3D Mesh Compression in Open3DGC

Khaled MAMMOU

OPPORTUNITIES FOR COMPRESSION

- Indexed Face Set
  - Geometry: positions
  - Connectivity: list of triangles

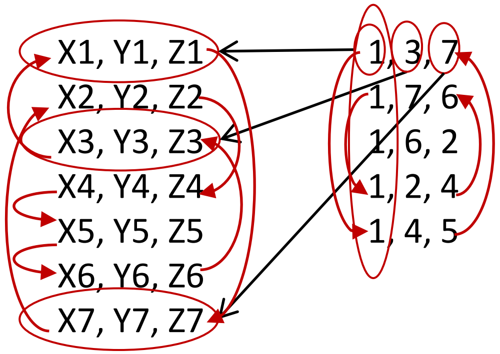
  Requires 192 bits per vertex!

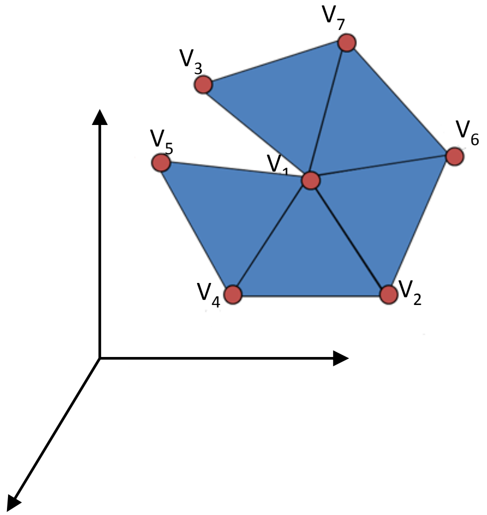
- Redundancy
  - Indexes repeated multiple times
  - No need to preserve triangles and vertices order
  - No need for 32-bit precision for positions/attributes
  - Neighbour vertices exhibit high geometry correlations

⬇

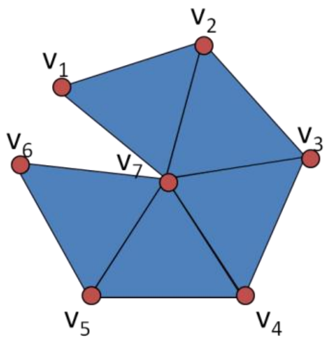
Lossy geometry compression and lossless connectivity encoding reduce the stream size to **10-20 bpv**

Geometry Connectivity



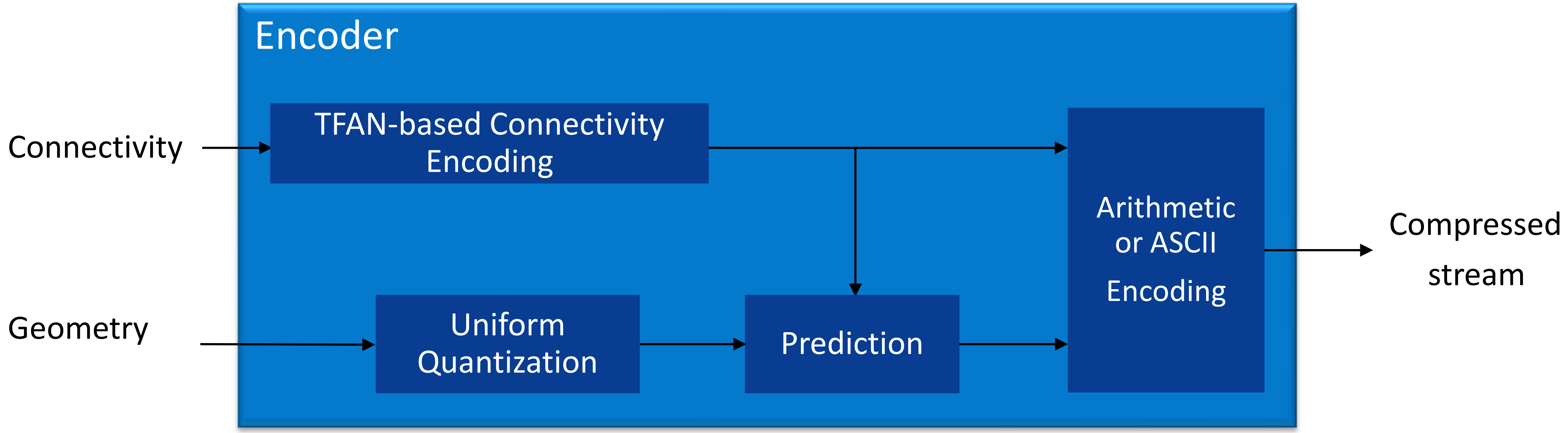


Reordered geometry	Reordered Connectivity
X3, Y3, Z3	7, 5, 6
X7, Y7, Z7	7, 4, 5
X6, Y6, Z6	7, 3, 4
X2, Y2, Z2	7, 1, 2
X4, Y4, Z4	7, 2, 3
X5, Y5, Z5	
X1, Y1, Z1	



ENCODER OVERVIEW

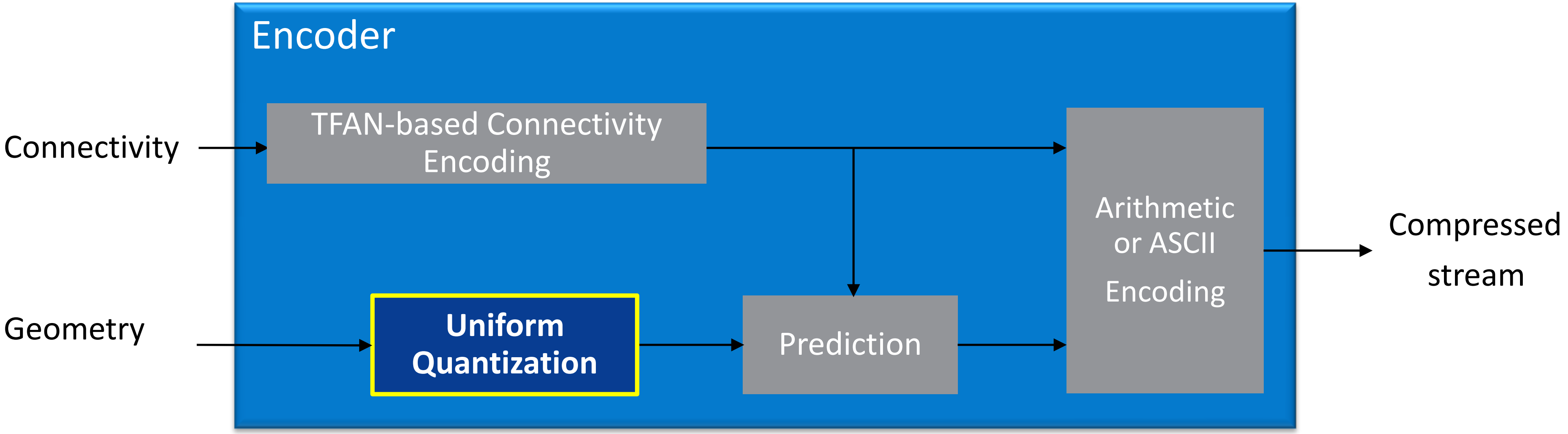
- Algorithm based on TFAN codec [Mammou'09]
  - Triangular meshes with attributes
  - Arbitrary topologies (e.g., manifold or not, open/closed, oriented or not, arbitrary genus, holes…)
- MPEG-SC3DMC (Scalable Complexity 3D Mesh Coding) published in 2010



[Mammou'09] K. Mammou, T. Zaharia, F. Prêteux, "TFAN: A low complexity 3D mesh compression algorithm" Computer Animation and Virtual Worlds, Vol. 20(2-3), pp. 343–354, 2009

ENCODER OVERVIEW

- Algorithm based on TFAN codec [Mammou’09]
  - Triangular meshes with attributes
  - Arbitrary topologies (e.g., manifold or not, open, closed, holes…)
- MPEG-SC3DMC (Scalable Complexity 3D Mesh Coding) published in 2010

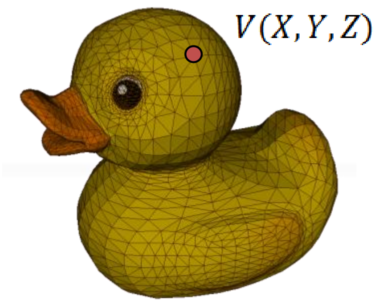


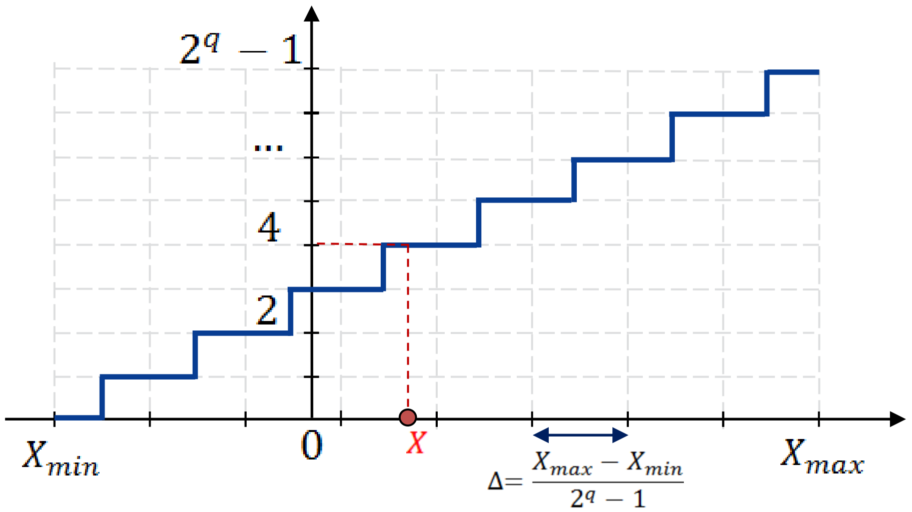
[Mammou’09] K. Mammou, T. Zaharia, F. Prêteux, “TFAN: A low complexity 3D mesh compression algorithm” Computer Animation and Virtual Worlds, Vol. 20(2-3), pp. 343–354, 2009

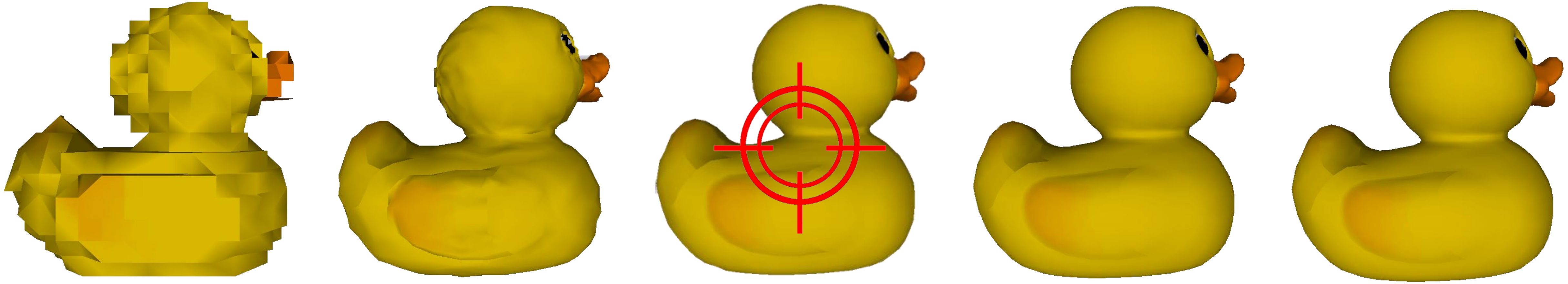
ENCODER OVERVIEW

- Uniform quantization
  - Map real numbers to integers $\{0, 1, \ldots, 2^q - 1\}$
  - Maximum quantization error $\Delta/2$

Uniform quantization reduces the number of bits per vertex for positions from 96 bpv to **24 bpv**



