1 源代码

1.1 designer/child_setting.py

```python
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'child_setting.ui'
#
# Created by: PyQt5 UI code generator 5.15.10
#
# WARNING: Any manual changes made to this file will be lost when pyuic5 is
# run again.  Do not edit this file unless you know what you are doing.


from PyQt5 import QtCore, QtGui, QtWidgets


class Ui_MainWindow(object):
    def setupUi(self, MainWindow):
        MainWindow.setObjectName("MainWindow")
        MainWindow.resize(354, 366)
        self.centralwidget = QtWidgets.QWidget(MainWindow)
        self.centralwidget.setObjectName("centralwidget")
        self.tabWidget = QtWidgets.QTabWidget(self.centralwidget)
        self.tabWidget.setGeometry(QtCore.QRect(10, 20, 321, 251))
        self.tabWidget.setObjectName("tabWidget")
        self.tab = QtWidgets.QWidget()
        self.tab.setObjectName("tab")
        self.tab1_combobox = QtWidgets.QComboBox(self.tab)
        self.tab1_combobox.setGeometry(QtCore.QRect(70, 0, 171, 22))
        self.tab1_combobox.setObjectName("tab1_combobox")
        self.tab1_label = QtWidgets.QLabel(self.tab)
        self.tab1_label.setGeometry(QtCore.QRect(0, 30, 311, 191))
        self.tab1_label.setObjectName("tab1_label")
        self.tabWidget.addTab(self.tab, "")
        self.tab_2 = QtWidgets.QWidget()
        self.tab_2.setObjectName("tab_2")
        self.tabWidget.addTab(self.tab_2, "")
        self.pushButton = QtWidgets.QPushButton(self.centralwidget)
        self.pushButton.setGeometry(QtCore.QRect(230, 290, 75, 23))
        self.pushButton.setObjectName("pushButton")
        MainWindow.setCentralWidget(self.centralwidget)
        self.menubar = QtWidgets.QMenuBar(MainWindow)
        self.menubar.setGeometry(QtCore.QRect(0, 0, 354, 23))
        self.menubar.setObjectName("menubar")
        MainWindow.setMenuBar(self.menubar)
        self.statusbar = QtWidgets.QStatusBar(MainWindow)
        self.statusbar.setObjectName("statusbar")
        MainWindow.setStatusBar(self.statusbar)

        self.retranslateUi(MainWindow)
        QtCore.QMetaObject.connectSlotsByName(MainWindow)

    def retranslateUi(self, MainWindow):
        _translate = QtCore.QCoreApplication.translate
        MainWindow.setWindowTitle(_translate("MainWindow", "MainWindow"))
        self.tab1_label.setText(_translate("MainWindow", "TextLabel"))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab), _translate("MainWindow", "Tab 1"))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab_2), _translate("MainWindow", "Tab 2"))
        self.pushButton.setText(_translate("MainWindow", "PushButton"))

```

```python

if __name__ == "__main__":
    import sys
    app = QtWidgets.QApplication(sys.argv)
    MainWindow = QtWidgets.QMainWindow()
    ui = Ui_MainWindow()
    ui.setupUi(MainWindow)
    MainWindow.show()
    sys.exit(app.exec_())
```

1.2 designer/child_setting.ui

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>MainWindow</class>
 <widget class="QMainWindow" name="MainWindow">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>354</width>
    <height>366</height>
   </rect>
  </property>
  <property name="windowTitle">
   <string>MainWindow</string>
  </property>
  <widget class="QWidget" name="centralwidget">
   <widget class="QTabWidget" name="tabWidget">
    <property name="geometry">
     <rect>
      <x>10</x>
      <y>20</y>
      <width>321</width>
      <height>251</height>
     </rect>
    </property>
    <widget class="QWidget" name="tab">
     <attribute name="title">
      <string>Tab 1</string>
     </attribute>
     <widget class="QComboBox" name="tab1_combobox">
      <property name="geometry">
       <rect>
        <x>70</x>
        <y>0</y>
        <width>171</width>
        <height>22</height>
       </rect>
      </property>
     </widget>
     <widget class="QLabel" name="tab1_label">
      <property name="geometry">
       <rect>
        <x>0</x>
        <y>30</y>
        <width>311</width>
        <height>191</height>
       </rect>
      </property>
      <property name="text">
       <string>TextLabel</string>
      </property>
     </widget>
    </widget>
    <widget class="QWidget" name="tab_2">
     <attribute name="title">
      <string>Tab 2</string>
     </attribute>
    </widget>
   </widget>
   <widget class="QPushButton" name="pushButton">
    <property name="geometry">
     <rect>
      <x>230</x>
      <y>290</y>
      <width>75</width>
```

```
      <height>23</height>
     </rect>
    </property>
    <property name="text">
     <string>PushButton</string>
    </property>
   </widget>
  </widget>
  <widget class="QMenuBar" name="menubar">
   <property name="geometry">
    <rect>
     <x>0</x>
     <y>0</y>
     <width>354</width>
     <height>23</height>
    </rect>
   </property>
  </widget>
  <widget class="QStatusBar" name="statusbar"/>
 </widget>
 <resources/>
 <connections/>
</ui>
```

1.3 designer/child_test.py

```
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'child_test.ui'
#
# Created by: PyQt5 UI code generator 5.9.2
#
# WARNING! All changes made in this file will be lost!

