Cubical Marching Squares: Adaptive Feature Preserving Surface Extraction from Volume Data

Chien-Chang Ho   Fu-Che Wu   Bing-Yu Chen
Yung-Yu Chuang   Ming Ouhyoung

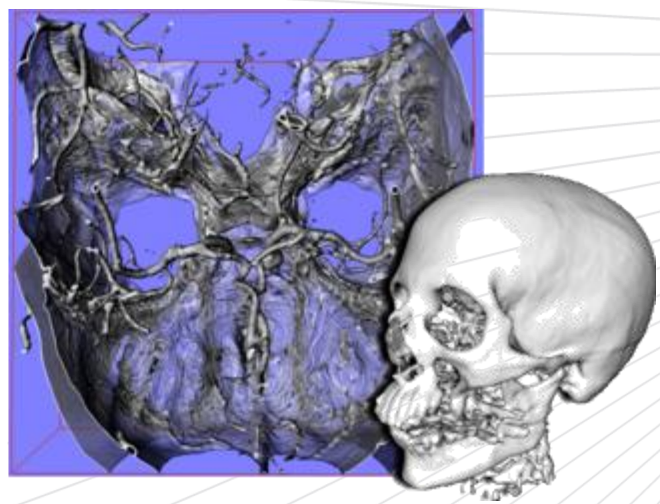
National Taiwan University

Overview: Marching Cubes

- The most cited paper in history of SIGGRAPH
  - http://www.siggraph.org/conferences/reports/s2004/articles/Visualizing_SIGGRAPH.html

William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Proceedings of ACM SIGGRAPH 1987*, pages 163–169.

from www.nasa.gov



from www.openqvis.com

from graphics.csie.ntu.edu.tw

Overview: Problems & Goals

- Adaptive Resolution
- Consistent Topology
- Sharp Features
- Parallel Processing

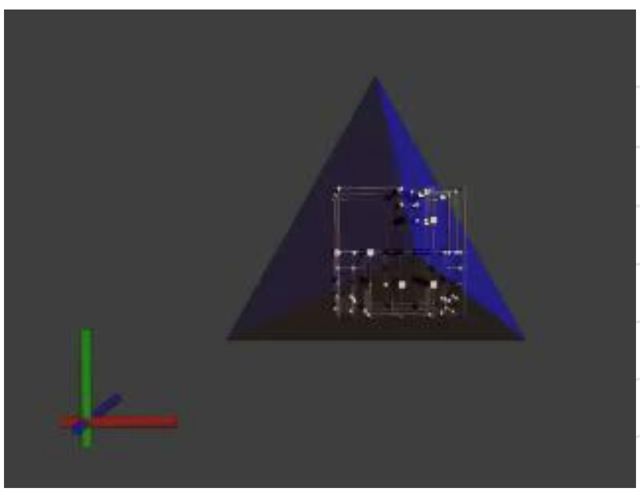
Overview: Problems & Goals

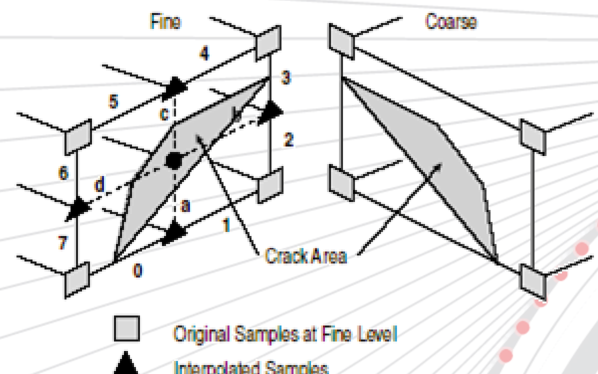
Adaptive Resolution

Consistent Topology

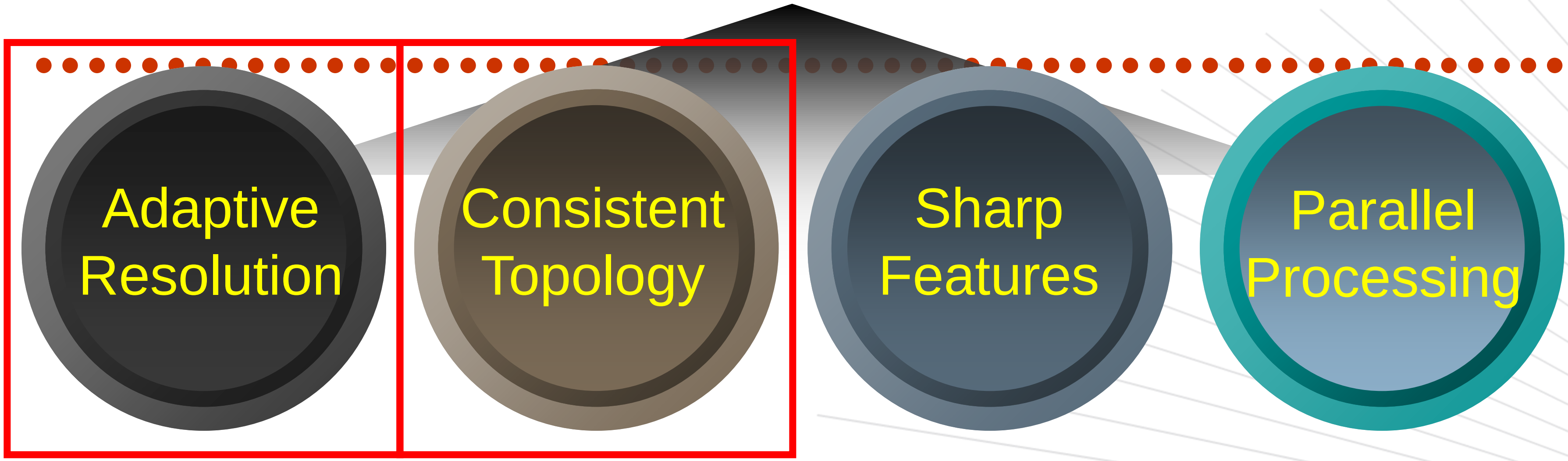
Sharp Features

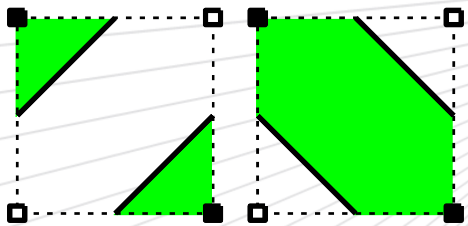
Parallel Processing





Overview: Problems & Goals





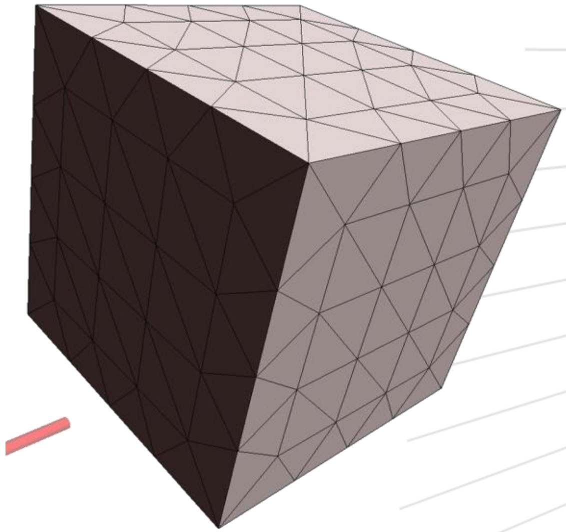
Overview: Problems & Goals

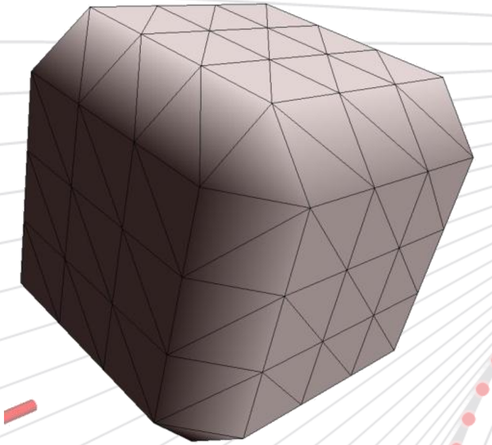
Adaptive Resolution

Consistent Topology

Sharp Features

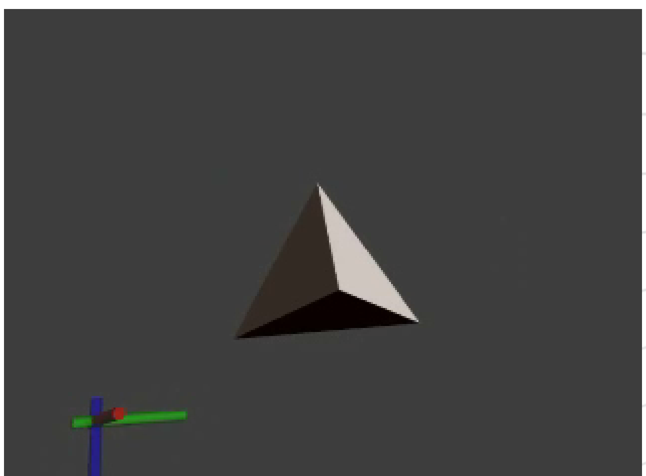
Parallel Processing

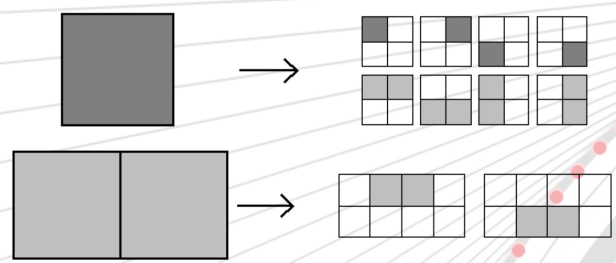




Overview: Problems & Goals

- Adaptive Resolution
- Consistent Topology
- Sharp Features
- Parallel Processing





Outline

- Previous work & Problems
- Solutions
- Results & Conclusion

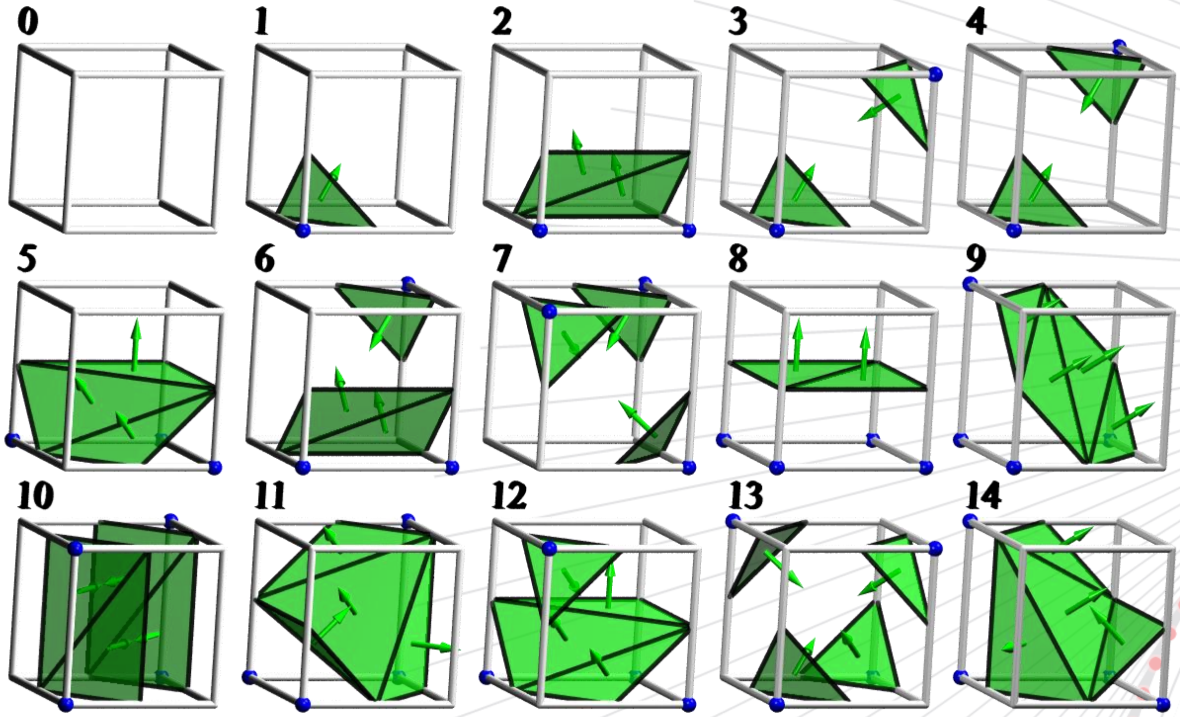
Previous work & Problems

Solutions

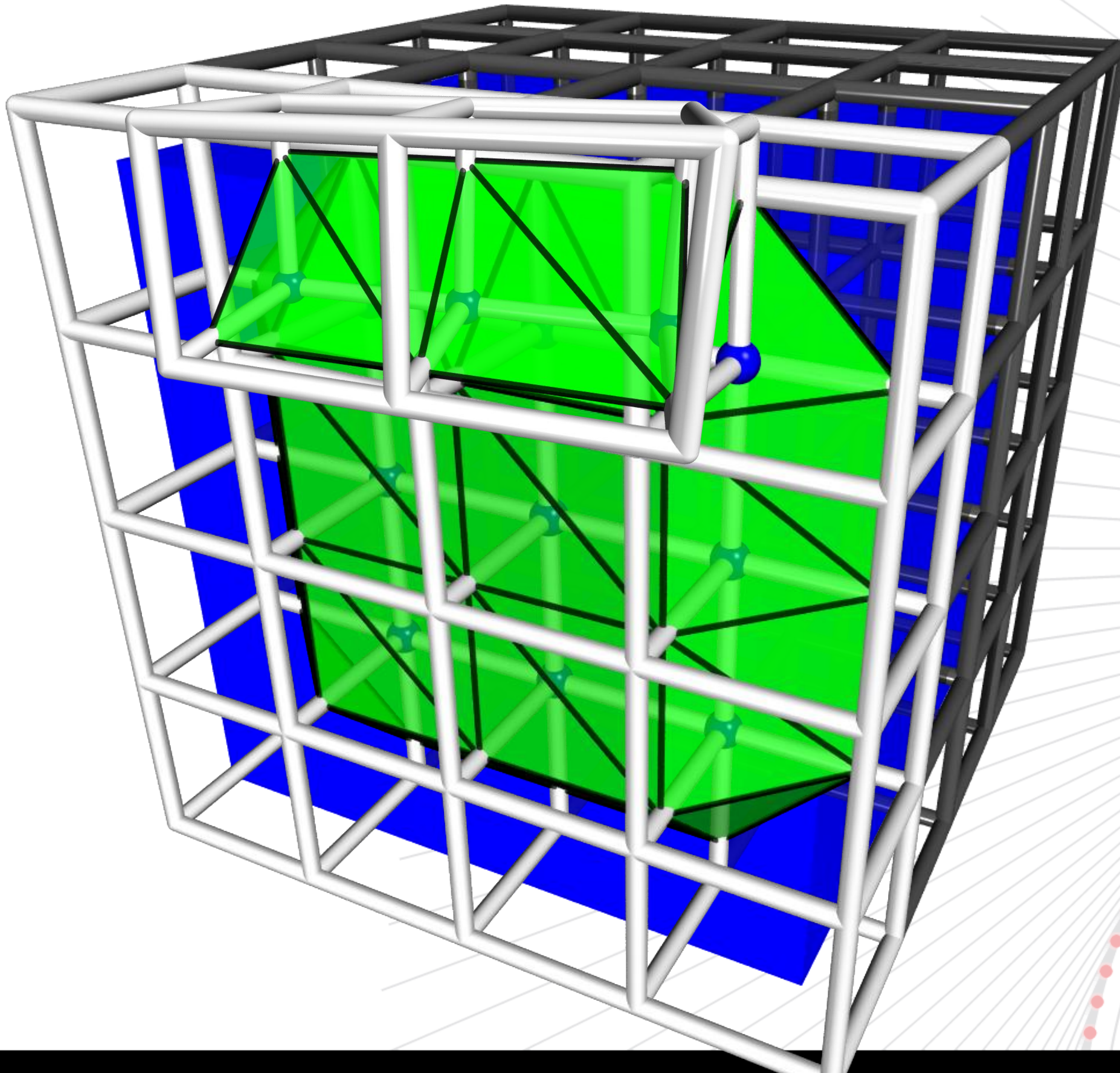
Results & Conclusion

Marching Cubes Table

- Using binary pattern of eight vertices
- Totally 256 cases in 15 configurations

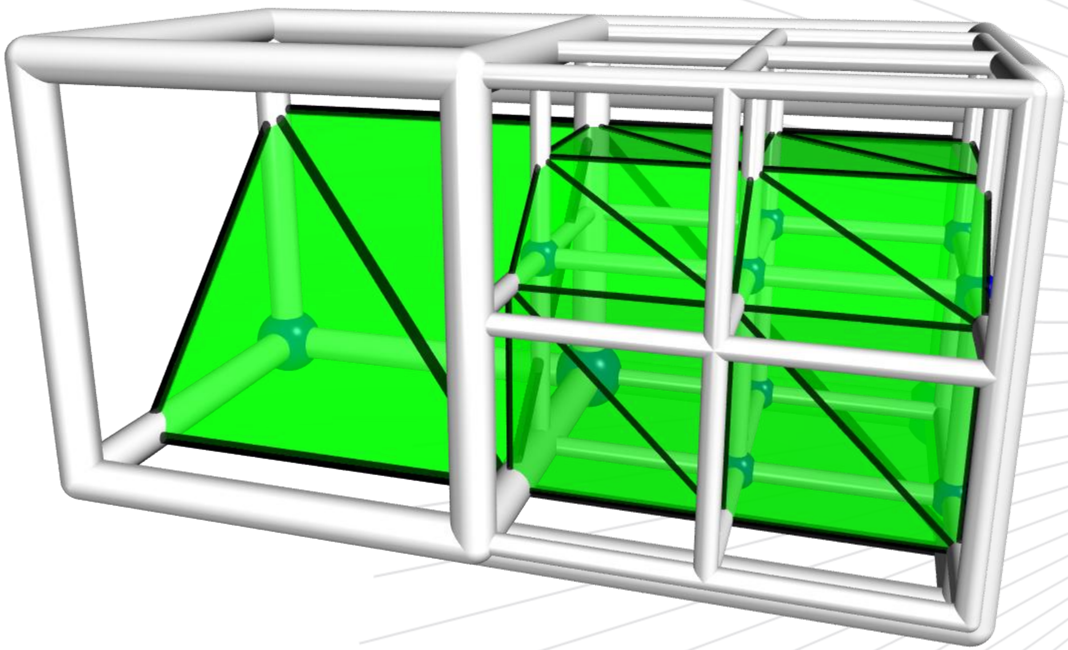


Steps of marching cubes



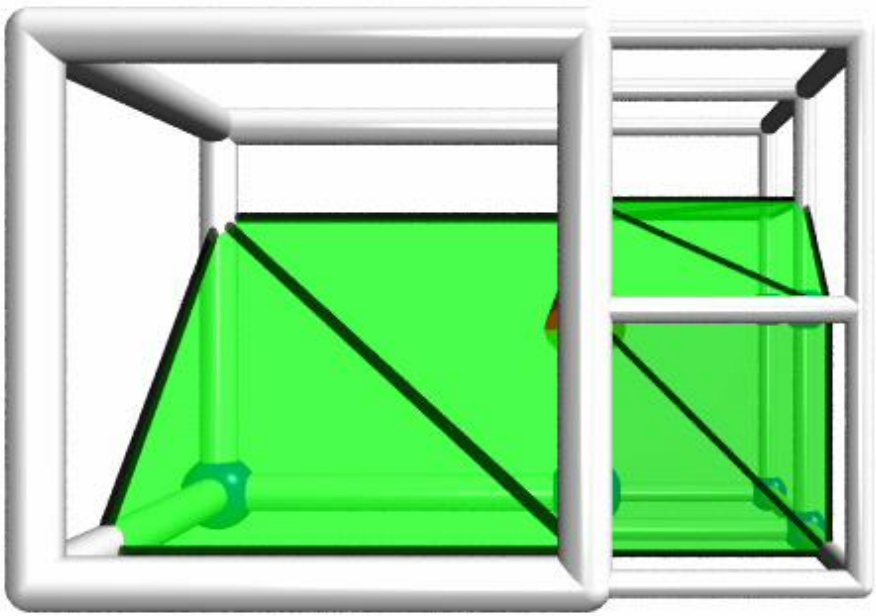
Adaptive resolution

- [WILHELMS J., GELDER A. V.,1992]
- [SHU R. et al, 1995], etc.

