
Lab 7 README File:

How to Use:
****

From the zip/7z package:

Unzip all lab7_austin_jonathan.zip and execute tasks in Spyder
Make sure to pip install the proper libraries used at the top of each file
View the program screenshots and "VIDEO" for the following requirements below:

1. FFT/IFFT Audio Signal Processing – Noise Cancelling Application
2. Heart Rate Analysis – Time Domain Measurements – Biotechnology
3. Game Development – Red Alert

Acknowledgments:
****

WAV File Generator:

https://www.audiocheck.net/audiofrequencysignalgenerator_index.php
https://www.wavtones.com/functiongenerator.php

Heartbeat Sound Bank:

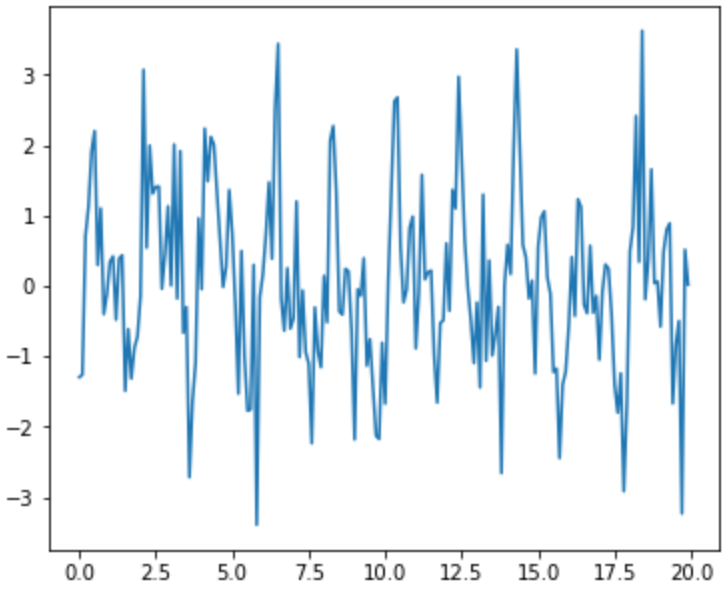
https://www.kaggle.com/kinguistics/heartbeat-sounds

Author:
****

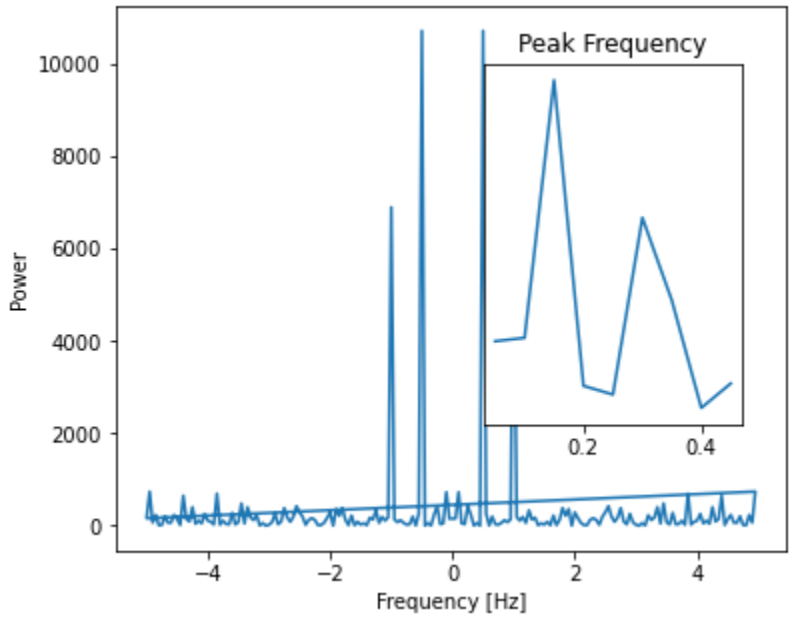
Jonathan Austin <jonathan.austin@sjsu.edu>

