Структуры данных

- Ограничивающие объемы
- Сетки
- Октодеревья
- Kd-деревья
- BSP-деревья (двоичное деление пространства)

Цель: ускорение трассировки

Ускорение РТ

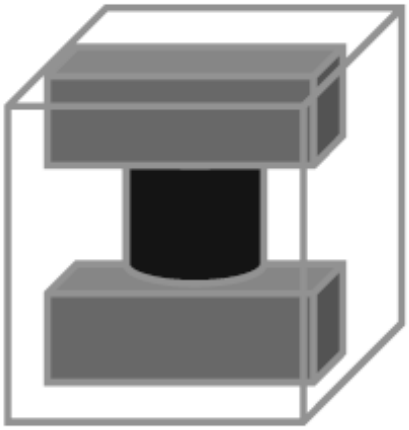
- Пускать по возможности меньше лучей
  - наиболее важно в рекурсивном РТ
- Ускорять каждый тест пересечения луча с поверхностью
  - оптимизировать пересечение луча с треугольником, со сферой, с боксом
  - компилировать с ключами оптимизации
- Делать меньше тестов пересечения луча с поверхностью
  - последующие попадания на тот же объект часто происходят на том же самом полигоне
- Кэширование теневых зондов (лучей на источники). Когда теневой зонд попадает на объект, запоминаем этот объект и проверяем его первым при следующих проверках теневых зондов направленных на тот же источник. Если луч попадает на тот же объект, значит это уже зона тени данного источника. Неважно, что теневой зонд может иметь на своем пути к источнику другой более близкий объект.

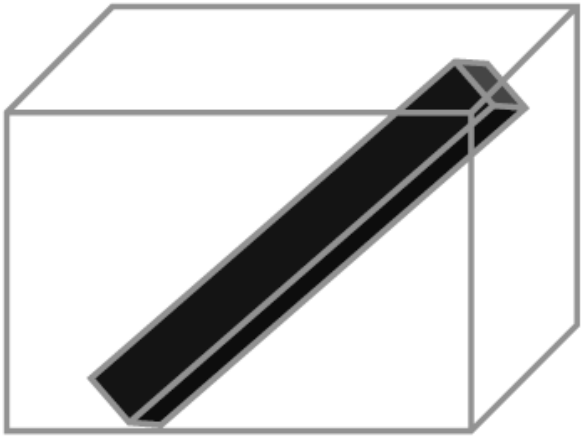
Пространственные структуры данных

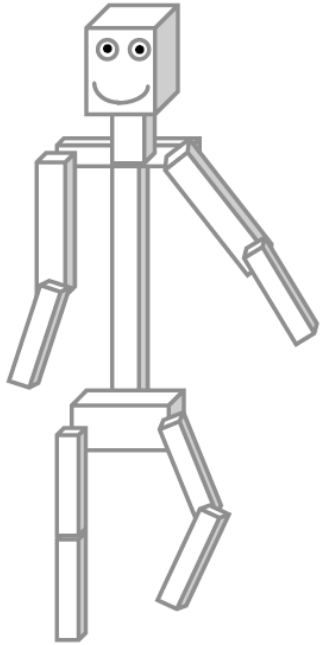
- ПСД применяются для эффективного хранения геометрической информации
- ПСД полезны для
  - collision detection (взаимопроникновение 2-х домов)
  - запросов по местоположению (где ближайший киоск)
  - биохимические имитации (какой протеин взаимодействует с данной молекулой лекарства)
  - визуализация сцен (попадает ли самолет в камеру) и др.
- Хорошая ПСД может дать ускорение РТ в 10, 100 и более раз
- Мы рассмотрим
  - Иерархические ограничивающие объемы
  - Сетки
  - Октодеревья
  - Двоичное деление пространства

Ограничивающие объемы (ОО)

