LAMPIRAN

1. Lampiran Program *Background Subtraction*
   a. *Library* yang digunakan

```
import cv2
import time
from tkinter import filedialog
from tkinter import *
from tkinter import messagebox

window = Tk()
window.geometry("7000x3000")
window.title("Motion Detection")
```

   b. Program untuk menampilkan video asli

```
def Real_Video():
    video_source = filedialog.askopenfilename(title="Select file",
                                              filetypes=(("AVI files", "*.avi"),
                                                         ("WMV files", "*.wmv"),
                                                         ("MP4 files", "*.mp4")))
    if len(video_source) > 0:
        cap = cv2.VideoCapture(video_source)
    def rescale_frame(frame, percent=80):
        width = int(frame.shape[1] * percent / 100)
        height = int(frame.shape[0] * percent / 100)
        dim = (width, height)
        return cv2.resize(frame, dim, interpolation=cv2.INTER_AREA)
    if (cap.isOpened() == False):
        messagebox.showwarning("Invalid Video", "Video Error")
    start = time.time()
    i = 0
    while (cap.isOpened()):
        done = time.time()
        i = i + 1
        dif = done - start
        ret, frame = cap.read()
        if ret == True:
            frame = rescale_frame(frame, percent=100)
            cv2.imshow('Vidio.avi', frame)
            if dif == 0:
                continue
            else:
                fps = i / dif
                print(str(fps) + ' fps')
            if cv2.waitKey(25) & 0xFF == ord('q'):
                break
```

c. Program untuk menampilkan video hasil pendeteksian objek bergerak metode *background subtraction*

```
def Background_Subtraction_Method():
    video_source = filedialog.askopenfilename(title="Select file",
                                              filetypes=(("AVI files", "*.avi"),
                                                         ("WMV files", "*.wmv"),
                                                         ("MP4 files", "*.mp4")))
    if len(video_source) > 0:
        cap = cv2.VideoCapture(video_source)
        # fps = FPS().start()
        ret, first_frame = cap.read()
        first_gray = cv2.cvtColor(first_frame, cv2.COLOR_BGR2GRAY)

        def rescale_frame(frame, percent=50):
            width = int(frame.shape[1] * percent / 100)
            height = int(frame.shape[0] * percent / 100)
            dim = (width, height)
            return cv2.resize(frame, dim, interpolation=cv2.INTER_AREA)

        if (cap.isOpened() == False):
            messagebox.showwarning("Invalid Video", "Video Error/tidak dapat dibuka")
        start = time.time()
        i = 0
```

d. Program inti metode *background subtraction*

```
ret, first_frame = cap.read()
first_gray = cv2.cvtColor(first_frame, cv2.COLOR_BGR2GRAY)

def rescale_frame(frame, percent=50):
    width = int(frame.shape[1] * percent / 100)
    height = int(frame.shape[0] * percent / 100)
    dim = (width, height)
    return cv2.resize(frame, dim, interpolation=cv2.INTER_AREA)

if (cap.isOpened() == False):
    messagebox.showwarning("Invalid Video", "Video Error/tidak dapat dibuka")
start = time.time()
i = 0
while (cap.isOpened()):
    #time.sleep(0.003)
    done = time.time()
    i = i + 1
    dif = done - start

    ret, frame = cap.read()
    gray_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    gray_frames = cv2.GaussianBlur(gray_frame, (5, 5), 0)
    difference = cv2.absdiff(first_gray, gray_frames)
    ret, thresh = cv2.threshold(difference, 40, 225, cv2.THRESH_BINARY)
    dilated = cv2.dilate(thresh, None, iterations=1)
    contours, _ = cv2.findContours(dilated, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
    # cv2.drawContours (frame, contours, -1, (0, 255, 0), 2)
```

e. Program untuk menampilkan *rectangle* metode *background subtraction*

```
for contour in contours:
    (x, y, w, h) = cv2.boundingRect(contour)
    if cv2.contourArea(contour) < 700:
        continue
    cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)
    cv2.putText(frame, "Status: {}".format('Movement'), (10, 20),
                cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2)

res_1 = rescale_frame(frame, percent=110)
cv2.imshow('Background Subtraction', res_1)

if dif == 0:
    continue
else:
    fps = i / dif
    print(str(fps) + ' fps')
key = cv2.waitKey(2)
if key == ord('q'):
    break
```

f. Program untuk menampilkan proses operasi pengurangan metode *background subtraction*

```
def Processed_Video():
    video_source = filedialog.askopenfilename(title="Select file",
                                              filetypes=(("AVI files", "*.avi"),
                                                         ("WMV files", "*.wmv"),
                                                         ("MP4 files", "*.mp4")))
    if len(video_source) > 0:
        cap = cv2.VideoCapture(video_source)
        # fps = FPS().start()
        ret, first_frame = cap.read()
        first_gray = cv2.cvtColor(first_frame, cv2.COLOR_BGR2GRAY)

        def rescale_frame(frame, percent=50):
            width = int(frame.shape[1] * percent / 100)
            height = int(frame.shape[0] * percent / 100)
            dim = (width, height)
            return cv2.resize(frame, dim, interpolation=cv2.INTER_AREA)

        if (cap.isOpened() == False):
            messagebox.showwarning("Invalid Video", "Video Error/tidak dapat dibuka")
        start = time.time()
        i = 0
        while (cap.isOpened()):
```

