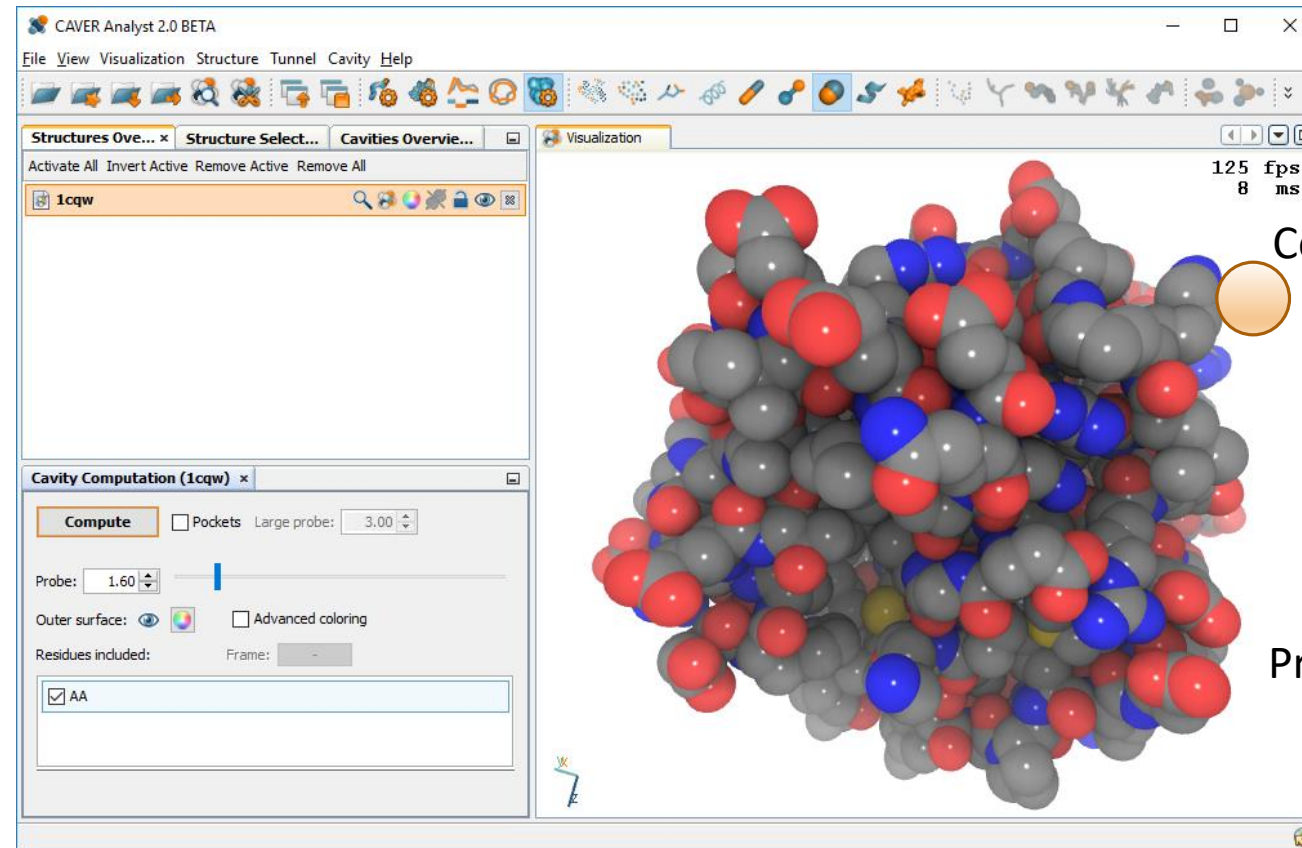


Computing the solvent excluded volume by GPU-based ray casting

MARTIN MAŇÁK

*UNIVERSITY OF WEST BOHEMIA
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
PILSEN, CZECH REPUBLIC*

Motivation – CAVER Analyst 2.0

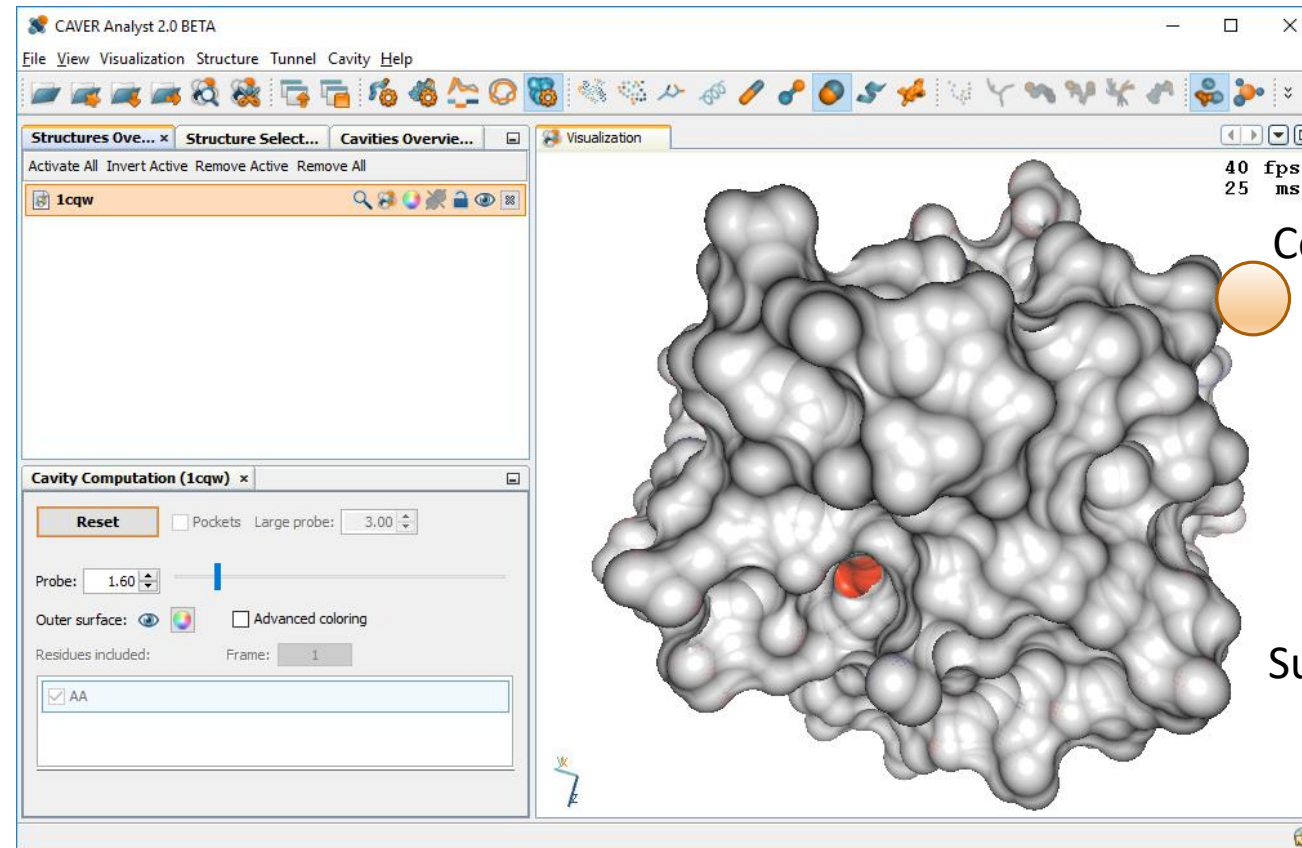


Collision-avoiding probe

Protein model

Jurcik, A. et al, 2018. CAVER Analyst 2.0: analysis and visualization of channels and tunnels in protein structures and molecular dynamics trajectories. *Bioinformatics*, pp. bty386.