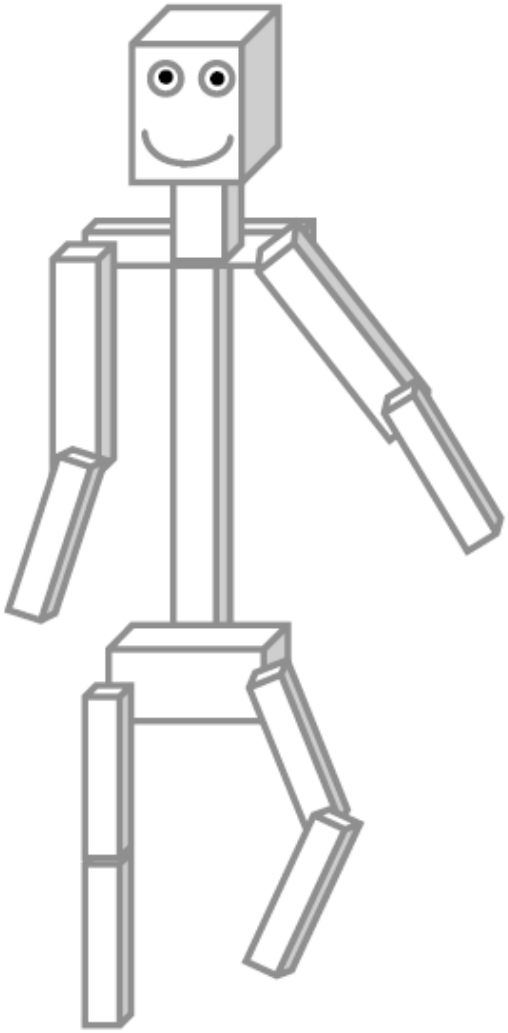
- Простой подход: преврати предметы, которые трудно поддаются тесту на пересечение с лучом, в предметы, которые легко проверять на этот тест
  - пример: вместо сложной сетки взять ее габаритный бокс
  - луч не может попасть на объект, если он не пересек этот бокс
  - это добавляет немного сложности, но вообще-то окупается
- Наиболее применимые типы ОО: сфера и бокс. Бокс может располагаться параллельно осям или нет
- Требуется тщательное описывание габарита







