Patching Catmull-Clark Meshes

— completing quadrilateral meshes as smooth Nurbs surfaces —

Jörg Peters, University of Florida  http://www.cise.ufl.edu/~jorg
Patching Catmull-Clark Meshes

— completing quadrilateral meshes as smooth Nurbs surfaces —

Jörg Peters, University of Florida  http://www.cise.ufl.edu/~jorg
Patching Catmull-Clark Meshes

— completing quadrilateral meshes as smooth Nurbs surfaces —

Jörg Peters, University of Florida  http://www.cise.ufl.edu/~jorg
Patching Catmull-Clark Meshes

— completing quadrilateral meshes as smooth Nurbs surfaces —

Jörg Peters, University of Florida  http://www.cise.ufl.edu/~jorg
Patching Catmull-Clark Meshes

— completing quadrilateral meshes as smooth Nurbs surfaces —

Jörg Peters, University of Florida  http://www.cise.ufl.edu/~jorg
Patching Catmull-Clark Meshes

— completing quadrilateral meshes as smooth Nurbs surfaces —

+ separate irregular nodes

Jörg Peters, University of Florida  http://www.cise.ufl.edu/~jorg
Patching Catmull-Clark Meshes

— completing quadrilateral meshes as smooth Nurbs surfaces —

+ separate irregular nodes
+ distribute curvature or fit data

Jörg Peters, University of Florida  http://www.cise.ufl.edu/~jorg
Patching Catmull-Clark Meshes

— completing quadrilateral meshes as smooth Nurbs surfaces —

+ separate irregular nodes
+ distribute curvature or fit data
+ simpler than splines for smooth surfacing with irregular layout

Jörg Peters, University of Florida  http://www.cise.ufl.edu/~jorg
Patching Catmull-Clark Meshes

— completing quadrilateral meshes as smooth Nurbs surfaces —

+ separate irregular nodes  + distribute curvature or fit data

- FALSE: simpler than splines for smooth surfacing with irregular layout

Jörg Peters, University of Florida  http://www.cise.ufl.edu/~jorg
Patching Catmull-Clark Meshes

— completing quadrilateral meshes as smooth Nurbs surfaces —

+ separate irregular nodes
+ distribute curvature or fit data

little additional surface generated

Jörg Peters, University of Florida  http://www.cise.ufl.edu/~jorg
Patching Catmull-Clark Meshes

— completing quadrilateral meshes as smooth Nurbs surfaces —

+ separate irregular nodes  + distribute curvature

little additional surface generated
divergent curvature

Jörg Peters, University of Florida  http://www.cise.ufl.edu/~jorg
Patching Catmull-Clark Meshes

— completing quadrilateral meshes as smooth Nurbs surfaces —
(parametrized) refinement + finite (standard) representation

Jörg Peters, University of Florida  http://www.cise.ufl.edu/~jorg
Patching Catmull-Clark Meshes

— completing quadrilateral meshes as smooth Nurbs surfaces —
(parametrized) refinement + finite (standard) representation

- need both subdivision data structures and surface splines

Jörg Peters, University of Florida  http://www.cise.ufl.edu/~jorg
Patching Catmull-Clark Meshes

— completing quadrilateral meshes as smooth Nurbs surfaces —
(parametrized) refinement + finite (standard) representation

- FALSE: need both subdivision data structures and surface splines

Jörg Peters, University of Florida  http://www.cise.ufl.edu/~jorg
Examples
Overview

- Review of basics and literature
- Algorithm specification
- Discussion of the properties of output
- Array-based, permanent data structures
- Questions: creases, knot spacing
Quick Review: Nurbs

Non-uniform rational basic-spline patch \( Q(u, v) \in \mathbb{R}^3 \)
of order 4 (\( \equiv \) bicubic tensor-product spline)
is outlined by control net formed from \( k^2 \) control points \( Q_{uv} \in \mathbb{R}^3 \).
Quick Review: Nurbs

Non-uniform rational basic-spline patch \( Q(u, v) \in \mathbb{R}^3 \) of order 4 (\( \equiv \text{bicubic} \) tensor-product spline) is outlined by control net formed from \( k^2 \) control points \( Q_{uv} \in \mathbb{R}^3 \).