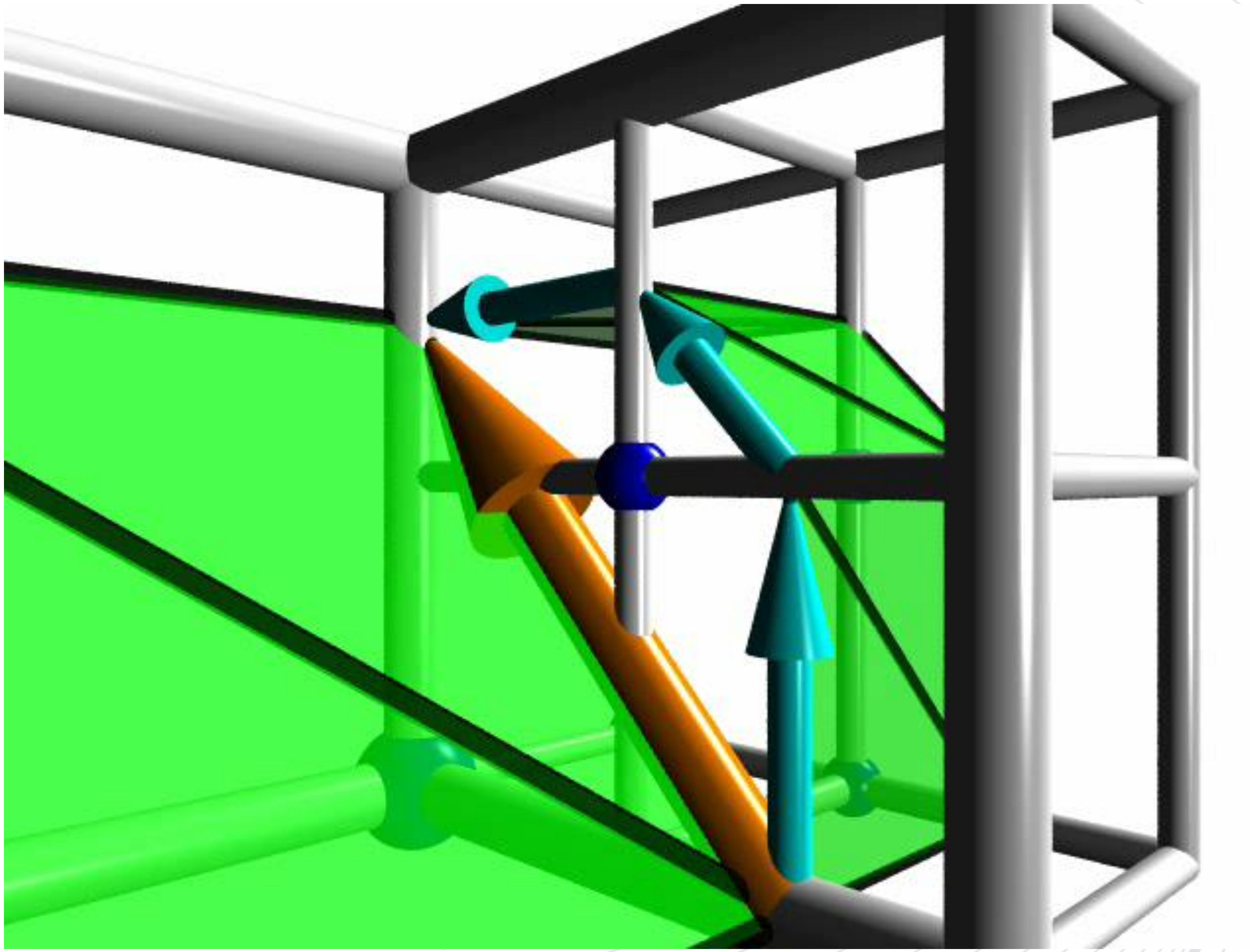


Adaptive resolution

Adaptive Resolution



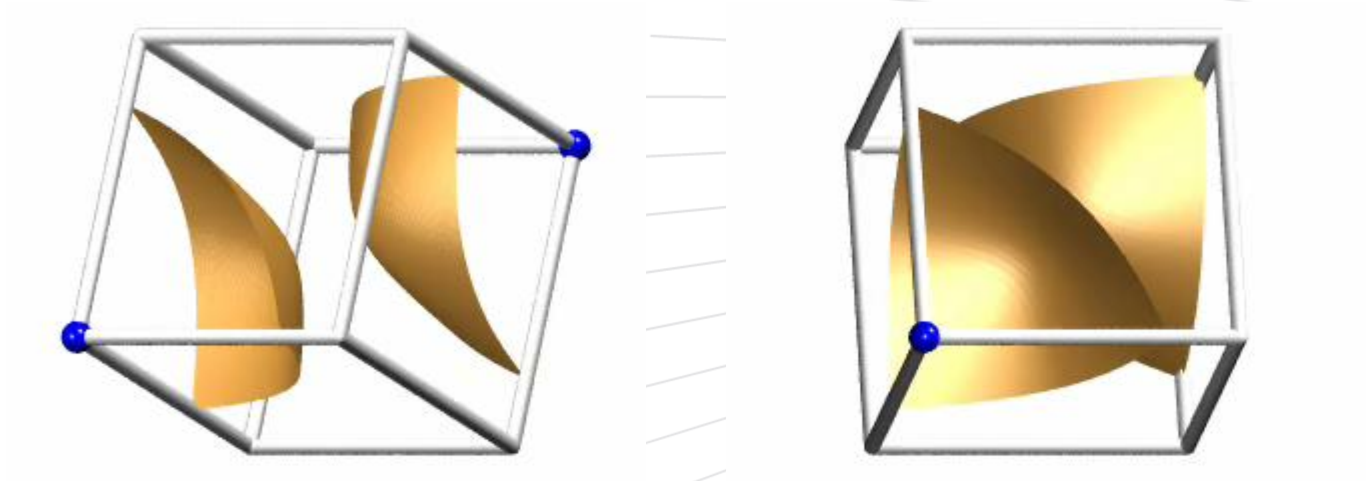
Crack patching



Consistent topology

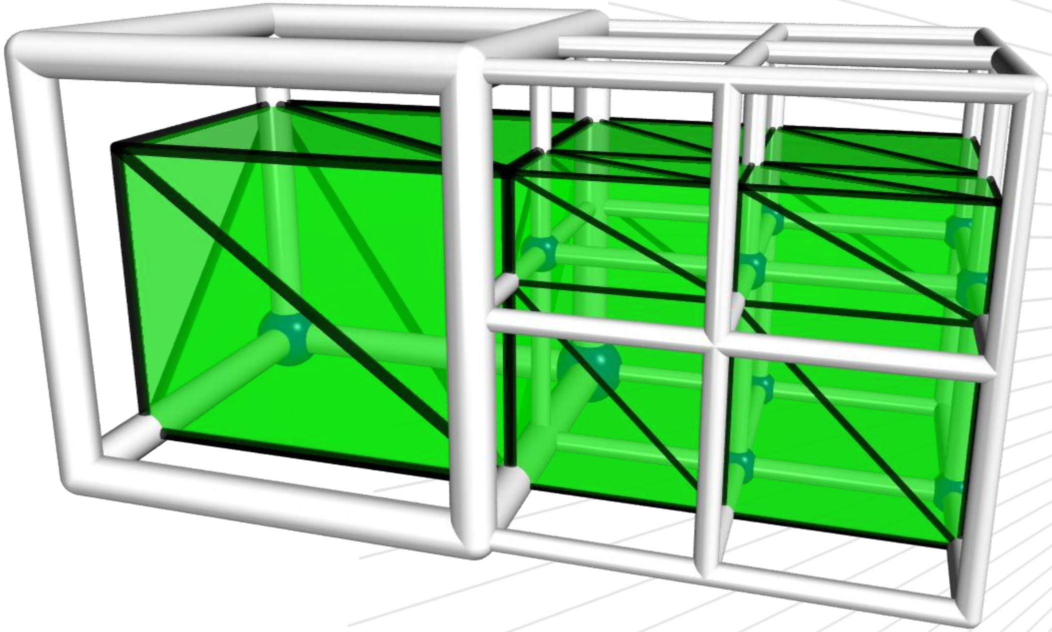
Consistent Topology

- Ambiguity problems
  - [NIELSON G. M., HAMANN B. 1991]
  - [NATARAJAN B. K., 1994]
  - [CHERNYAEV E., 1995], etc.



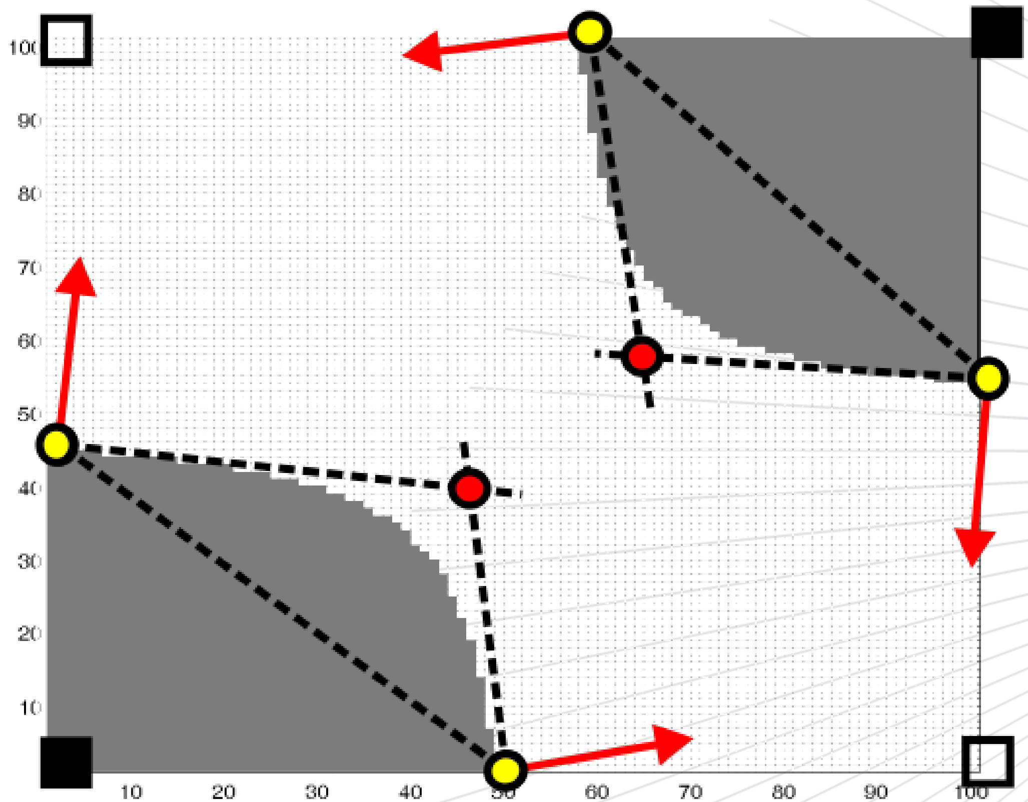
Sharp features

- [KOBBELT L. P. et al, 2001], EMC (Extended Marching Cubes).
- [JU T. et al, 2002], DC (Dual Contouring)



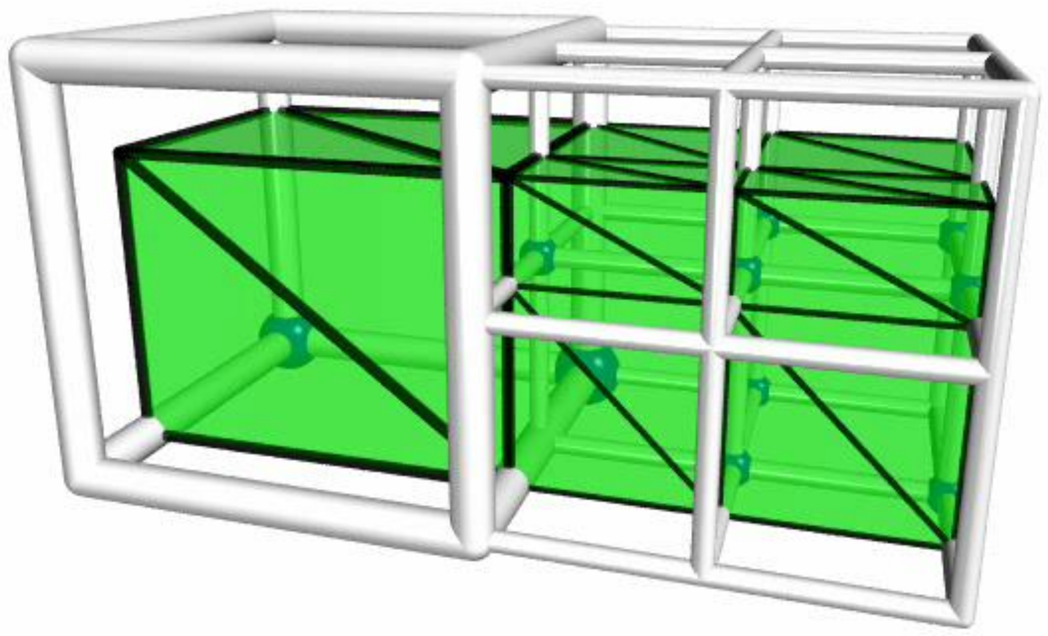
Hermite data: with additional normal direction