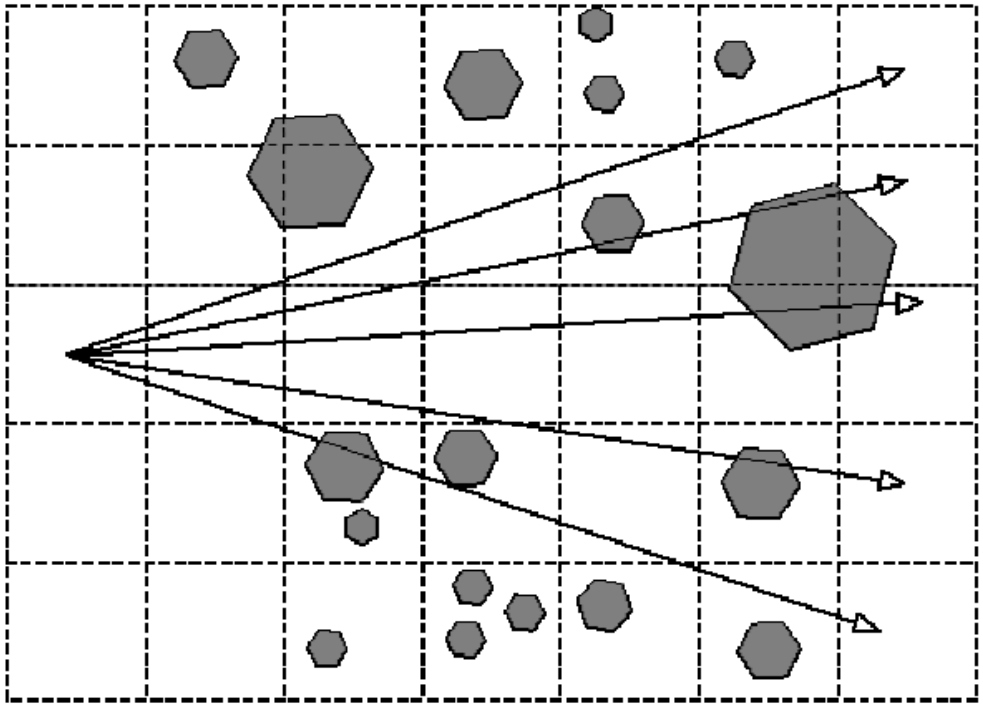
Ограничивающие объемы



Древовидная структура данных:
– Список ОО (сферы, боксы)
– Каждый ОО содержит список под-ОО

```
intersect(BV)
  if (ray misses BV) return MISS
  closest = infinity
  for (each subvolume stored in BV)
    if (ray intersects subvolume,
      and closer than closest)
      closest = subvolume
return closest
```

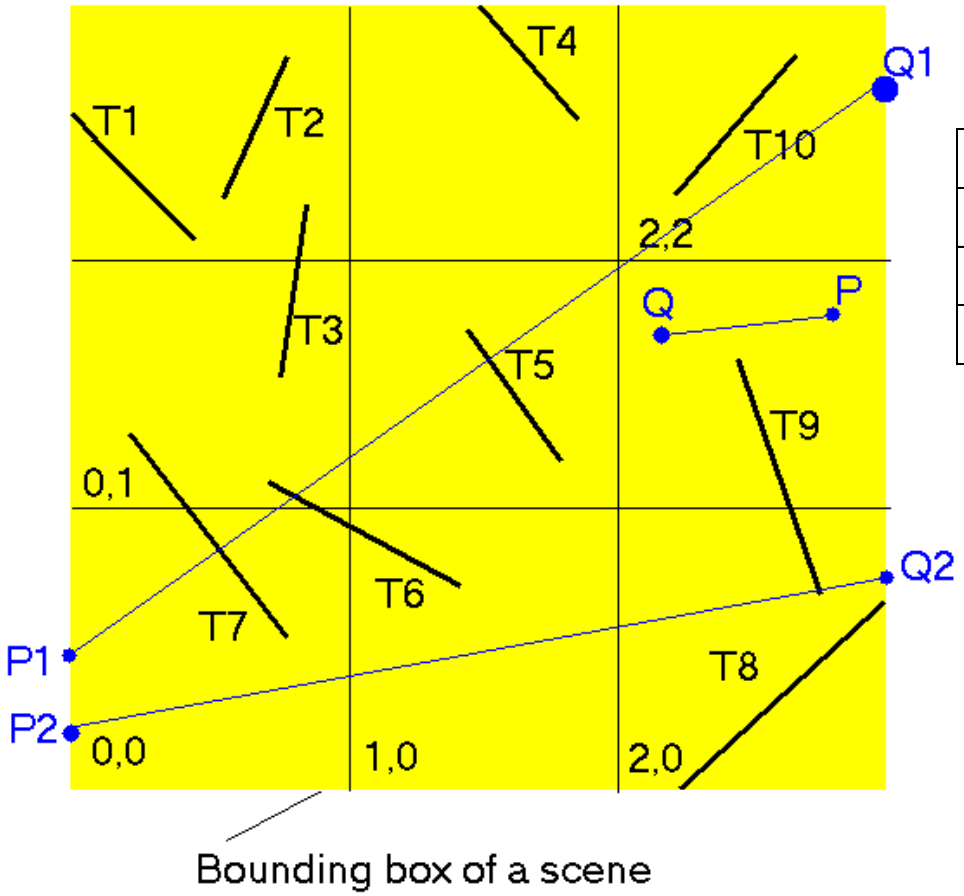
Сетки (grid)



- Пространство разбивается на равные ячейки (квадраты, кубы)
- Ячейка — элемент массива со ссылкой на список попадающих в неё объектов