g. Program untuk menampilkan video hasil pengurangan metode *background subtraction*

```
while (cap.isOpened()):
    #time.sleep(0.03)
    done = time.time()
    i = i + 1
    dif = done - start
    ret, frame = cap.read()
    gray_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    gray_frames = cv2.GaussianBlur(gray_frame, (5, 5), 0)
    difference = cv2.absdiff(first_gray, gray_frames)
    ret, thresh = cv2.threshold(difference, 40, 225, cv2.THRESH_BINARY)
    dilated = cv2.dilate(thresh, None, iterations=1)
    contours, _ = cv2.findContours(dilated, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
    # cv2.drawContours (frame, contours, -1, (0, 255, 0), 2)
    for contour in contours:
        (x, y, w, h) = cv2.boundingRect(contour)
        if cv2.contourArea(contour) < 700:
            continue
        cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)
        cv2.putText(frame, "Status: {}".format('Movement'), (10, 20),
                    cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2)
    res_1 = rescale_frame(difference, percent=110)
    cv2.imshow('Processed_Video', res_1)
    if dif == 0:
        continue
    else:
        fps = i / dif
        print(str(fps) + ' fps')
    key = cv2.waitKey(2)
    if key == ord('q'):
        break
```

h. Program GUI metode *background subtraction*

```
label_1 = Label(window, text="Motion Detection Program", fg='black',
                bg='light blue', relief="solid", font=("arial", 24, "bold"))
label_1.pack()

label2 = Label(window, text="Background Subtraction Method", width=25, font=("arial", 20, "bold"))
label2.place(x=40, y=150)

tombol_1 = Button(window, text="Original_Video", width=24, fg='black', bg='light blue', relief=RIDGE,
                  font=("arial", 18, "bold"), command=_Real_Video)
tombol_1.place(x=60, y=210)

tombol_2 = Button(window, text="Background Subtraction_Video", width=24, fg='black', bg='light blue',
                  font=("arial", 18, "bold"), command=_Background_Subtraction_Method)
tombol_2.place(x=60, y=300)

tombol_3 = Button(window, text="Processed_Video", fg='black', width=24, bg='light blue', relief=RIDGE,
                  font=("arial", 18, "bold"), command=_Processed_Video)
tombol_3.place(x=60, y=390)

tombol_3 = Button(window, text="Exit Program", fg='black', width=24, bg='light blue', relief=RIDGE,
                  font=("arial", 18, "bold"), command=_window.destroy)
tombol_3.place(x=60, y=480)

window.mainloop()
```

2. Lampiran Program *Frame Differencing*
   a. Library yang digunakan

```
import cv2
import time
from tkinter import filedialog
from tkinter import *
from tkinter import messagebox

window = Tk()
window.geometry("7000x3000")
window.title("Motion Detection")
```

   b. Program untuk menampilkan video asli

```
def Real_Video():
    video_source = filedialog.askopenfilename(title="Select file",
                                              filetypes=(("AVI files", "*.avi"),
                                                         ("WMV files", "*.wmv"),
                                                         ("MP4 files", "*.mp4")))
    if len(video_source) > 0:
        cap = cv2.VideoCapture(video_source)
    def rescale_frame(frame, percent=80):
        width = int(frame.shape[1] * percent / 100)
        height = int(frame.shape[0] * percent / 100)
        dim = (width, height)
        return cv2.resize(frame, dim, interpolation=cv2.INTER_AREA)
    if (cap.isOpened() == False):
        messagebox.showwarning("Invalid Video", "Video Error")
    start = time.time()
    i = 0
    while (cap.isOpened()):
        done = time.time()
        i = i + 1
        dif = done - start
        ret, frame = cap.read()
        if ret == True:
            frame = rescale_frame(frame, percent=100)
            cv2.imshow('Vidio.avi', frame)
            if dif == 0:
                continue
            else:
                fps = i / dif
                print(str(fps) + ' fps')
            if cv2.waitKey(25) & 0xFF == ord('q'):
                break
```

c. Program untuk menampilkan video hasil pendeteksian objek bergerak metode *frame differencing*

```
def Frame_Differencing_Method():
    video_source = filedialog.askopenfilename(title="Select file",
                                              filetypes=(("AVI files", "*.avi"),
                                                         ("WMV files", "*.wmv"),
                                                          ("MP4 files", "*.mp4")))
    if len(video_source) > 0:
        cap = cv2.VideoCapture(video_source)
        ret, frame1 = cap.read()
        ret, frame2 = cap.read()

        def rescale_frame(frame, percent=50):
            width = int(frame.shape[1] * percent / 100)
            height = int(frame.shape[0] * percent / 100)
            dim = (width, height)
            return cv2.resize(frame, dim, interpolation=cv2.INTER_AREA)

        if (cap.isOpened() == False):
            messagebox.showwarning("Invalid Video", "Video Error/tidak dapat dibuka")
        start = time.time()
        i = 0
```

