Game Report: Loser

Contents

1. Introduction

Game “**Loser”** is a side-scrolling side tank shooter 2D game built using Pygame. It involves the player driving a tank with a rotation cannon, fighting air and ground targets constantly advancing on it. The game has two distinct levels, both progressing by difficulty level with the addition of new enemy groups. The game has used both visual and sound special effects to attain the highest level of immersion factor and to provide an engaging experience.

2. Storyboard Narrative

It's the future where the world has been devastated by the mechanical wars, and you'll emerge out of the ashes to battle the enemy air fleet and troop armies. Playing as the tank commander, you'll be operating inside enemy territories, with the game levels ending with boss battles to challenge your reflexes along with your strategic deployment of firepower along with shields. Your mission: survive, earn coins, level up your tank, and take back your territory.

3. Main Characters and Abilities:

- **Player Tank:** Equipped with a rotating cannon, shield ability, and upgradeable weapon systems (bomb gun, rapid fire).
- **Enemies:** Helicopters, jets, tanks, a massive boss, and a medic helicopter (bonus objective).

4. Environment Settings:

- **Level 1:** A rocky wasteland with basic air enemies.
- **Level 2:** A dense warzone with tanks, jets, and a powerful boss unit.

5. Plot and Mechanics:

Players must aim and shoot enemies while dodging attacks and managing their shield meter. Coins collected allow upgrades. Special enemies like the medic helicopter offer bonus rewards if not attacked.

6. Game Assets

Visual Assets:

- Backgrounds: menu_bg.png, level1_bg.png, level2_bg.png

- Sprites: player_tank.png, player_cannon.png, enemy sprites (helicopter, jet, tank, boss), medic_helicopter.png
- Pickups: coin.png, shield_pickup.png
- Effects: explosion.png, bullets (various types)

Audio Assets:

- Background Music: menu_music.mp3, level1_music.mp3, level2_music.mp3
- Sound Effects: player_fire.wav, enemy_explosion.wav, pickup.wav, game_over.wav

Other Assets:

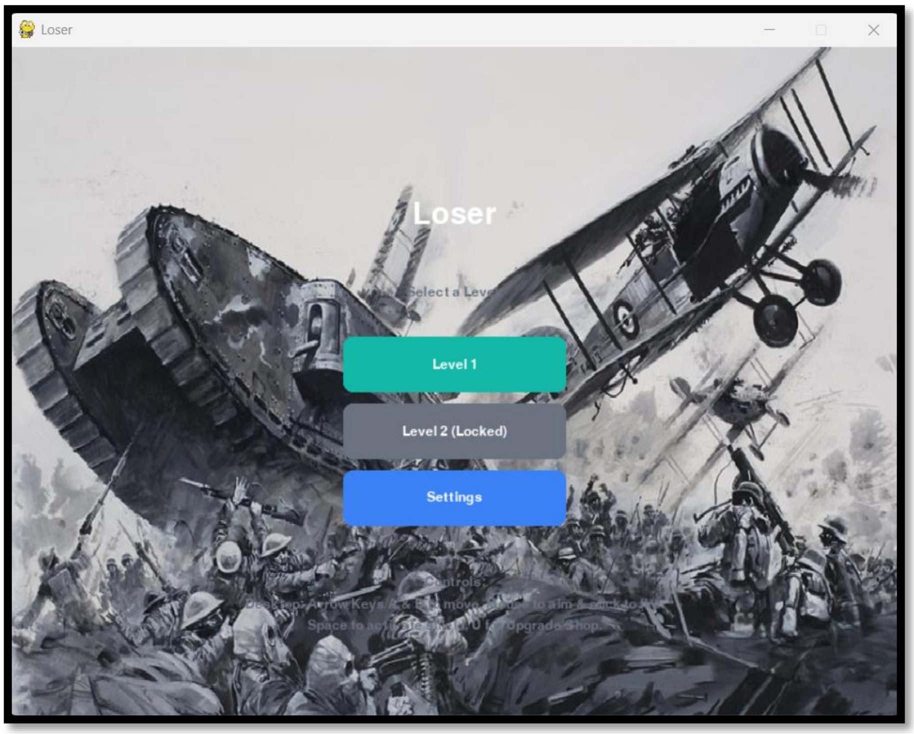
- Special effect indicators for lasers and shield.
- HUD elements for health, shield, coins, andprogress bar.