1. FFT/IFFT Audio Signal Processing – Noise Cancelling Application

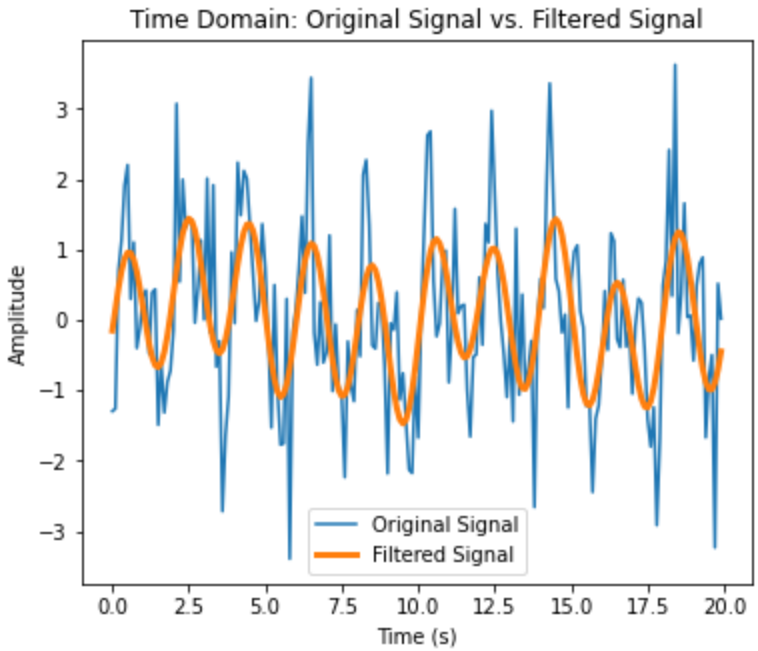
    a. Generated Signal



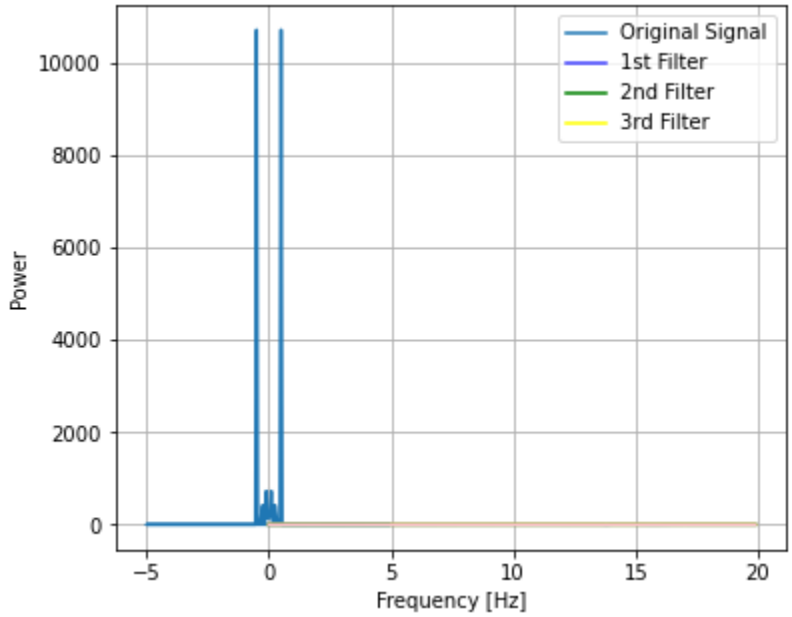
b. (Original) Frequency Domain



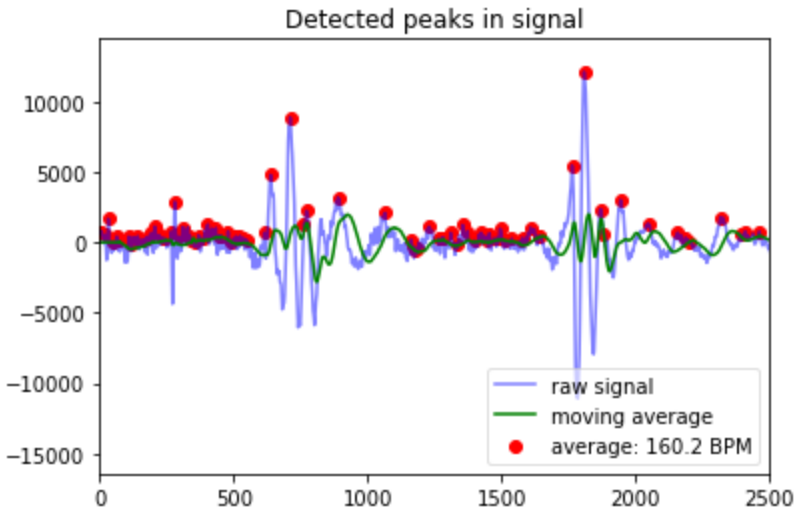
c. (Filtered) Time Domain

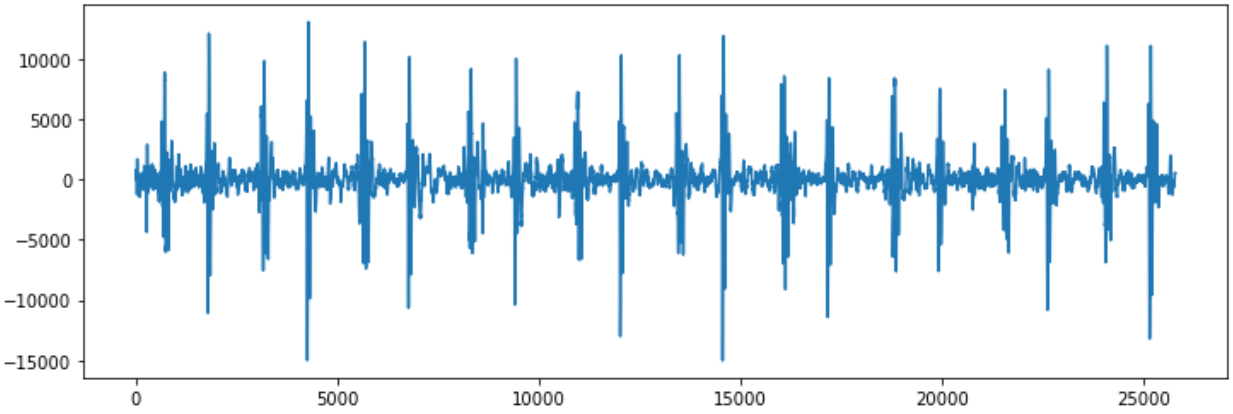


d. (Filtered) Frequency Domain



2. Heart Rate Analysis – Time Domain Measurements – Biotechnology



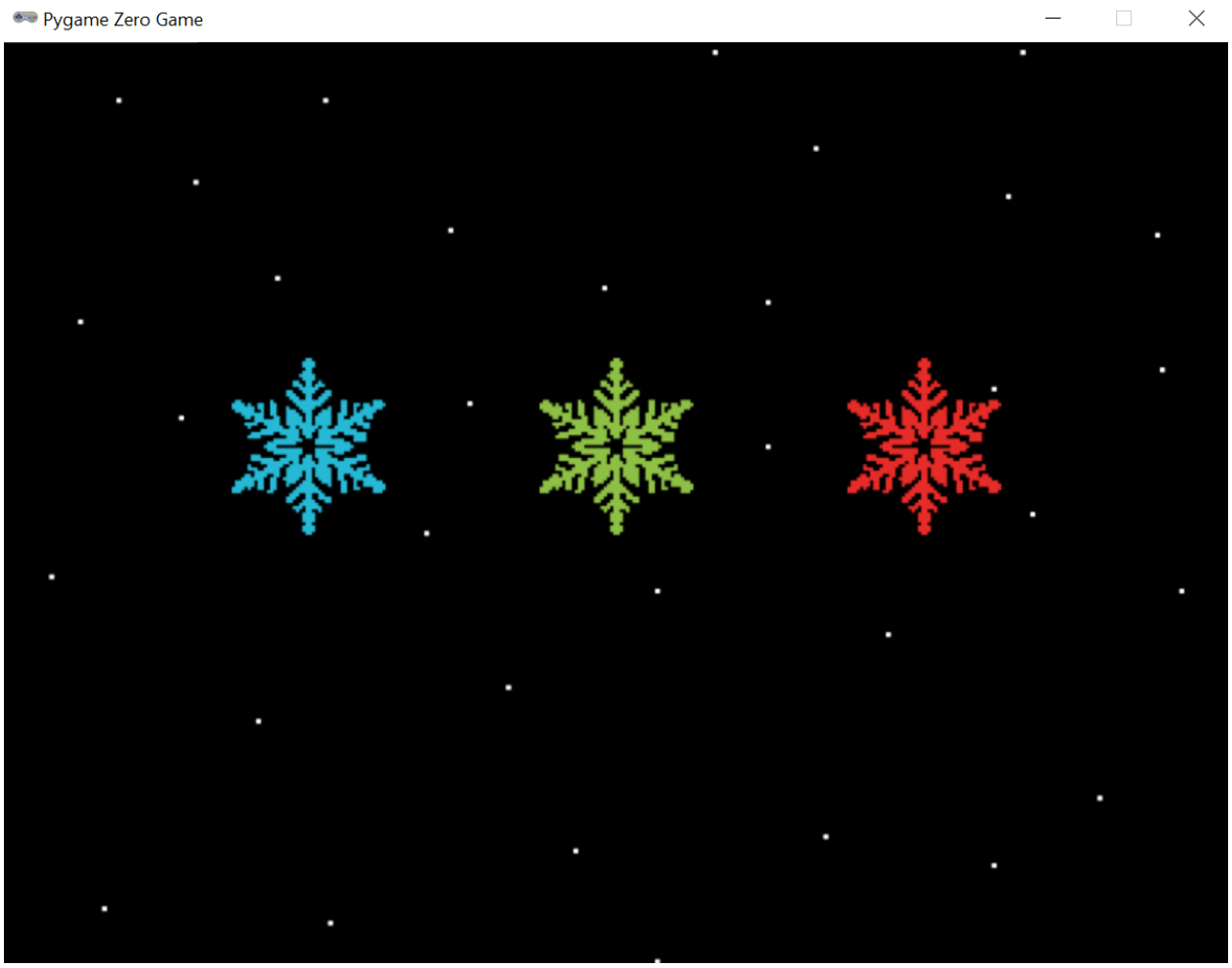


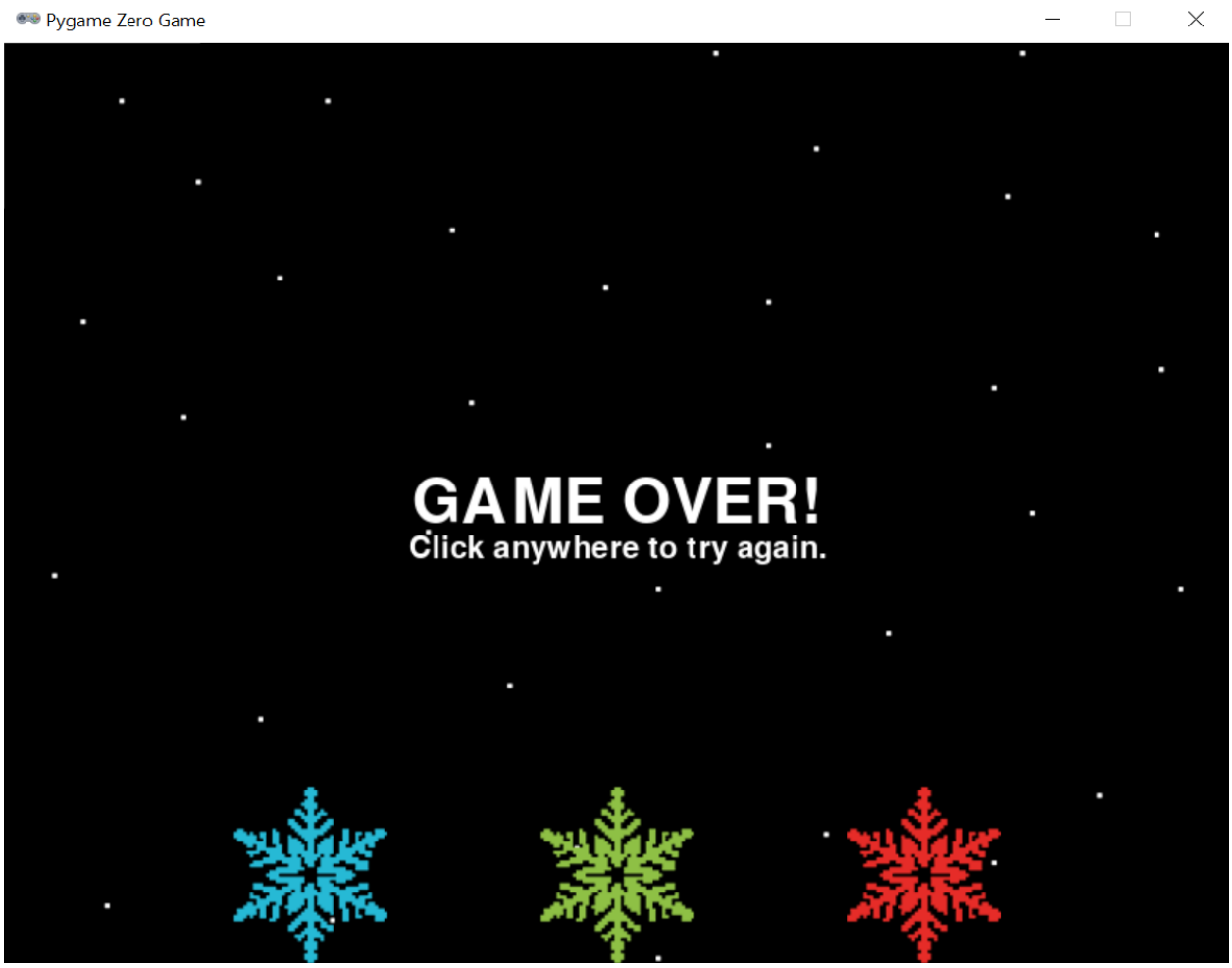
3. Game Development – Red Alert

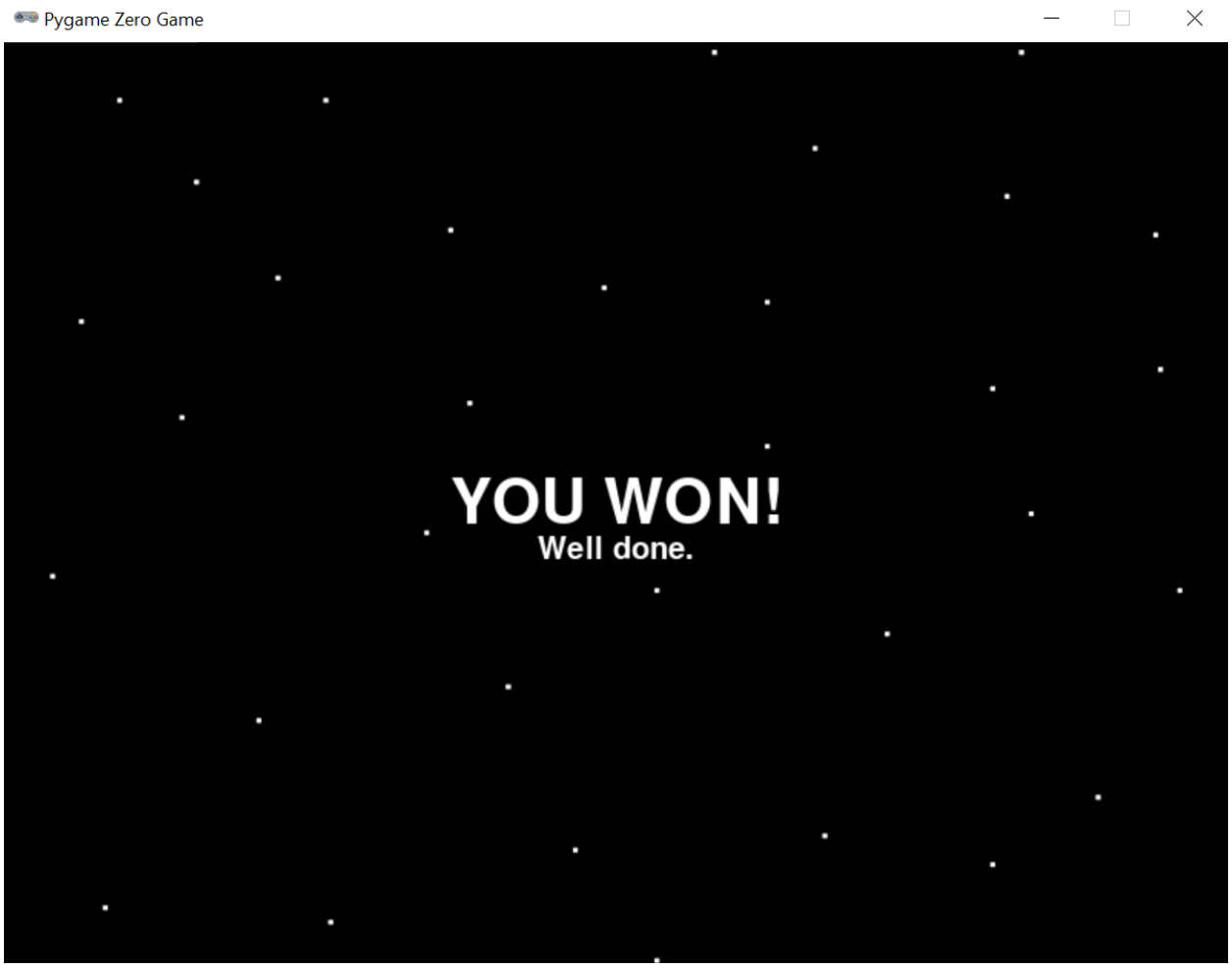
    a. Changed Code

```