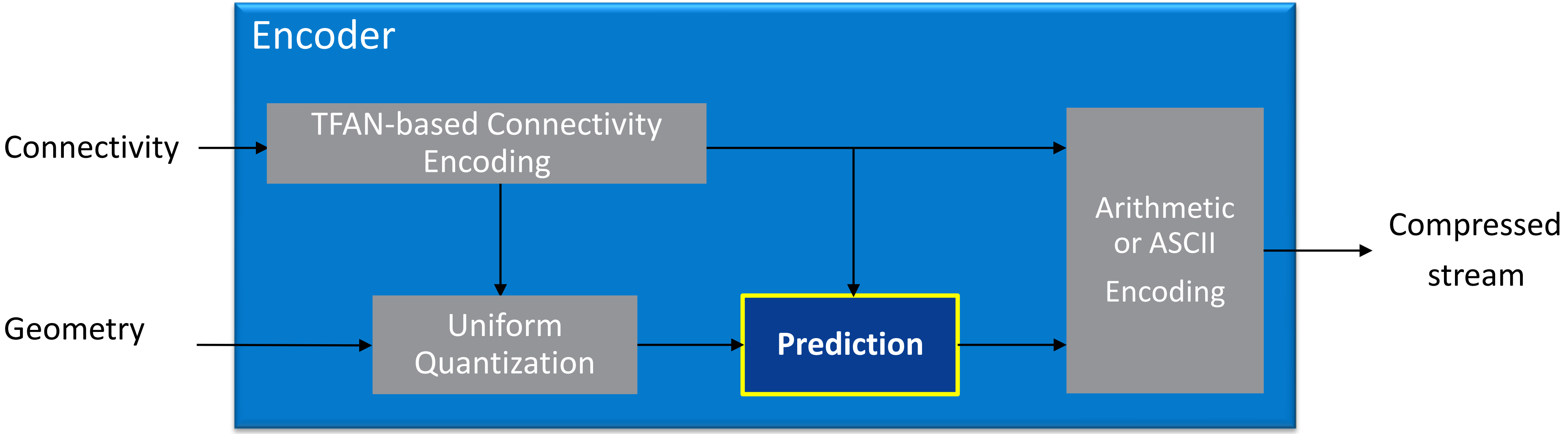




q=4 | q=6 | q=8 | q=10 | Original

ENCODER OVERVIEW

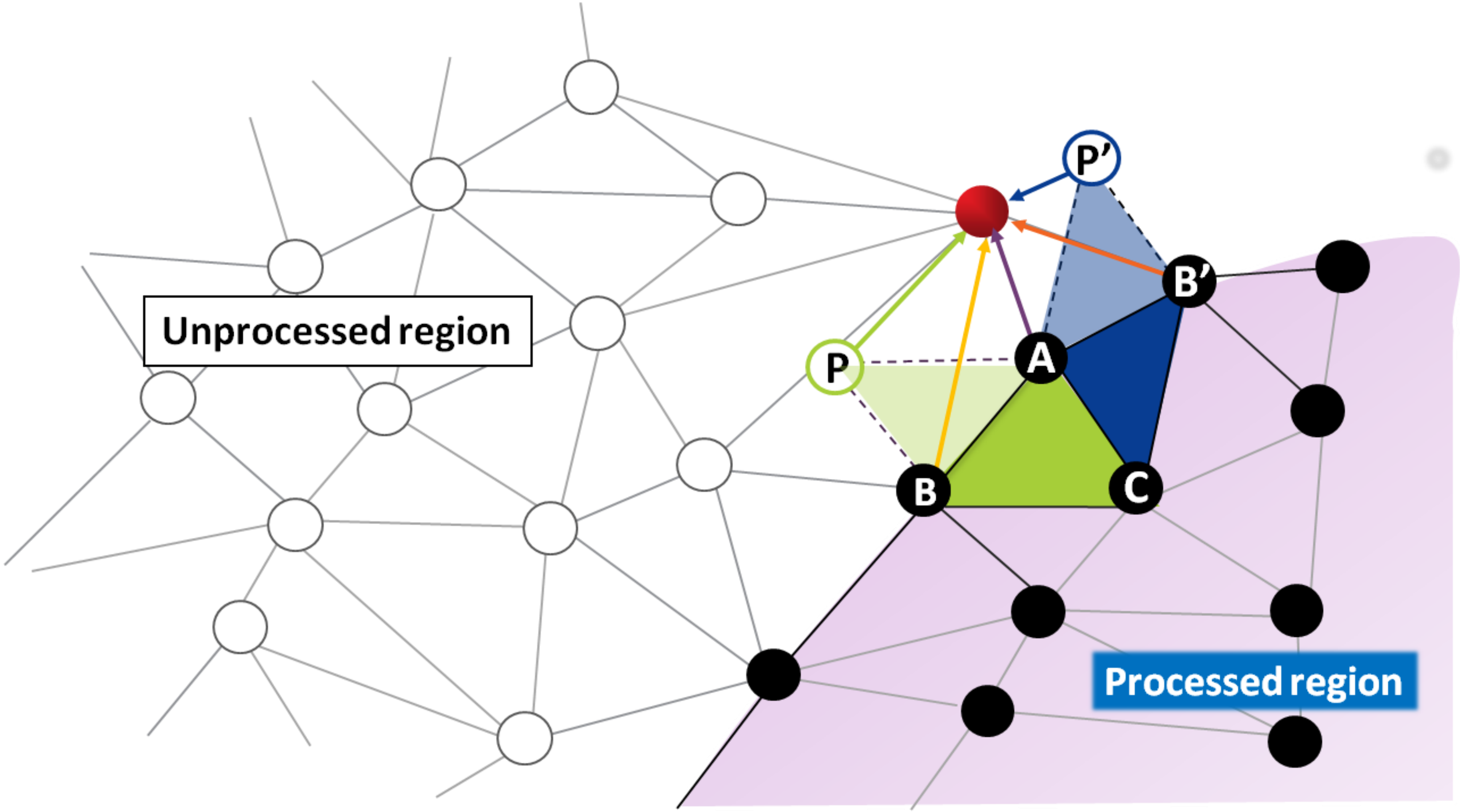
- Algorithm based on TFAN codec [Mammou'09]
  - Triangular meshes with attributes
  - Arbitrary topologies (e.g., manifold or not, open, closed, holes…)
- MPEG-SC3DMC (Scalable Complexity 3D Mesh Coding) published in 2010



[Mammou'09] K. Mammou, T. Zaharia, F. Prêteux, "TFAN: A low complexity 3D mesh compression algorithm" Computer Animation and Virtual Worlds, Vol. 20(2-3), pp. 343–354, 2009

ENCODER OVERVIEW

- Geometry prediction
  - Exploit connectivity information
  - Differential and “parallelogram” prediction

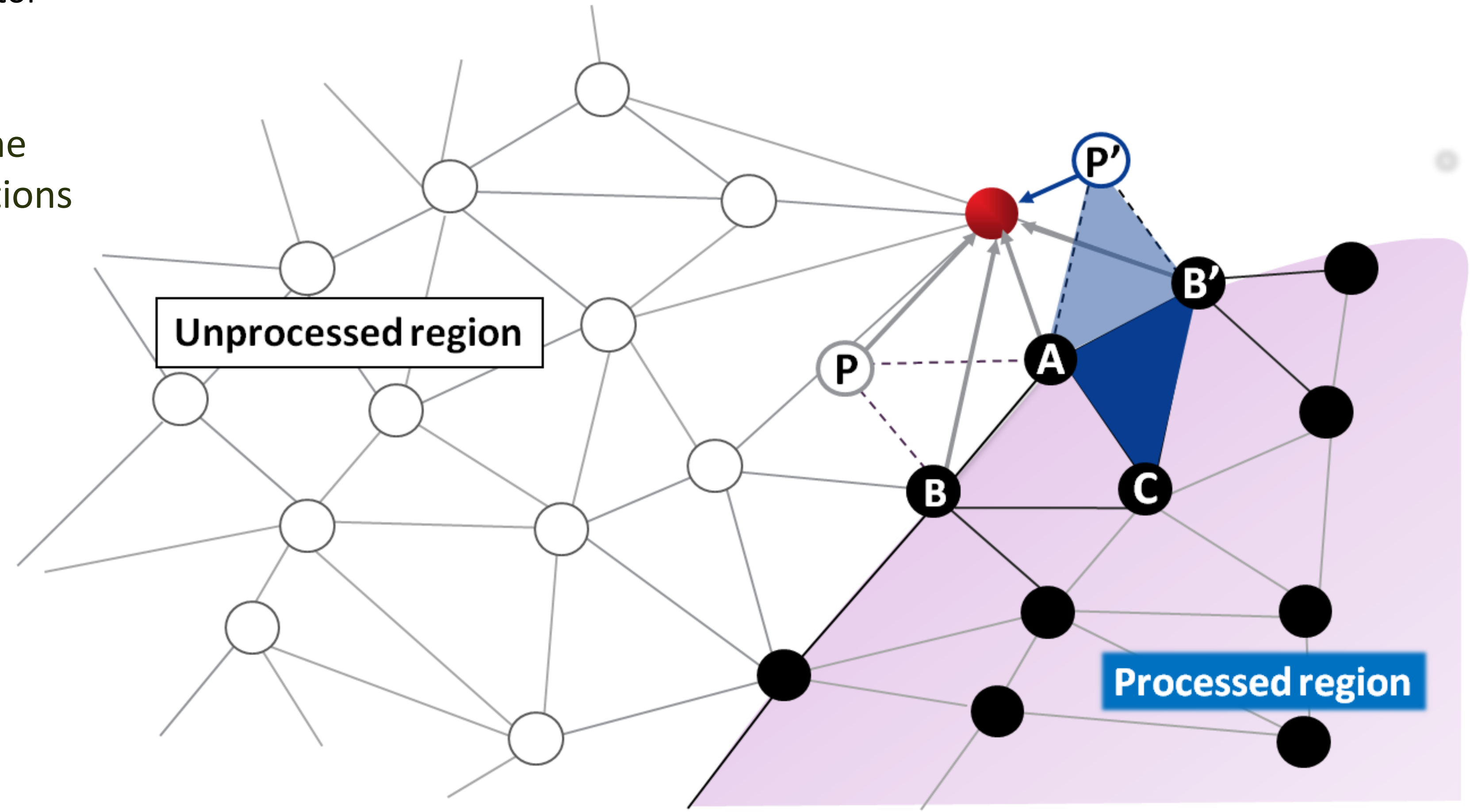


ENCODER OVERVIEW

- Geometry prediction
  - Exploit connectivity information
  - Differential and “parallelogram” prediction
  - Adaptively choose the best predictor

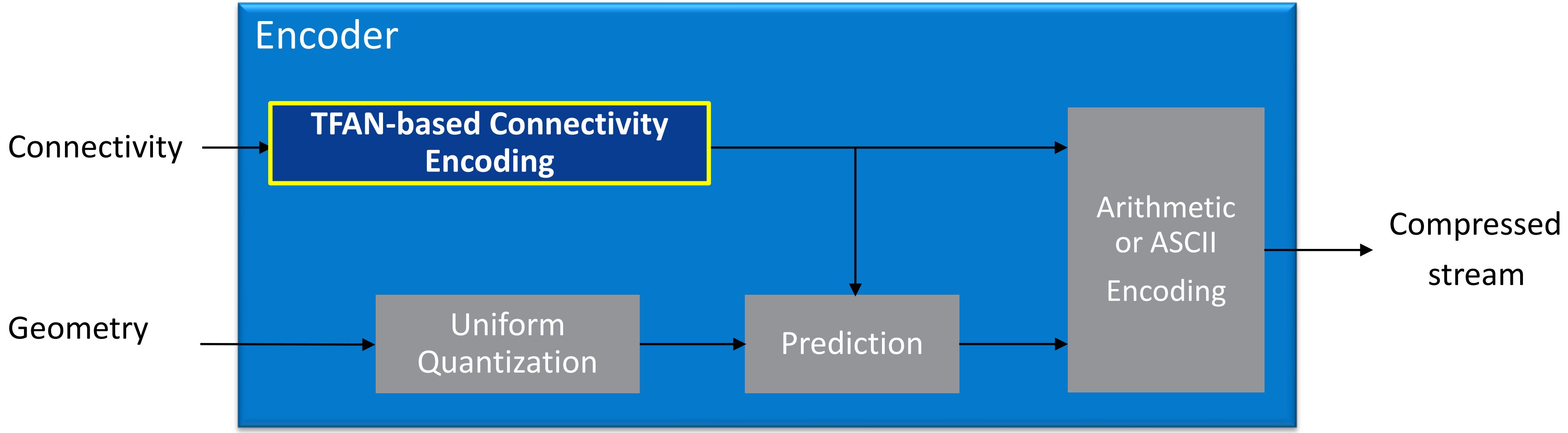
⬇

Geometry prediction reduces the number of bits per vertex for positions from 24 bpv to **10.3 bpv**



ENCODER OVERVIEW

- Algorithm based on TFAN codec [Mammou'09]
  - Triangular meshes with attributes
  - Arbitrary topologies (e.g., manifold or not, open, closed, holes…)
- MPEG-SC3DMC (Scalable Complexity 3D Mesh Coding) published in 2010

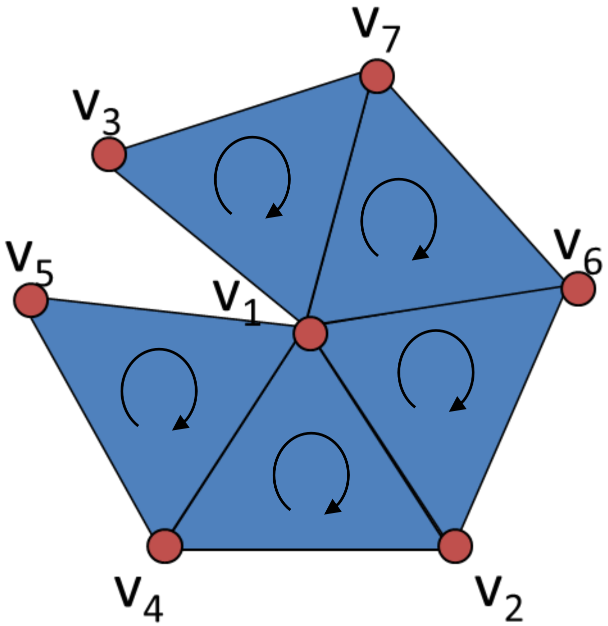


[Mammou'09] K. Mammou, T. Zaharia, F. Prêteux, "TFAN: A low complexity 3D mesh compression algorithm" Computer Animation and Virtual Worlds, Vol. 20(2-3), pp. 343–354, 2009

ENCODER OVERVIEW

- Triangle FAN
  - Each two successive triangles share a common edge
  - All triangles share a common vertex (i.e., center of the TFAN)
  - All the triangles have the same orientation
  - Described by enumerating the vertices in their traversal order

Implicitly encodes adjacency information

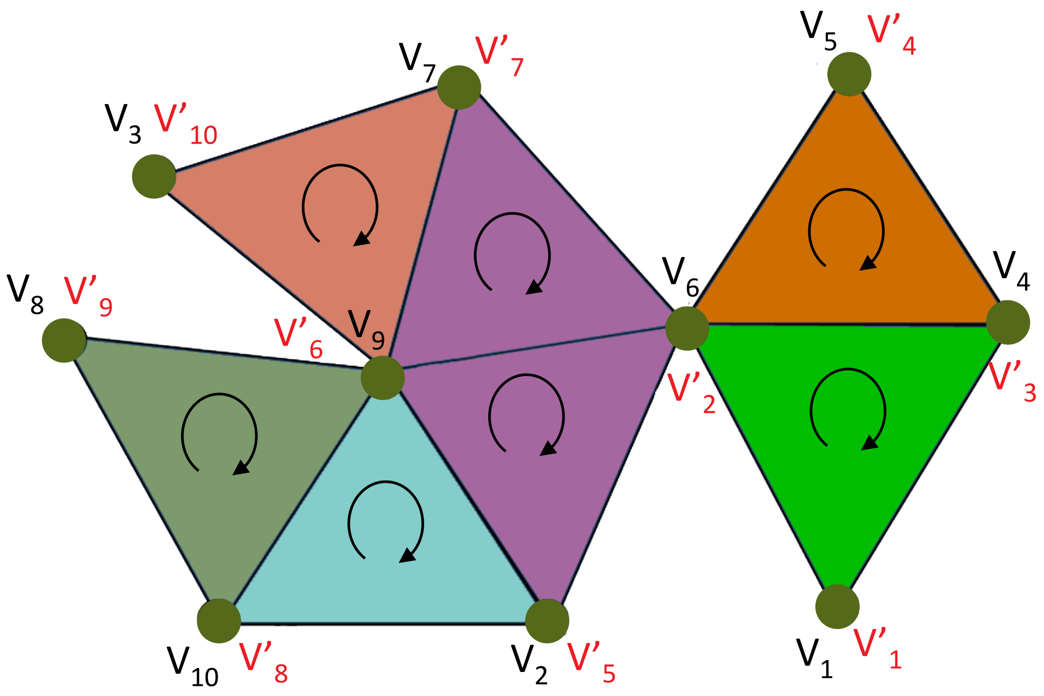


(1,3,7,6,2,4,5)

