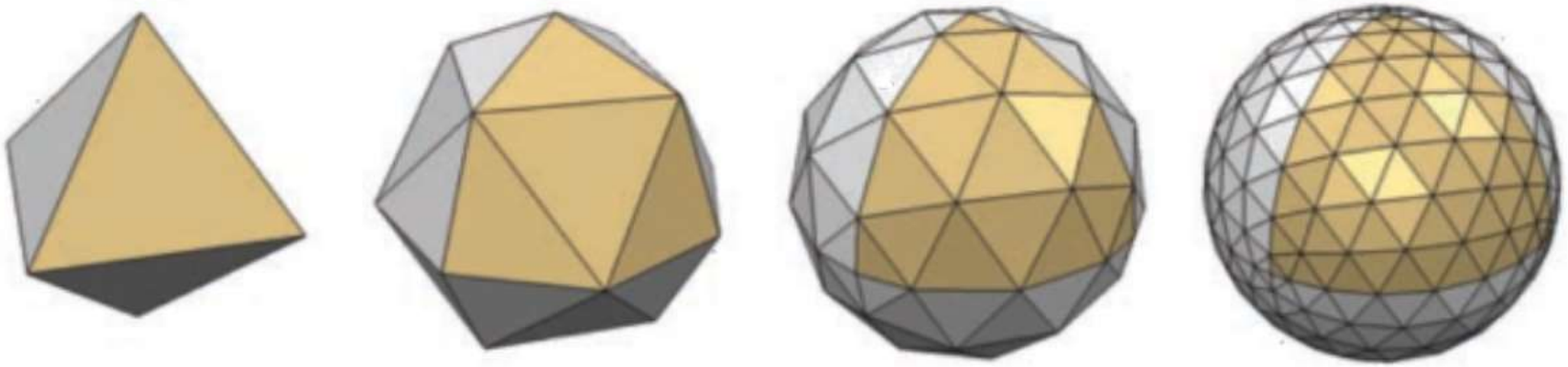
Procesamiento Geométrico y Análisis de Formas

Ivan Sipiran

Anteriormente

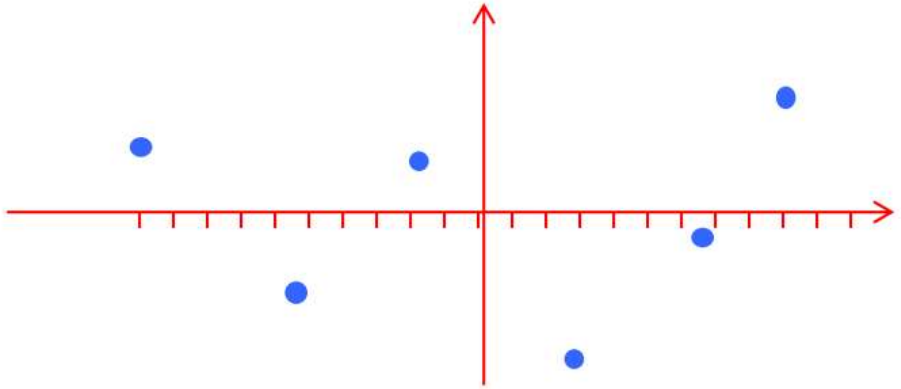
Ahora

- Representación de superficies con mallas triangulares
- Propiedades combinatorias
- Simplificación de mallas
- Subdivisión de mallas



Motivación: Aproximación de curvas

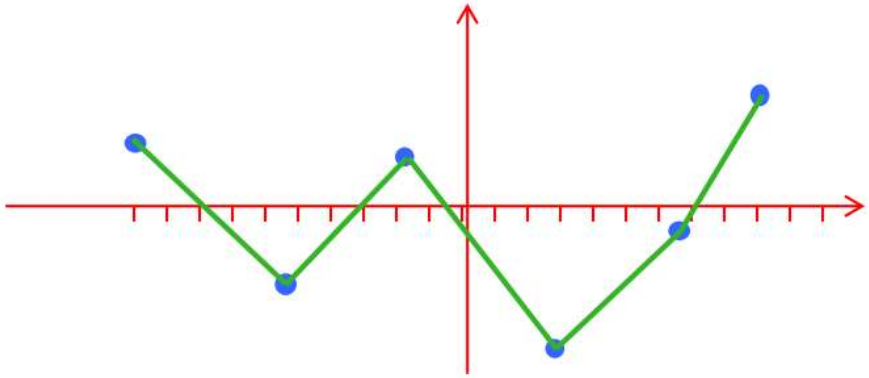
- En 1D
  - Dado un conjunto de pares de puntos $\{x_i, y_i\}$, aproximar la función f sujeta a $f(x_i) = y_i \ \forall i$



Nube de puntos

Motivación: Aproximación de curvas

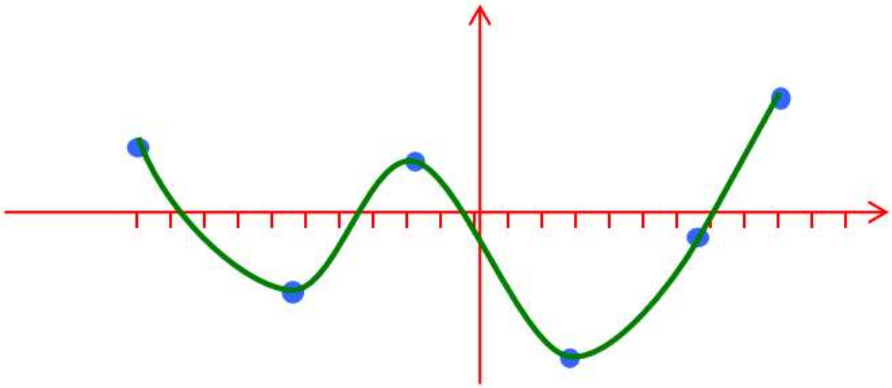
- En 1D
  - Dado un conjunto de pares de puntos $\{x_i, y_i\}$, aproximar la función f sujeta a $f(x_i) = y_i \ \forall i$



  - Solución más simple
    - Interpolación lineal

Motivación: Aproximación de curvas

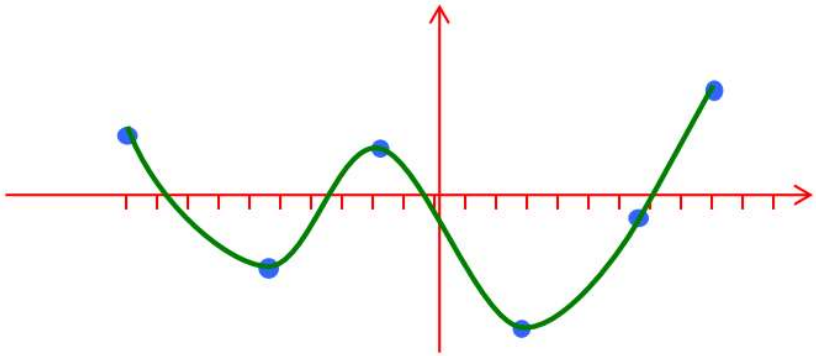
- En 1D
  - Dado un conjunto de pares de puntos $\{x_i, y_i\}$, aproximar la función f sujeta a $f(x_i) = y_i \ \forall i$



  - Solución suave
    - Interpolación de alto orden: polinomio de grado d

Motivación: Aproximación de curvas

- En 1D
  - Dado un conjunto de pares de puntos $\{x_i, y_i\}$, aproximar la función f sujeta a $f(x_i) = y_i \; \forall i$

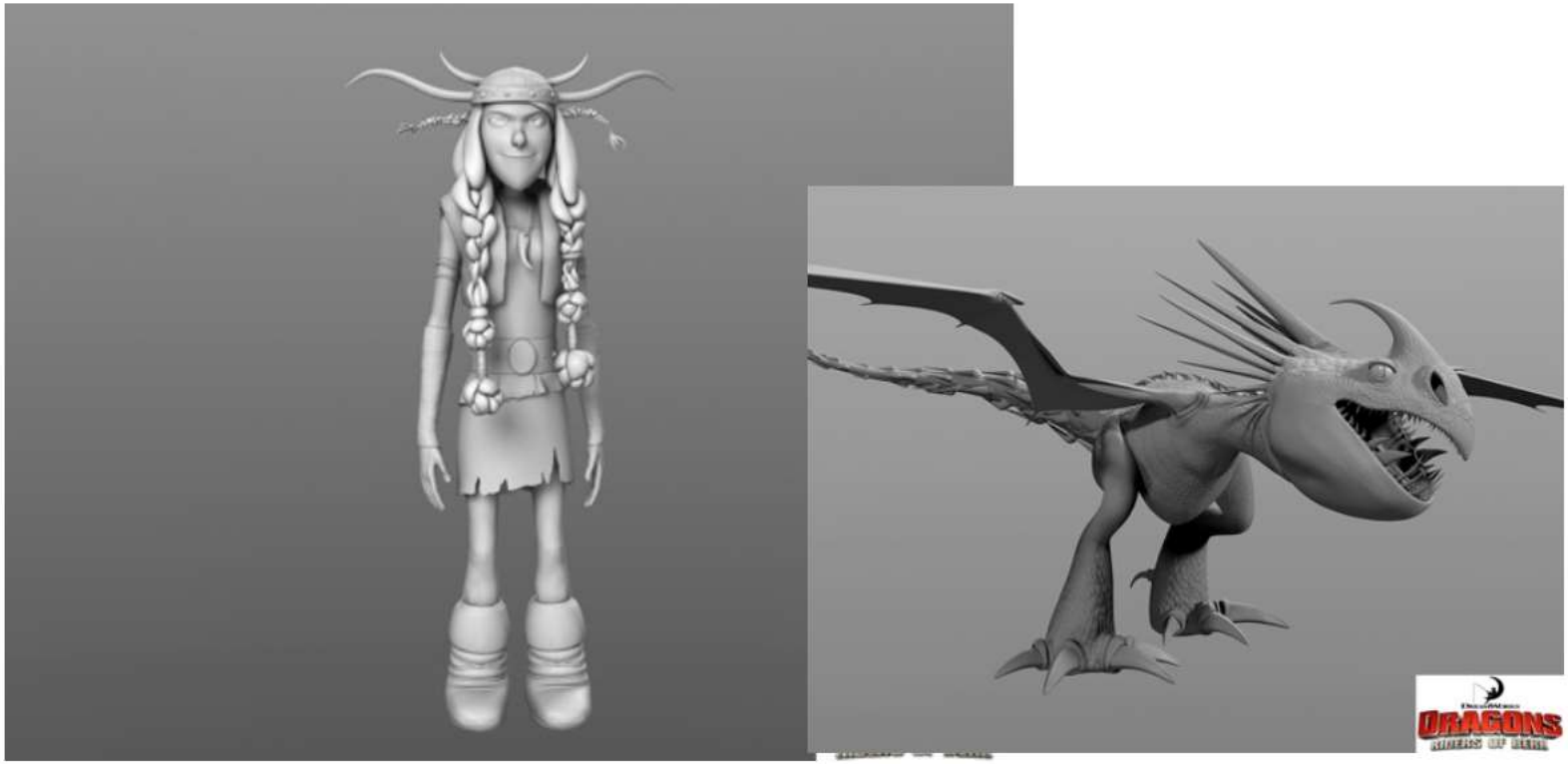


  - Teorema generalizado de valor medio: si f es un polinomio de grado p, entonces el error de aproximación es

$$|f(t) - g(t)| \leq \frac{1}{(p+1)!} \max f^{(p+1)} \prod_{i=0}^{p} (x_i - x) = O(h^{p+1})$$

