Zombie Survival Game

https://github.com/judacribz/zombie_shooter.git

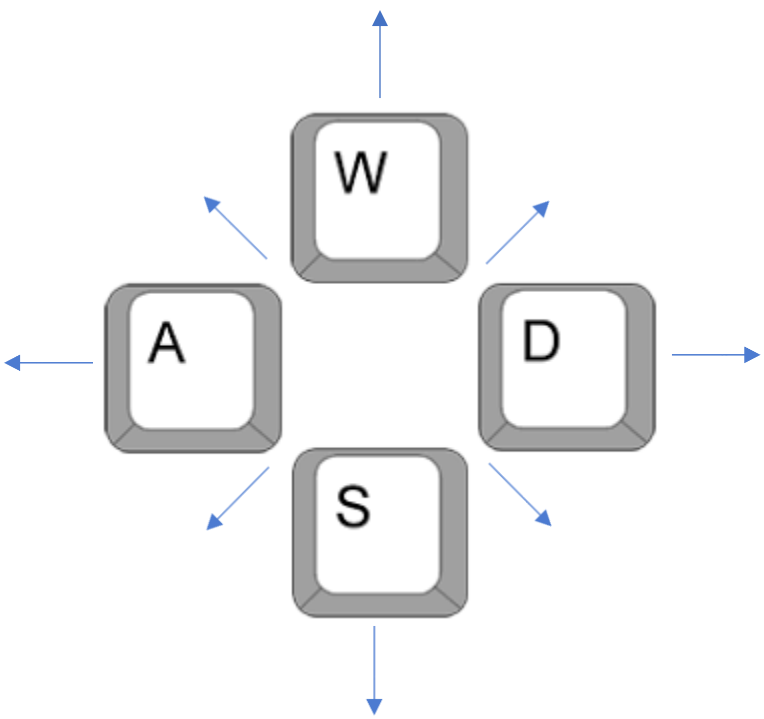
CSCI 3010U Final Project

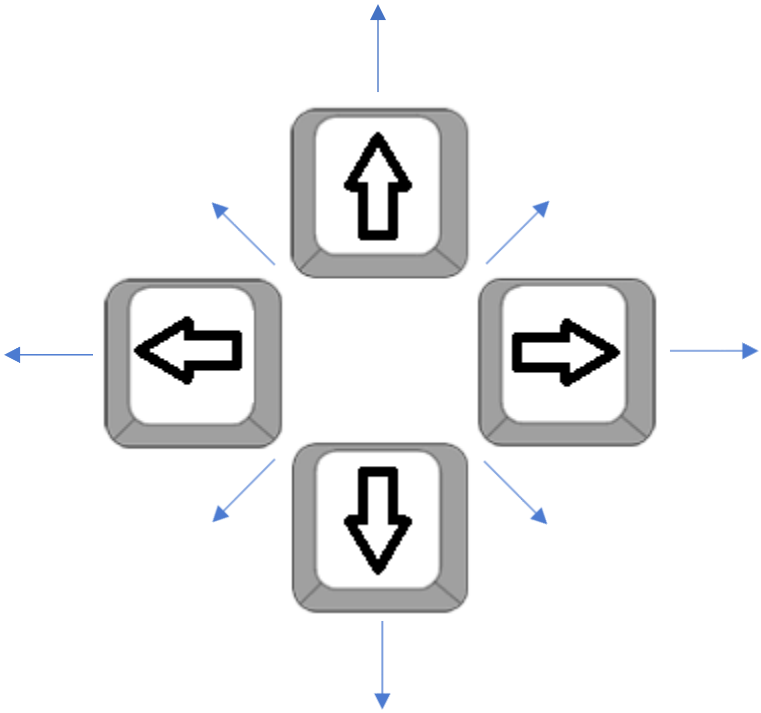
Sheron Balasingam

100504990

How to Play

Player Movement:





Fire Weapon:

Change Weapon:

Switch to Pistol

Switch to Machine Gun

Switch to Shotgun

File Tree

Root Directory:

In the root directory, there is a media directory and sprites directory. main.py starts the game, creates the sprites, and continuously updates them. util.py contains helper functions that are used by all the python files.

zombie_shooter[]

| ---- media[]

| ---- sprites[]

| ---- main.py

| ---- util.py

Media Directory:

The media directory contains all media such as background music, sound effects, and background images.

media[]

| ---- bg_image[]

| ---- bg_music[]

| ---- sound_fx[]

Sprites Directory:

The sprites directory has a directory for bullets, zombies and the player. Within each of these directories, there's a sprite_imgs directory that holds the sprite images for the specific sprite. The sprite class within each of these directories is stored in a python file such as bullet_sprite.py

sprites[]

| ---- bullet[]

| ---- sprite_imgs[] → contains the sprite image for each bullet type used

| ---- pistol.png

| ---- machine.png

| ---- shotgun.png

| ---- bullet_sprite.py

| ---- zombie[]

| ---- sprite_imgs[]

| ---- attack[] → contains the sprite images for when a zombie attacks the player

| ---- dead[] → contains a single sprite image for when a zombie is killed

| ---- motion[] → contains the sprite images for when the zombie moves around

| ---- zombie_sprite.py

| ---- player[]

| ---- player_imgs[]

| ---- dead[] → contains a single sprite image for when the player dies

| ---- motion[] → contains the sprite images for when the player moves around

| ---- player_sprite.py

In Code

main.py:

This file sets up the game and continues to update the screen and all its sprites. Initially 5 zombies are created and positioned randomly near the window edges. The player is positioned in the center.

