Universidad de Huelva

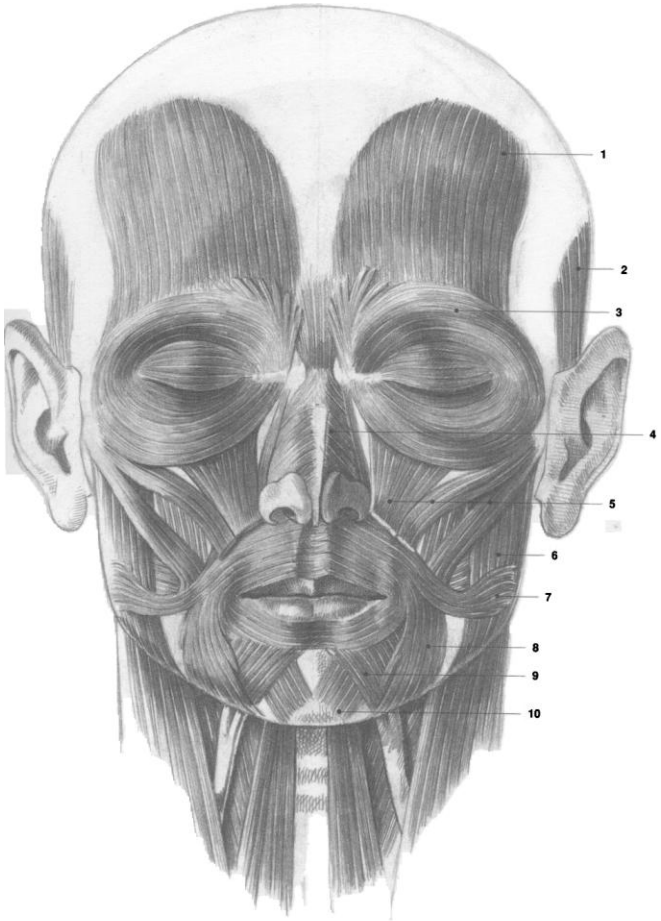
Tema 12

Expresiones faciales y Morphing

Índice

Índice

Músculos faciales



Expresiones universales

- Tristeza
- Enfado
- Felicidad
- Miedo
- Asco
- Sorpresa

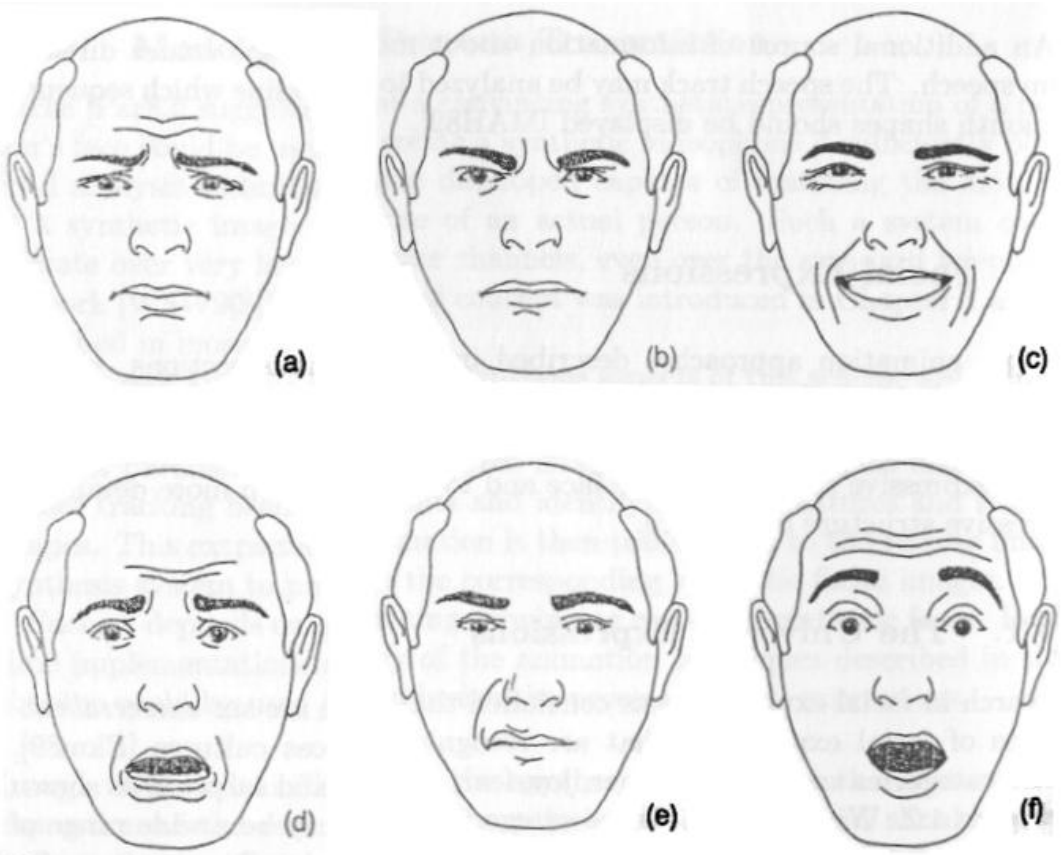
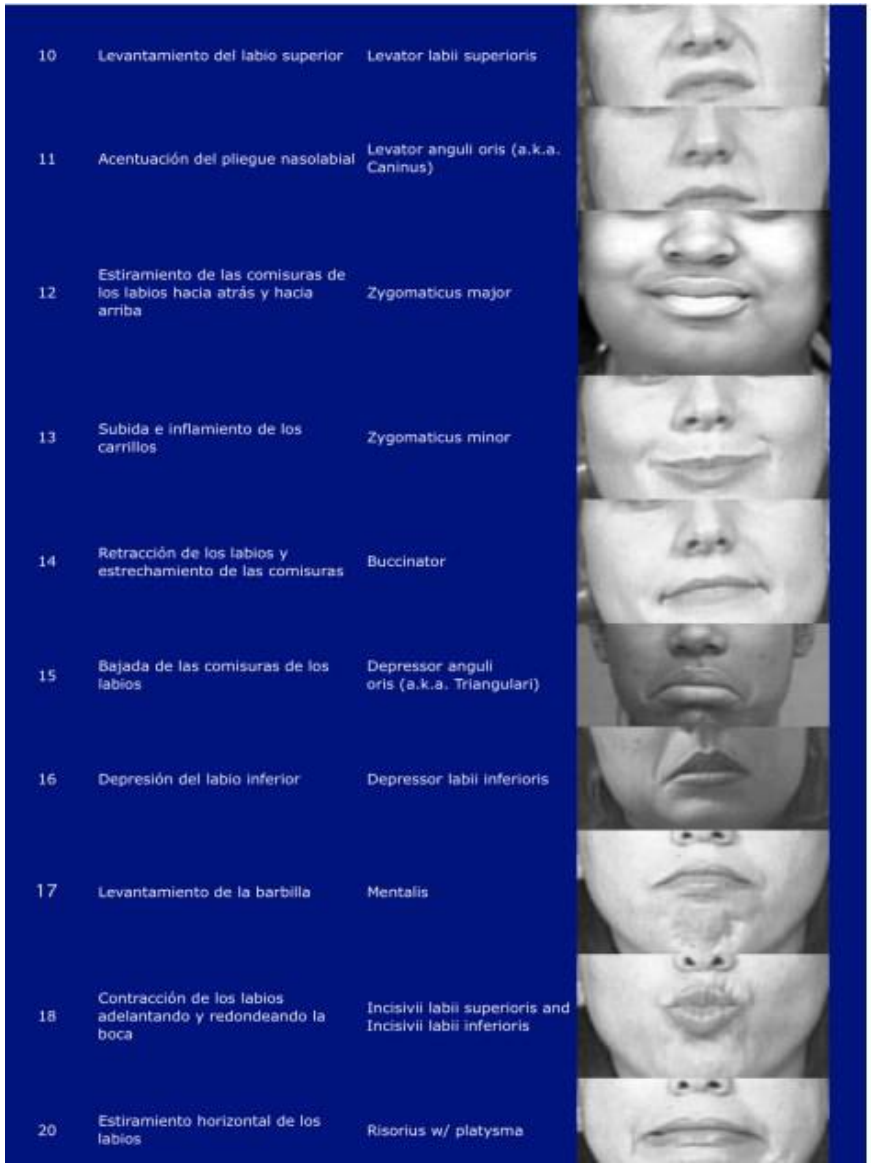