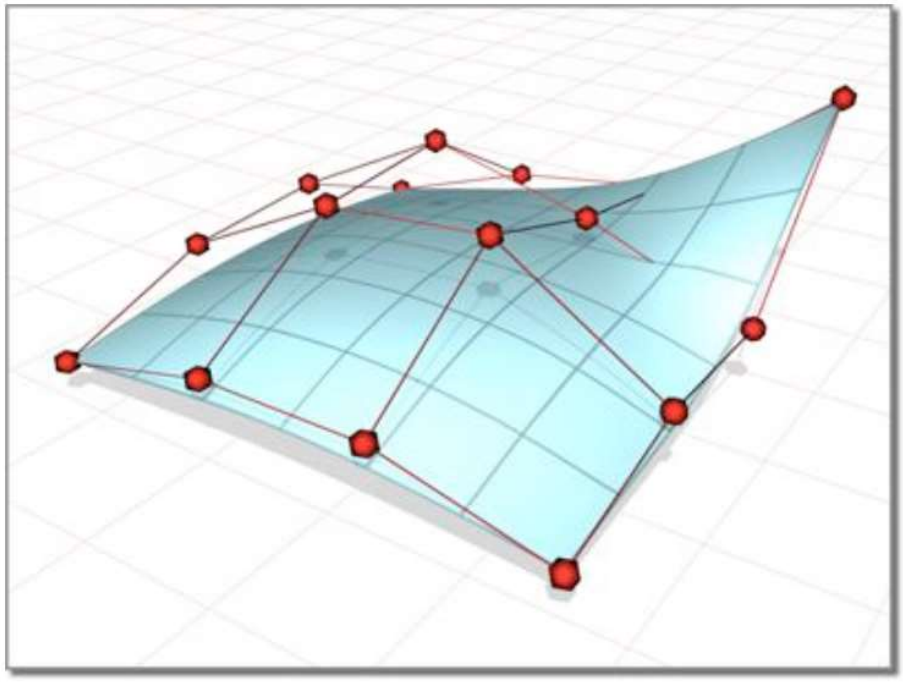
Motivación: Superficies NURBS

- En 3D
  - Aproximaciones de alto orden a superficies



Motivación: Superficies NURBS

- En 3D
  - Aproximaciones de alto orden a superficies

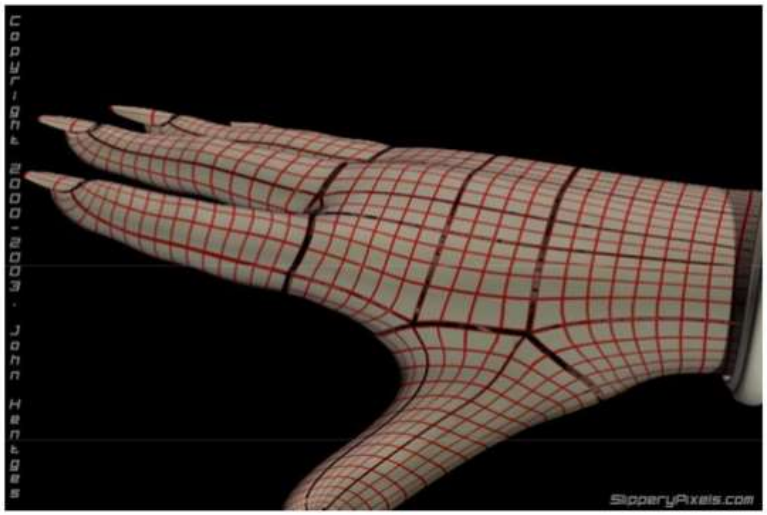


Definida via una latice de control con polígonos de control

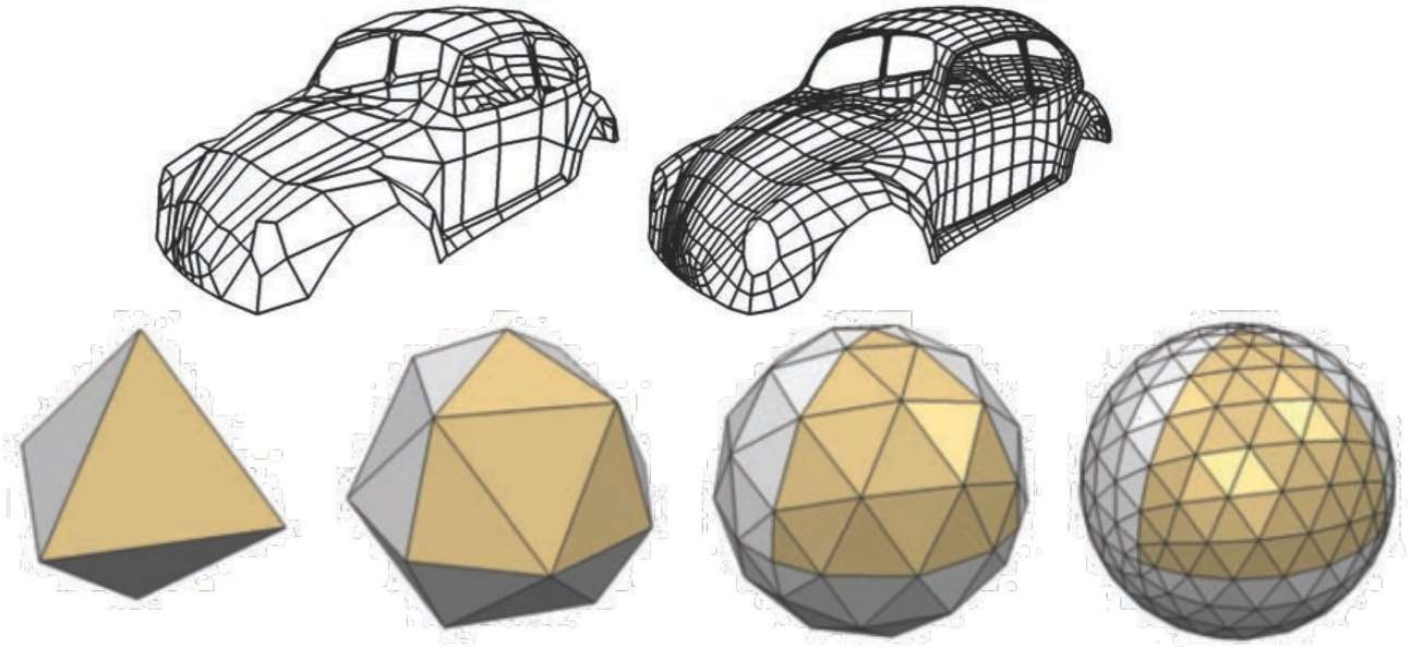
Motivación: Superficies NURBS

- NURBS
  - Inherentemente continuas
  - Control intuitivo
  - Limitado a dominios en grilla
  - Una NURB tiene la topología de una hoja, cilindro o toro

- Debes usar múltiples parches para representar formas complejas
- Ocurren artefactos después de aplicar deformaciones



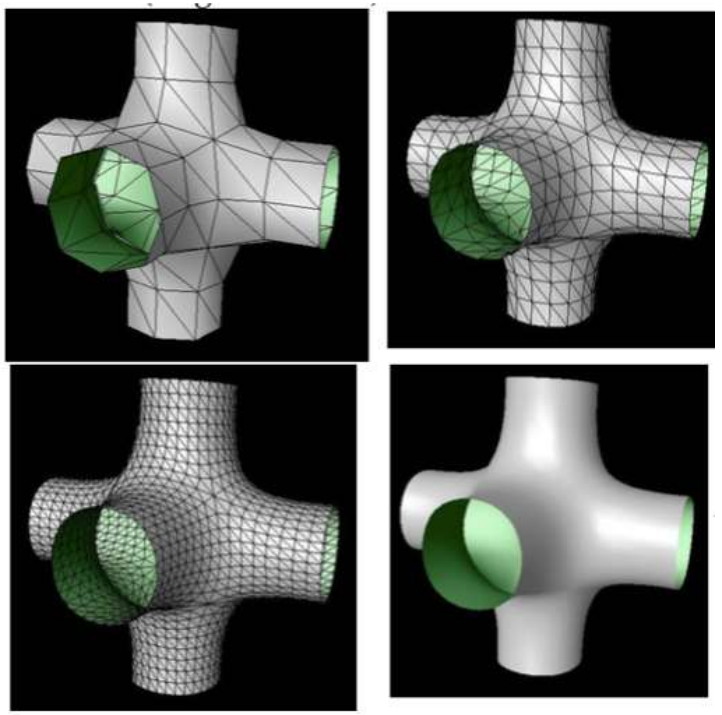
Motivación: Superficies NURBS



Superficie representada como una colección de vértices, aristas y caras

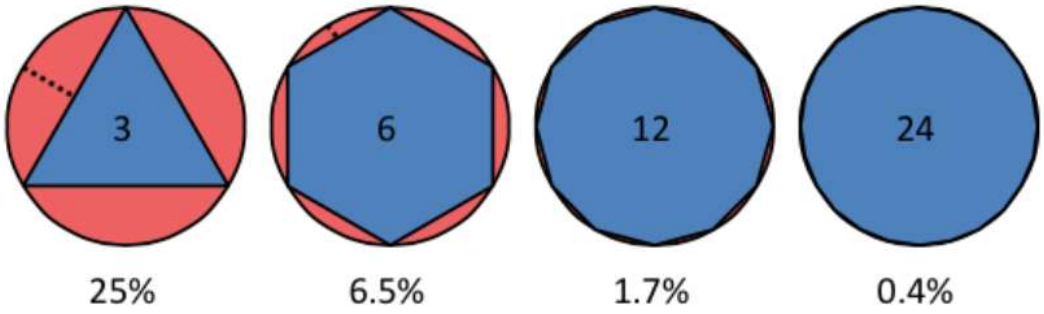
NURBS vs. Mallas triangulares

- Mallas triangulares
  - Inherentemente discretas
  - No es necesario reglas especiales para unir parches
  - Puedes modelar formas con topología arbitraria
  - Puedes añadir la resolución que sea necesaria
  - Permite subdivisión para obtener suavidad
  - Fácil de renderizar

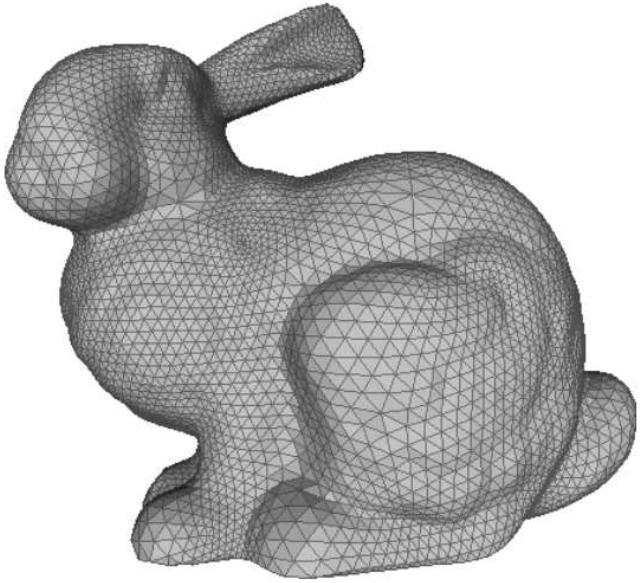


Porqué mallas triangulares

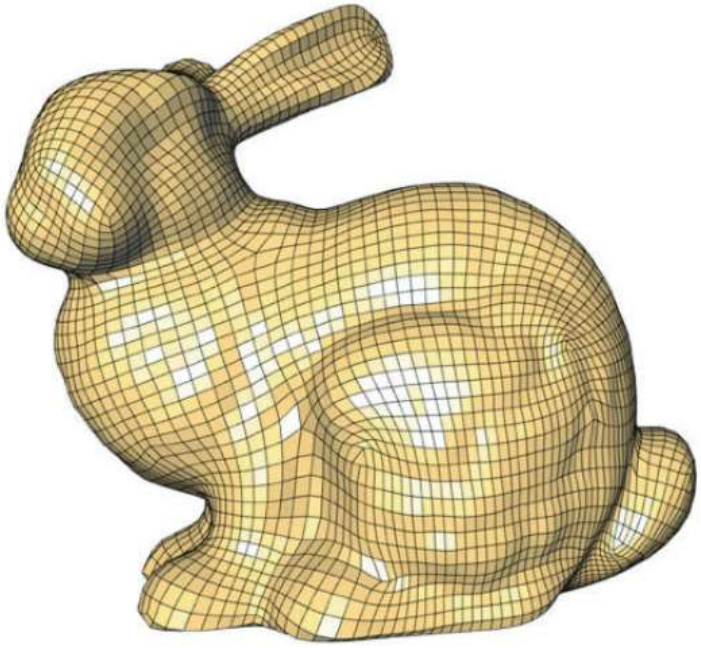
- Proveen aproximación lineal por piezas a la superficie
  - Error es $O(h^2)$
- Doblando el número de vértices, reduces el error en 4 veces



Porqué triángulos?

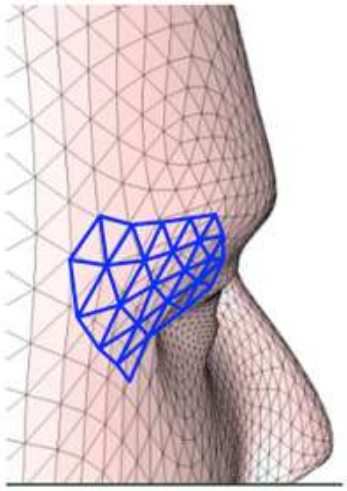


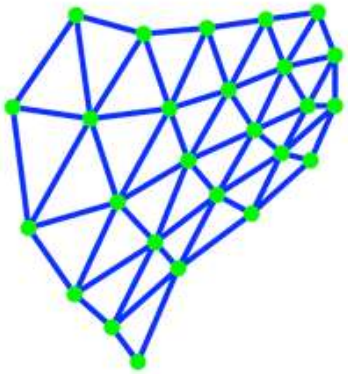
- Elemento lineal más simple
- Fácil de reconstruir desde nube de puntos
- Fácil de representar



- Quad-meses usadas en animación
- Requieren intervención humana para reconstrucción
- Son flexibles a deformación

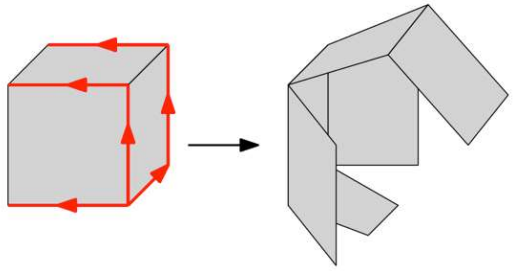
Mallas triangulares

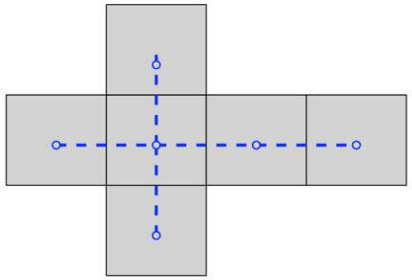




Superficie: conjunto de vértices, aristas y caras que definen una superficie polihedro en 3D (aproximación discreta de una forma)

Estructura combinatoria + geometría

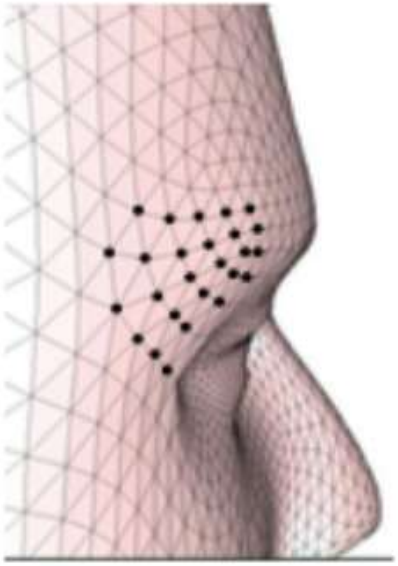




Porqué mallas triangulares

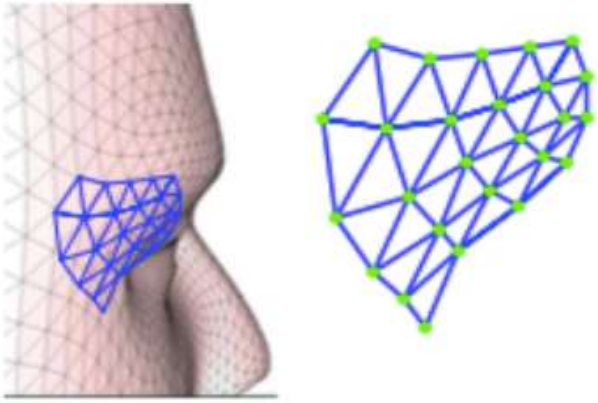
- Estructura geométrica + Estructura combinatoria