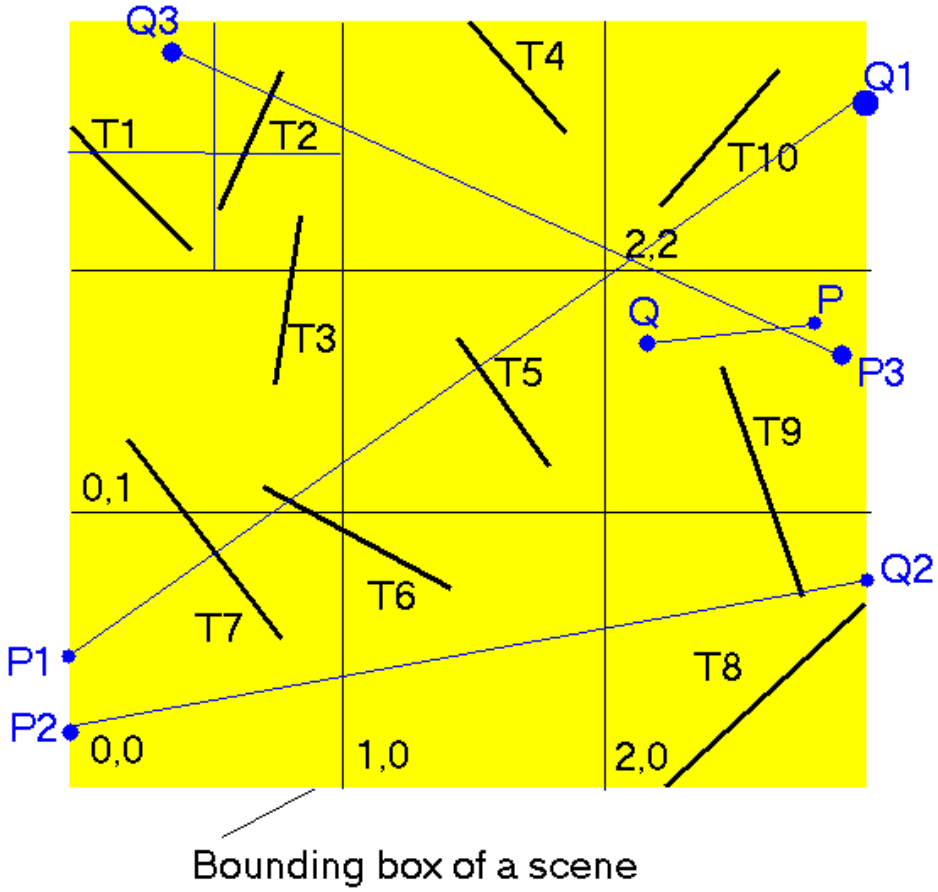
Проблема: чайник на стадионе

Grid



	1	2	3
1	T1,T2,T3	T4	T10
2	T3,T7,T6	T5	T9
3	T7,T6	T6	T8,T9

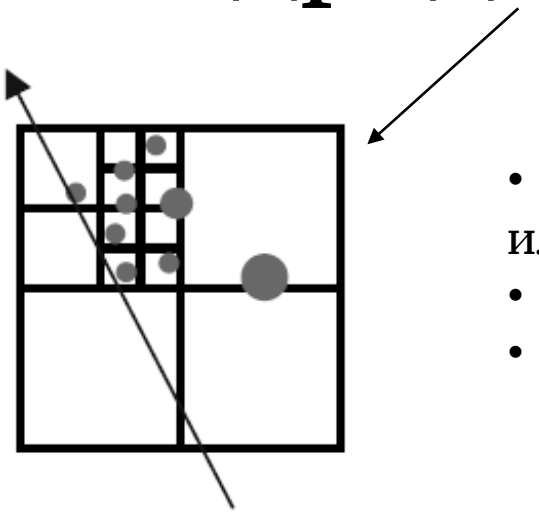
Nested Grid



	1	2	3
1		T4	T10
2	T3,T7,T6	T5	T9
3	T7,T6	T6	T8,T9

	1	2
1	T1	T2
2	T1	T2,T3

Квадродеревья и октодеревья



- Остановить разбиение, когда число достаточно мало или глубина дерева максимальна
- Внутренние узлы хранят указатели на потомков
- Листья дерева – списки поверхностей

```
trace(cell, ray) { // returns object hit or NONE
if (cell is leaf)
  return closest(objects_in_cell(cell))
for (child cells pierced by ray, in order)
  obj = trace(child, ray)
  if obj != NONE return obj
return NONE
```

