

Simulation

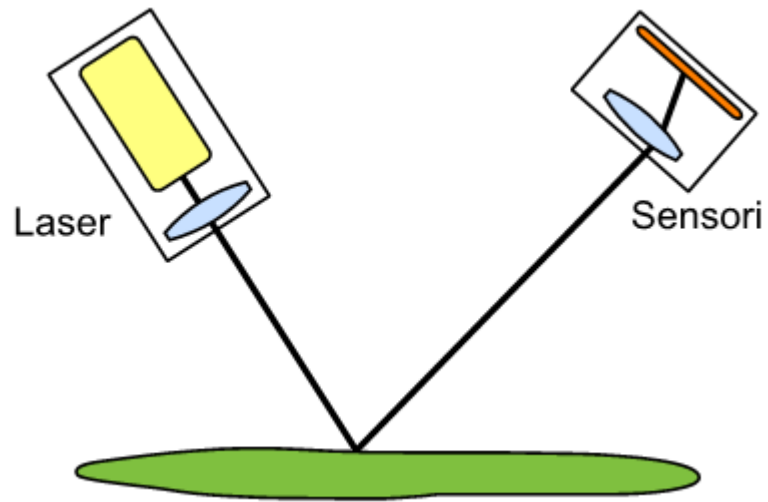
Variance reduction

Example

M C Example

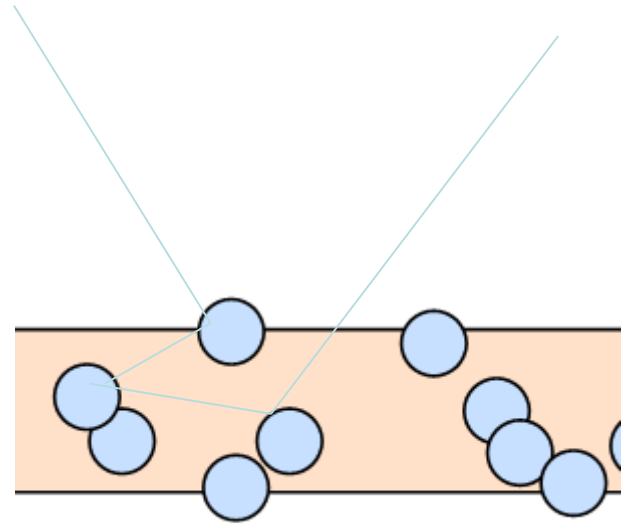
- Consider scattering of laser beam from a material layer
- MSc thesis of Jukka Räbinä 2005
- Goal is to simulate different statistics of the scattered image using Monte-Carlo

Experimental set up



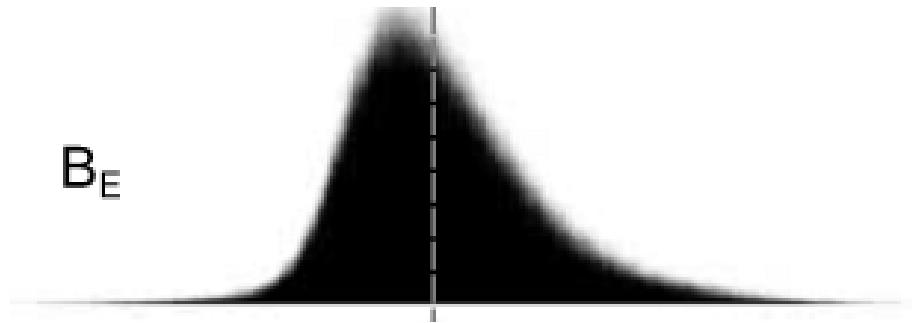
Scattering

- Simulate propagation of ray in cloud of particles
- Basically ray tracing
- Positions and scattering directions of particles are random



Goal of simulation

- Compute the intensity, center of mass etc of the scatter image captured by camera
- I.e. an integral of a function involving the intensity distribution.