Geometría



Coordenadas de vértices

Conectividad: triangulación



Simplificación de mallas

Simplificación de Mallas

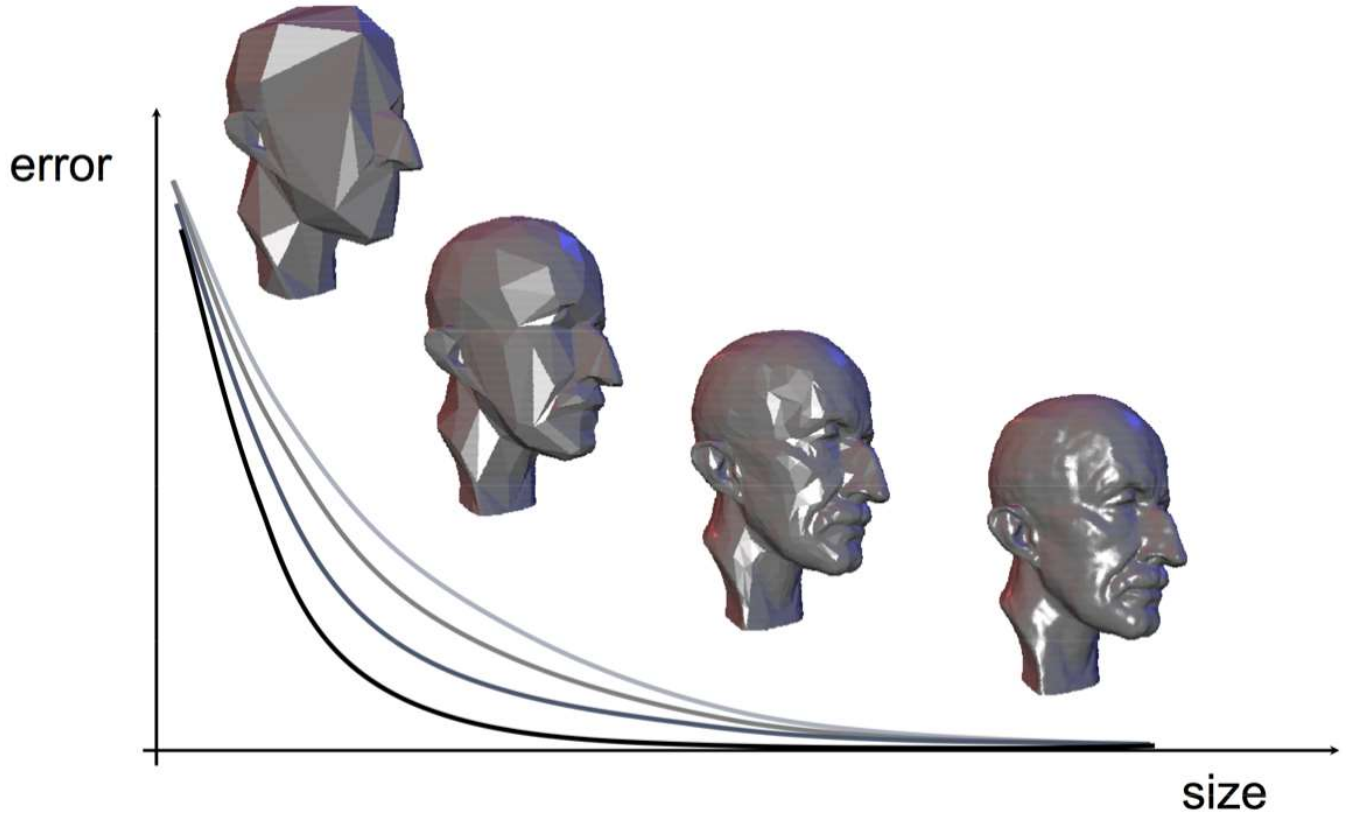
600k triángulos

60k triángulos

6k triángulos

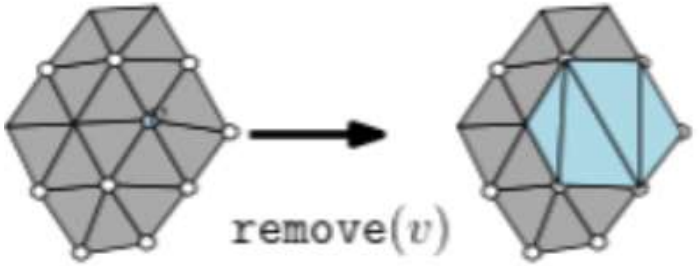
600 triángulos

Simplificación de Mallas

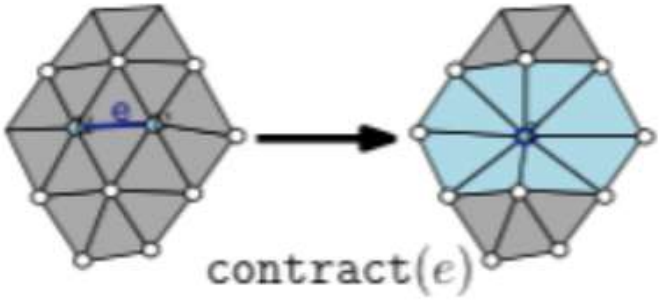


Decimación incremental - General

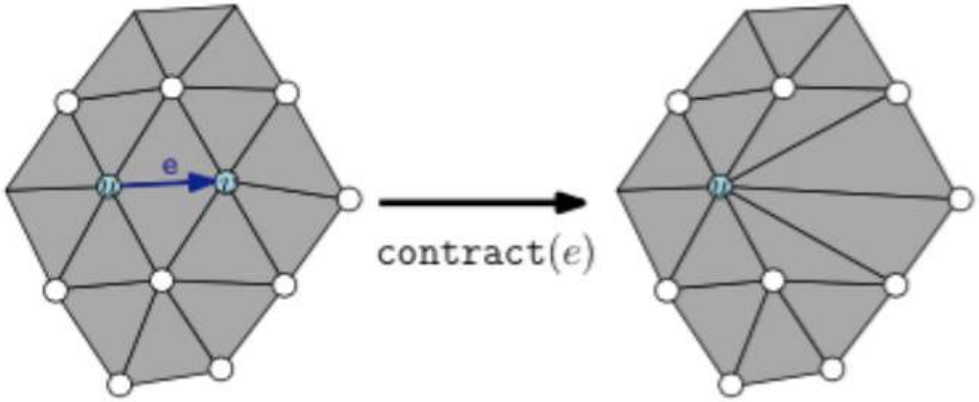
- Evaluar error local
- Seleccionar elementos removibles



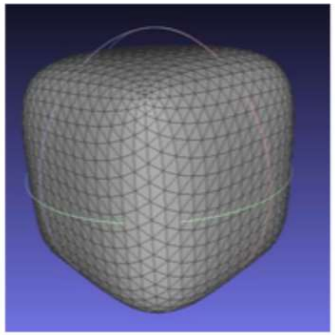
- Mientras $error > \varepsilon$
  - Seleccionar vértice o arista
  - Remover vértice o arista
  - Actualizar error local en geometría vecina

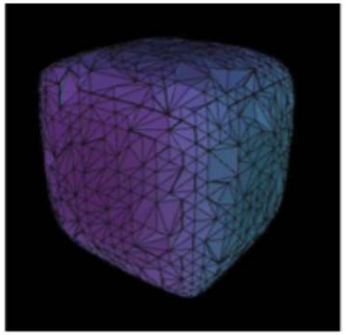


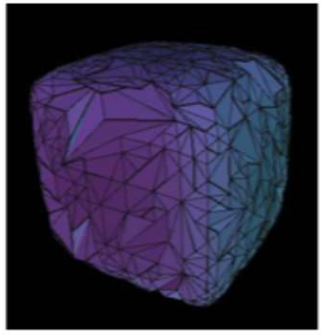
Decimación incremental – Contracción de arista



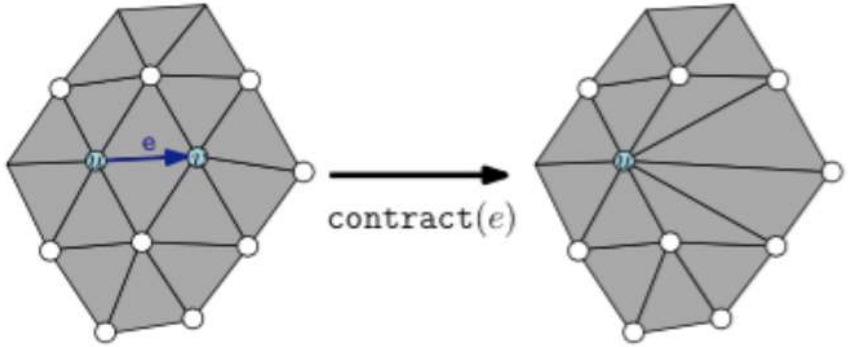
Contracciones basadas en selección aleatoria (sin criterio geométrico)



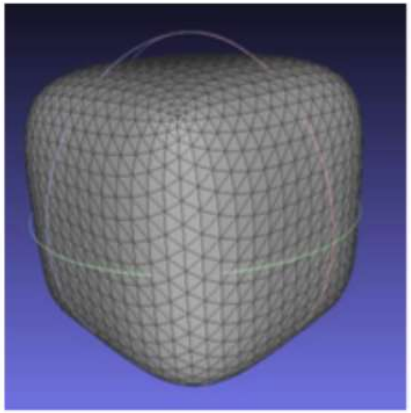


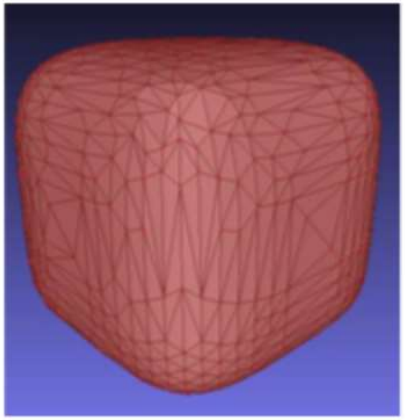


Decimación incremental – Contracción de arista



Contracciones basadas en criterio geométrico





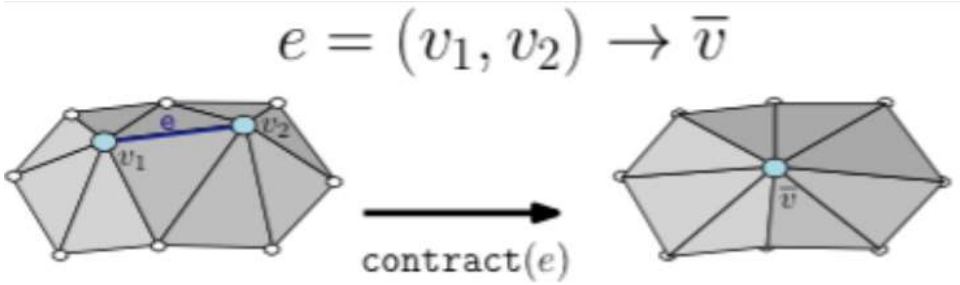
Aproximación de error con Quadrics

- Sea $v = [x, y, z, 1]$ un vértice en coordenadas homogéneas
- Asociar a cada vértice una matriz de 4x4 Q
- Error en el vértice v

$$\Delta(v) = v^T Q v$$

- Asociar un error al nuevo vértice $\bar{v}$

$$\bar{Q} = Q_1 + Q_2$$

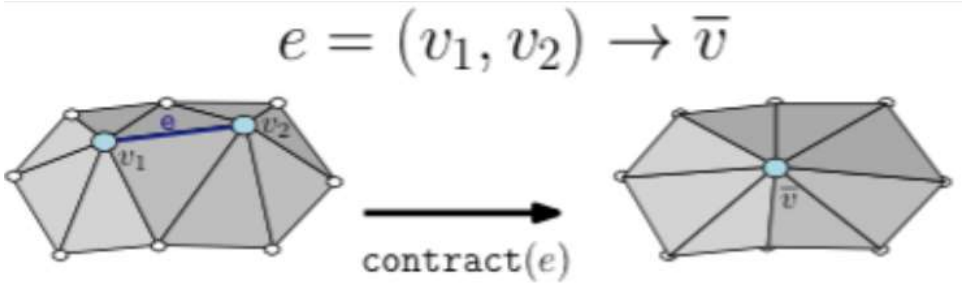


Aproximación de error con Quadrics

- Asociar un error al nuevo vértice $\bar{v}$

$$\Delta(v) = v^T Q v$$

$$\bar{Q} = Q_1 + Q_2$$



- Encontrar $\bar{v}$ que minimice $\Delta(\bar{v}) = \bar{v}^T \bar{Q} \bar{v}$

Aproximación de error con Quadrics

- Encontrar $\bar{v}$ que minimice $\Delta(\bar{v}) = \bar{v}^T \bar{Q} \bar{v}$
- Resolver un sistema lineal

$$\begin{cases} \frac{\partial \Delta(\bar{v})}{\partial x} = 0 \\ \frac{\partial \Delta(\bar{v})}{\partial y} = 0 \\ \frac{\partial \Delta(\bar{v})}{\partial z} = 0 \end{cases} \quad \longleftrightarrow \quad \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{12} & q_{22} & q_{23} & q_{24} \\ q_{13} & q_{23} & q_{33} & q_{24} \\ 0 & 0 & 0 & 1 \end{bmatrix} \overline{\mathbf{v}} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Aproximación de error con Quadrics

- Encontrar $\bar{v}$ que minimice $\Delta(\bar{v}) = \bar{v}^T \bar{Q} \bar{v}$
- Si la matriz es invertible

$$\overline{\mathbf{v}} = \begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{12} & q_{22} & q_{23} & q_{24} \\ q_{13} & q_{23} & q_{33} & q_{24} \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

- Sino escoger el promedio de v_1 y v_2

Aproximación de error con Quadrics

- Cómo se calcula el error?

$$\Delta(\mathbf{v}) = \Delta([v_x\ v_y\ v_z\ 1]^T) = \sum_{\mathbf{p}\in\text{planes}(\mathbf{v})} (\mathbf{p}^T\mathbf{v})^2$$

$$\begin{aligned}\Delta(\mathbf{v}) &= \sum_{\mathbf{p}\in\text{planes}(\mathbf{v})} (\mathbf{v}^T\mathbf{p})(\mathbf{p}^T\mathbf{v}) \\ &= \sum_{\mathbf{p}\in\text{planes}(\mathbf{v})} \mathbf{v}^T(\mathbf{p}\mathbf{p}^T)\mathbf{v} \\ &= \mathbf{v}^T\left(\sum_{\mathbf{p}\in\text{planes}(\mathbf{v})} \mathbf{K}_\mathbf{p}\right)\mathbf{v}\end{aligned}$$

$$ax + by + cz + d = 0$$

$$a^2 + b^2 + c^2 = 1$$

$$\mathbf{p} = [a\ b\ c\ 1]$$

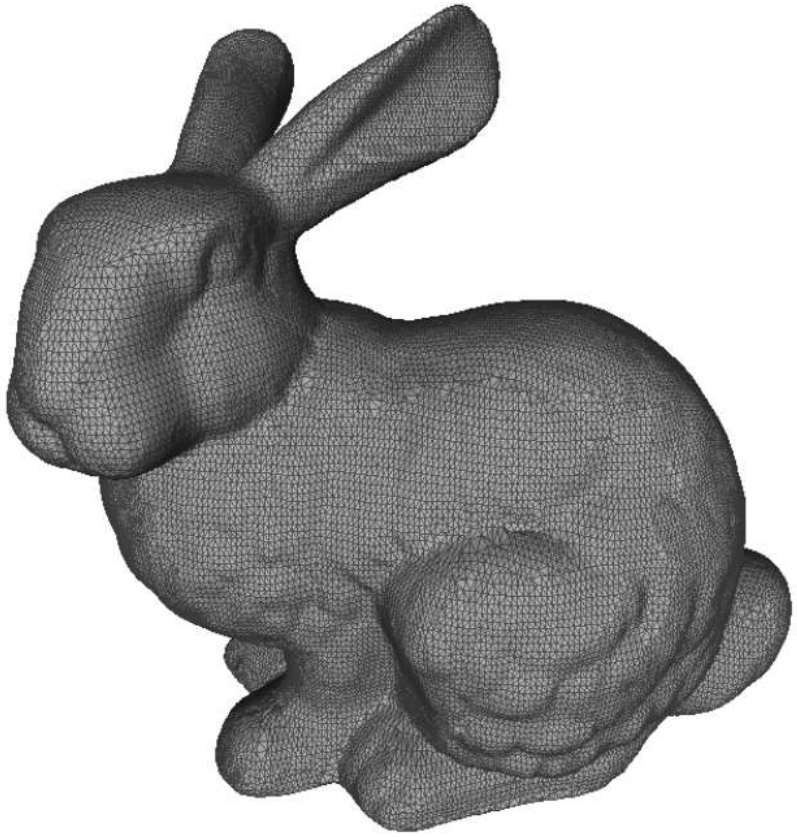
$$\mathbf{K}_\mathbf{p} = \mathbf{p}\mathbf{p}^T = \begin{bmatrix} a^2 & ab & ac & ad \\ ab & b^2 & bc & bd \\ ac & bc & c^2 & cd \\ ad & bd & cd & d^2 \end{bmatrix}$$