Motivation – solvent excluded surface



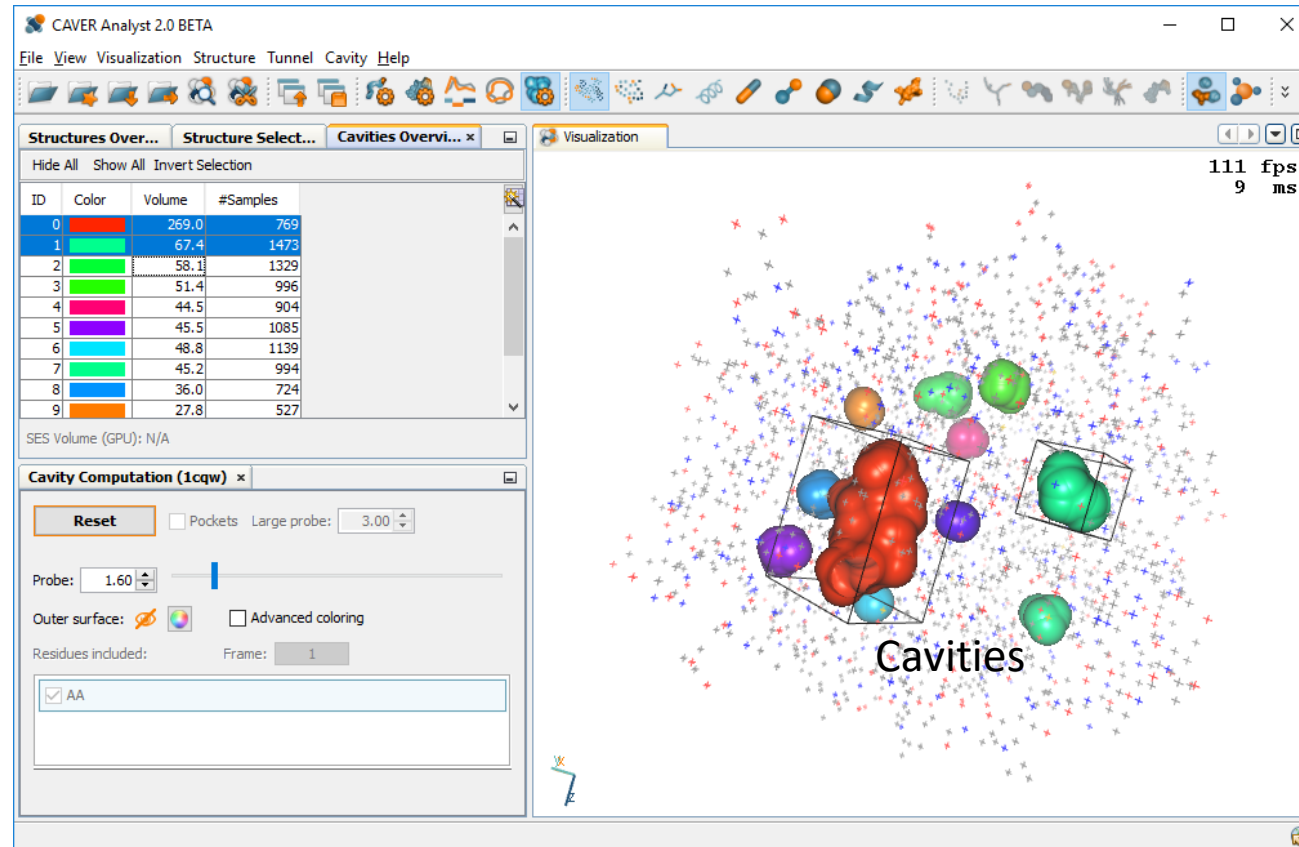
Collision-avoiding probe

Volume(Surface) = ?

Surface

Jurcik, A. et al, 2018. CAVER Analyst 2.0: analysis and visualization of channels and tunnels in protein structures and molecular dynamics trajectories. *Bioinformatics*, pp. bty386.

Motivation – cavities

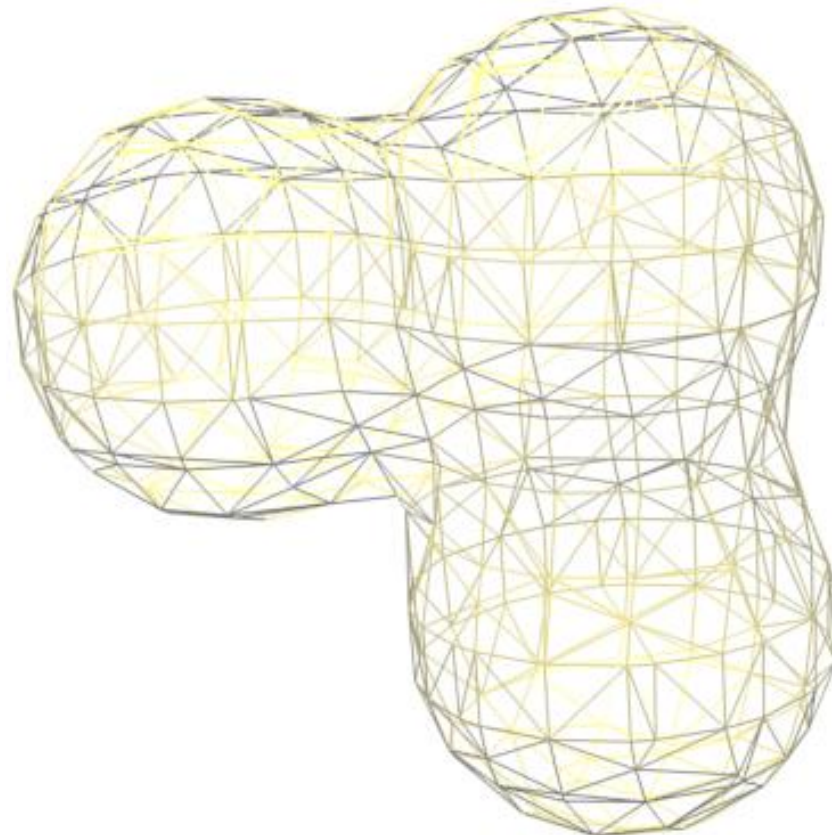
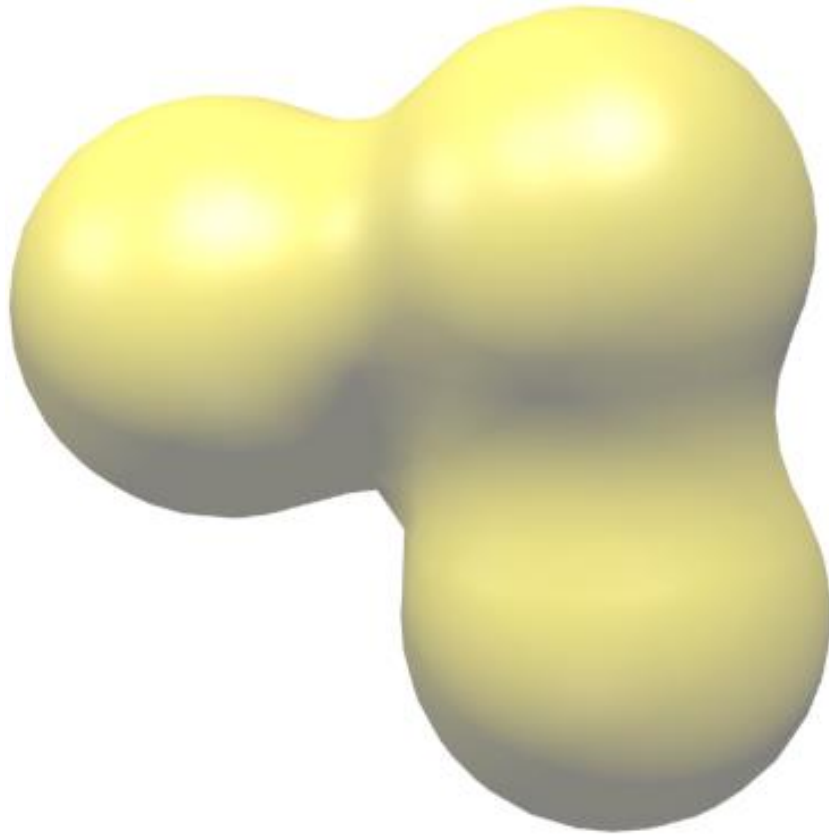