7. Visual and Audio Special Effects

Visual effects include laser warnings, beam attacks, explosions (using sprite fading), and dynamic shield overlays. Audio feedback such as gunfire, explosions, and pickups enhance user experience. Player interactions trigger audio cues, and game events such as boss attacks are synchronized with effects.

8. Game Scene Screenshots

Main menu is where all the levels and settings are accessible. This is the first look into the game as it is shown when the game is turned on. When the game is turned on for the first time will have level 1 will be accessible while level 2 will be locked.



Controls:
Desktop: Arrow Keys/A & D to move, [illegible] to aim & click to [illegible]
Space to activate [illegible], U for Upgrade Shop.

Figure 8.1 ***Main Menu***

Level 1 have multiple enemy types, simple and slow enemies which drop coins when destroyed and shield refills. Theres also Medic helicopter which appears from time to time which is to be avoided as hitting the medic is a war crime, which will cause you to lose points and coins.

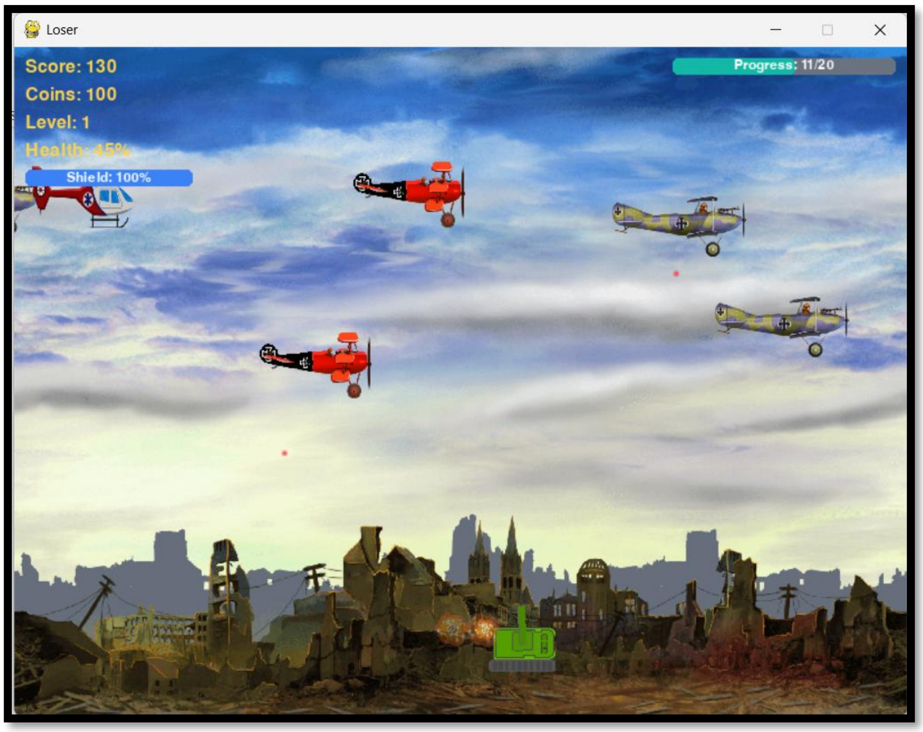


Figure 8.2 ***Level 1 Gameplay***

Level 1 boss is the first boss level enemy of the game, boss enemy appears after the first objective of the level is met. Boss enemy have multiple attack patterns and have a lot of health. Boss enemy must be defeated before the level is completed.