#Declare constants
FONT_COLOR = (255, 255, 255)
WIDTH = 800
HEIGHT = 600
CENTER_X = WIDTH / 2
CENTER_Y = HEIGHT / 2
CENTER = (CENTER_X, CENTER_Y)
FINAL_LEVEL = 6
START_SPEED = 10 # "A Need for Speed" : changed speed from 20 -> 10 (need for speed)
COLORS = ["green", "blue"]

#Declare global variables
game_over = False
game_complete = False
current_level = 1

#Keep track of the stars on the screen
stars = []
animations = []

#Draw the stars
def draw():
    global stars, current_level, game_over, game_complete
    screen.clear()
    screen.blit("space", (0,0)) #add a background image to the game window
    if game_over: # "Try Again" : indicate game restart to user on mouse click
        display_message("GAME OVER!", "Click anywhere to try again.")
    if game_complete:
        display_message("YOU WON!", "Well done.")
    else:
        for star in stars:
            star.draw()
```

```
72          star = Actor("snowflake-" + color) # "Change the Actor" : adjusted naming convention for other actor files
73          new_stars.append(star)
74      return new_stars
75
76  def layout_stars(stars_to_layout):
77      #pass
78      number_of_gaps = len(stars_to_layout) + 1
79      gap_size = WIDTH / number_of_gaps
80      random.shuffle(stars_to_layout)
81      for index, star in enumerate(stars_to_layout):
82          new_x_pos = (index + 1) * gap_size
83          star.x = new_x_pos
84
85  def animate_stars(stars_to_animate):
86      #pass
87      for star in stars_to_animate:
88          duration = START_SPEED - current_level
89          star.anchor = ("center", "bottom")
90          animation = animate(star, duration=duration, y=HEIGHT) # "Try Again" : removed duration-caused-end-game
91          animations.append(animation)
92
93  def handle_game_over():
94      global game_over
95      game_over = True
96
97
98  def on_mouse_down(pos):
99      global stars, game_over, current_level, stars, animations
100     if game_over == True: # "Try Again" : if statement to reset the game on next mouse click
101         game_over = False # "Try Again" : reset game_over variable
102         current_level = 1 # "Try Again" : reset current level variable
103         stars = []        # "Try Again" : reset stars list
104         animations = []   # "Try Again" : reset animations list
```


Pygame Zero Game


Pygame Zero Game
GAME OVER!
Click anywhere to try again.


Pygame Zero Game
YOU WON!
Well done.