d. Program inti metode *frame differencing*

```
        while (cap.isOpened()):
            time.sleep(0.003)
            done = time.time()
            i = i + 1
            dif = done - start

            diff = cv2.absdiff(frame1, frame2)
            gray = cv2.cvtColor(diff, cv2.COLOR_BGR2GRAY)
            blur = cv2.GaussianBlur(gray, (5, 5), 0)
            _, thresh = cv2.threshold(blur, 5, 225, cv2.THRESH_BINARY)
            dilated = cv2.dilate(thresh, None, iterations=1)
            contours, _ = cv2.findContours(dilated, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
            # cv2.drawContours(frame1, contours, -1, (0, 255, 0), 2)
```

e. Program untuk menampilkan *rectangle* metode *frame differencing*

```
        for contour in contours:
            (x, y, w, h) = cv2.boundingRect(contour)
            if cv2.contourArea(contour) < 700:
                continue
            cv2.rectangle(frame1, (x, y), (x + w, y + h), (0, 255, 0), 2)
            cv2.putText(frame1, "Status: {}".format('Movement'), (10, 20),
                        cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 3)
        res_1 = rescale_frame(frame1, percent=110)
        cv2.imshow('Frame differencing', res_1)
        frame1 = frame2

        ret, frame2 = cap.read()
        if dif == 0:
            continue
        else:
            fps = i / dif
            print(str(fps) + ' fps')
        key = cv2.waitKey(2)
        if key == ord('q'):
            break
```

f. Program untuk menampilkan proses operasi pengurangan metode *frame differencing*

```
104 def Processed_Video():
105     video_source = filedialog.askopenfilename(title="Select file",
106                                               filetypes=(("AVI files", "*.avi"),
107                                                          ("WMV files", "*.wmv"),
108                                                          ("MP4 files", "*.mp4")))
109     if len(video_source) > 0:
110         cap = cv2.VideoCapture(video_source)
111         cap = cv2.VideoCapture(video_source)
112         ret, frame1 = cap.read()
113         ret, frame2 = cap.read()
114 
115         def rescale_frame(frame, percent=50):
116             width = int(frame.shape[1] * percent / 100)
117             height = int(frame.shape[0] * percent / 100)
118             dim = (width, height)
119             return cv2.resize(frame, dim, interpolation=cv2.INTER_AREA)
120 
121         if (cap.isOpened() == False):
122             messagebox.showwarning("Invalid Video", "Video Error/tidak dapat dibuka")
123         start = time.time()
124         i = 0
```

g. Program untuk menampilkan video hasil pengurangan metode *frame differencing*

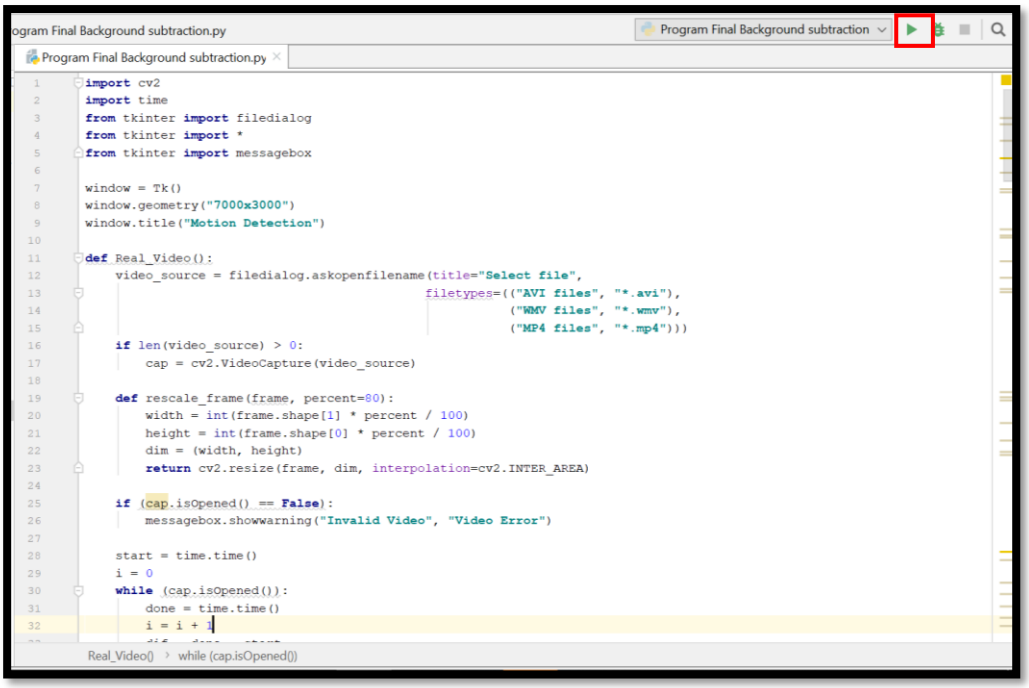
```
        while (cap.isOpened()):
            done = time.time()
            i = i + 1
            dif = done - start

            diff = cv2.absdiff(frame1, frame2)
            gray = cv2.cvtColor(diff, cv2.COLOR_BGR2GRAY)
            blur = cv2.GaussianBlur(gray, (5, 5), 0)
            _, thresh = cv2.threshold(blur, 5, 225, cv2.THRESH_BINARY)
            dilated = cv2.dilate(thresh, None, iterations=1)
            contours, _ = cv2.findContours(dilated, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
            # cv2.drawContours(frame1, contours, -1, (0, 255, 0), 2)
            for contour in contours:
                (x, y, w, h) = cv2.boundingRect(contour)
                if cv2.contourArea(contour) < 700:
                    continue
                cv2.rectangle(frame1, (x, y), (x + w, y + h), (0, 255, 0), 2)
                cv2.putText(frame1, "Status: {}".format('Movement'), (10, 20),
                            cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 3)
            res_1 = rescale_frame(diff, percent=110)
            cv2.imshow('Frame differencing', res_1)
            frame1 = frame2
            ret, frame2 = cap.read()
            if dif == 0:
                continue
            else:
                fps = i / dif
                print(str(fps) + ' fps')
            key = cv2.waitKey(2)
            if key == ord('q'):
                break
```