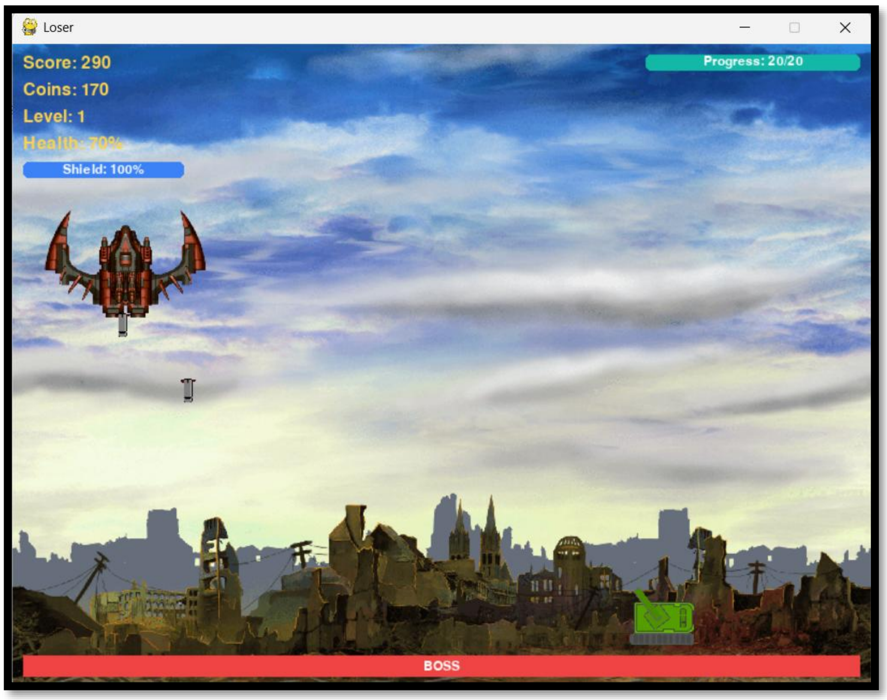
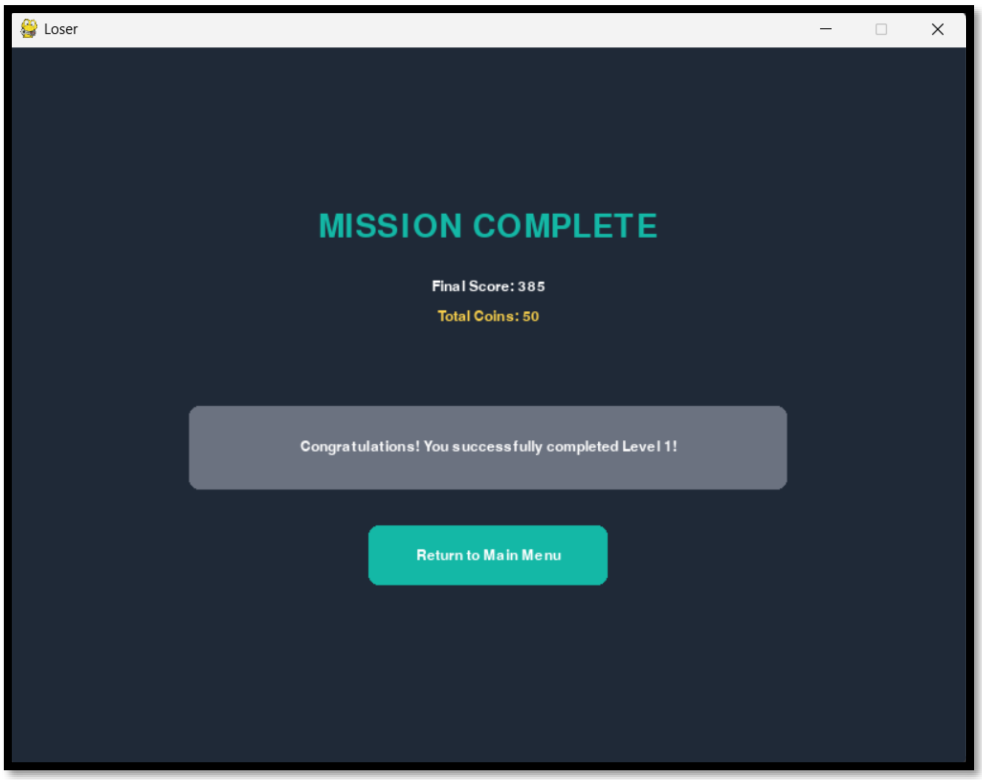


Figure 8.3 ***Level 1 Boss***

After each level is completed, the mission completion prompt appears which takes the player to main menu. This completion prompt also shows the score and coins earned in the level.