Sharp Features

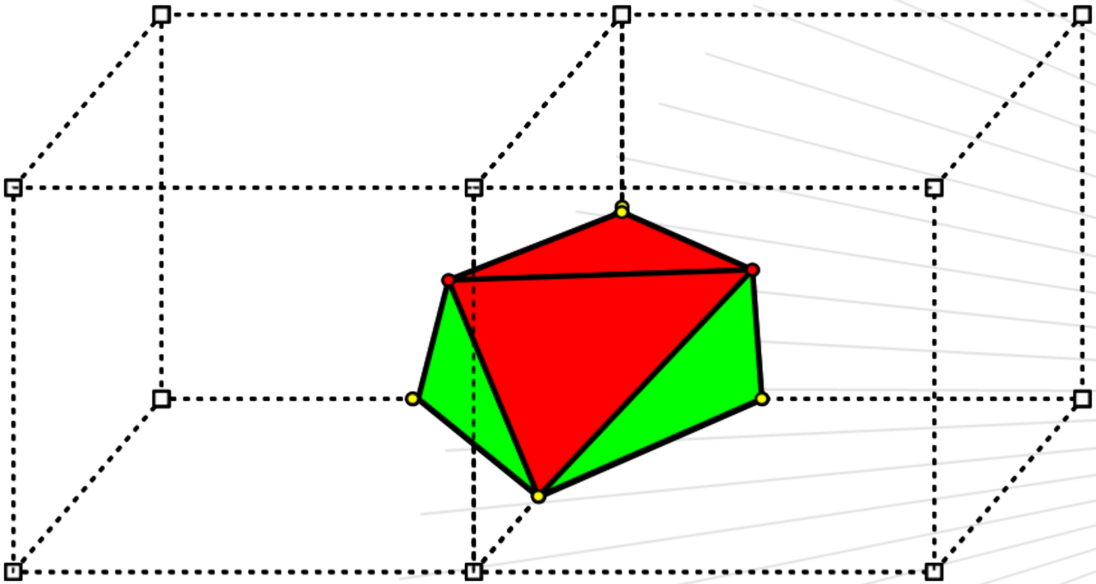


Real-time through parallel processing

- limited by inter-cell dependency

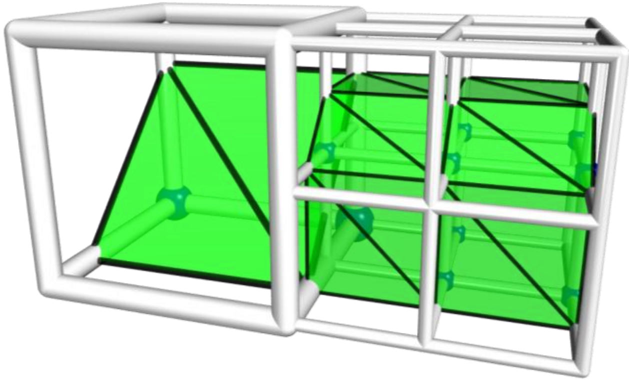


Inter-cell dependency

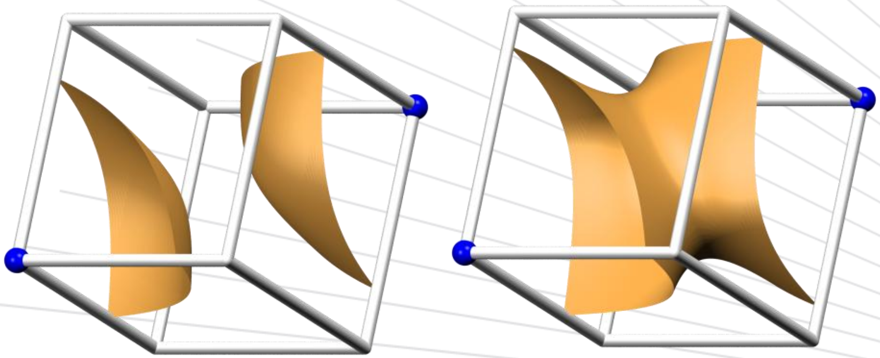


Problems & Goals

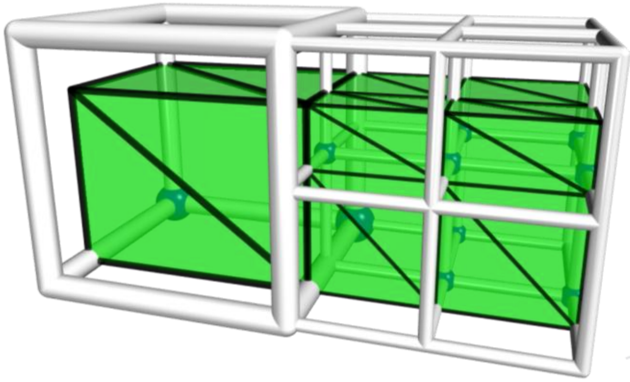
- Adaptive resolution



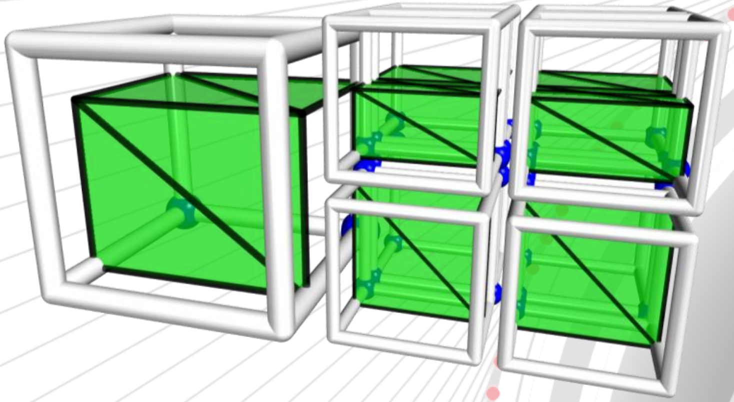
- Consistent topology



- Sharp features



- Parallel processing



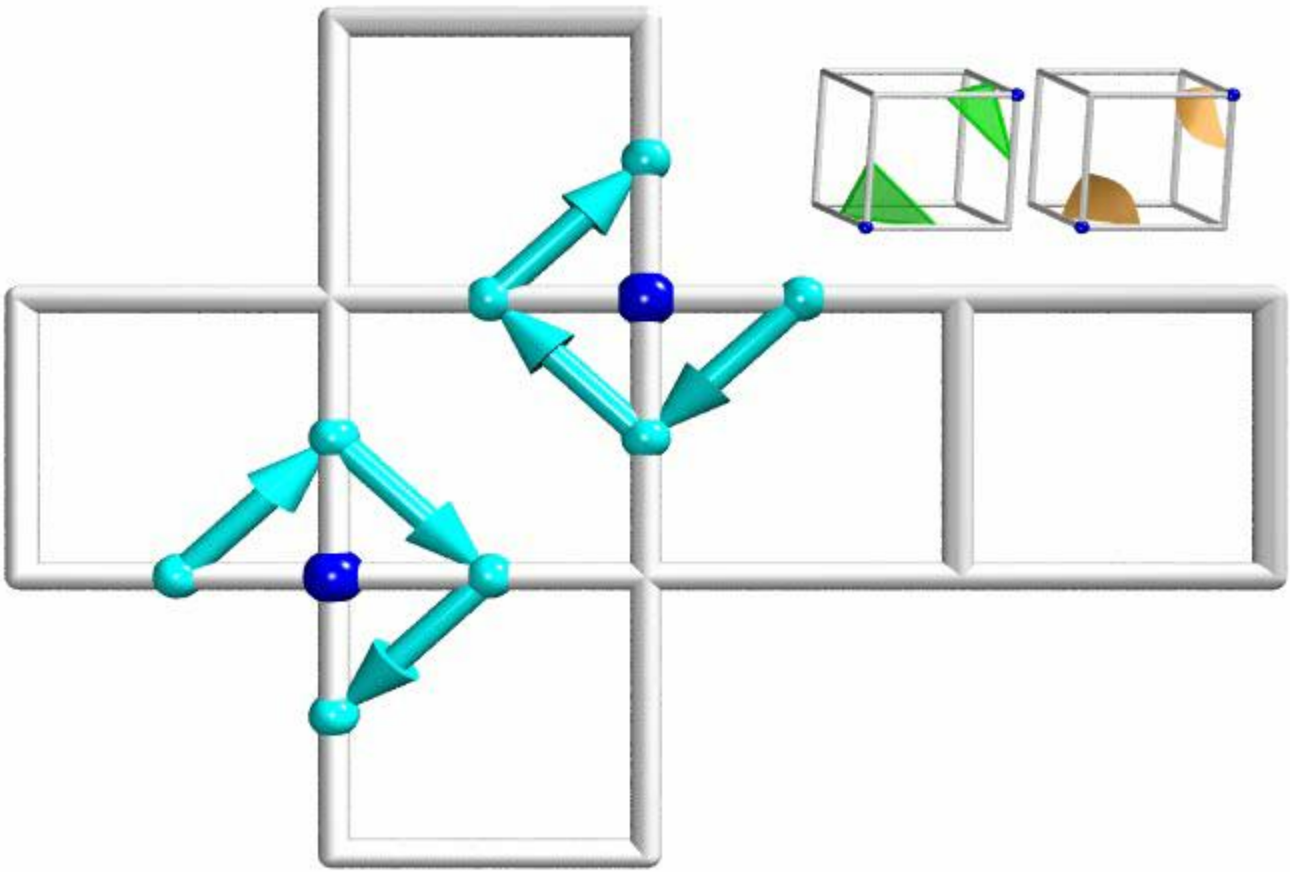
Previous work & Problems

Solutions

Results & Conclusion

Cubical Marching Squares

Adaptive Resolution

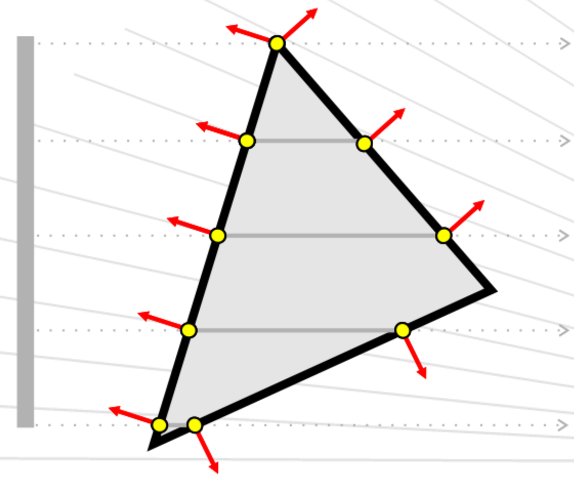


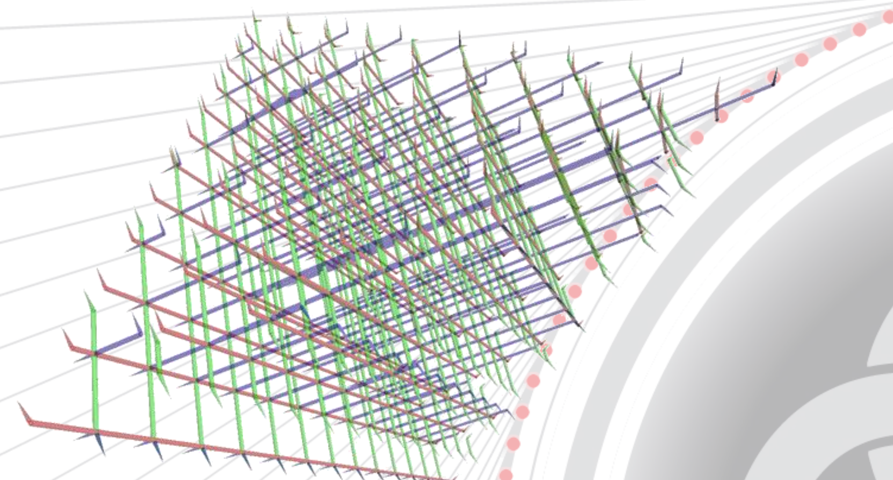
CMS Algorithm

```
procedure CubicalMarchingSquares(HermiteData H)
    InitializeBaseGrid(B);                    ▷ initialize a coarse base grid B
    for each cell c in B
        SubdivideCell(H, c);
    end for
    for each leaf face f
        GenerateSegment(f);
    end for
    for each leaf cell c
        ExtractSurface(c);
    end for
end procedure
```

CMS Algorithm

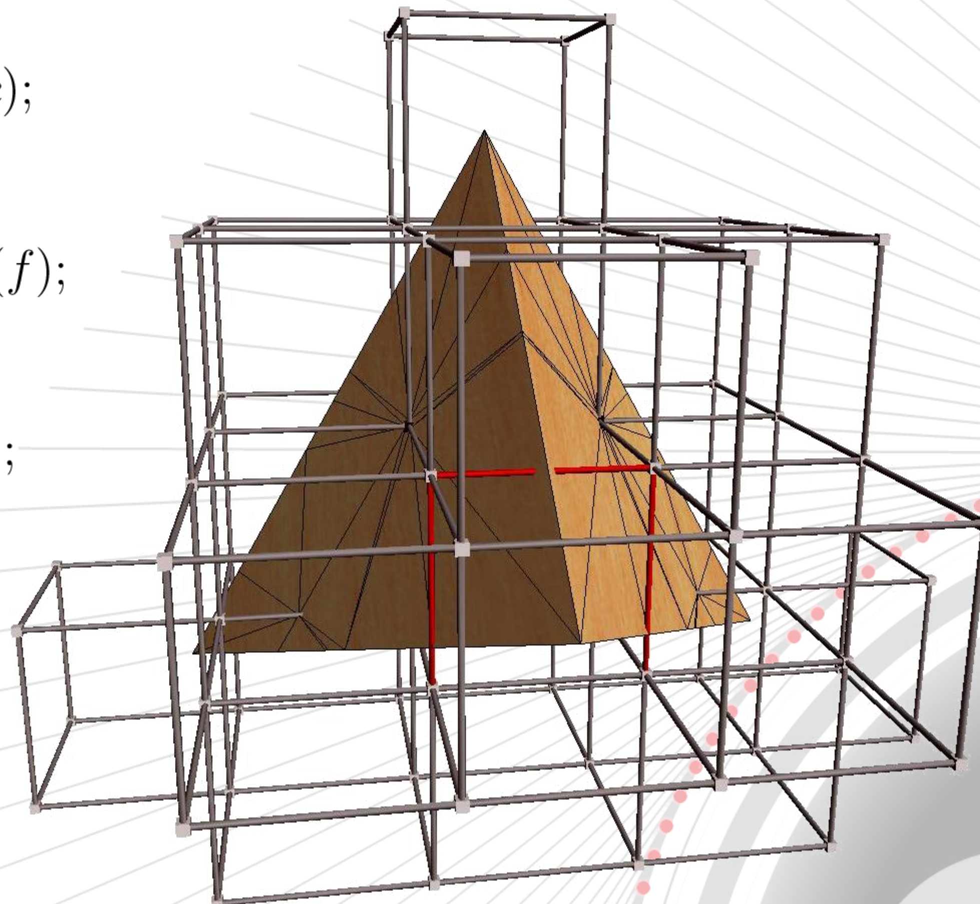
```
procedure CubicalMarchingSquares(HermiteData H)
    InitializeBaseGrid(B);                    ▷ initialize a coarse base grid B
    for each cell c in B
        SubdivideCell(H, c);
    end for
    for each leaf face f
        GenerateSegment(f);
    end for
    for each leaf cell c
        ExtractSurface(c);
    end for
end procedure
```





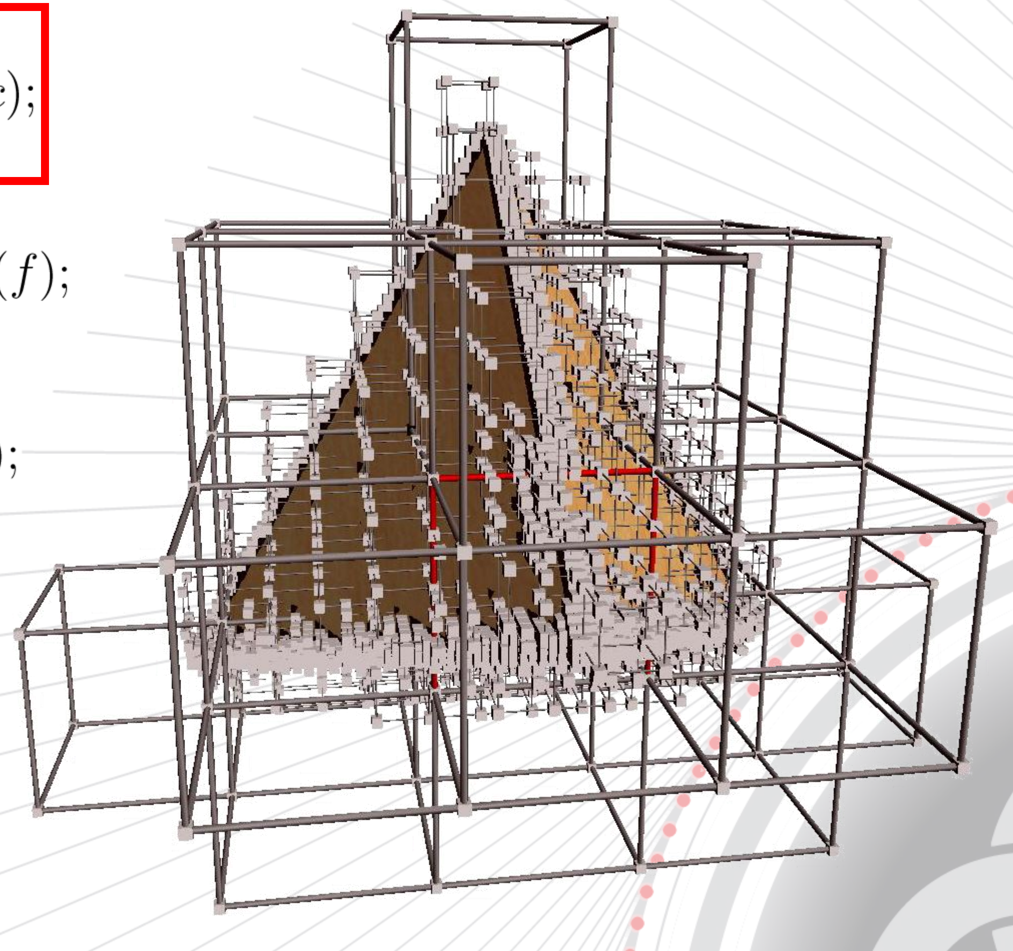
CMS Algorithm

procedure CubicalMarchingSquares(HermiteData H)
    InitializeBaseGrid(B);   ▷ initialize a coarse base grid B
    for each cell c in B
        SubdivideCell(H, c);
    end for
    for each leaf face f
        GenerateSegment(f);
    end for
    for each leaf cell c
        ExtractSurface(c);
    end for
end procedure



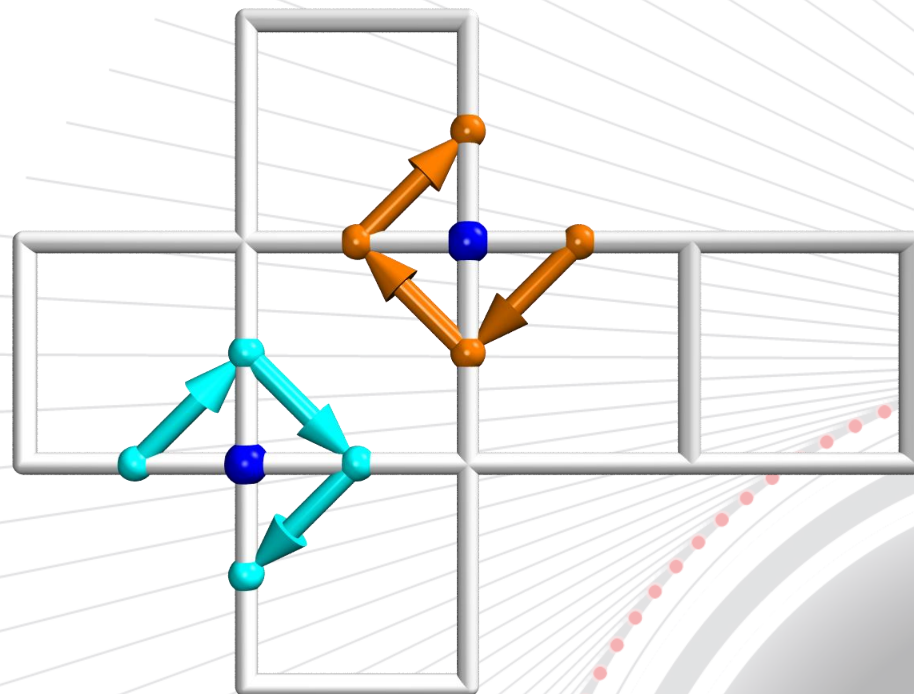
CMS Algorithm

```
procedure CubicalMarchingSquares(HermiteData H)
    InitializeBaseGrid(B);                    ▷ initialize a coarse base grid B
    for each cell c in B
        SubdivideCell(H, c);
    end for
    for each leaf face f
        GenerateSegment(f);
    end for
    for each leaf cell c
        ExtractSurface(c);
    end for
end procedure
```