Figure 4.2.
The universal expressions: (a) sadness, (b) anger, (c) joy, (d) fear, (e) disgust, and (f) surprise.

Facial Action Coding System (Ekman)

- Describe un conjunto de "Unidades de Acción" (AUs) que corresponden a acciones básicas (algunos corresponden a músculos individuales, pero otros implican múltiples músculos, o incluso el movimiento articular)
- Ejemplos:
  1. Levantar ceja interior
  2. Levantar ceja exterior
  14. Oyuelos
  17. Levantar barbilla
  19. Sacar la lengua
  20. Estirar los labios
  29. Forzar la mandíbula
  30. Doblar la mandíbula
  31. Apretar la mandíbula

12.1 Descripción de las expresiones faciales



- Las Expresiones son construcciones desde unidades básicas de Acción
- Felicidad:
  1. Levantamiento de la ceja interior (Frontalis, Pars Medialis)
  6. Levantamiento de la mejilla(Orbicularis Oculi, Pars Orbitalis)
  12. Levantamiento de la comisura de los labios hacia atrás y arriba (Zygomatic Major)
  14. Oyuelos (Buccinator)

12.1 Descripción de las expresiones faciales

- Los estados emocionales pueden representarse de manera informal utilizando dos ejes
- X=Happy/Sad
- Y=Excited/Relaxed

Bibliografía

- Libros
  - "The Artist's Complete Guide to Facial Expression" (Faigin)
  - "The Expression of Emotions in Man and Animals" (Darwin)
  - "Computer Facial Animation" (Parke, Waters)

- Artículos
  - "A Survey of Facial Modeling and Animation Techniques" (Noh)

Índice

Método basado en articulaciones

- El uso de las articulaciones y skining para hacer el hueso de la mandíbula y la bola de los ojos expresivos.
- También se puede utilizar un sistema de esqueleto bastante estándar para hacer músculos y deformaciones de la piel facial, utilizando mezcla de pesos en skinning.
- Esto le da una gran cantidad de control y es adecuada para la animación de calidad media.

Método basado en Interpolación de la forma

- Uno de los métodos más populares en la práctica es el uso de la interpolación de la forma (*shape interpolation*).
- Varias expresiones claves diferentes son esculpidas anticipadamente.
- Las expresiones clave se pueden mezclar en el momento para generar una expresión final.
- Se puede interpolar toda la cara (desde alegría a tristeza) o zonas localizadas (párpado izquierdo, cejas, fosa nasal ...).

- La interpolación de la forma permite mezclar entre varias expresiones pre-esculpidas para generar una expresión final.
- Es una técnica muy popular, ya que en última instancia puede dar un control total sobre todos los vértices si es necesario.
- Sin embargo, tiende a requerir una gran cantidad de tiempo
- Se conoce por muchos nombres:
  - Morphing
  - Morph Targets
  - Multi-Target Blending
  - Vertex Blending
  - Geometry Interpolation
  - etc.

- Se comienza con un modelo 3D para la cara en una expresión neutral, conocida como *base*.
- A partir de ahí se crean varios *objetivos* individuales (*morph targets*) moviendo vértices del modelo base.
- La *topología* de las mallas destino debe ser el mismo que el modelo base (es decir, mismo número de vértices y triángulos, y la misma conectividad).
- Cada objetivo está controlado por un GDL Φi que oscilará de 0 (posición base) a 1 (posición objetivo).

- Se necesitan GDLs para controlar la interpolación.
- Estas oscilarán generalmente entre 0 y 1.
- Es por esto que es bueno tener una clase GDLs que pueda ser utilizado por las articulaciones, el *morph targets*, o cualquier otra cosa que se desee animar.
- A alto nivel el código no necesita distinguir entre animar el GDL de un codo o una ceja.

Algoritmo de interpolación

- Para computar una posición de un vértice mezcla:

$$\mathbf{v}' = \mathbf{v}_{base} + \sum \phi_i \cdot (\Delta \mathbf{v}_i) \qquad \text{donde } \Delta \mathbf{v}_i = \mathbf{v}_i - \mathbf{v}_{base}$$

- La posición de mezclado es la posición base más la contribución de cada objetivo cuyo valor de GDL es mayor que 0 (objetivos con un valor de 0 en GDL son esencialmente “no tiene efecto”).
- Si varios objetivos afectan el mismo vértice, sus resultados se combinan de una manera "razonable“.

- Suma ponderada:

$$x' = \sum_{i=0} w_i x_i$$

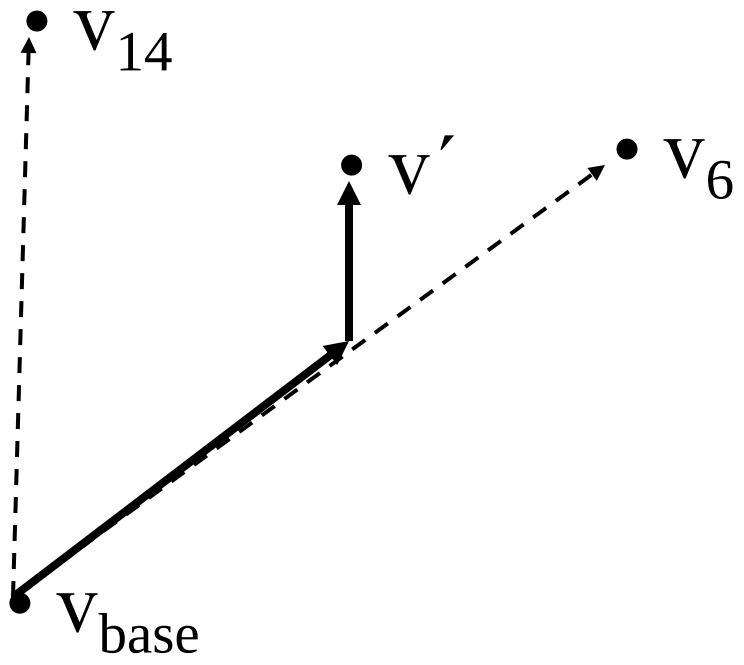
$$\sum_{i=0} w_i = 1$$

$$0 \le w_i \le 1$$

- Mezcla aditiva:

$$x' = x_0 + \sum_{i=1} w_i (x_i - x_0) = \left(1 - \sum_{i=1} w_i\right) x_0 + \sum_{i=1} w_i x_i$$

- Mezcla aditiva



$\Phi_6=0.5$

$\Phi_{14}=0.25$

- Para computar la normal:

$$\mathbf{n}^* = \mathbf{n}_{base} + \sum \phi_i \cdot (\Delta \mathbf{n}_i) \qquad \text{donde } \Delta \mathbf{n}_i = \mathbf{n}_i - \mathbf{n}_{base}$$

$$n' = n^* / |n^*|$$

- Nota: Si la normal va a someterse a una transformación más tarde (es decir, *skinning*), podríamos posponer la etapa de normalización hasta más tarde.

