

3.3 Realisasi

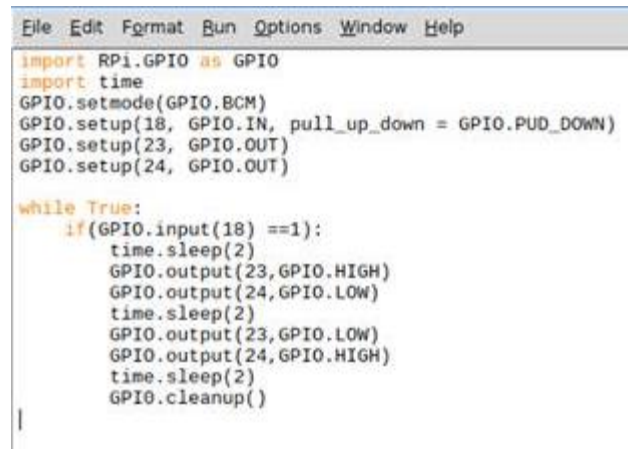
3.3.1 Realisasi Perangkat Keras

3.3.2 Realisasi Perangkat Lunak

3.3.2.1 Realisasi Program

Pada pembahasan sebelumnya telah dibuat diagram alir untuk program sistem secara keseluruhan. *Coding* program yang dihasilkan dari diagram alir dituangkan ke dalam *platform* mikrokontroler. Mulai dari program sensor cahaya pendeteksi tikus, program aktuator motor dc, pengolahan citra menghitung tikus.

Berikut adalah beberapa *listing program* untuk sensor cahaya pendeteksi tikus, program aktuator motor dc, pengolahan citra menghitung tikus.



```
File Edit Format Run Options Window Help
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.IN, pull_up_down = GPIO.PUD_DOWN)
GPIO.setup(23, GPIO.OUT)
GPIO.setup(24, GPIO.OUT)

while True:
    if (GPIO.input(18) == 1):
        time.sleep(2)
        GPIO.output(23, GPIO.HIGH)
        GPIO.output(24, GPIO.LOW)
        time.sleep(2)
        GPIO.output(23, GPIO.LOW)
        GPIO.output(24, GPIO.HIGH)
        time.sleep(2)
        GPIO.cleanup()
```

Gambar 3.1 Program Aktuator Motor DC

Pada Gambar III.1 terdapat perintah “import GPi dan time”. GPI merupakan port dari Raspberry. Terdapat dua jenis port, yaitu IN dan OUT. Port IN digunakan untuk menerima data dari sensor cahaya, sedangkan port OUT untuk memberi daya pada rangkaian aktuator motor DC. Pada listing program, ada dua port output yang akan aktif ketika port in terbaca *high*. Sedangkan “import time” di butuhkan untuk memberi fungsi delay. Agar pergerakan motor DC dapat diatur. Program aktuator DC ini adalah yang mengatur pintu jebakan, agar tikus masuk perangkap dan tidak dapat keluar lagi.

Berikut adalah *listing program* untuk pengolahan citra mendeteksi tikus

```
import cv2
import numpy as np

img_rgb = cv2.imread('Bahrain_sharp.jpeg')
img_gray = cv2.cvtColor(img_rgb, cv2.COLOR_BGR2GRAY)

template = cv2.imread('bahrain_snap.jpg',0)
w, h = template.shape[::-1]

res = cv2.matchTemplate(img_gray,template,cv2.TM_CCOEFF_NORMED)

threshold = 0.8
loc = np.where( res >= threshold)

for pt in zip(*loc[::-1]):
    cv2.rectangle(img_rgb, pt, (pt[0] + w, pt[1] + h), (0,255,255), 2)

cv2.imshow('Detected',img_rgb)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Gambar 3.2 Program Template Matching

Pada Gambar III. Merupakan proses mendeteksi tikus, untuk melakukan proses ini dibutuhkan library opencv dan numpy. Pada awal program merupakan proses *import* gambar berupa foto tikus pada perangkat dan pengubahan gambar menjadi tidak berwarna untuk memudahkan deteksi. Kemudian dilanjut dengan *import* gambar tikus sebagai template. Program template matching adalah program untuk mendeteksi sebuah gambar kecil yaitu berupa template, pada gambar lain yang lebih besar.

```
lspoint=[]
lspoint2=[]
count = 0
for pt in zip(*loc[::-1]):
    if pt[0] not in lspoint and pt[1] not in lspoint2:
        cv2.rectangle(img_rgb, pt, (pt[0] + w, pt[1] + h), (0,255,255), 2)
        for i in range(((pt[0])-9), ((pt[0])+9),1):
            lspoint.append(i)
        for k in range(((pt[1])-9), ((pt[1])+9),1):
            lspoint2.append(k)
        count+=1
    else:
        continue
print "total objects found ", count
```

Gambar 3.3

Pada Gambar 3.3 merupakan proses menghitung tikus yang terdeteksi, yaitu dengan memasukan variable count yang akan di tambahkan jumlahnya ketika proses template matching berhasil mendeteksi.

3.3.1 Realisasi Mekanik