Performing the simulation

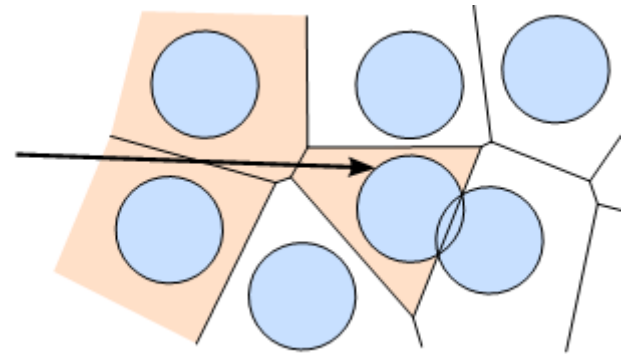
- We have to simulate several "images" and compute desired quantities with confidence intervals
- How to implement a single "image" and how does this reflect in confidence intervals
 - Point of view from variance reduction

Straightforward approach

- Create a random particle cloud
 - Sizes prescribed, centers randomly distributed in given layer
- Simulate a fixed number of rays
 - Each ray scatters from particle surface to a random direction (case dependent distributions)
 - Count rays that reach the camera

Straightforward approach

- Hard to find the hits to the particle cloud (additional data structures needed)
- Only few particles hit the camera



Straightforward approach

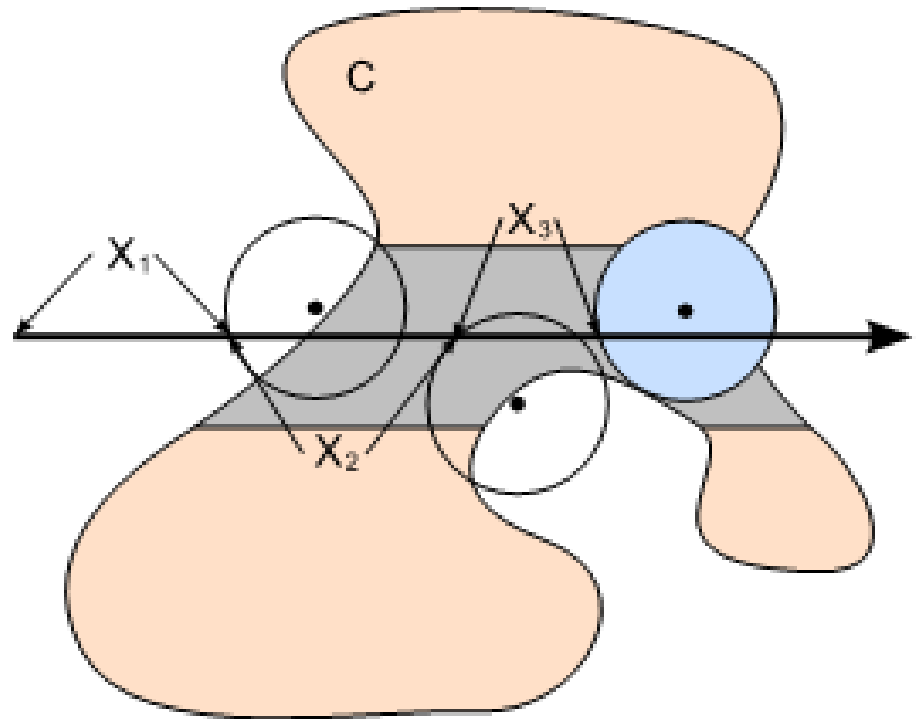
- Limited possibilities for variance reduction
 - In practice only antithetic variables when generating the laser beam
 - Due to many collisions only small correlation between antithetic rays
- Smallish number of images (distinct particle clouds) shows up as variance in the results

Dynamic particles

- Can the a priori fixed random particle cloud be replaced somehow
 - We can derive the expected mean free path of rays in the cloud
 - Requires statistical analysis/understanding of the situation
 - Particles/collisions can be generated dynamically
 - Draw the free path and angle of attack to the next particle -> next collision can be modelled

Dynamic particles

- Draw next free path (Exp-distributed) and distance of center point from ray-line (Unif-distrib.) -> we can define the centerpoint for new particle
- If center is in the area of interest, generate a particle and compute collision, otherwise draw a new free path to the same direction
- The collision is modelled as for a fixed particle