7 indices instead of 15 for IFS

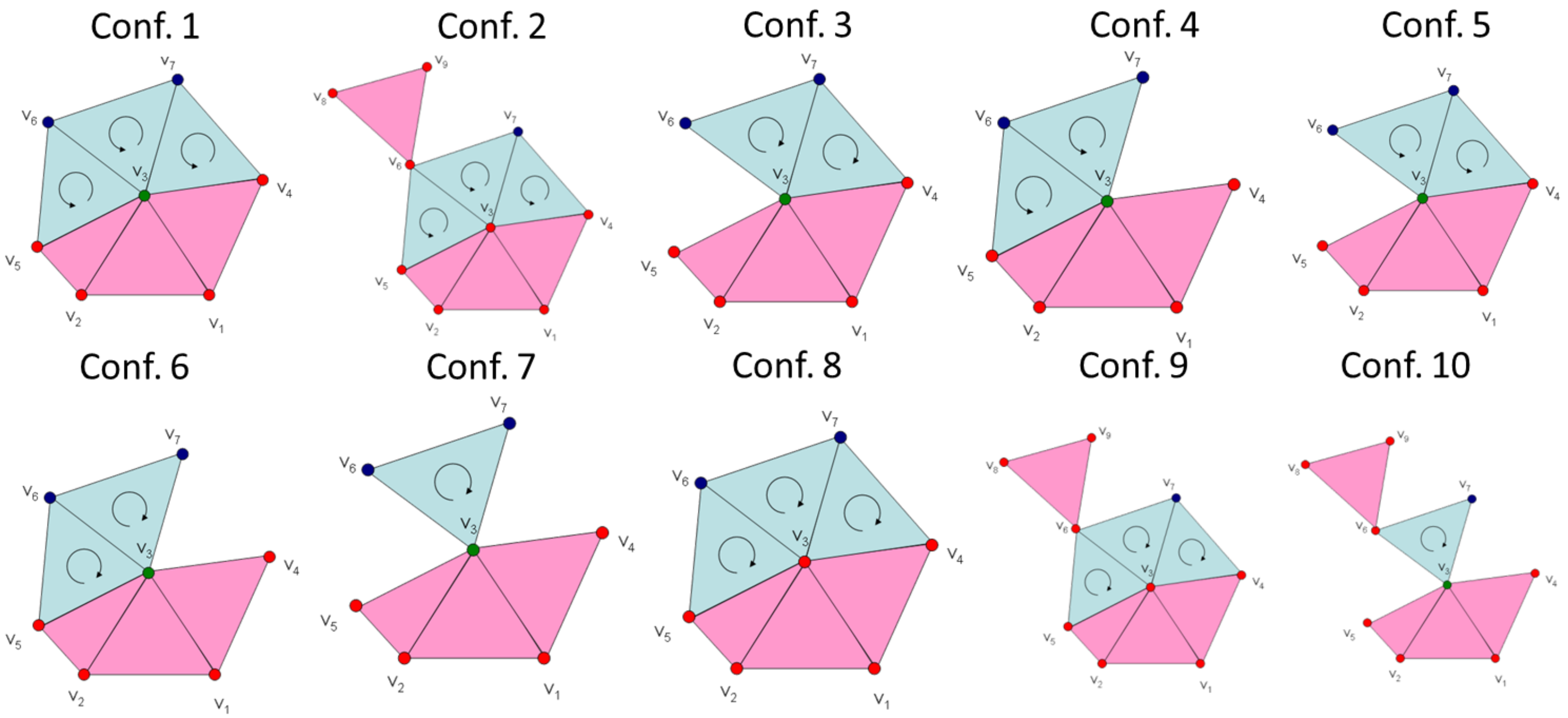
ENCODER OVERVIEW

– TFAN-based connectivity compression
  – Decompose the mesh into a set of TFANs
    – Traverse the vertices from neighbour to neighbour
    – Rename vertices according to the traversal order
    – For each vertex, group its incident non-visited triangles into TFANs



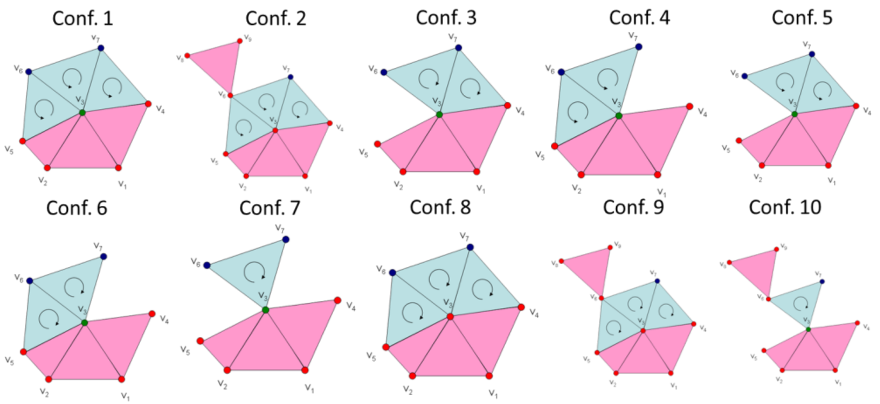
ENCODER OVERVIEW

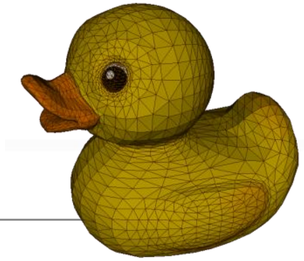
- TFAN-based connectivity compression
  - Decompose the mesh into a set of TFANs
    - Traverse the vertices from neighbour to neighbour
    - Rename vertices according to the traversal order
    - For each vertex, group its incident non-visited triangles into TFANs
  - Encode TFANs
    - Distinguish 10 topological configurations
    - Use local indices instead of absolute ones



ENCODER OVERVIEW

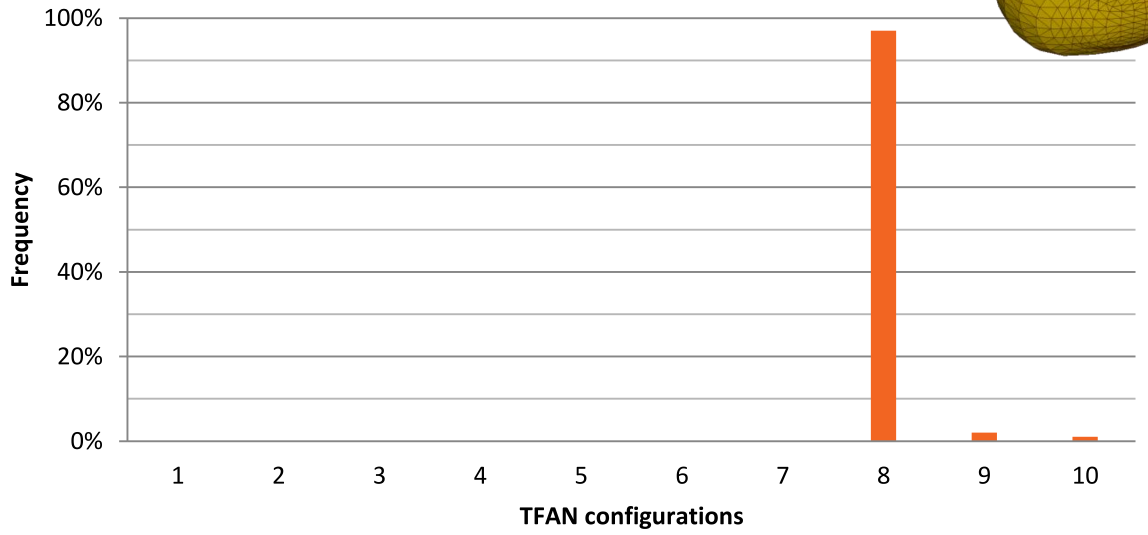
- TFAN-based connectivity compression
  - Decompose the mesh into a set of TFANs
    - Traverse the vertices from neighbour to neighbour
    - Rename vertices according to the traversal order
    - For each vertex, group its incident non-visited triangles into TFANs
  - Encode TFANs
    - Distinguish 10 topological configurations
    - Use local indices instead of absolute ones
  - Entropy encoding
    - Exploit statistical redundancy





⬇

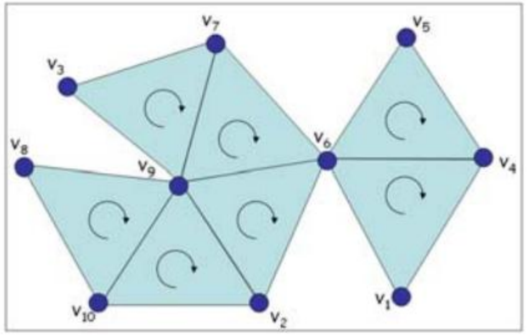
TFAN-based connectivity compression reduces the number of bits per vertex for connectivity from 96 bpv to **3.1 bpv**



CONNECTIVITY ENCODING EXAMPLE

Vertex
re-ordering

Neighbors
list

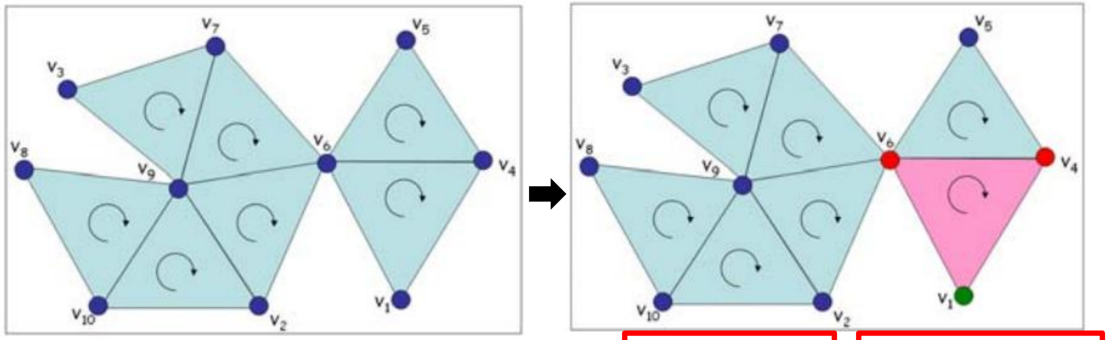


(a) $V = 10$

Bitstream

10

CONNECTIVITY ENCODING EXAMPLE



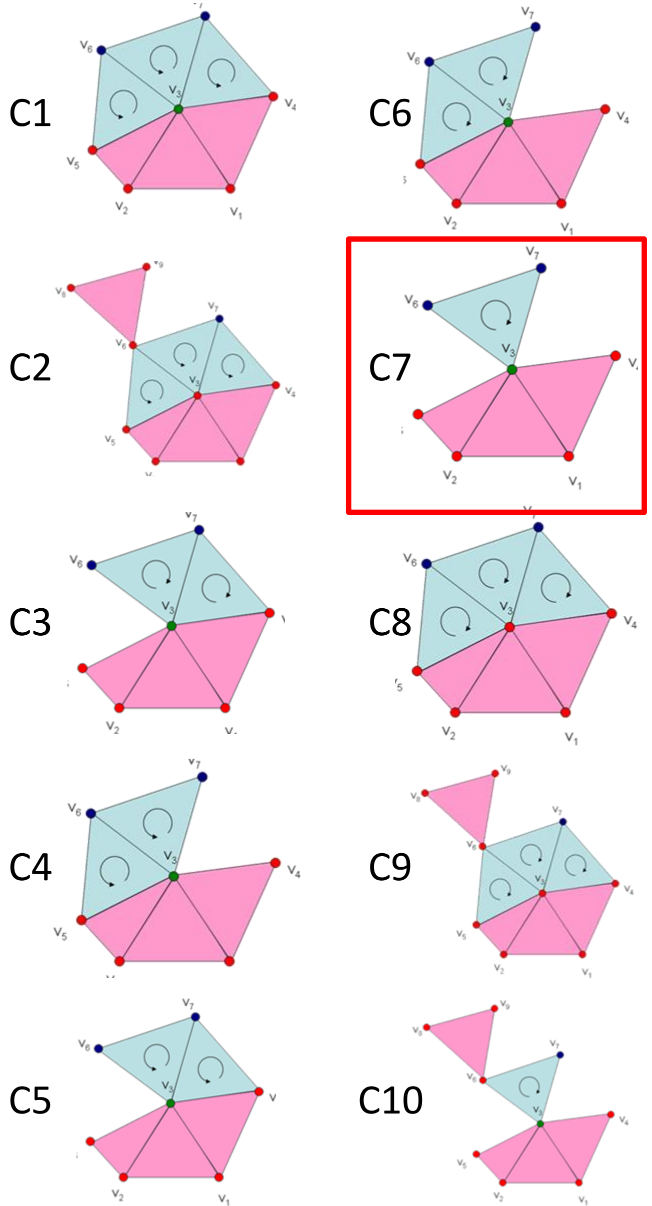
(a) $V = 10$ (b) $N(1) = 1$, $d(1,1) = 1$
$\mathcal{S}(1,1) = \{0,0\}$

Vertex re-ordering

V1 → V1

Neighbors list

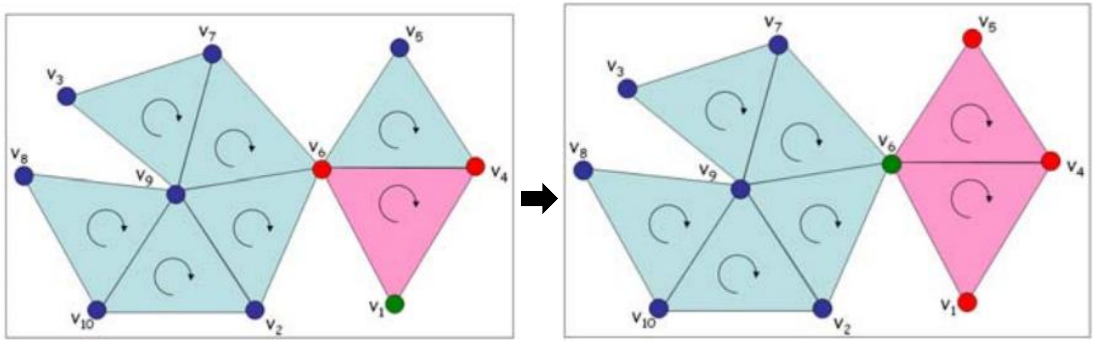
{}



Bitstream

10 1,(7,1)

CONNECTIVITY ENCODING EXAMPLE



(b) $N(1) = 1$, $d(1,1) = 1$ $\mathcal{S}(1,1) = \{0,0\}$

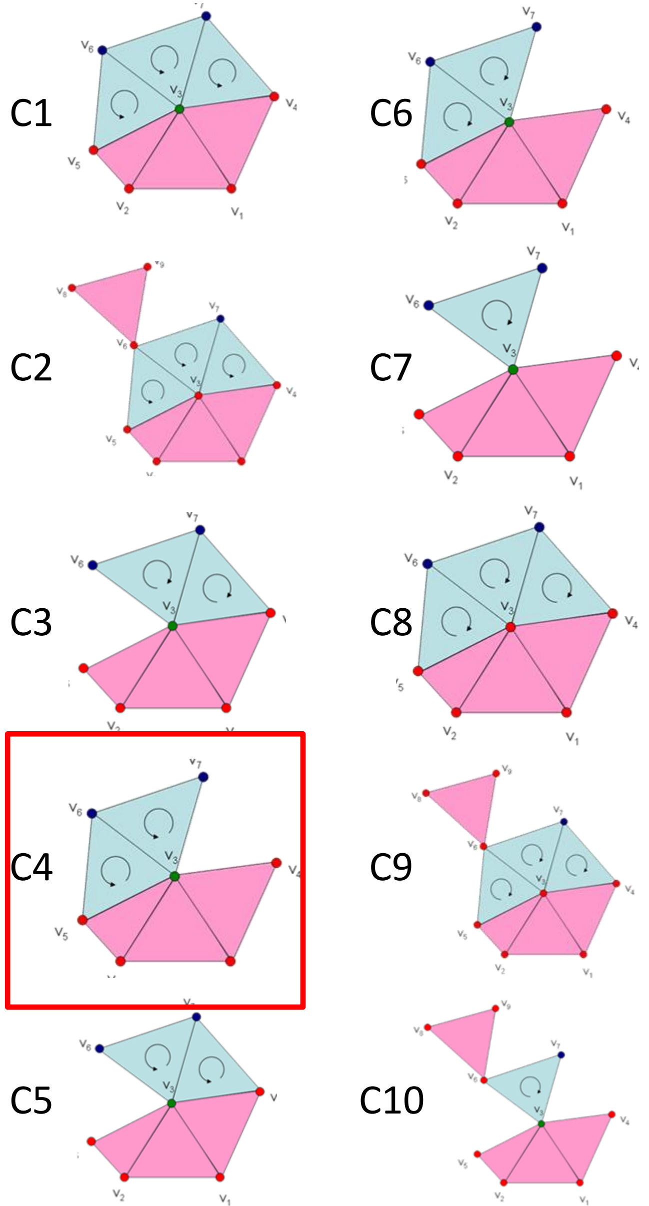
(c) $N(2) = 2$, $d(2,1) = 1$ $\mathcal{S}(2,1) = \{0,1\}$, $\mathcal{I}(2,1) = \{1\}$