from PyQt5 import QtCore, QtGui, QtWidgets

class Ui_Form(object):
    def setupUi(self, Form):
        Form.setObjectName("Form")
        Form.resize(400, 310)
        self.tabWidget = QtWidgets.QTabWidget(Form)
        self.tabWidget.setGeometry(QtCore.QRect(10, 10, 381,
        ↪   261))
        self.tabWidget.setObjectName("tabWidget")
        self.tab = QtWidgets.QWidget()
        self.tab.setObjectName("tab")
        self.layoutWidget = QtWidgets.QWidget(self.tab)
        self.layoutWidget.setGeometry(QtCore.QRect(0, 10, 371,
        ↪   211))
        self.layoutWidget.setObjectName("layoutWidget")
        self.horizontalLayout_2 =
        ↪   QtWidgets.QHBoxLayout(self.layoutWidget)
        self.horizontalLayout_2.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout_2.setObjectName("horizontalLayo⌟
        ↪   ut_2")
        spacerItem = QtWidgets.QSpacerItem(40, 20,
        ↪   QtWidgets.QSizePolicy.Expanding,
        ↪   QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_2.addItem(spacerItem)
        self.verticalLayout_2 = QtWidgets.QVBoxLayout()
        self.verticalLayout_2.setObjectName("verticalLayout_2⌟
        ↪   ")
        self.verticalLayout = QtWidgets.QVBoxLayout()
        self.verticalLayout.setObjectName("verticalLayout")
        self.horizontalLayout = QtWidgets.QHBoxLayout()
        self.horizontalLayout.setObjectName("horizontalLayout⌟
        ↪   ")
        spacerItem1 = QtWidgets.QSpacerItem(30, 20,
        ↪   QtWidgets.QSizePolicy.Maximum,
        ↪   QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout.addItem(spacerItem1)
        self.tab1_combobox =
        ↪   QtWidgets.QComboBox(self.layoutWidget)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePol⌟
        ↪   icy.MinimumExpanding,
        ↪   QtWidgets.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.tab1_combobox.sizePolicy().hasHeightForWidth())
        self.tab1_combobox.setSizePolicy(sizePolicy)
        self.tab1_combobox.setObjectName("tab1_combobox")
        self.horizontalLayout.addWidget(self.tab1_combobox)
        spacerItem2 = QtWidgets.QSpacerItem(30, 20, QtWidgets.QSizePolicy.Maximum, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout.addItem(spacerItem2)
        self.verticalLayout.addLayout(self.horizontalLayout)
        spacerItem3 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.verticalLayout.addItem(spacerItem3)
        self.verticalLayout_2.addLayout(self.verticalLayout)
        self.tab1_label = QtWidgets.QLabel(self.layoutWidget)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Ignored, QtWidgets.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.tab1_label.sizePolicy().hasHeightForWidth())
        self.tab1_label.setSizePolicy(sizePolicy)
        self.tab1_label.setMinimumSize(QtCore.QSize(256, 144))
        self.tab1_label.setMaximumSize(QtCore.QSize(256, 144))
        self.tab1_label.setAlignment(QtCore.Qt.AlignCenter)
        self.tab1_label.setObjectName("tab1_label")
        self.verticalLayout_2.addWidget(self.tab1_label)
        self.horizontalLayout_2.addLayout(self.verticalLayout_2)
        spacerItem4 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_2.addItem(spacerItem4)
        self.tabWidget.addTab(self.tab, "")
        self.tab_2 = QtWidgets.QWidget()
        self.tab_2.setObjectName("tab_2")
        self.layoutWidget_2 = QtWidgets.QWidget(self.tab_2)
        self.layoutWidget_2.setGeometry(QtCore.QRect(0, 10, 371, 211))
        self.layoutWidget_2.setObjectName("layoutWidget_2")
        self.horizontalLayout_7 = QtWidgets.QHBoxLayout(self.layoutWidget_2)
        self.horizontalLayout_7.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout_7.setObjectName("horizontalLayout_7")
        spacerItem5 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_7.addItem(spacerItem5)
        self.verticalLayout_5 = QtWidgets.QVBoxLayout()
        self.verticalLayout_5.setObjectName("verticalLayout_5")
        self.verticalLayout_6 = QtWidgets.QVBoxLayout()
        self.verticalLayout_6.setObjectName("verticalLayout_6")
        self.horizontalLayout_8 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_8.setObjectName("horizontalLayout_8")
        spacerItem6 = QtWidgets.QSpacerItem(30, 20, QtWidgets.QSizePolicy.Maximum, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_8.addItem(spacerItem6)
        self.tab1_combobox_3 = QtWidgets.QComboBox(self.layoutWidget_2)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.MinimumExpanding, QtWidgets.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.tab1_combobox_3.sizePolicy().hasHeightForWidth())
        self.tab1_combobox_3.setSizePolicy(sizePolicy)
        self.tab1_combobox_3.setObjectName("tab1_combobox_3")
        self.horizontalLayout_8.addWidget(self.tab1_combobox_3)
        spacerItem7 = QtWidgets.QSpacerItem(30, 20, QtWidgets.QSizePolicy.Maximum, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_8.addItem(spacerItem7)
        self.verticalLayout_6.addLayout(self.horizontalLayout_8)
        spacerItem8 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.verticalLayout_6.addItem(spacerItem8)
        self.verticalLayout_5.addLayout(self.verticalLayout_6)
        self.tab1_label_3 = QtWidgets.QLabel(self.layoutWidget_2)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Ignored, QtWidgets.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.tab1_label_3.sizePolicy().hasHeightForWidth())
        self.tab1_label_3.setSizePolicy(sizePolicy)
        self.tab1_label_3.setMinimumSize(QtCore.QSize(256, 144))
        self.tab1_label_3.setMaximumSize(QtCore.QSize(256, 144))
        self.tab1_label_3.setAlignment(QtCore.Qt.AlignCenter)
        self.tab1_label_3.setObjectName("tab1_label_3")
        self.verticalLayout_5.addWidget(self.tab1_label_3)
        self.horizontalLayout_7.addLayout(self.verticalLayout_5)
        spacerItem9 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_7.addItem(spacerItem9)
        self.tabWidget.addTab(self.tab_2, "")
        self.tab_3 = QtWidgets.QWidget()
        self.tab_3.setObjectName("tab_3")
        self.layoutWidget_3 = QtWidgets.QWidget(self.tab_3)
        self.layoutWidget_3.setGeometry(QtCore.QRect(0, 10, 371, 211))
        self.layoutWidget_3.setObjectName("layoutWidget_3")
        self.horizontalLayout_4 = QtWidgets.QHBoxLayout(self.layoutWidget_3)
        self.horizontalLayout_4.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout_4.setObjectName("horizontalLayout_4")
        spacerItem10 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_4.addItem(spacerItem10)
        self.verticalLayout_7 = QtWidgets.QVBoxLayout()
        self.verticalLayout_7.setObjectName("verticalLayout_7")
        self.verticalLayout_8 = QtWidgets.QVBoxLayout()
        self.verticalLayout_8.setObjectName("verticalLayout_8")
        self.horizontalLayout_9 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_9.setObjectName("horizontalLayout_9")
        spacerItem11 = QtWidgets.QSpacerItem(30, 20, QtWidgets.QSizePolicy.Maximum, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_9.addItem(spacerItem11)
        self.tab1_combobox_4 = QtWidgets.QComboBox(self.layoutWidget_3)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.MinimumExpanding, QtWidgets.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.tab1_combobox_4.sizePolicy().hasHeightForWidth())
        self.tab1_combobox_4.setSizePolicy(sizePolicy)
        self.tab1_combobox_4.setObjectName("tab1_combobox_4")
        self.horizontalLayout_9.addWidget(self.tab1_combobox_4)
        spacerItem12 = QtWidgets.QSpacerItem(30, 20, QtWidgets.QSizePolicy.Maximum, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_9.addItem(spacerItem12)
        self.verticalLayout_8.addLayout(self.horizontalLayout_9)
        spacerItem13 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.verticalLayout_8.addItem(spacerItem13)
        self.verticalLayout_7.addLayout(self.verticalLayout_8)
        self.tab1_label_4 = QtWidgets.QLabel(self.layoutWidget_3)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Ignored, QtWidgets.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.tab1_label_4.sizePolicy().hasHeightForWidth())
        self.tab1_label_4.setSizePolicy(sizePolicy)
        self.tab1_label_4.setMinimumSize(QtCore.QSize(256, 144))
        self.tab1_label_4.setMaximumSize(QtCore.QSize(256, 144))
        self.tab1_label_4.setAlignment(QtCore.Qt.AlignCenter)
        self.tab1_label_4.setObjectName("tab1_label_4")
        self.verticalLayout_7.addWidget(self.tab1_label_4)
        self.horizontalLayout_4.addLayout(self.verticalLayout_7)
        spacerItem14 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_4.addItem(spacerItem14)
        self.tabWidget.addTab(self.tab_3, "")
        self.tab_4 = QtWidgets.QWidget()
        self.tab_4.setObjectName("tab_4")
        self.layoutWidget1 = QtWidgets.QWidget(self.tab_4)
```

```python
        self.layoutWidget1.setGeometry(QtCore.QRect(0, 10, 371, 211))
        self.layoutWidget1.setObjectName("layoutWidget1")
        self.horizontalLayout_13 = QtWidgets.QHBoxLayout(self.layoutWidget1)
        self.horizontalLayout_13.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout_13.setObjectName("horizontalLayout_13")
        spacerItem15 = QtWidgets.QSpacerItem(40, 40, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_13.addItem(spacerItem15)
        self.verticalLayout_13 = QtWidgets.QVBoxLayout()
        self.verticalLayout_13.setObjectName("verticalLayout_13")
        self.verticalLayout_14 = QtWidgets.QVBoxLayout()
        self.verticalLayout_14.setObjectName("verticalLayout_14")
        self.horizontalLayout_14 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_14.setObjectName("horizontalLayout_14")
        spacerItem16 = QtWidgets.QSpacerItem(31, 20, QtWidgets.QSizePolicy.Maximum, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_14.addItem(spacerItem16)
        self.tab1_combobox_7 = QtWidgets.QComboBox(self.layoutWidget1)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.MinimumExpanding, QtWidgets.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.tab1_combobox_7.sizePolicy().hasHeightForWidth())
        self.tab1_combobox_7.setSizePolicy(sizePolicy)
        self.tab1_combobox_7.setMinimumSize(QtCore.QSize(160, 0))
        self.tab1_combobox_7.setObjectName("tab1_combobox_7")
        self.horizontalLayout_14.addWidget(self.tab1_combobox_7)
        spacerItem17 = QtWidgets.QSpacerItem(31, 20, QtWidgets.QSizePolicy.Maximum, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_14.addItem(spacerItem17)
        self.verticalLayout_14.addLayout(self.horizontalLayout_14)
        spacerItem18 = QtWidgets.QSpacerItem(256, 55, QtWidgets.QSizePolicy.Minimum, QtWidgets.QSizePolicy.Minimum)
        self.verticalLayout_14.addItem(spacerItem18)
        self.verticalLayout_13.addLayout(self.verticalLayout_14)
        self.progressBar = QtWidgets.QProgressBar(self.layoutWidget1)
        self.progressBar.setProperty("value", 0)
        self.progressBar.setTextVisible(False)
        self.progressBar.setObjectName("progressBar")
        self.verticalLayout_13.addWidget(self.progressBar)
        spacerItem19 = QtWidgets.QSpacerItem(40, 60, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.verticalLayout_13.addItem(spacerItem19)
        self.horizontalLayout_13.addLayout(self.verticalLayout_13)
        spacerItem20 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_13.addItem(spacerItem20)
        self.tabWidget.addTab(self.tab_4, "")
        self.OK = QtWidgets.QPushButton(Form)
        self.OK.setGeometry(QtCore.QRect(260, 280, 111, 23))
        self.OK.setObjectName("OK")

        self.retranslateUi(Form)
        self.tabWidget.setCurrentIndex(0)
        QtCore.QMetaObject.connectSlotsByName(Form)

```

