# A Novel Fully Progressive Lossy-to-Lossless Coder for Arbitrarily-Connected Triangle-Mesh Models of Images and Other Bivariate Functions

Jiacheng Guo

University of Victoria

*jiacheng@uvic.com*

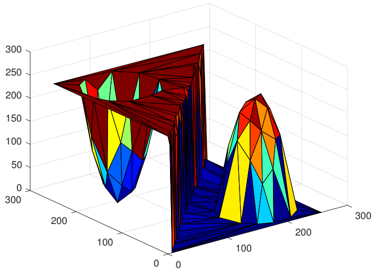
August 2, 2018

# Overview

1. Introduction and Background
   - Mesh Modeling of Bivariate Function
   - Mesh Coding
   - Background
2. Proposed Mesh-Coding Method
   - Bivariate-Function Description (BFD) Tree
   - Encoding Algorithm
   - Decoding Algorithm
3. Evaluation
   - Test Data
   - Evaluation of Proposed Mesh Coding Method
   - Evaluation of the Mesh-based Image Coding System
4. Conclusion

# Mesh Modeling of Bivariate Function

- one class of representation for bivariate functions
- constructed using a triangulation
- information in mesh
    - geometry information
    - connectivity information
    - function value information
- 2.5-D vs 3-D
- usually very large, compression needed



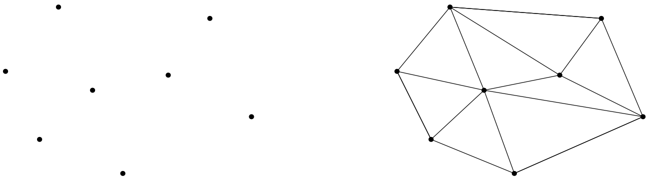
# Mesh Coding

- 2.5-D vs 3-D mesh coding method
  - 2.5-D meshes more constrained
  - 3-D mesh coder far from ideal
- progressive coding
  - no need to wait for entire bitstream
  - more bitstream received, better approximations obtained
- motivation
  - 3-D mesh coding methods: inefficient
  - 2.5-D mesh coding methods: cannot code meshes with arbitrary connectivity
  - objectives: 2.5-D mesh coder; progressive; and arbitrary connectivity.

## Triangulation

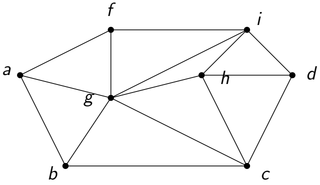
(Triangulation) A triangulation $T$ of the set $P$ of points in $\mathbb{R}^2$ is a set $T$ of non-degenerate triangles that satisfies the following conditions:

1. the union of all triangles in $T$ is the convex hull of $P$;
2. the set of the vertices in all triangles of $T$ is $P$; and
3. the interiors of any two triangles faces in $T$ do not intersect.



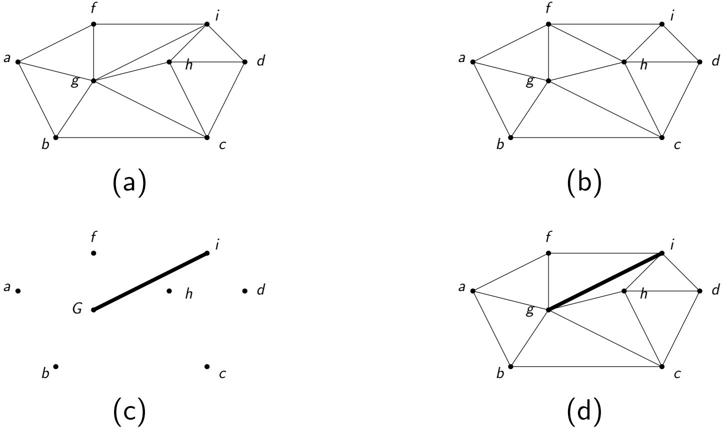
(a) (b)

# Coding Triangulation with Arbitrary Connectivity I



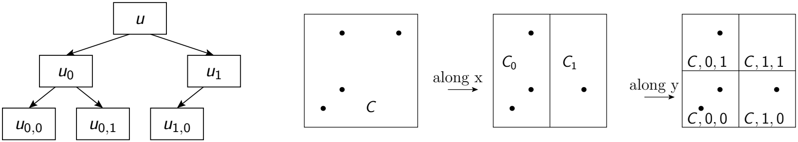

- naive way: code information of all edges
  - inefficient
- solution:
  - given a triangulation T of a set P of points
  - compute a triangulation $T_1$ of certain connectivity (preferred Delaunay triangulation)
  - find difference between $T$ and $T_1$, set $E$ as constrained edges
  - arbitrary connectivity to triangulation with edge-constraint