Final Score: 385
Total Coins: 50

Congratulations! You successfully completed Level 1!

Figure 8.4 ***level 1 Completion Prompt***

After the completion of level 1, when the player retunes to main menu the level 2 which was previously locked can now be accessed. This restricts the player to jump to level 2 before having understanding of the game. Level 2 have more and difficult enemies. It also have a harder bosses and new enemy types.

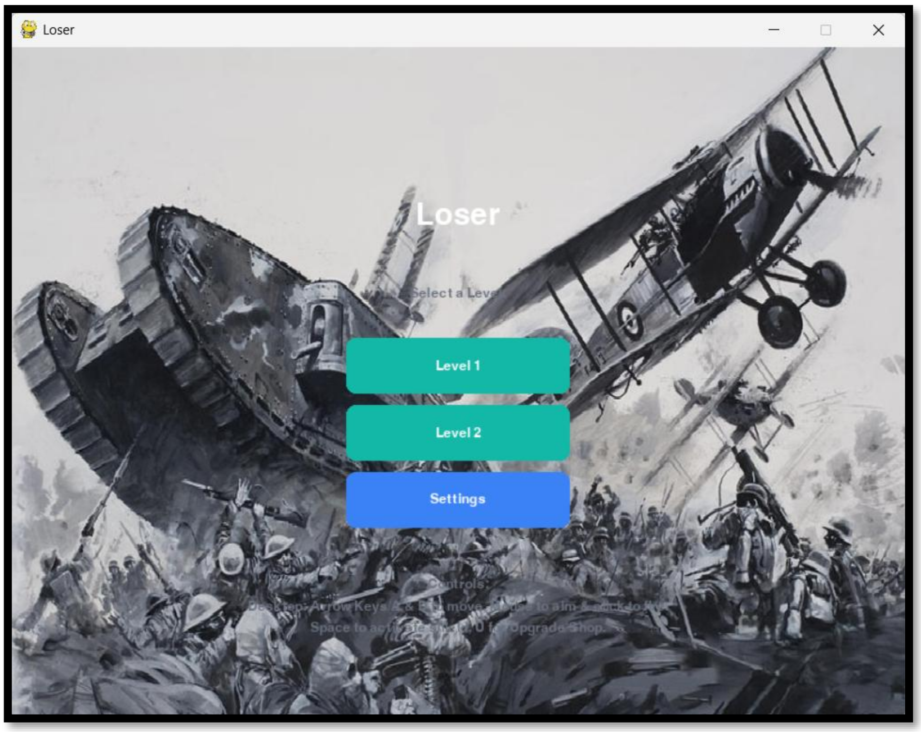


Figure 8.5 ***Main Menu after Completion of Level 1***

In game music and main menu music can be turned off from the main menu settings, this allows player to focus on the game more if the music is distracting them.

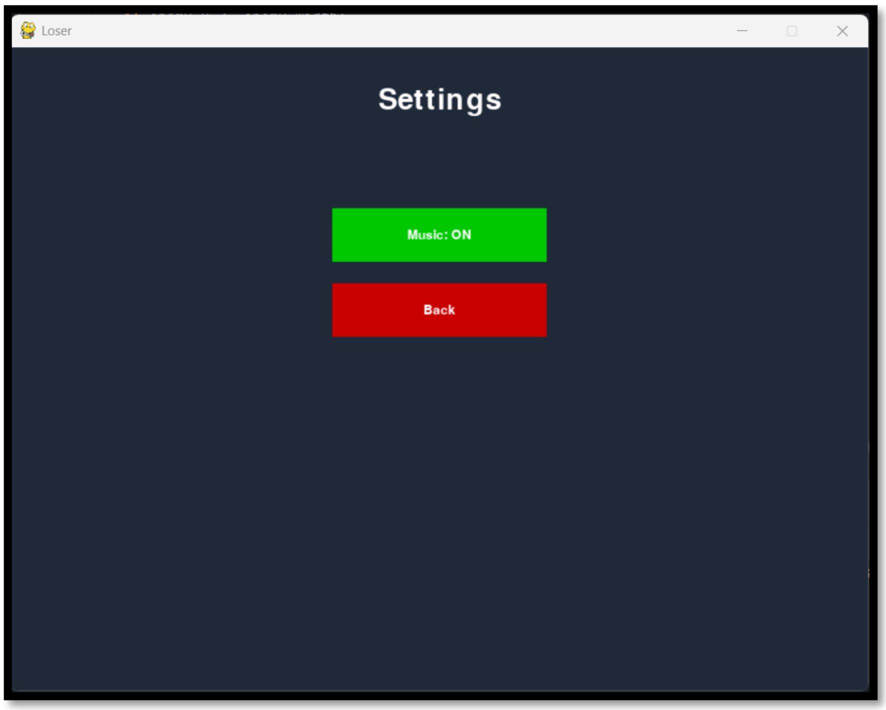


Figure 8.6 ***Settings***

While in game player can access upgrade shop by pressing “U” which allows player to spend their in game coins to upgrade their player character abilities such as increasing the fire rate, increasing the shield up time and turning their bullets to missiles.

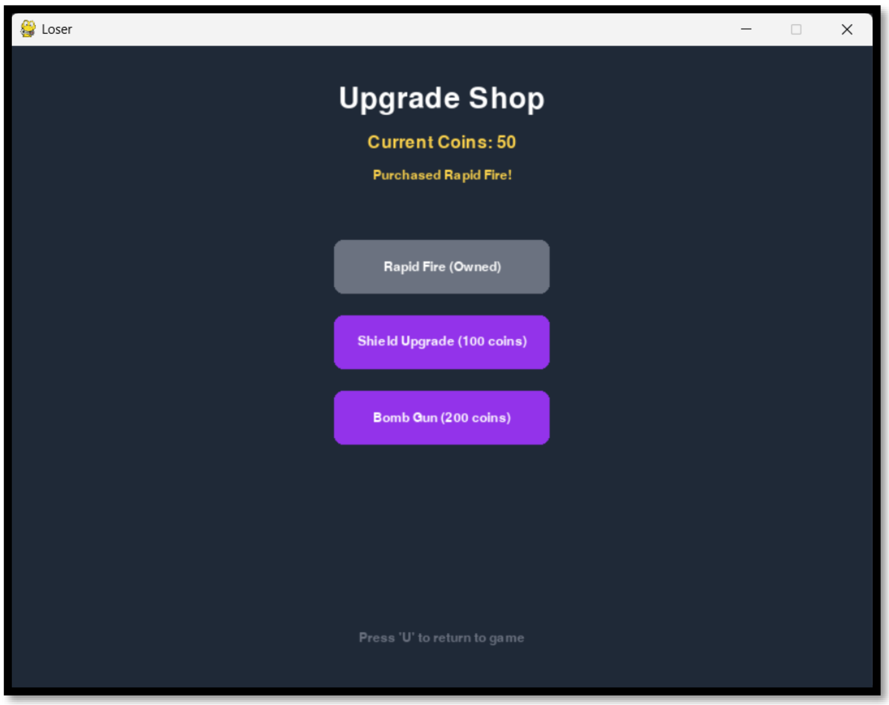


Figure 8.7 ***Upgrade Shop inside the game***

9. Explanation of Concepts Used

Game Loser utilizes many imaging and special effect methods to gain the extent of game playing immersion, graphic acuity, and sound feedback. The graphic assets of the game, the backgrounds, sprites, etc., arrive imported with raster image rendering by pygame.image.load() before being blitted onto the game screen with blit() (Shinners, 2000)(McGugan, 2007). The menu and level specific backgrounds get scaled on the fly to the screen size by pygame.transform.scale() to gain adaptable resolution. The manipulation of the backgrounds allows for layer based composing for the game look where the game items sit on top of the non-moving scenery.