Vertex re-ordering

V1 → V1
V6 → V2
V4 → V3

Neighbors list

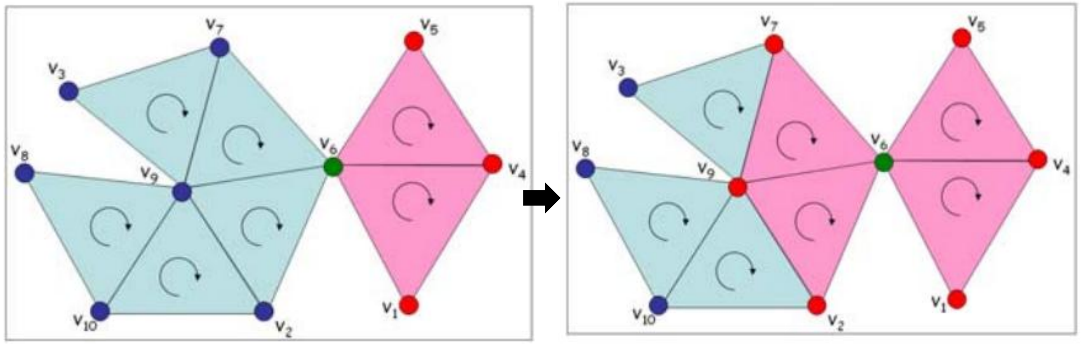
{V4}



Bitstream

10 1,(7,1) 2,(4,1)

CONNECTIVITY ENCODING EXAMPLE



(c) $N(2) = 2$, $d(2,1) = 1$ $\mathcal{S}(2,1) = \{0,1\}$, $\mathcal{I}(2,1) = \{1\}$

(d) $d(2,2) = 2$, $\mathcal{S}(2,2) = \{0,0,0\}$

Vertex re-ordering

V1 → V1

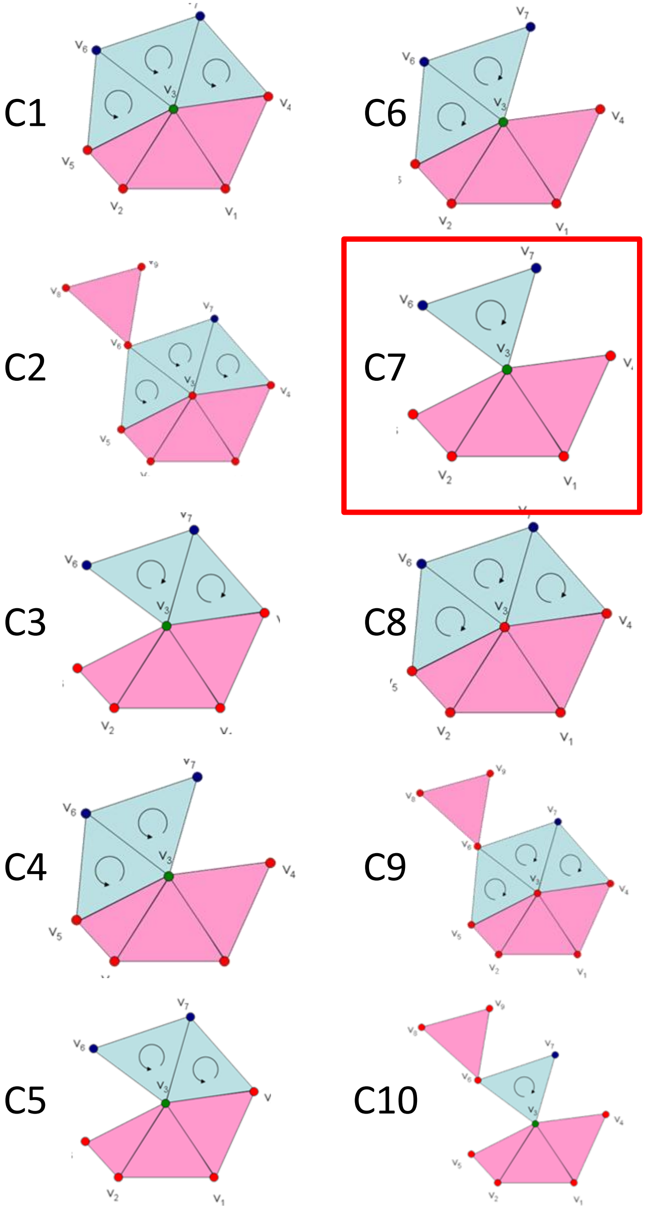
V6 → V2

V4 → V3

V5 → V4

Neighbors list

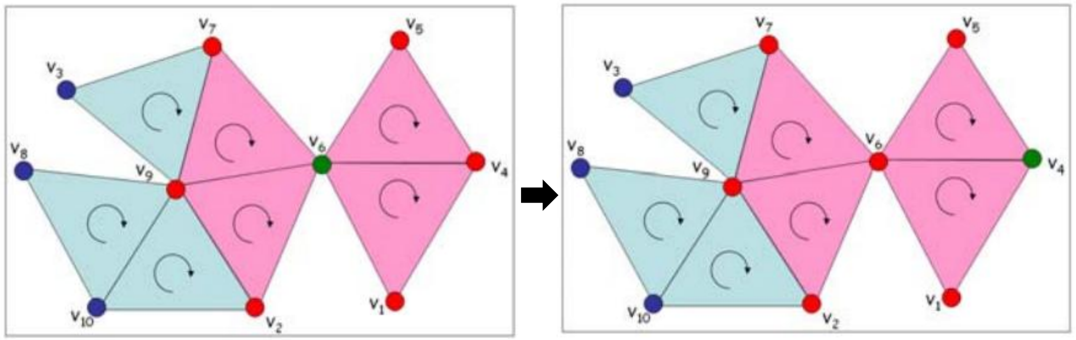
{V4, V5}



Bitstream

10 1,(7,1) 2,(4,1),(7,2)

CONNECTIVITY ENCODING EXAMPLE



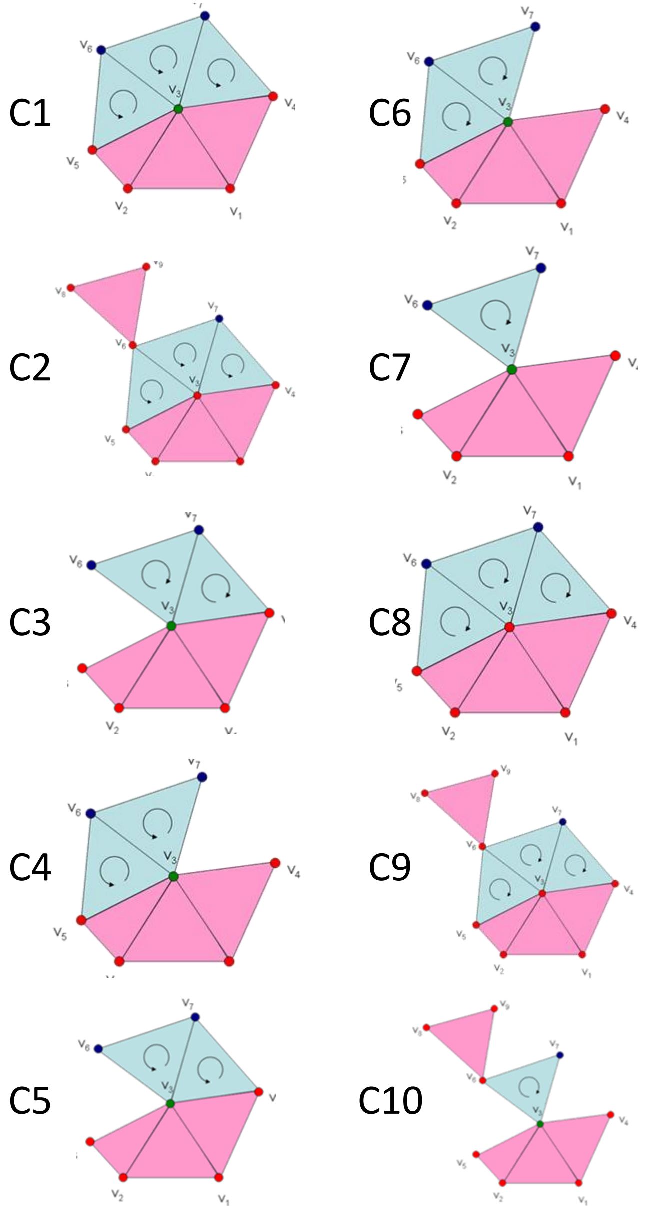
(d) $d(2,2) = 2,\ \mathcal{S}(2,2) = \{0,0,0\}$ (e) $N(3) = 0$

Vertex re-ordering

V1 → V1
V6 → V2
V4 → V3
V5 → V4
V2 → V5
V9 → V6
V7 → V7

Neighbors list

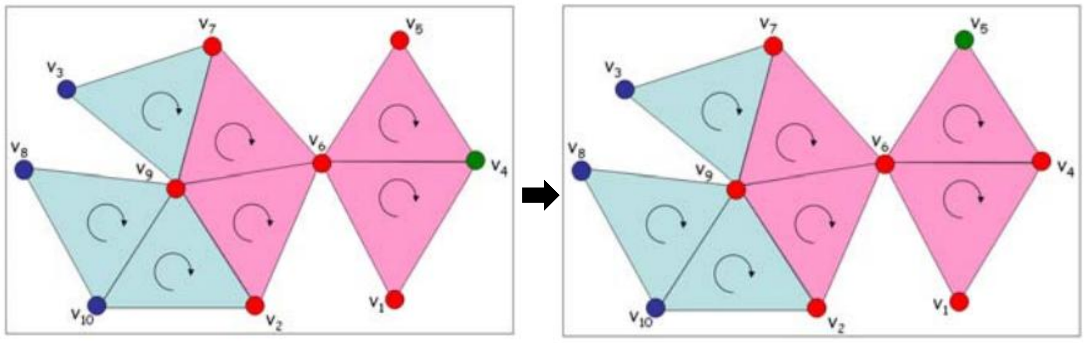
{V5, V2, V9, V7}



Bitstream

10 1,(7,1) 2,(4,1),(7,2) 0

CONNECTIVITY ENCODING EXAMPLE



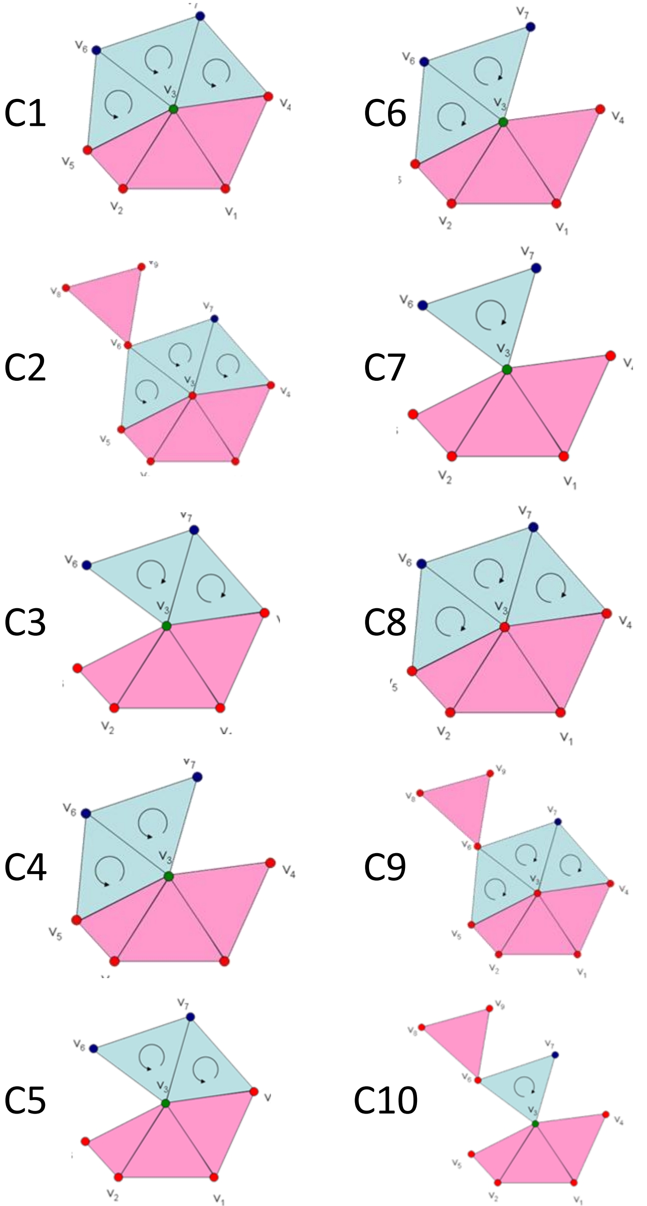
(e) $N(3) = 0$ (f) $N(4) = 0$

Vertex re-ordering

V1 → V1
V6 → V2
V4 → V3
V5 → V4
V2 → V5
V9 → V6
V7 → V7

Neighbors list

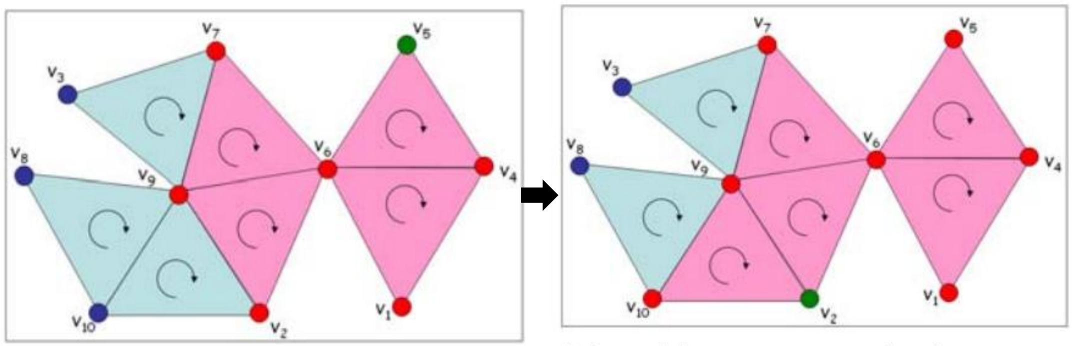
{}



Bitstream

10 1,(7,1) 2,(4,1),(7,2) 0 0

CONNECTIVITY ENCODING EXAMPLE



(f) $N(4) = 0$

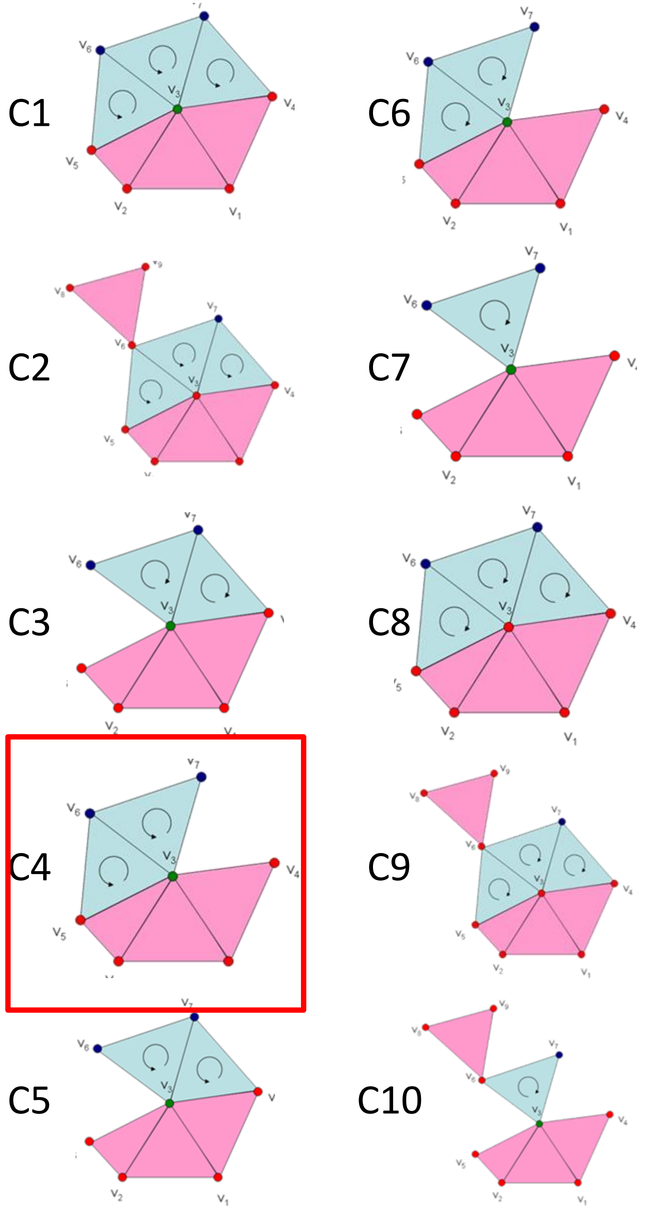
(g) $N(5) = 1$, $d(5,1) = 1$ $\mathcal{S}(5,1) = \{0,1\}$, $\mathcal{I}(5,1) = \{1\}$