# Coding Triangulation with Arbitrary Connectivity II



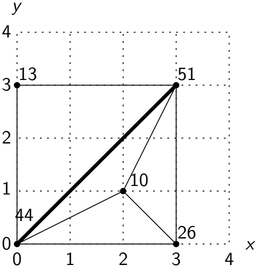

- $E$ is minimal set such that $\text{CPDT}(P, E)$ has same connectivity as $T$
- $E$ normally less than 25%, can not exceed 50%

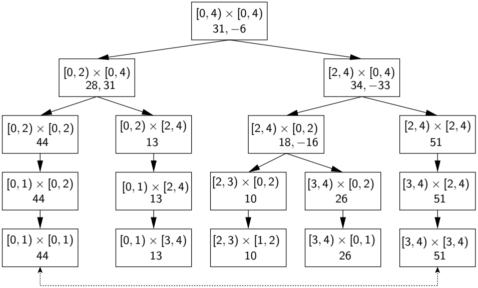
# Bivariate-Function Description (BFD) Tree I




- captures all of information needed to characterize a 2.5-D mesh dataset
- a cell is a rectangle region might or might not contain one or more sample points
- a cell is said to be occupied if it contains at least one sample point
- node is present only if its corresponding cell is occupied

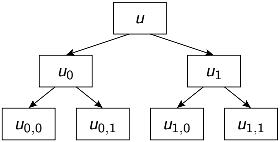
# Bivariate-Function Description (BFD) Tree II

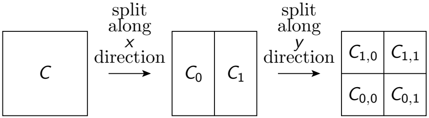



- geometry information:
  - corresponding cell
  - representation vertex: approximate sample point
- connectivity information:
  - leaf node contains node's set of constraint-connected neighbors
- function value information:
  - approximation coefficient: approximate function value
  - detail coefficient if has two children

# Child-configuration and edge-constraint (CCEC) Coding Procedure



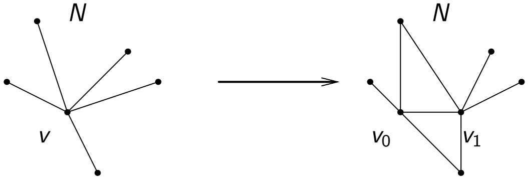

## Child-configuration coding

- for a given node, CCEC adds any children and grandchildren of this node
- 2 potential children, 4 potential grandchildren
- information to code:
  - a count $n$ of how many grandchildren present
  - which $n$ grandchildren present

# Child-configuration and edge-constraint (CCEC) Coding Procedure

## Edge-constraint Coding

- vertex split: add two child nodes $v_0$ and $v_1$ to node $v$
- let $N$ denote the set of constraint-connected neighbors of $v$.
- information to code for each $u \in N$:
  - whether $u$ is connected to both $v_0$ and $v_1$
  - if not, which of one of $v_0$ or $v_1$ is $u$ connected to.
- whether $v_0$ and $v_1$ constraint connected



# Detail-Coefficient Refinement (DCR) Coding Procedure

- only need to code detail-coefficient for node has two children
- function value, $n$-bit unsigned integer
- detail-coefficient, $(n+1)$-bit signed integer
- start from most significant magnitude bit
- each time code a magnitude bit, code sign bit after coding first nonzero magnitude bit

# Decoding Algorithm

- encoding and decoding mostly symmetric
- reconstruct tree progressively based on received information
- when decoding terminated:
  - current leaf nodes represented as their representative vertices
  - edge-constraints generated based on constraint-connected relationships of leaf nodes

# Test Data

- meshes generated from a variety of bivariate functions, which include images and elevation map
- 40 PD (preferred Delaunay) meshes and 58 non-PD (non-preferred Delaunay) meshes
- 2.5-D coders: ADIT, IT, and SDC. 3-D coders: Edgebreaker and Wavemesh.

