LAMPIRAN

Lampiran 1. *Listing Code*

- *Form* Utama Aplikasi

```
def configure(self, main_window):
        self.window = main_window
        self.window.keyPressEvent = self.keyboardPress

        self.mode_apps.addItems(["Implementasi", "Pengujian"])
        self.mode = [self.frame_mode, self.frame_change]
        self.mode_apps.currentIndexChanged.connect(self.changeMode)
        for i in range(len(self.mode)):
            self.mode[i].hide()
            self.viewTotal.show()

        self.jml_asli.setText("0")
        self.jml_asli.setAttribute(QtCore.Qt.WA_MacShowFocusRect, 0)
        self.jml_asli.textChanged.connect(self.changeJmlAsli)
        self.last_jml = 0
        self.ok.clicked.connect(self.submitTotalAsli)
        self.video = QThreadVideo()
        self.ok.setCursor(QtCore.Qt.PointingHandCursor)
        self.window = MainWindow
        self.ukuran           =           (self.input_video.width(),
self.input_video.height())
        self.video.setUkuran(self.ukuran)
        self.isPlay = False
        self.submit.clicked.connect(self.uploadVideo)
        self.submit.setCursor(QtCore.Qt.PointingHandCursor)
        self.video.output_frame.connect(self.play)
        self.video.data_real.connect(self.change_dataReal)
        self.video.data_total.connect(self.change_dataTotal)
        self.video.akurasi.connect(self.akurasiSistem)

        self.css = [["cond"], ["new", "last"]]
        self.changeFrame = [self.btn_bs, self.btn_ccl, self.btn_jml]
        self.changeHr = [self.hr_dilasi, self.hr_ccl, self.hr_jml]
        self.changeRLine    =    [self.line_dilasi,    self.line_ccl,
self.line_jml]

        self.combine    =    [self.changeFrame,    self.changeRLine,
self.changeHr]
        for i in range(len(self.combine)):
            self.changeFrame[i].mousePressEvent = lambda _, pos=i:
self.modeStep(
                pos)

self.changeFrame[i].setCursor(QtCore.Qt.PointingHandCursor)

            css = self.css[1][1]
            if(i == 2):
                css = self.css[1][0]
            for j in range(len(self.changeHr)):
                self.combine[i][j].setProperty(self.css[0][0], css)
        self.last_mode = 0

    def modeStep(self, pos):
        if(self.isPlay == False and len(self.video.simpan_data) > 0):
```

```
            if(pos < 2):
                self.akurasi.setText(
                    "<html><head/><body><p><span    style=\"    font-
weight:600;\">Akurasi : -</span></p></body></html>")
                num = 0
                if(pos == 1):
                    num = self.video.b

self.dataReal.setText("<strong>{}</strong>".format(num))

self.dataTotal.setText("<strong>{}</strong>".format(num))
                self.play(self.video.simpan_data[0][pos])
            else:
                if(self.last_jml <= 0):
                    QtWidgets.QMessageBox.question(
                        self.window,   'Inputan   Error!',   "Inputan
Jumlah   Harus   lebih   dari   0",   QtWidgets.QMessageBox.Yes   |
QtWidgets.QMessageBox.No)
                else:
                    self.dataReal.setText(
                        "<strong>{}</strong>".format(self.video.b))
                    self.dataTotal.setText(
                        "<strong>{}</strong>".format(self.video.b))
                    self.video.setJmlAsli(self.last_jml)
        else:
            msg = "Harap Input Citra Video Kemudian Proses Citra Video"
            if(len(self.video.simpan_data) > 0):
                msg = "Video Masih Di Proses"
            QtWidgets.QMessageBox.question(
                self.window,       'Porses       Error!',       msg,
QtWidgets.QMessageBox.Yes | QtWidgets.QMessageBox.No)

    def keyboardPress(self, e):
        return

    def akurasiSistem(self, akurasi):
        self.akurasi.setText(
            "<html><head/><body><p><span        style=\"        font-
weight:600;\">Akurasi                                          :
{}%</span></p></body></html>".format(akurasi))

    def change_dataTotal(self, total):
        pos = self.mode_apps.currentIndex()