- Para computar una posición de vértice mezclada:

$$\mathbf{v}' = \mathbf{v}_{base} + \sum \phi_i \cdot (\mathbf{v}_i - \mathbf{v}_{base})$$

- La posición de mezclado es la posición de base más la contribución de cada objetivo cuyo valor GDL es mayor que 0.
- Para mezclar las normales, se usa una ecuación similar:

$$\mathbf{n}^* = \mathbf{n}_{base} + \sum \phi_i \cdot (\mathbf{n}_i - \mathbf{n}_{base})$$

- No vamos a normalizarlos ahora, ya que esto se hará más adelante en la fase de *skinning*.

- Por lo general, la interpolación de forma se realiza en el espacio local del skin.
- En otras palabras, se hace *antes* de los cálculos de suaviazado de la piel.

Algoritmo de suavizado de piel

- La posición final del vértice se calcula como la media ponderada respecto a todas las articulaciones a las que está vinculado el vértice. Cada articulación transforma la posición del vértice como si estuviera atado rígidamente a ella y la posición final se calcula utilizando los diferentes pesos.

$$\mathbf{v''} = \sum w_i \, \mathbf{W}_i \cdot \mathbf{B}_i^{-1} \cdot \mathbf{v'}$$

donde:

- $\mathbf{v''}$ es la posición final del vértice en coordenadas de la escena.
- w_i es el peso del vértice sobre la articulación *i*.
- $\mathbf{v'}$ es la posición del vértice en coordenadas locales.
- $\mathbf{B}_i$ es la matriz de adyacencia (La matriz mundo de la articulación *i* en la posición inicial).
- $\mathbf{W}_i$ es la matriz mundo de la articulación *i* tras el posicionamiento del esqueleto.

Algoritmo de suavizado de piel

- Nota:
  - **B** es constante así que $\mathbf{B}^{-1}$ puede precalcularse.
  - $\mathbf{W} \cdot \mathbf{B}^{-1}$ puede calcularse para cada articulación antes de ejecutar el algoritmo de skinning.
- La suma de los pesos debe ser 1.

$$\sum w_i = 1$$

Algoritmo de suavizado de piel

- Con las normales es esencialmente lo mismo, excepto que lo transformamos como direcciones (x,y,z,0) y renormalizamos los resultados.

$$\mathbf{n^*} = \sum w_i \, \mathbf{W}_i \cdot \mathbf{B}_i^{-1} \cdot \mathbf{n'}$$

$$n'' = n^* / \, |n^*|$$

- Skeleton

$$\mathbf{L} = \mathbf{L}_{\mathbf{joint}}(\phi_1, \phi_2, \ldots, \phi_N)$$
$$\mathbf{W} = \mathbf{W}_{\mathbf{parent}} \cdot \mathbf{L}$$

- Morphing

$$\mathbf{v}' = \mathbf{v}_{base} + \sum \phi_i \cdot (\mathbf{v}_i - \mathbf{v}_{base})$$
$$\mathbf{n}' = \mathbf{n}_{base} + \sum \phi_i \cdot (\mathbf{n}_i - \mathbf{n}_{base})$$

- Skinning

$$\mathbf{v}'' = \sum w_i \, \mathbf{W}_i \cdot \mathbf{B}_i^{-1} \cdot \mathbf{v}'$$
$$\mathbf{n}^* = \sum w_i \, \mathbf{W}_i \cdot \mathbf{B}_i^{-1} \cdot \mathbf{n}'$$
$$\mathbf{n}'' = \mathbf{n}^* / |\mathbf{n}^*|$$

Uso de la memoria

- La interpolación de forma puede ocupar una gran cantidad de memoria. Este es un gran problema para los videojuegos, pero menos en las películas.
- El modelo base se almacena normalmente de cualquier manera a como un modelo 3D sería almacenado internamente (vértices, normales, triángulos, mapas de texturas, coordenadas de textura ...)
- Los targets, sin embargo, no necesitan toda esa información, ya que gran parte de ella se mantendrá constante (triángulos, mapas de texturas ...)

Uso de la memoria

- Además, en la mayoría de las expresiones de destino (target expressions) sólo se modificarán un pequeño porcentaje de los vértices.
- Por lo tanto, los targets realmente sólo necesitan almacenar las posiciones y las normales de los vértices que se han alejado de la posición de base (y los índices de esos vértices).

Uso de la memoria

- Además, no se necesita almacenar la posición y la normal entera, sólo la diferencia desde la posición y la normal base.
- Es decir, no almacenar v3, almacenamos v3-vbase
- Hay dos principales ventajas para hacer esto:
  - Menos restas de vectores en tiempo de ejecución (ahorra tiempo)
  - Como los deltas serán típicamente más pequeños, debemos ser capaces de obtener una mejor compresión (ahorro de espacio)

- En la etapa de preprocesado los valores se obtienen comparando el modelo modificado con el modelo base y almacenando la diferencia.
- La información puede almacenarse en algo de este tipo:

```
class MorphTarget {
        int NumVerts;
        int Index [ ];
        Vector3 DeltaPosition [ ];
        Vector3 DeltaNormal [ ];
}
```

Otras propiedades

- Además de interpolar las posiciones y las normales también se pueden interpolar otros atributos de los vértices:
  - Colores
  - Opacidad (canal alpha)
  - Coordenadas de textura
  - Otras propiedades auxiliares

Expresión vascular

- La expresión vascular es un término para describir sonrojos y otros fenómenos relacionados con el cambio de color en la cara.
- La adición de cambios sutiles en el color facial que se relacionan con la distorsión de la piel puede ayudar a mejorar el realismo.
- Esto se puede lograr por mezcla de los valores de color con cada vértice (junto con la posición y normal)
- Alternativamente, se podría usar un mapa de textura de "rubor" controlado por un valor de intensidad de mezclado en cada vértice.