| Name | V† | E† | F† | Fraction of Non-LPD† Edges (%) | Unpadded Bounding Box | Bits Per Sample | Type |
|---|---|---|---|---|---|---|---|
| kodim15@0.02 | 7864 | 23583 | 15720 | 20.5 | 768×512 | 8 | image |
| lena@0.03 | 7864 | 15578 | 23441 | 21.0 | 512×512 | 8 | image |
| ct@0.01 | 2621 | 7840 | 5220 | 19.9 | 512×512 | 12 | image |
| cr@0.005 | 17858 | 53419 | 35562 | 19.8 | 1744×2048 | 10 | image |
| muttart@0.0025 | 3637 | 10858 | 7222 | 21.7 | 1912×761 | 8 | image |
| question2@0.02 | 38400 | 115193 | 76794 | 19.8 | 1200×1600 | 8 | image |
| checkerboard@0.01 | 2621 | 7753 | 5133 | 19.1 | 512×512 | 8 | image |
| n27@0.0025 | 3606 | 10711 | 7106 | 13.9 | 1201×1201 | 13 | EM† |
| n35@0.0025 | 3606 | 10715 | 7110 | 13.8 | 1201×1201 | 12 | EM |
| n49@0.01 | 14424 | 43132 | 28709 | 13.0 | 1201×1201 | 12 | EM |
| animal@0.005 | 7397 | 22146 | 14750 | 0 | 1238×1195 | 8 | image |
| bull@0.02 | 15728 | 47073 | 31346 | 0 | 1024×768 | 8 | image |
| peppers@0.04 | 10485 | 31297 | 20813 | 0 | 512×512 | 8 | image |
| wheel@0.04 | 3451 | 10346 | 6896 | 0 | 512×512 | 8 | image |
| n45@0.01 | 14424 | 43036 | 28613 | 0 | 1201×1201 | 10 | EM |

# Lossless Coding Performance

Table: lossless results for non-PD meshes

| Mesh | Bit rate (bits/vertex) | | |
|---|---|---|---|
| | Proposed | Edgebreaker | Wavemesh |
| n49@0.01 | **21.30** | 25.90 | 34.39 |
| kodim15@0.02 | **18.32** | 21.67 | 27.71 |
| lena@0.03 | **17.92** | 20.80 | 25.65 |
| ct@0.01 | **22.90** | 25.52 | 36.36 |
| cr@0.005 | **21.16** | 24.29 | 29.02 |
| n27@0.0025 | **24.41** | 28.70 | 43.00 |
| n35@0.0025 | **23.18** | 26.64 | 35.52 |
| question2@0.02 | **15.42** | 18.35 | 23.00 |
| muttart@0.0025 | **22.78** | 31.24 | 43.74 |
| checkerboard@0.01 | **17.10** | 20.18 | 23.79 |

- outperforms Edgebreaker by margins of 2.04 to 12.05 bits/vertex
- outperforms Wavemesh by margins of 7.09 to 25.46 bits/vertex

# Lossless Coding Performance

Table: lossless results for PD meshes

| Mesh | Bit rate (bits/vertex) | | | | | |
|---|---|---|---|---|---|---|
| | Proposed | SDC | IT | ADIT | Edgebreaker | Wavemesh |
| animal@0.005 | 14.28 | **14.25** | 14.39 | 14.34 | 20.00 | 24.15 |
| bull@0.02 | **12.79** | 12.83 | 12.89 | 12.84 | 18.90 | 19.86 |
| peppers@0.04 | 13.15 | 13.18 | 13.19 | **13.11** | 18.94 | 20.85 |
| wheel@0.04 | 12.10 | **11.92** | 12.16 | 12.06 | 19.25 | 25.67 |
| n45@0.01 | **15.47** | 15.75 | 15.54 | 15.54 | 21.30 | 27.34 |

- ADIT, IT, and SDC can not code meshes with arbitrary connectivity
- comparable to ADIT, IT, and SDC; margin less than 0.1 bits/vertex