```
    def retranslateUi(self, Form):
        _translate = QtCore.QCoreApplication.translate
        Form.setWindowTitle(_translate("Form", "Form"))
        self.tab1_label.setText(_translate("Form", "TextLabel"))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab), _translate("Form", "行为"))
        self.tab1_label_3.setText(_translate("Form", "TextLabel"))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab_2), _translate("Form", "面部"))
        self.tab1_label_4.setText(_translate("Form", "TextLabel"))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab_3), _translate("Form", "眼睛"))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab_4), _translate("Form", "麦克风"))
        self.OK.setText(_translate("Form", "确认"))

```

1.4 designer/child_test.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>Form</class>
 <widget class="QWidget" name="Form">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>400</width>
    <height>310</height>
   </rect>
  </property>
```

```xml
 <property name="windowTitle">
  <string>Form</string>
 </property>
 <widget class="QTabWidget" name="tabWidget">
  <property name="geometry">
   <rect>
    <x>10</x>
    <y>10</y>
    <width>381</width>
    <height>261</height>
   </rect>
  </property>
  <property name="currentIndex">
   <number>0</number>
  </property>
  <widget class="QWidget" name="tab">
   <attribute name="title">
    <string>行为</string>
   </attribute>
   <widget class="QWidget" name="layoutWidget">
    <property name="geometry">
     <rect>
      <x>0</x>
      <y>10</y>
      <width>371</width>
      <height>211</height>
     </rect>
    </property>
    <layout class="QHBoxLayout" name="horizontalLayout_2">
     <item>
      <spacer name="horizontalSpacer_10">
       <property name="orientation">
        <enum>Qt::Horizontal</enum>
       </property>
       <property name="sizeHint" stdset="0">
        <size>
         <width>40</width>
         <height>20</height>
        </size>
       </property>
      </spacer>
     </item>
     <item>
      <layout class="QVBoxLayout" name="verticalLayout_2">
       <item>
        <layout class="QVBoxLayout" name="verticalLayout">
         <item>
          <layout class="QHBoxLayout" name="horizontalLayout">
           <item>
            <spacer name="horizontalSpacer">
             <property name="orientation">
              <enum>Qt::Horizontal</enum>
             </property>
             <property name="sizeType">
              <enum>QSizePolicy::Maximum</enum>
             </property>
             <property name="sizeHint" stdset="0">
              <size>
               <width>30</width>
               <height>20</height>
              </size>
             </property>
            </spacer>
           </item>
           <item>
            <widget class="QComboBox" name="tab1_combobox">
             <property name="sizePolicy">
              <sizepolicy hsizetype="MinimumExpanding" vsizetype="Fixed">
               <horstretch>0</horstretch>
               <verstretch>0</verstretch>
              </sizepolicy>
             </property>
            </widget>
           </item>
           <item>
            <spacer name="horizontalSpacer_2">
```

```
               <property name="orientation">
                <enum>Qt::Horizontal</enum>
               </property>
               <property name="sizeType">
                <enum>QSizePolicy::Maximum</enum>
               </property>
               <property name="sizeHint" stdset="0">
                <size>
                 <width>30</width>
                 <height>20</height>
                </size>
               </property>
              </spacer>
             </item>
            </layout>
           </item>
           <item>
            <spacer name="horizontalSpacer_9">
             <property name="orientation">
              <enum>Qt::Horizontal</enum>
             </property>
             <property name="sizeHint" stdset="0">
              <size>
               <width>40</width>
               <height>20</height>
              </size>
             </property>
            </spacer>
           </item>
          </layout>
         </item>
         <item>
          <widget class="QLabel" name="tab1_label">
           <property name="sizePolicy">
            <sizepolicy hsizetype="Ignored" vsizetype="Fixed">
             <horstretch>0</horstretch>
             <verstretch>0</verstretch>
            </sizepolicy>
           </property>
```

```
       <property name="minimumSize">
        <size>
         <width>256</width>
         <height>144</height>
        </size>
       </property>
       <property name="maximumSize">
        <size>
         <width>256</width>
         <height>144</height>
        </size>
       </property>
       <property name="text">
        <string>TextLabel</string>
       </property>
       <property name="alignment">
        <set>Qt::AlignCenter</set>
       </property>
      </widget>
     </item>
    </layout>
   </item>
   <item>
    <spacer name="horizontalSpacer_11">
     <property name="orientation">
      <enum>Qt::Horizontal</enum>
     </property>
     <property name="sizeHint" stdset="0">
      <size>
       <width>40</width>
       <height>20</height>
      </size>
     </property>
    </spacer>
   </item>
  </layout>
 </widget>
</widget>
<widget class="QWidget" name="tab_2">
```

```
     <attribute name="title">
      <string>面部</string>
     </attribute>
     <widget class="QWidget" name="layoutWidget_2">
      <property name="geometry">
       <rect>
        <x>0</x>
        <y>10</y>
        <width>371</width>
        <height>211</height>
       </rect>
      </property>
      <layout class="QHBoxLayout" name="horizontalLayout_7">
       <item>
        <spacer name="horizontalSpacer_15">
         <property name="orientation">
          <enum>Qt::Horizontal</enum>
         </property>
         <property name="sizeHint" stdset="0">
          <size>
           <width>40</width>
           <height>20</height>
          </size>
         </property>
        </spacer>
       </item>
       <item>
        <layout class="QVBoxLayout" name="verticalLayout_5">
         <item>
          <layout class="QVBoxLayout" name="verticalLayout_6">
           <item>
            <layout class="QHBoxLayout"
             ↪  name="horizontalLayout_8">
             <item>
              <spacer name="horizontalSpacer_16">
               <property name="orientation">
                <enum>Qt::Horizontal</enum>
               </property>
               <property name="sizeType">
                <enum>QSizePolicy::Maximum</enum>
               </property>
               <property name="sizeHint" stdset="0">
                <size>
                 <width>30</width>
                 <height>20</height>
                </size>
               </property>
              </spacer>
             </item>
             <item>
              <widget class="QComboBox" name="tab1_combobox_3">
               <property name="sizePolicy">
                <sizepolicy hsizetype="MinimumExpanding"
                ↪  vsizetype="Fixed">
                 <horstretch>0</horstretch>
                 <verstretch>0</verstretch>
                </sizepolicy>
               </property>
              </widget>
             </item>
             <item>
              <spacer name="horizontalSpacer_17">
               <property name="orientation">
                <enum>Qt::Horizontal</enum>
               </property>
               <property name="sizeType">
                <enum>QSizePolicy::Maximum</enum>
               </property>
               <property name="sizeHint" stdset="0">
                <size>
                 <width>30</width>
                 <height>20</height>
                </size>
               </property>
              </spacer>
             </item>
            </layout>
           </item>
```

```
  <item>
   <spacer name="horizontalSpacer_18">
    <property name="orientation">
     <enum>Qt::Horizontal</enum>
    </property>
    <property name="sizeHint" stdset="0">
     <size>
      <width>40</width>
      <height>20</height>
     </size>
    </property>
   </spacer>
  </item>
 </layout>
</item>
<item>
 <widget class="QLabel" name="tab1_label_3">
  <property name="sizePolicy">
   <sizepolicy hsizetype="Ignored" vsizetype="Fixed">
    <horstretch>0</horstretch>
    <verstretch>0</verstretch>
   </sizepolicy>
  </property>
  <property name="minimumSize">
   <size>
    <width>256</width>
    <height>144</height>
   </size>
  </property>
  <property name="maximumSize">
   <size>
    <width>256</width>
    <height>144</height>
   </size>
  </property>
  <property name="text">
   <string>TextLabel</string>
  </property>
  <property name="alignment">
```

```
        <set>Qt::AlignCenter</set>
       </property>
      </widget>
     </item>
    </layout>
   </item>
   <item>
    <spacer name="horizontalSpacer_19">
     <property name="orientation">
      <enum>Qt::Horizontal</enum>
     </property>
     <property name="sizeHint" stdset="0">
      <size>
       <width>40</width>
       <height>20</height>
      </size>
     </property>
    </spacer>
   </item>
  </layout>
 </widget>
</widget>
<widget class="QWidget" name="tab_3">
 <attribute name="title">
  <string>眼睛</string>
 </attribute>
 <widget class="QWidget" name="layoutWidget_3">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>10</y>
    <width>371</width>
    <height>211</height>
   </rect>
  </property>
  <layout class="QHBoxLayout" name="horizontalLayout_4">
   <item>
    <spacer name="horizontalSpacer_20">
     <property name="orientation">
      <enum>Qt::Horizontal</enum>
     </property>
     <property name="sizeHint" stdset="0">
      <size>
       <width>40</width>
       <height>20</height>
      </size>
     </property>
    </spacer>
   </item>
   <item>
    <layout class="QVBoxLayout" name="verticalLayout_7">
     <item>
      <layout class="QVBoxLayout" name="verticalLayout_8">
       <item>
        <layout class="QHBoxLayout" name="horizontalLayout_9">
         <item>
          <spacer name="horizontalSpacer_5">
           <property name="orientation">
            <enum>Qt::Horizontal</enum>
           </property>
           <property name="sizeType">
            <enum>QSizePolicy::Maximum</enum>
           </property>
           <property name="sizeHint" stdset="0">
            <size>
             <width>30</width>
             <height>20</height>
            </size>
           </property>
          </spacer>
         </item>
         <item>
          <widget class="QComboBox" name="tab1_combobox_4">
           <property name="sizePolicy">
            <sizepolicy hsizetype="MinimumExpanding" vsizetype="Fixed">
             <horstretch>0</horstretch>
```

```
                <verstretch>0</verstretch>
               </sizepolicy>
              </property>
             </widget>
            </item>
            <item>
             <spacer name="horizontalSpacer_6">
              <property name="orientation">
               <enum>Qt::Horizontal</enum>
              </property>
              <property name="sizeType">
               <enum>QSizePolicy::Maximum</enum>
              </property>
              <property name="sizeHint" stdset="0">
               <size>
                <width>30</width>
                <height>20</height>
               </size>
              </property>
             </spacer>
            </item>
           </layout>
          </item>
          <item>
           <spacer name="horizontalSpacer_21">
            <property name="orientation">
             <enum>Qt::Horizontal</enum>
            </property>
            <property name="sizeHint" stdset="0">
             <size>
              <width>40</width>
              <height>20</height>
             </size>
            </property>
           </spacer>
          </item>
         </layout>
        </item>
        <item>
```

```
          <widget class="QLabel" name="tab1_label_4">
           <property name="sizePolicy">
            <sizepolicy hsizetype="Ignored" vsizetype="Fixed">
             <horstretch>0</horstretch>
             <verstretch>0</verstretch>
            </sizepolicy>
           </property>
           <property name="minimumSize">
            <size>
             <width>256</width>
             <height>144</height>
            </size>
           </property>
           <property name="maximumSize">
            <size>
             <width>256</width>
             <height>144</height>
            </size>
           </property>
           <property name="text">
            <string>TextLabel</string>
           </property>
           <property name="alignment">
            <set>Qt::AlignCenter</set>
           </property>
          </widget>
         </item>
        </layout>
       </item>
       <item>
        <spacer name="horizontalSpacer_22">
         <property name="orientation">
          <enum>Qt::Horizontal</enum>
         </property>
         <property name="sizeHint" stdset="0">
          <size>
           <width>40</width>
           <height>20</height>
          </size>
```

```
     </property>
    </spacer>
   </item>
  </layout>
 </widget>
</widget>
<widget class="QWidget" name="tab_4">
 <attribute name="title">
  <string>麦克风</string>
 </attribute>
 <widget class="QWidget" name="layoutWidget">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>10</y>
    <width>371</width>
    <height>211</height>
   </rect>
  </property>
  <layout class="QHBoxLayout" name="horizontalLayout_13">
   <item>
    <spacer name="horizontalSpacer_31">
     <property name="orientation">
      <enum>Qt::Horizontal</enum>
     </property>
     <property name="sizeHint" stdset="0">
      <size>
       <width>40</width>
       <height>40</height>
      </size>
     </property>
    </spacer>
   </item>
   <item>
    <layout class="QVBoxLayout" name="verticalLayout_13">
     <item>
      <layout class="QVBoxLayout" name="verticalLayout_14">
       <item>
```

```
<layout class="QHBoxLayout"
↪  name="horizontalLayout_14">
 <item>
  <spacer name="horizontalSpacer_32">
   <property name="orientation">
    <enum>Qt::Horizontal</enum>
   </property>
   <property name="sizeType">
    <enum>QSizePolicy::Maximum</enum>
   </property>
   <property name="sizeHint" stdset="0">
    <size>
     <width>31</width>
     <height>20</height>
    </size>
   </property>
  </spacer>
 </item>
 <item>
  <widget class="QComboBox" name="tab1_combobox_7">
   <property name="sizePolicy">
    <sizepolicy hsizetype="MinimumExpanding"
    ↪  vsizetype="Fixed">
     <horstretch>0</horstretch>
     <verstretch>0</verstretch>
    </sizepolicy>
   </property>
   <property name="minimumSize">
    <size>
     <width>160</width>
     <height>0</height>
    </size>
   </property>
  </widget>
 </item>
 <item>
  <spacer name="horizontalSpacer_33">
   <property name="orientation">
    <enum>Qt::Horizontal</enum>
```

```
               </property>
               <property name="sizeType">
                <enum>QSizePolicy::Maximum</enum>
               </property>
               <property name="sizeHint" stdset="0">
                <size>
                 <width>31</width>
                 <height>20</height>
                </size>
               </property>
              </spacer>
             </item>
            </layout>
           </item>
           <item>
            <spacer name="horizontalSpacer_34">
             <property name="orientation">
              <enum>Qt::Horizontal</enum>
             </property>
             <property name="sizeType">
              <enum>QSizePolicy::Minimum</enum>
             </property>
             <property name="sizeHint" stdset="0">
              <size>
               <width>256</width>
               <height>55</height>
              </size>
             </property>
            </spacer>
           </item>
          </layout>
         </item>
         <item>
          <widget class="QProgressBar" name="progressBar">
           <property name="value">
            <number>0</number>
           </property>
           <property name="textVisible">
            <bool>false</bool>
           </property>
          </widget>
         </item>
         <item>
          <spacer name="horizontalSpacer_36">
           <property name="orientation">
            <enum>Qt::Horizontal</enum>
           </property>
           <property name="sizeHint" stdset="0">
            <size>
             <width>40</width>
             <height>60</height>
            </size>
           </property>
          </spacer>
         </item>
        </layout>
       </item>
       <item>
        <spacer name="horizontalSpacer_35">
         <property name="orientation">
          <enum>Qt::Horizontal</enum>
         </property>
         <property name="sizeHint" stdset="0">
          <size>
           <width>40</width>
           <height>20</height>
          </size>
         </property>
        </spacer>
       </item>
      </layout>
     </widget>
    </widget>
   </widget>
   <widget class="QPushButton" name="OK">
    <property name="geometry">
     <rect>
      <x>260</x>
      <y>280</y>
      <width>111</width>
      <height>23</height>
     </rect>
    </property>
    <property name="text">
     <string>确认</string>
    </property>
   </widget>
  </widget>
  <resources/>
  <connections/>
 </ui>
```