Volume(Cavity) = ?

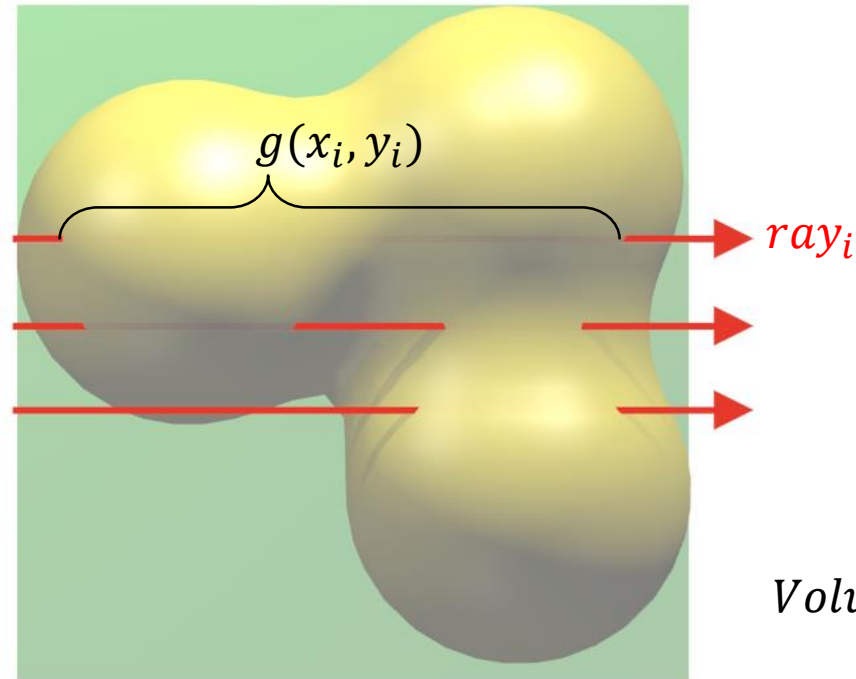
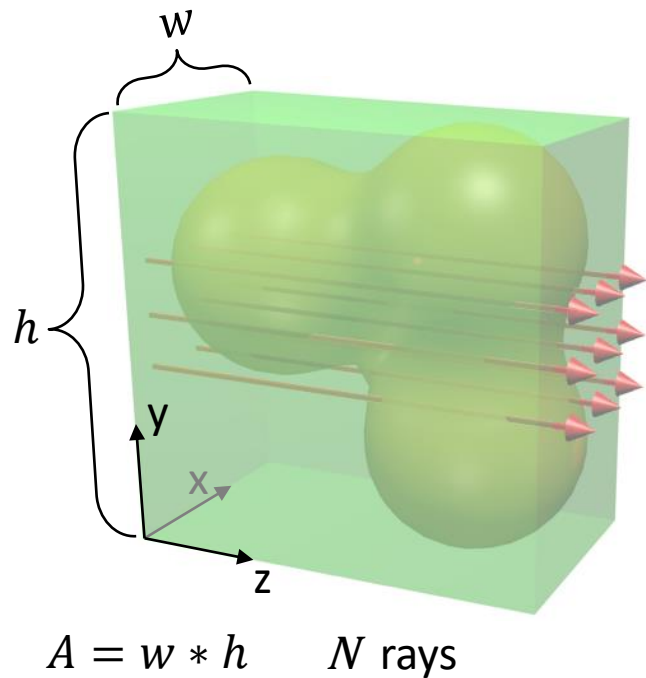
Jurcik, A. et al, 2018. CAVER Analyst 2.0: analysis and visualization of channels and tunnels in protein structures and molecular dynamics trajectories. *Bioinformatics*, pp. bty386.

Triangulation?

Volume computation is easy...



Volume computation by ray casting on CPU



$$Volume \doteq \frac{A}{N} \sum_{i=1}^N g(x_i, y_i)$$

Phillips, M. et al, 2010. Measuring properties of molecular surfaces using ray casting. *2010 IEEE International Symposium on Parallel Distributed Processing, Workshops and Phd Forum (IPDPSW)*, Atlanta, Georgia, pp. 1-7.

Solvent excluded surface (SES)

Van der Waals model

- Protein atoms – partially-overlapping balls (a)