h. Program GUI metod *frame differencing*

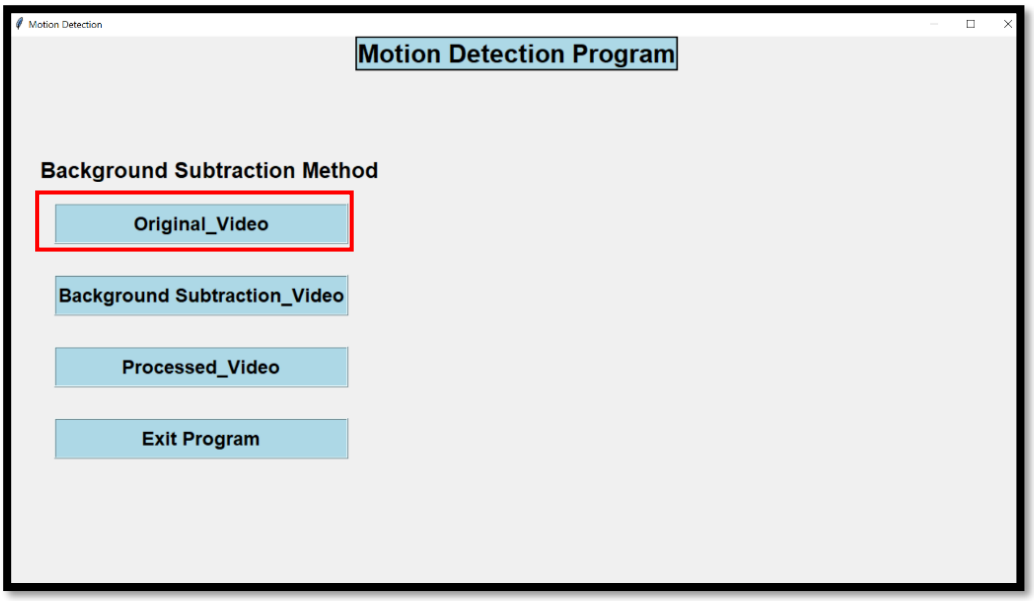
```
label_1 = Label(window, text="Motion Detection Program", fg='white',
                bg='brown', relief="solid", font=("arial", 24, "bold"))
label_1.pack()

label2 = Label(window, text="Frame Differencing Method", width=24, font=("arial", 20, "bold"))
label2.place(x=40, y=150)

tombol_1 = Button(window, text="Original Video", width=24, fg='white', bg='brown', relief=RAISED,
                  font=("arial", 18, "bold"), command= Real_Video)
tombol_1.place(x=60, y=210)

tombol_2 = Button(window, text="Frame Differencing Video", width=24, fg='white', bg='brown',
                  relief=RAISED,font=("arial", 18, "bold"), command= Frame_Differencing_Method)
tombol_2.place(x=60, y=300)

tombol_3 = Button(window, text="Processed Video", fg='white', width=24, bg='brown', relief=RAISED,
                  font=("arial", 18, "bold"), command= Processed_Video)
tombol_3.place(x=60, y=390)

tombol_3 = Button(window, text="Exit Program", fg='white', width=24, bg='brown', relief=RAISED,
                  font=("arial", 18, "bold"), command= window.destroy)
tombol_3.place(x=60, y=480)

window.mainloop()
```