OpenGL

\[
gluNurbsSurface( \text{obj, uknotcount, *uknot, vknotcount, *vknot, ustride, vstride, *controlpoints, uorder, vorder, fromto})
\]

\[
gluNurbsSurface( \text{obj, k + 4, *uknot, k + 4, *vknot, 3, vstride, *Q_{uv}, 4, 4, \text{GL}_\text{MAP2_VERTEX}_3})
\]
**Knot insertion** for splines (increasing $k$):
subdivide parameter domain
 correspondingly refine or [sic] subdivide the control net.
**Knot insertion** for splines (increasing $k$): subdivide parameter domain correspondingly refine or [sic] *subdivide* the control net.

Catmull, Clark 1978:

\[
4F \leftarrow \begin{array}{ccc} 
1 & 1 & 1 \\
1 & 1 & 1 \\
\end{array} \quad 16E \leftarrow \begin{array}{ccc} 
1 & 1 & 1 \\
1 & 1 & 1 \\
\end{array} \\
\]

\[
\begin{array}{ccc} 
1 & 6 & 1 \\
64V \leftarrow 6 & 36 & 6 \\
1 & 6 & 1 \\
\end{array} \quad A = 4n^2 - 7n.
\]
What we have

Have:

- Catmull-Clark subdivision meshes [Catmull 78]
What we have

Have:
- Catmull-Clark subdivision meshes [Catmull 78]
- apply local, global shape modification to CC [DeRose et al 98, Halstead et al. 93]
What we have

Have:
- Catmull-Clark subdivision meshes [Catmull 78]
- apply local, global shape modification to CC [DeRose et al 98, Halstead et al. 93]
- direct evaluation via eigendecomposition [Stam 98]
What we have

Have:

- Catmull-Clark subdivision meshes [Catmull 78]
- apply local, global shape modification to CC [DeRose et al 98, Halstead et al. 93]
- direct evaluation via eigendecomposition [Stam 98]
- analysis [Sabin 78, Ball & Storry 88, Peters & Reif 98].
... and what we might want

Have:
- Catmull-Clark subdivision meshes [Catmull 78]
- apply local, global shape modification to CC [DeRose et al 98, Halstead et al. 93]
- direct evaluation via eigendecomposition [Stam 98]
- analysis [Sabin 78, Ball & Storry 88, Peters & Reif 98].

Would be nice:
- At any subdivision step: apply a simple transformation to get
- a compact, explicit surface representation:
... and what we might want

Have:
- Catmull-Clark subdivision meshes [Catmull 78]
- apply local, global shape modification to CC [DeRose et al. 98, Halstead et al. 93]
- direct evaluation via eigendecomposition [Stam 98]
- analysis [Sabin 78, Ball & Storry 88, Peters & Reif 98].

Would be nice:
- At any subdivision step: apply a simple transformation to get a compact, explicit surface representation:
- maximally large, standard spline (Nurbs) patches
- join as smoothly and largely agree with the Catmull-Clark limit surface.

PCCM Patching Catmull-Clark Meshes
Quick review: Nurbs and smooth surfaces

PCCM is new. 1 patch per quad

- DeRose, Kass and Truong 98 [Catmull-Clark ?78] for rendering. Many small patches (glMap2); not finite.
Nurbs and smooth surfaces

PCCM is new. 1 patch per quad

- DeRose, Kass and Truong 98 [Catmull-Clark ?78] for rendering. Many small patches (glMap2); not finite.

- Prautzsch 97 filling $n$-sided holes; curvature continuous! requires order 7 patches; 9 times as many patches as PCCM.
Nurbs and smooth surfaces

PCCM is new. 1 patch per quad

- DeRose, Kass and Truong 98 [Catmull-Clark ‘78] for rendering. Many small patches (glMap2); not finite.

- Prautzsch 97 filling $n$-sided holes; curvature continuous! Requires order 7 patches; 9 times as many patches as PCCM.

- Grimm and Hughes 95 subdivision as a preprocessing. High degree HKS rational patches. 9 times as many patches as PCCM.
PCCM is new. 1 patch per quad

- DeRose, Kass and Truong 98 [Catmull-Clark ?78] for rendering. Many small patches (glMap2); not finite.

- Prautzsch 97 filling $n$-sided holes; curvature continuous! requires order 7 patches; 9 times as many patches as PCCM.

- Grimm and Hughes 95 subdivision as a preprocessing. High degree HKS rational patches. 9 times as many patches as PCCM.