# Progressive Coding Performance

| Mesh | Rate (bytes) | PSNR (dB) | | | | |
|---|---|---|---|---|---|---|
| | | Proposed | SDC | IT | ADIT | Wavemesh |
| animal@0.05 | 1000 | **25.70** | 15.75 | 23.31 | 23.44 | 12.46 |
| | 4000 | **30.01** | 19.71 | 28.42 | 27.92 | 19.14 |
| | 7000 | **35.02** | 22.70 | 31.10 | 30.61 | 24.65 |
| | 10000 | **41.43** | 29.53 | 33.13 | 36.74 | 27.00 |
| | 13000 | **43.56** | 38.01 | 40.49 | 43.56 | 33.80 |
| bull@0.02 | 1100 | **25.23** | 13.34 | 23.52 | 23.67 | 15.55 |
| | 5600 | **29.66** | 18.83 | 27.57 | 28.23 | 24.23 |
| | 10100 | **31.50** | 22.38 | 30.19 | 30.64 | 27.03 |
| | 15600 | **36.50** | 25.94 | 32.52 | 34.02 | 30.69 |
| | 20600 | **42.08** | 30.99 | 36.28 | 40.25 | 34.88 |
| peppers@0.04 | 1000 | **21.02** | 11.62 | 19.69 | 19.68 | 10.58 |
| | 4900 | **24.46** | 13.02 | 22.58 | 23.02 | 17.51 |
| | 8500 | **26.77** | 17.77 | 24.98 | 25.29 | 19.18 |
| | 12400 | **32.93** | 22.66 | 26.37 | 30.23 | 21.75 |
| | 16300 | **34.35** | 28.11 | 29.70 | 34.25 | 26.02 |

- outperforms Wavemesh (7.08 to 13.28 dB)
- outperforms SDC (5.55 to 11.90 dB), IT (0.81 to 8.58 dB), ADIT (0 to 4.29 dB)

# Evaluation of the Mesh-based Image Coding System

- can be combined with a mesh generator to achieve a mesh-based image coding system
- also depends on performance of mesh generator

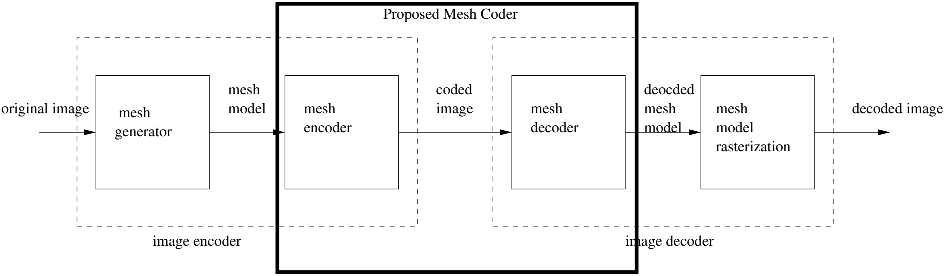


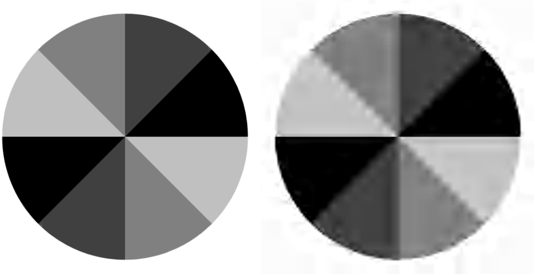

Figure: The image coding system consisting of the proposed coding method and a mesh generator

# Evaluation of the Mesh-based Image Coding System

| Image | Comp. Ratio | PSNR (dB) Mesh | PSNR (dB) JPEG 2000 |
|---|---|---|---|
| bull | 40:1 | 44.63 | **45.67** |
| | 100:1 | **42.51** | 41.20 |
| | 200:1 | **39.41** | 38.00 |
| | 250:1 | **39.08** | 37.22 |
| animal | 114:1 | **43.22** | 42.83 |
| | 241:1 | **41.84** | 40.28 |
| | 435:1 | **40.47** | 37.63 |
| | 526:1 | **39.84** | 35.68 |
| wheel | 71:1 | $\infty$ | 39.42 |
| | 121:1 | **52.20** | 34.22 |
| | 200:1 | **46.98** | 30.13 |
| | 250:1 | **39.63** | 28.30 |

- outperforms JPEG 2000 by average margin of 1.93 dB
- for image wheel: 11.33 to 17.98 dB

# Evaluation of the Mesh-based Image Coding System



(a) (b)

Figure: (a) proposed (39.63 dB) and (b) JPEG 2000 (28.30 dB) methods at compression ratio 250:1.

- outperforms JPEG 2000 for images nearly piecewise smooth
- JPEG 2000: very blurry

# Conclusion

- proposed a new progressive lossy-to-lossless coding method for 2.5-D meshes with arbitrary connectivity
- reduced connectivity coding cost by encoding edges not locally PD
- performance of proposed method:
  - outperforms 3-D coders in terms of lossless bit rate
  - outperforms both 2.5-D coders and 3-D coder in terms of progressive coding performance
  - can be combined with a mesh generator to form a mesh-based image coding system
  - mesh-based image coding system outperforms JPEG 2000 for images nearly piecewise smooth

# THANK YOU! Q&A