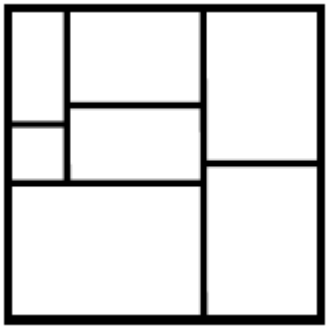
Что лучше для РТ?

- Сетки легче программировать, но они берут памяти очень много (и медленны) для неоднородных сцен (а это все сцены)
- Окто довольно хороши, но не так быстры как сетки для ряда сцен
- Nested grids кажется наискорейшие для статических сцен
- Если сцена динамическая основной становится цена на перегенерацию и модификацию структуры, а это ?!
- Здесь могут оказаться лучшими ИОО
- ИОО легко программировать если ваша модель по природе иерархическая (человек), а иначе – нет

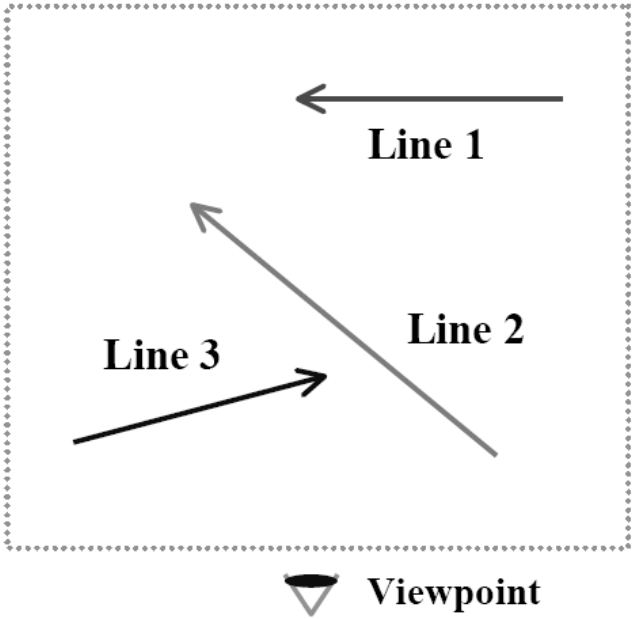
- Какие есть еще?

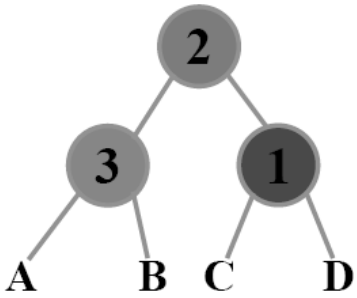
k-d Trees and BSP Trees

k-d (k-dimensional tree)

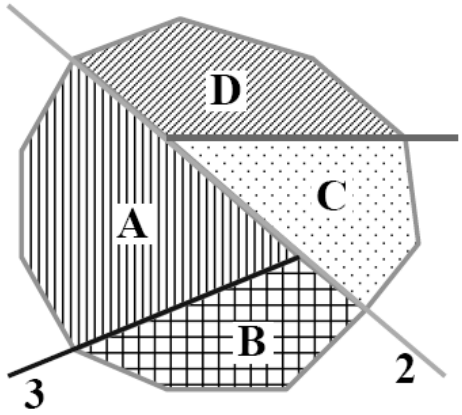


BSP (Binary space partition)