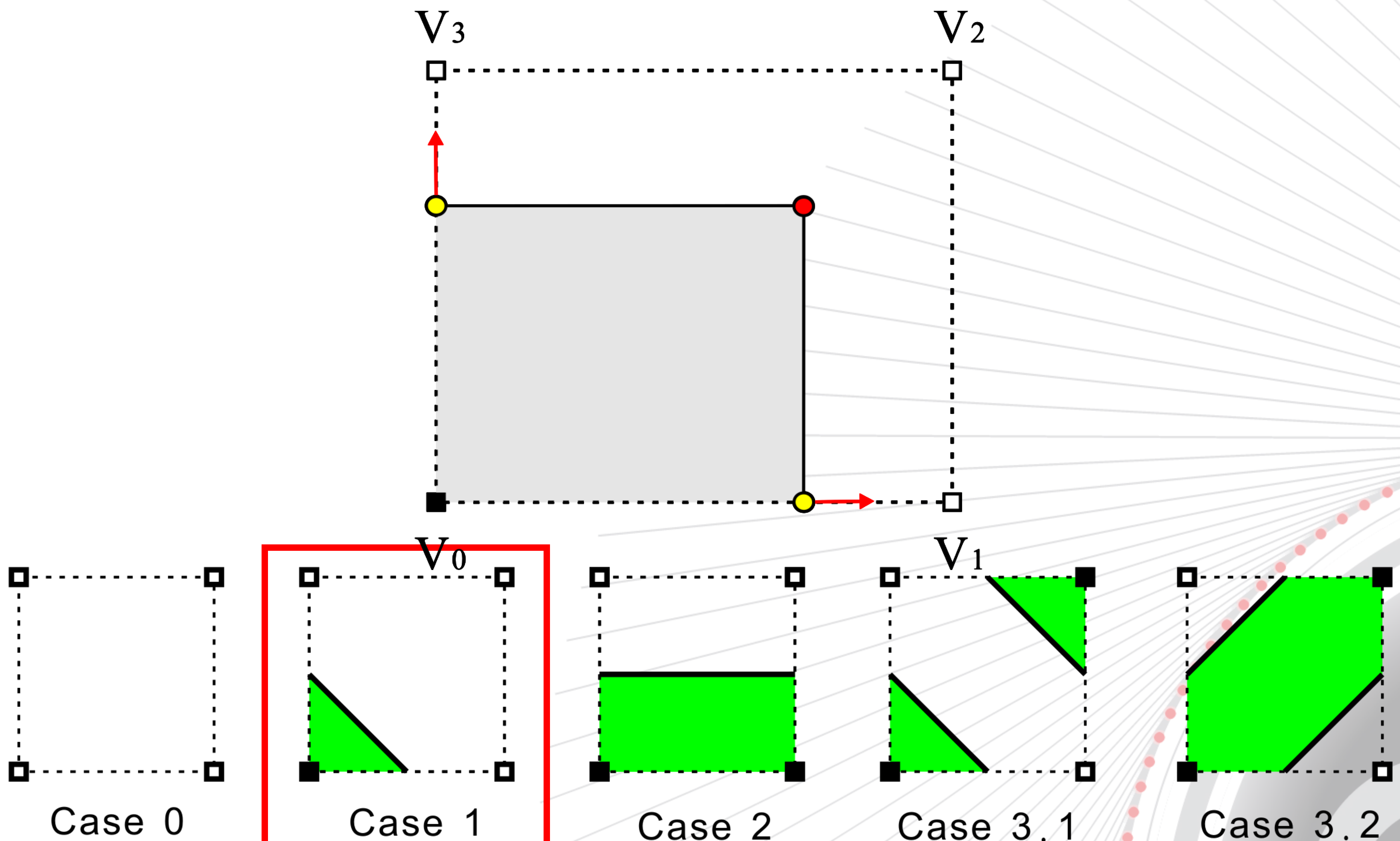


CMS Algorithm

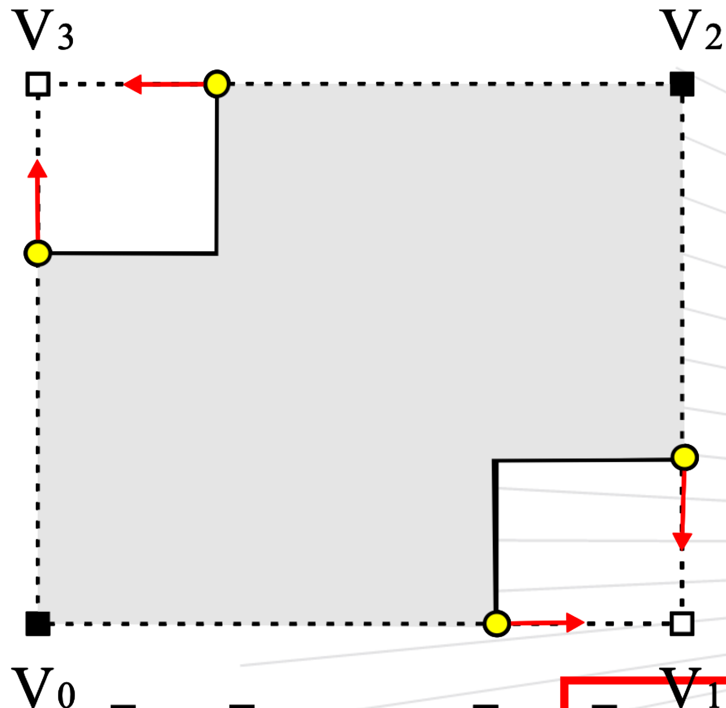
```
procedure CubicalMarchingSquares(HermiteData H)
    InitializeBaseGrid(B);                    ▷ initialize a coarse base grid B
    for each cell c in B
        SubdivideCell(H, c);
    end for
    for each leaf face f
        GenerateSegment(f);
    end for
    for each leaf cell c
        ExtractSurface(c);
    end for
end procedure
```