3. Lampiran cara mengoperasikan GUI

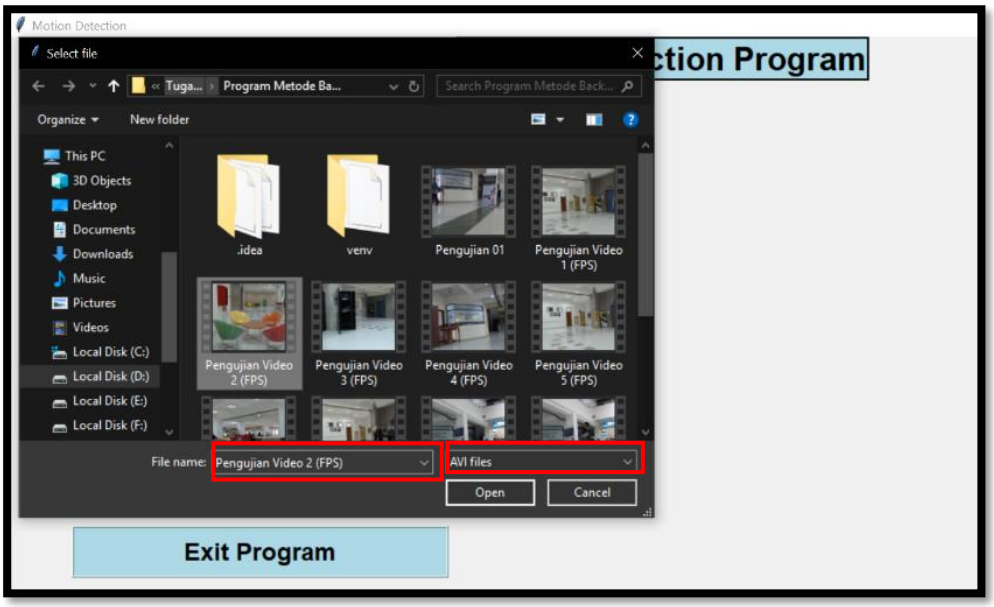
a. *Run* Program *Background subtraction*