Algoritmo

- Computar la matriz de error Q_p de cada punto p
- Para cada arista candidata $\overline{pq}$
  - Minimizar $\Delta(r) = r^T(Q_p + Q_q)r$ para encontrar la posición óptima del vértice intermedio
  - Almacenar el error $\Delta(r)$ en una cola de prioridad

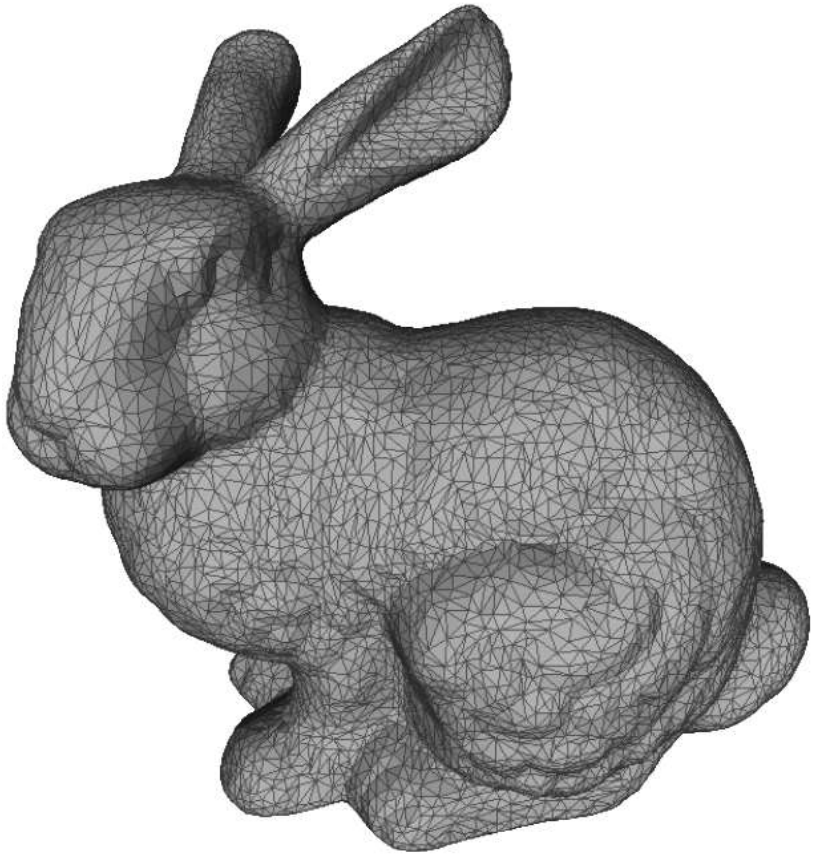
- Iterar:
  - Escoger la arista con el error más pequeño de la cola
  - Colapsar la arista y colocar el nuevo vértice colapsado
  - Actualizar las métricas de error de las aristas adyacentes

Resultados



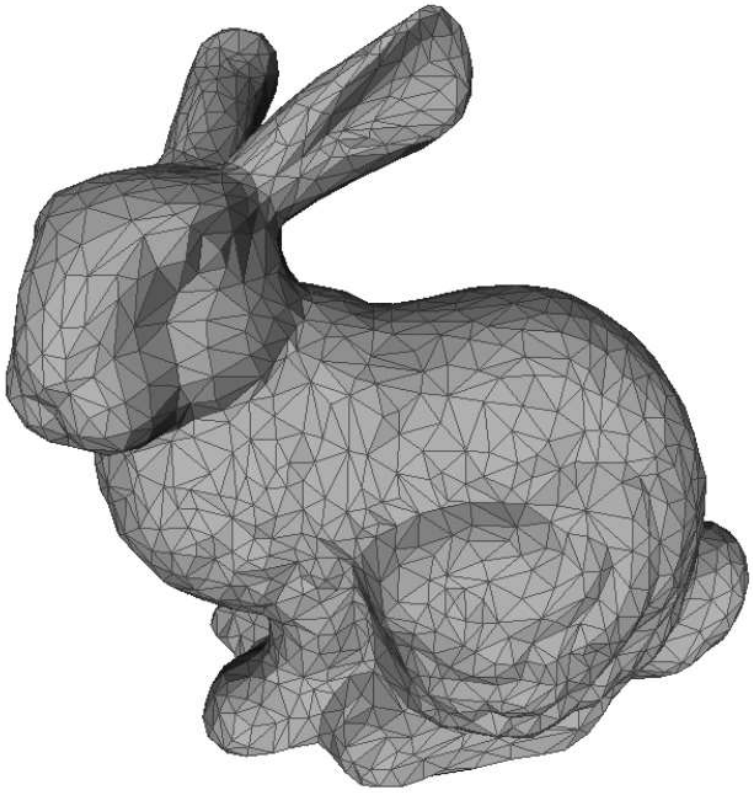
35k vertices

Resultados



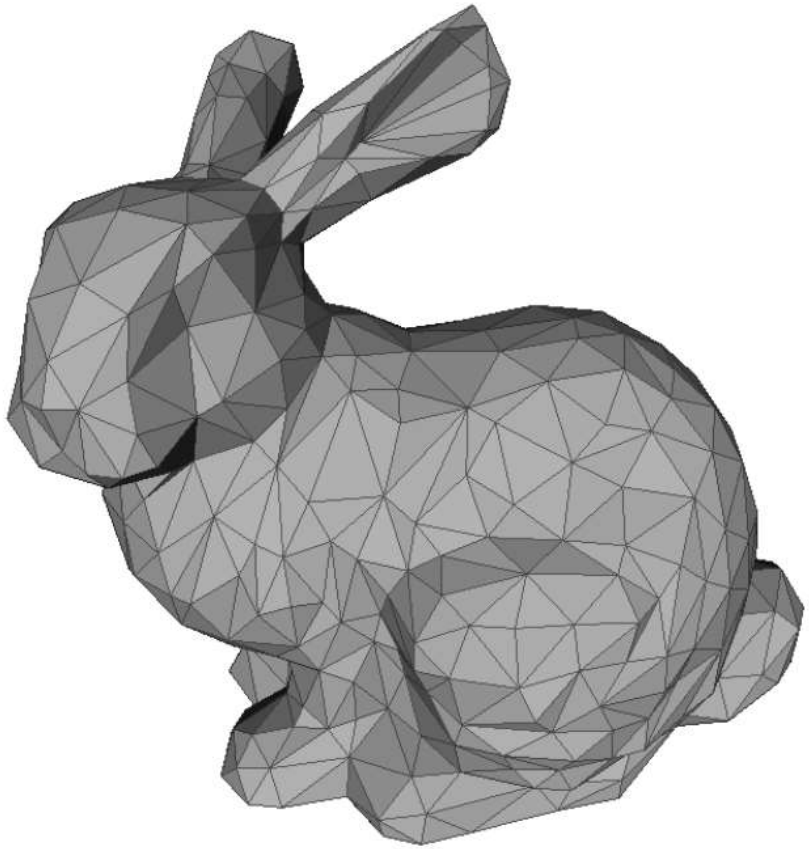
8.7k vertices

Resultados



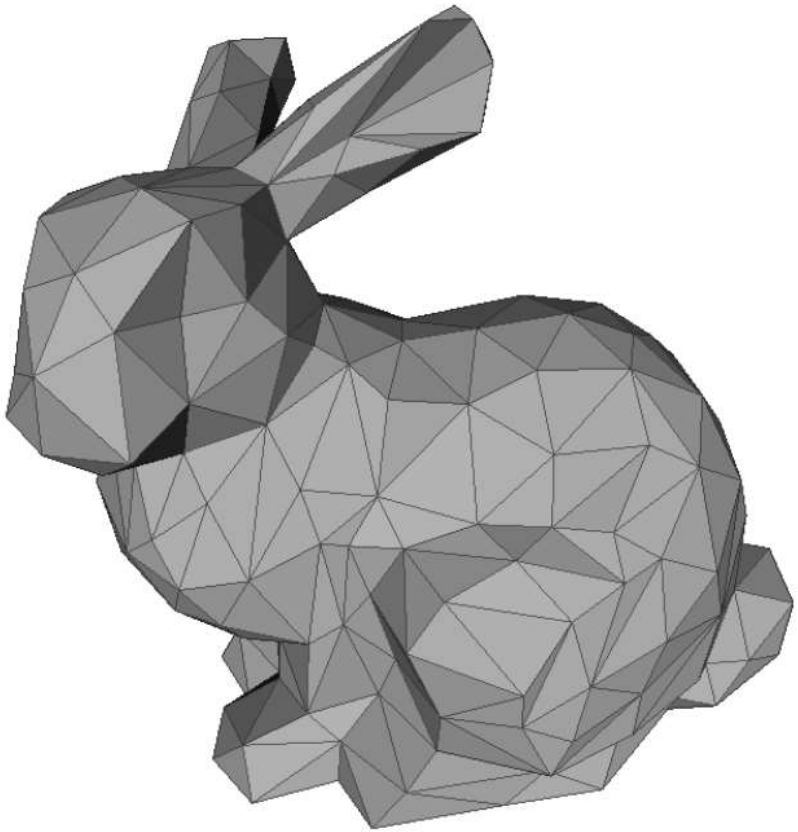
2,1k vertices

Resultados



560 vertices

Resultados

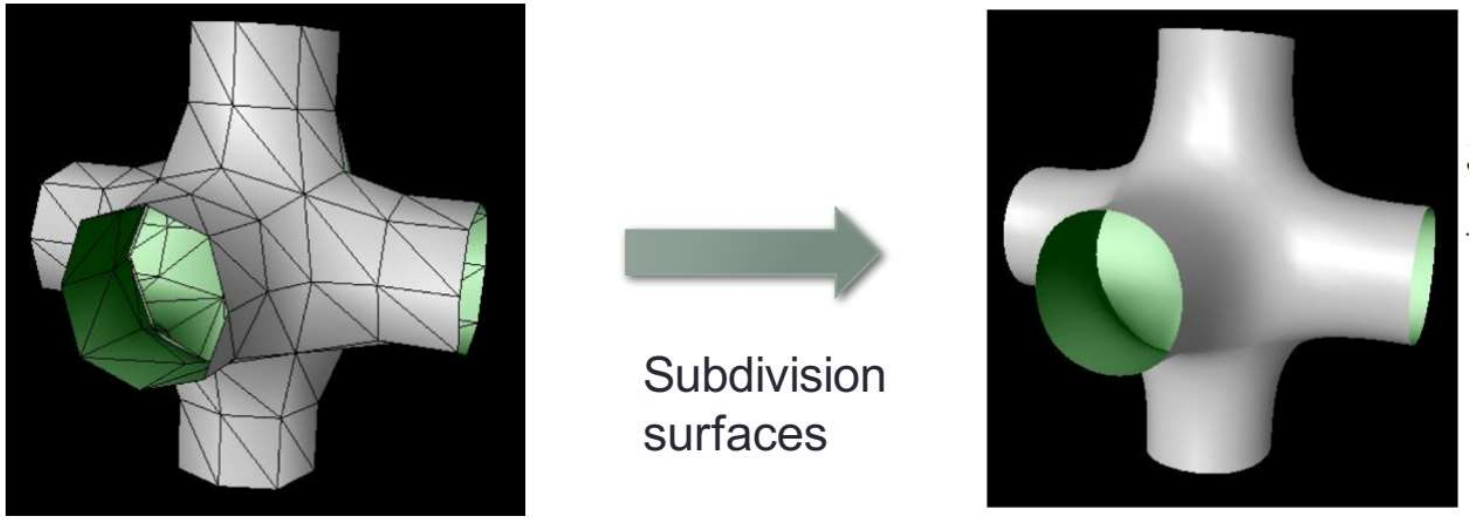


280 vertices

Subdivisión de mallas

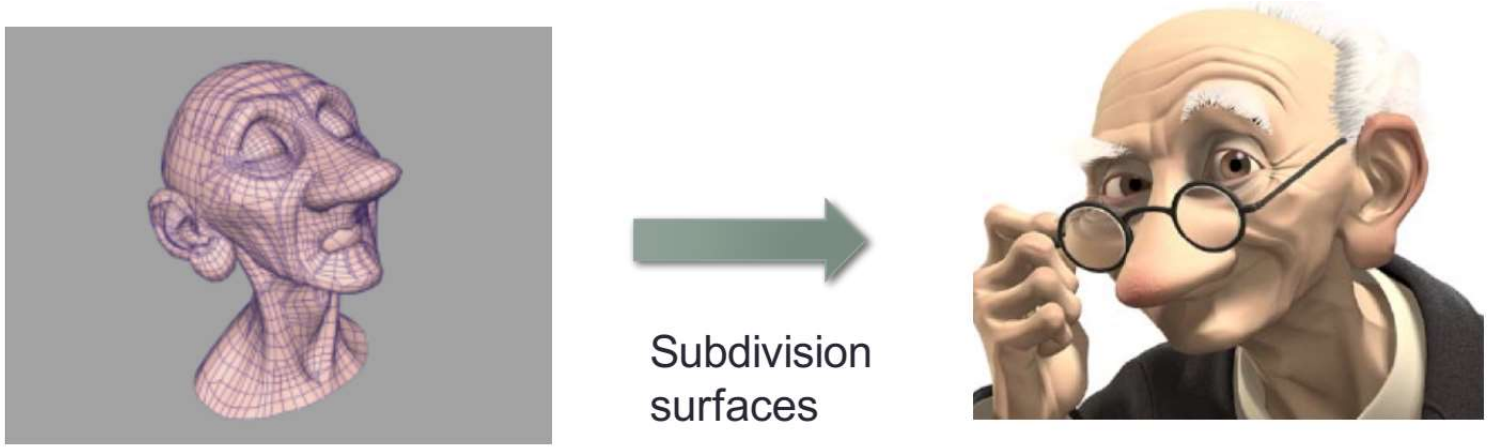
Superficies de subdivisión

- Trade-off entre suavidad y representación lineal por piezas
  - Inherentemente continuas
  - Puedes modelar formas con topología arbitraria



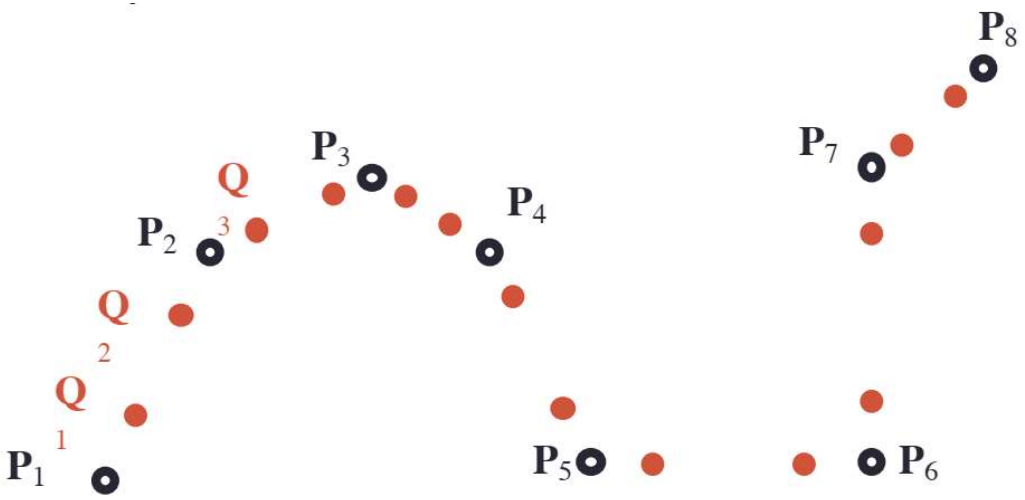
Superficies de subdivisión

- Trade-off entre suavidad y representación lineal por piezas
  - Inherentemente continuas
  - Puedes modelar formas con topología arbitraria