Solvent

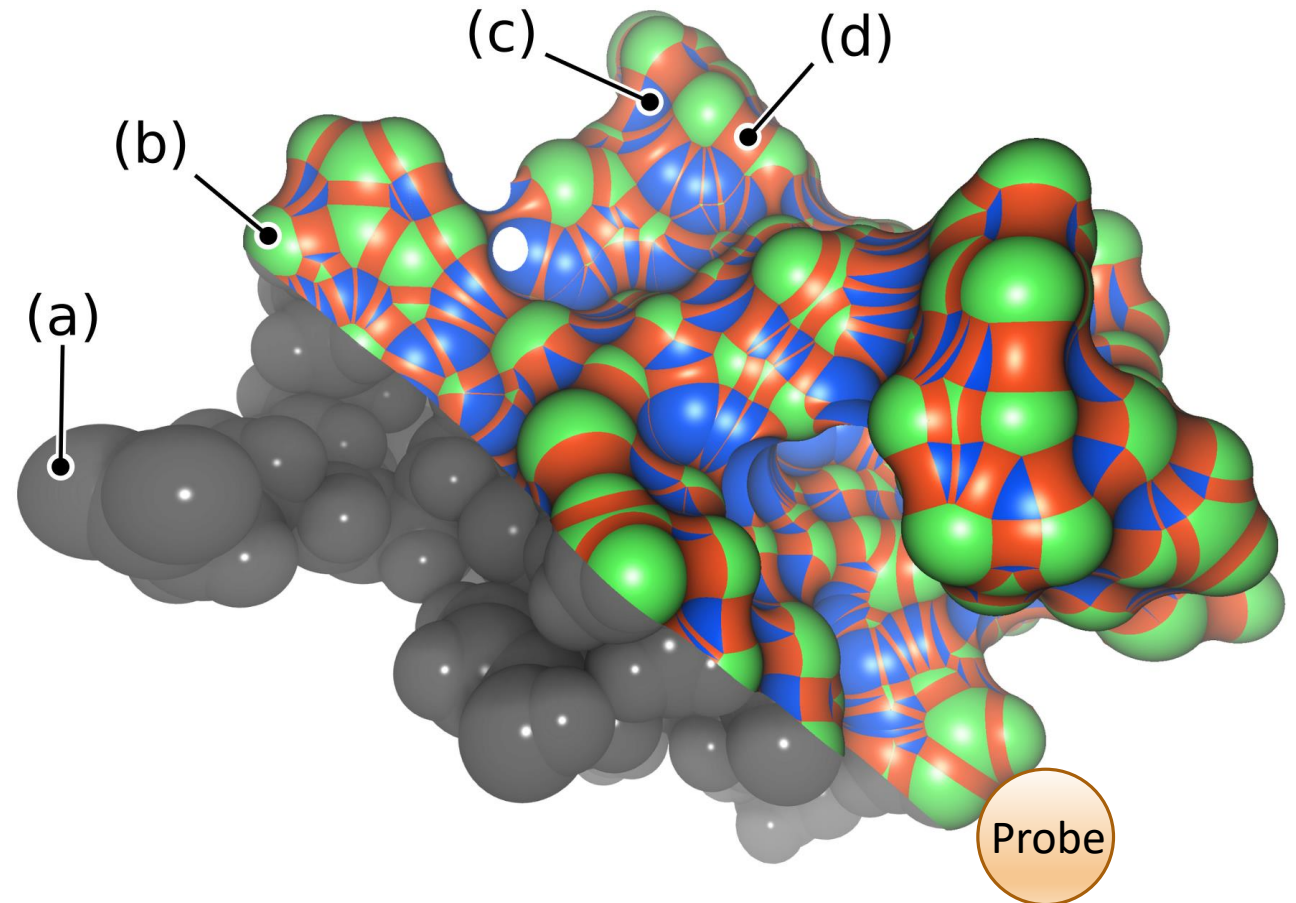
- spherical probe, constant radius

SES

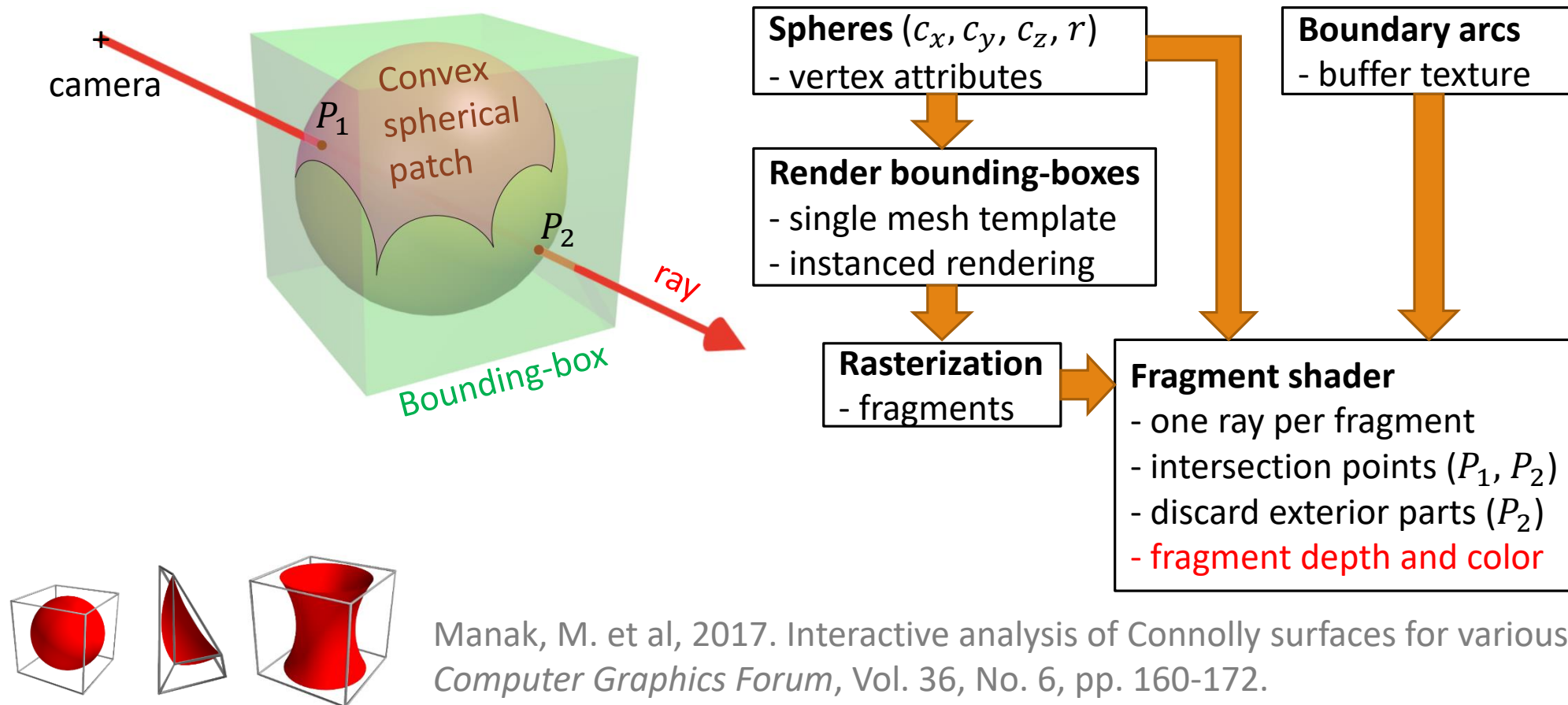
- The topological boundary of the union of all probes that avoid collisions with the van der Waals model

SES-elements

- convex spherical patches (b)
- concave spherical patches (c)
- toroidal patches (d)