- **Peters 92** parametrized subdivision as a preprocessing for $C^1$ free-form surfacing. 16 times as many bicubic Bézier patches as PCCM.
Overview

- Review of basics and literature
- Algorithm specification
- Discussion of the properties of output
- Array-based, permanent data structures
- Questions: creases, knot spacing
Corner point is placed directly on the Catmull-Clark limit surface:

\[ Q_{00}(1) = \ldots = Q_{00}(n) = \frac{\sum nP_{00}(i) + 4P_{30}(i) + P_{33}(i)}{n(n + 5)}. \]

Normal is free to choose, eg. as normal of Catmull-Clark limit surface.
PCCM: (2) Corner Smoothing

Nurbs patches $C^0$ at EON; $C^2$ everywhere else.
Nurbs patches $C^0$ at EON; $C^2$ everywhere else.

Collect $\mathbf{Q}_{uv}$ for $uv \in \{00, 10, 20, 40\}$.
Nurbs patches $C^0$ at EON; $C^2$ everywhere else.

Collect $\bar{Q}_{uv}$ for $uv \in \{00, 10, 20, 40\}$

Use $n \times n$ matrices $A_n$ and $B_n$
\[ Q_{10} = Q_{00} + A_n \bar{Q}_{10} \quad \ast \]
\[ Q_{20} = (Q_{40} + 6Q_{10} - 2Q_{00})/5 \]
\[ Q_{11} = B_n \left( Q_{10} + \frac{\cos(2\pi/n)}{6}(Q_{40} - Q_{20}) \right) \]
\[ Q_{10} = Q_{00} + \alpha A_n P_{30} + \beta (A_n + A_n^+) P_{33} \]
\[ Q_{20} = (Q_{40} + 6Q_{10} - 2Q_{00})/5 \]
\[ Q_{11} = B_n \left( Q_{10} + \frac{\cos(2\pi/n)}{6} (Q_{40} - Q_{20}) \right) \]
Only if \( n \) is even and greater than 4, 
\[
q = \sum_{i=1}^{n} (-1)^i \overline{Q}_{40}(i)/n
\]
and if \( q \neq 0 \) add \( h_i = (-1)^i r \) to \( Q_{40}(i) \),
\( Q_{41}(i) \), \( Q_{14}(i - 1) \).

\[
Q_{10} = Q_{00} + \alpha A_n P_{30} + \beta (A_n + A_n^+) P_{33}
\]
\[
Q_{20} = (Q_{40} + 6Q_{10} - 2Q_{00})/5
\]
\[
Q_{11} = B_n \left( Q_{10} + \frac{\cos(2\pi/n)}{6}(Q_{40} - Q_{20}) \right)
\]

For \( i = 1, \ldots, n \),

\( Q_{v0}(i + 1) = Q_{0v}(i) \) for \( v \in \{1, 2, 4\} \) and

add \( Q_{20}(i) - \overline{Q}_{20}(i) \) to \( Q_{21}(i) \) and \( Q_{12}(i - 1) \).
From Mesh to Surface
Properties of output

- Maximally large Nurbs patches.
  1 patch per quad independent of subdivision level!
Properties of output

- Maximally large Nurbs patches.
- Nurbs of standard order 4 (degree 3), in interpolating form with 4-fold knots.
Properties of output

- Maximally large Nurbs patches.

- Nurbs of standard order 4 (degree 3), in interpolating form with 4-fold knots.

- The Nurbs patches differ from the limit surface of the Catmull-Clark subdivision only near the EONs. Interpolate position and normal of CC (Nurbs have *finite curvature*, Catmull-Clark limit surface infinite.)
Properties of output

- Maximally large Nurbs patches.

- Nurbs of standard order 4 (degree 3), in interpolating form with 4-fold knots.

- The Nurbs patches differ from the limit surface of the Catmull-Clark subdivision only near the EONs. Interpolate position and normal of CC (Nurbs have \textit{finite curvature}, Catmull-Clark limit surface infinite.)

Difference:
Generically only in \textit{bi-3 polynomial corner pieces} of NURBS at EON; clamped: \textit{positions and tangents agree} with Catmull-Clark at boundaries and EON!
Difference shrinks like \( O(h^5) \)
Properties of output

- Maximally large Nurbs patches.
- Nurbs of standard order 4 (degree 3), in interpolating form with 4-fold knots.
- The Nurbs patches differ from the limit surface of the Catmull-Clark subdivision only near the EONs. Interpolate position and normal of CC.