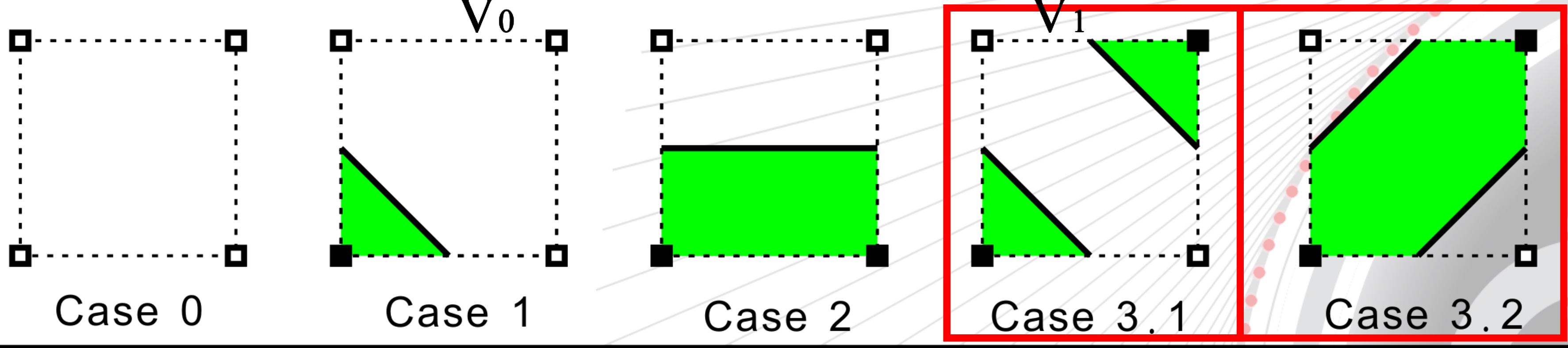


Segment generation on faces for normal cases



Segment generation on faces for ambiguity cases

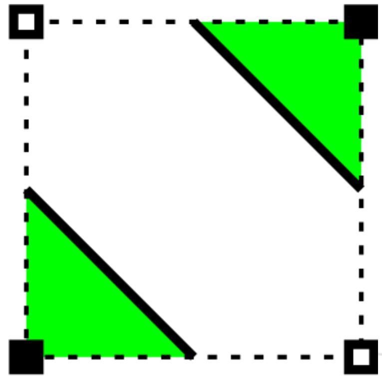




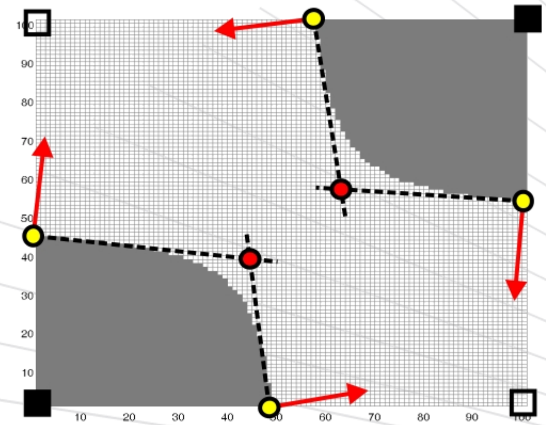
Analysis of face ambiguity

Consistent Topology

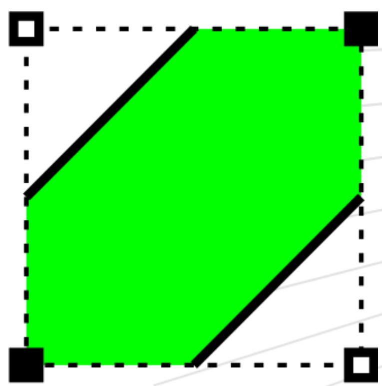
- Separated



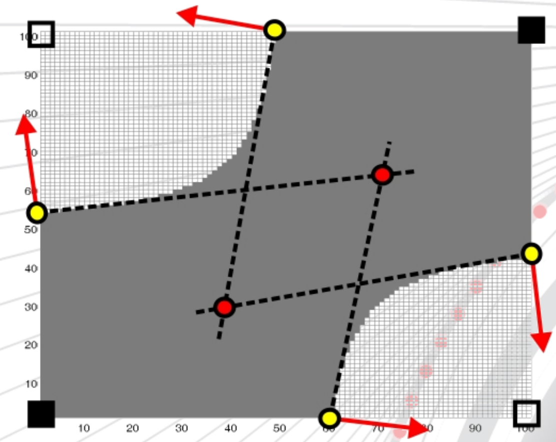
Case 3.1



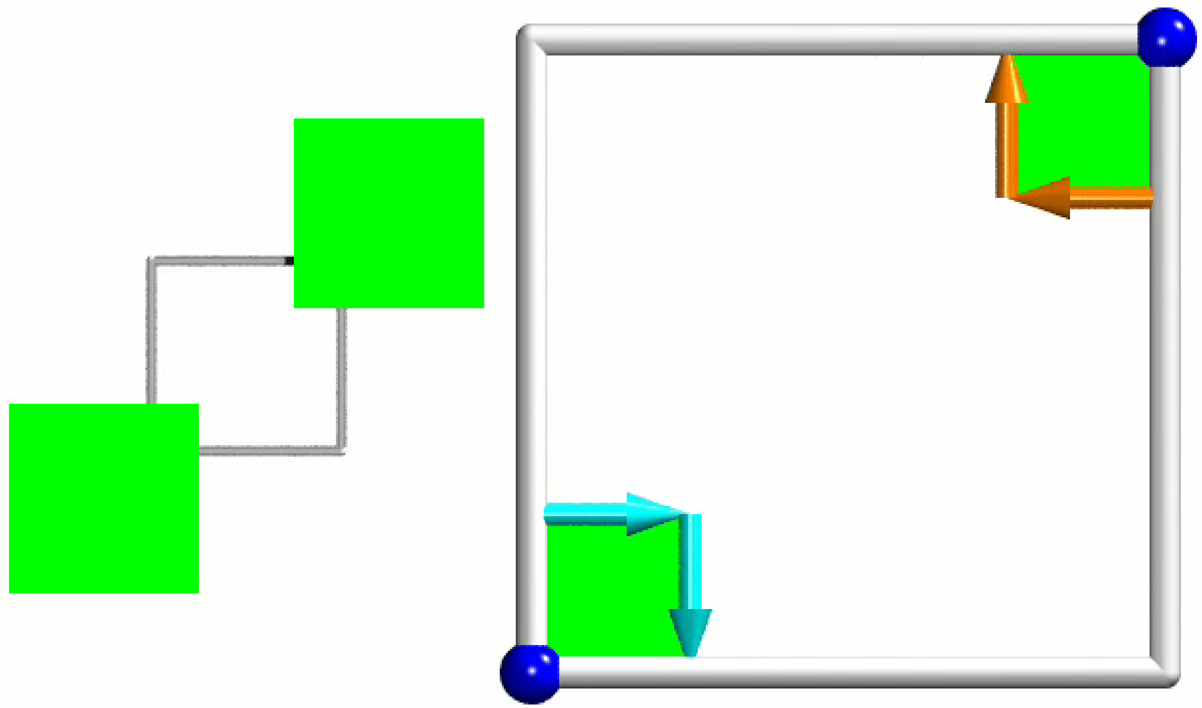
- Joined



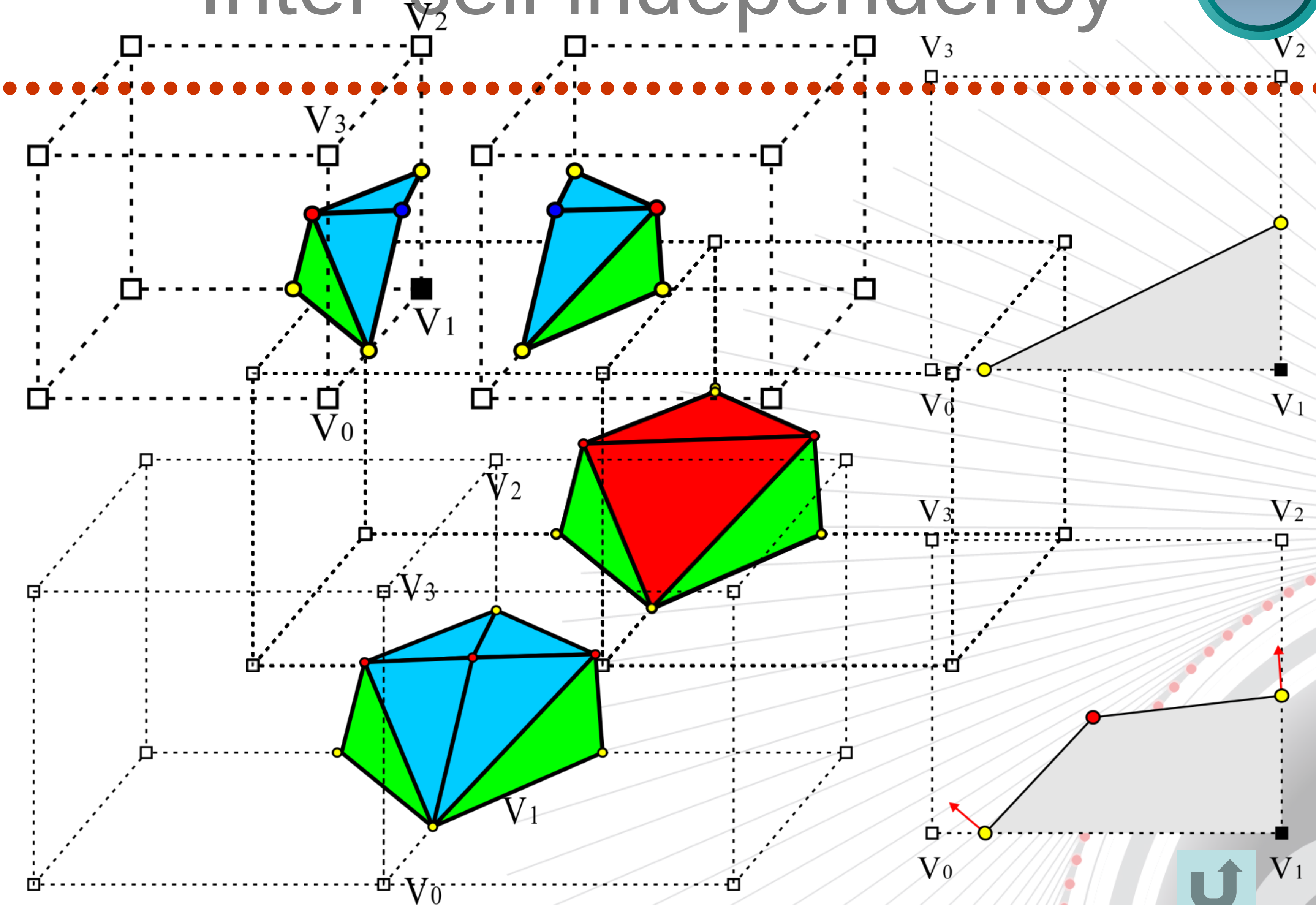
Case 3.2



Segment generation in motion

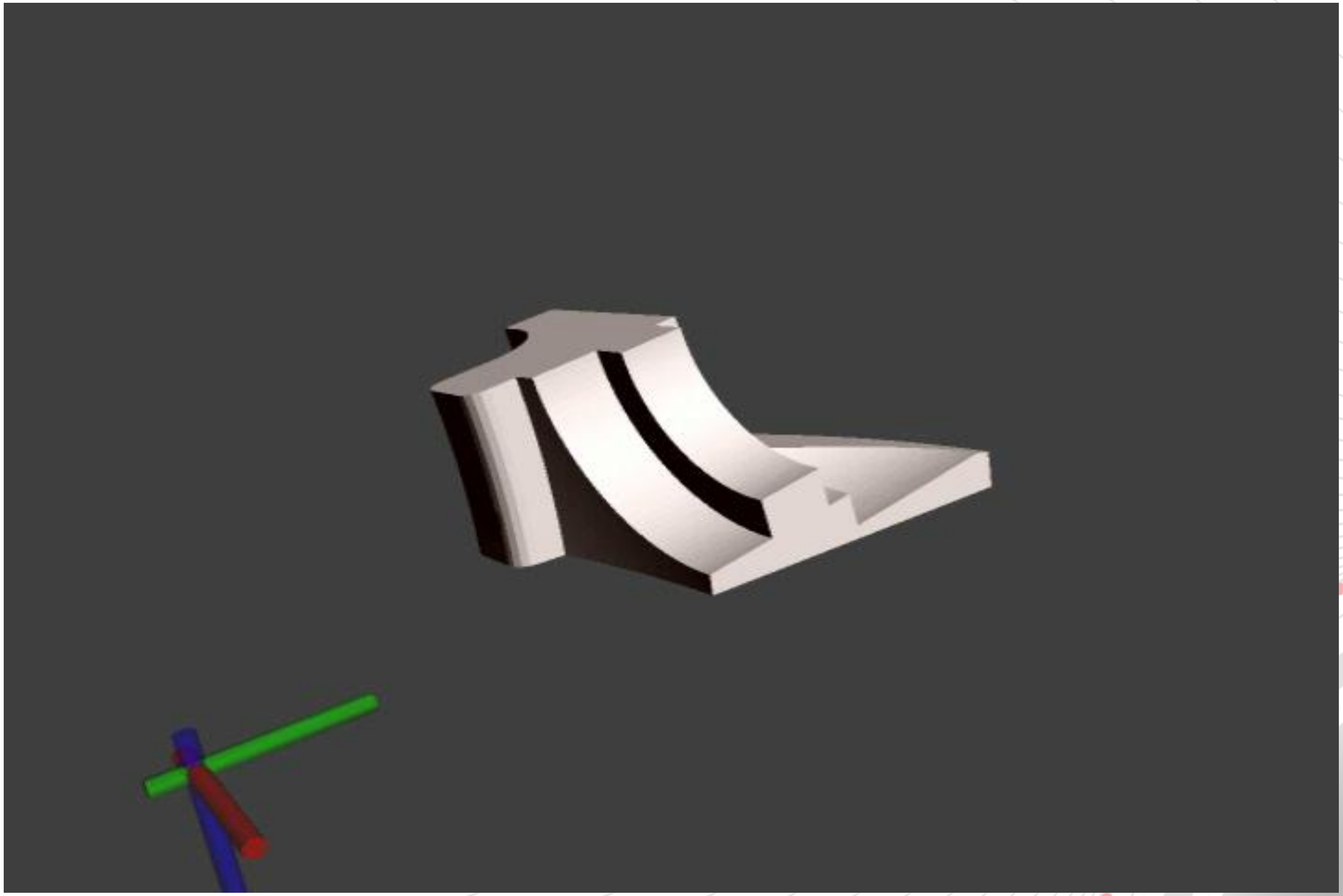


Inter-cell independency



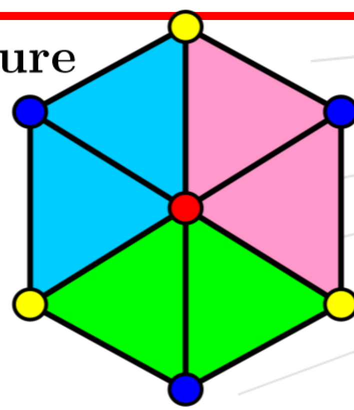
Benefits - inter-cell independency

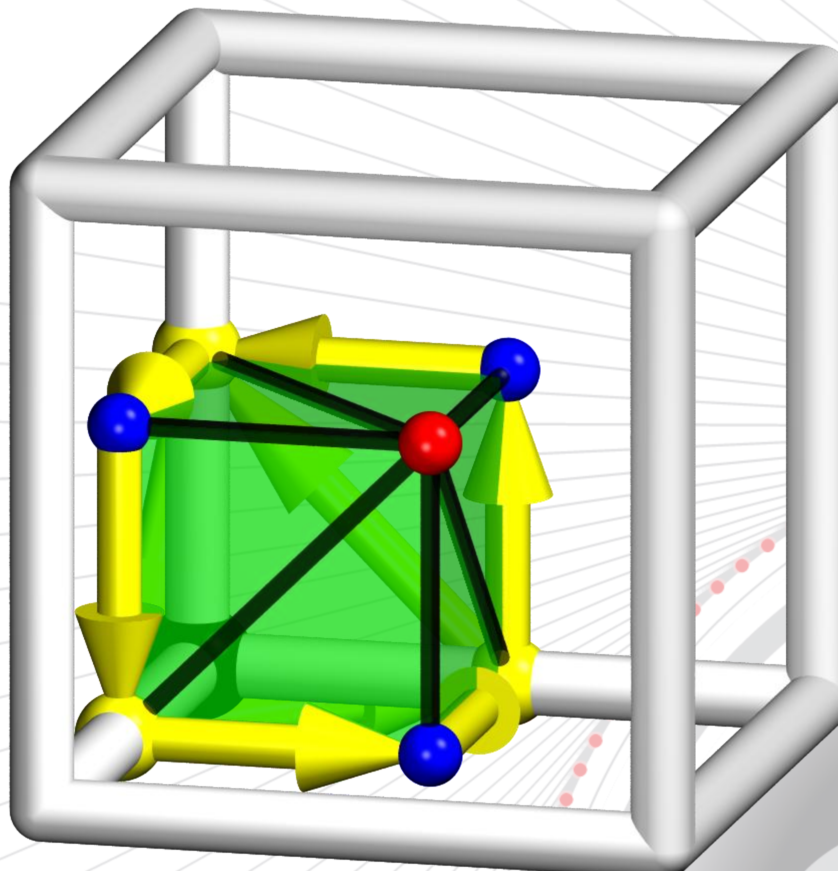
1.Faster 2.Parallelizable 3.Lower Error



Algorithm – CubicalMarchingSquares

```
procedure CubicalMarchingSquares(HermiteData H)
    InitializeBaseGrid(B);                      ▷ initialize a coarse base grid B
    for each cell c in B
        SubdivideCell(H, c);
    end for
    for each leaf face f
        GenerateSegment(f);
    end for
    for each leaf cell c
        ExtractSurface(c);
    end for
end procedure
```