Vertex re-ordering

V1 → V1
V6 → V2
V4 → V3
V5 → V4
V2 → V5
V9 → V6
V7 → V7

Neighbors list

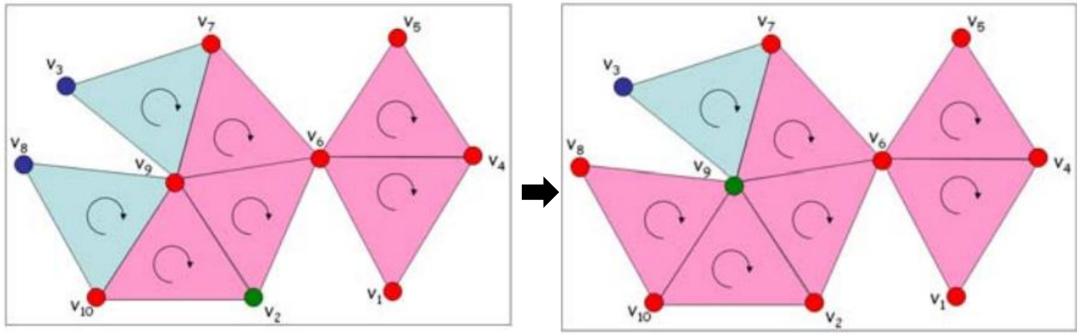
{V9}



Bitstream

10 1,(7,1) 2,(4,1),(7,2) 0 0 1,(4, 1)

CONNECTIVITY ENCODING EXAMPLE



(g) $N(5) = 1,\ d(5,1) = 1$ $\mathcal{S}(5,1) = \{0,1\}, \mathcal{I}(5,1) = \{1\}$

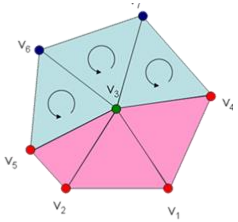
(h) $N(6) = 2,\ d(6,1) = 1$ $\mathcal{S}(6,1) = \{1,0\}, \mathcal{I}(6,1) = \{2\}$

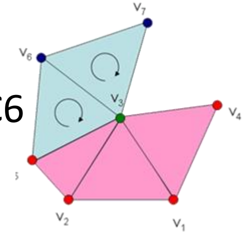
Vertex re-ordering

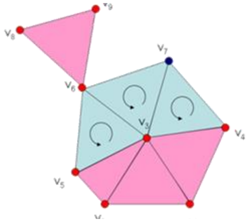
V1 → V1
V6 → V2
V4 → V3
V5 → V4
V2 → V5
V9 → V6
V7 → V7
V10 → V8

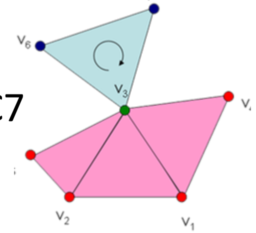
Neighbors list

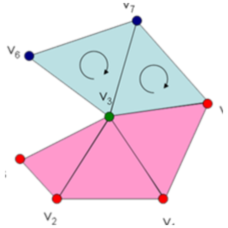
{V7,V10}

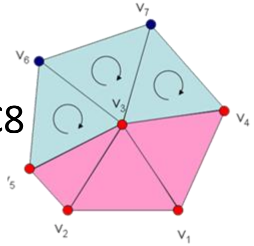
C1


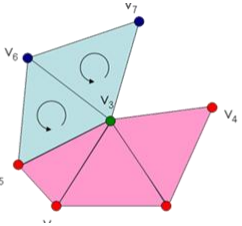
C6


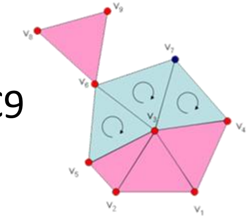
C2


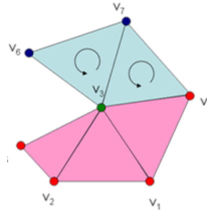
C7


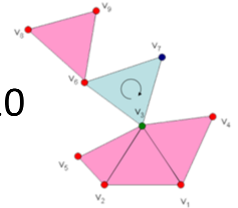
C3


C8


C4


C9


C5


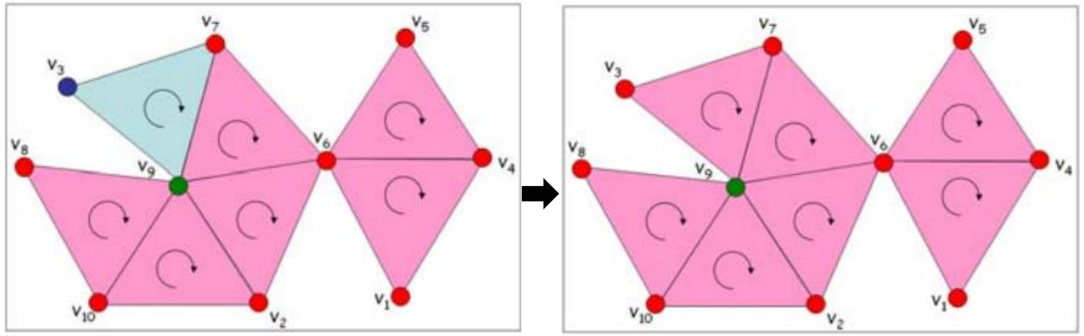
C10


Bitstream

10 1,(7,1) 2,(4,1),(7,2) 0 0 1,(4, 1)

2,(6,2)

CONNECTIVITY ENCODING EXAMPLE



(h) $N(6) = 2,\ d(6,1) = 1$ $\mathcal{S}(6,1) = \{1,0\},\ \mathcal{I}(6,1) = \{2\}$

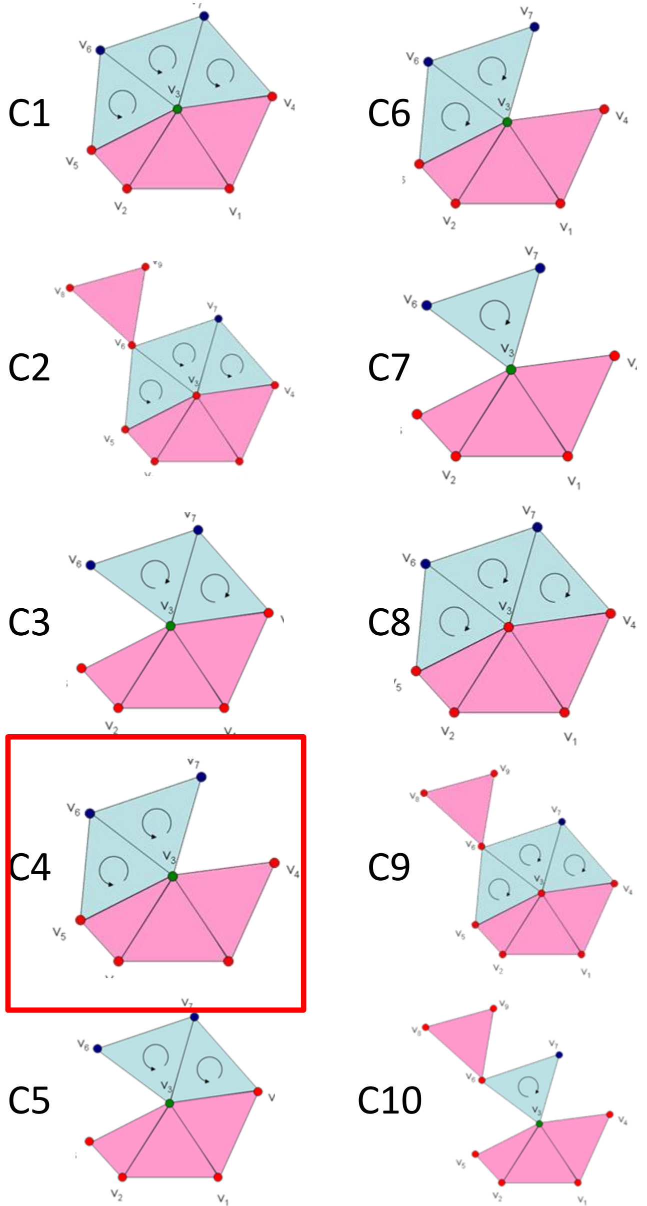
(i) $d(6,2) = 1,\ \mathcal{S}(6,2) = \{0,1\}$ $\mathcal{I}(6,2) = \{1\}$

Vertex re-ordering

V1 → V1
V6 → V2
V4 → V3
V5 → V4
V2 → V5
V9 → V6
V7 → V7
V10 → V8
V8 → V9

Neighbors list

{V7,V10,V8}

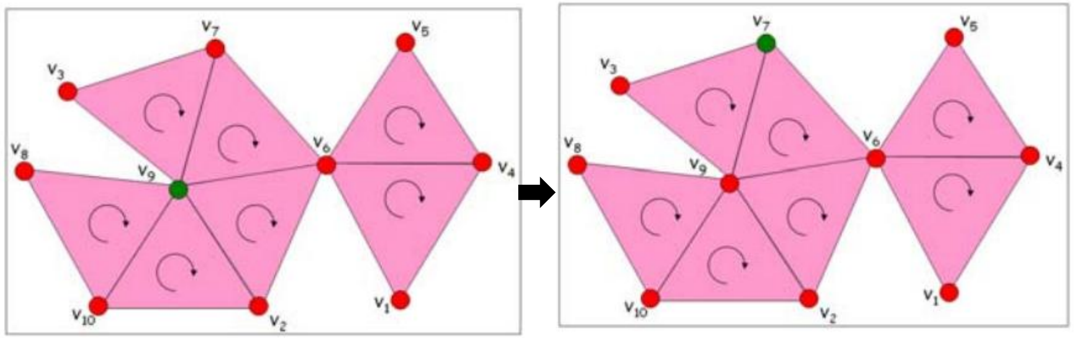


Bitstream

10 1,(7,1) 2,(4,1),(7,2) 0 0 1,(4, 1)

2,(6,2),(4,1)

CONNECTIVITY ENCODING EXAMPLE



(i) $d(6,2) = 1$, $\mathcal{S}(6,2) = \{0,1\}$ $\mathcal{I}(6,2) = \{1\}$

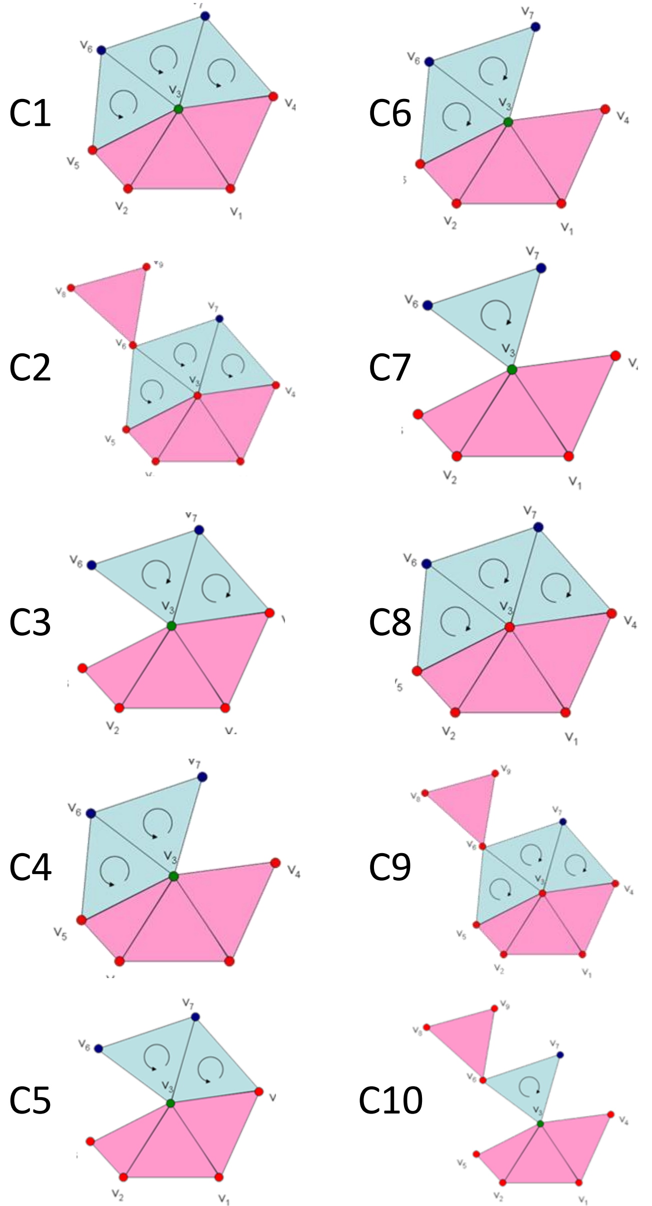
(j) $N(7) = 0$

Vertex re-ordering

V1 → V1
V6 → V2
V4 → V3
V5 → V4
V2 → V5
V9 → V6
V7 → V7
V10 → V8
V8 → V9
V3 → V10

Neighbors list

{V3}

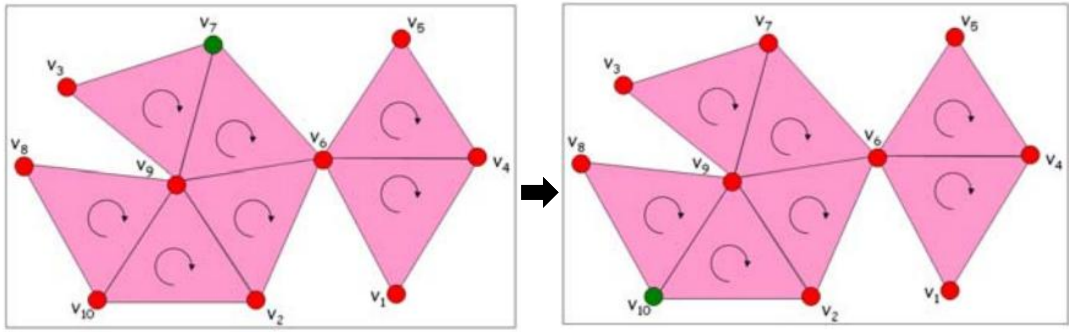


Bitstream

10 1,(7,1) 2,(4,1),(7,2) 0 0 1,(4, 1)