Higher-order saddle points when \( n \geq 6 \) is even:
Quadratic boundary segments result in unnecessarily flat sections. Fix: adjust second layer, not layer adjacent to EON.
Properties of output

- Maximally large Nurbs patches.
- Nurbs of standard order 4 (degree 3), in interpolating form with 4-fold knots.
- The Nurbs patches differ from the limit surface of the Catmull-Clark subdivision only near the EONs. Interpolate position and normal of CC
- Higher-order saddle points when $n \geq 6$ is even: adjust second layer, not layer adjacent to EON.
- $C^2$ almost everywhere, tangent continuous near the EONs.
Properties of output

- Maximally large Nurbs patches.

- Nurbs of standard order 4 (degree 3), in interpolating form with 4-fold knots.

- The Nurbs patches differ from the limit surface of the Catmull-Clark subdivision only near the EONs. Interpolate position and normal of CC higher-order saddle points when \( n \geq 6 \) is even: adjust second layer, not layer adjacent to EON.

- \( C^2 \) almost everywhere, tangent continuous near the EONs.

- Nurbs patches, Catmull-Clark subdivision and the PCCM algorithm use the same *array-based data structures*. 
Overview

- Review of basics and literature
- Algorithm specification
- Discussion of the properties of output
- Array-based, permanent data structures
- Questions: creases, knot spacing
Array-based permanent data structures

Simple algorithm, simple data structures

quad_mipmap

quad_pccm
Array-based permanent data structures

Simple algorithm, simple data structures

1. For each quad:
   - Catmull-Clark: *mipmap* of $k + 2$ by $k + 2$ arrays $x, y, z$ node positions
   - PCCM: $k + 4$ by $k + 4$. Entry 00 = corner coefficient.

2. For each EON, access to adjacent quad corners (== B-rep of quads)
To create the mipmap level $\ell + 1$ from level $\ell$.

a. For each quad: apply \textit{B-spline subdivision rules}.
To create the mipmap level $\ell + 1$ from level $\ell$.

a. For each quad: apply $B$-spline subdivision rules.

b. For each EON:
   - collect $\circ$ at level $\ell$, compute $CC$, distribute $\bullet$ to level $\ell + 1$. 
Array-based permanent data structures: PCCM

quad_mipmap

quad_pccm

a. For each quad: apply *Knot Insertion*.

b. For each EON:
   - **collect** $Q_{00}(1)$ and $\overline{Q}_{uv}(i)$, $uv \in \{10, 20, 40\}$.
   - **Compute** $Q_{uv}(i)$, $uv \in \{10, 20, 11\}$, $Q_{20} - \overline{Q}_{20}$ and possibly $Q_{40} - \overline{Q}_{40}$.
   - **Distribute** $Q_{uv}(i)$, $uv \in \{10, 01, 20, 02, 11\}$ and add to $\{21, 12\}$ and possibly $\{04, 40, 14, 41\}$.
Arra y-based permanent data structures

Minimize connectivity and dependence

- CC: replicated points at edges is numerically acceptable. Nurbs subdivision rules divide by multiples of 2. EON only computed once.
Array-based permanent data structures

Minimize connectivity and dependence

- CC: replicated points at edges is numerically acceptable. Nurbs subdivision rules divide by multiples of 2. EON only computed once.

+ All space for subdivision level $\ell$ can be allocated at the outset (no additional B-rep is generated) and
Array-based permanent data structures

Minimize connectivity and dependence

- CC: replicated points at edges is numerically acceptable. Nurbs subdivision rules divide by multiples of 2. EON only computed once.

+ All space for subdivision level $\ell$ can be allocated at the outset (no additional B-rep is generated) and

+ the B-rep (connectivity) remains unchanged throughout.
Minimize connectivity and dependence

- CC: replicated points at edges is numerically acceptable. Nurbs subdivision rules divide by multiples of 2. EON only computed once.

+ All space for subdivision level $\ell$ can be *allocated at the outset* (no additional B-rep is generated) and

+ the *B-rep (connectivity) remains unchanged* throughout.

+ The quad-arrays can be input directly to *gluNurbsSurface* or displayed as quad-meshes.
Creases in Array-based permanent data structures

Creases in Array-based permanent data structures


Array of level $\ell$ captures the blend ratios (or smoothed creases) of the Catmull-Clark mesh up to level $\ell$.