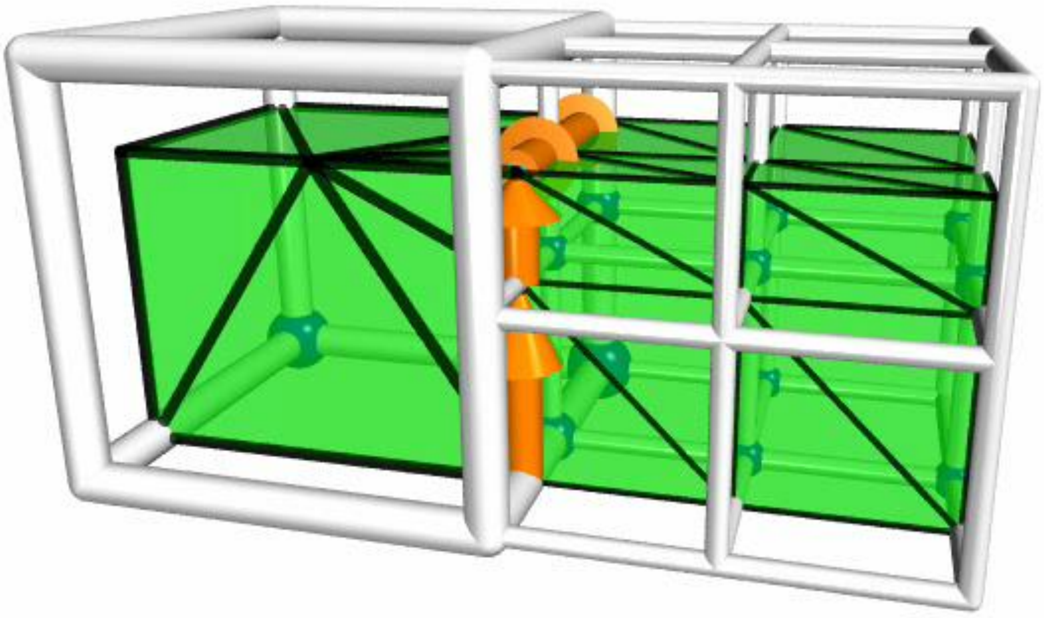


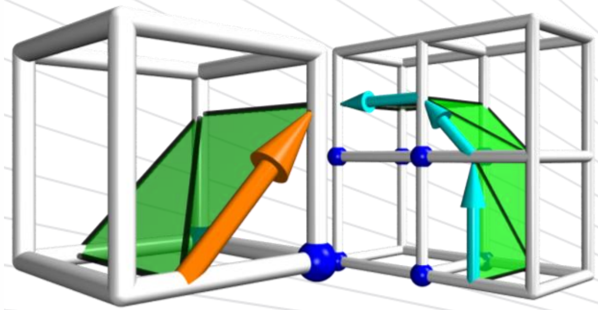


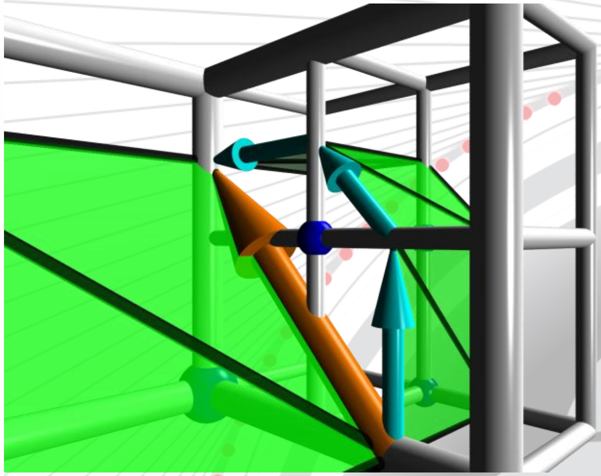
CMS is crack free

Adaptive Resolution

- Cracks are avoided using CMS

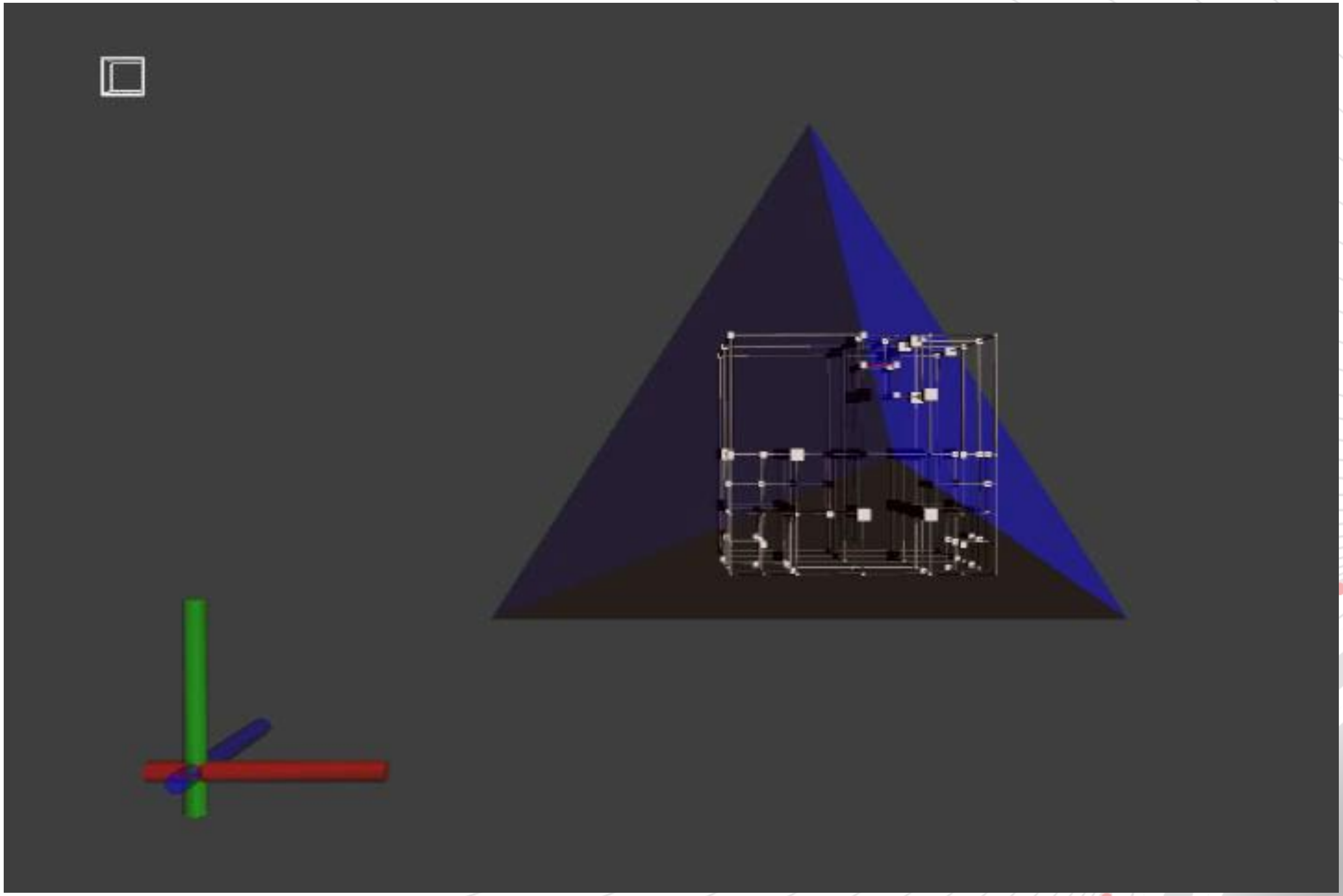






Benefits – adaptive & crack free

1. Smooth Shape 2. Reduce 3D ➔ 2D



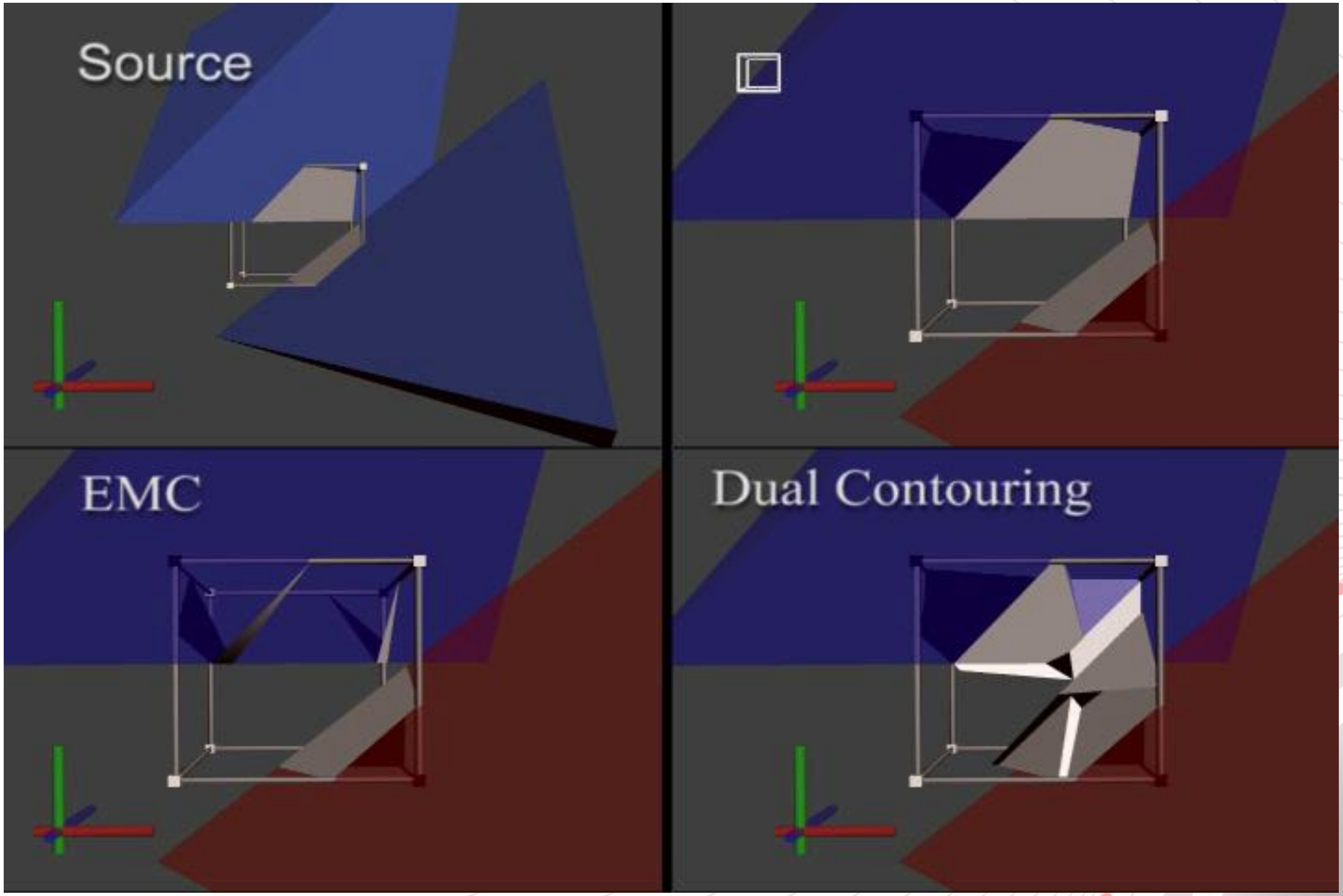
Previous work & Problems

Solutions

Results & Conclusion

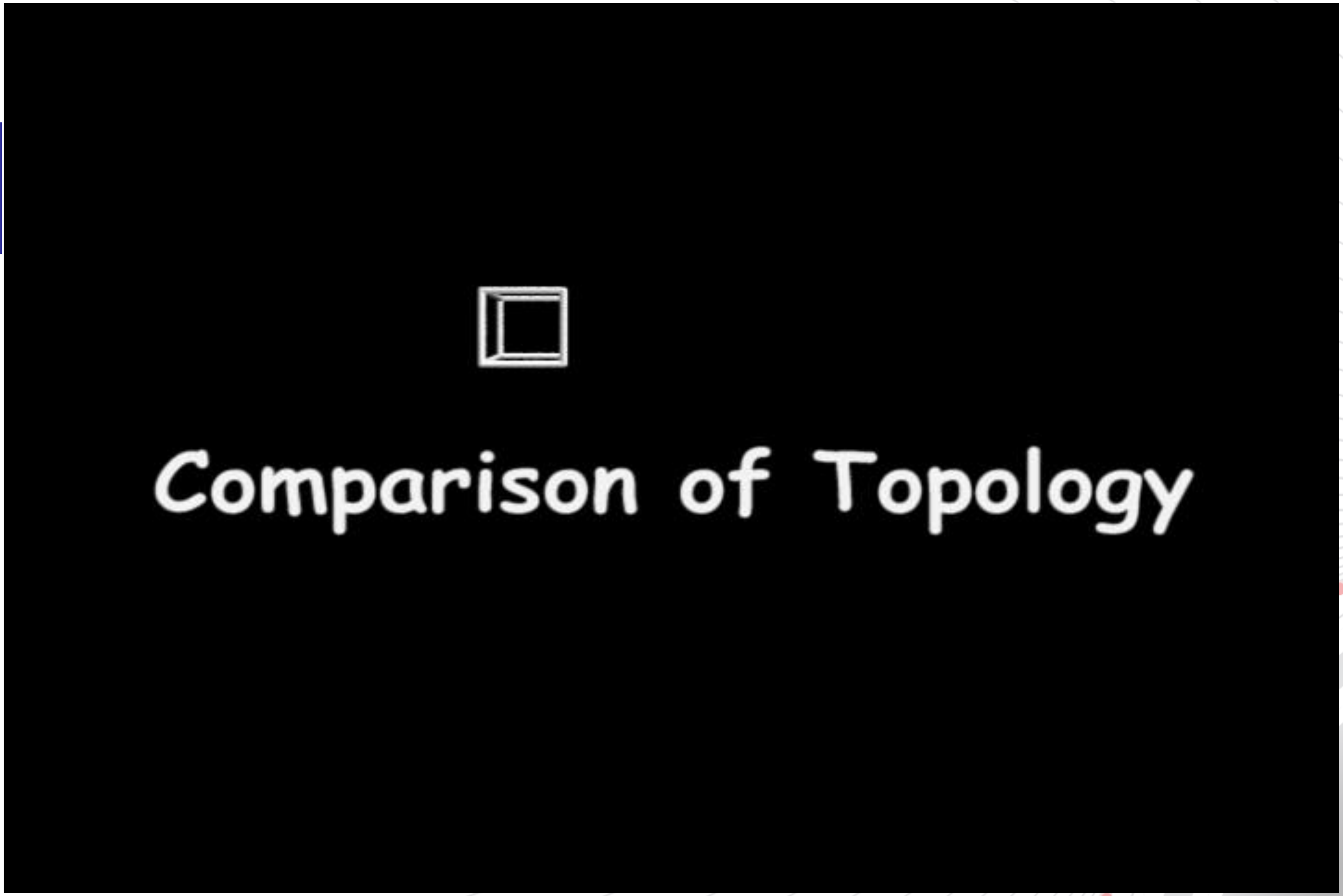
Benefits - consistent topology

1. Correct Shape 2. Lower Error



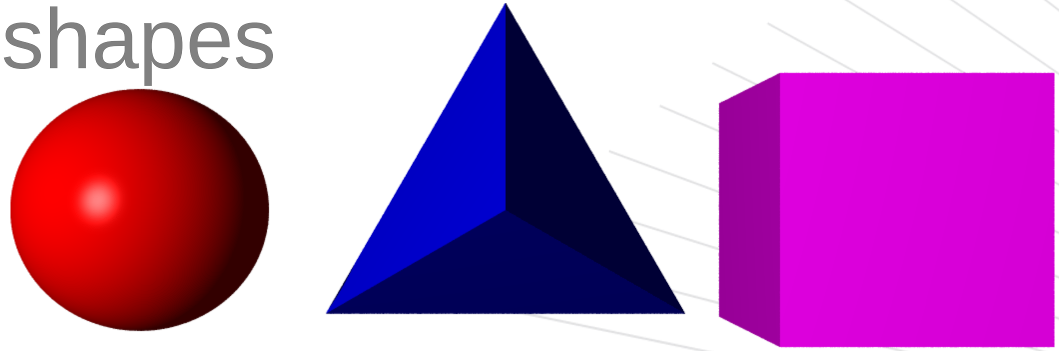
Benefits - consistent topology

1. Correct Shape 2. Lower Error

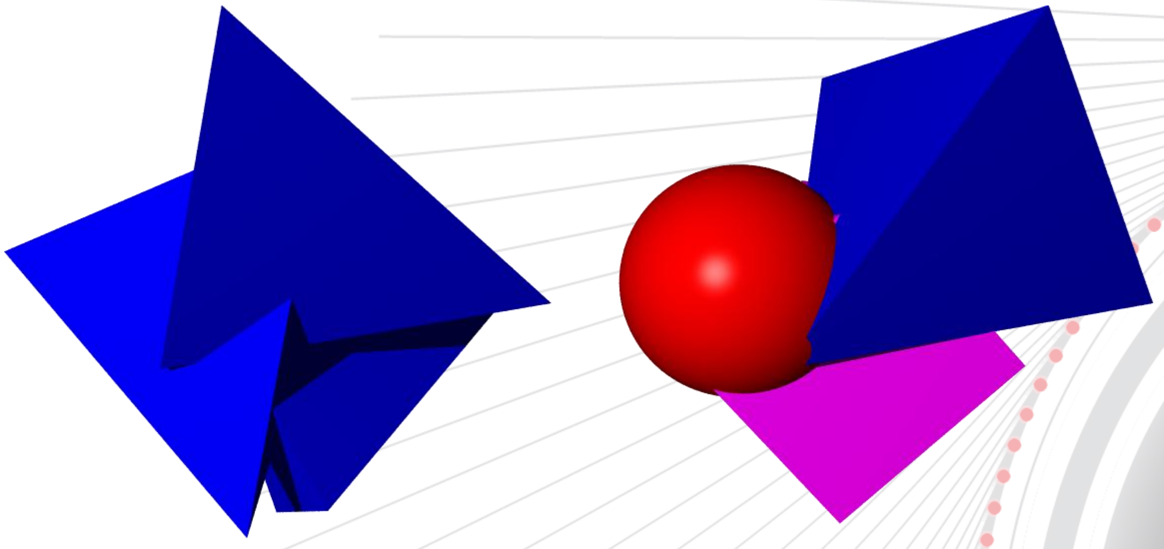


Simulation

- Available shapes

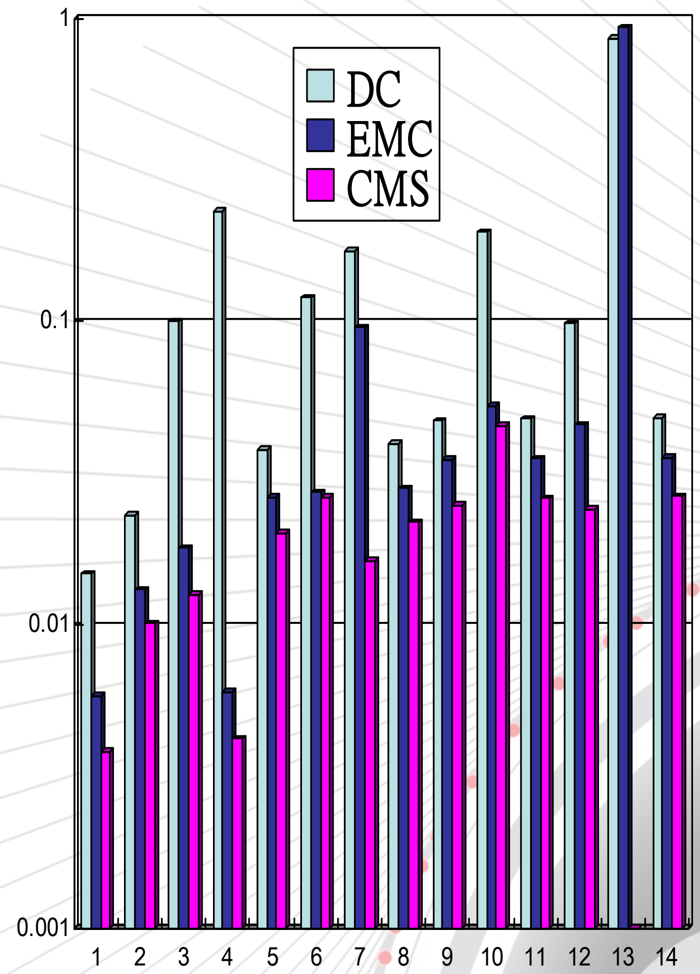


- generated randomly in a limited space

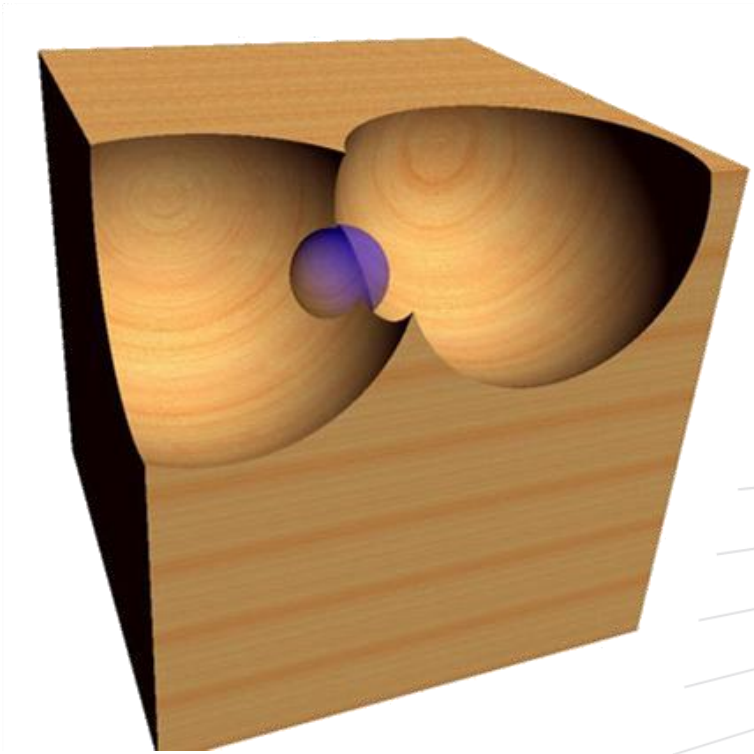


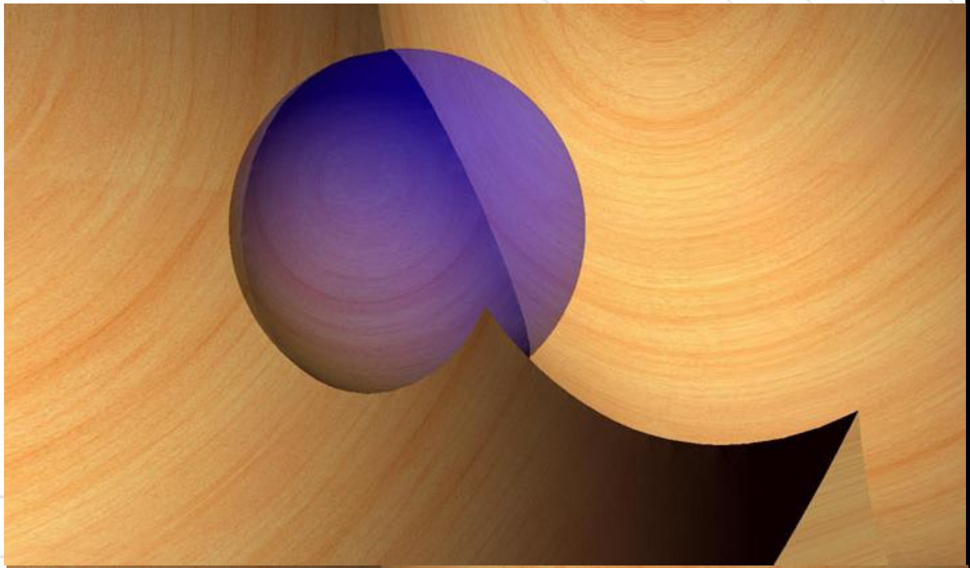
Average geometric errors

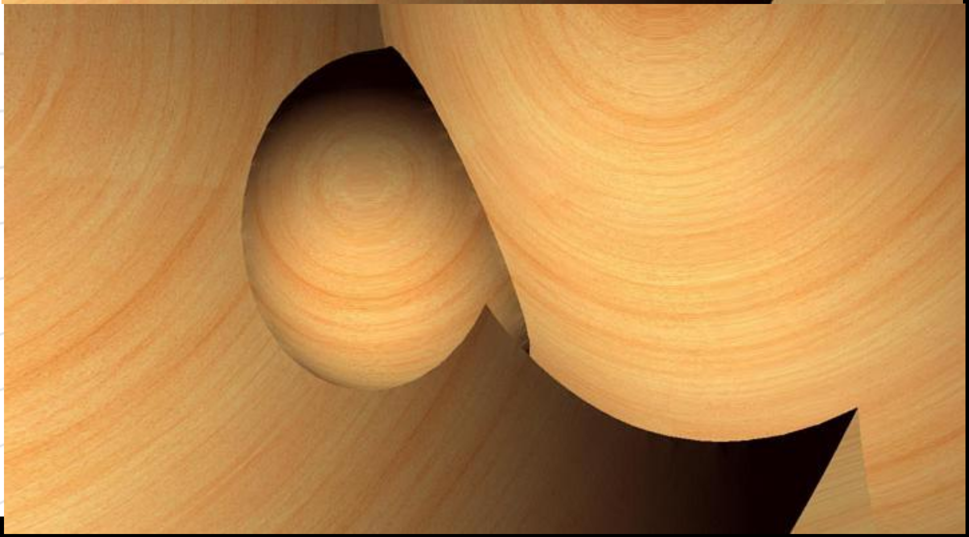
case	times	DC	EMC	CMS
1	3,590,980	0.01473	0.00586	0.00383
2	1,554,028	0.02309	0.01310	0.01013
3	207,302	0.10027	0.01801	0.01263
4	30,972	0.23064	0.00601	0.00422
5	803,311	0.03779	0.02631	0.02011
6	101,875	0.11998	0.02737	0.02633
7	12,198	0.17139	0.09565	0.01628
8	109,141	0.03979	0.02831	0.02184
9	72,201	0.04721	0.03525	0.02492
10	4,237	0.19682	0.05283	0.04541
11	70,238	0.04789	0.03535	0.02620
12	30,706	0.09845	0.04559	0.02419
13	1,405	0.85461	0.92935	0.00100
14	70,238	0.04821	0.03573	0.02653