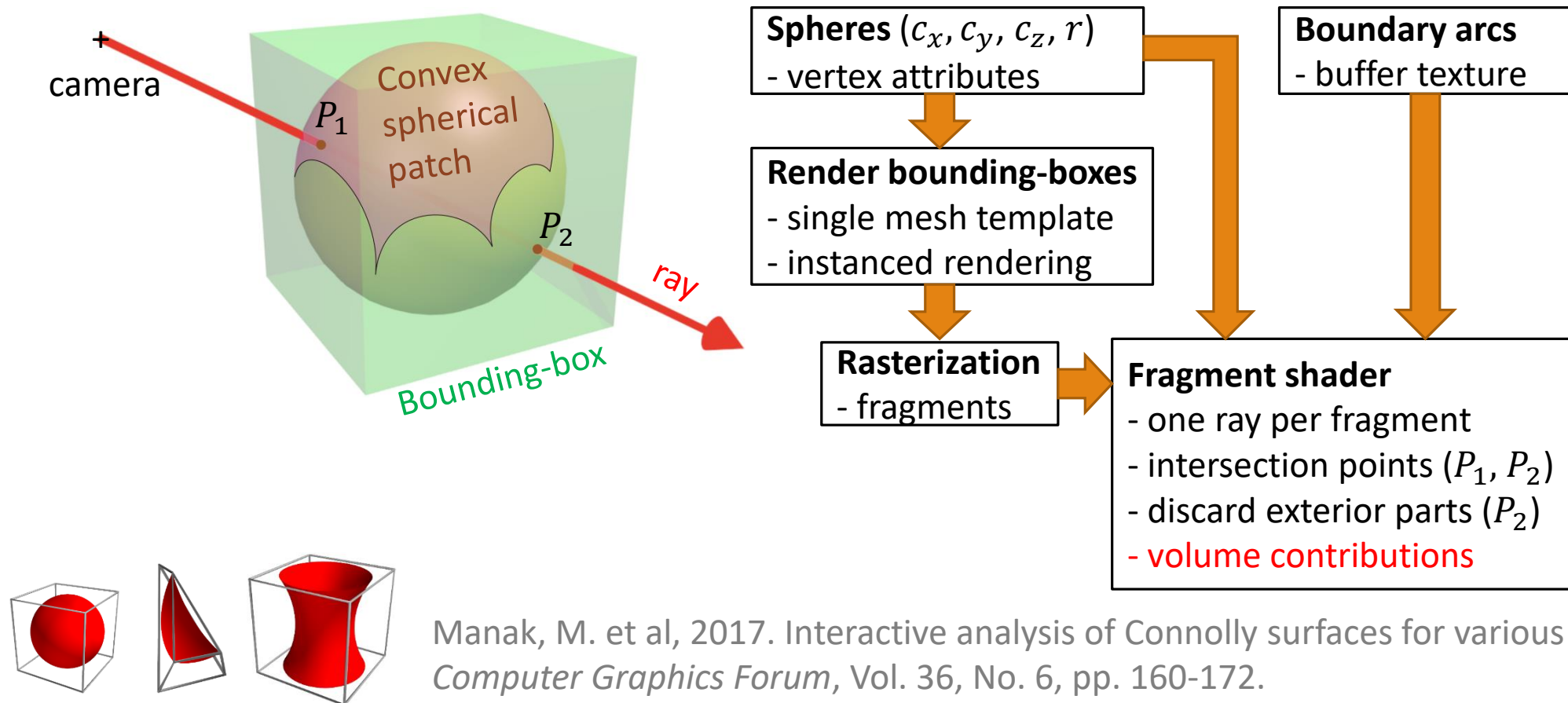


Rendering SES-elements by ray casting for visualization



Manak, M. et al, 2017. Interactive analysis of Connolly surfaces for various probes. *Computer Graphics Forum*, Vol. 36, No. 6, pp. 160-172.

Rendering SES-elements by ray casting for volume computation



Manak, M. et al, 2017. Interactive analysis of Connolly surfaces for various probes. *Computer Graphics Forum*, Vol. 36, No. 6, pp. 160-172.

Rendering setup

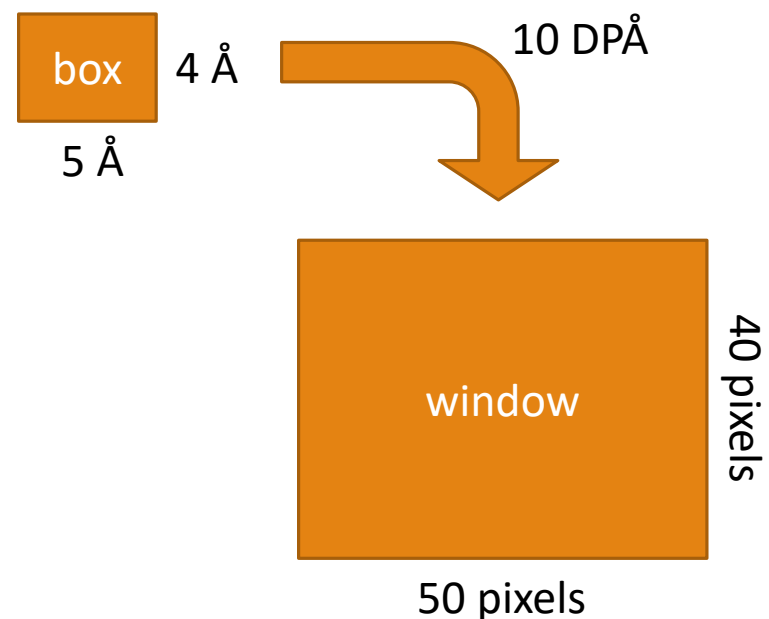
Resolution

- DPA – Dots per angstrom (unit of length, $1 \text{ \AA} = 10^{-10} \text{ m}$)
- Window size from DPA and bounding box extents in the X- and Y-axis (AABB of the whole SES)
- Example: box extents $5 \times 4 \times 3$ and 10 DPA \Rightarrow window size $50 \times 40 \Rightarrow$ 2000 pixels \Rightarrow 2000 rays

Orthographic projection

Destination - empty frame-buffer

- Default width and height (hardware limit 16384×16384)
- No memory allocation (color, z-buffer, ...)
- OpenGL 4.3

