LABORATORIO 1

Il programma supporta diverse modalità di disegno delle curve di Bezier, incluse una modalità predefinita, Catmull-Rom, Adattiva e Continuità.

Le funzioni che ho implementato nel programma sono:

`void myMouseFunc(int button, int state, int x, int y)`: Questa funzione gestisce l'input del mouse. Quando si preme il pulsante sinistro del mouse, viene posizionato un nuovo punto di controllo. Se un punto esistente è sufficientemente vicino al punto del clic, quel punto esistente viene selezionato invece di aggiungere un nuovo punto.

`void motion(int x, int y)`: Questa funzione gestisce l'input del mouse quando il tasto sinistro è premuto e il mouse viene spostato. Se un punto di controllo è stato selezionato, il suo posizionamento viene aggiornato in base al movimento del mouse.

`vec2 deCasteljau(float Array[][2], float t, int NumPts, float fArray[][2] = nullptr, float sArray[][2] = nullptr)`: Questa funzione implementa l'algoritmo di de Casteljau per calcolare il punto sulla curva di Bezier data la matrice dei punti di controllo e il parametro t.

Le funzioni `defaultModality`, `suddivisioneAdattiva`, `catmullRom` e `continuity` implementano le diverse modalità di disegno delle curve in base all'input dell'utente.

Il programma principale inizializza la finestra di visualizzazione OpenGL e entra nel loop principale di GLUT per gestire gli eventi e il rendering delle scene.

Il contenuto principale del lab risiede nelle 4 funzioni `defaultModality`, `suddivisioneAdattiva`, `catmullRom` e `continuity`

`void defaultModality(float tempArray[][2], int NumPts)`: Questa funzione calcola i punti sulla curva di Bezier utilizzando l'algoritmo di de Casteljau. Prende in input un array di punti temporanei `tempArray` e il numero di punti di controllo `NumPts`. Utilizza il parametro `resolution` per specificare quanti punti devono essere calcolati sulla curva e salva i risultati nell'array `CurveArray`. La curva viene quindi disegnata con la funzione `drawArrayData`.

`void suddivisioneAdattiva(float Array[][2], int NumPts)`: Questa funzione implementa l'algoritmo di suddivisione adattiva per disegnare curve di Bezier. Prende in input un array di punti `Array` e il numero di punti di controllo `NumPts`. L'algoritmo inizia controllando se i punti di controllo sono sufficientemente vicini a formare una curva approssimata da una linea retta (test di planarità). Se la curva è sufficientemente piatta, vengono disegnati i segmenti tra i punti di controllo estremi e la funzione termina. Altrimenti, l'algoritmo applica ricorsivamente l'algoritmo di de Casteljau al punto medio e crea due nuovi insiemi di punti di controllo. Questo processo di suddivisione e controllo della planarità continua fino a quando la curva approssimata è sufficientemente piatta. La variabile `resolution` tiene traccia del

numero di punti calcolati sulla curva, e alla fine la curva viene disegnata con la funzione `drawArrayData`.

`void catmullRom(float Array[MaxNumPts][2], int NumPts)`: Questa funzione implementa il metodo Catmull-Rom per disegnare curve di Bezier. Il metodo Catmull-Rom è una particolare forma di curva di Bezier che interpola tutti i punti di controllo. La funzione inizia creando nuovi punti di controllo intermedi tra i punti di controllo esistenti. Quindi applica l'algoritmo di suddivisione adattiva ai nuovi punti di controllo intermedi per disegnare la curva Catmull-Rom. Come nel caso precedente la variabile `resolution` tiene traccia del numero di punti calcolati sulla curva, e alla fine la curva viene disegnata con la funzione `drawArrayData`.

`void continuity(float Array[MaxNumPts][2], int NumPts)`: Questa funzione gestisce il comportamento di continuità della curva. La funzione utilizza `continuityType`, che può essere `C0`, `C1` o `G1`, per determinare il tipo di continuità da applicare. Se `continuityType` è `C1`, vengono modificati i punti di controllo in modo che la curva abbia una derivata continua, mentre se `continuityType` è `G1`, i punti di controllo vengono modificati in modo si abbiano direzioni tangenti uguali ma di modulo qualunque. La funzione utilizza quindi l'algoritmo di suddivisione adattiva per disegnare la curva con i punti di controllo modificati.