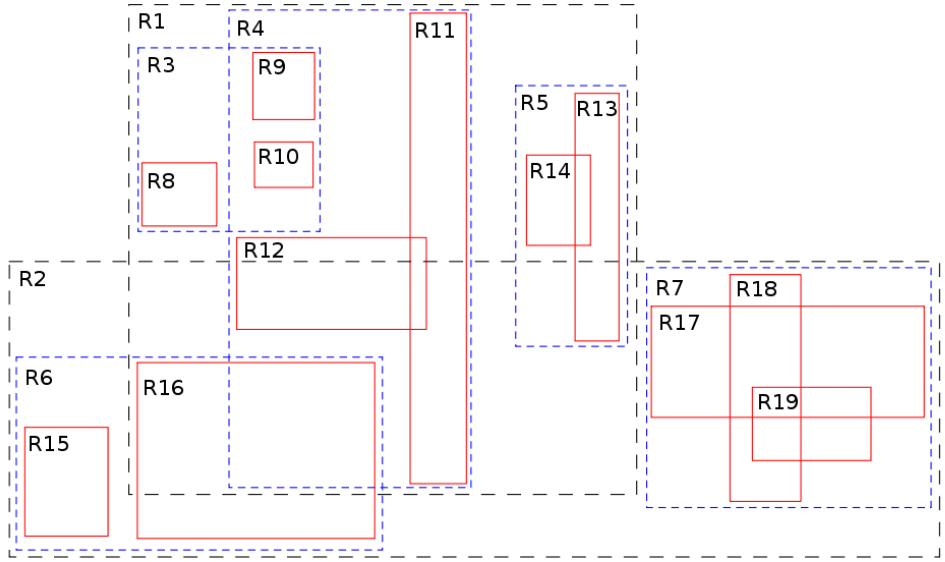
a BSP tree using 2 as root

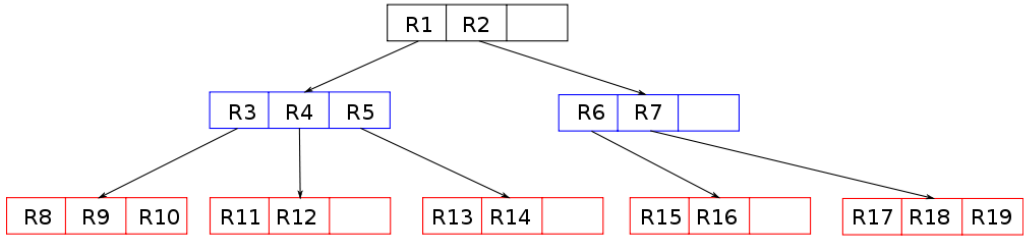


the subdivision of space it implies

MySQL — SPATIAL INDEX

R-trees with quadratic splitting



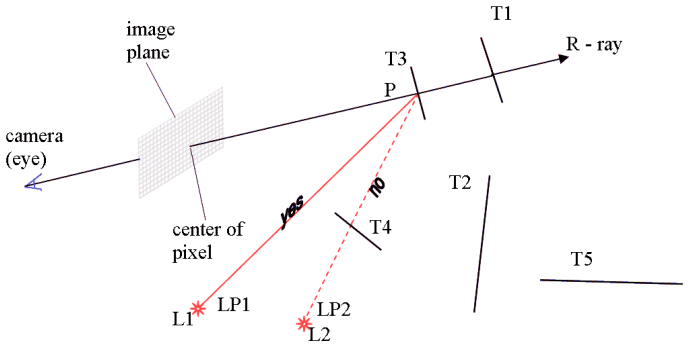



```
mysql> CREATE TABLE geom (g GEOMETRY NOT NULL, SPATIAL INDEX(g)) ENGINE=MyISAM;
mysql> SET @poly = -> 'Polygon((30000 15000,31000 15000,31000 16000,30000 16000,30000 15000))';
mysql> SELECT fid,AsText(g) FROM geom WHERE -> MBRContains(GeomFromText(@poly),g);
```

Разница трудоемкости MCC – ОРЛТ (Grid!)