1.5 designer/gui.py

```python
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'gui.ui'
#
# Created by: PyQt5 UI code generator 5.15.10
#
# WARNING: Any manual changes made to this file will be lost when pyuic5 is
# run again.  Do not edit this file unless you know what you are doing.


from PyQt5 import QtCore, QtGui, QtWidgets


class Ui_MainWindow(object):
    def setupUi(self, MainWindow):
        MainWindow.setObjectName("MainWindow")
        MainWindow.resize(1048, 1025)
        icon = QtGui.QIcon()
        icon.addPixmap(QtGui.QPixmap("../icon.ico"), QtGui.QIcon.Normal, QtGui.QIcon.Off)
        MainWindow.setWindowIcon(icon)
        self.centralwidget = QtWidgets.QWidget(MainWindow)
        self.centralwidget.setObjectName("centralwidget")
        self.verticalLayout_4 =
        ↪   QtWidgets.QVBoxLayout(self.centralwidget)
        self.verticalLayout_4.setObjectName("verticalLayout_4⌟
        ↪   ")
        self.h_img_lay = QtWidgets.QHBoxLayout()
        self.h_img_lay.setObjectName("h_img_lay")
        self.original_img_v_lay = QtWidgets.QVBoxLayout()
        self.original_img_v_lay.setObjectName("original_img_v⌟
        ↪   _lay")
        spacerItem = QtWidgets.QSpacerItem(20, 40,
        ↪   QtWidgets.QSizePolicy.Minimum,
        ↪   QtWidgets.QSizePolicy.Expanding)
        self.original_img_v_lay.addItem(spacerItem)
        self.original_name_lbl =
        ↪   QtWidgets.QLabel(self.centralwidget)
        self.original_name_lbl.setScaledContents(False)
        self.original_name_lbl.setAlignment(QtCore.Qt.AlignCe⌟
        ↪   nter)
        self.original_name_lbl.setObjectName("original_name_l⌟
        ↪   bl")
        self.original_img_v_lay.addWidget(self.original_name_⌟
        ↪   lbl)
        spacerItem1 = QtWidgets.QSpacerItem(20, 40,
        ↪   QtWidgets.QSizePolicy.Minimum,
        ↪   QtWidgets.QSizePolicy.Expanding)
        self.original_img_v_lay.addItem(spacerItem1)
        self.original_frame_lbl =
        ↪   QtWidgets.QLabel(self.centralwidget)
        self.original_frame_lbl.setMinimumSize(QtCore.QSize(5⌟
        ↪   00,
        ↪   300))
        self.original_frame_lbl.setObjectName("original_frame⌟
        ↪   _lbl")
        self.original_img_v_lay.addWidget(self.original_frame⌟
        ↪   _lbl)
        self.selecamera1 =
        ↪  QtWidgets.QComboBox(self.centralwidget)
        self.selecamera1.setObjectName("selecamera1")
        self.original_img_v_lay.addWidget(self.selecamera1)
        spacerItem2 = QtWidgets.QSpacerItem(20, 40,
        ↪  QtWidgets.QSizePolicy.Minimum,
        ↪  QtWidgets.QSizePolicy.Expanding)
        self.original_img_v_lay.addItem(spacerItem2)
        self.h_img_lay.addLayout(self.original_img_v_lay)
        self.processed_img_v_lay = QtWidgets.QVBoxLayout()
        self.processed_img_v_lay.setObjectName("processed_img⌟
        ↪  _v_lay")
        spacerItem3 = QtWidgets.QSpacerItem(20, 40,
        ↪  QtWidgets.QSizePolicy.Minimum,
        ↪  QtWidgets.QSizePolicy.Expanding)
        self.processed_img_v_lay.addItem(spacerItem3)
        self.processed_name_lbl =
        ↪  QtWidgets.QLabel(self.centralwidget)
        self.processed_name_lbl.setAlignment(QtCore.Qt.AlignC⌟
        ↪  enter)
        self.processed_name_lbl.setObjectName("processed_name⌟
        ↪  _lbl")
        self.processed_img_v_lay.addWidget(self.processed_nam⌟
        ↪  e_lbl)
        spacerItem4 = QtWidgets.QSpacerItem(20, 40,
        ↪  QtWidgets.QSizePolicy.Minimum,
        ↪  QtWidgets.QSizePolicy.Expanding)
        self.processed_img_v_lay.addItem(spacerItem4)
        self.processed_frame_lbl =
        ↪  QtWidgets.QLabel(self.centralwidget)
        self.processed_frame_lbl.setMinimumSize(QtCore.QSize(⌟
        ↪  500,
        ↪  300))
        self.processed_frame_lbl.setBaseSize(QtCore.QSize(0,
        ↪  0))
        self.processed_frame_lbl.setText("")
        self.processed_frame_lbl.setObjectName("processed_fra⌟
        ↪  me_lbl")
        self.processed_img_v_lay.addWidget(self.processed_frame_lbl)
        self.selecamera2 = QtWidgets.QComboBox(self.centralwidget)
        self.selecamera2.setObjectName("selecamera2")
        self.processed_img_v_lay.addWidget(self.selecamera2)
        spacerItem5 = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum, QtWidgets.QSizePolicy.Expanding)
        self.processed_img_v_lay.addItem(spacerItem5)
        self.h_img_lay.addLayout(self.processed_img_v_lay)
        self.verticalLayout_4.addLayout(self.h_img_lay)
        self.horizontalLayout = QtWidgets.QHBoxLayout()
        self.horizontalLayout.setObjectName("horizontalLayout")
        self.verticalLayout_2 = QtWidgets.QVBoxLayout()
        self.verticalLayout_2.setObjectName("verticalLayout_2")
        self.label_3 = QtWidgets.QLabel(self.centralwidget)
        self.label_3.setObjectName("label_3")
        self.verticalLayout_2.addWidget(self.label_3)
        self.eye_lbl = QtWidgets.QLabel(self.centralwidget)
        self.eye_lbl.setMinimumSize(QtCore.QSize(500, 300))
        self.eye_lbl.setText("")
        self.eye_lbl.setObjectName("eye_lbl")
        self.verticalLayout_2.addWidget(self.eye_lbl)
        self.comboBox_2 = QtWidgets.QComboBox(self.centralwidget)
        self.comboBox_2.setObjectName("comboBox_2")
        self.verticalLayout_2.addWidget(self.comboBox_2)
        self.horizontalLayout.addLayout(self.verticalLayout_2)
        self.verticalLayout = QtWidgets.QVBoxLayout()
        self.verticalLayout.setObjectName("verticalLayout")
        self.label_2 = QtWidgets.QLabel(self.centralwidget)
        self.label_2.setObjectName("label_2")
        self.verticalLayout.addWidget(self.label_2)
        self.voice_lbl = QtWidgets.QLabel(self.centralwidget)
        self.voice_lbl.setMinimumSize(QtCore.QSize(500, 300))
        self.voice_lbl.setText("")
        self.voice_lbl.setObjectName("voice_lbl")
        self.verticalLayout.addWidget(self.voice_lbl)
        self.comboBox =
        ↪  QtWidgets.QComboBox(self.centralwidget)
        self.comboBox.setObjectName("comboBox")
        self.verticalLayout.addWidget(self.comboBox)
        self.horizontalLayout.addLayout(self.verticalLayout)
        self.verticalLayout_4.addLayout(self.horizontalLayout)
        self.h_btn_lay = QtWidgets.QHBoxLayout()
        self.h_btn_lay.setObjectName("h_btn_lay")
        self.initButton =
        ↪  QtWidgets.QPushButton(self.centralwidget)
        self.initButton.setObjectName("initButton")
        self.h_btn_lay.addWidget(self.initButton)
        self.pushButton =
        ↪  QtWidgets.QPushButton(self.centralwidget)
        self.pushButton.setObjectName("pushButton")
        self.h_btn_lay.addWidget(self.pushButton)
        spacerItem6 = QtWidgets.QSpacerItem(40, 20,
        ↪  QtWidgets.QSizePolicy.Minimum,
        ↪  QtWidgets.QSizePolicy.Expanding)
        self.h_btn_lay.addItem(spacerItem6)
        self.label = QtWidgets.QLabel(self.centralwidget)
        self.label.setObjectName("label")
        self.h_btn_lay.addWidget(self.label)
        self.lineEdit =
        ↪  QtWidgets.QLineEdit(self.centralwidget)
        self.lineEdit.setObjectName("lineEdit")
        self.h_btn_lay.addWidget(self.lineEdit)
        self.startButton =
        ↪  QtWidgets.QPushButton(self.centralwidget)
        self.startButton.setObjectName("startButton")
        self.h_btn_lay.addWidget(self.startButton)
        self.endButton =
        ↪  QtWidgets.QPushButton(self.centralwidget)
        self.endButton.setObjectName("endButton")
        self.h_btn_lay.addWidget(self.endButton)
        self.shotButton =
        ↪  QtWidgets.QPushButton(self.centralwidget)
        self.shotButton.setObjectName("shotButton")
        self.h_btn_lay.addWidget(self.shotButton)
        self.verticalLayout_4.addLayout(self.h_btn_lay)
        self.v_filters_lay = QtWidgets.QVBoxLayout()
        self.v_filters_lay.setObjectName("v_filters_lay")
        self.verticalLayout_4.addLayout(self.v_filters_lay)
        MainWindow.setCentralWidget(self.centralwidget)
        self.menubar = QtWidgets.QMenuBar(MainWindow)
        self.menubar.setGeometry(QtCore.QRect(0, 0, 1048, 37))
        self.menubar.setObjectName("menubar")
        self.menuHelp = QtWidgets.QMenu(self.menubar)
        self.menuHelp.setObjectName("menuHelp")
        self.menu = QtWidgets.QMenu(self.menubar)
        self.menu.setObjectName("menu")
        self.menu_2 = QtWidgets.QMenu(self.menubar)
        self.menu_2.setObjectName("menu_2")
        MainWindow.setMenuBar(self.menubar)
        self.statusbar = QtWidgets.QStatusBar(MainWindow)
        self.statusbar.setMinimumSize(QtCore.QSize(500, 30))
        self.statusbar.setObjectName("statusbar")
        MainWindow.setStatusBar(self.statusbar)
        self.actionOpen_image = QtWidgets.QAction(MainWindow)
        self.actionOpen_image.setObjectName("actionOpen_image")
        self.actionSave_original_image = QtWidgets.QAction(MainWindow)
        self.actionSave_original_image.setObjectName("actionSave_original_image")
        self.actionSave_processed_image = QtWidgets.QAction(MainWindow)
        self.actionSave_processed_image.setObjectName("actionSave_processed_image")
        self.actionExit = QtWidgets.QAction(MainWindow)
        self.actionExit.setObjectName("actionExit")
        self.actionLicense = QtWidgets.QAction(MainWindow)
        self.actionLicense.setObjectName("actionLicense")
        self.actionAbout = QtWidgets.QAction(MainWindow)
        self.actionAbout.setObjectName("actionAbout")
        self.actionmanual = QtWidgets.QAction(MainWindow)
        self.actionmanual.setObjectName("actionmanual")
        self.action12 = QtWidgets.QAction(MainWindow)
        self.action12.setObjectName("action12")
        self.actionse = QtWidgets.QAction(MainWindow)
        self.actionse.setObjectName("actionse")
        self.menuHelp.addAction(self.actionmanual)
        self.menuHelp.addAction(self.actionAbout)
        self.menu_2.addAction(self.action12)
        self.menu_2.addAction(self.actionse)
        self.menubar.addAction(self.menu.menuAction())
        self.menubar.addAction(self.menu_2.menuAction())
        self.menubar.addAction(self.menuHelp.menuAction())

        self.retranslateUi(MainWindow)
        QtCore.QMetaObject.connectSlotsByName(MainWindow)

    def retranslateUi(self, MainWindow):
        _translate = QtCore.QCoreApplication.translate
        MainWindow.setWindowTitle(_translate("MainWindow", "面部表型特征提取系统---山东省精神卫生中心"))
        self.original_name_lbl.setText(_translate("MainWindow", "原始图像"))
        self.original_frame_lbl.setText(_translate("MainWindow", "<html><head/><body><p><br/></p></body></html>"))
        self.processed_name_lbl.setText(_translate("MainWindow", "面部特征"))
        self.label_3.setText(_translate("MainWindow", "标记图像"))
        self.label_2.setText(_translate("MainWindow", "情绪识别"))
        self.initButton.setText(_translate("MainWindow", "检测视频设备"))
        self.pushButton.setText(_translate("MainWindow", "停用视频设备"))
        self.label.setText(_translate("MainWindow", "样本编号:"))
        self.startButton.setText(_translate("MainWindow", "开始提取"))
        self.endButton.setText(_translate("MainWindow", "结束提取"))
        self.shotButton.setText(_translate("MainWindow", "视频截图"))
        self.menuHelp.setTitle(_translate("MainWindow", "帮助"))
        self.menu.setTitle(_translate("MainWindow", " 文件"))
        self.menu_2.setTitle(_translate("MainWindow", "设置"))
        self.actionOpen_image.setText(_translate("MainWindow", "Open image"))
        self.actionSave_original_image.setText(_translate("MainWindow", "Save original image"))
        self.actionSave_processed_image.setText(_translate("MainWindow", "Save processed image"))
        self.actionExit.setText(_translate("MainWindow", "Exit"))
        self.actionLicense.setText(_translate("MainWindow", "License"))
        self.actionAbout.setText(_translate("MainWindow", "关于"))
        self.actionmanual.setText(_translate("MainWindow", "操作说明"))
        self.action12.setText(_translate("MainWindow", "设置视频与音频设备"))
        self.actionse.setText(_translate("MainWindow", "设置文件保存路径"))


if __name__ == "__main__":
    import sys
    app = QtWidgets.QApplication(sys.argv)
    MainWindow = QtWidgets.QMainWindow()
    ui = Ui_MainWindow()
    ui.setupUi(MainWindow)
    MainWindow.show()
    sys.exit(app.exec_())
```