        if(pos == 0):
            self.viewTotal.setText(
                "<strong>Jumlah           Lele           Terdeteksi:
{}</strong>".format(total))
        else:
            self.dataTotal.setText(str(total))

    def change_dataReal(self, total):
        pos = self.mode_apps.currentIndex()
        if(pos == 1):
            self.dataReal.setText(str(total))
        else:
            self.dataReal.setText("0")

    def play(self, data):
        panjang = data.shape[0]
        if panjang > 0:
            qt_img = self.convert_cv_qt(data)
```

```
            self.input_video.setPixmap(qt_img)
        else:
            self.isPlay = False
            for i in range(len(self.combine)):
                css = self.css[1][1]
                if(i == 2):
                    css = self.css[1][0]
                for j in range(len(self.changeHr)):
                    self.combine[i][j].setProperty(self.css[0][0],
css)
        #     self.input_video.clear()

    def convert_cv_qt(self, cv_img):
        # convert from an opencv imager to QPixmap
        rgb_image = cv2.cvtColor(cv_img, cv2.COLOR_BGR2RGB)
        h, w, ch = rgb_image.shape
        bytes_per_line = ch * w
        convert_to_Qt_format = QtGui.QImage(
            rgb_image.data,        w,        h,        bytes_per_line,
QtGui.QImage.Format_RGB888)
        p = convert_to_Qt_format.scaled(
            self.ukuran[0],                              self.ukuran[1],
QtCore.Qt.KeepAspectRatio)
        return QtGui.QPixmap.fromImage(p)

    def uploadVideo(self, event):
        if(self.isPlay == False):
            options = QtWidgets.QFileDialog.Options()
            options |= QtWidgets.QFileDialog.DontUseNativeDialog
            fileName,                                                  _
                                                                       =
QtWidgets.QFileDialog.getOpenFileName(self.window,   "Upload   Video",
os.getcwd(
            ),    "Video    Files    (*.avi    *.mp4    *.flv    *.mov)",
options=options)
            if (len(fileName) > 0):
                self.akurasi.setText(
                    "<html><head/><body><p><span    style=\"    font-
weight:600;\">Akurasi : -</span></p></body></html>")
                self.isPlay = True
                self.video.setVideo(fileName)
                self.video.start()

        pos = self.mode_apps.currentIndex()
        if(pos == 0):
            self.viewTotal.setText("<strong>Jumlah  Lele  Terdeteksi:
</strong>")
        else:
            self.dataReal.setText("<strong>{}</strong>".format(0))
            self.dataTotal.setText("<strong>{}</strong>".format(0))


    def submitTotalAsli(self, e):
        if(self.last_jml <= 0):
            QtWidgets.QMessageBox.question(
                self.window, 'Inputan Error!', "Inputan Jumlah Harus
lebih dari 0", QtWidgets.QMessageBox.Yes | QtWidgets.QMessageBox.No)
        else:
            self.video.setJmlAsli(self.last_jml)

    def changeJmlAsli(self, e):
        if(len(e) == 0):
```

```
            self.last_jml = 0
        else:
            try:
                jml = int(e)
                self.last_jml = jml
                if(jml < 0):
                    self.last_jml = 0
            except:
                self.jml_asli.setText(str(self.last_jml))

    def changeMode(self, pos):
        if(self.isPlay == False):
            if pos != self.last_mode:
                self.input_video.clear()
            self.last_mode = pos
            for i in range(len(self.mode)):
                if(pos == 0):
                    self.viewTotal.setText(
                        "<strong>Jumlah Lele Terdeteksi: 0</strong>")
                    self.mode[i].hide()
                else:
                    self.viewTotal.setText(
                        "<strong>Jumlah Lele Terdeteksi: </strong>")
                    self.mode[i].show()


self.dataReal.setText("<strong>{}</strong>".format(0))

self.dataTotal.setText("<strong>{}</strong>".format(0))
        else:
            self.mode_apps.setCurrentIndex(self.last_mode)
```