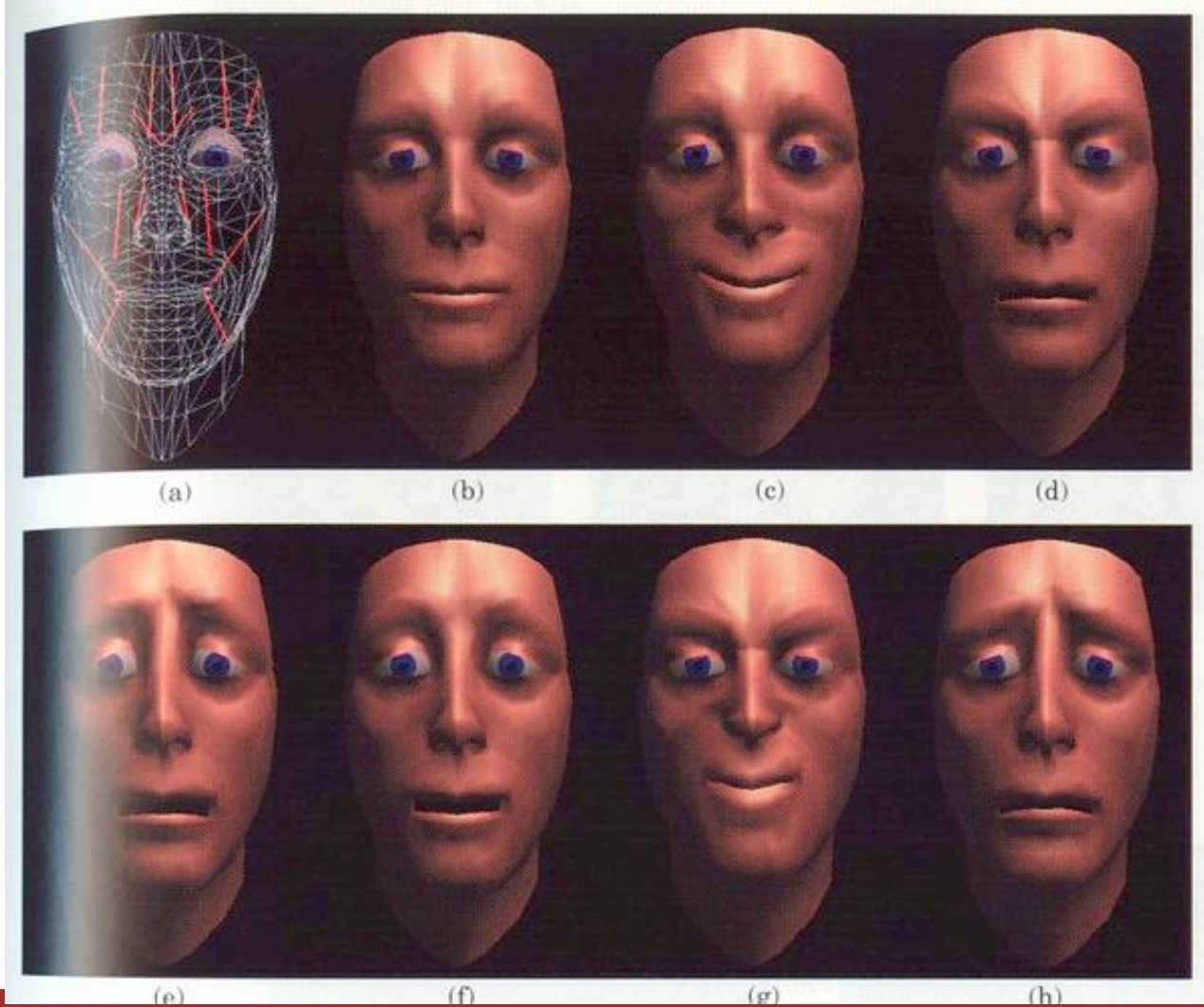
Arrugas

- Una aplicación de interpolación de datos auxiliar es la adición de arrugas.
- Cada vértice almacena una propiedad auxiliar que indica la forma en que la zona está arrugada.
- En el modelo base esta propiedad sería probablemente 0 en la mayoría de los vértices, que indica el estado sin arrugas.
- Expresiones de destino (target expression) pueden tener este conjunto de propiedades cerca de 1 en las zonas arrugadas.
- Cuando se mezclan las expresiones faciales, esta propiedad se mezcla por vértice, al igual que las posiciones y normales (pero debe ser fijado entre 0 y 1 cuando se hace).
- Para renderizar, este valor se utiliza como un factor de escalado sobre un mapa de textura de "arrugas" que se mezcla con el principal de textura de la cara.
- Aún mejor, se podría utilizar un mapa de protuberancia de arrugas o un mapa de desplazamiento.

Métodos de Músculos Artificiales

- Con esta técnica, los músculos se modelan como deformaciones que afectan a las regiones locales de la cara.
- Las deformaciones se pueden construir a partir de unas instrucciones sencillas, las articulaciones, targets de interpolación, FFDS, u otras técnicas.

Métodos de Músculos Artificiales

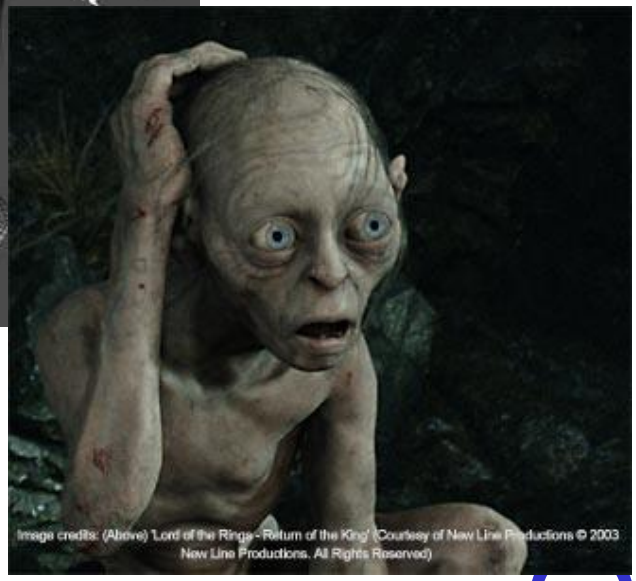


Características Faciales

- Claves en las características faciales:
  - Piel deformable
  - Pelo
  - Ojos
  - Mandíbula articulada (dientes…)
  - Lengua
  - Interior de la boca
- Cada una de estas puede requerir una estrategia o técnica diferente.