1.6 designer/gui.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>MainWindow</class>
 <widget class="QMainWindow" name="MainWindow">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>1048</width>
    <height>1025</height>
   </rect>
  </property>
  <property name="windowTitle">
   <string>面部表型特征提取系统---山东省精神卫生中心</string>
  </property>
  <property name="windowIcon">
   <iconset>
    <normaloff>../icon.ico</normaloff>../icon.ico</iconset>
  </property>
  <widget class="QWidget" name="centralwidget">
   <layout class="QVBoxLayout" name="verticalLayout_4">
    <item>
     <layout class="QHBoxLayout" name="h_img_lay">
      <item>
       <layout class="QVBoxLayout" name="original_img_v_lay">
        <item>
         <spacer name="verticalSpacer">
          <property name="sizeHint" stdset="0">
           <size>
            <width>20</width>
            <height>40</height>
           </size>
          </property>
         </spacer>
```

```
          </item>
          <item>
           <widget class="QLabel" name="original_name_lbl">
            <property name="text">
             <string>原始图像</string>
            </property>
            <property name="scaledContents">
             <bool>false</bool>
            </property>
            <property name="alignment">
             <set>Qt::AlignCenter</set>
            </property>
           </widget>
          </item>
          <item>
           <spacer name="verticalSpacer_2">
            <property name="sizeHint" stdset="0">
             <size>
              <width>20</width>
              <height>40</height>
             </size>
            </property>
           </spacer>
          </item>
          <item>
           <widget class="QLabel" name="original_frame_lbl">
            <property name="minimumSize">
             <size>
              <width>500</width>
              <height>300</height>
             </size>
            </property>
            <property name="text">
             <string>&lt;html&gt;&lt;head/&gt;&lt;body&gt;&lt;p&gt;&lt;br/&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>
            </property>
           </widget>
          </item>
```