2,(6,2),(4,1) 0

CONNECTIVITY ENCODING EXAMPLE



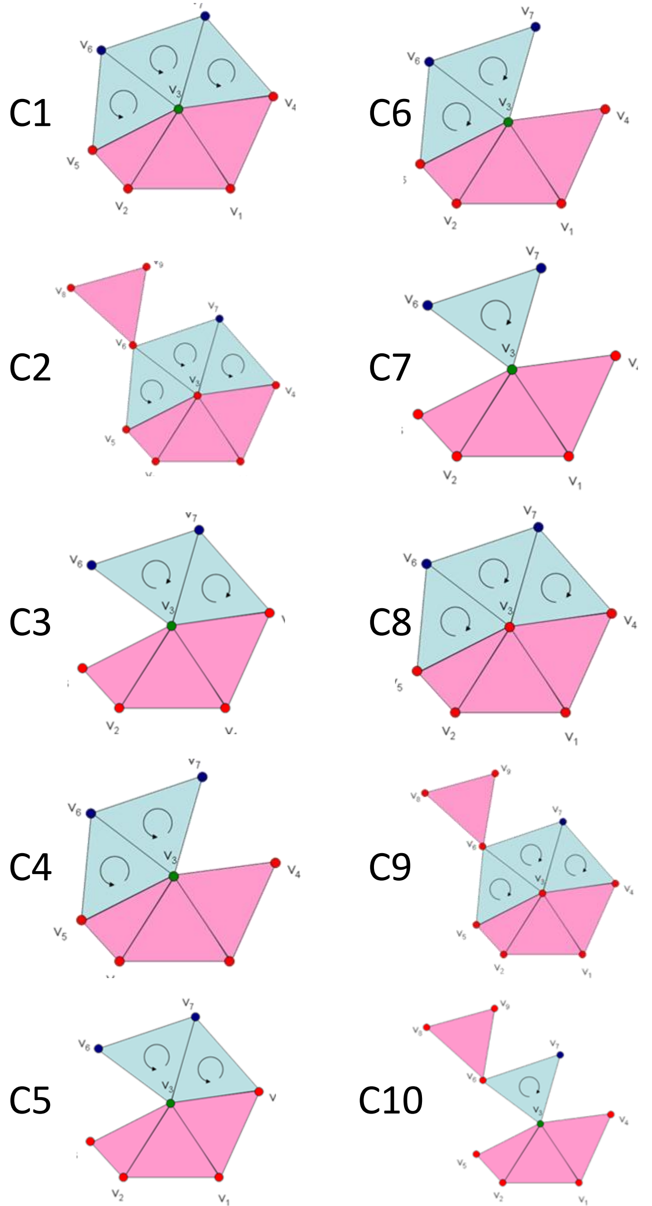
(j) $N(7) = 0$ (k) $N(8) = 0$

Vertex re-ordering

V1 → V1
V6 → V2
V4 → V3
V5 → V4
V2 → V5
V9 → V6
V7 → V7
V10 → V8
V8 → V9
V3 → V10

Neighbors list

{V8}

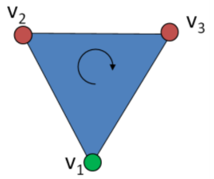


Bitstream

10 1,(7,1) 2,(4,1),(7,2) 0 0 1,(4, 1)

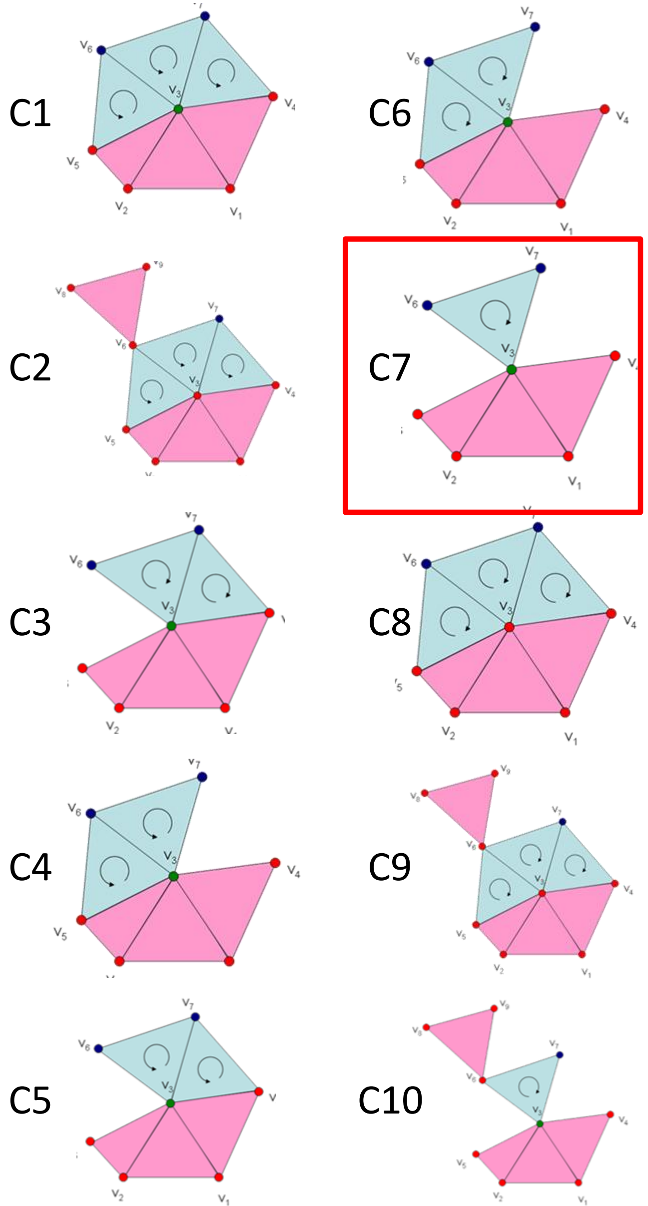
2,(6,2),(4,1) 0 0

CONNECTIVITY DECODING EXAMPLE



Neighbors list

{}

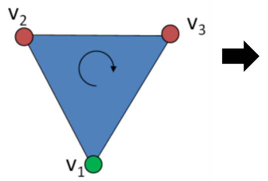


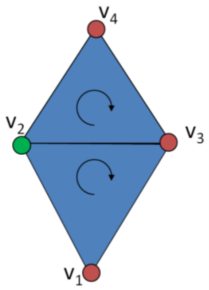
Bitstream

10 1,(7,1) 2,(4,1),(7,2) 0 0 1,(4, 1)

2,(6,2),(4,1) 0 0 0 0

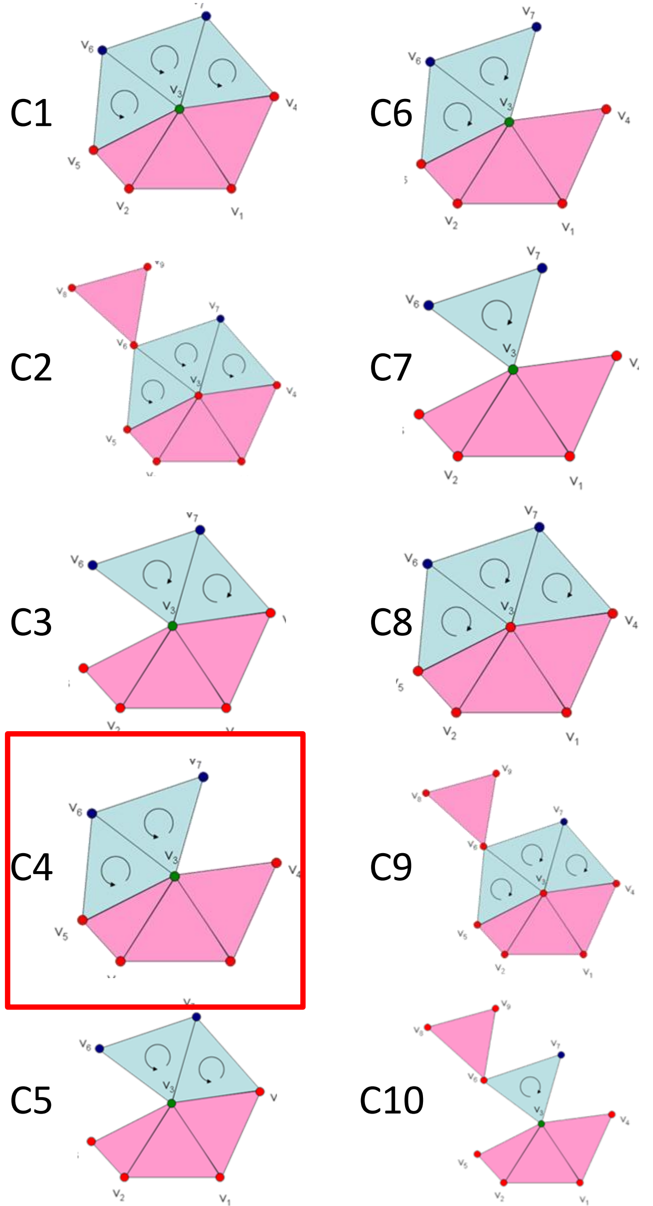
CONNECTIVITY DECODING EXAMPLE





Neighbors list

{V3}

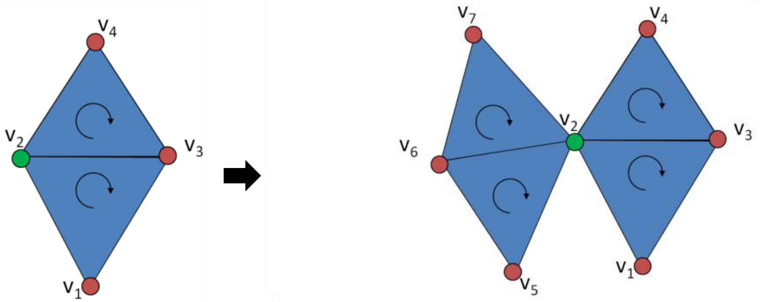


Bitstream

10 1,(7,1) 2,(4,1),(7,2) 0 0 1,(4, 1)

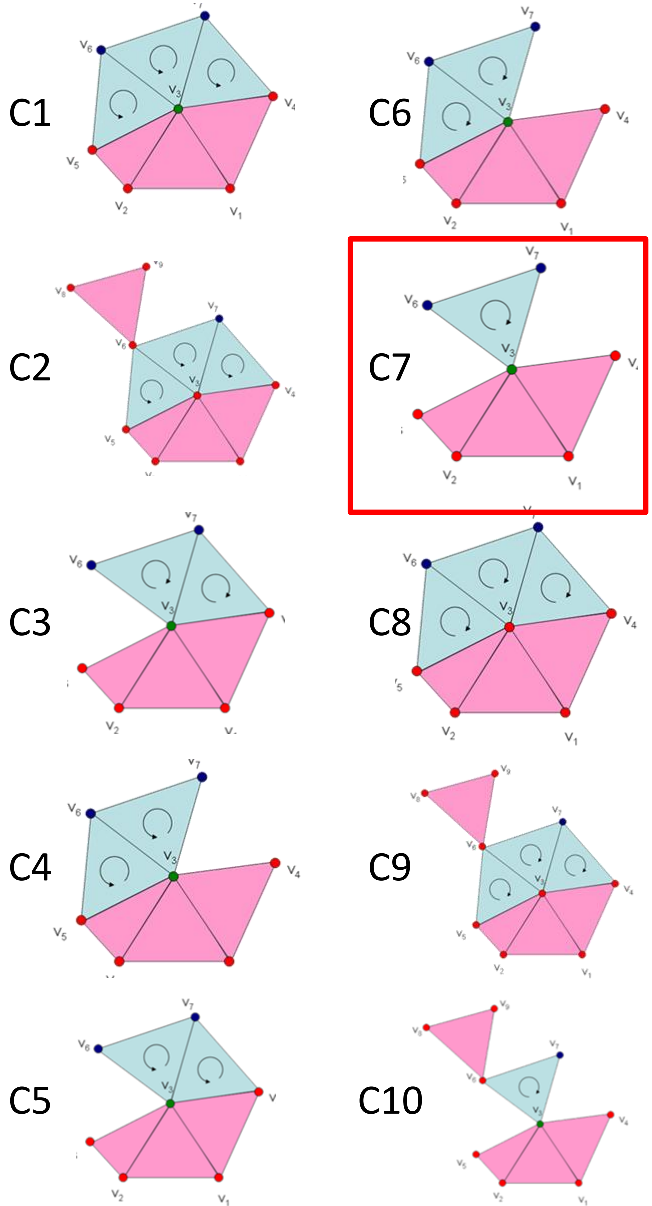
2,(6,2),(4,1) 0 0 0 0

CONNECTIVITY DECODING EXAMPLE



Neighbors list

{V3, V4}

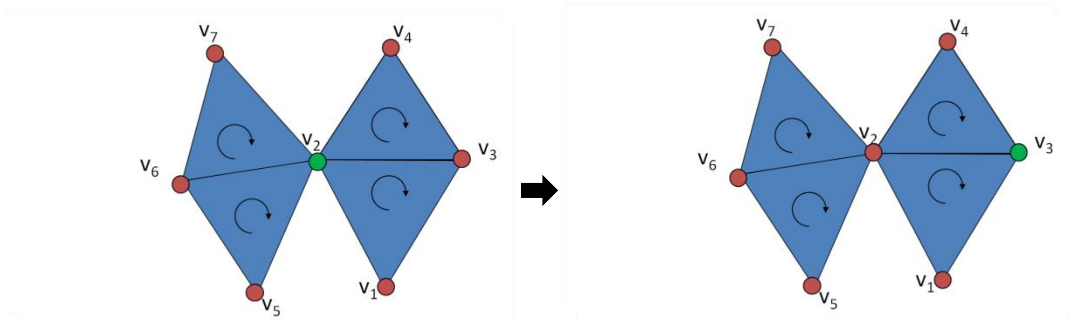


Bitstream

10 1,(7,1) 2,(4,1),(7,2) 0 0 1,(4, 1)