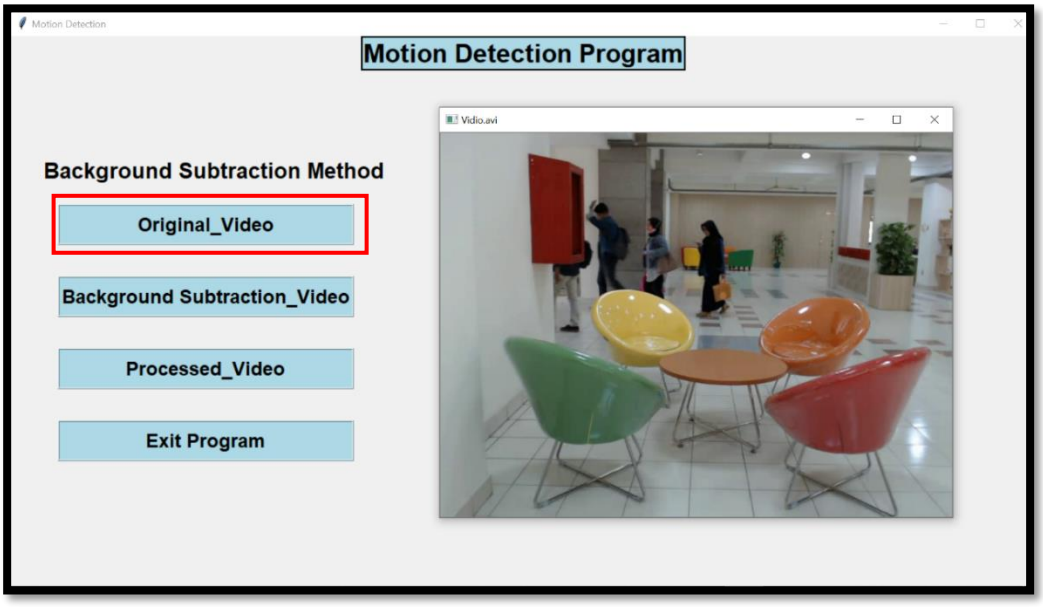
b. Klik tombol Original_video



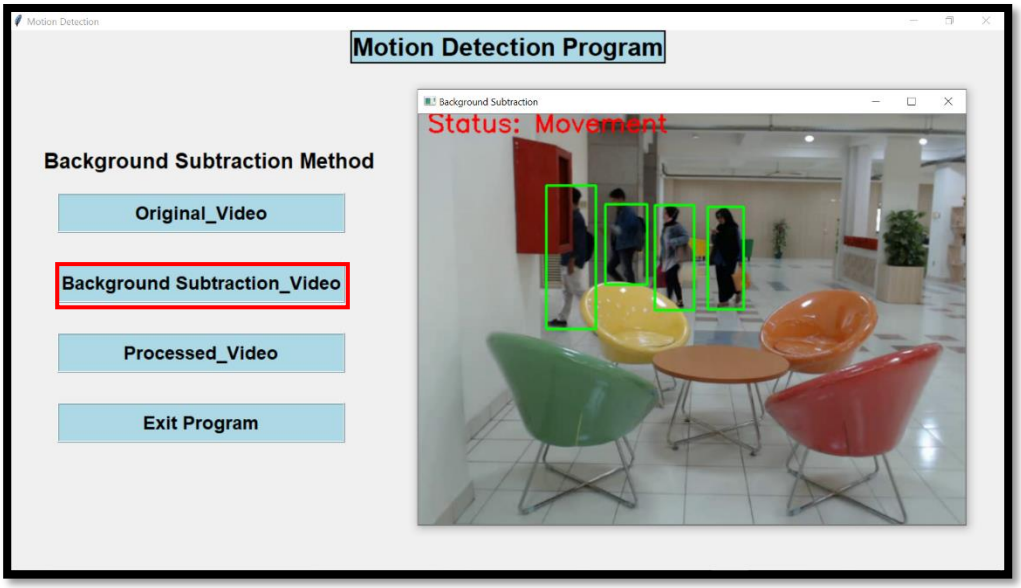
c. Setiap *button* kecuali tombol *Exit Program* yang dipilih akan menampilkan opsi untuk memilih video dan menentukan format video berupa avi, mp4, dan wmv. Tombol *exit program* akan menutup *display* menu GUI.



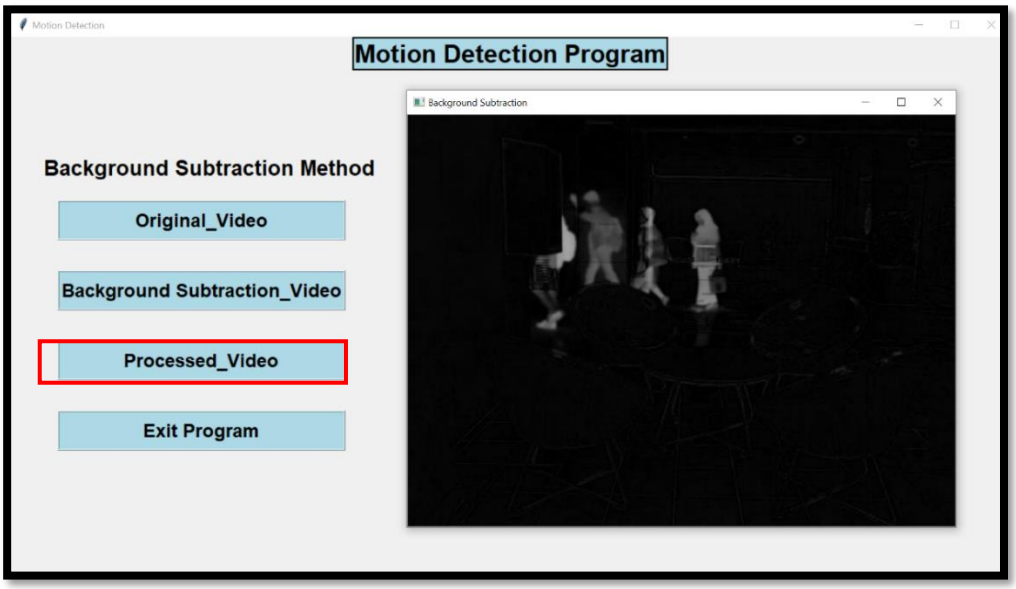
d. Menampilkan Video asli



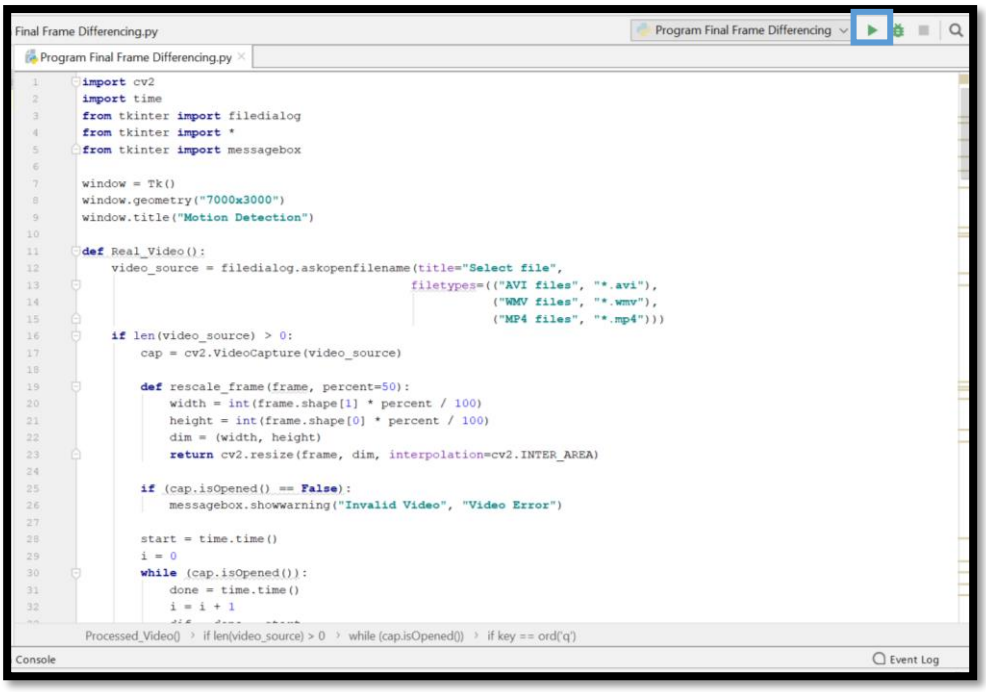
e. Menampilkan citra video hasil pendeteksian objek bergerak metode *background subtraction*