```
        <item>
         <widget class="QComboBox" name="selecamera1"/>
        </item>
        <item>
         <spacer name="verticalSpacer_3">
          <property name="sizeHint" stdset="0">
           <size>
            <width>20</width>
            <height>40</height>
           </size>
          </property>
         </spacer>
        </item>
       </layout>
      </item>
      <item>
       <layout class="QVBoxLayout" name="processed_img_v_lay">
        <item>
         <spacer name="verticalSpacer_4">
          <property name="sizeHint" stdset="0">
           <size>
            <width>20</width>
            <height>40</height>
           </size>
          </property>
         </spacer>
        </item>
        <item>
         <widget class="QLabel" name="processed_name_lbl">
          <property name="text">
           <string>面部特征</string>
          </property>
          <property name="alignment">
           <set>Qt::AlignCenter</set>
          </property>
         </widget>
        </item>
        <item>
         <spacer name="verticalSpacer_5">
```

```
          <property name="sizeHint" stdset="0">
           <size>
            <width>20</width>
            <height>40</height>
           </size>
          </property>
         </spacer>
        </item>
        <item>
         <widget class="QLabel" name="processed_frame_lbl">
          <property name="minimumSize">
           <size>
            <width>500</width>
            <height>300</height>
           </size>
          </property>
          <property name="baseSize">
           <size>
            <width>0</width>
            <height>0</height>
           </size>
          </property>
          <property name="text">
           <string/>
          </property>
         </widget>
        </item>
        <item>
         <widget class="QComboBox" name="selecamera2"/>
        </item>
        <item>
         <spacer name="verticalSpacer_6">
          <property name="sizeHint" stdset="0">
           <size>
            <width>20</width>
            <height>40</height>
           </size>
          </property>
         </spacer>
        </item>
       </layout>
      </item>
     </layout>
    </item>
    <item>
     <layout class="QHBoxLayout" name="horizontalLayout">
      <item>
       <layout class="QVBoxLayout" name="verticalLayout_2">
        <item>
         <widget class="QLabel" name="label_3">
          <property name="text">
           <string>标记图像</string>
          </property>
         </widget>
        </item>
        <item>
         <widget class="QLabel" name="eye_lbl">
          <property name="minimumSize">
           <size>
            <width>500</width>
            <height>300</height>
           </size>
          </property>
          <property name="text">
           <string/>
          </property>
         </widget>
        </item>
        <item>
         <widget class="QComboBox" name="comboBox_2"/>
        </item>
       </layout>
      </item>
      <item>
       <layout class="QVBoxLayout" name="verticalLayout">
        <item>
         <widget class="QLabel" name="label_2">
          <property name="text">
```

```
           <string>情绪识别</string>
          </property>
         </widget>
        </item>
        <item>
         <widget class="QLabel" name="voice_lbl">
          <property name="minimumSize">
           <size>
            <width>500</width>
            <height>300</height>
           </size>
          </property>
          <property name="text">
           <string/>
          </property>
         </widget>
        </item>
        <item>
         <widget class="QComboBox" name="comboBox"/>
        </item>
       </layout>
      </item>
     </layout>
    </item>
    <item>
     <layout class="QHBoxLayout" name="h_btn_lay">
      <item>
       <widget class="QPushButton" name="initButton">
        <property name="text">
         <string>检测视频设备</string>
        </property>
       </widget>
      </item>
      <item>
       <widget class="QPushButton" name="pushButton">
        <property name="text">
         <string>停用视频设备</string>
        </property>
       </widget>
      </item>
      <item>
       <spacer name="horizontalSpacer_2">
        <property name="sizeHint" stdset="0">
         <size>
          <width>40</width>
          <height>20</height>
         </size>
        </property>
       </spacer>
      </item>
      <item>
       <widget class="QLabel" name="label">
        <property name="text">
         <string>样本编号：</string>
        </property>
       </widget>
      </item>
      <item>
       <widget class="QLineEdit" name="lineEdit"/>
      </item>
      <item>
       <widget class="QPushButton" name="startButton">
        <property name="text">
         <string>开始提取</string>
        </property>
       </widget>
      </item>
      <item>
       <widget class="QPushButton" name="endButton">
        <property name="text">
         <string>结束提取</string>
        </property>
       </widget>
      </item>
      <item>
       <widget class="QPushButton" name="shotButton">
        <property name="text">
         <string>视频截图</string>
        </property>
       </widget>
      </item>
     </layout>
    </item>
    <item>
     <layout class="QVBoxLayout" name="v_filters_lay"/>
    </item>
   </layout>
  </widget>
  <widget class="QMenuBar" name="menubar">
   <property name="geometry">
    <rect>
     <x>0</x>
     <y>0</y>
     <width>1048</width>
     <height>37</height>
    </rect>
   </property>
   <widget class="QMenu" name="menuHelp">
    <property name="title">
     <string>帮助</string>
    </property>
    <addaction name="actionmanual"/>
    <addaction name="actionAbout"/>
   </widget>
   <widget class="QMenu" name="menu">
    <property name="title">
     <string>文件</string>
    </property>
   </widget>
   <widget class="QMenu" name="menu_2">
    <property name="title">
     <string>设置</string>
    </property>
    <addaction name="action12"/>
    <addaction name="actionse"/>
   </widget>
   <addaction name="menu"/>
```

```
 <addaction name="menu_2"/>
 <addaction name="menuHelp"/>
</widget>
<widget class="QStatusBar" name="statusbar">
 <property name="minimumSize">
  <size>
   <width>500</width>
   <height>30</height>
  </size>
 </property>
</widget>
<action name="actionOpen_image">
 <property name="text">
  <string>Open image</string>
 </property>
</action>
<action name="actionSave_original_image">
 <property name="text">
  <string>Save original image</string>
 </property>
</action>
<action name="actionSave_processed_image">
 <property name="text">
  <string>Save processed image</string>
 </property>
</action>
<action name="actionExit">
 <property name="text">
  <string>Exit</string>
 </property>
</action>
<action name="actionLicense">
 <property name="text">
  <string>License</string>
 </property>
</action>
<action name="actionAbout">
 <property name="text">
  <string>关于</string>
```

```
    </property>
   </action>
   <action name="actionmanual">
    <property name="text">
     <string>操作说明</string>
    </property>
   </action>
   <action name="action12">
    <property name="text">
     <string>设置视频与音频设备</string>
    </property>
   </action>
   <action name="actionse">
    <property name="text">
     <string>设置文件保存路径</string>
    </property>
   </action>
  </widget>
  <resources/>
  <connections/>
 </ui>
```

1.7 designer/test.py

```python
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'test.ui'
#
# Created by: PyQt5 UI code generator 5.9.2
#
# WARNING! All changes made in this file will be lost!

from PyQt5 import QtCore, QtGui, QtWidgets

class Ui_Form(object):
    def setupUi(self, Form):
        Form.setObjectName("Form")
        Form.resize(400, 300)
        self.tabWidget = QtWidgets.QTabWidget(Form)
        self.tabWidget.setGeometry(QtCore.QRect(10, 10, 371, 261))
        self.tabWidget.setObjectName("tabWidget")
        self.tab = QtWidgets.QWidget()
        self.tab.setObjectName("tab")
        self.splitter = QtWidgets.QSplitter(self.tab)
        self.splitter.setGeometry(QtCore.QRect(0, 10, 361, 221))
        self.splitter.setOrientation(QtCore.Qt.Vertical)
        self.splitter.setObjectName("splitter")
        self.widget = QtWidgets.QWidget(self.splitter)
        self.widget.setObjectName("widget")
        self.horizontalLayout = QtWidgets.QHBoxLayout(self.widget)
        self.horizontalLayout.setSizeConstraint(QtWidgets.QLayout.SetMinAndMaxSize)
        self.horizontalLayout.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout.setObjectName("horizontalLayout")
        spacerItem = QtWidgets.QSpacerItem(90, 20, QtWidgets.QSizePolicy.Maximum, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout.addItem(spacerItem)
        self.tab1_combobox = QtWidgets.QComboBox(self.widget)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.MinimumExpanding, QtWidgets.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.tab1_combobox.sizePolicy().hasHeightForWidth())
        self.tab1_combobox.setSizePolicy(sizePolicy)
        self.tab1_combobox.setObjectName("tab1_combobox")
        self.horizontalLayout.addWidget(self.tab1_combobox)
        spacerItem1 = QtWidgets.QSpacerItem(90, 20, QtWidgets.QSizePolicy.Maximum, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout.addItem(spacerItem1)
        self.tab1_label = QtWidgets.QLabel(self.splitter)
        self.tab1_label.setObjectName("tab1_label")
        self.tabWidget.addTab(self.tab, "")
        self.tab_2 = QtWidgets.QWidget()
        self.tab_2.setObjectName("tab_2")
        self.splitter_2 = QtWidgets.QSplitter(self.tab_2)
        self.splitter_2.setGeometry(QtCore.QRect(0, 10, 361,
        ↪  221))
        self.splitter_2.setOrientation(QtCore.Qt.Vertical)
        self.splitter_2.setObjectName("splitter_2")
        self.widget1 = QtWidgets.QWidget(self.splitter_2)
        self.widget1.setObjectName("widget1")
        self.horizontalLayout_3 =
        ↪  QtWidgets.QHBoxLayout(self.widget1)
        self.horizontalLayout_3.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout_3.setObjectName("horizontalLayo⌟
        ↪  ut_3")
        spacerItem2 = QtWidgets.QSpacerItem(90, 20,
        ↪  QtWidgets.QSizePolicy.Maximum,
        ↪  QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_3.addItem(spacerItem2)
        self.tab2_combobox_2 =
        ↪  QtWidgets.QComboBox(self.widget1)
        self.tab2_combobox_2.setObjectName("tab2_combobox_2")
        self.horizontalLayout_3.addWidget(self.tab2_combobox_⌟
        ↪  2)
        spacerItem3 = QtWidgets.QSpacerItem(90, 20,
        ↪  QtWidgets.QSizePolicy.Maximum,
        ↪  QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_3.addItem(spacerItem3)
        self.tab2_label_2 = QtWidgets.QLabel(self.splitter_2)
        self.tab2_label_2.setObjectName("tab2_label_2")
        self.tabWidget.addTab(self.tab_2, "")
        self.tab_3 = QtWidgets.QWidget()
        self.tab_3.setObjectName("tab_3")
        self.splitter_3 = QtWidgets.QSplitter(self.tab_3)
        self.splitter_3.setGeometry(QtCore.QRect(0, 10, 361,
        ↪  221))
        self.splitter_3.setOrientation(QtCore.Qt.Vertical)
        self.splitter_3.setObjectName("splitter_3")
        self.widget2 = QtWidgets.QWidget(self.splitter_3)
        self.widget2.setObjectName("widget2")
        self.horizontalLayout_4 = QtWidgets.QHBoxLayout(self.widget2)
        self.horizontalLayout_4.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout_4.setObjectName("horizontalLayout_4")
        spacerItem4 = QtWidgets.QSpacerItem(90, 20, QtWidgets.QSizePolicy.Maximum, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_4.addItem(spacerItem4)
        self.tab3_combobox_3 = QtWidgets.QComboBox(self.widget2)
        self.tab3_combobox_3.setObjectName("tab3_combobox_3")
        self.horizontalLayout_4.addWidget(self.tab3_combobox_3)
        spacerItem5 = QtWidgets.QSpacerItem(90, 20, QtWidgets.QSizePolicy.Maximum, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_4.addItem(spacerItem5)
        self.tab3_label_3 = QtWidgets.QLabel(self.splitter_3)
        self.tab3_label_3.setObjectName("tab3_label_3")
        self.tabWidget.addTab(self.tab_3, "")
        self.tab_4 = QtWidgets.QWidget()
        self.tab_4.setObjectName("tab_4")
        self.progressBar = QtWidgets.QProgressBar(self.tab_4)
        self.progressBar.setGeometry(QtCore.QRect(90, 120, 191, 21))
        self.progressBar.setProperty("value", 24)
        self.progressBar.setObjectName("progressBar")
        self.widget3 = QtWidgets.QWidget(self.tab_4)
        self.widget3.setGeometry(QtCore.QRect(0, 10, 361, 22))
        self.widget3.setObjectName("widget3")
        self.horizontalLayout_5 = QtWidgets.QHBoxLayout(self.widget3)
        self.horizontalLayout_5.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout_5.setObjectName("horizontalLayout_5")
```