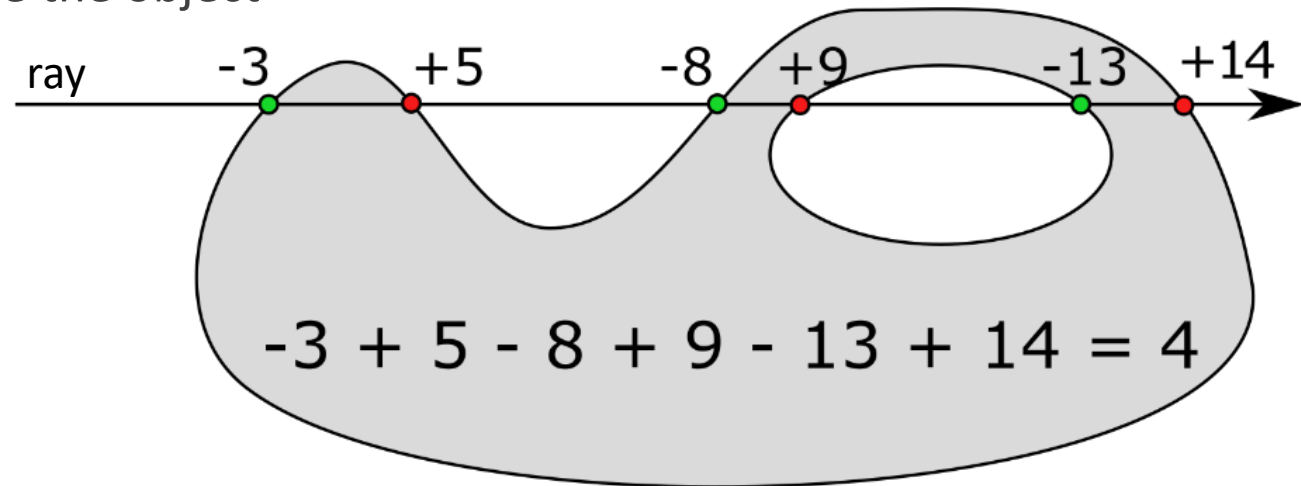


Volume contributions along a ray

Intersection points

- Computed in the fragment shader (ray casting, rays are parallel to the Z-axis)
- Distance from the XY-plane (the Z-coordinate of the point)
- Entry-points negative, exit-points positive (dot-product, $\text{sign}(\text{ray} \cdot \text{normal})$)

The sum = the length of the ray inside the object



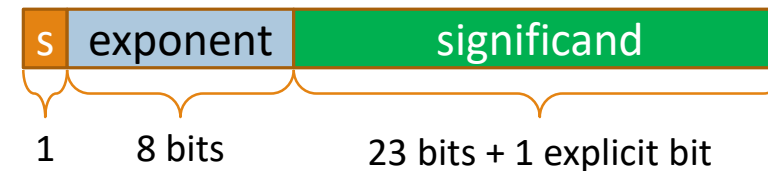
Volume contributions

- Floating-point numbers, SP (32-bits)
- Problem 1: Precision loss
- Problem 2: Thread safety

Floats to big integers

According to IEEE-754 (32-bit floats)

- $f = (-1)^{\text{sign}} \times 2^{\text{exponent}} \times \text{significand}$
- $\text{exponent} \in [-126, 127]$
- f is an integral multiple of the smallest positive subnormal value 2^{-149}

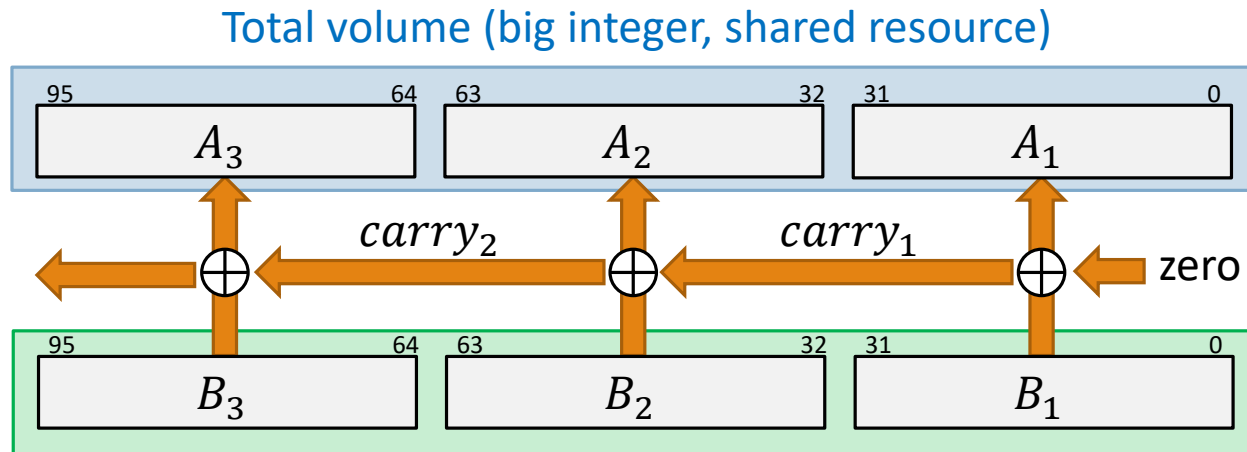


Therefore, $f \times 2^{149}$ is an integer

- $f \rightarrow$ binary rep. (32 bits) $\rightarrow \text{sign}, \text{exponent}, \text{significand} \rightarrow (-1)^{\text{sign}} \times 2^{\text{exponent}+149} \times \text{significand}$
- The result is a big integer (277 bits)

So, we can accumulate volume contributions as big integers without any loss of precision.