*Implementation*:
— 8 additional numbers per array and
— one additional pass along the boundary of the array.
Overview

- Review of basics and literature
- Algorithm specification
- Discussion of the properties of output
- Array-based, permanent data structures
- Odds and Ends
**Unequal Knot spacing**: Knot spacings of the Catmull-Clark mesh $P$ may be chosen non-uniformly. This changes Corner Smoothing. Adjusting knot spacings yields a *second way to introduce creases* that Catmull-Clark does not offer!
Unequal Knot spacing: 

Knot spacings of the Catmull-Clark mesh $P$ may be chosen non-uniformly. This changes Corner Smoothing. Adjusting knot spacings yields a second way to introduce creases that Catmull-Clark does not offer!

+ **Hierarchical structure**
  — captured by mipmap (displacement + Corner Smoothing)
  — can add hierarchical B-splines.
**Odds and Ends**

+ *Unequal Knot spacing:* Knot spacings of the Catmull-Clark mesh \( P \) may be chosen non-uniformly. This changes Corner Smoothing. Adjusting knot spacings yields a *second way to introduce creases* that Catmull-Clark does not offer!

+ *Hierarchical structure*
  — captured by mipmap (displacement + Corner Smoothing) and
  — hierarchical B-splines

- *Patching Loop Meshes:* [Peters 200x] no additional B-rep needed, maximal triangular Bézier patches output.
+ **Unequal Knot spacing:** Knot spacings of the Catmull-Clark mesh \( P \) may be chosen non-uniformly. This changes Corner Smoothing. Adjusting knot spacings yields a *second way to introduce creases* that Catmull-Clark does not offer!

+ **Hierarchical structure**
  — captured by mipmap (displacement + Corner Smoothing) and
  — hierarchical B-splines

- **Patching Loop Meshes:** [Peters 200x] no additional B-rep needed, maximal triangular Bézier patches output.

- **Coming soon:** *curvature continuity* at EON.
  (Needs degree \( \geq 4 \))
Summary

PCCM:

✓ converts Catmull-Clark meshes to closed-form, smoothly-connected, standard Nurbs patches,
Summary

PCCM:

✓ converts Catmull-Clark meshes to closed-form, smoothly-connected, standard Nurbs patches,

✓ with simple, explicit formulas,
Summary

PCCM:

✓ converts Catmull-Clark meshes to closed-form, smoothly-connected, standard Nurbs patches,

✓ with simple, explicit formulas,

✓ integrates seamlessly with the array-based view of subdivision
Summary

PCCM:

✓ converts Catmull-Clark meshes to closed-form, smoothly-connected, standard Nurbs patches,

✓ with simple, explicit formulas,

✓ integrates seamlessly with the array-based view of subdivision

✓ remains local so that almost all patch transitions across patch boundaries are parametrically $C^2$. 
Summary

PCCM:

✓ converts Catmull-Clark meshes to closed-form, smoothly-connected, standard Nurbs patches,

✓ with simple, explicit formulas,

✓ integrates seamlessly with the array-based view of subdivision

✓ remains local so that almost all patch transitions across patch boundaries are parametrically $C^2$.

Smooth Nurbs free-form surfaces are easy to implement!
PCCM:

✓ converts Catmull-Clark meshes to closed-form, smoothly-connected, standard Nurbs patches,

✓ with simple, explicit formulas,

✓ integrates seamlessly with the array-based view of subdivision

✓ remains local so that almost all patch transitions across patch boundaries are parametrically $C^2$.

Smooth Nurbs free-form surfaces are easy to implement!
For the user: all mesh points are free to move!
PCCM:

✓ converts Catmull-Clark meshes to closed-form, smoothly-connected, standard Nurbs patches,

✓ with simple, explicit formulas,

✓ integrates seamlessly with the array-based view of subdivision

✓ remains local so that almost all patch transitions across patch boundaries are parametrically $C^2$.

Thanks to the referees, Henry Moreton, David Lutterkort, Malcolm Sabin, Andy Shiue and Georg Umlauf at SurfLab for their constructive comments.
Summary

PCCM:

✓ converts Catmull-Clark meshes to closed-form, smoothly-connected, standard Nurbs patches,

✓ with simple, explicit formulas,

✓ integrates seamlessly with the array-based view of subdivision

✓ remains local so that almost all patch transitions across patch boundaries are parametrically $C^2$.

Thanks to the referees, Henry Moreton, David Lutterkort, Malcolm Sabin, Andy Shiue and Georg Umlauf at SurfLab for their constructive comments.

THANK YOU