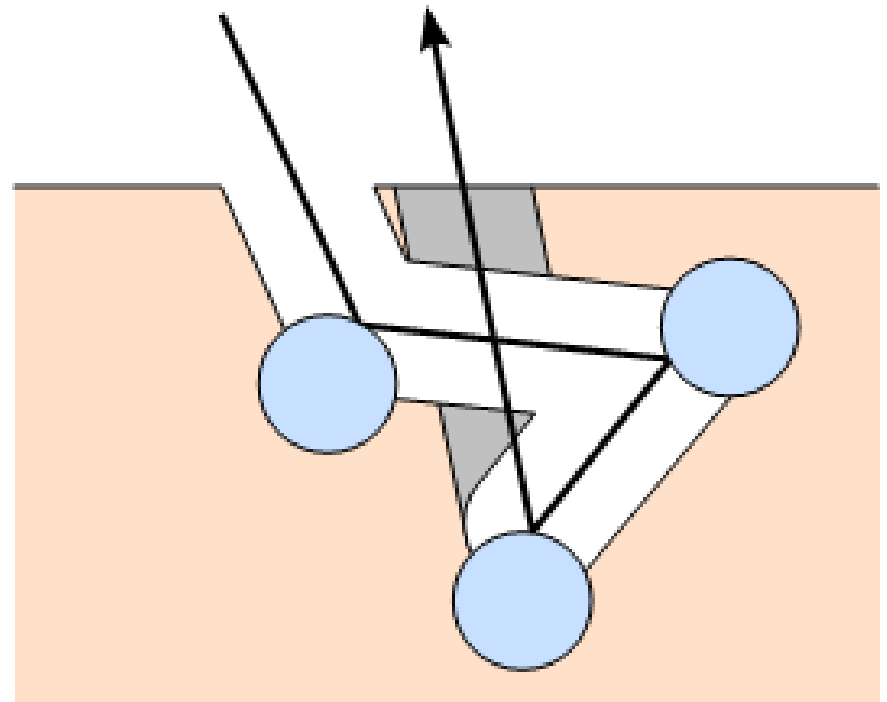


Dynamic particles

- Computation is lighter (15-40% effort)
 - Only needed particles are generated
 - No search is needed to find collisions
- For each ray a new collection of particles
 - Smaller variance (by 50%) as images are based on average particle cloud instead of a fixed cloud

Dynamic particles

- Results are biased if ray history is not accounted in generation of particles
- Direct backscattering should not be blocked on the way back
- The previous path (and observed particle free zone) has to be remembered
- Computing time grows (max 50%)
- Still twice faster and more accurate than basic approach



Method of expectations

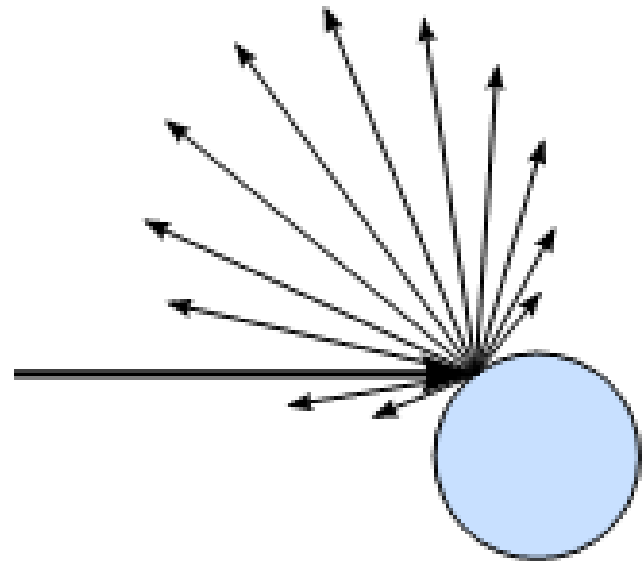
- Use of dynamic particles does not change the frequency of rays hitting the camera
 - Ray propagates unattenuated through the simulation
 - Only a small fraction of rays reach the camera
 - Can we increase the number of rays contributing to the image

Method of expectations

- On each collision, divide the intensity to two parts
 - Compute the expectation of the intensity scattering to the direction of the camera
 - Ray with this intensity is sent towards the camera (given direction, random mean path and accounting for known particles)
 - Rest of the energy is scattered as one ray to a random direction