Captura de movimiento

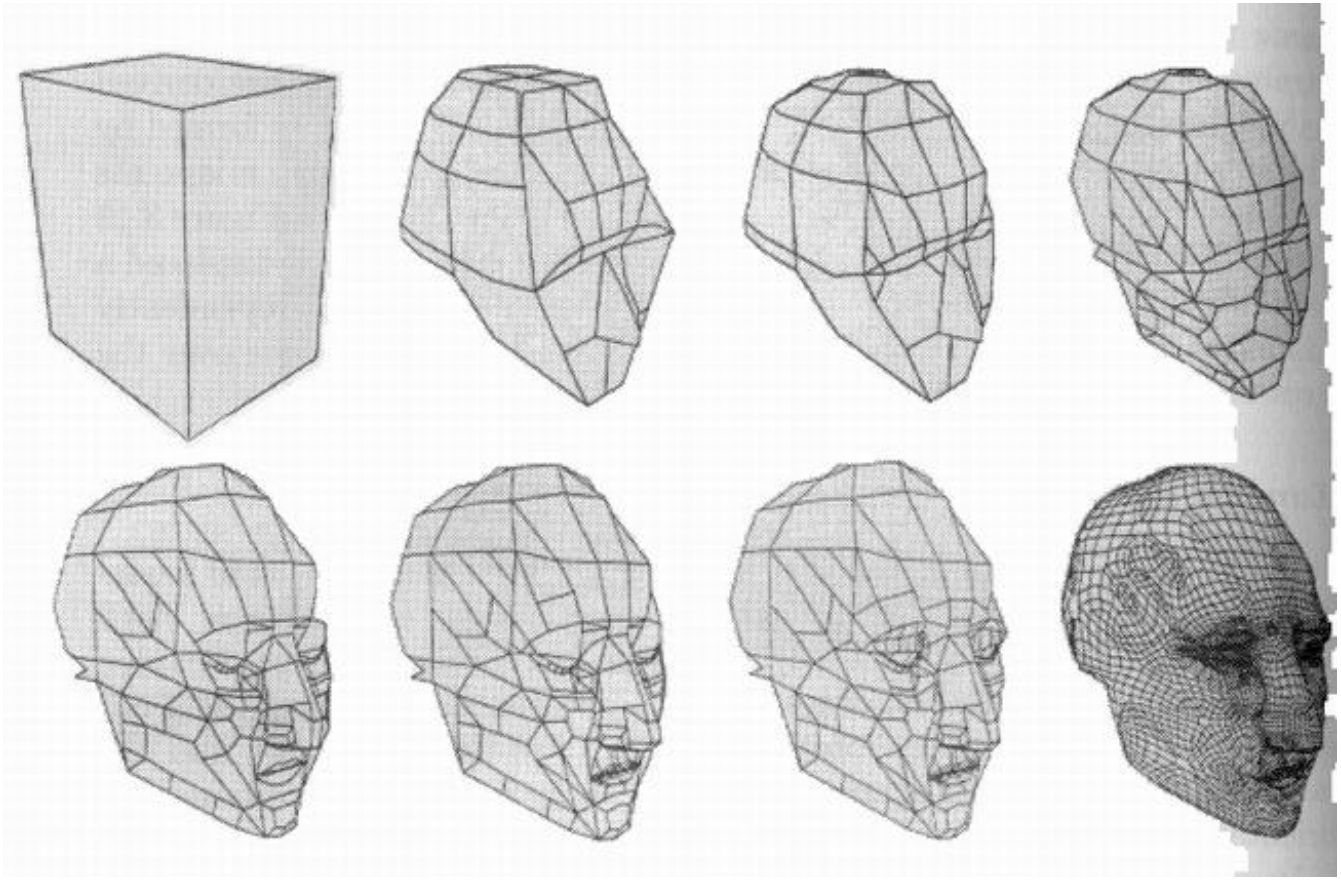
Captura de movimiento



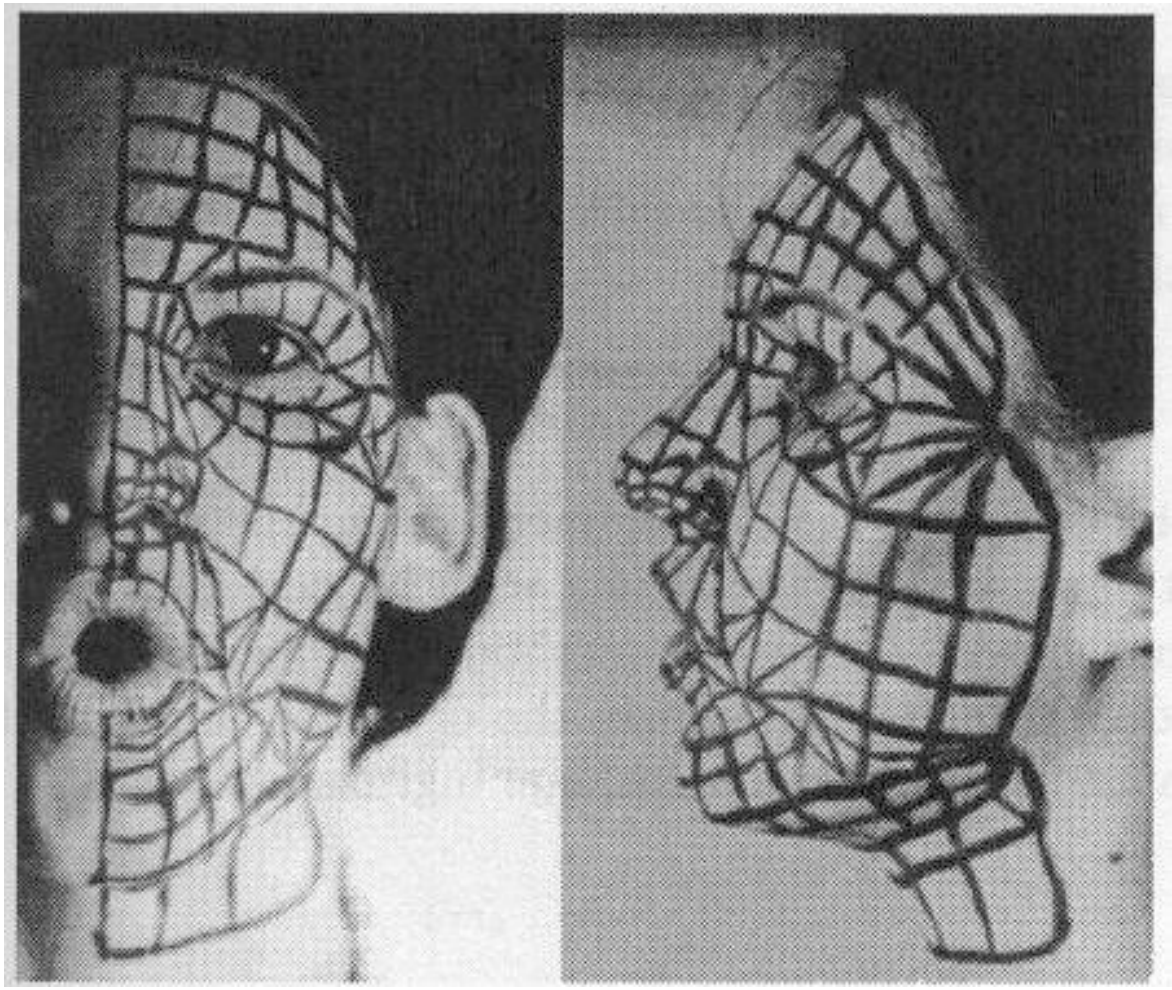
Índice

- La preparación de la geometría facial y todas las expresiones necesarias pueden requerir bastante trabajo.
- Hay varias categorías de técnicas para modelado facial:
  - Modelado Tradicional (en un modelador interactivo 3D)
  - Fotografiar y digitalizar (en 2D con un ratón)
  - Esculpir y digitalizar (con un digitalizador 3D)
  - Escaneado láser
  - Visión (imagen 2D o vídeo)

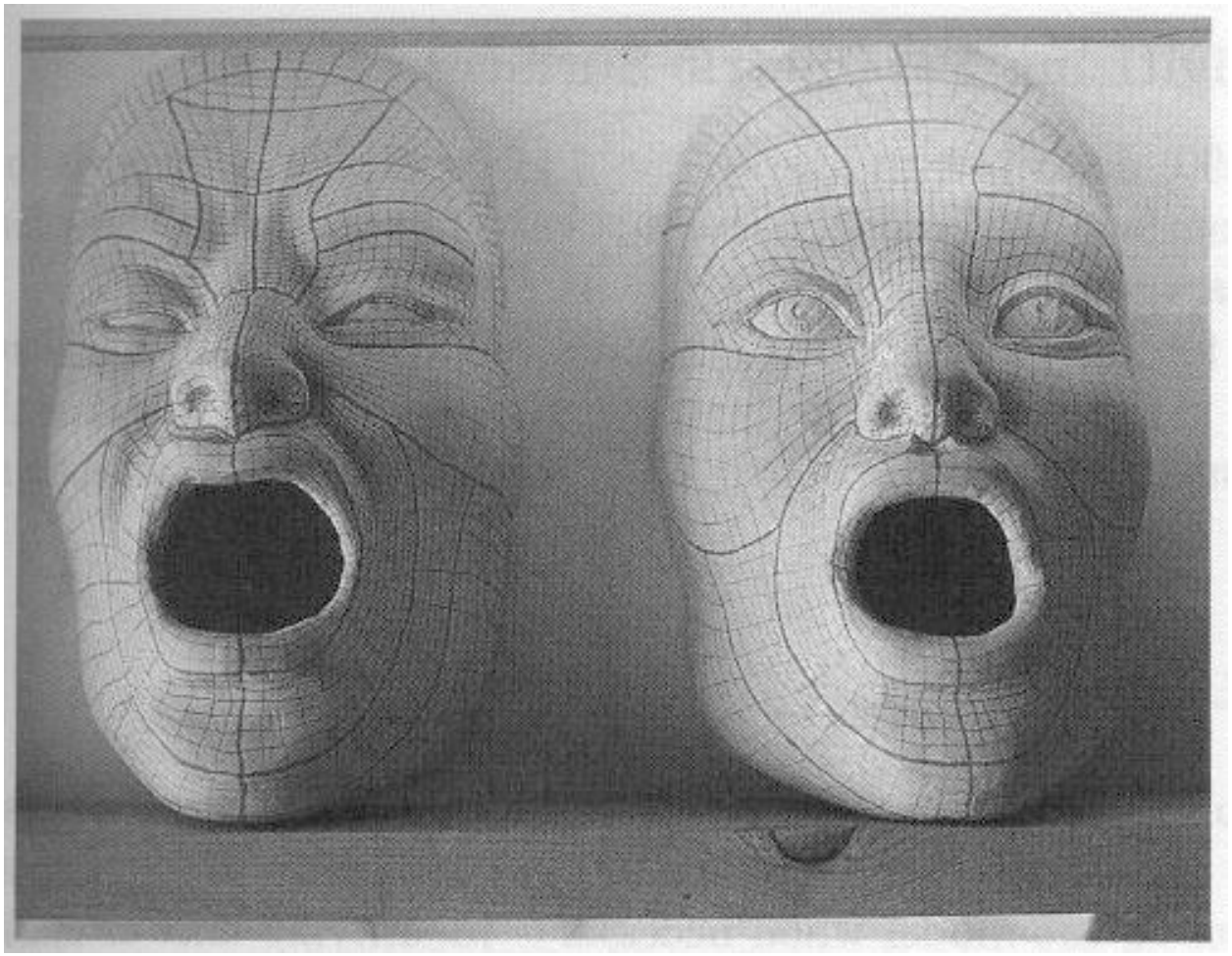
Modelado tradicional



Fotografiar y digitalizar

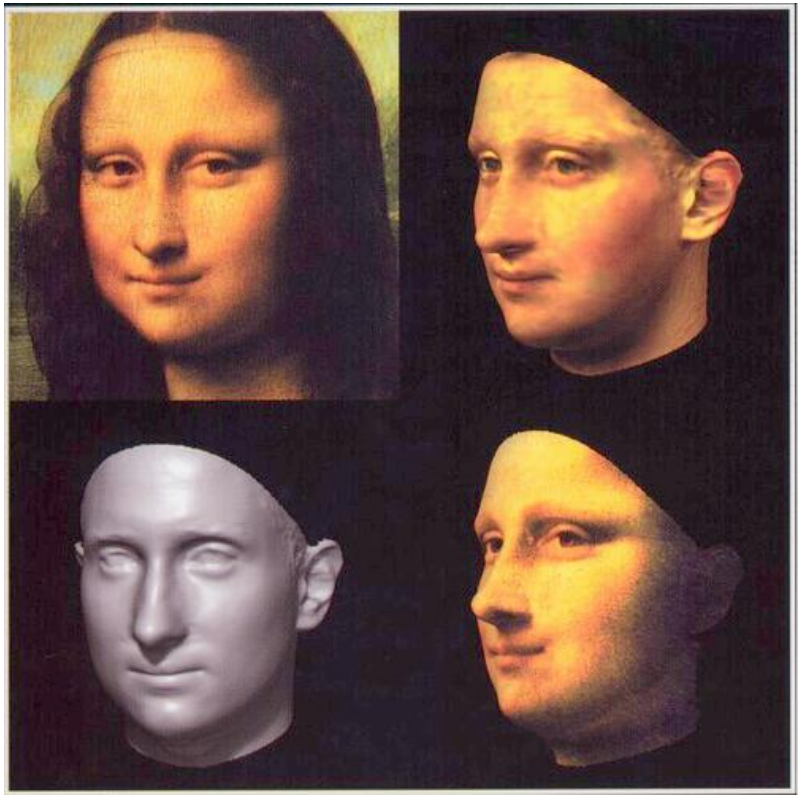


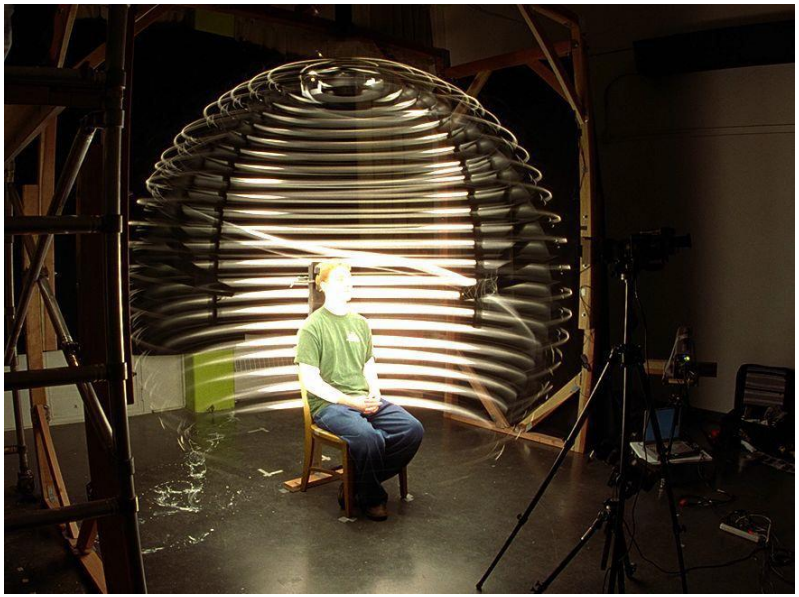
Esculpir y digitalizar

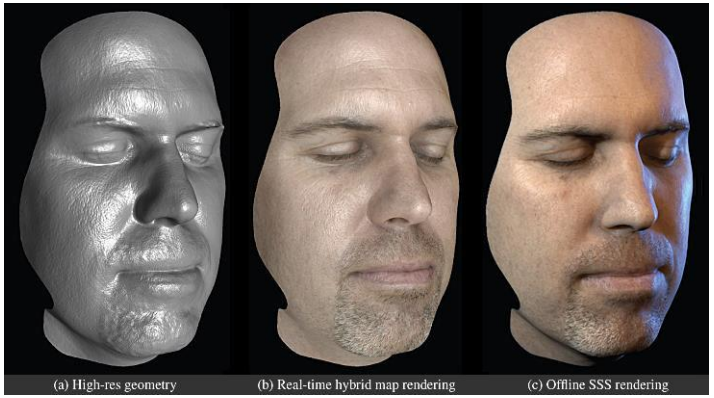


Escaneado láser

Técnicas de visión por computador



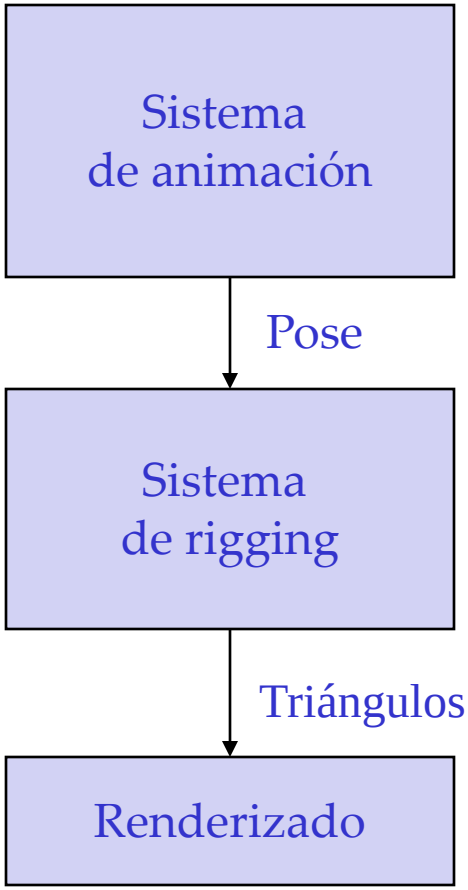




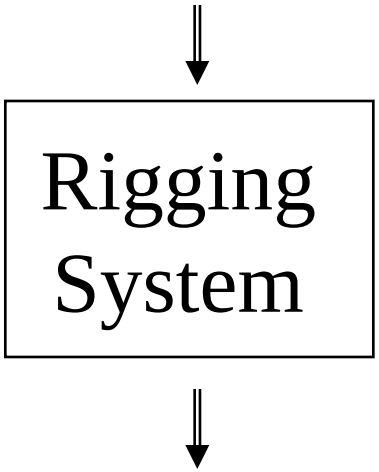
Índice

- Un rig es como una marioneta virtual.
- Un contenedor rig tiene varios grados de libertad, cada uno correspondiente al parámetro animatable dentro del títere.
- Los GDLs pueden controlar:
  - Rotaciones articuladas, traslaciones
  - Morph targets
  - Otras cosas ...
- En el código de mas alto nivel se especifican los valores de los grados de libertad (es decir, la pose del rig).

- En última instancia, el rig toma los valores de GDL del sistema de animación y genera la pose geométrica que representa el personaje en el espacio mundo.
- Esto implica:
  - Computar las matrices mundo de la articulación (pose del esqueleto)
  - Interpolar los vertices en el espacio local (morphing)
  - Transformar los vertices al espacio mundo (skining)
- Esta geometría es entonces renderizada a través de un sistema de renderizado (OpenGL, Vulkan, Direct3D).


Sistema
de animación
Pose
Sistema
de rigging
Triángulos
Renderizado

$$\mathbf{\Phi} = \begin{bmatrix} \phi_1 & \phi_2 & \dots & \phi_N \end{bmatrix}$$



$$\mathbf{v}'', \mathbf{n}''$$

Datos de esqueleto, forma y skin

$$[\phi_1 \quad \phi_2 \quad \dots \quad \phi_M] \qquad [\phi_{M+1} \quad \phi_{M+2} \quad \dots \quad \phi_N]$$

$$\mathbf{L} = \mathbf{L}_{jnt}(\phi_1, \phi_2, \dots, \phi_m)$$
$$\mathbf{W} = \mathbf{L} \cdot \mathbf{W}_{parent}$$

$$\mathbf{v}' = \mathbf{v}_{base} + \sum \phi_i \cdot (\mathbf{v}_i - \mathbf{v}_{base})$$