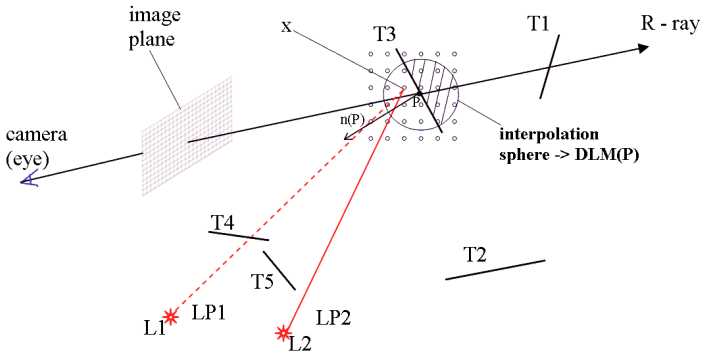
ОРЛТ (Обратная рекурсивная лучевая трассировка, RRTA)

- nPix первичных лучей
- nPix * nL Long теневых лучей



MCC (Метод световых сеток, LMM)

- nPix первичных лучей
- nPix * n(kI*h) Short теневых лучей
- nL * nLM Long теневых лучей на весь расчет



$$\Delta t = \left\{ N_{LM} \cdot nL \cdot LONG + Npix \cdot n(ki \cdot h) \cdot SHORT \right\} - \left\{ Npix \cdot nL \cdot LONG \right\}$$

Naïve vs. Grid (11x11x11)

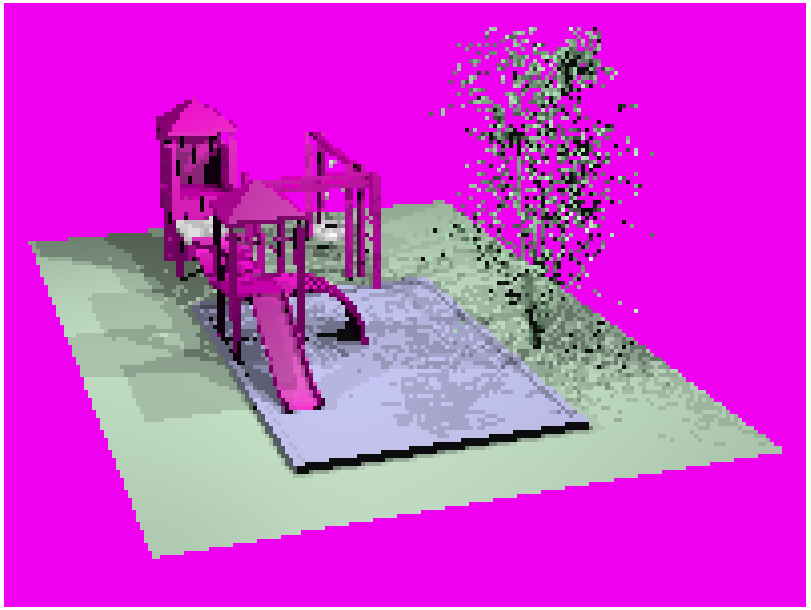
Method	Resolution	Time (sec.)	nPP	nBkg	nLM	n(h*kI)
RRTA	200x100	8	6770	13230		
LMM		100			32861	57.5
RRTA	1000x500	208	166848	333152		
LMM		2087			42388	55.4
RRTA	3000x1500	1873	1501022	2998978		
LMM	Huge, Not computed					

Method	Resolution	Time	nPP	nBkg	nLM	n(h*kI)	kI
RRTA	200x100	0,5	8306	11694			
LMM		2			24694	30.4	2.5
LMM		3			33496	53.1	3
RRTA	1000x500	7	204183	295817			
LMM		28			31370	30.5	2.5
LMM		45			39830	53.4	3
RRTA	3000x1500	60	1837406	2662594			
LMM		150			24876	16.5	2.1
LMM		242			32474	30.6	2.5
LMM		400			40790	53.4	3