Once all the zombies are killed, the player position is reset, and 6 zombies are now added. Each time all the zombies are killed, the same number of zombies (plus 1) are recreated. This would represent the start of each level. The previous steps are repeated until 5 levels are complete.

If the player is killed, the sprite image will be changed to a dead player and the zombies will continue to be random

In the main function, use input is recorded. Depending on the user input for moving the player (wasd or arrow keys), one of the following direction vectors(normalized first) would be given to the player sprite to use with its acceleration:

Arrow Keys	up	up + right	right	down + right	down	down + left	left	up + left
WASD Keys	w	w + d	d	s + d	s	s + a	a	w + a
Directions	[0, -1]	[1, -1]	[1, 0]	[1, 1]	[0, 1]	[-1, 1]	[-1, 0]	[-1, -1]

If none of the above keys are pressed, the direction vector to set for the player acceleration will be [0, 0] making the acceleration 0.

There are boundary conditions set for the player and zombie sprites where if they move past the window boundary, they come out the opposite side.

util.py

This file contains all the common helper functions used by the other python files. It also contains global constants such as directory locations for sprite images or distinct types of media.

Functions	Description	Math
rotate(sprite, pos)	Rotates a sprite image based on the new position given by pos.	The angle is first determined using: $\tan\Theta = x/y$ where x is the difference between the old position and the new one in the x direction and y is the same as for x but in the y direction.

		We use this angle to rotate the sprite image.
animate(sprite)	Every time this function is called, it updates the sprite image into the next sprite in the sprites image list.	We use sprite_img[i % num_imgs] to get the next image for the sprite animation. (i is incremented each time this function is called)
check_boundary(sprite, pos)	This function is called with a sprites new position. If the sprite goes out of the window limits, it appears on the opposite side of the window. (This function is not called with the bullet sprites)	
out_of_bounds(pos)	This function returns a boolean value based on if the pos is outside of the window bounds. (This is used only by the bullet sprite)	
length(v)	Returns the length of the vector v	Returns sqrt(v[0]^2 + v[1]^2)
normalize(v)	Returns the unit direction vector for vector v	Returns [v[0]/length(v), v[1]/length(v)]

player_sprite.py:

Motion:

To determine the new positions and velocities, the ode rk4 solver was used. The player also has its velocity magnitude set at 3.0 and acceleration magnitude at 1.0. I added a drag constant of 0.09 so that the player does not keep accelerating and reaches terminal velocity. Whenever the player uses one of the direction keys, the direction vector of acceleration is set to that normalized direction. When no direction is pressed, the acceleration will be [0, 0] and a moving player will be slowed down by the drag.

Animation:

The player has a directory full of sprite images which will animate the player running. The images are all loaded into a list at setup. The animate and rotate functions are used from util.py at each update.

Sounds:

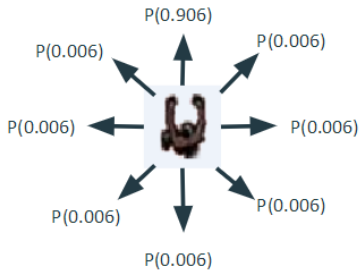
The player sprite has 3 sounds associated with it. The flesh rip sound is played whenever the

players health is decreased by a zombie in proximity. A step sound (in sand) is played every time the player moves. When the player is killed, a specific sound is also played then.

zombie_sprite.py:

Motion:

To determine the new positions and velocities, the ode rk4 solver was used. When the distance between a zombie and player is greater than 200 units, it moves randomly. I had to make sure the zombies did not change direction too often. For this case, I set the zombie's probability to continue in the direction its facing, to be 0.95 + 0.05/8. To change to any of the other 7 directions, the zombie has a 0.05/8 probability.



When the zombie is within 200 units of the player, the zombie image will get rotated towards the player continuously and the zombie will accelerate towards them.

Animation:

The zombie has a directory full of sprite images which will animate the player running. The images are all loaded into a list at setup. The animate and rotate functions are used from util.py at each update.

Sounds:

The zombie sprite has 1 sound associated with it which is used to make the sound for a bullet hitting flesh.

Collisions:

When a bullet hits the zombie, the zombies velocity will change depending on the velocity and mass of the bullet at impact. The following formula was used:

$$v_f = (v_{i(zombie)} * (mass_{(zombie)}) + v_{i(bullet)} * (mass_{(bullet)})) / (mass_{(bullet)} + mass_{(zombie)})$$

(derived from m1v1i + m2v2i = m1v1f +m2v2f where v1f and v2f should be the same since the bullet will be stuck in the zombie after collision)

bullet_sprite.py:

When a bullet is fired, it is initially placed at the center of the player sprite and is set at a velocity with the direction the player is facing. No drag or acceleration is set because since it is a game, we want the possibility of the bullet slowing down to be negligible. The bullets are also slowed down for the user to see, but the velocity is multiplied by a factor of 10 to use in zombie-bullet collisions. The bullet configuration based on the weapon is shown below:

```
#              Weapon Type  Bullet Speed, Bullet Mass, Bullet Dimen, Damage
GUN_CONFIG = {"pistol"  : [15,           1,           10,           10], \
              "machine" : [10,           1,           15,           1], \
              "shotgun" : [15,           10.0,        15,           50]}
```