$$\mathbf{n}' = \mathbf{n}_{base} + \sum \phi_i \cdot (\mathbf{n}_i - \mathbf{n}_{base})$$

$$\mathbf{v}'' = \sum w_i \mathbf{v}' \cdot \mathbf{B}_i^{-1} \cdot \mathbf{W}_i$$
$$\mathbf{n}^* = \sum w_i \mathbf{n}' \cdot \mathbf{B}_i^{-1} \cdot \mathbf{W}_i$$
$$\mathbf{n}'' = \frac{\mathbf{n}^*}{|\mathbf{n}^*|}$$

$$\mathbf{v}'', \mathbf{n}''$$

Aproximación por capas

- Se usa un enfoque por capas
  - Cinemática del Esqueleto
  - Interpolación de la Forma
  - Suavizado del Skinning
- La mayoría de los sistemas de rigging de personajes se basan en algún tipo de sistema de enfoque por capas combinados con el data flow para permitir la personalización.

- Skeleton

$$\mathbf{L} = \mathbf{L}_{jnt}\left(\phi_1, \phi_2, \ldots, \phi_N\right)$$

$$\mathbf{W} = \mathbf{L} \cdot \mathbf{W}_{parent}$$

- Morphing

$$\mathbf{v}' = \mathbf{v}_{base} + \sum \phi_i \cdot \left(\mathbf{v}_i - \mathbf{v}_{base}\right)$$

$$\mathbf{n}' = \mathbf{n}_{base} + \sum \phi_i \cdot \left(\mathbf{n}_i - \mathbf{n}_{base}\right)$$

- Skinning

$$\mathbf{v}'' = \sum w_i \mathbf{v}' \cdot \mathbf{B}_i^{-1} \cdot \mathbf{W}_i$$

$$\mathbf{n}^* = \sum w_i \mathbf{n}' \cdot \mathbf{B}_i^{-1} \cdot \mathbf{W}_i$$

$$\mathbf{n}'' = \frac{\mathbf{n}^*}{\left|\mathbf{n}^*\right|}$$

Índice

Tipos de Grados de Libertad

- Además de controlar las articulaciones y la forma, los GDL pueden ser extendidos para manipular cualquier parámetro de alto nivel que el animador quiera controlar
- Se podría hacer GDL para
  - Tornar el personaje a verde
  - Extender / Flexionar todos los dedos en una mano de forma simultánea
  - Alzar el pelo del personaje
  - Morph el personaje desde un hombre a un monstruo peludo
  - Controlar la intensidad de una luz
  - Controlar la velocidad de creación de un sistema de partículas