Sprite rendering for the character, player tank, cannon, enemy units (helicopter, jet, tank, boss), and special units (e.g., medic helicopter) is obtained by animating the sprites with transformation techniques including scaling, flipping (wherever applicable, for instance, in directional rendering), and rotation (most noticeably for the player's cannon, spinning on the fly towards the mouse cursor's location). Transparency with sprites is handled by .convert_alpha() for satisfactory blending with the background.

Handling of item picking up for example coin and shield boost is handled by standard icon based sprites, accomplished at runtime, picked up on contact with the game character. Event centric rendering ensures these items do not show up unless necessary and vanish after contact. Explosions and other combat feedbacks employ fading animation sprites. The explosion effect, for example, vanishes after some time by leveraging the set_alpha() function to simulate the fading on the eyes. The lasers and the laser alertness are also implemented by semi opaque surfaces (pygame.SRCALPHA) and runtime drawn lines (Adobe, n.d.). The above implementations make use of color blending along with transparency to show on the screen the presence of hazard and build pressure as the character fights bosses. Sound assets are played in their entirety for player feedback. The level and menu background music is preloaded and controlled using pygame.mixer.music, with contextually triggered playing to switch music appropriately with game state changes. Music playing is cycled to continuously loop, and turning it off/on can be found in the game options menu. Discrete sound effects like player fire, enemy projectiles that blow up, collected items, game over audio, etc., are also controlled using pygame.mixer.Sound (Shinners, 2000). The discrete sound effects are triggered by contextually appropriate game events and played back on multiple channels to provide overlapping audio, enabling responsive, layered gameplay (FreeSound, n.d.).

Other than these, the active shield overlay is generated by rendering translucent circular panels on the fly. The look of the shield changes opacity with the remaining shield meter, thus enabling the player with an instinctual and engaging HUD experience (McGugan, 2007). The HUD aspects of the level bars, the coin numbers, the bars for the health, and the meter for the shields are also rendered by pygame.draw.rect() and get modified with each frame, thus offering the player with live reports on the game character performance along with status.

All these special effects imaging modalities to on-the-fly transformation, behavioral hint layering, to sound syncing add to the image-dense, reactive, and exciting playing world for the Loser. Careful deployment of transparency, scaling, blending, and sound design follows the underlying imaging conventions and game design, achieving animation of the game mechanisms with mood.

10. Conclusion

The game **Loser** effectively demonstrates imaging and special effects through gameplay and visuals. Strengths include smooth player controls, layered enemy design, and meaningful upgrades.

Areas of improvement could include a wider variety of levels and enemy AI behaviors. Visuals as the direction of the enemies are not well defined in the code which makes it difficult to visually match the direction of the enemies. And the lack of visuals pleasing animations.

Future enhancements involve additional levels, multiplayer support, new upgrade paths, permanent upgrades which will be accessible only from main menu, more in-game upgrades, more enemies, more bosses with different combat patterns, better visuals and animations, more settings available in the main menu. Boss specific background music and much more.

11. References

Shinners, P. (2000). *Pygame documentation*. Pygame.org.
https://www.pygame.org/docs/

McGugan, W. (2007). *Beginning Game Development with Python and Pygame: From Novice to Professional*. Apress.
https://link.springer.com/book/10.1007/978-1-4302-0994-1

Unity Technologies. (n.d.). *Introduction to particle systems*. Unity Learn.
https://learn.unity.com/tutorial/introduction-to-particle-systems

FreeSound. (n.d.). *Freesound: Collaborative database of creative-commons licensed sound for musicians and sound lovers*.
https://freesound.org/

Adobe. (n.d.). *What is alpha compositing?* Adobe Help Center.
https://www.adobe.com/creativecloud/design/discover/alpha-compositing.html