```python
        spacerItem6 = QtWidgets.QSpacerItem(90, 20, QtWidgets.QSizePolicy.Maximum, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_5.addItem(spacerItem6)
        self.tab3_combobox_4 = QtWidgets.QComboBox(self.widget3)
        self.tab3_combobox_4.setObjectName("tab3_combobox_4")
        self.horizontalLayout_5.addWidget(self.tab3_combobox_4)
        spacerItem7 = QtWidgets.QSpacerItem(90, 20, QtWidgets.QSizePolicy.Maximum, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_5.addItem(spacerItem7)
        self.tabWidget.addTab(self.tab_4, "")
        self.OK = QtWidgets.QPushButton(Form)
        self.OK.setGeometry(QtCore.QRect(254, 270, 111, 23))
        self.OK.setObjectName("OK")

        self.retranslateUi(Form)
        self.tabWidget.setCurrentIndex(0)
        QtCore.QMetaObject.connectSlotsByName(Form)

    def retranslateUi(self, Form):
        _translate = QtCore.QCoreApplication.translate
        Form.setWindowTitle(_translate("Form", "Form"))
        self.tab1_label.setText(_translate("Form", "TextLabel"))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab), _translate("Form", "Tab 1"))
        self.tab2_label_2.setText(_translate("Form", "TextLabel"))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab_2), _translate("Form", "Tab 2"))
        self.tab3_label_3.setText(_translate("Form", "TextLabel"))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab_3), _translate("Form", "Tab 3"))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab_4), _translate("Form", "Tab 4"))
        self.OK.setText(_translate("Form", "PushButton"))

```

1.8 main.py

```
if __name__ == '__main__':
    import sys,platform
    from PyQt5.QtWidgets import QApplication
    #from CvPyGui import Main
    from CvPyGui import main as Main
    if platform.system() == "Windows":
        import ctypes
        ctypes.windll.shell32.SetCurrentProcessExplicitAppUserModelID("myappid")
    app = QApplication(sys.argv)
    window = Main.MyApp()
    window.show()
    global update1
    update1 = 0
    global update2
    update2 = 0
    sys.exit(app.exec_())
```

1.9 CvPyGui/Child_File_Setting.py

```
import json
import time
from PyQt5.QtWidgets import QWidget
from PyQt5 import QtWidgets
from CvPyGui.ui import child_file
Ui_ChildWindow_2 = child_file.Ui_Form
```

```
class child_file_setting(QWidget,Ui_ChildWindow_2):
    # 功能：选择文件路径。。显示当前的文件路径。。
    # 选择音视频保存的格式  是名字的格式还是文件格式?
    def __init__(self):
        super().__init__()
        self.setupUi(self)
        self.InitChildWindwo()
    def InitChildWindwo(self):
        # 初始化,,, 读取 json 文件，显示当前的保存路径
        with open("config/db.json","r",encoding='UTF-8') as dbfile_r:
            file_save_path = json.load(dbfile_r)
        self.file_path_label.setText(file_save_path["save_path"])
        self.CreateButtons()
        if file_save_path["video_format"] == ".avi":
            self.radioButton_avi.setChecked(True)
        else:
            self.radioButton_mp4.setChecked(True)
        if file_save_path["voice_format"] == ".wav":
            self.radioButton_wav.setChecked(True)
        else:
            self.radioButton_mp3.setChecked(True)
    def msg(self):
        m = QtWidgets.QFileDialog.getExistingDirectory(None, "选取文件夹", "record")  # 起始路径
        with open("config/db.json","r",encoding='UTF-8') as file_r:
            savepath = json.load(file_r)
        # 原路径保存至 former_save_path
        former_save_path = savepath["save_path"]
        print(m)# 打印刚刚获取的当前路径
        #如果获取的路径为空，就不改变路径，把之前的路径赋给新的路径
        if m == "":
            m = former_save_path
        print(m)
        savepath["save_path"] = m
        # 将更改后（或获取空时不更改）的路径写入 json 文件
        with open("config/db.json","w",encoding='UTF-8') as dbfile:
            json.dump(savepath,dbfile)
        time.sleep(0.5)
        self.update_file_save_path()
    def update_file_save_path(self):
        with open("config/db.json","r",encoding='UTF-8') as dbfile_r:
            file_save_path = json.load(dbfile_r)
        self.file_path_label.setText(str(file_save_path['save_path']))
    def ok_close(self):
        self.close()
    def mp3_button(self):
        if self.radioButton_mp3.isChecked() == True:
            self.voiceformat = ".mp3"
        else:
            self.voiceformat = ".wav"
        with open("config/db.json","r",encoding='UTF-8') as dbfile:
            json_file = json.load(dbfile)
        json_file["voice_format"] = self.voiceformat
        with open("config/db.json","w",encoding='UTF-8') as dbfile:
            json.dump(json_file,dbfile)
        print(self.voiceformat)
    def video_radio_button(self):
        if self.radioButton_mp4.isChecked() == True:
            self.videoformat = ".mp4v"
        else:
            self.videoformat = ".avi"
        with open("config/db.json","r",encoding='UTF-8') as dbfile:
            json_file = json.load(dbfile)
        json_file["video_format"] = self.videoformat
        with open("config/db.json","w",encoding='UTF-8') as dbfile:
            json.dump(json_file,dbfile)
        print(self.videoformat)
```

```python
    def CreateButtons(self):
        self.choice_file_path_button.clicked.connect(self.msg)
        self.file_ok.clicked.connect(self.ok_close)
        self.radioButton_mp3.clicked.connect(self.mp3_button)
        self.radioButton_wav.clicked.connect(self.mp3_button)
        self.radioButton_mp4.clicked.connect(self.video_radio_button)
        self.radioButton_avi.clicked.connect(self.video_radio_button)
```

1.10 CvPyGui/FilterCvQtContainer.py

