Raycasting Engine

Release 1.0.0

Filip Chrzanowski

Jun 13, 2023

CONTENTS:

CHAPTER

ONE

RAYCASTING-ENGINE-PYTHON

1.1 GameEngine module

class GameEngine.**GameEngine**(*width*, *height*, *level_map*, *player*, *window*, *textures*, *sprites*)

Bases: object

render(*canvas*)

rendering all entities :param canvas: canvas containing all objects on the screen :return canvas:

update()

executing commands, updating all entities in a game and checking collisions

1.2 InputHandler module

class InputHandler.**InputHandler**(*window*, *actor*)

Bases: object

class for handling input from keyboard

handle_input()

this method checks which key is present in input_buffer and returns command_buffer array that contains command

objects

Returns

command buffer

keydown(*e*)

detects key being pressed and adds it to input_buffer :param e: event

keyup(*e*)

detects key being released and removes it from input_buffer :param e: event

1.3 RaycastingEngine module

class RaycastingEngine.**RaycastingEngine**(*width*, *height*, *level*, *player*, *textures*, *sprites*)

Bases: object

class responsible for 3D (2.5D actually) view

calculate_sprite_screen_parameters(*sprite*)

calculating all parameters necessary to render sprite on screen :param sprite: :return: screen x coordinate, width, height, brightness modifier, isVisible, distance to player

cast_rays()

method performing a ray casting algorithm :return: array of textured stripes

check_ray_collision(*ray_angle*, *player_x*, *player_y*, *level*)

casting a single ray :param ray_angle: angle of the ray relative to the world :param player_x: player x coordinate :param player_y: player y coordinate :param level: level matrix - zeros represents empty space, any number higher than 0 is a texture id :return: ray length, wall hit-point x coordinate, wall hit-point y coordinate, index of texture

render(*canvas*)

rendering objects in the furthest to the nearest order :param canvas: canvas :return: canvas: updated canvas

1.4 SpriteRender module

class SpriteRender.**SpriteRender**(*params*)

Bases: object

class encapsulating all params to render sprite

render(*canvas*)

1.5 commands package

1.5.1 Submodules

1.5.2 commands.ChangeWeaponCommand module

class commands.ChangeWeaponCommand.**ChangeWeaponCommand**(*player:* Player, *index: int*)

Bases: *Command*

execute()

changes weapon to a given index :return:

undo()

undoes the command :return:

1.5.3 commands.Command module

class commands.Command.**Command**

Bases: ABC

abstract class for all commands. Implemented according to Command design pattern.

abstract execute()

executes the command :return:

abstract undo()

undoes the command :return:

1.5.4 commands.MoveActorCommand module

class commands.MoveActorCommand.**MoveActorCommand**(*actor:* Actor, *dx: float*, *dy: float*)

Bases: *Command*

execute()

moves actor to new position :return:

undo()

undoes the command :return:

1.5.5 commands.RotateActorCommand module

class commands.RotateActorCommand.**RotateActorCommand**(*actor:* Actor, *d_angle: float*)

Bases: *Command*

execute()

rotates actor to new angle :return:

undo()

undoes the command :return:

1.5.6 commands.ZoomCommand module

class commands.ZoomCommand.**ZoomCommand**(*actor:* Actor, *da: float*)

Bases: *Command*

execute()

zooms the view by modifying actor's vertical and horizontal field of view :return:

undo()

undoes the command :return:

1.5.7 Module contents

1.6 main module

1.7 objects package

1.7.1 Submodules

1.7.2 objects.Actor module

class objects.Actor.**Actor**(*x*, *y*, *angle*, *speed*, *rotation_speed*, *fov*, *vertical_angle*, *vision_distance*, *radius*)

Bases: *GameObject*

move_to(*x*, *y*)

render(*canvas*)

rotate_to(*angle*)

updates angle of an actor in [0,2PI] range

update()

1.7.3 objects.DirectionalSprite module

class objects.DirectionalSprite.**DirectionalSprite**(*x*, *y*, *radius*, *render_radius*, *angle*, *folder_path*)

Bases: *GameObject*

load_images(*folder_path*)

loading sprite images :param folder_path: path to the folder containing images of sprite in 8 directions :return:

render(*canvas*, *x*, *y*, *width*, *height*, *brightness*)

rendering sprite in current direction

update(*player_x*, *player_y*, *player_angle*)

updating sprite current direction based on angle between player facing direction and sprite facing direction :param player_x: player x coordinate :param player_y: player y coordinate :param player_angle: :return:

1.7.4 objects.GameObject module

class objects.GameObject.**GameObject**

Bases: ABC

abstract class for every object (entity) in a game. Implemented according to Update Method design pattern

abstract render(*canvas*)

abstract update()

1.7.5 objects.Level module

class objects.Level.**Level**(*level_map*, *screen_height*, *screen_width*)

Bases: object

render(*canvas*)

renders level in 2D view :param canvas: :return: updated canvas

update(*player_x*, *player_y*, *player_angle*)

updates all elements of the level that player can interact with. Right now only opens doors :param player_x: :param player_y: :param player_angle: :return:

1.7.6 objects.Player module

class objects.Player.**Player**(*x*, *y*, *angle*, *speed*, *rotation_speed*, *fov*, *vertical_angle*, *vision_distance*, *radius*, *weapons*)

Bases: *Actor*

change_weapon(*index*)

changes weapon :param index: weapon number :return:

render(*canvas*, *map_tile_size*)

renders selected weapon :param canvas: :param map_tile_size: :return: updated canvas

1.7.7 objects.Sprite2D module

class objects.Sprite2D.**Sprite2D**(*x*, *y*, *radius*, *render_radius*, *path*)

Bases: *GameObject*

render(*canvas*, *x*, *y*, *width*, *height*, *brightness*)

rendering sprite :param canvas: :param x: :param y: :param width: :param height: :param brightness: :return:

update(*player_x*, *player_y*, *player_angle*)

1.7.8 objects.Texture module

class objects.Texture.**Texture**(*texture_path*)

Bases: object

create_reversed()

reverses the texture in x dimension

load()

loading texture from file to the array of pixels in rgb format

render(*canvas*)

rendering texture in 2D view (only for testing) :param canvas: :return: updated canvas

1.7.9 objects.TextureStripe module

class objects.TextureStripe.**TextureStripe**(*segments*)

Bases: object

render(*canvas*)

creates rectangles on canvas representing pixels of texture :param canvas: :return: updated canvas

1.7.10 objects.Weapon module

class objects.Weapon.**Weapon**(*damage*, *speed*, *auto*, *image_path*)

Bases: object

render(*canvas*)

rendering weapon on screen :param canvas: :return:

update()

1.7.11 Module contents

1.8 settings module

setting all necessary parameters and key bindings

1.9 utils module

utils.**hex_to_rgb**(*value*)

returns tuple of rgb values :param value: hex string :return: rgb tuple

utils.**return_rotated_actor_position**(*actor_x*, *actor_y*, *angle_to_rotate*, *max_x*, *max_y*)

rotates actor around the center of the map :param actor_x: :param actor_y: :param angle_to_rotate: :param max_x: :param max_y: :return: x_rotated, y_rotated

utils.**return_rotated_matrix**(*matrix*)

funtion that returns rotated matrix by 90 degrees :param matrix: :return:

utils.**rgb_to_hex**(*rgb*)

returns color value string in hexadecimal format :param rgb: rgb tuple :return:

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX