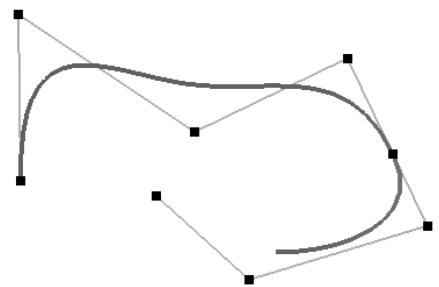
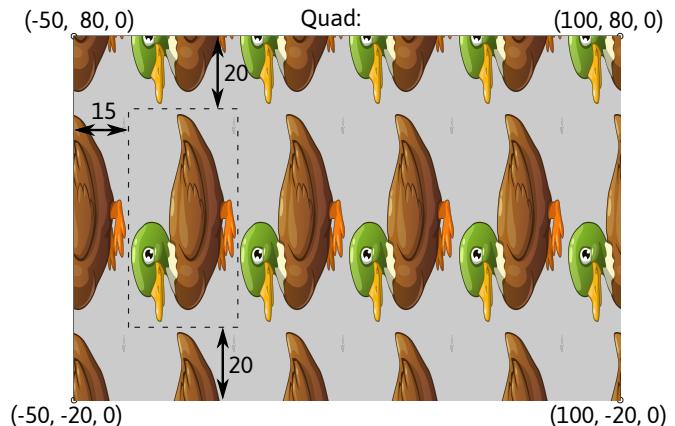


**Ejercicio X (25pts):** Dada la B-Spline de la figura: **a)** Proponga dos posibles vectores de knots. ¿Cuáles son los límites del parámetro  $t$  en cada caso? **b)** Analice los grados de continuidades (geométrica y paramétrica) a lo largo de la curva (señale las condiciones generales y los puntos donde ocurren cosas particulares). Nota: todos los puntos remarcados son puntos de control.



**Ejercicio X (15pts):** ¿Qué es una subdivision surface? ¿Cuáles son las ventajas y desventajas de utilizar este tipo de superficies frente a superficies NURBS o de Bezier? (ayuda: considere en la 2da respuesta, además de las diferencias más obvias, las ventajas/desventajas a la hora de aplicarles texturas o calcularles iluminación).

**Ejercicio X (20pts):** **a)** Defina un mapeo manual para el ejemplo de la figura. A la izquierda se muestra la textura, a la derecha un quad con las coordenadas de sus vértices y la textura aplicada. **b)** Suponga un fragmento que tiene coordenadas de textura  $s=-0.207$ ,  $t=1.159$ . Los modos de repetición están seteados en GL\_REPEAT para ambas coordenadas. La textura es de 100x100 téxeles, el modo de aplicación es GL\_REPLACE, y los filtros en GL\_LINEAR. Exprese la fórmula para obtener el color final del fragmento, en función del contenido de la textura. Nota: reemplace todas las variables que conozca por sus valores y deje expresado el color de un texel de la fila  $i$ , columna  $j$  de la imagen como  $C_{i,j}$ .



**Ejercicio X (15pts):** Si tengo un diagrama de Voronoï, **a)** ¿Cómo construyo la triangulación de Delaunay para ese mismo conjunto de puntos? **b)** Una vez obtenida la triangulación, ¿cómo podría verificar si el diagrama era correcto? (cómo verificar si la triangulación obtenida es efectivamente Delaunay).

**Ejercicio X (25pts):** Dado el siguiente algoritmo de renderizado:

1. Limpiar los buffers de color (en negro) y Z (en zfar)
2. Dibujar los "objetos" opacos
3. Enmascarar el depth-buffer
4. Dibujar los objetos semi-transparentes

Este algoritmo "disimula" los problemas del z-buffer con las transparencias; pero si hay transparencias múltiples el resultado todavía puede no ser correcto. Responda: **a)** En los pasos 2 y 4, ¿cómo podemos separar automáticamente las "cosas" opacas de las semi-transparentes? **b)** ¿Cuál es el problema del algoritmo del z-buffer con las transparencias? ¿Por qué la solución propuesta no es 100% correcta cuando hay transparencias múltiples? **c)** Escriba un algoritmo (puede ser pseudocódigo, no hace falta sintaxis correcta) que permita "ver" las zonas donde el problema podría manifestarse: pintar de rojo fuerte los fragmentos donde se han renderizado transparencias múltiples, y oscurecer los demás.