Subdivisión de curvas

- B-spline uniforme de orden 2: algoritmo de Chaikin

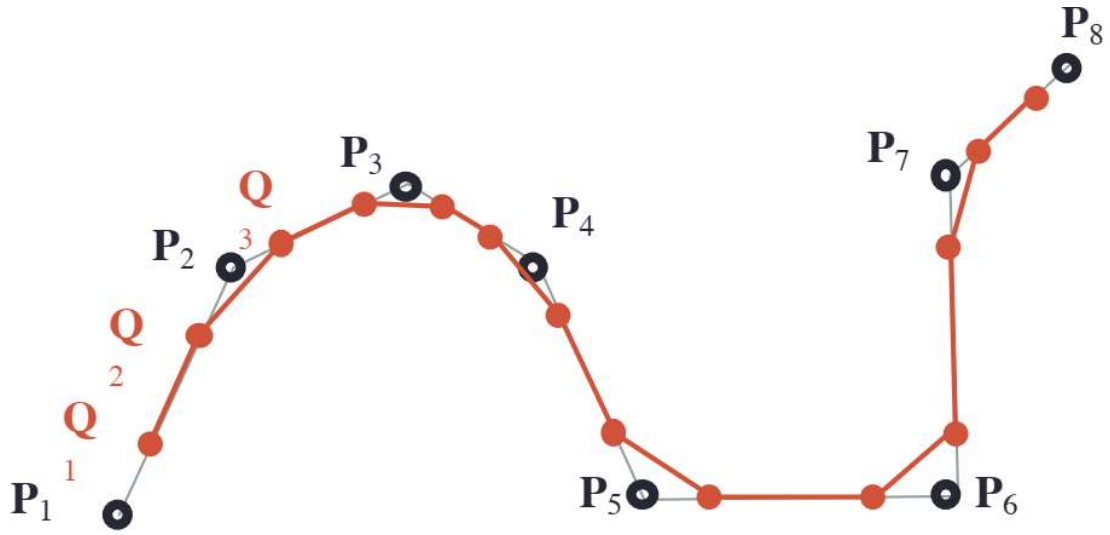


$$j \text{ odd: } \mathbf{Q}_j = \frac{3}{4}\mathbf{P}_{(j+1)/2} + \frac{1}{4}\mathbf{P}_{(j+3)/2}$$

$$j \text{ even: } \mathbf{Q}_j = \frac{1}{4}\mathbf{P}_{j/2} + \frac{3}{4}\mathbf{P}_{(j+2)/2}$$

Subdivisión de curvas

- B-spline uniforme de orden 2: algoritmo de Chaikin

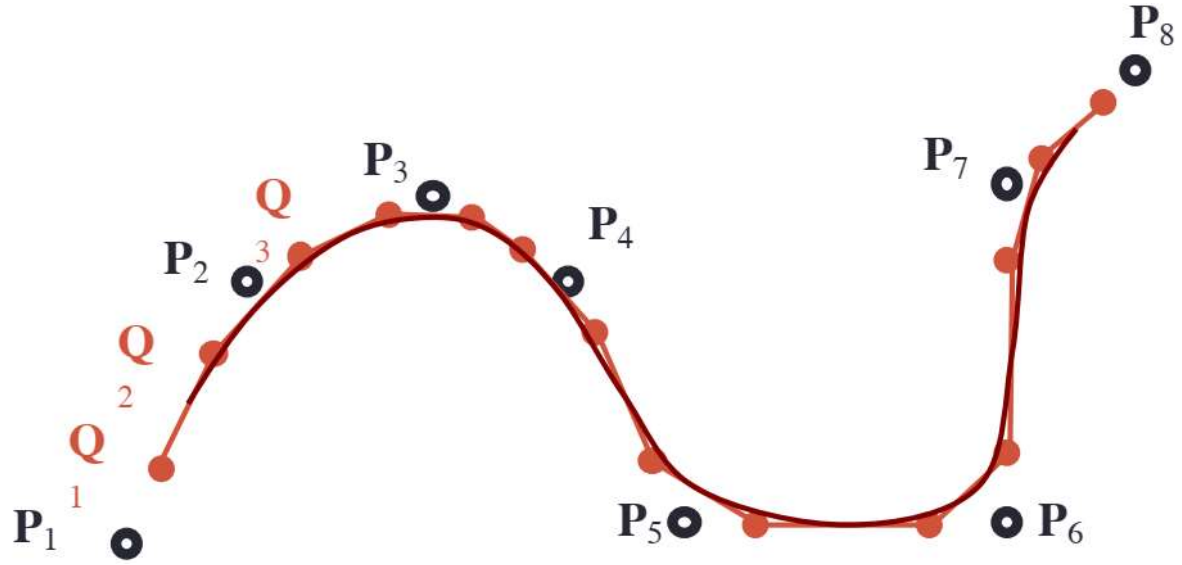


Dados n puntos: $\mathbf{P}_i, i \in (1, 2, \ldots, n)$

Se producen $2(n-1)$ puntos: $\mathbf{Q}_j, j \in (1, 2, \ldots, 2n-2)$

Subdivisión de curvas

- B-spline uniforme de orden 2: algoritmo de Chaikin



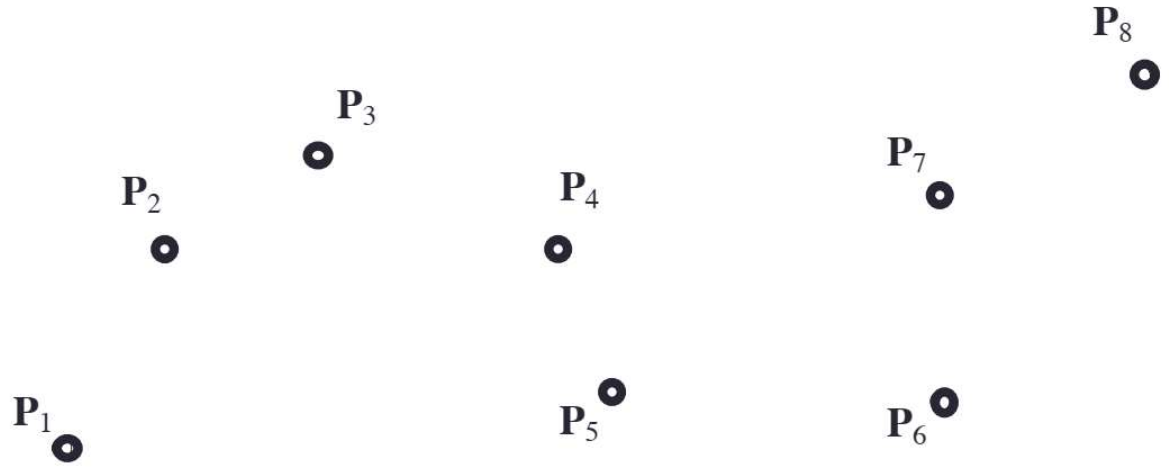
Dados n puntos: $\mathbf{P}_i, i \in (1, 2, \ldots, n)$

Se producen $2(n-1)$ puntos: $\mathbf{Q}_j, j \in (1, 2, \ldots, 2n-2)$

Iterar hasta alcanzar la cantidad de puntos deseada.

Subdivisión de curvas

- B-spline uniforme de orden 3: algoritmo de Chaikin



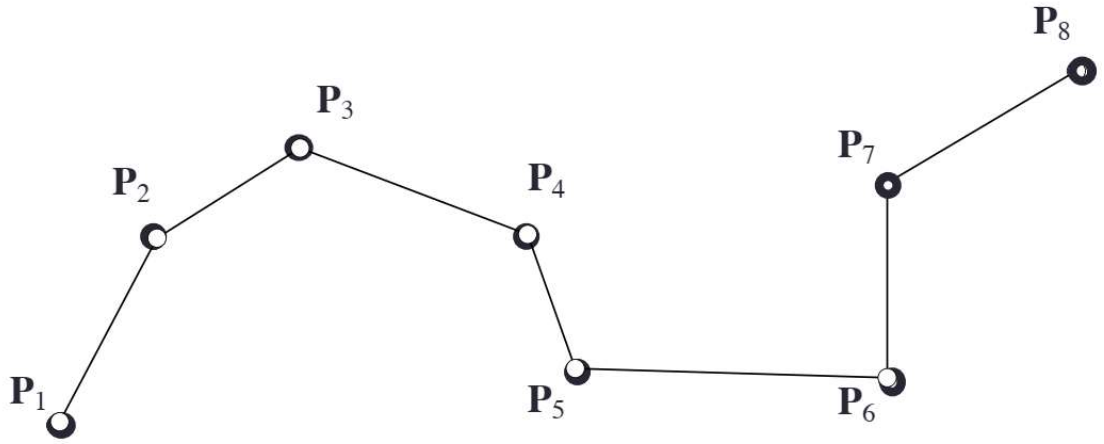
En cada iteración, producir $2(n-1)-1$ puntos

$$\mathbf{Q}_{2i-1} = \frac{1}{2}\mathbf{P}_i + \frac{1}{2}\mathbf{P}_{i+1}$$

$$\mathbf{Q}_{2i} = \frac{1}{8}\mathbf{P}_{i-1} + \frac{3}{4}\mathbf{P}_i + \frac{1}{8}\mathbf{P}_{i+1}$$

Subdivisión de curvas

- B-spline uniforme de orden 3: algoritmo de Chaikin



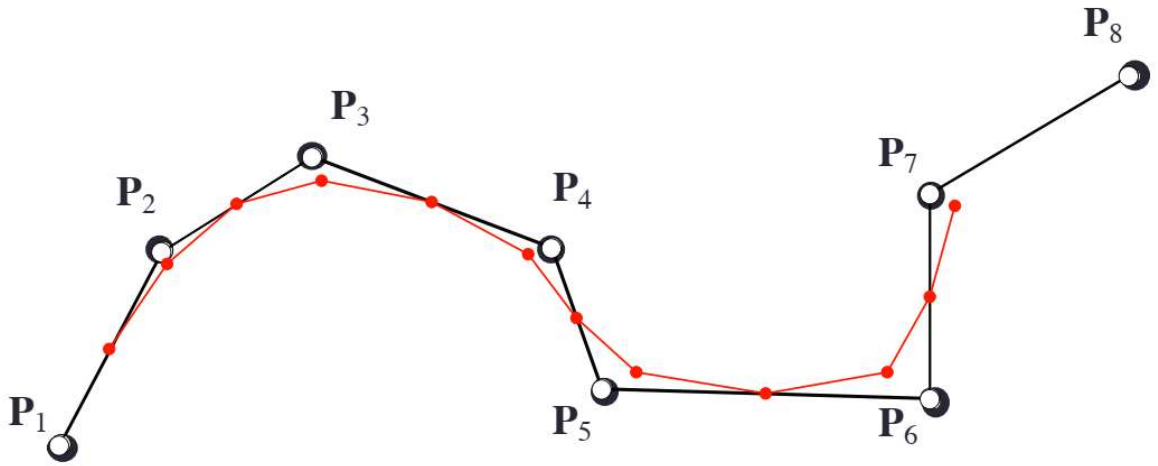
En cada iteración, producir $2(n-1)-1$ puntos

$$\mathbf{Q}_{2i-1} = \frac{1}{2}\mathbf{P}_i + \frac{1}{2}\mathbf{P}_{i+1}$$

$$\mathbf{Q}_{2i} = \frac{1}{8}\mathbf{P}_{i-1} + \frac{3}{4}\mathbf{P}_i + \frac{1}{8}\mathbf{P}_{i+1}$$

Subdivisión de curvas

- B-spline uniforme de orden 3: algoritmo de Chaikin



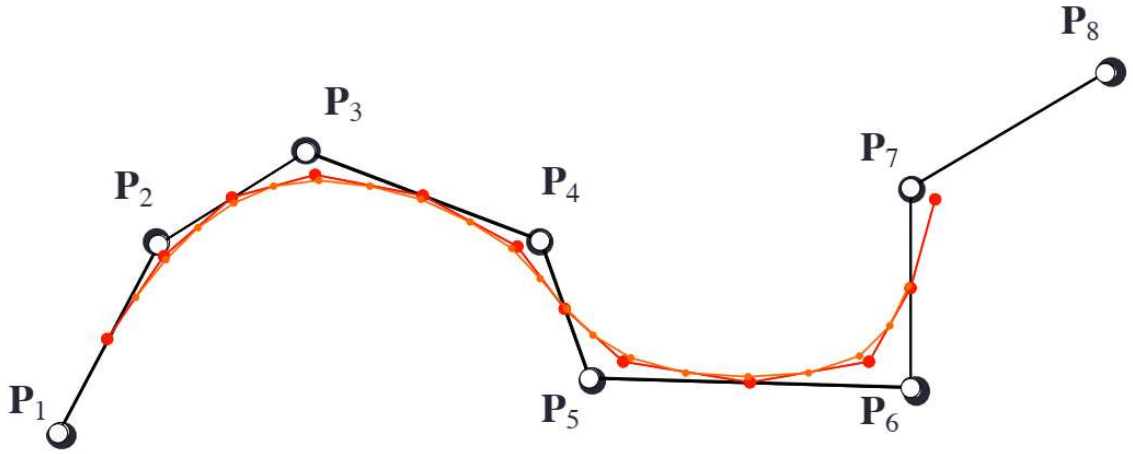
En cada iteración, producir $2(n-1)-1$ puntos

$$\mathbf{Q}_{2i-1} = \frac{1}{2}\mathbf{P}_i + \frac{1}{2}\mathbf{P}_{i+1}$$

$$\mathbf{Q}_{2i} = \frac{1}{8}\mathbf{P}_{i-1} + \frac{3}{4}\mathbf{P}_i + \frac{1}{8}\mathbf{P}_{i+1}$$

