



Nama: **Muhammad Fadil Ataullah Rifqi (122140205)**

Jason Surya Padantya (122140237)

Tugas Ke: **Tugas Besar**

Mata Kuliah: **Multimedia System and Technology (IF25-40305)**

Tanggal: 12/12/2025

1 Pendahuluan

Pada tugas besar ini, penulis ditugaskan untuk membuat sebuah program Python yang mengimitasi filter-filter populer di *TikTok* atau *Instagram Stories*. Program yang kami kembangkan berjudul **Sapi Go**, yaitu sebuah permainan interaktif di mana pemain dapat mengontrol karakter sapi untuk melewati rintangan dengan menggunakan input suara.

Mekanisme utama permainan ini memanfaatkan *pitch* (tinggi rendah nada) suara pemain. Pemain harus mengatur *pitch* suaranya untuk mengontrol aksi karakter, yaitu apakah sapi akan menunduk atau merampingkan badan guna menghindari rintangan yang muncul.

2 Pembahasan

Untuk mencapai keberhasilan implementasi program *Sapi Go*, kami menggunakan beberapa pustaka (*library*) Python yang terintegrasi untuk menangani grafik, pemrosesan audio, dan antarmuka pengguna.

2.1 Implementasi Visual dengan Pygame

Pygame digunakan sebagai kerangka kerja (*framework*) utama permainan. Pustaka ini berfungsi untuk mengatur jendela permainan (*windowing*), memuat aset gambar (*load image*), membuat animasi sprite, serta menangani pergerakan karakter.

Berikut adalah salah satu implementasi menggunakan *pygame* yang mengatur logika animasi dan status karakter berdasarkan input yang diterima:

```
1 import pygame
2
3 class Sapi(pygame.sprite.Sprite):
4     def __init__(self, normal_path, low_path, high_path, x_pos, target_size=(320, 320)):
5         super().__init__()
6
7         self.target_w, self.target_h = target_size
8         self.frame_count = 6
9         self.animation_speed = 0.2
10
11         # Load ketiga strip animasi untuk setiap kondisi
12         self.animations = {
13             'normal': self.load_strip(normal_path),
14             'nunduk': self.load_strip(low_path),
15             'ramping': self.load_strip(high_path)
16         }
17
18         self.state = 'normal'
19         self.frame_index = 0
20         self.image = self.animations[self.state][0]
```

```

21
22     # Inisialisasi Hitbox (Rect)
23     self.rect = self.image.get_rect()
24     self.rect.x = x_pos
25
26     # Simpan ukuran visual dasar
27     self.base_w = self.target_w
28     self.base_h = self.target_h
29
30     # Offset visual untuk mengatur posisi gambar sapi di atas lantai
31     self.visual_offset_y = 170
32

```

Kode 1: Implementasi Kelas Sapi menggunakan Pygame

2.2 Analisis Audio dengan Librosa

Untuk mendeteksi input suara pemain, kami menggunakan pustaka *Librosa*. Fitur utama yang digunakan adalah fungsi `yin`, yang merupakan algoritma untuk estimasi *fundamental frequency* (F0) atau *pitch*.

Algoritma ini memungkinkan program untuk membedakan nada rendah dan nada tinggi yang dikeluarkan oleh pemain secara *real-time* (dengan pemrosesan *frame*).

```

1  # Deteksi pitch menggunakan metode YIN dari Librosa
2  f0 = librosa.yin(
3      frame,
4      fmin=self.fmin,      # Frekuensi minimum yang dideteksi
5      fmax=self.fmax,      # Frekuensi maksimum yang dideteksi
6      sr=self.samplerate,  # Sample rate audio
7  )
8
9  # Ambil nilai median pitch yang terdeteksi dalam frame tersebut
10 pitch = np.nanmedian(f0)
11

```

Kode 2: Deteksi Pitch menggunakan Algoritma YIN

2.3 Pustaka Pendukung Lainnya

Selain *Pygame* dan *Librosa*, program ini juga didukung oleh beberapa pustaka lain:

- **Sounddevice:** Digunakan untuk menangkap (*capture*) input suara dari mikrofon pengguna secara langsung.
- **OpenCV (cv2):** Digunakan untuk mengakses kamera (*webcam*) yang berfungsi sebagai latar belakang (*background*) permainan, menciptakan efek *Augmented Reality* (AR) sederhana.

3 Penutup

Program *Sapi Go* yang telah kami kembangkan berhasil mengintegrasikan input suara sebagai kontrol utama permainan. Animasi karakter merespons perubahan *pitch* suara sesuai dengan logika yang dirancang.

Namun, berdasarkan hasil pengujian, terdapat kendala teknis berupa sensitivitas terhadap *noise* (gangguan suara latar). Suara bising di sekitar pemain sering kali tertangkap oleh `sounddevice` dan terdeteksi sebagai *pitch* oleh algoritma, yang menyebabkan ketidakakuratan kontrol karakter (karakter bergerak tanpa input yang disengaja).

Saran Pengembangan: Untuk pengembangan selanjutnya, disarankan untuk menambahkan filter pengurangan *noise* (*noise suppression*) sebelum pemrosesan *pitch* atau menerapkan ambang batas (*threshold*) amplitudo agar suara latar yang pelan tidak memicu pergerakan karakter.