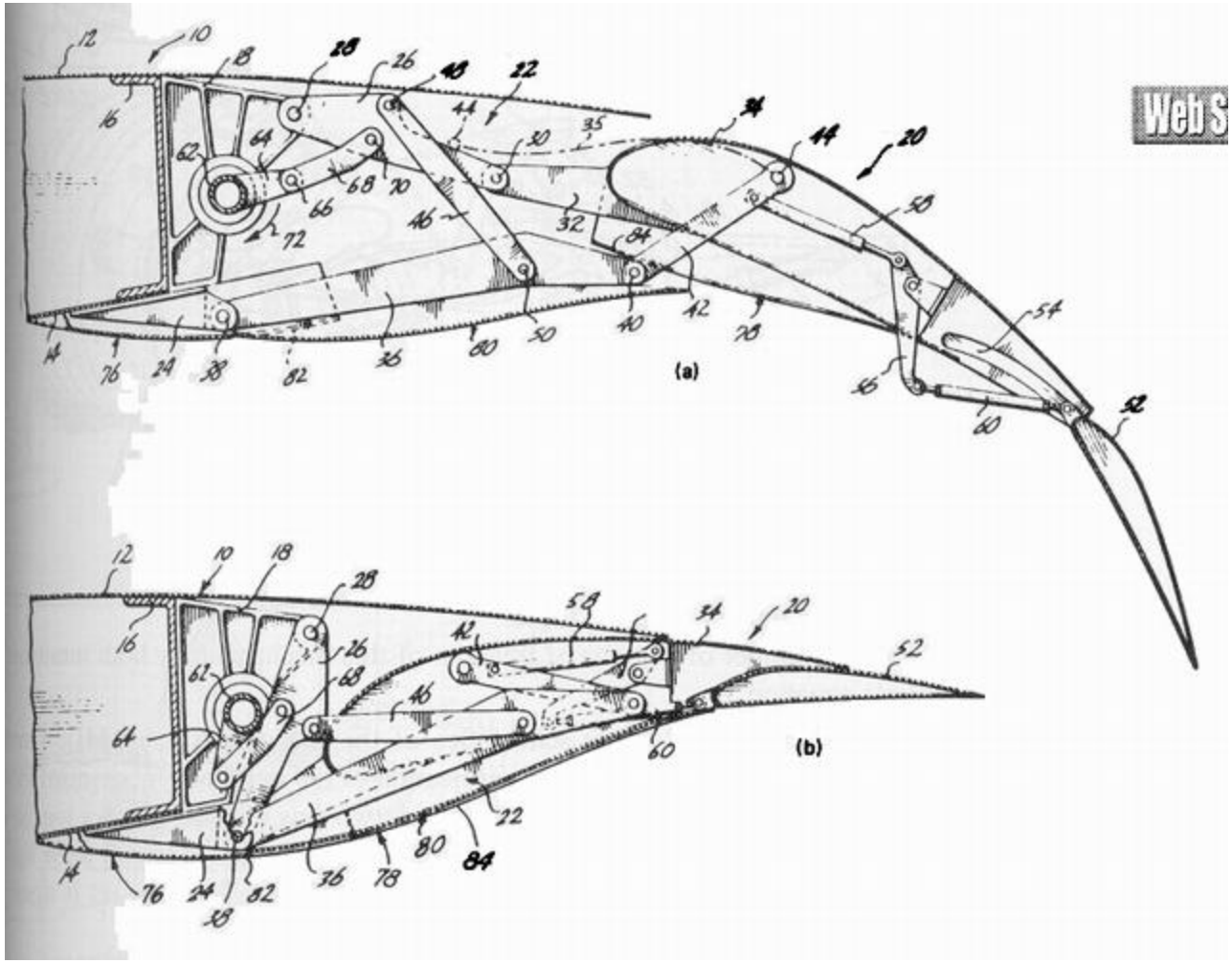
Morphing de cuerpo completo

- Se puede tambien riggear un GDL para hacer morphing a un personaje entero por ejemplo de un humano a un monstruo gigante peludo.
- El Morphing es mas fácil si la topología de los dos personajes coinciden (ambas topologías skeleton & skin)
- Para hacer esto se deben interpolar varios datos:
  - Posiciones del skin y las normales
  - Pesos del skin y datos adjuntos
  - Offset de hueso
  - Mapas de textura, otras propiedades visuales
  - Otras cosas ...

Agrupando GDLs

- Se puede tener un GDL para controlar varias propiedades. Por ejemplo:
  - GDL dedo que hace que todos las articulaciones de un dedo se flexione o extienda simultáneamente.
  - GDL codo que controla tanto la rotación del codo y un morph target para la deformación de bíceps.
  - GDL cabeza que gira varias vértebras en el cuello.
  - GDL para controlar un sistema mecánico muy restringido.
  - GDL sonreír que controla varios músculos individuales
- Para mayor flexibilidad se pueden utilizar expresiones …

Agrupando GDLs



Expresiones de GDL

- Para mayor flexibilidad, es bueno ser capaz de ejecutar expresiones arbitrarias con valores de GDL.
- Una expresión toma uno o más grados de libertad como entradas y establece el GDL externo como de salida.
- Una expresión puede ser, literalmente, cualquier expresión matemática:

$$DOF[27] = DOF[3] * 6.0 \sin (DOF[2]) + DOF[14]$$

- En lugar de ser codificados de forma rígida en C++, es bueno que las expresiones pueden ser interpretados en tiempo de ejecución.

Mapeo de GDL

- Un rig puede ser implementado como un array de punteros a GDLs.
- El orden de los GDLs es importante y debe ser consistente entre los sistemas de rigging y animación.
- Los GDLs mismos existen como objetos internos usados en el esqueleto, el sistema de Morphy y en expresiones.
- Normalmente, tendríamos una única plataforma que controla todos los grados de libertad del personaje que deseamos animar.
- Alternativamente, se podría:
  - Tener un rig que controla un subconjunto de GDLs de un personaje.
  - Tener un rig que se asigna a más de un personaje.
  - Tienen varios rigs diferentes para el mismo personaje que se utilizará para diferentes propósitos.

Push vs. Pull

- Hay diferentes formas de implementar rigging.
- Un enfoque es tener datos de los GDLs y asignarlos al rig (*Push*). La otra es obtener datos de los GDLs del rig (*Pull*). Hay varios otros detalles sutiles.
- *Pushing* es probablemente mejor, ya que permite múltiples configuraciones de rig de manera que se asignan al mismo GDL un poco más fácil.

Pushing

- Los GDLs se almacenan dentro de los componentes reales que los utilizan (articulaciones del esqueleto, los controles para los morphs targets, GDLs ficticios para los controles de alto nivel).
- Actualización:
  1. El sistema de animación genera un vector pose y se lo pasa al *rig*.
  2. El *rig* toma los floats desde la pose y coloca (*pushes*) los valores de sus grados de libertad (DOF[i]->SetValue(Pose[i]))
  3. El *rig* luego ejecuta todas las expresiones (que se deben ejecutar en el orden correcto). Las expresiones a continuación, establecen valores en otros grados de libertad (*push*).
  4. El esqueleto, morphing, y el skinning se ejecutan utilizando sus valores de GDL.

Rigging minimalista

- Es buena idea utilizar el menor número de grados de libertad como sea posible al *riggear* un personaje.
- Algunas razones incluyen:
  - Mantener la interfaz mas simple para controlar el carácter. Esto hace la vida de la animación más fácil.
  - Reduce la cantidad de datos de animación necesarios para la reproducción. Esto es importante en los videojuegos, ya que los datos de animación tiende a ocupar una gran cantidad de espacio.