Subdivisión de curvas

- B-spline uniforme de orden 3: algoritmo de Chaikin



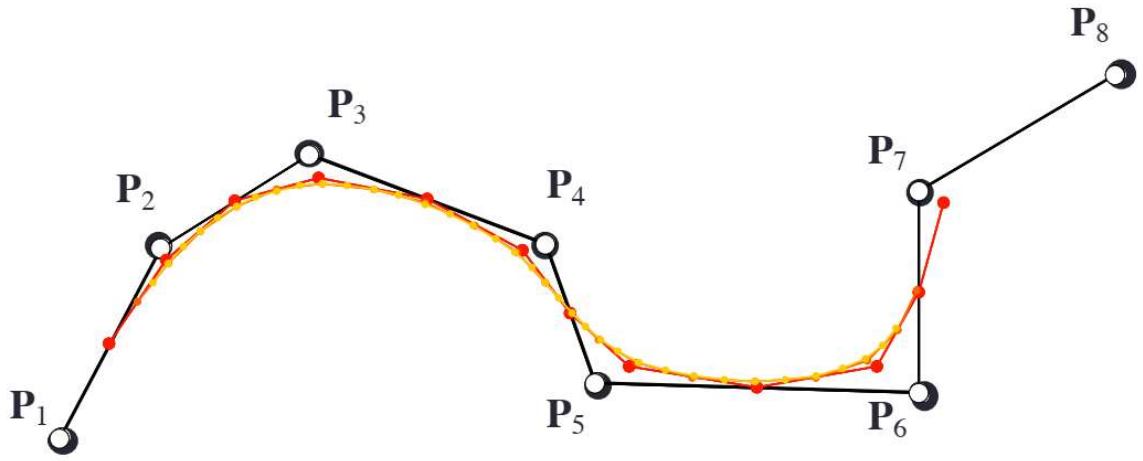
En cada iteración, producir $2(n-1)-1$ puntos

$$\mathbf{Q}_{2i-1} = \frac{1}{2}\mathbf{P}_i + \frac{1}{2}\mathbf{P}_{i+1}$$

$$\mathbf{Q}_{2i} = \frac{1}{8}\mathbf{P}_{i-1} + \frac{3}{4}\mathbf{P}_i + \frac{1}{8}\mathbf{P}_{i+1}$$

Subdivisión de curvas

- B-spline uniforme de orden 3: algoritmo de Chaikin



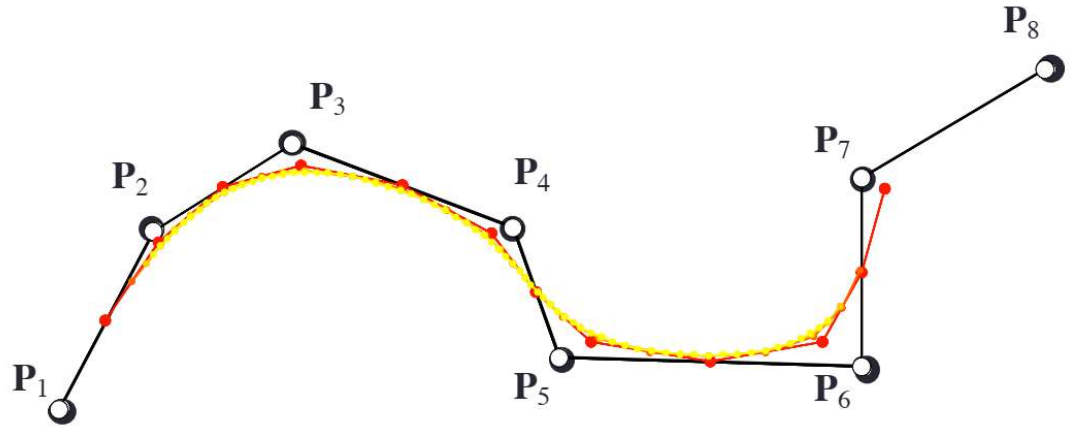
En cada iteración, producir $2(n-1)-1$ puntos

$$\mathbf{Q}_{2i-1} = \frac{1}{2}\mathbf{P}_i + \frac{1}{2}\mathbf{P}_{i+1}$$

$$\mathbf{Q}_{2i} = \frac{1}{8}\mathbf{P}_{i-1} + \frac{3}{4}\mathbf{P}_i + \frac{1}{8}\mathbf{P}_{i+1}$$

Subdivisión de curvas

- B-spline uniforme de orden 3: algoritmo de Chaikin

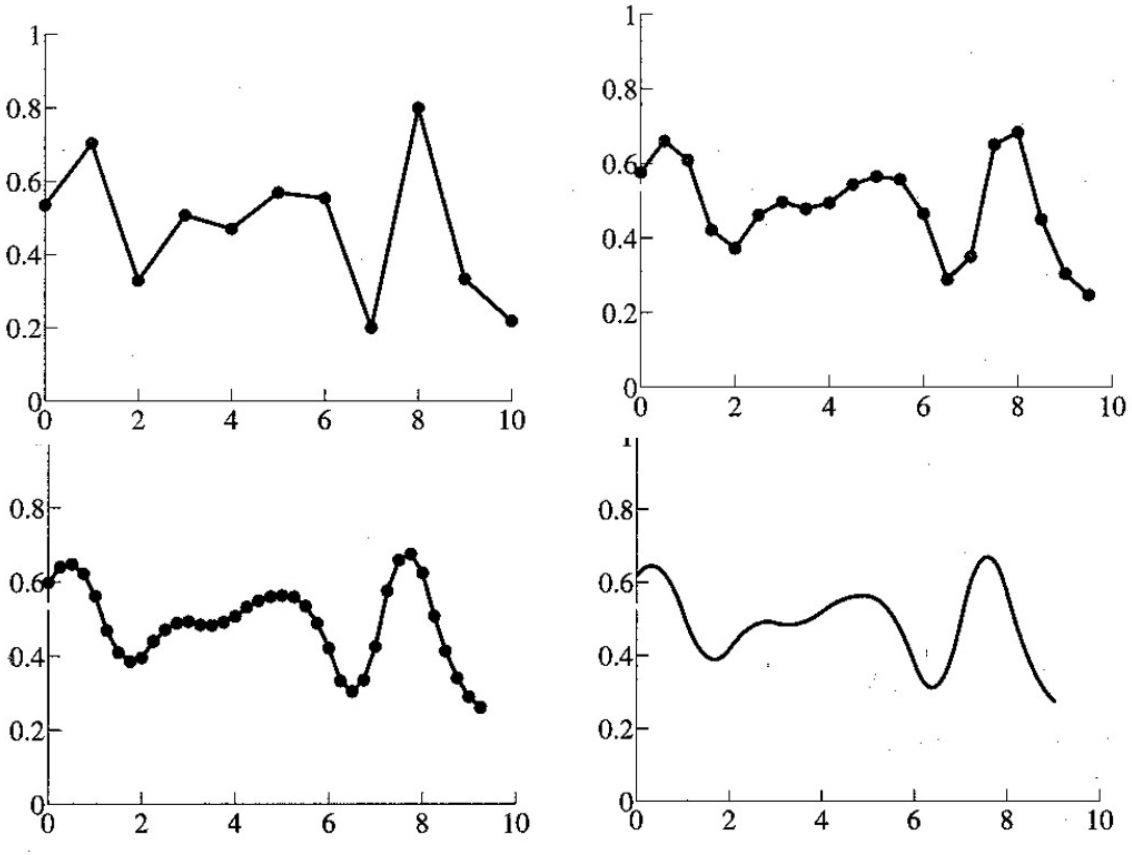


En cada iteración, producir $2(n-1)-1$ puntos

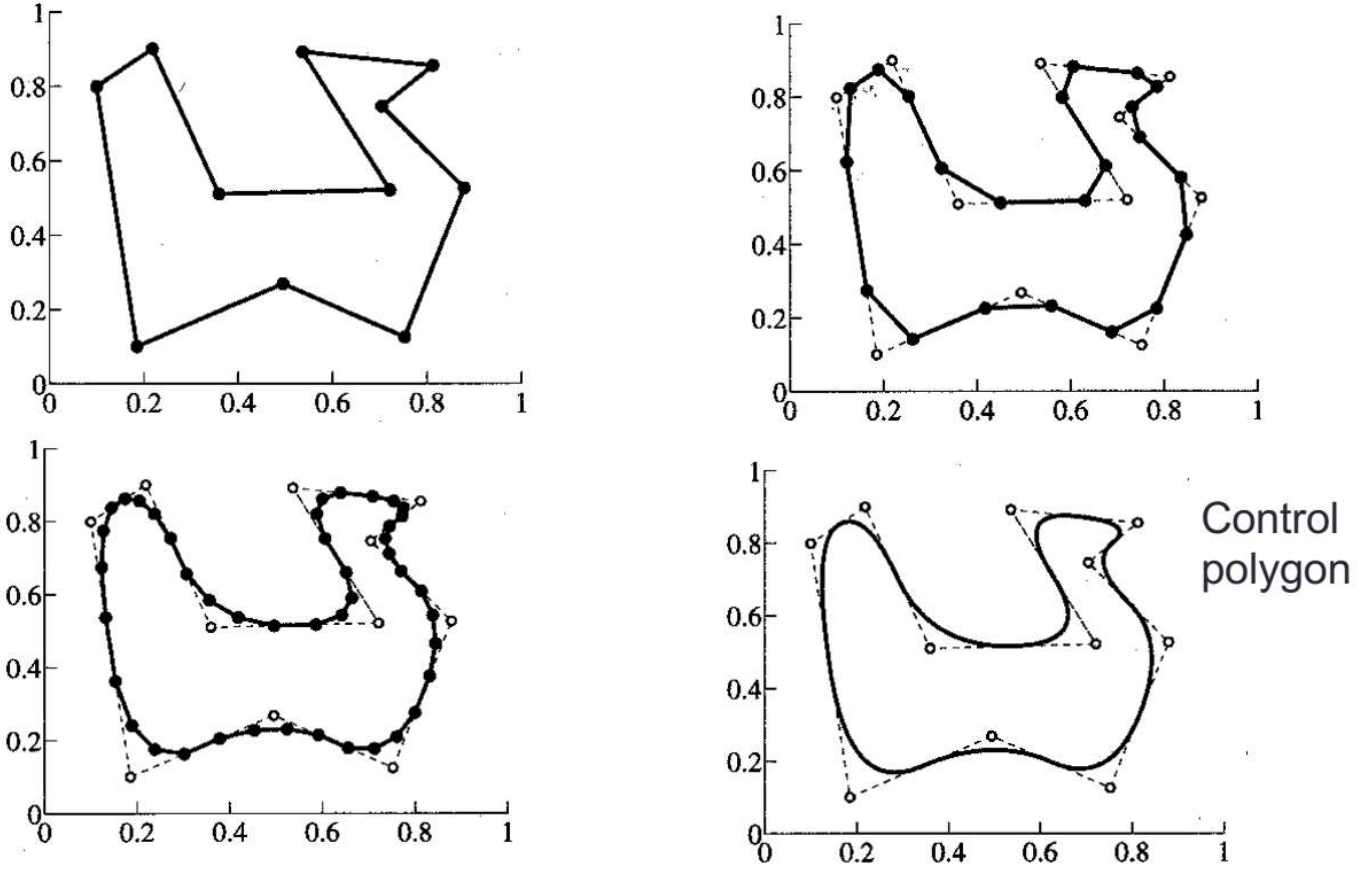
$$\mathbf{Q}_{2i-1} = \frac{1}{2}\mathbf{P}_i + \frac{1}{2}\mathbf{P}_{i+1}$$

$$\mathbf{Q}_{2i} = \frac{1}{8}\mathbf{P}_{i-1} + \frac{3}{4}\mathbf{P}_i + \frac{1}{8}\mathbf{P}_{i+1}$$

Subdivisión de curvas

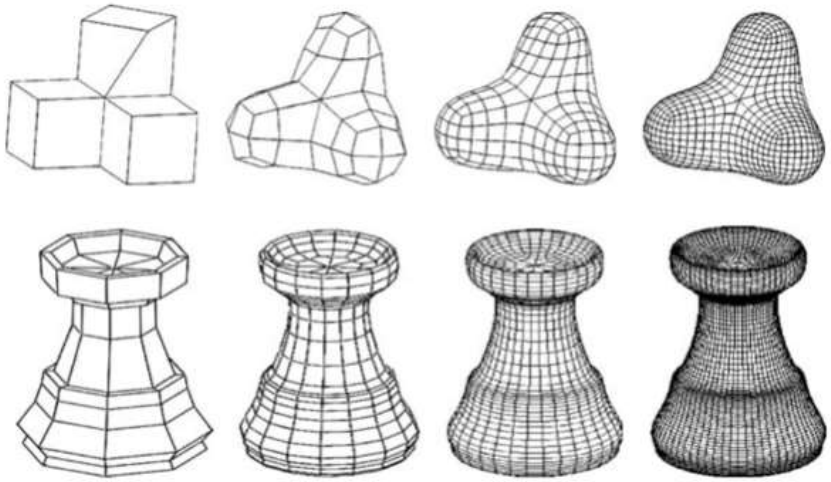


Subdivisión de curvas



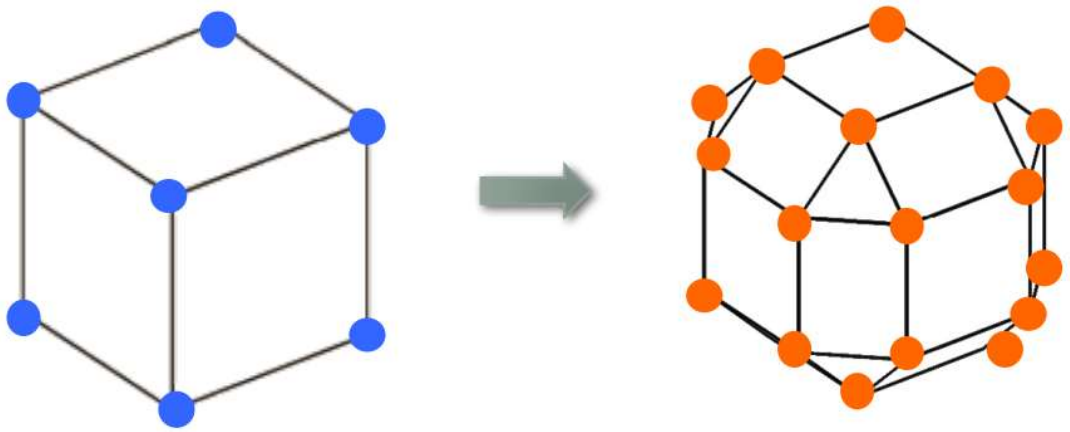
Subdivisión de superficies

- Aplicar las mismas ideas para generar superficies suaves
- Enfoque general
  - Empezar con un polítopo de control
  - En cada iteración refinar el polítopo de acuerdo a reglas
  - Detener la ejecución cuando la resolución sea lo suficiente



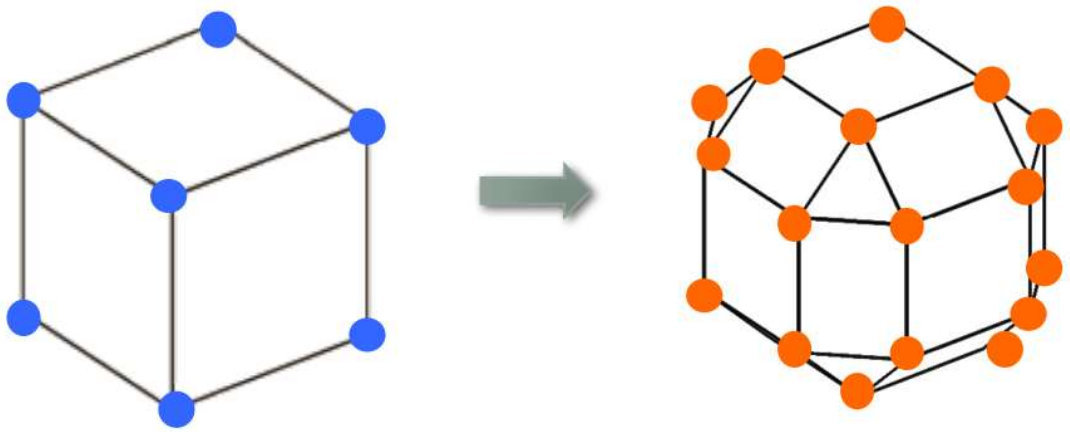
Reglas de subdivisión

- Hay cambios topológicos y geométricos
- Geométricos:
  - Cambio en las posiciones de los vértices
- Topológicos
  - Cambio en la conectividad