- *Class* Pemrosesan Metode *Background Subtraction* dan *Connected Component Labelling*

```
class QThreadVideo(QtCore.QThread):
    output_frame = QtCore.pyqtSignal(np.ndarray)
    data_real = QtCore.pyqtSignal(int)
    data_total = QtCore.pyqtSignal(int)
    akurasi = QtCore.pyqtSignal(int)
    pindah_menu = QtCore.pyqtSignal(int)

    def __init__(self):
        super(QThreadVideo, self).__init__()
        self.jml_asli = 0
        self.width = 0
        self.height = 0
        self.video = ""
        self.a = 0
        self.b = 0
        self.cond = True
        self.simpan_data = []
        self.counter = 0

    def getAkurasi(self):
        if(self.jml_asli > 0 and self.cond == False):
            self.em = abs(self.a - self.b)
            self.er = (self.em / self.a) * 100
            akurasi_total = 100 - self.er
            self.akurasi.emit(akurasi_total)
```

```
            print(self.er)
        else:
            print("Video Process")

    def setUkuran(self, ukuran):
        self.width = ukuran[0]
        self.height = ukuran[1]

    def setJmlAsli(self, total):
        self.jml_asli = total
        self.a = self.jml_asli
        self.getAkurasi()

    def setVideo(self, video_name):
        self.video = cv2.VideoCapture(video_name)

    def run(self):
        self.cond = True
        akumulasi = []
        if ui.last_mode == 1:
            self.simpan_data = []
            self.counter = 0
        while(self.cond):
            _, frame = self.video.read()
            if(_):
                frame = cv2.resize(frame, (500, 500))
                # frame = cv2.resize(frame, (self.width,self.height))
                rgb_image = cv2.split(frame)
                result_image = []
                result_norm_image = []
                for image in rgb_image:
                    kernel_erosi = np.ones((5, 5), np.uint8)
                    frame_dilasi = cv2.dilate(image, kernel_erosi)
                    frame_median = cv2.medianBlur(frame_dilasi, 21)
                    frame_diff    =    255    -    cv2.absdiff(image,
frame_median)
                    frame_normalisasi = cv2.normalize(
                        frame_diff,    None,    alpha=0,    beta=255,
norm_type=cv2.NORM_MINMAX, dtype=cv2.CV_8UC1)
                    result_image.append(frame_diff)
                    result_norm_image.append(frame_normalisasi)
                result = cv2.merge(result_image)
                result_norm = cv2.merge(result_norm_image)
                frame_final        =        cv2.cvtColor(result_norm,
cv2.COLOR_BGR2GRAY)
                thresh = cv2.threshold(
                    frame_final,    0,    255,    cv2.THRESH_OTSU    +
cv2.THRESH_BINARY_INV)[1]
                output = cv2.connectedComponentsWithStats(
                    thresh, 4, cv2.CV_32S)
                (numLabels, labels, stats, centroids) = output
                mask = np.zeros(thresh.shape, dtype="uint8")
                last = 0
                for i in range(1, numLabels):
                    x = stats[i, cv2.CC_STAT_LEFT]
                    y = stats[i, cv2.CC_STAT_TOP]
                    w = stats[i, cv2.CC_STAT_WIDTH]
                    h = stats[i, cv2.CC_STAT_HEIGHT]
                    area = stats[i, cv2.CC_STAT_AREA]

                    if(area >= 450 and w >= 15):
```

```
                        last += 1
                        componentMask = (labels == i).astype("uint8")
* 255
                        mask = cv2.bitwise_or(mask, componentMask)
                        cv2.rectangle(frame, (x, y), (x + w, y + h),
                                      (0, 255, 0), 3)
                        cv2.putText(frame, str(
                            last), (x, y), cv2.FONT_HERSHEY_SIMPLEX,
0.5, (0, 0, 255), 2, cv2.LINE_AA)
                frame = cv2.resize(frame, (self.width, self.height))
                self.data_real.emit(last)
                akumulasi.append(last)
                cv2.waitKey(1)
                if(self.counter == 0 and ui.last_mode == 1):
                    self.counter = 1
                    thresh    =    cv2.resize(thresh,    (self.width,
self.height))
                    self.simpan_data.append([thresh, frame])
            else:
                from collections import Counter
                mode = Counter(akumulasi)
                mode = mode.most_common(1)[0][0]
                mean = statistics.mean(akumulasi)
                median = statistics.median(akumulasi)
                self.cond = False
                frame = np.array([])
                self.data_total.emit(mode)
                self.b = mode
            self.output_frame.emit(frame)
```

Lampiran 2. Biodata Penulis

DATA PRIBADI

Nama	:	Nisa’ Nurrochmah Hani
NIM	:	1741720164
Tempat, Tanggal Lahir	:	Tulungagung, 11 Juli 1998
Jenis Kelamin	:	Perempuan
Agama	:	Islam
Jurusan / Program Studi	:	Teknologi Informasi / DIV -Teknik Informatika
Alamat	:	Jalan Pahlawan I, RT 7 RW 1, Desa Rejoagung, Kecamatan Kedungwaru, Kabupaten Tulungagung, Jawa Timur
E-mail	:	nisanhanii@gmail.com

RIWAYAT PENDIDIKAN

2017 - 2021	:	Politeknik Negeri Malang
2014 - 2017	:	SMA Negeri 1 Boyolangu
2011 - 2014	:	SMP Negeri 1 Tulungagung
2005 - 2011	:	SD Negeri 1 Kampungdalem