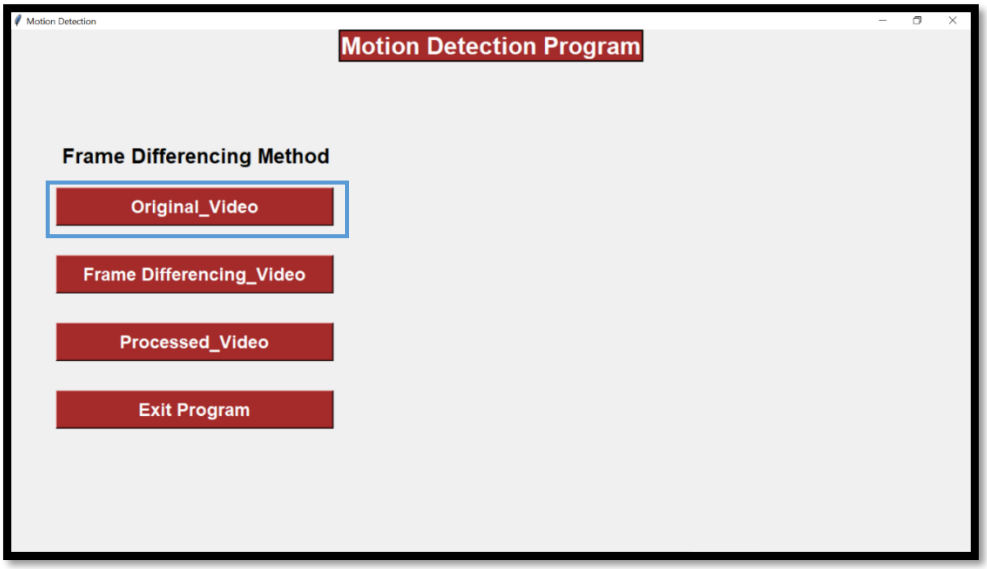
f. Program untuk menampilkan proses operasi pengurangan metode *background subtraction*



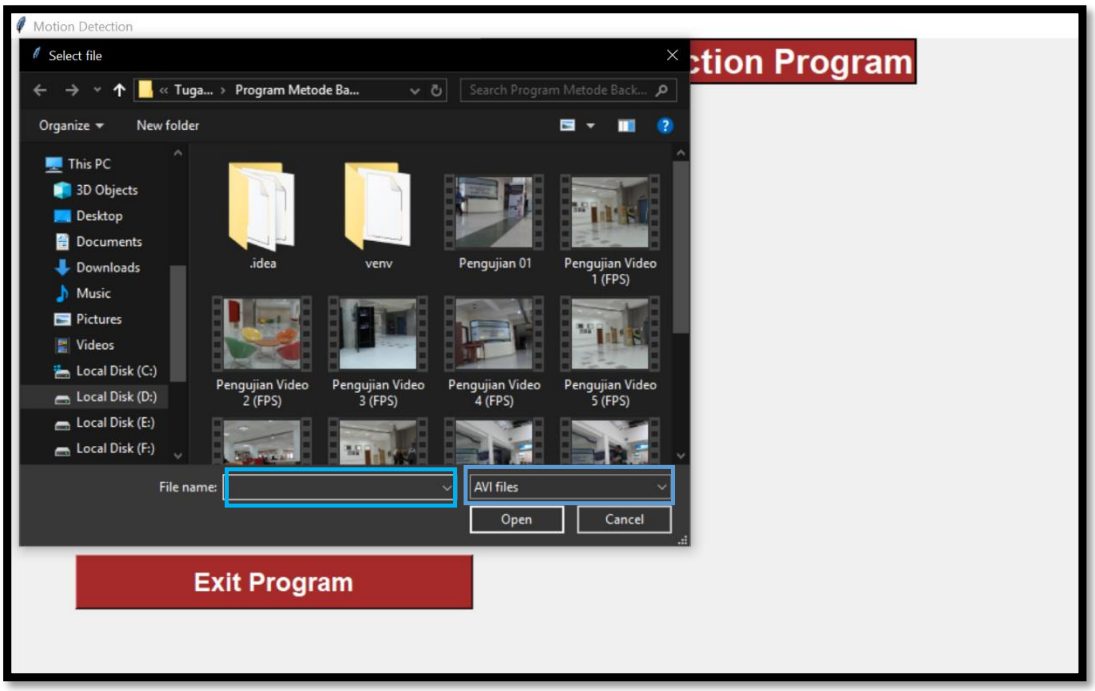
4. Lampiran cara menggunakan GUI metode *frame differencing*
   a. Run Program *frame differencing*