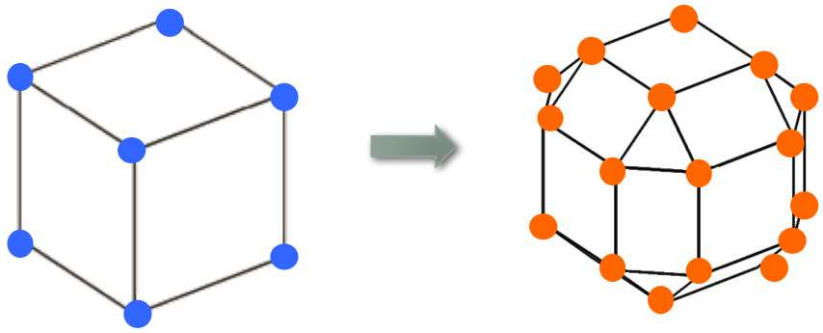
Reglas de subdivisión

- Hay cambios topológicos y geométricos
- Típicamente, ambos cambios son locales:
  - Nuevos vértices, aristas y caras dependen de una vecindad pequeña



Subdivisión de Doo-Sabin

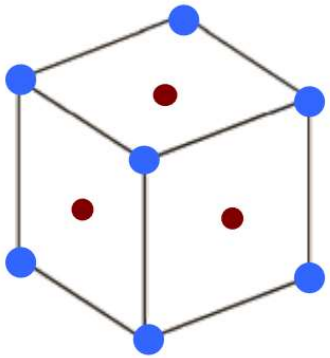
- Generalización de Chaikin



- En cada iteración:
  - Considerar el baricentro de las caras antiguas
  - Construir centroides entre el centro y los vértices antiguos
  - Conectarlos en forma natural
  - Repetir

Subdivisión de Doo-Sabin

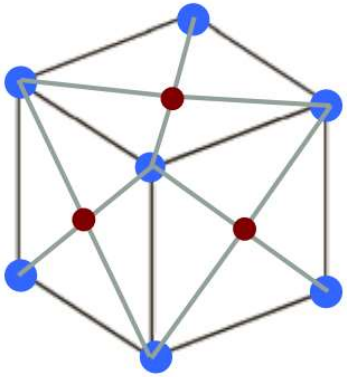
- Generalización de Chaikin



- En cada iteración:
  - Considerar el baricentro de las caras antiguas
  - Construir centroides entre el centro y los vértices antiguos
  - Conectarlos en forma natural
  - Repetir

Subdivisión de Doo-Sabin

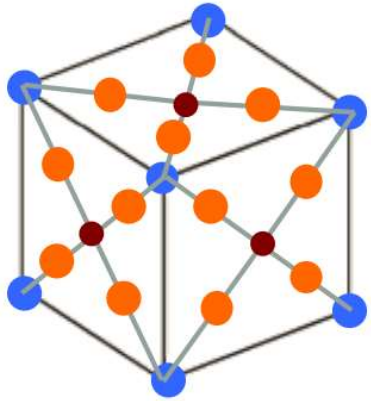
- Generalización de Chaikin



- En cada iteración:
  - Considerar el baricentro de las caras antiguas
  - Construir centroides entre el centro y los vértices antiguos
  - Conectarlos en forma natural
  - Repetir

Subdivisión de Doo-Sabin

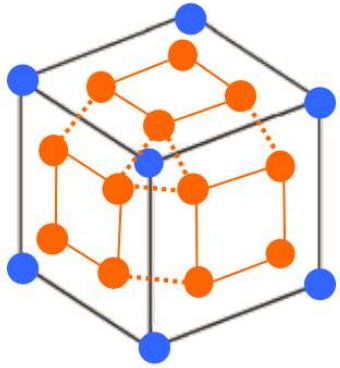
- Generalización de Chaikin



- En cada iteración:
  - Considerar el baricentro de las caras antiguas
  - Construir centroides entre el centro y los vértices antiguos
  - Conectarlos en forma natural
  - Repetir

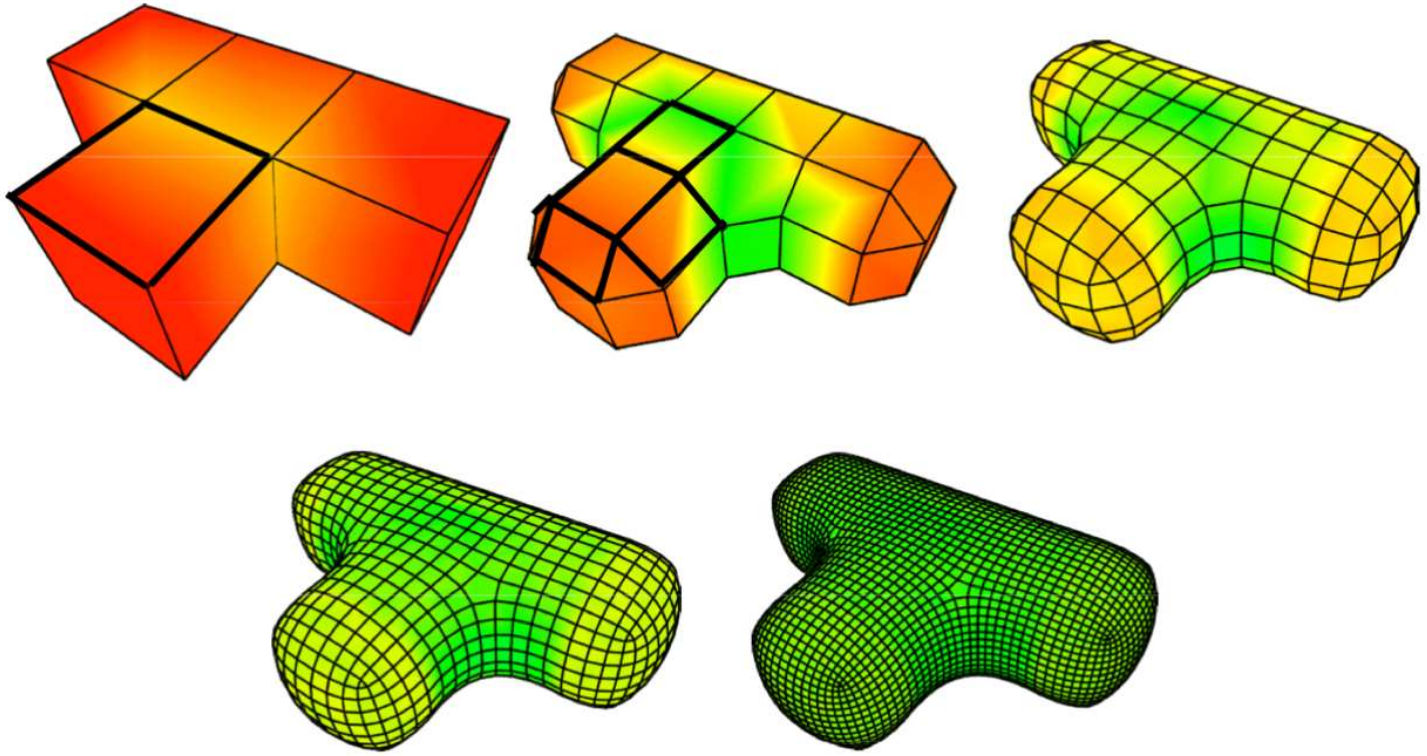
Subdivisión de Doo-Sabin

- Generalización de Chaikin



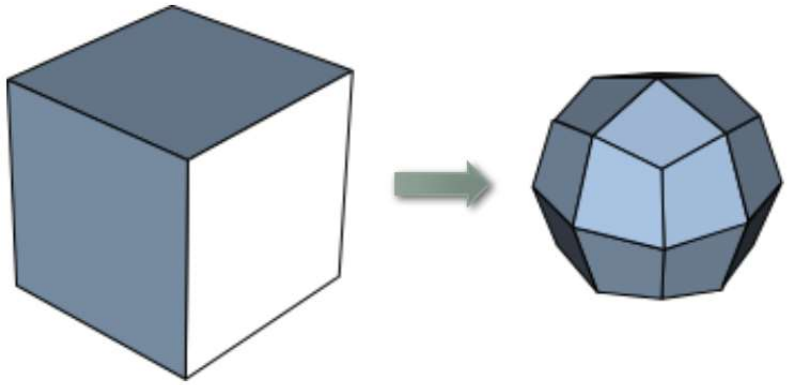
- En cada iteración:
  - Considerar el baricentro de las caras antiguas
  - Construir centroides entre el centro y los vértices antiguos
  - Conectarlos en forma natural
  - Repetir

Subdivisión de Doo-Sabin



Subdivisión Catmull-Clark

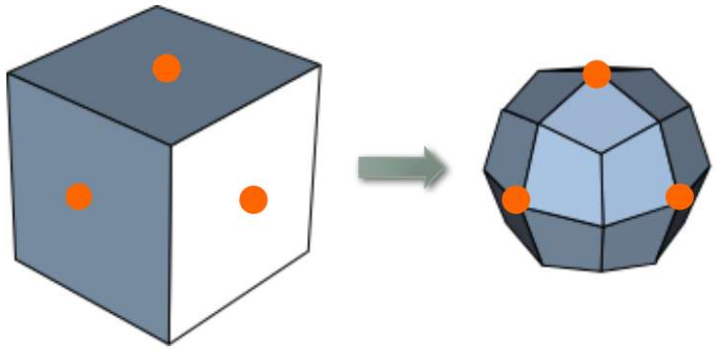
- Generalización de la subdivisión de spline cúbica de superficies



- Superficie de aproximación
- Operaciones locales
- Subdivisión usada en Pixar

Subdivisión Catmull-Clark

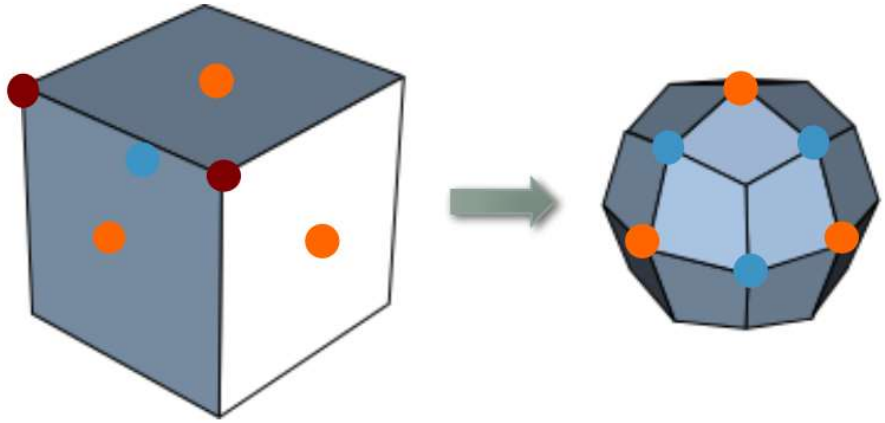
- Generalización de la subdivisión de spline cúbica de superficies



- En cada iteración
  - Construir vértices de caras: baricentros
  - Construir vértices de aristas
  - Actualizar vértices existentes
  - Conectarlos en forma natural
  - Repetir

Subdivisión Catmull-Clark

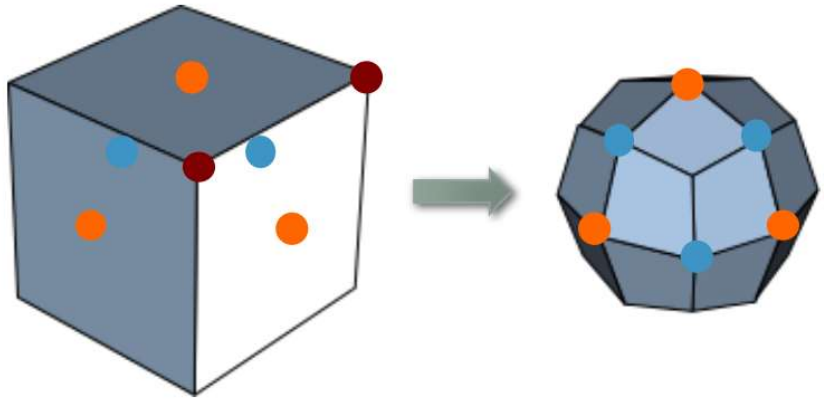
- Generalización de la subdivisión de spline cúbica de superficies



- En cada iteración
  - Construir vértices de caras: baricentros
  - Construir vértices de aristas: promediar vértices de aristas antiguos y vértices de cara asociados

Subdivisión Catmull-Clark

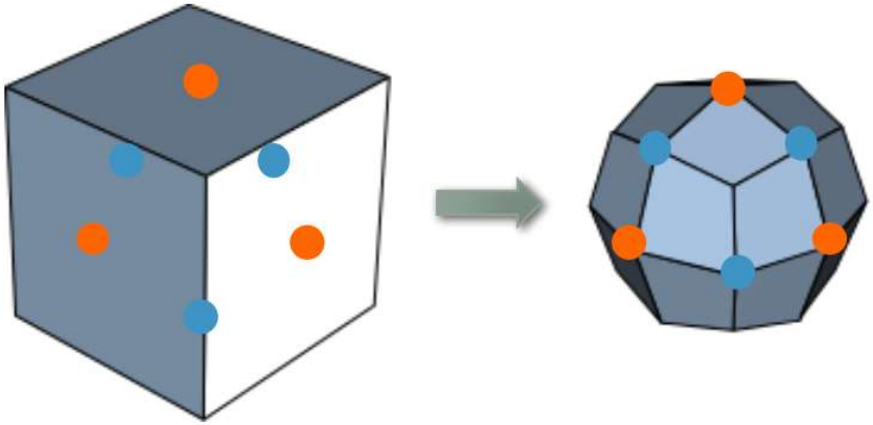
- Generalización de la subdivisión de spline cúbica de superficies



- En cada iteración
  - Construir vértices de caras: baricentros
  - Construir vértices de aristas: promediar vértices de aristas antiguos y vértices de cara asociados

Subdivisión Catmull-Clark

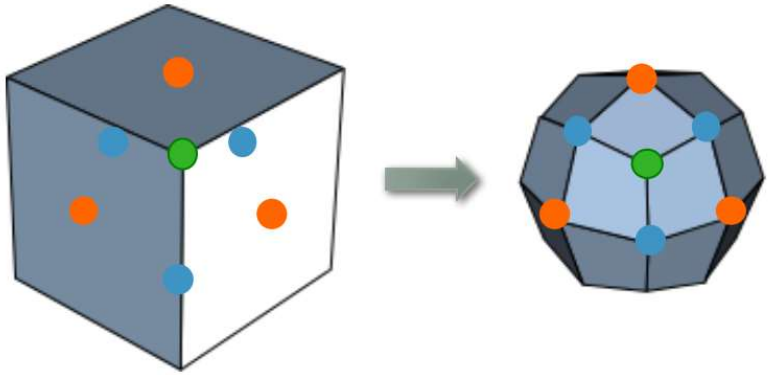
- Generalización de la subdivisión de spline cúbica de superficies