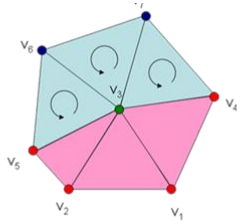
2,(6,2),(4,1) 0 0 0 0

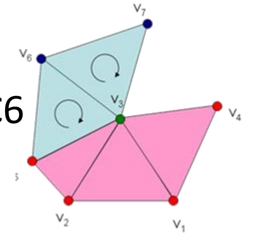
CONNECTIVITY DECODING EXAMPLE

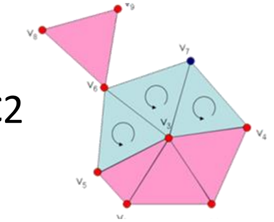


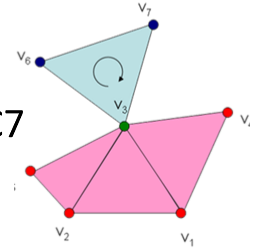
Neighbors list

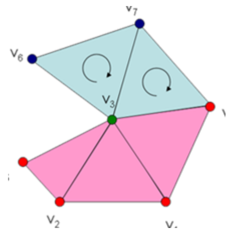
{V4}

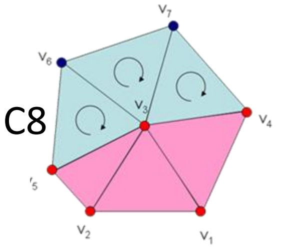
C1


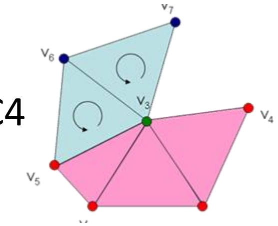
C6


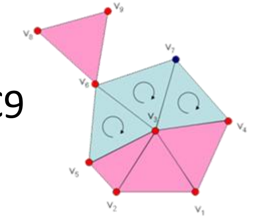
C2


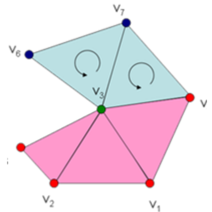
C7


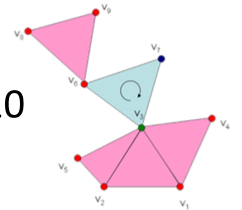
C3


C8


C4


C9


C5


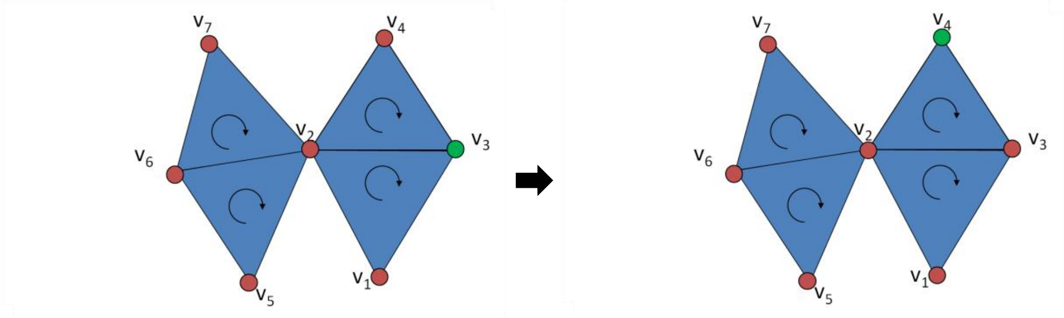
C10


Bitstream

10 1,(7,1) 2,(4,1),(7,2) 0 0 1,(4, 1)

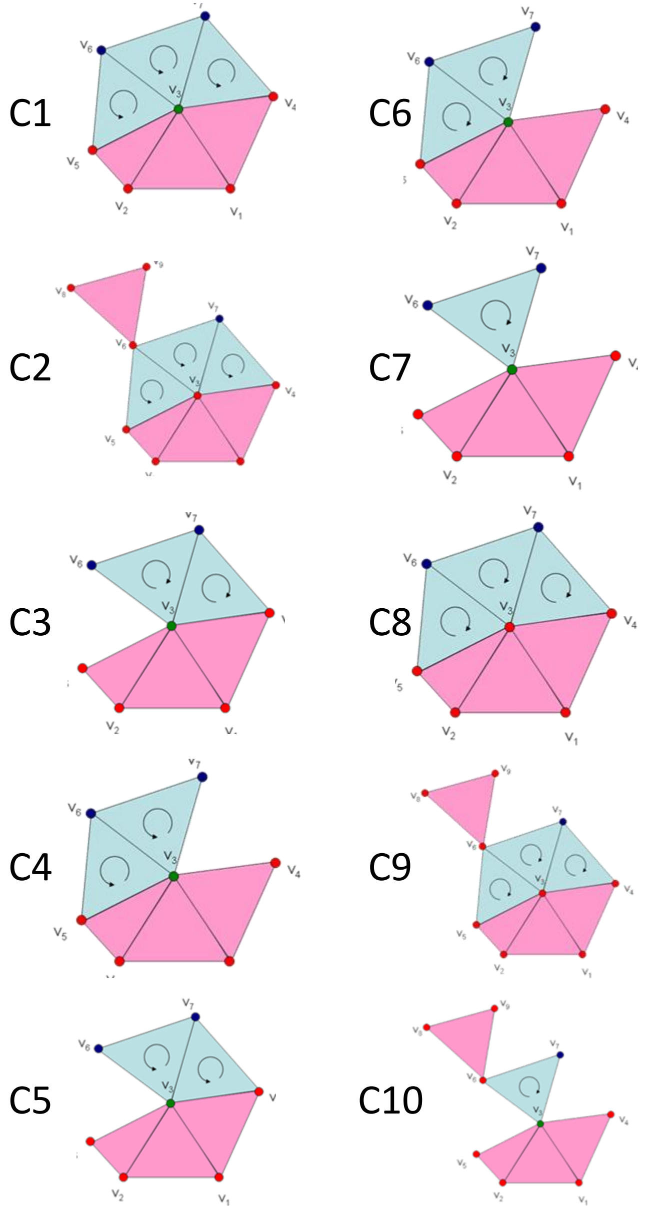
2,(6,2),(4,1) 0 0 0 0

CONNECTIVITY DECODING EXAMPLE



Neighbors list

{}

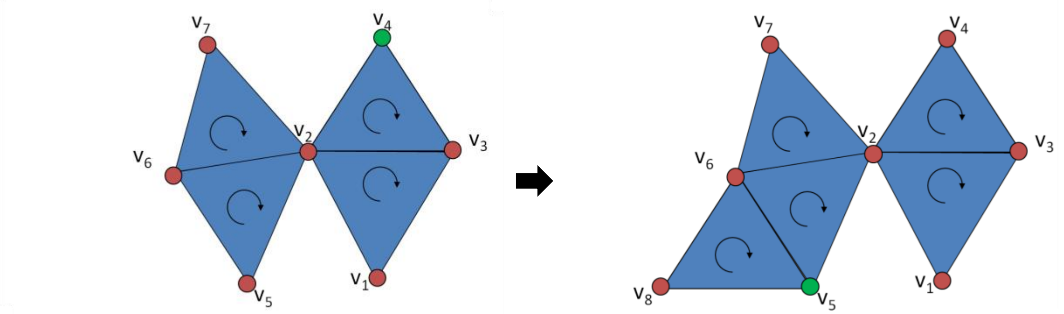


Bitstream

10 1,(7,1) 2,(4,1),(7,2) 0 0 1,(4, 1)

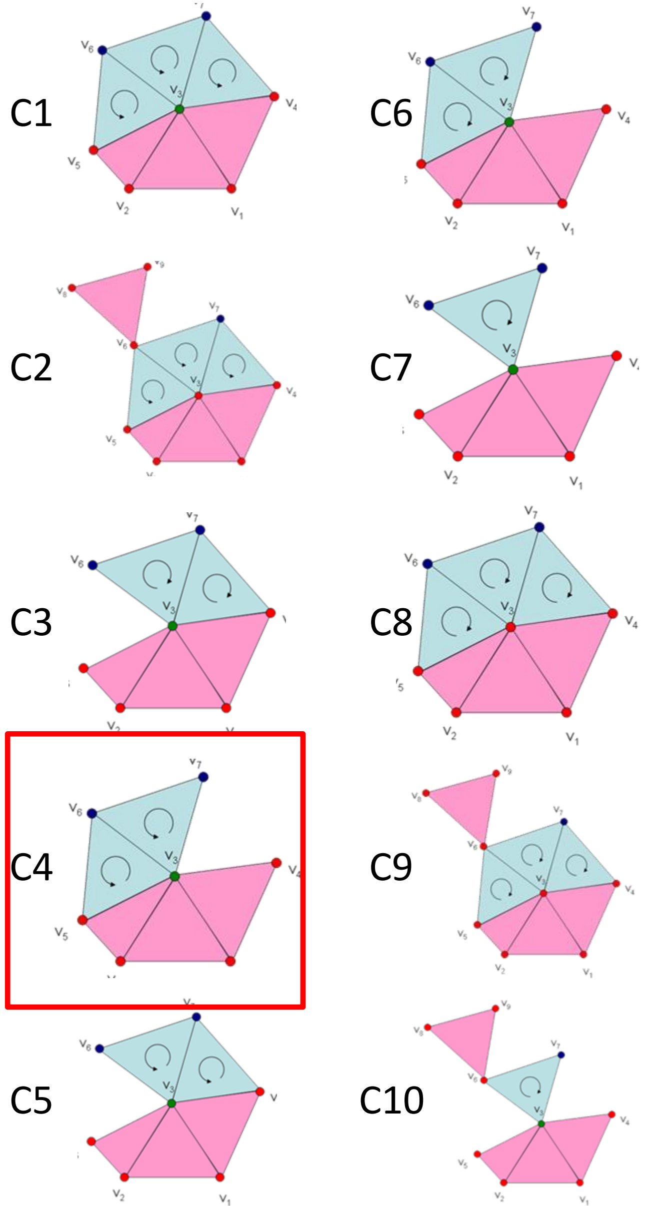
2,(6,2),(4,1) 0 0 0 0

CONNECTIVITY DECODING EXAMPLE



Neighbors list

{V6}

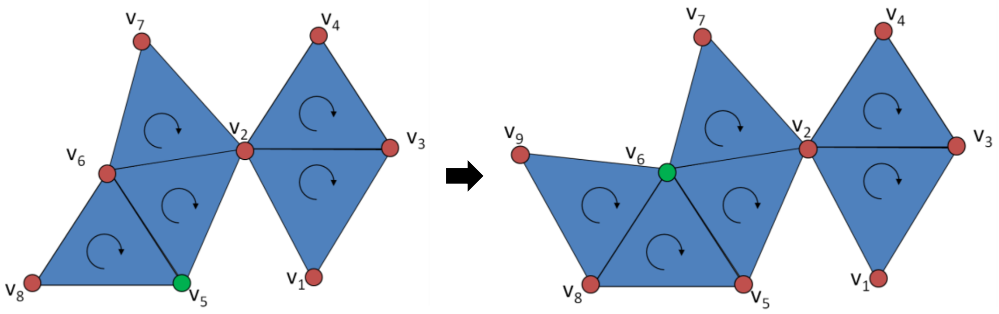


Bitstream

10 1,(7,1) 2,(4,1),(7,2) 0 0 1,(4, 1)

2,(6,2),(4,1) 0 0 0 0

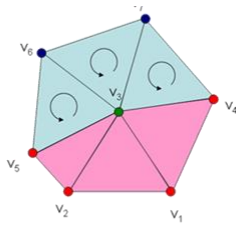
CONNECTIVITY DECODING EXAMPLE



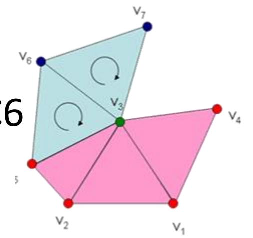
Neighbors list

{V7, V8}

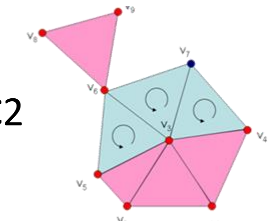
C1



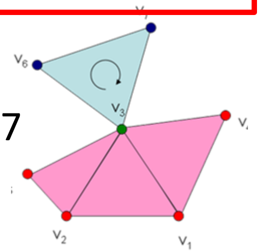
C6



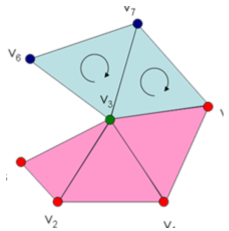
C2



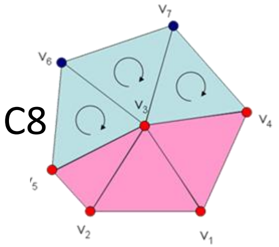
C7



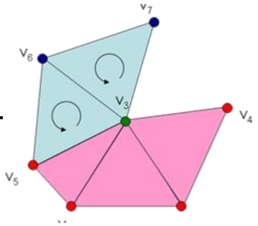
C3



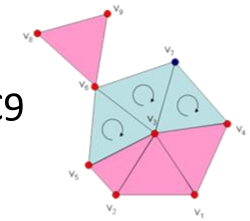
C8



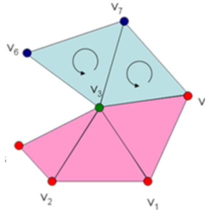
C4



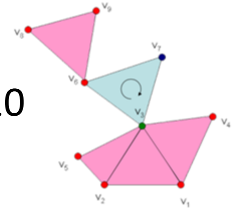
C9



C5



C10

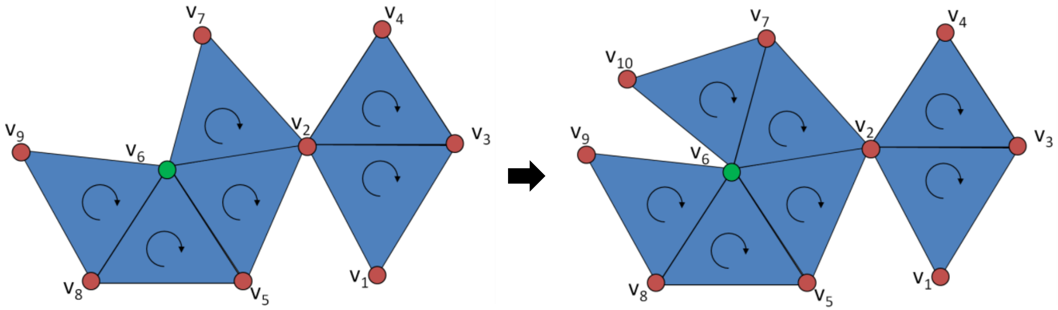


Bitstream

10 1,(7,1) 2,(4,1),(7,2) 0 0 1,(4, 1)

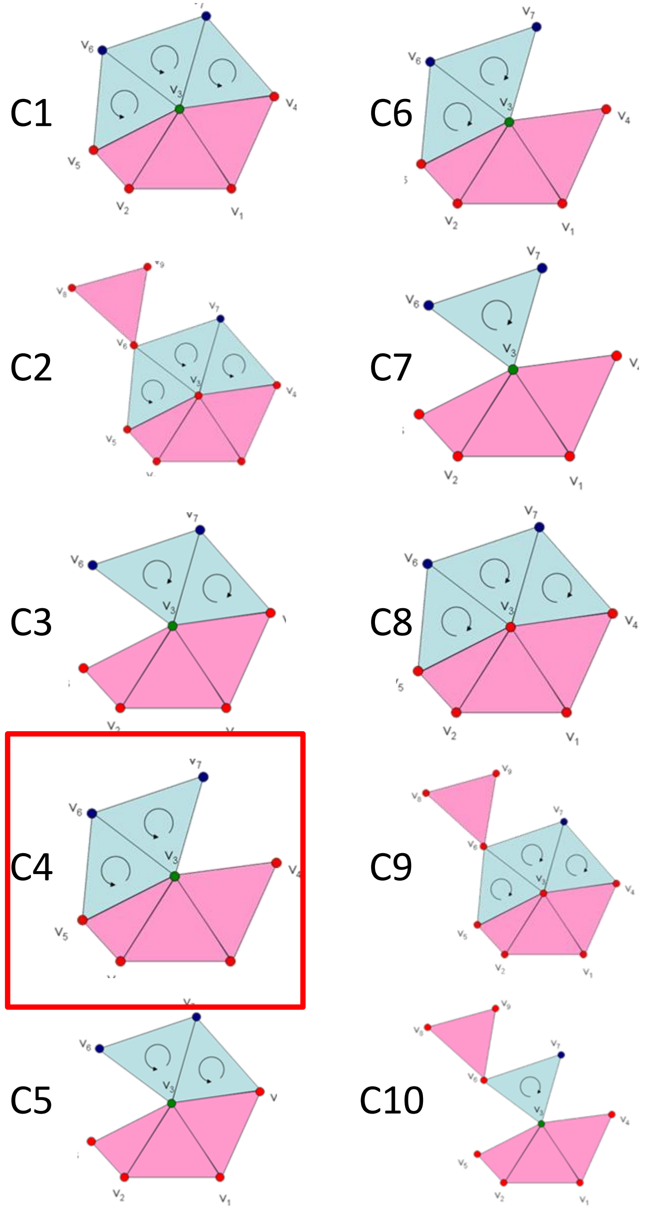
2,(6,2),(4,1) 0 0 0 0

CONNECTIVITY DECODING EXAMPLE



Neighbors list

{V7, V8, V9}

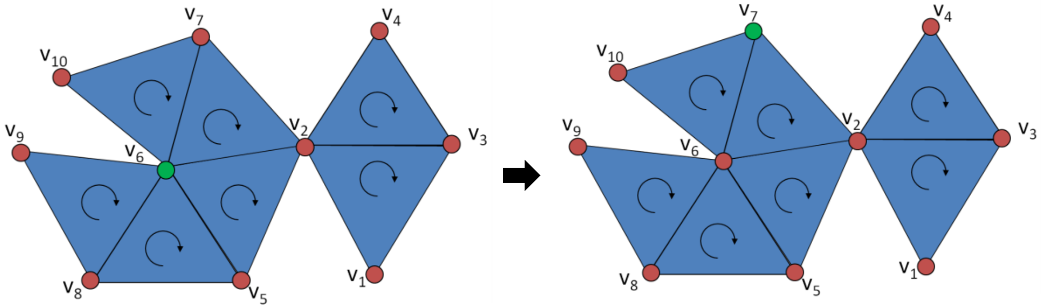


Bitstream

10 1,(7,1) 2,(4,1),(7,2) 0 0 1,(4, 1)

