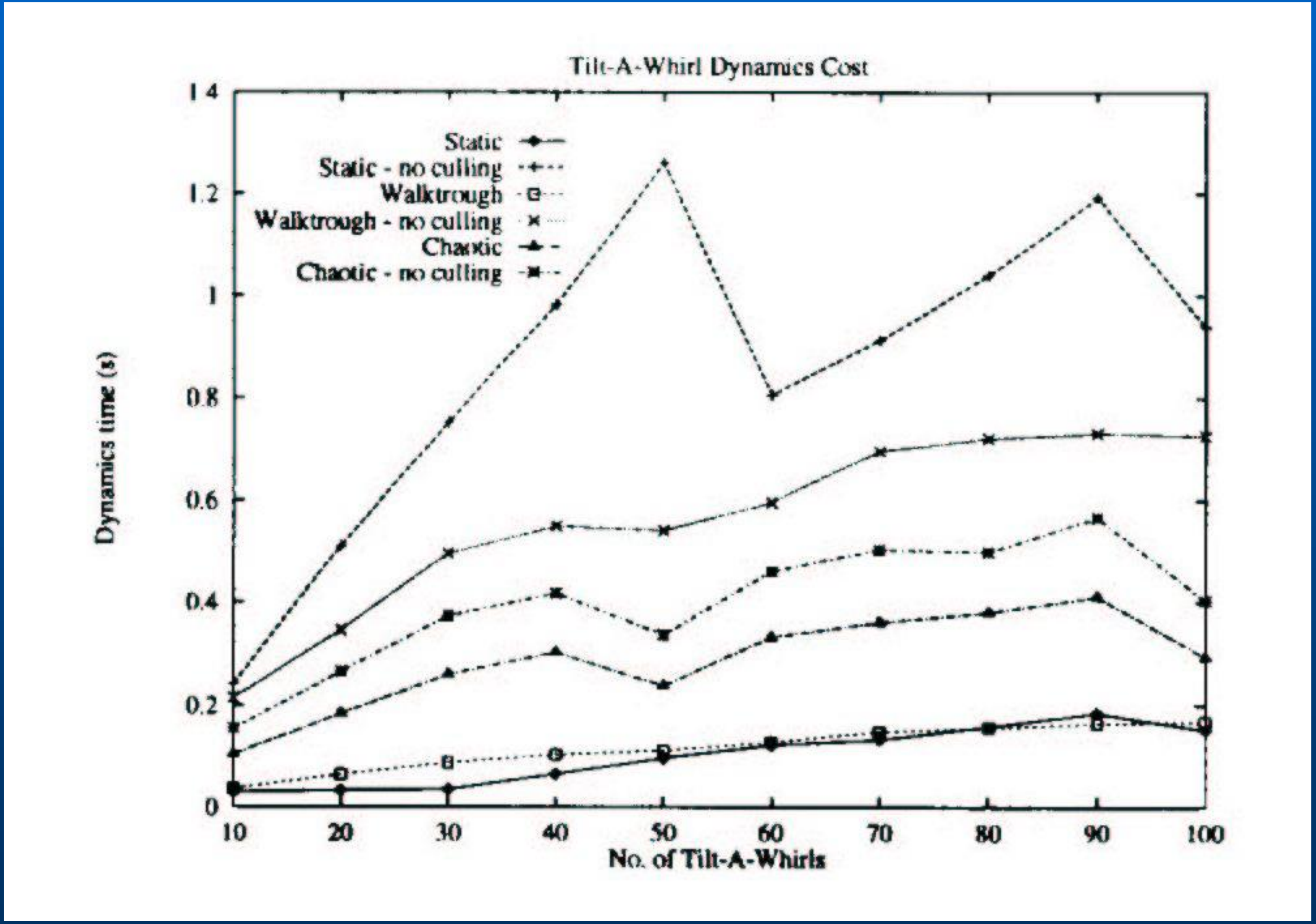
- Determine which phase the tilt-a-whirl is in and find a state which matches the last observation
- In general, can integrate forward to get state
- For run state, can get state from probability distribution
- For decay state, can determine energy remaining in system, and choose state accordingly
- For stationary state, only one option

Building the Distribution

- Physically simulate the run state over a long time
- Create discrete cells corresponding to ranges of states
- At each step, increment a counter corresponding to the state the system is in
- The probability of being in a state *i* is

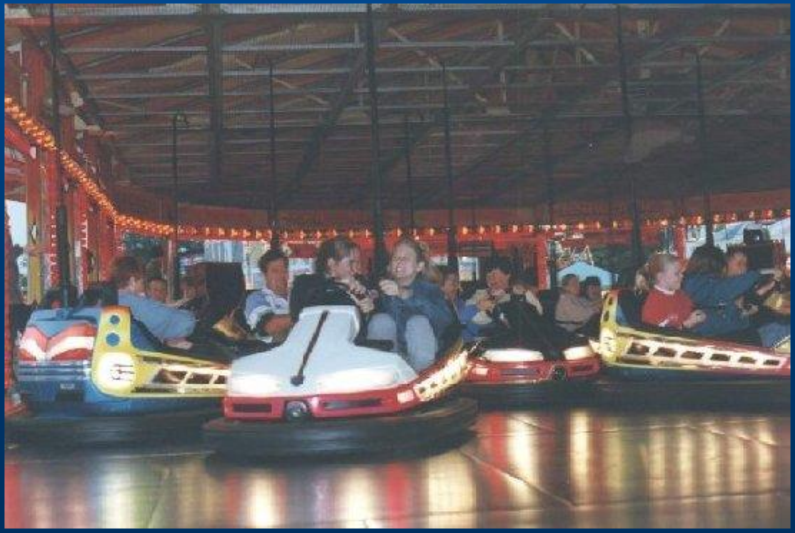
$$P_i = \frac{count_i}{\sum_j count_j}$$

Simulation Cost



Bumper Car Parameters

- Position of each car on an elliptical track
- Orientation+angular velocity of each car
- Velocity of each car
- State for each car is given by $\left(r, \phi, \rho, \dot{u}, \dot{v}, \dot{\rho}\right)$



Simulation

- Use perturbed motion of 12 cars following an elliptical path
- Sample the position of each car independently
- If a collision occurs, move the cars so that they are farther apart, but their mutual center is maintained

Influence of Initial State

- The uncertainty of the state of each car grows with time, making it harder for a viewer to predict where the cars ought to be
- After sufficient time, the states of the car may be chosen from a distribution