geometry.h

Boston Brooks

Work in progress
Not fully tested

int matrix_multiply (float A[4][4], float B[4][4], float C[4][4]);

This function takes a reference to two 4 by 4 matrices and stores their product, by reference, in a third matrix.

load_elevations();

This function loads an image to CPU memory, extracts an array of elevation data for the game map, then frees the image from memory. The x dimension of the image corresponds to the i dimension of the map, and the y dimension of the image corresponds to the j dimension of the map. The 2 dimensional array of values for the elevation data/height map is 3 elements larger than the game map in length and width. First because for example, an 8 by 8 square chess board is made up of a 9 by 9 array of vertices. Second because we have added a 1 square buffer of map tiles to be used for bi-linear interpolation.

Float_3d Normalise_Float_3d (Float_3d foo);

This function takes a vector of 3 floating point numbers and returns a similar vector, with a length of 1

int tile_coords_get_elevation(int i, int j);

This function returns the elevation on the map corresponding to the the tile coordinates i and j, where (0, 0) is actually given by elevations[1][1]. See load_elevations(); Undefined values return 0.

int floordiv (int a , int b);

This function computes a divided by b, always rounding towards negative infinity.

Tile_Coords map_to_tile (Map_Coords map);

This function returns the coordinates of the tile that contains the point given by the map coordinates.

Screen_Coords map_to_screen_top_left (Map_Coords mc);

This function computes the screen coordinates (x, y) of a point in 3d space (i, j, k), given that (0, 0, 0) corresponds to (0, 0). This function will be used to generate vertex arrays for plotting the curved ground surface of the map to the screen.

Screen_Coords map_to_screen_centre_viewpoint (Map_Coords mc);

This function computes the screen coordinates (x, y) of a point in 3d space (i, j, k), given that the global value “Map_Coords viewpoint;” corresponds to the centre of the screen. This function will be used to decide where to draw sprites to the screen.

Map_Coords screen_to_map_k_0 (Screen_Coords sc);

This function calculates the point in 3d space corresponding to a point on the screen, given that the elevation, k, is equal to 0. This gives the northernmost point on the map that could correspond to the point on screen.

Map_Coords screen_to_map_k_max (Screen_Coords sc);

This function calculates the point in 3d space corresponding to a point on the screen, given that the elevation, k, takes on its maximum value. This gives the southernmost point on the map that could correspond to the point on screen.

int interpolate_elevation (int i, int j);

This function returns the elevation (k) given a point on the map (i, j), by using linear interpolation given the three nearest vertices.

int map_coords_update_elevations(Map_Coords* map_coords);

This function updates the elevation (k) of a 3d vector, (i, j, k), so that the vector lies on the ground surface of the map.

int point_within_triangle_screen_coords (Screen_Coords point, Screen_Coords vertex1, Screen_Coords vertex2, Screen_Coords vertex3);

This function returns 1 if a point on the screen is within the triangle on the screen where the vertices of the triangle correspond to the points in 3d space.

Map_Coords interpolate_map_coords (Screen_Coords p, Map_Coords vertex1, Map_Coords vertex2, Map_Coords vertex3);

This function translates a point on the screen (x, y) to a point in 3d space (i, j, k), given that the point lies on a flat surface in 3d given by 3 3d vertices.

int point_within_tile (Tile_Coords tile, Screen_Coords p);

This function returns 1 if the point on screen is within the area of the northernmost half of the map tile, as it is displayed on screen.

This function returns 2 if the point on screen is within the area of the southernmost half of the map tile, as it is displayed on screen.

Otherwise, this function returns 0.

Map_Coords screen_to_map_centre_viewpoint (Screen_Coords point);

This function returns the point, in 3d space, of the ground surface, corresponding to a point on screen.

int update_bicubic_coefficients(Tile_Coords tile);

This function updates the values of the coefficients used for bi-linear interpolation of the ground surface.

int map_coords_get_elevation_bicubic(Map_Coords mc);

This function calculates the elevation of a point on the ground surface, using bicubic interpolation.

Float_3d map_coords_get_normal_bicubic(Map_Coords mc);

This function calculates a normal vector to a point on the surface using bicubic interpolation. This will be used for calculation hill shading.