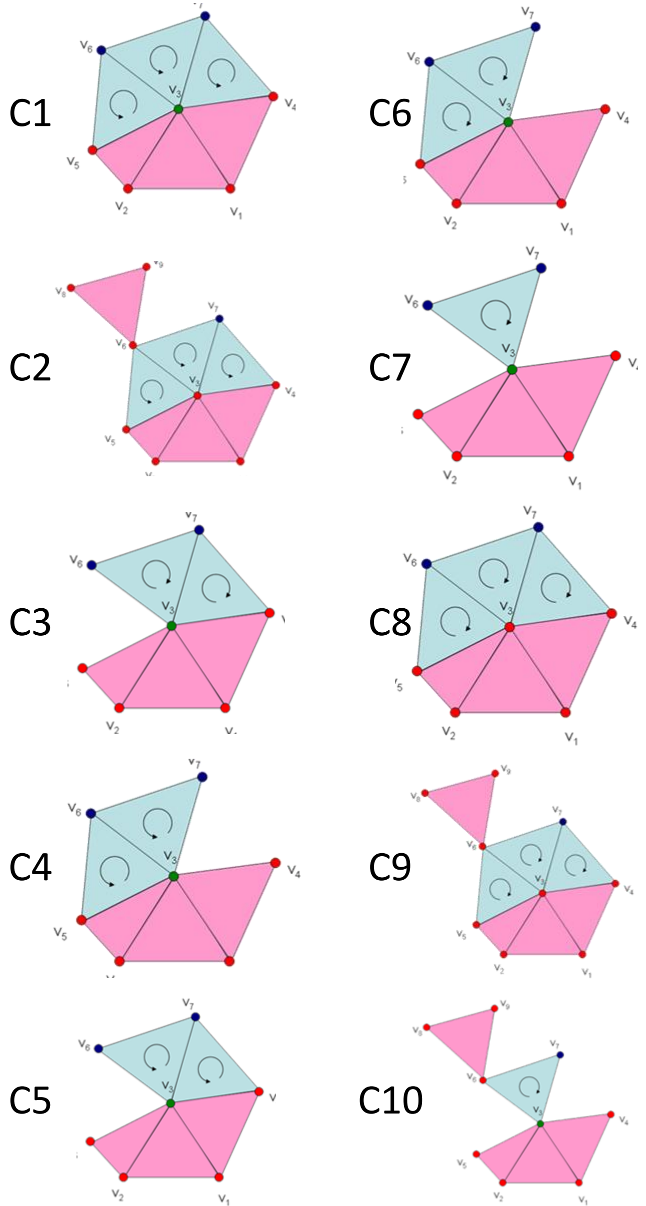
2,(6,2),(4,1) 0 0 0 0

CONNECTIVITY DECODING EXAMPLE



Neighbors list

{V9}

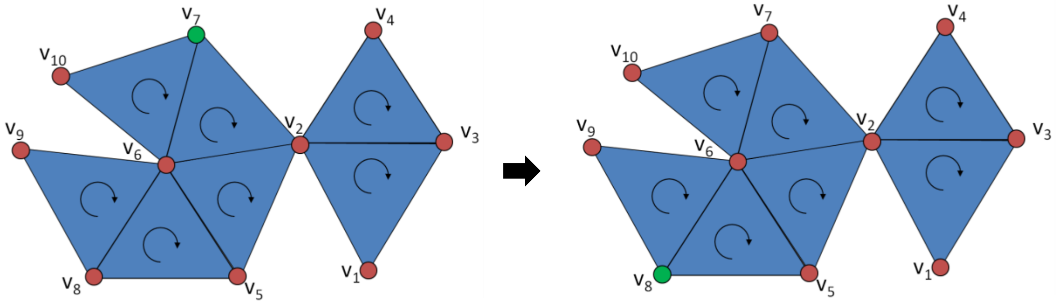


Bitstream

10 1,(7,1) 2,(4,1),(7,2) 0 0 1,(4, 1)

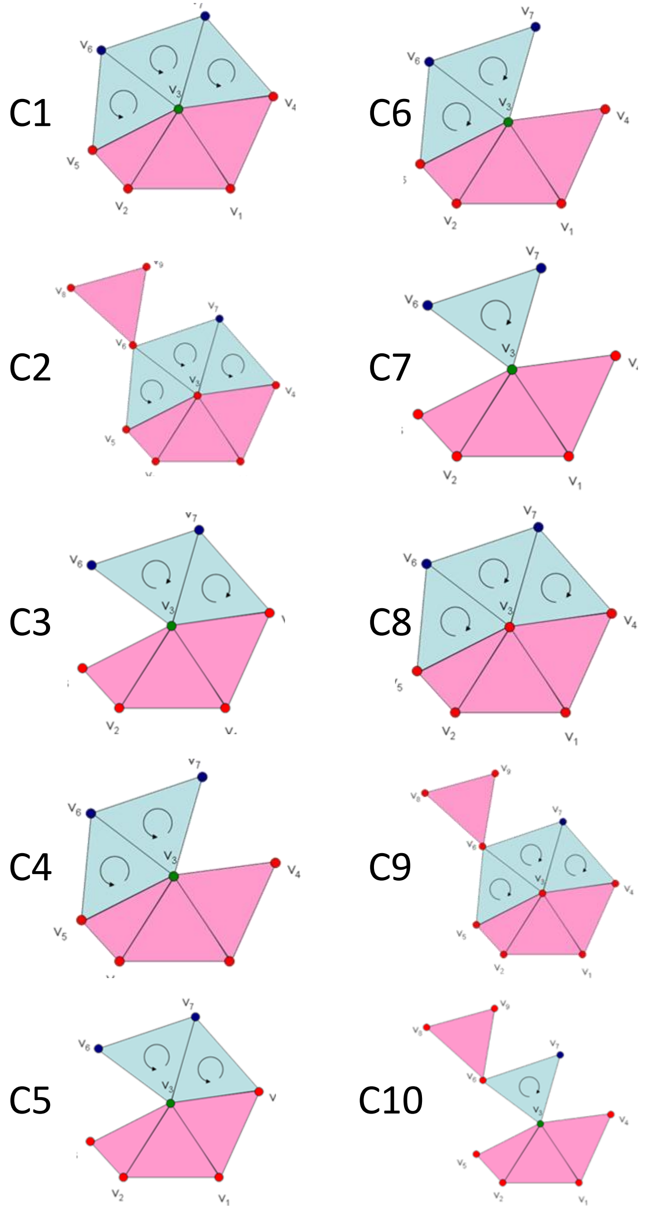
2,(6,2),(4,1) 0 0 0 0

CONNECTIVITY DECODING EXAMPLE



Neighbors list

{V10}

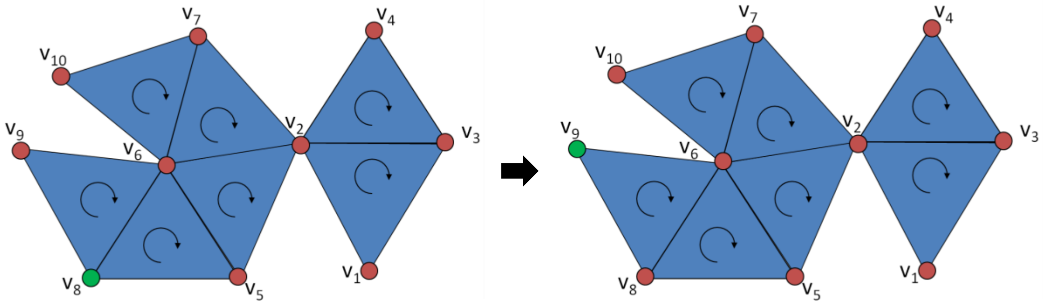


Bitstream

10 1,(7,1) 2,(4,1),(7,2) 0 0 1,(4, 1)

2,(6,2),(4,1) 0 0 0 0

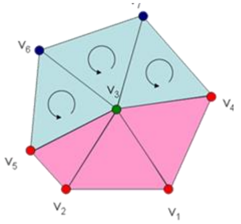
CONNECTIVITY DECODING EXAMPLE



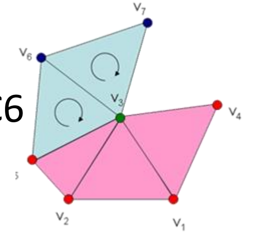
Neighbors list

{}

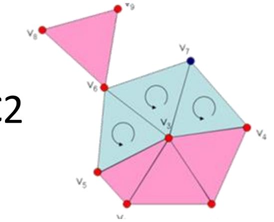
C1



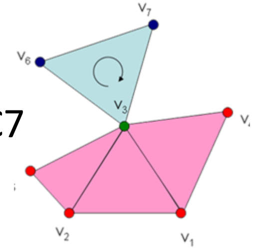
C6



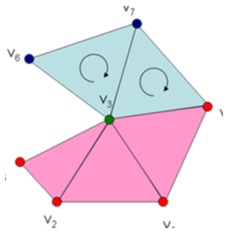
C2



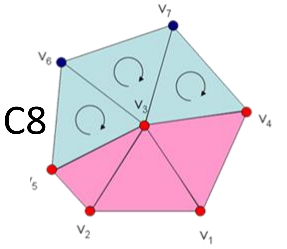
C7



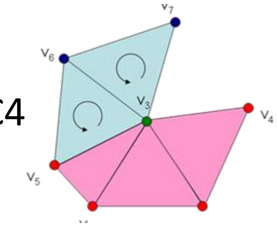
C3



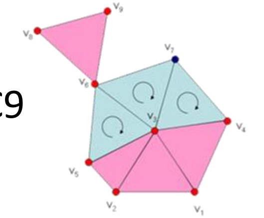
C8



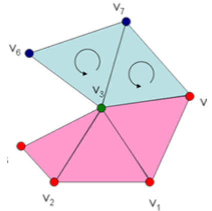
C4



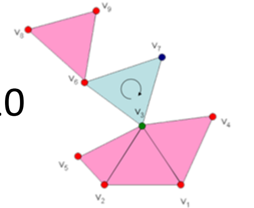
C9



C5



C10

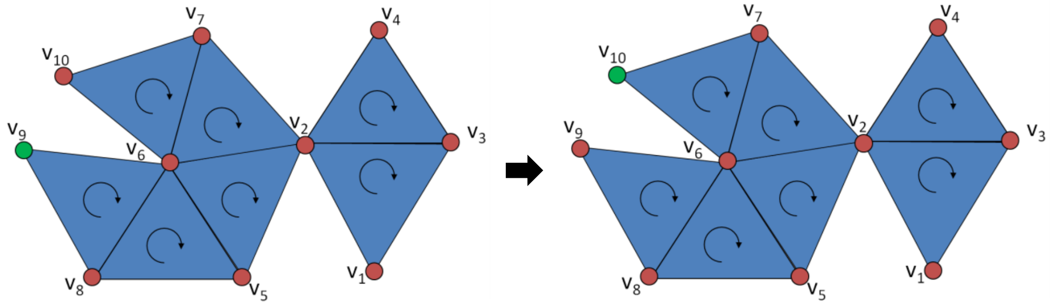


Bitstream

10 1,(7,1) 2,(4,1),(7,2) 0 0 1,(4, 1)

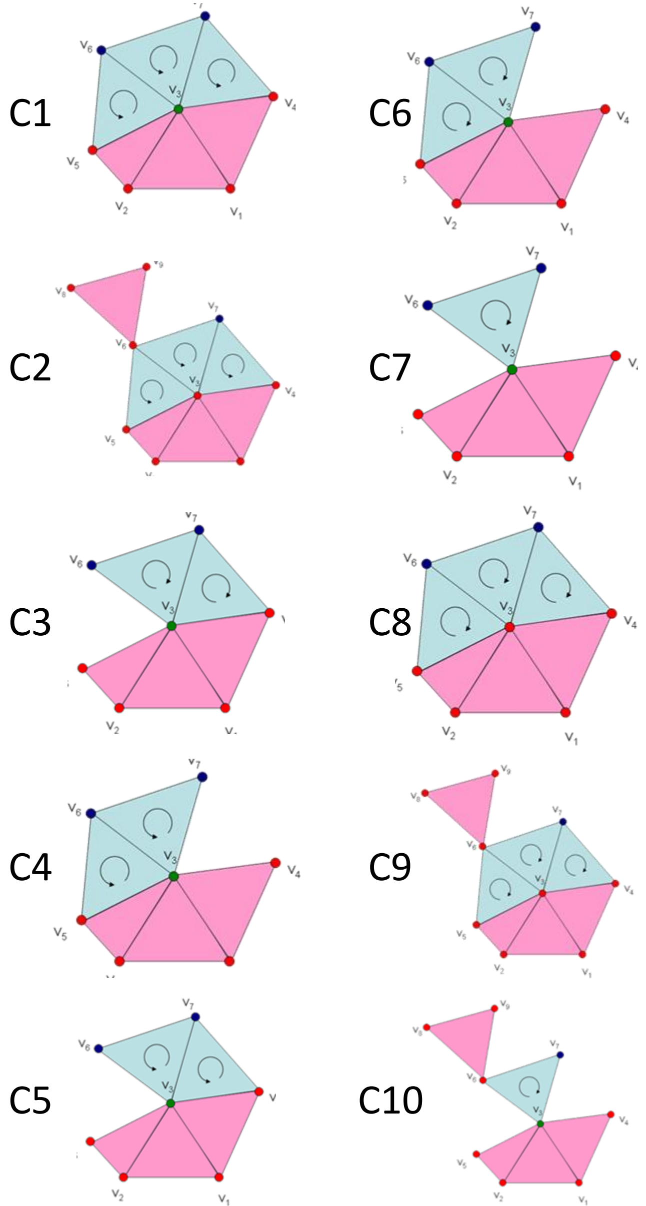
2,(6,2),(4,1) 0 0 0 0

CONNECTIVITY DECODING EXAMPLE



Neighbors list

{}



Bitstream

10 1,(7,1) 2,(4,1),(7,2) 0 0 1,(4, 1)

2,(6,2),(4,1) 0 0 0 0