Thread-safe addition of big integers on GPU

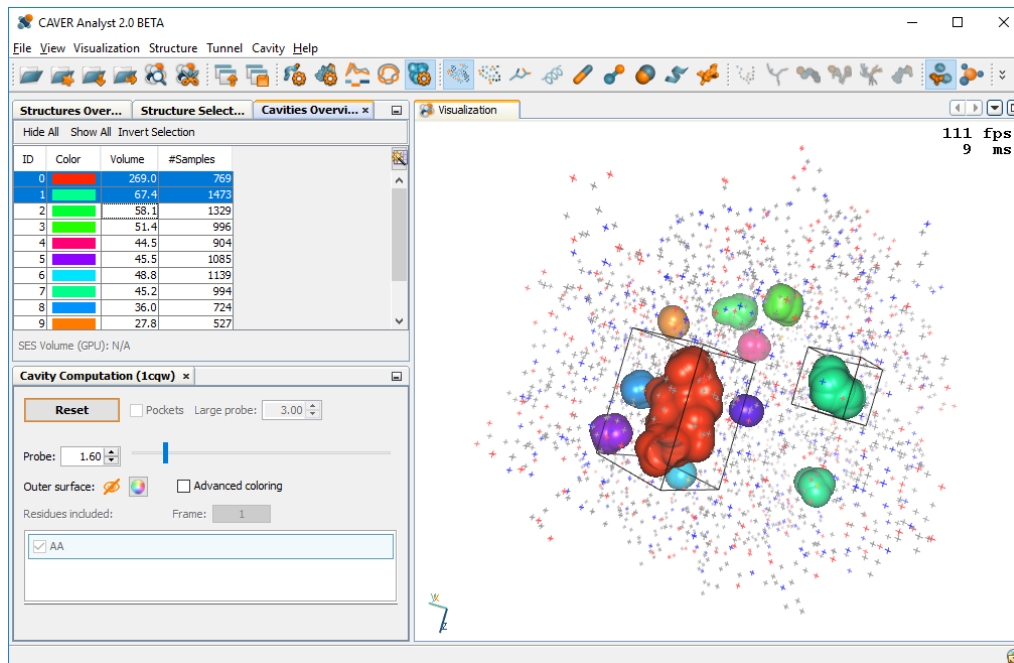


Volume contribution

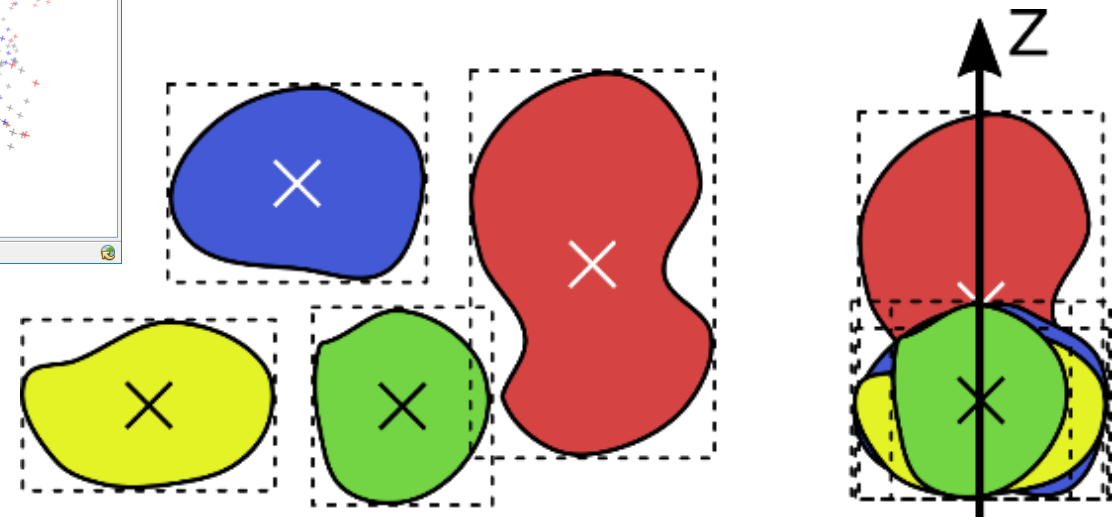
- big integer
- local variable (fragment shader)

```
 $\oplus$  if ( $B_i + carry_{i-1} \neq 0$ ) {  
    uint X = atomicAdd( $A_i$ ,  $B_i + carry_{i-1}$ );  
    uint Y = X +  $B_i + carry_{i-1}$ ;  
     $carry_i = ((X \& B_i) | ((X | B_i) \& \sim Y)) \gg 31$ ;  
} else {  
     $carry_i = B_i == 0 ? 0 : 1$ ;  
}
```

Volume of cavities



- Total volume accumulated per cavity
- Cavities aligned with the z-axis
- The surface of each cavity must be closed
- Only a subset of cavity-clip-planes is used




Waiting for GPU to finish

Easy, just call `glFinish()`...

- This does not always work
- NVIDIA Linux drivers pretend that the “work is done” after 11 seconds

Solution

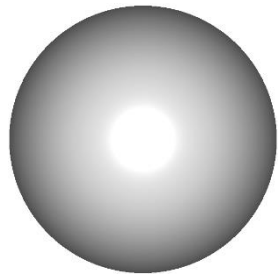
1. Create a fence-sync object
 2. Wait for the sync to become signaled with a timeout (5 seconds?)
 3. Timeout occurred → delete the sync and repeat from step 1.
- 

MS Windows – Timeout Detection and Recovery (TDR)

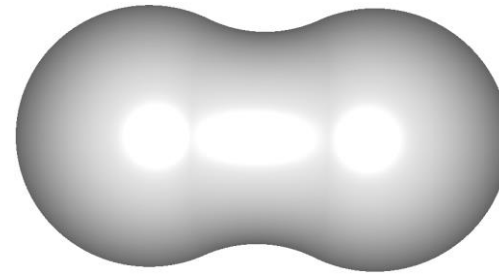
- Registry keys `TdrDelay` and `TdrDdiDelay` must be set to higher values

Experiments and results

Two simple cases and the minimal resolution needed to achieve required number of digits



$$V = \frac{4}{3}\pi \doteq 4.1887902 \text{ \AA}^3$$

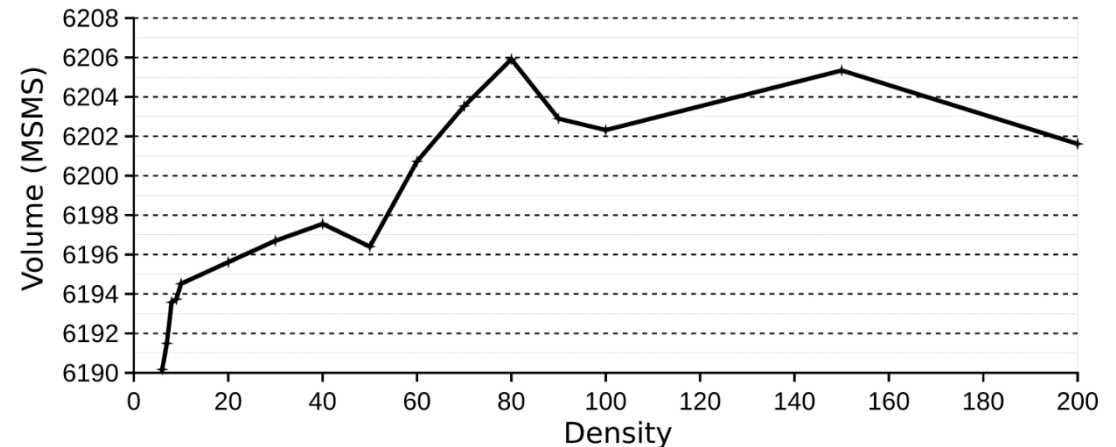
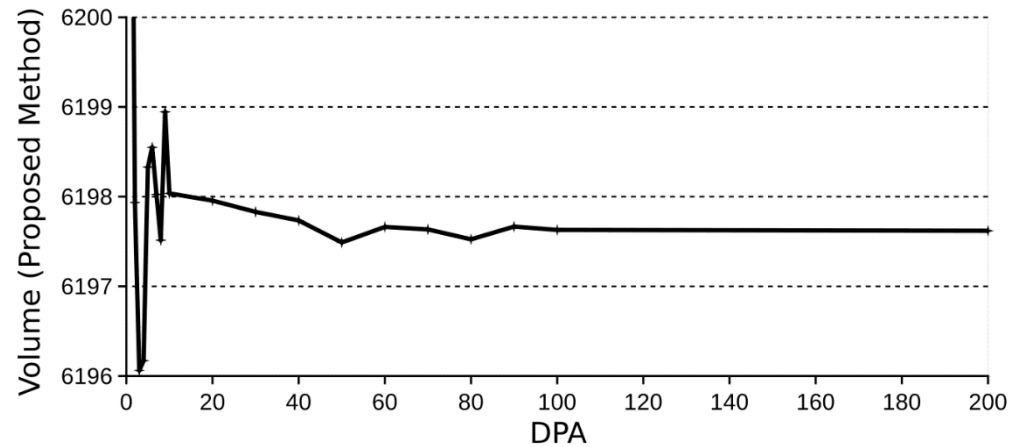
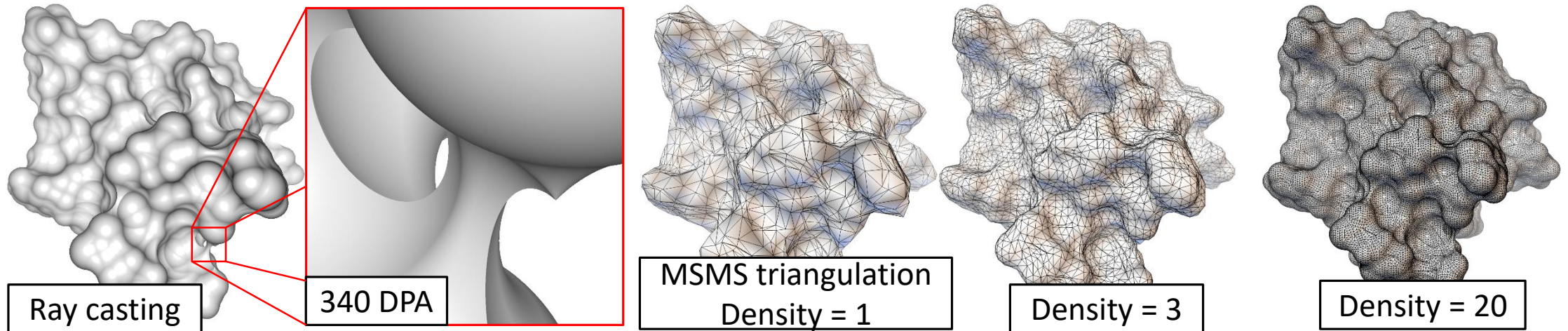