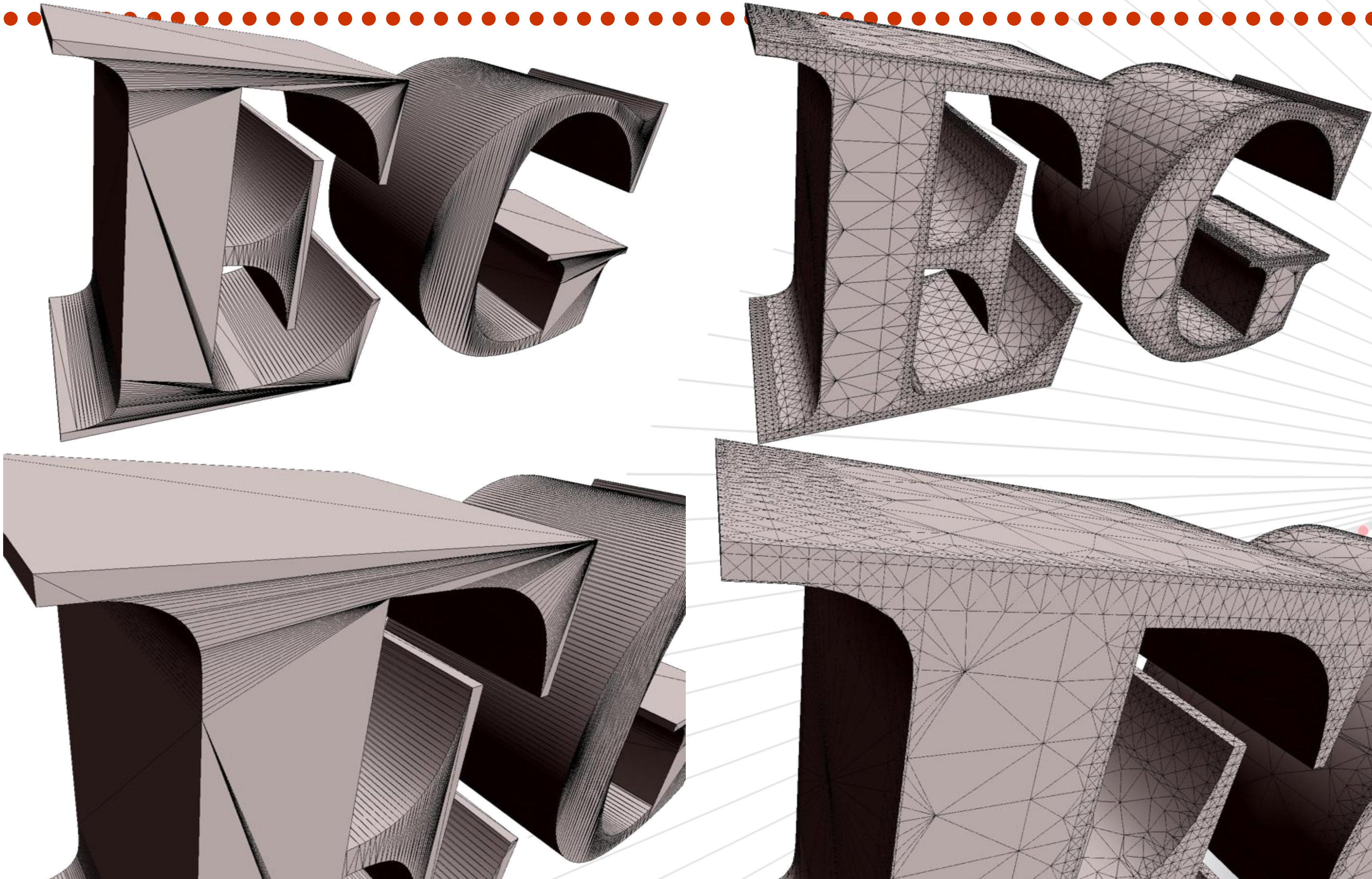
Virtual Sculptor



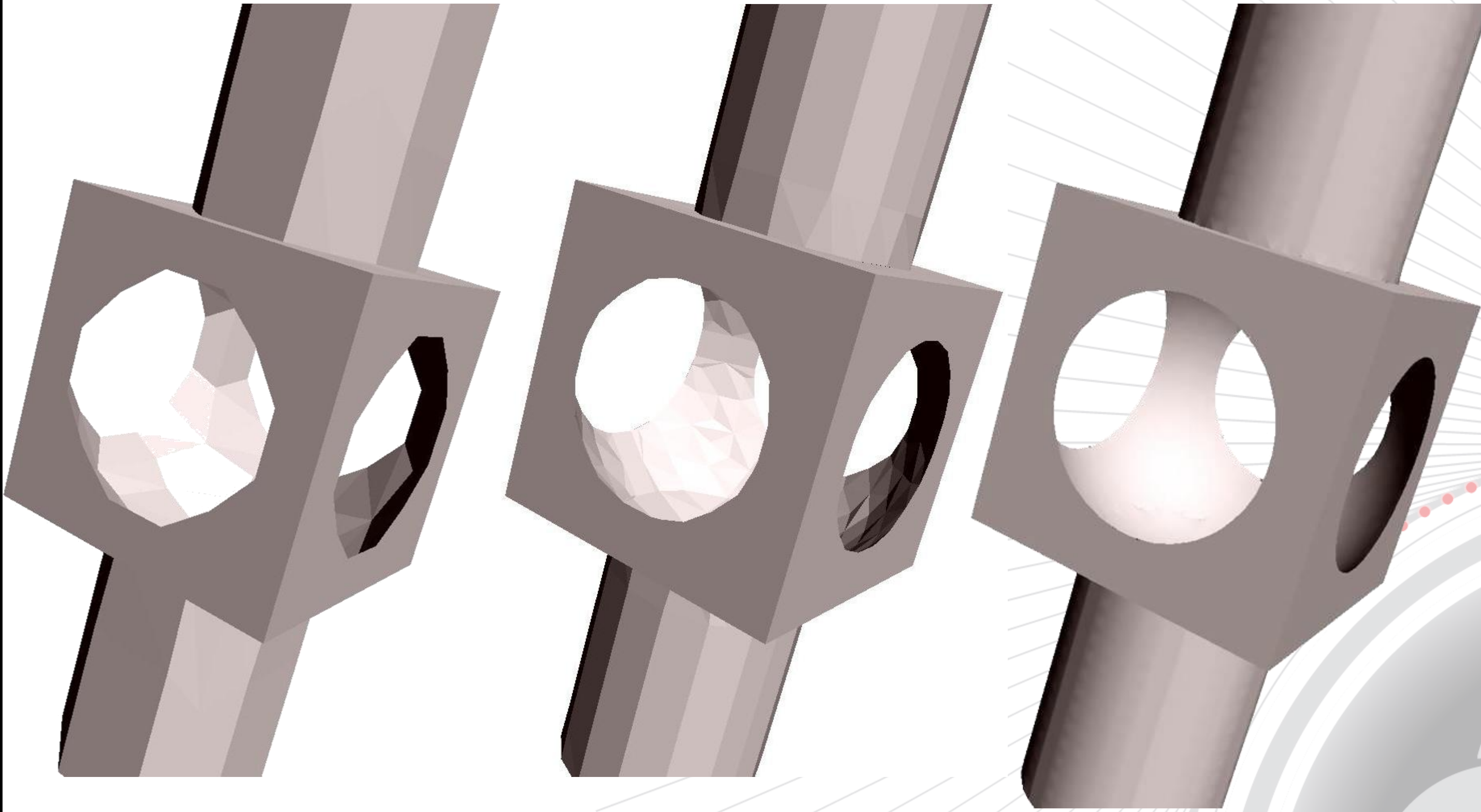




Remeshing



CSG & LOD

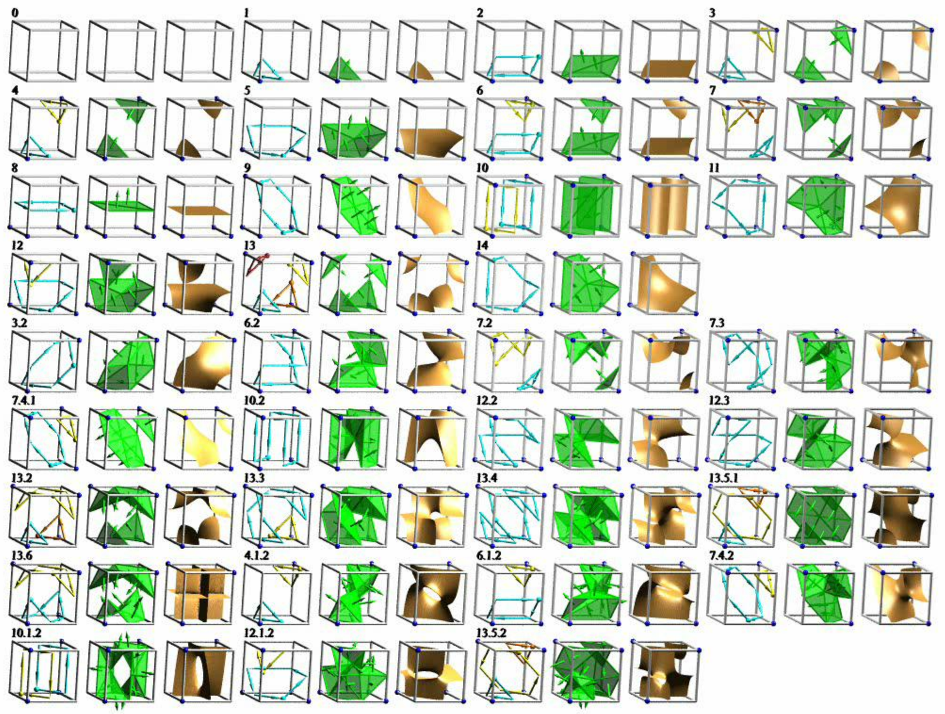


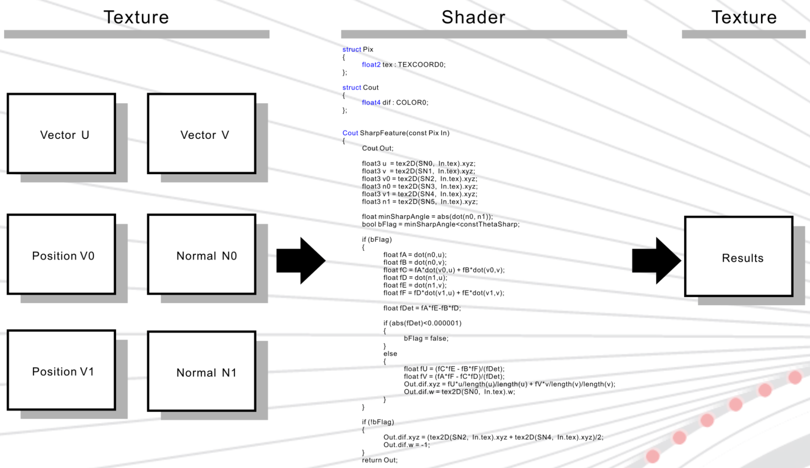
Conclusion

	Parallel Processing	Sharp Features	Consistent Topology	Adaptive Resolution
Marching Cubes	★			
Topological Marching Cubes	★		★	
Extended Marching Cubes		★		
Dual Contouring		★		★
Cubical Marching Squares	★	★	★	★

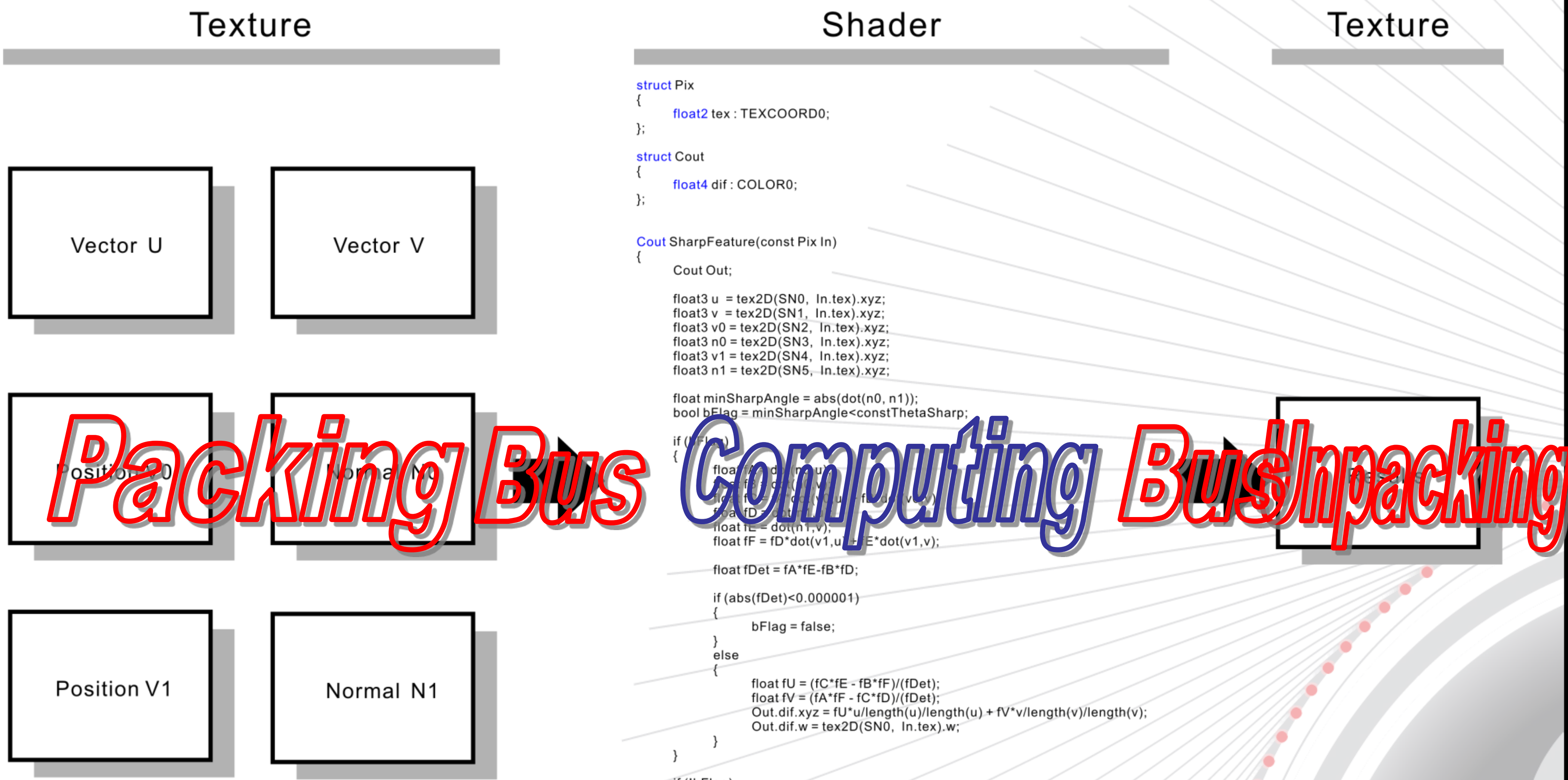
Future work

- Medical imaging by CMS
- Acceleration using GPU





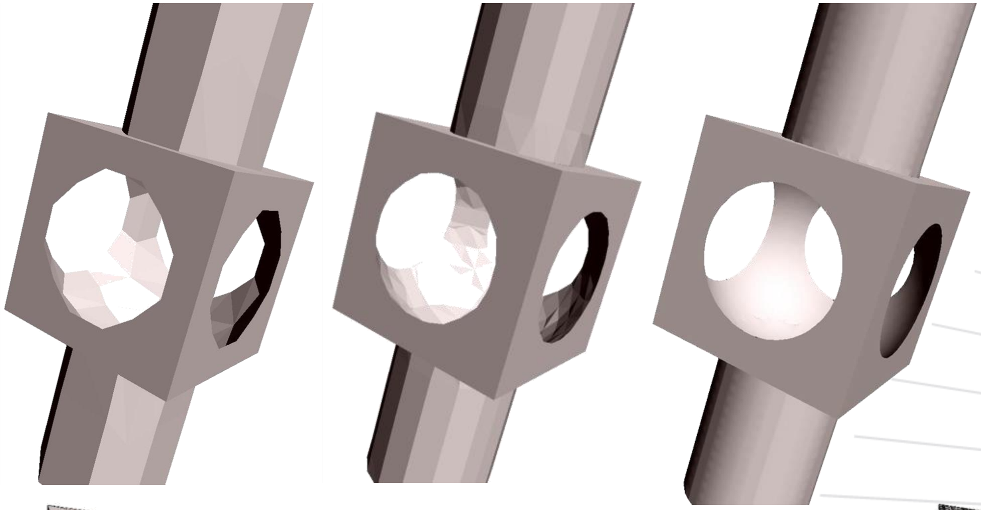
Preliminary result: Acceleration using GPU

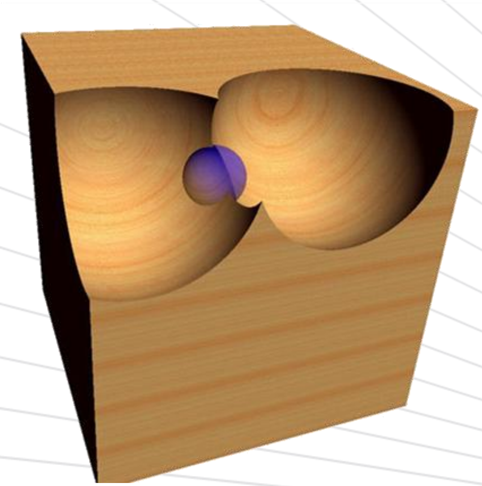


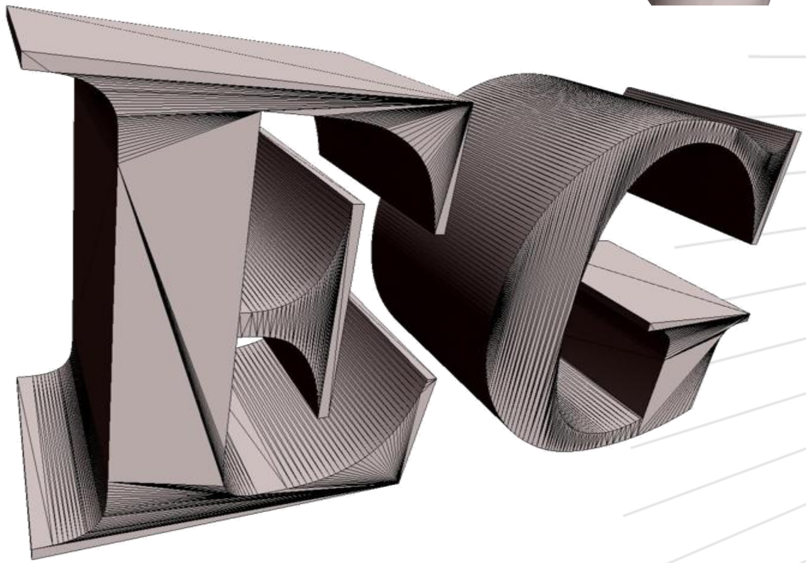

```
struct Pix
{
	float2 tex : TEXCOORD0;
};

struct Cout
{
	float4 dif : COLOR0;
};


Cout SharpFeature(const Pix In)
{
	Cout Out;

	float3 u  = tex2D(SN0,  In.tex).xyz;
	float3 v  = tex2D(SN1,  In.tex).xyz;
	float3 v0 = tex2D(SN2,  In.tex).xyz;
	float3 n0 = tex2D(SN3,  In.tex).xyz;
	float3 v1 = tex2D(SN4,  In.tex).xyz;
	float3 n1 = tex2D(SN5,  In.tex).xyz;

	float minSharpAngle = abs(dot(n0, n1));
	bool bFlag = minSharpAngle<constThetaSharp;

	if ([illegible])
	{
		float fA = d[illegible]u);
		[illegible] fB = [illegible]v);
		[illegible] fC = [illegible]u) + fD[illegible]v);
		[illegible] fD = [illegible]
		float fE = dot(n1,v);
		float fF = fD*dot(v1,u) + fE*dot(v1,v);

		float fDet = fA*fE-fB*fD;

		if (abs(fDet)<0.000001)
		{
			bFlag = false;
		}
		else
		{
			float fU = (fC*fE - fB*fF)/(fDet);
			float fV = (fA*fF - fC*fD)/(fDet);
			Out.dif.xyz = fU*u/length(u)/length(u) + fV*v/length(v)/length(v);
			Out.dif.w = tex2D(SN0,  In.tex).w;
		}
	}

	if (!bFlag)
	{
		Out.dif.xyz = (tex2D(SN2,  In.tex).xyz + tex2D(SN4,  In.tex).xyz)/2;
		Out.dif.w = -1;
	}
	return Out;
```