- En cada iteración
  - Construir vértices de caras: baricentros
  - Construir vértices de aristas: promediar vértices de aristas antiguos y vértices de cara asociados

Subdivisión Catmull-Clark

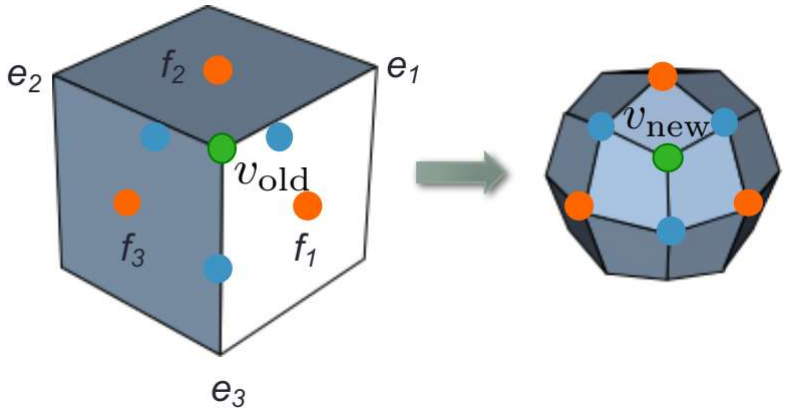
- Generalización de la subdivisión de spline cúbica de superficies



- En cada iteración
  - Construir vértices de caras: baricentros
  - Construir vértices de aristas
  - Actualizar vértices existentes
  - Conectarlos en forma natural
  - Repetir

Subdivisión Catmull-Clark

- Generalización de la subdivisión de spline cúbica de superficies



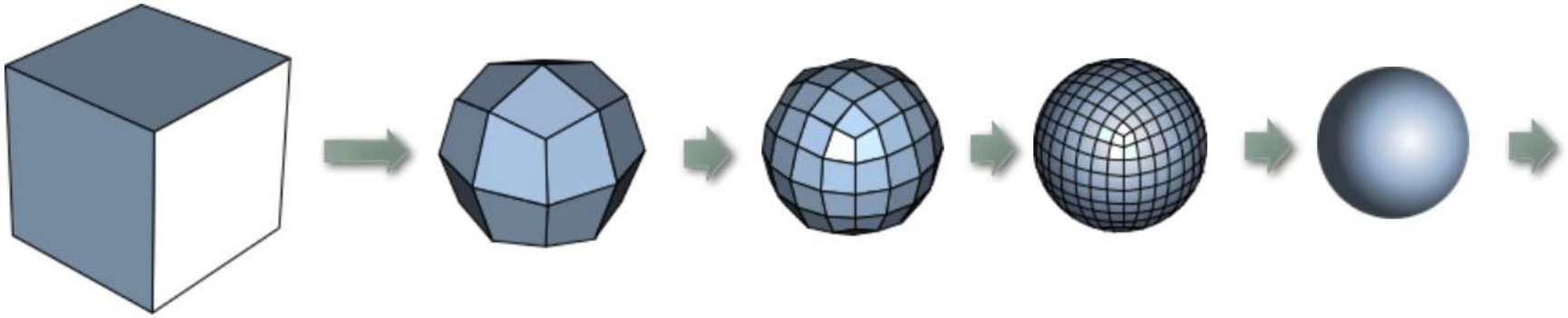
- En cada iteración
  - Construir vértices de caras: baricentros
  - Construir vértices de aristas
  - Actualizar vértices existentes

$$v_{\text{new}} = v_{\text{old}} + \frac{1}{n^2}\sum_{j=1}^{n}(e_j - v_{\text{old}}) + \frac{1}{n^2}\sum_{j=1}^{n}(f_j - v_{\text{old}})$$

e_j: vértice incidente en arista j
f_j: vértice en cara j
n: número de aristas incidentes

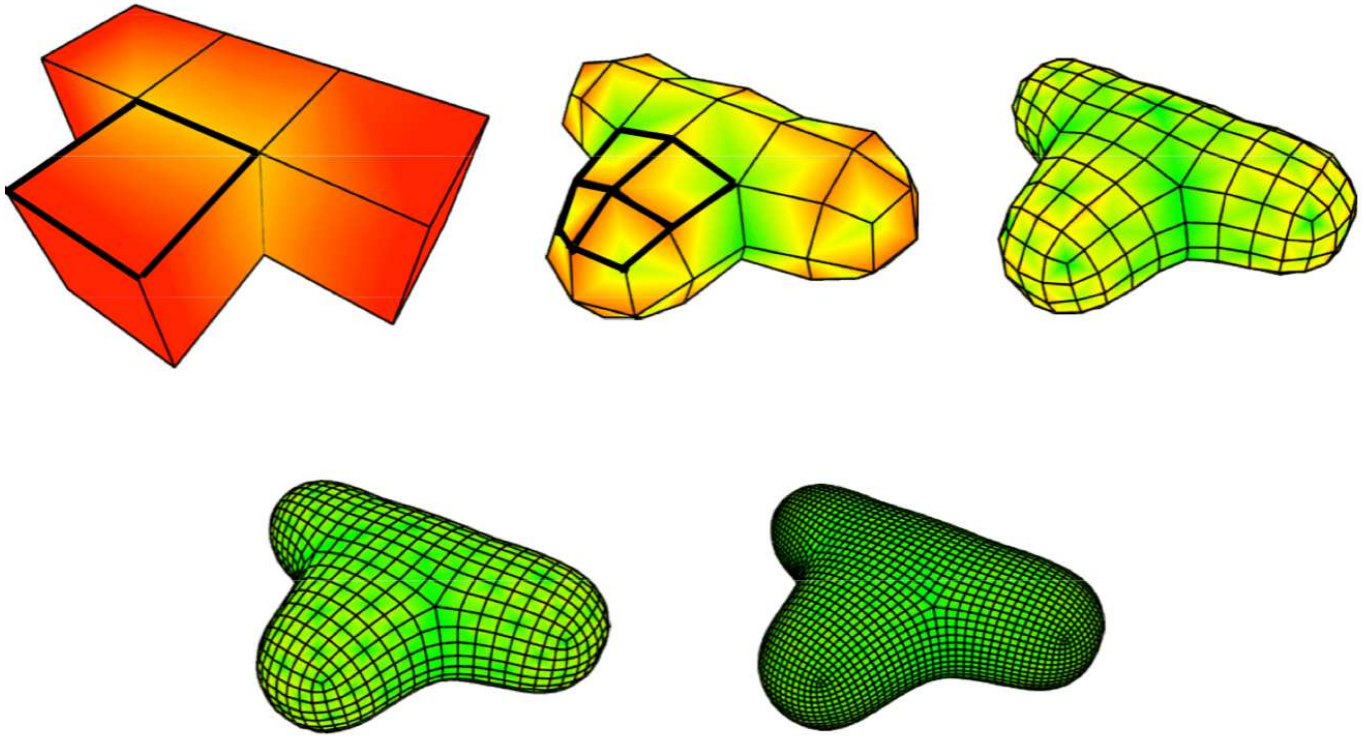
Subdivisión Catmull-Clark

- Generalización de la subdivisión de spline cúbica de superficies



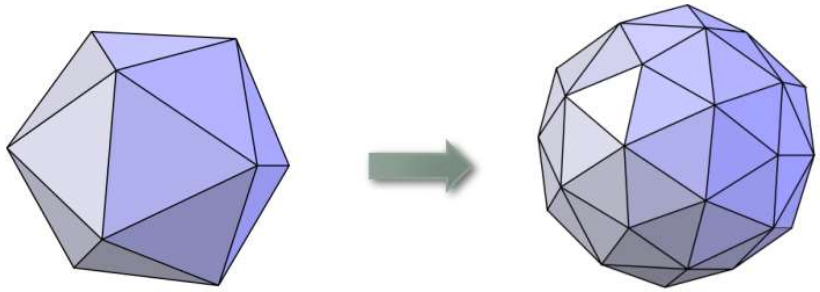
Subdivisión Catmull-Clark

- Generalización de la subdivisión de spline cúbica de superficies



Subdivisión Loop

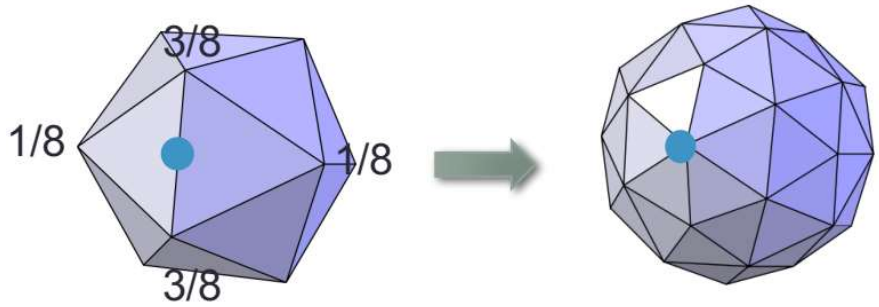
- Subdivisión de triángulos



- En cada iteración
  - Construir vértices de aristas
  - Actualizar vértices existentes
  - Conectarlos en forma natural
  - Repetir

Subdivisión Loop

- Subdivisión de triángulos

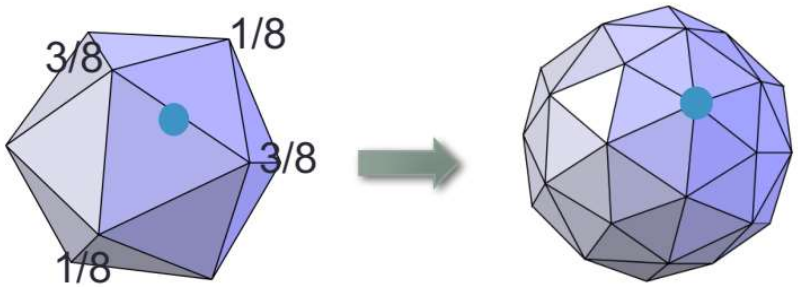


- En cada iteración
  - Construir vértices de aristas

$$e_i = \frac{3}{8}v_{e1} + \frac{3}{8}v_{e2} + \frac{1}{8}v_{t1} + \frac{1}{8}v_{t2}$$

Subdivisión Loop

- Subdivisión de triángulos

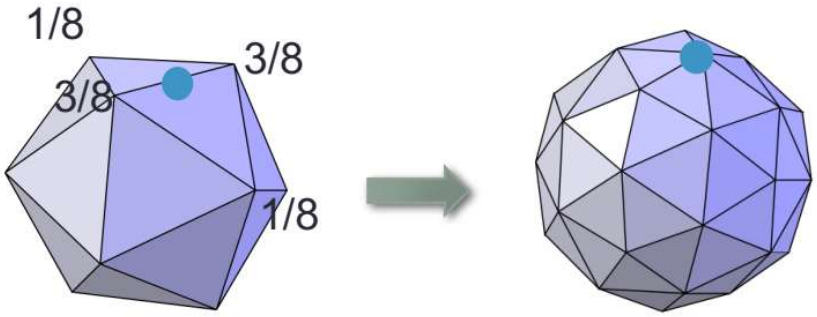


- En cada iteración
  - Construir vértices de aristas

$$e_i = \frac{3}{8}v_{e1} + \frac{3}{8}v_{e2} + \frac{1}{8}v_{t1} + \frac{1}{8}v_{t2}$$

Subdivisión Loop

- Subdivisión de triángulos

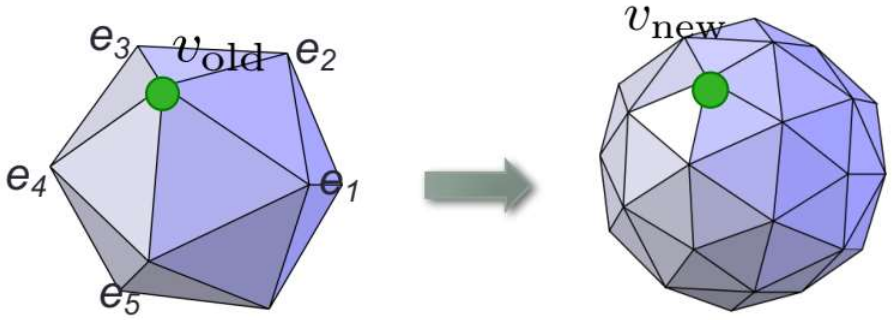


- En cada iteración
  - Construir vértices de aristas

$$e_i = \frac{3}{8}v_{e1} + \frac{3}{8}v_{e2} + \frac{1}{8}v_{t1} + \frac{1}{8}v_{t2}$$

Subdivisión Loop

- Subdivisión de triángulos



- En cada iteración
  - Construir vértices de aristas
  - Actualizar vértices existentes

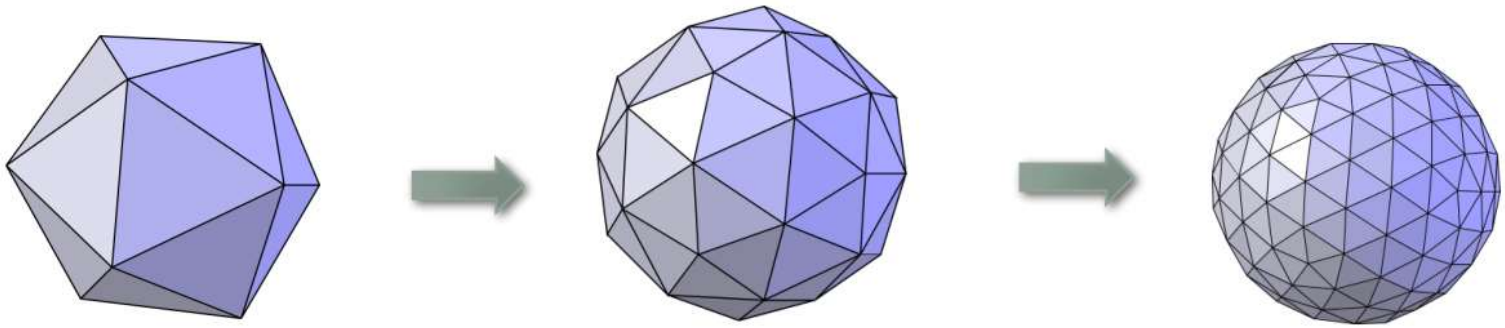
$$v_{\text{new}} = (1 - \alpha n) v_{\text{old}} + \alpha \sum_{j=1}^{n} e_j$$

$$\alpha = \begin{cases} \frac{3}{16} & \text{if } n = 3 \\ \frac{3}{8n} & \text{if } n > 3 \end{cases}$$

e_j: vértice incidente a la arista j
n: número de aristas incidentes

Subdivisión Loop

- Subdivisión de triángulos



Subdivisión Loop