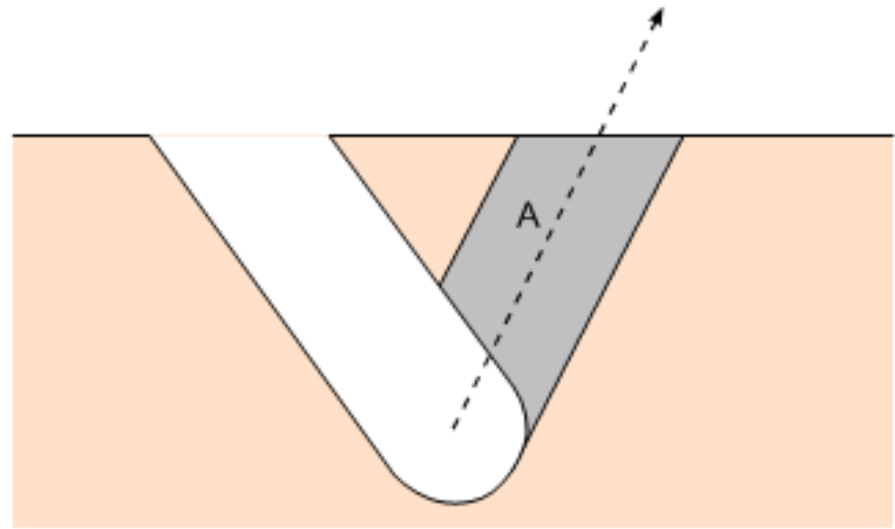
Method of expectations

- Requires that particles scatter the rays (no specular reflections)



Method of expectations

- Does the ray heading towards camera get through
- If we know of a particle that blocks the route to the camera, the ray is lost for sure
- Otherwise we draw a free path and see if it leads out from the particle layer

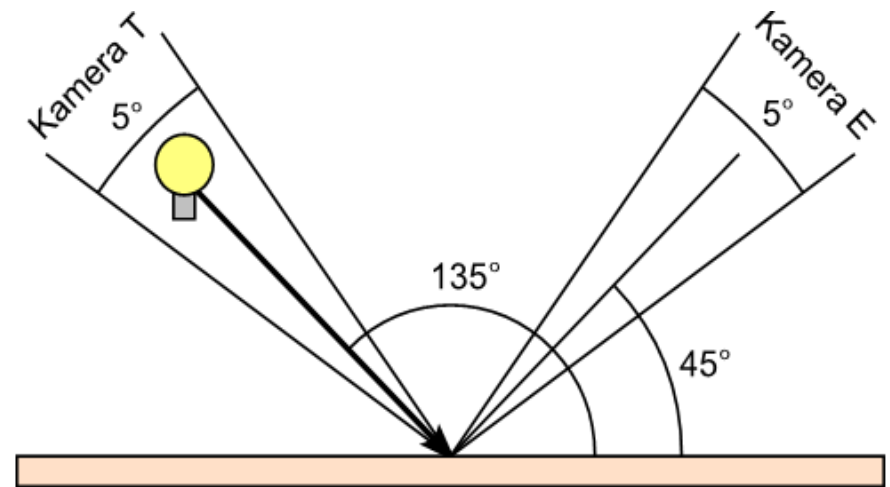


Method of expectations

- Big fraction of collisions can send some energy towards the camera
 - About twice more computation
- More hits to a single pixel
- Hits have smaller intensity (so each hit has smaller effect to the image)
- -> Smaller variance (less than $1/5$ compared to dynamic particles)

Simulation experiment

- Send parallel rays with normally distributed intensity
- Collect the (few) rays scattered to the camera



Simulated results

- Same amount of rays and images using three methods (S static, D dynamic particles, E expectations.)
- E method about 250 times more efficient than the static approach
- Most of efficiency comes from reduced variance

A_E	Intensiteetti
S	1.81881 ± 0.00547
D_∞	1.81603 ± 0.00274
E_∞	1.81688 ± 0.00044

	Aika T_A
S	1505s
D_∞	391s
E_∞	840s