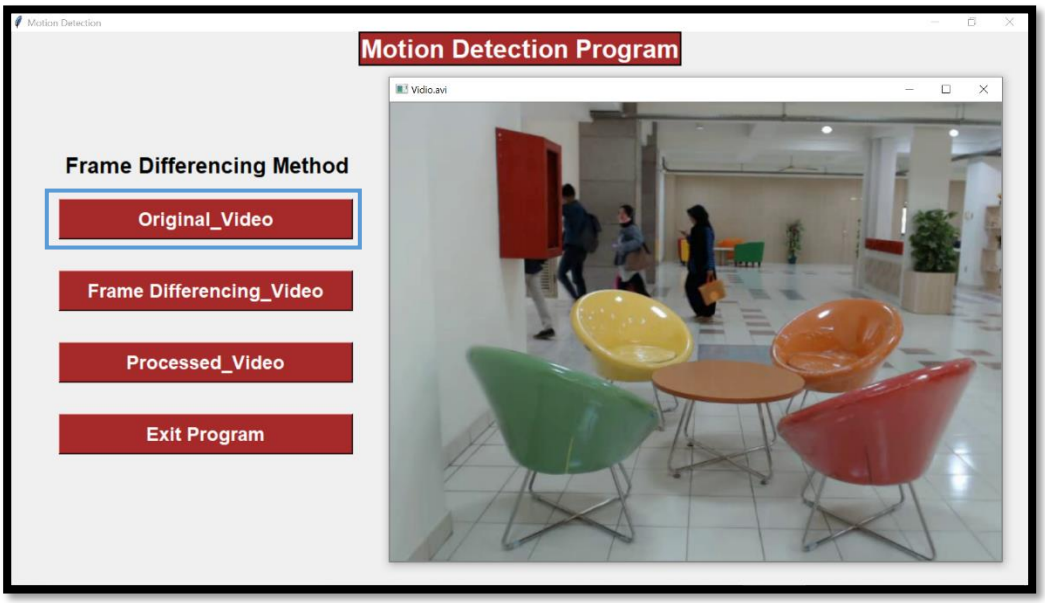
   g. Klik tombol Original_video



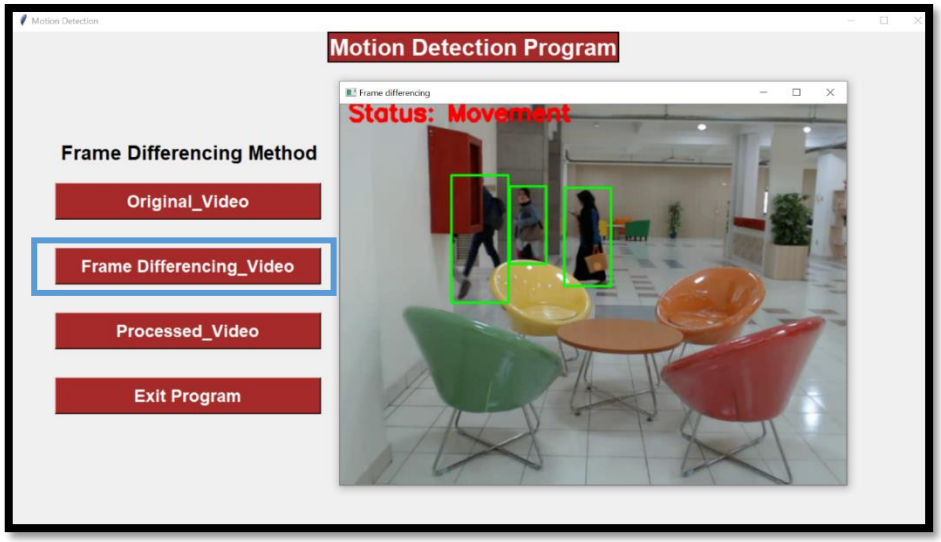
h. Setiap *button* kecuali tombol *Exit Program* yang dipilih akan menampilkan opsi untuk memilih video dan menentukan format video berupa avi, mp4, dan wmv. Tombol *exit program* akan menutup *display* menu GUI.



i. Menampilkan Video asli



j. Menampilkan citra video hasil pendeteksian objek bergerak metode *frame differencing*



k. Program untuk menampilkan proses operasi pengurangan metode *frame differencing*