$$V \doteq 14.75567 \text{ \AA}^3$$

| Unit sphere | Proposed method | | MSMS | | 2 atoms | Proposed method | | MSMS | |
|---------------------|-----------------|-----------|---------|-----------|---------------------|-----------------|---------|---------|-----------|
| V [Å ³] | DPA | Window | Density | Triangles | V [Å ³] | DPA | Window | Density | Triangles |
| 4 | 2 | 4×4 | 11 | 248 | 14 | 1 | 5×5 | 5 | 276 |
| 4.1 | 7 | 14×14 | 23 | 556 | 14.7 | 5 | 23×23 | 64 | 4024 |
| 4.18 | 39 | 78×78 | 215 | 5344 | 14.75 | 20 | 45×45 | – | – |
| 4.188 | 65 | 130×130 | 1423 | 35490 | 14.755 | 30 | 133×133 | – | – |
| 4.18879 | 316 | 632×632 | – | – | 14.7556 | 201 | 885×885 | – | – |
| 4.188790 | 1621 | 3242×3242 | – | – | | | | | |

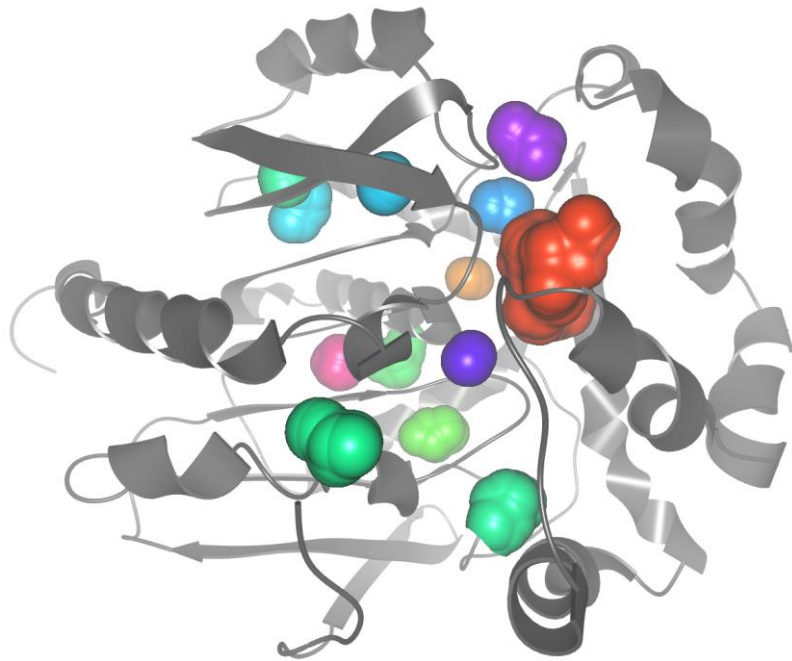
MSMS – Sanner, M. F. et al, 1996. Reduced surface: An efficient way to compute molecular surfaces. *Biopolymers*, Vol. 38, No. 3, pp. 305-320.

SES volume of insulin



SES volume, cavities

The method as a plugin in CAVER Analyst



New

Old – Monte Carlo sampling

| Color | Volume (GPU) | Volume | #Samples |
|-------------|--------------|--------|----------|
| Red | 278.4 | 276.1 | 20769 |
| Green | 65.0 | 65.6 | 21473 |
| Cyan | 60.0 | 59.7 | 21329 |
| Magenta | 50.9 | 50.7 | 20996 |
| Purple | 42.4 | 42.4 | 20904 |
| Blue | 46.1 | 46.5 | 21085 |
| Orange | 49.3 | 49.4 | 21139 |
| Dark Blue | 44.9 | 44.5 | 20994 |
| Light Blue | 35.9 | 35.8 | 20724 |
| Yellow | 27.3 | 27.5 | 20527 |
| Light Green | 24.3 | 24.6 | 20476 |
| Light Blue | 23.6 | 23.8 | 20449 |
| Dark Blue | 21.4 | 21.5 | 20407 |

Performance measurements

| 1A4U – 3926 atoms (medium size protein model) | | | | |
|---|-------------|---------------|---------------------|-----------|
| DPA | Window | Intersections | V [Å ³] | Time [ms] |
| 1 | 62×62 | 10632 | 61542.2 | 3 |
| 4 | 248×248 | 169760 | 61571.9 | 3 |
| 10 | 620×620 | 1060913 | 61592.5 | 17 |
| 50 | 3098×3098 | 26489332 | 61592.1 | 83 |
| 100 | 6195×6195 | 105924522 | 61591.2 | 240 |
| 265 | 16384×16384 | 740891998 | 61590.4 | 1399 |

| 1AON – 58688 atoms (large protein model) | | | | |
|--|-------------|---------------|---------------------|-----------|
| DPA | Window | Intersections | V [Å ³] | Time [ms] |
| 1 | 148×148 | 186292 | 868912.5 | 16 |
| 4 | 589×589 | 2953760 | 869336.3 | 18 |
| 10 | 1472×1472 | 18446945 | 869160.7 | 88 |
| 50 | 7357×7357 | 460809231 | 869154.5 | 1084 |
| 100 | 14714×14714 | 1843246275 | 869153.2 | 3936 |
| 111 | 16384×16384 | 2270912620 | 869117.1 | 4823 |

SES probe radius 1.4 Å

GPU: AMD Radeon RX 480 Graphics (POLARIS10)

Phillips et al. 2010 – ray casting on CPU, 1A4U, max 200x200, 270 ms on Intel Core 2 Quad, 2.66 GHz

MSMS – triangulation: 1A4U, density 25.6, 1064270 faces, 1460 ms on Intel Core i7-4930K, 3.4 GHz

Conclusion

The proposed method computes SES volume by ray casting on GPU

GPU-computation → larger windows (16384×16384 vs. 200×200) & better performance

Fast computation thanks to parallelism with atomic operations on GPU

Limitation – numeric operations in 32-bit floats

Achieved four or five decimal digits of the result (sufficient for SES cavities)

Questions & Answers