Ускорение MCC

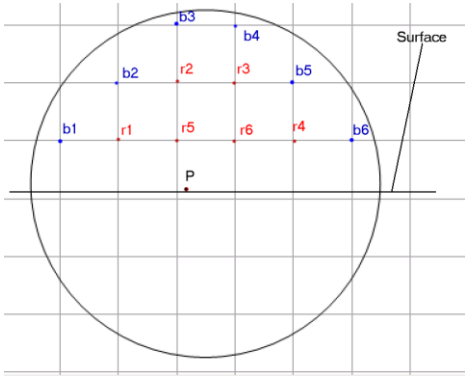
27296 triangles. Nested GRID, 1st level: 81x81x81 cells

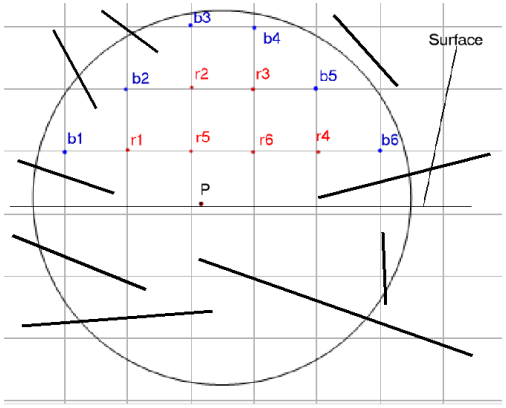
Method	Resolution	Time	nPP	nBkg	nLM	n(h*kI)	kI
RRTA	200x100	0,5	8306	11694			
LMM		2			24694	30.4	2.5
LMM		3			33496	53.1	3
RRTA	1000x500	7	204183	295817			
LMM		28			31370	30.5	2.5
LMM		45			39830	53.4	3
RRTA	3000x1500	60	1837406	2662594			
LMM		150			24876	16.5	2.1
LMM		242			32474	30.6	2.5
LMM		400			40790	53.4	3



2nd level - $\sqrt[3]{N}$

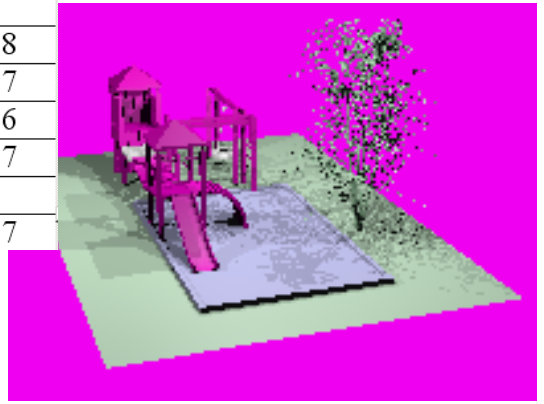
Ускорение MCC
27296 triangles. Nested GRID, 1st level: 81x81x81 cells





Method	Resolution	Time	nPP	nBkg	nLM	n(h*kI)	kI
RRTA	200x100	0,5	8306	11694			
LMM		2			24694	30.4	2.5
LMM		3			33496	53.1	3
RRTA	1000x500	7	204183	295817			
LMM		28			31370	30.5	2.5
LMM		45			39830	53.4	3
RRTA	3000x1500	60	1837406	2662594			
LMM		150			24876	16.5	2.1
LMM		242			32474	30.6	2.5
LMM		400			40790	53.4	3

Method	Resolution	Mesh	Vers.	Time Sec.	nPP	nBkg	nLM	nL	kI	n(h*kI)
RRTA	3072x1536			385	2662753	2055839		12		
LMM		100	G	325			24877		2	16.8
			F	210			24419			13.7
			G	808			39034		3	56.6
			F	528			38671			40.7
RRTA	6000x3000			1451	10157280	7842720		12		
LMM		200	F	1704			143612		3	38.7



2nd level - $\sqrt[3]{N}$