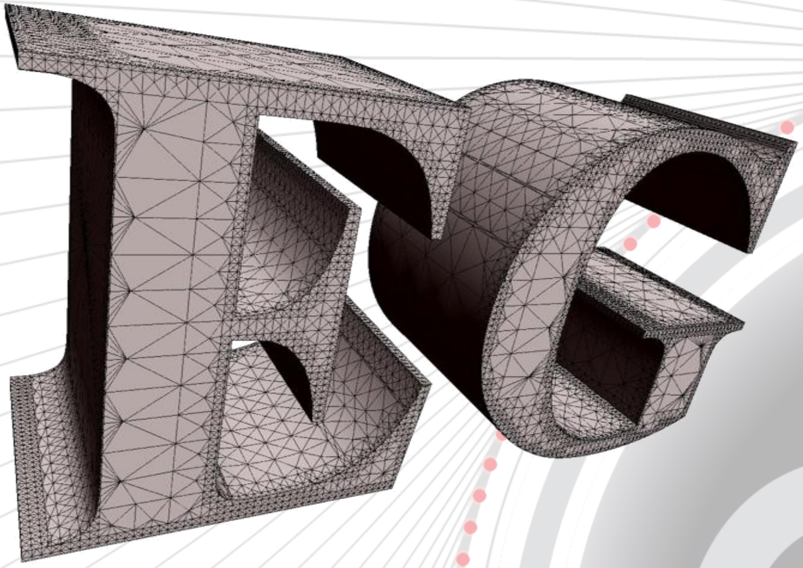
Thanks

Applications

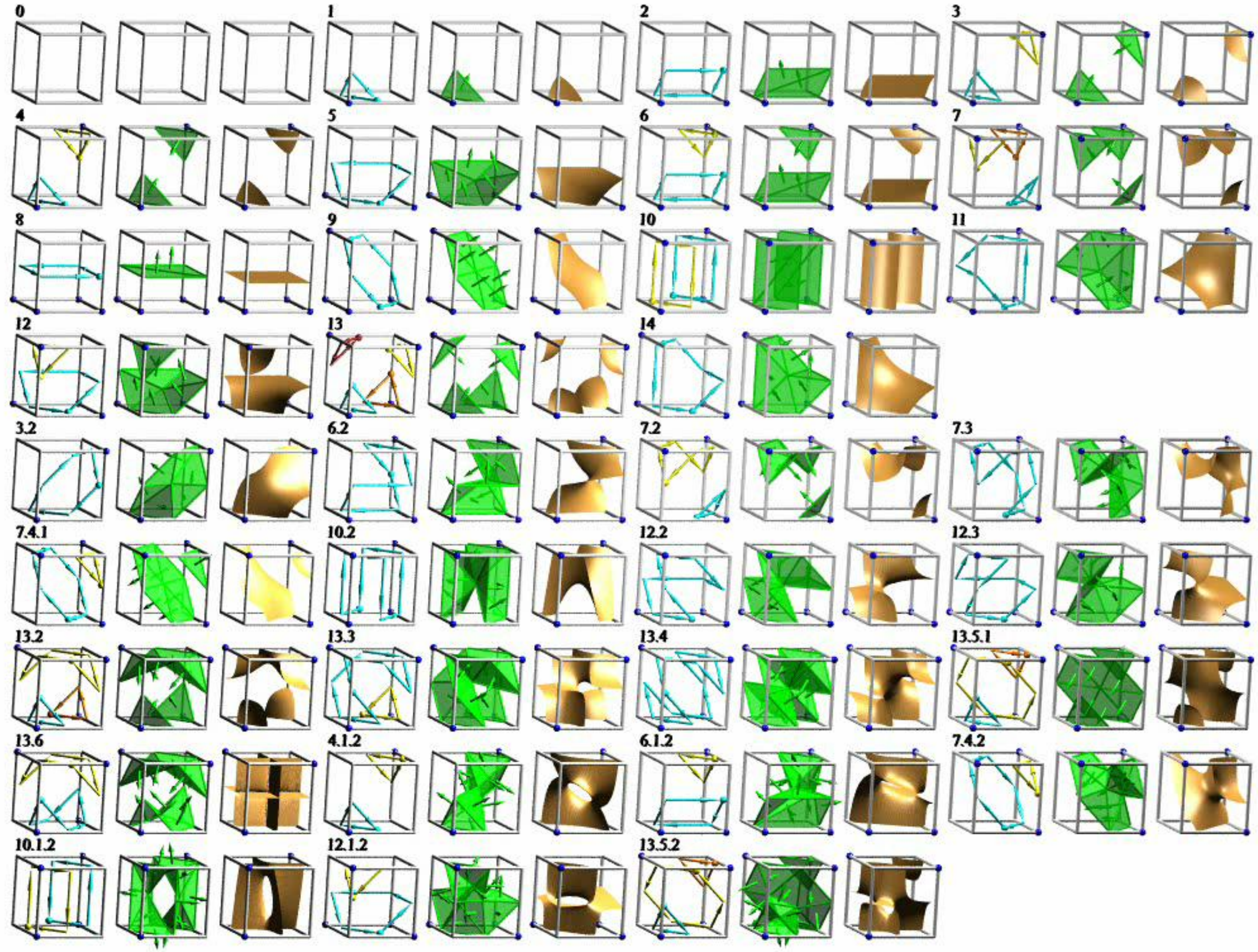




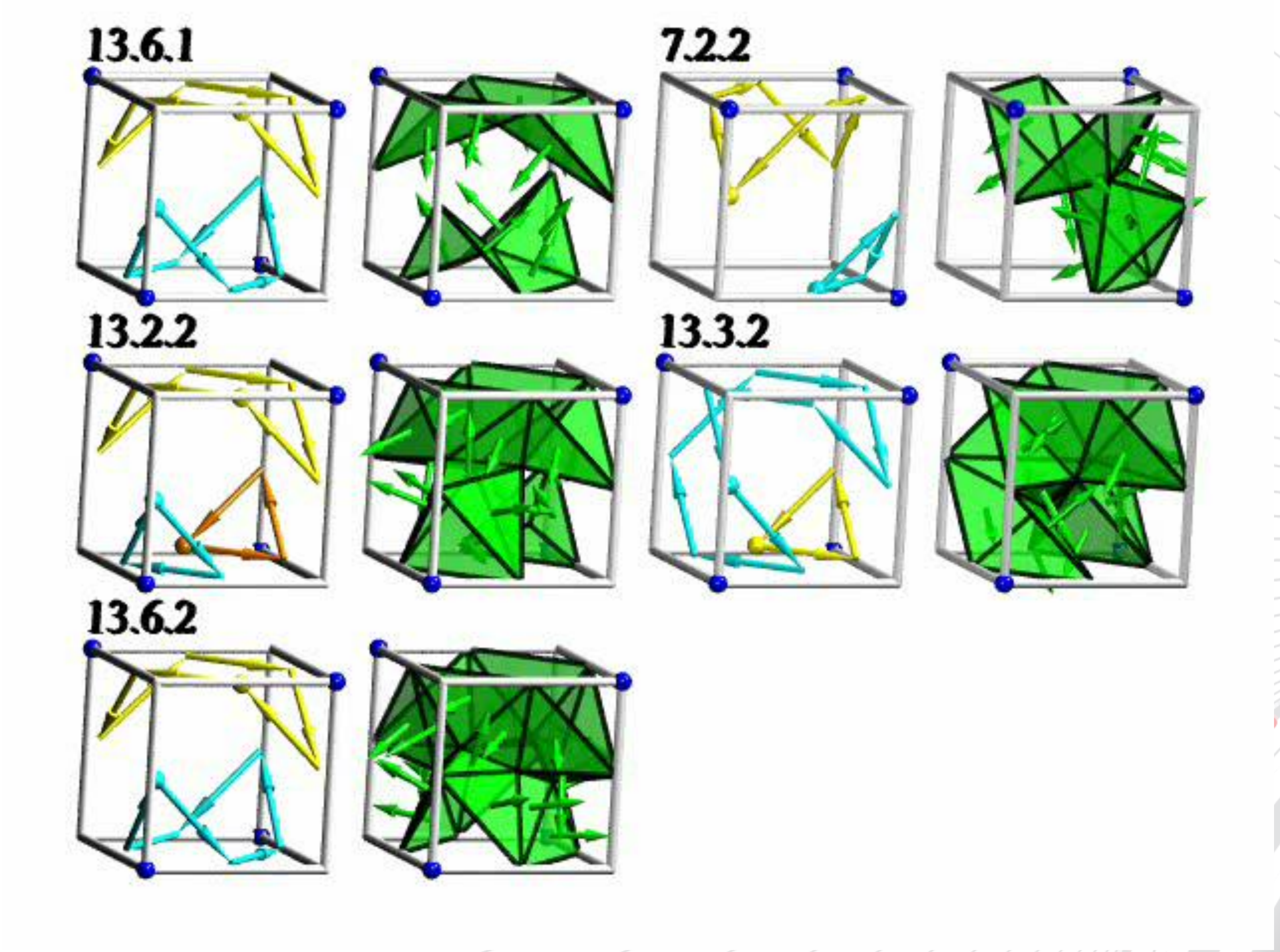




Cube based patterns by CMS



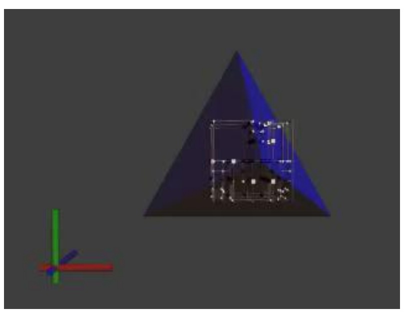
New patterns

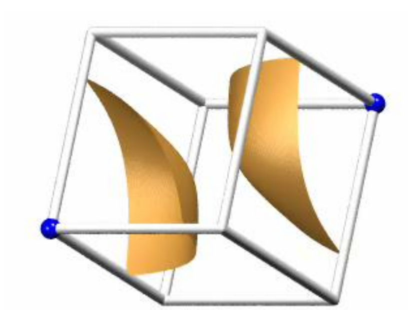


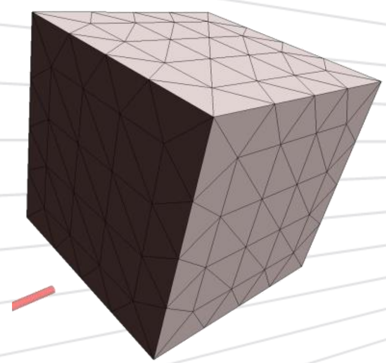
- Triangle count
- Memory space requirement
- Computation

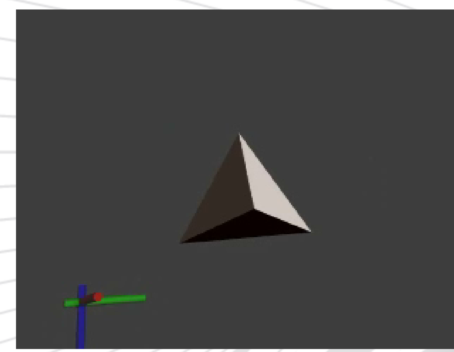
Overview: Problems & Goals

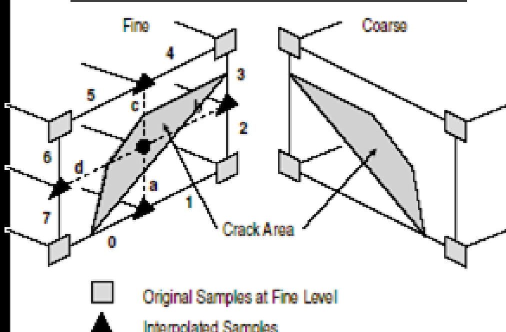
- Adaptive Resolution
- Consistent Topology
- Sharp Features
- Parallel Processing

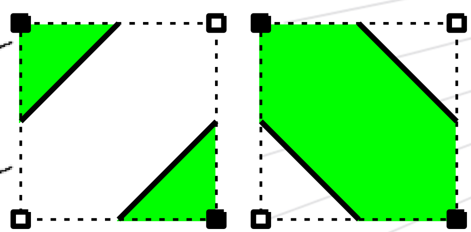


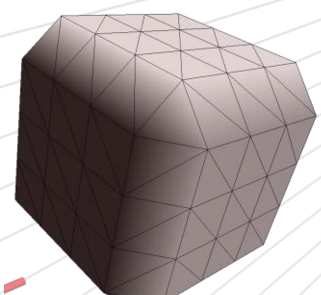


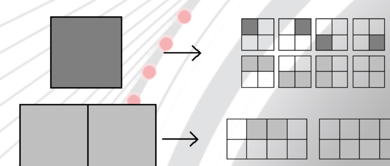




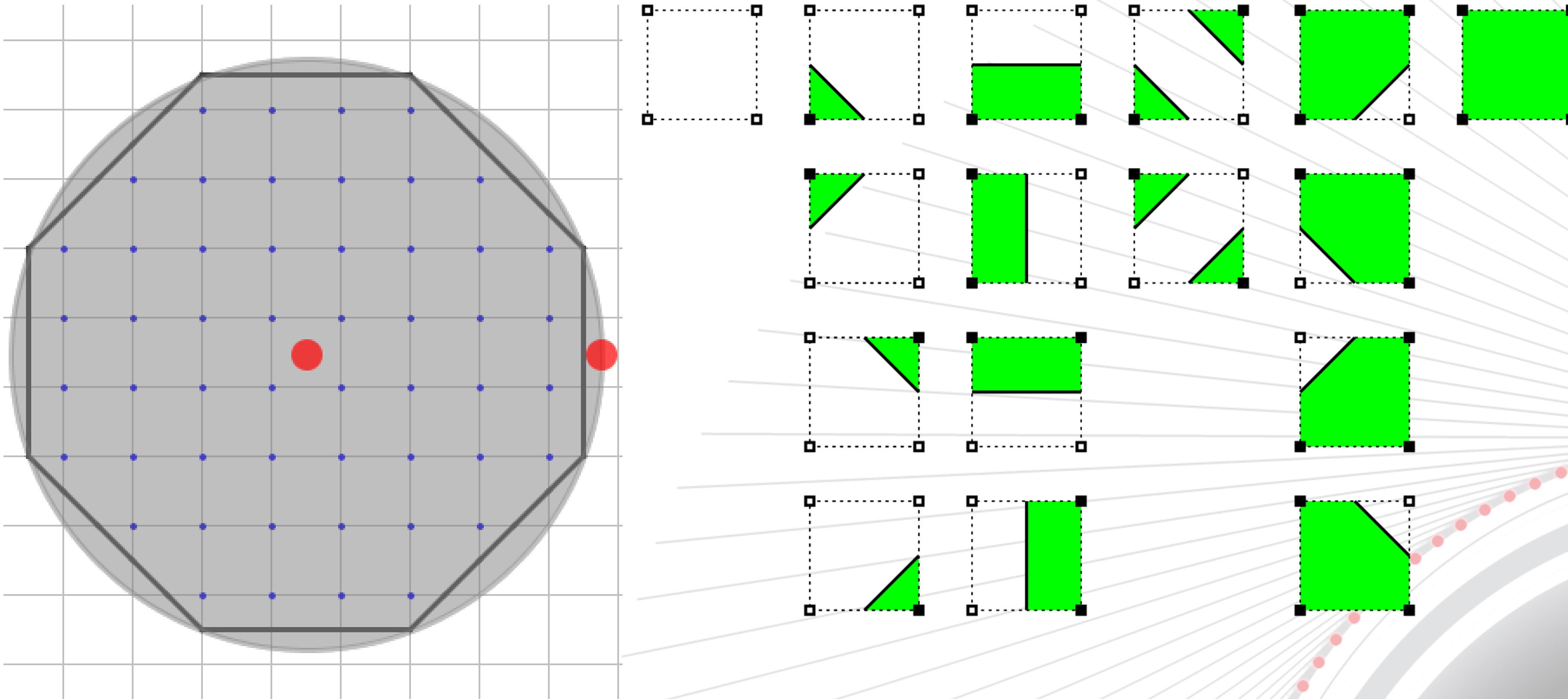






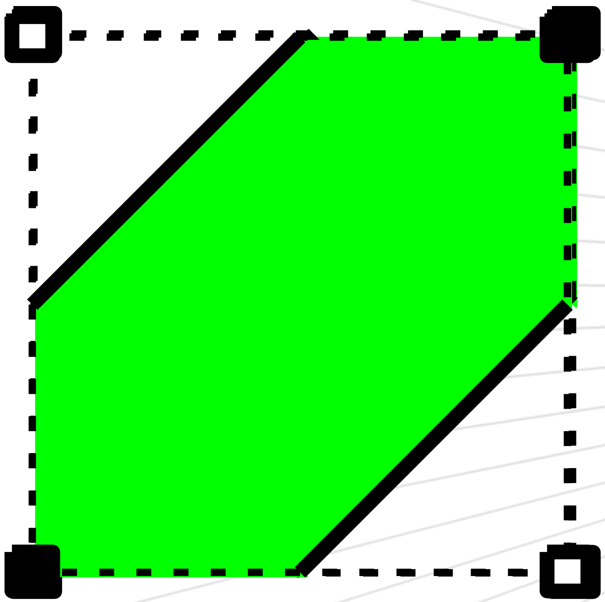


Marching squares (2D)…



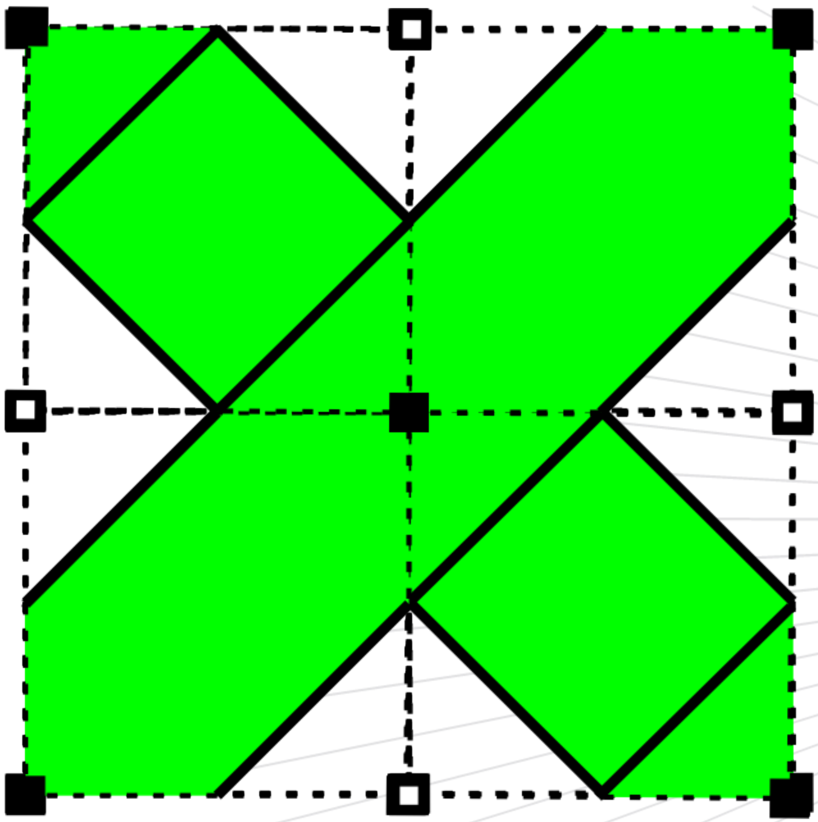
Ambiguity problem

- Face ambiguty
  - [NIELSON G. M., HAMANN B. 1991], etc.



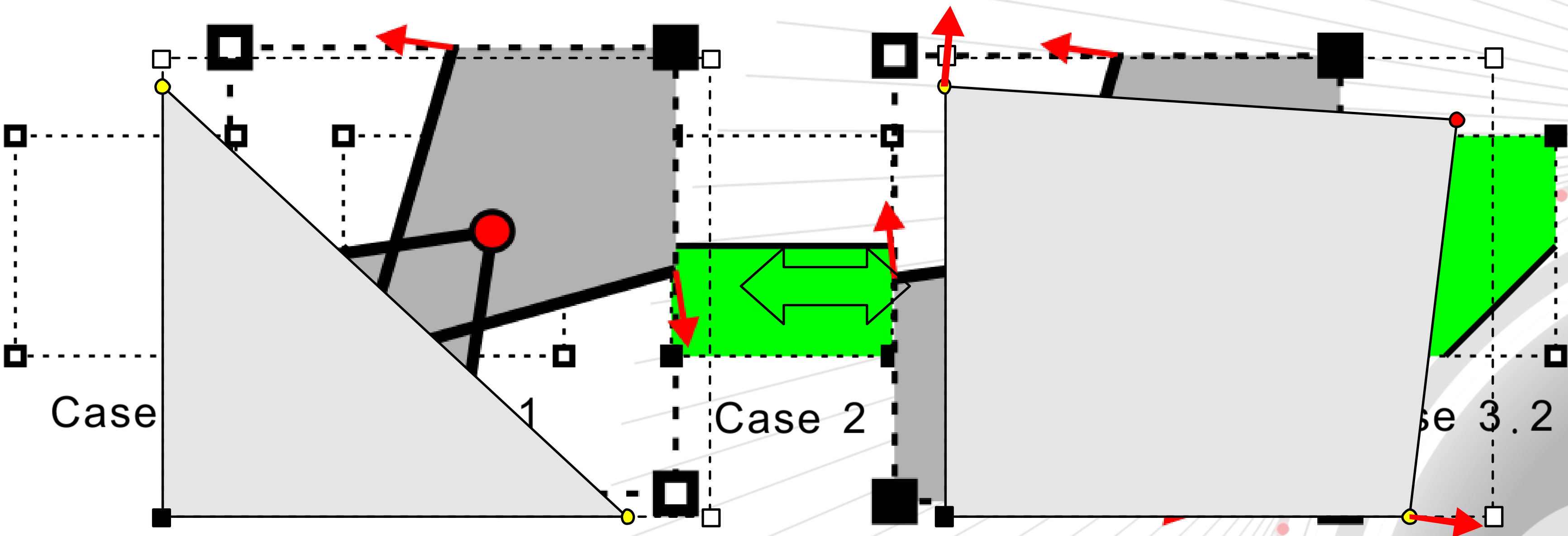
Ambiguity problem…

Consistent Topology



Segment generation on faces

- Table look up
- Detect sharp features
- Resolve topological ambiguity



Resolving face ambiguty

Consistent Topology