```python
from PyQt5.QtCore import Qt
from PyQt5.QtWidgets import (QWidget, QLabel, QHBoxLayout,
                             QPushButton, QSlider)
import cv2
import numpy as np
class Filter(QWidget):
    """Common base class for all filters"""
    defaultK = 3
    filterCount = 0
    def __init__(self, name, minValue, maxValue, init, num_of_k, parent=None):
        super().__init__()
        self.filter_number = Filter.filterCount
        self.name = name
        self.num_of_k = num_of_k
        self.k = [init]
        # Increase the number of filters created
        Filter.filterCount += 1
        # Set maximum height
        self.setMaximumHeight(65)
        # Variable for the slider/label layout
        self.lay = QHBoxLayout(self)
        # Variable for the constant of the OpenCV filter
        self.k[0] = self.defaultK
        # Label for the slider
        self.k_lbl = [QLabel(str(self.k[0]))]
        # Name for the slider
        self.name_lbl = QLabel(self.name + ': ')
        # Set default parameters
        self.setParameters(minValue, maxValue)
        # Create delete button
        self.delete_filter_btn = QPushButton('X')
        self.delete_filter_btn.clicked.connect(self.deleteFilter)
        # Adds the slider and it's label to the layout
        self.createLayout()
        # Function sending the slider signal to the processing function
        self.thresh_sld.valueChanged.connect(self.changeValue)
    def setParameters(self, minValue, maxValue):
        # Creates the slider for the OpenCV filter, with min, max, default and
        # step values
        self.thresh_sld = QSlider(Qt.Horizontal, self)
        self.thresh_sld.setFocusPolicy(Qt.NoFocus)
        self.thresh_sld.setMinimum(minValue)
        self.thresh_sld.setMaximum(maxValue)
        self.thresh_sld.setValue(self.k[0])
        self.thresh_sld.setSingleStep(2)
    def createLayout(self):
        # Adds the slider and its label to the bottom of the main layout
        self.lay.addWidget(self.name_lbl)
        self.lay.addWidget(self.k_lbl[0])
        self.lay.addWidget(self.thresh_sld)
        self.lay.addWidget(self.delete_filter_btn)
    def changeValue(self, value):
        # Function for setting the value of k1
        if value % 2 == 1:
            self.k[0] = value
        else:
            self.k[0] = value + 1
        self.thresh_sld.setValue(self.k[0])
        self.k_lbl[0].setText(str(self.k[0]))
        self.parent().parent().updateImages()
    def resetValue(self):
        # Resets the K value to it's default
        self.changeValue(self.defaultK)
    def deleteFilter(self):
        self.parent().parent().deleteFilter(self.filter_numbe⌋
        ↪   r)
    def process(self, cv_before, name):
        k = self.k[0]
        kernel = np.ones((k, k), np.uint8)
        if name == 'Invert':
            cv_before = cv2.cvtColor(cv_before,
            ↪   cv2.COLOR_RGB2GRAY)
            cv_after = cv2.bitwise_not(cv_before)
        elif name == 'Histogram Equalization':
            cv_before = cv2.cvtColor(cv_before,
            ↪   cv2.COLOR_RGB2GRAY)
            clahe = cv2.createCLAHE(clipLimit=2.0,
            ↪   tileGridSize=(8, 8))
            cv_after = clahe.apply(cv_before)
        elif name == 'Threshold':
            cv_before = cv2.cvtColor(cv_before,
            ↪   cv2.COLOR_RGB2GRAY)
            ret, cv_after = cv2.threshold(
                cv_before, k, 255, cv2.THRESH_BINARY)
        elif name == 'Gaussian Threshold':
            cv_before = cv2.cvtColor(cv_before,
            ↪   cv2.COLOR_RGB2GRAY)
            cv_after = cv2.adaptiveThreshold(cv_before, 255,
            ↪   cv2.ADAPTIVE_THRESH_GAUSSIAN_C,
                                             cv2.THRESH_BINAR⌋
                                             ↪   Y, k,
                                             ↪   2)
        elif name == 'HSV':
            cv_before = cv2.cvtColor(cv_before,
            ↪   cv2.COLOR_RGB2HSV)
            lower_color = np.array([k - 35, 0, 0])
            upper_color = np.array([k + 35, 255, 255])
            cv_after = cv2.inRange(cv_before, lower_color,
            ↪   upper_color)
        elif name == 'LAB':
            cv_before = cv2.cvtColor(cv_before,
            ↪   cv2.COLOR_RGB2LAB)
            L, a, b = cv2.split(cv_before)
            ret, cv_after = cv2.threshold(L, k, 255,
            ↪   cv2.THRESH_BINARY)
        elif name == 'Erosion':
            cv_before = cv2.cvtColor(cv_before,
            ↪   cv2.COLOR_RGB2GRAY)
            cv_after = cv2.erode(cv_before, kernel,
            ↪   iterations=1)
        elif name == 'Dilation':
            cv_before = cv2.cvtColor(cv_before,
            ↪   cv2.COLOR_RGB2GRAY)
            cv_after = cv2.dilate(cv_before, kernel,
            ↪   iterations=1)
        elif name == 'Opening':
            cv_before = cv2.cvtColor(cv_before,
            ↪   cv2.COLOR_RGB2GRAY)
            cv_after = cv2.morphologyEx(
                cv_before, cv2.MORPH_OPEN, kernel)
        elif name == 'Closing':
            cv_before = cv2.cvtColor(cv_before,
            ↪   cv2.COLOR_RGB2GRAY)
            cv_after = cv2.morphologyEx(
                cv_before, cv2.MORPH_CLOSE, kernel)
        elif name == 'Top Hat':
            cv_before = cv2.cvtColor(cv_before,
            ↪   cv2.COLOR_RGB2GRAY)
            cv_after = cv2.morphologyEx(
                cv_before, cv2.MORPH_TOPHAT, kernel)
        elif name == 'Black Hat':
            cv_before = cv2.cvtColor(cv_before,
            ↪   cv2.COLOR_RGB2GRAY)
            cv_after = cv2.morphologyEx(
                cv_before, cv2.MORPH_BLACKHAT, kernel)
        elif name == 'Canny':
            cv_before = cv2.cvtColor(cv_before,
            ↪   cv2.COLOR_RGB2GRAY)
            cv_after = cv2.Canny(cv_before, 100, k)
        elif name == 'Laplacian':
            cv_before = cv2.cvtColor(cv_before,
            ↪   cv2.COLOR_RGB2GRAY)
            cv_after = cv2.Laplacian(cv_before, cv2.CV_64F)
            cv_after = np.absolute(cv_after)
            cv_after = np.uint8(cv_after)
        return cv_after
```

1.11 CvPyGui/ImageCvQtContainer.py

```
from PyQt5.QtGui import *
from PyQt5.QtWidgets import *
from PyQt5.QtCore import Qt
class Image(QWidget):
    """Common base for the images"""
    def __init__(self, name, label):
        super().__init__()
        self.frame_lbl = label
        print (self.frame_lbl.size())
    def updateImage(self, opencv_rgb_image):
        self.cv_img_rgb = opencv_rgb_image
        height, width, channel = self.cv_img_rgb.shape
        bytesPerLine = 3 * width
        self.q_image = QImage(self.cv_img_rgb.data,
        ↪   width,height, bytesPerLine, QImage.Format_RGB888)
        self.frame_lbl.setPixmap(QPixmap.fromImage(self.q_ima⌋
        ↪   ge).scaled(self.frame_lbl.size(),
        ↪   aspectRatioMode=Qt.KeepAspectRatio))
    def saveImage(self):
        filter = "Images (*.png *.jpg)"
        image_path, _ = QFileDialog.getSaveFileName(self,
        ↪   filter=filter)
        cv_img_bgr = cv2.cvtColor(
            self.cv_img_rgb, cv2.COLOR_RGB2BGR)
        cv2.imwrite(image_path, cv_img_bgr)
```

1.12 CvPyGui/__init__.py

```python

```

1.13 CvPyGui/child_CameraVoice_setting.py

```python
import json
import time
import numpy as np
import pyaudio
from PyQt5.QtCore import (QTimer, pyqtSignal,QThread)
from CvPyGui import ImageCvQtContainer
import pygame
import cv2
import threading
from PyQt5.QtWidgets import QWidget
from CvPyGui.ui import child_test
Ui_ChildWindow = child_test.Ui_Form
hd_width = 1920
hd_higth= 1080
pleace_choice_camera = '请选择摄像头'
no_camera_ = '不选择摄像头'
pleace_choice_voice = '请选择麦克风'
no_voice = '不选择麦克风'
img_no_camera = cv2.imread('config/image/no_camera.jpg', cv2.IMREAD_COLOR)
img_no_voice = cv2.imread('config/image/no_voice.jpg', cv2.IMREAD_COLOR)
img_voice = cv2.imread('config/image/voice.jpg',cv2.IMREAD_COLOR)
tab1_judge = False # 判断下拉列表选择的内容是否是摄像头
tab2_judge = False
tab3_judge = False
tab4_judge = False
camera_judge = {}
thread_judge = {}
frame = {}
frame2 = {}
```

```
frame3 = {}
cap = {}
cap2 = {}
cap3 = {}
tab1_text = ''
tab2_text = ''
tab3_text = ''
tab4_text = ''
class UpdateVolume(QThread):
    update_data = pyqtSignal(str)
    def __init__(self):
        super().__init__()
        self.vo_judge = True
        self.voice_index_thread = 0
    def run(self):
        CHUNK = 1024
        FORMAT = pyaudio.paInt16
        CHANNELS = 1
        RATE = 44100
        INTERVAL = 5
        pa = pyaudio.PyAudio()
        stream = pa.open(format=FORMAT,
                         channels=CHANNELS,
                         rate=RATE,
                         input=True,
                         frames_per_buffer=CHUNK,
                         input_device_index=self.voice_index_thread)
        buffer = []
        while self.vo_judge:
            for i in range(int(INTERVAL * RATE / CHUNK)):  # STREAN INTERVAL
                if self.vo_judge == False:
                    break
                data = np.fromstring(stream.read(CHUNK), dtype=np.int16)
                self.un = int(np.amax(data))
                self.un = int(pow(self.un,0.5))
                if self.un >= 100:
                    self.un = 99
                self.update_data.emit(str(self.un))
        print('子线程结束')
#1:下拉列表选择后lable标签显示对应的视频。  获取下拉列表的内容
↪  根据内容确定显示的摄像头。
class Child_Window(QWidget,Ui_ChildWindow):
    def __init__(self):
        super().__init__()
        Ui_ChildWindow.__init__(self)
        self.setupUi(self)
        self.InitChildWindow()
        self.timer = QTimer()
        self.child_wind_judge = True
    def InitChildWindow(self):# 初始化子窗口
        pygame.init()
        pygame.camera.init()
        global cameralist_child
        cameralist_child = pygame.camera.list_cameras()
        self.update_tab1_image =
        ↪  ImageCvQtContainer.Image('tab1_label_camera',
        ↪  self.tab1_label)
        self.update_tab2_image =
        ↪  ImageCvQtContainer.Image('tab2_label_camera',
        ↪  self.tab1_label_3)
        self.update_tab3_image =
        ↪  ImageCvQtContainer.Image('tab3_label_camera',
        ↪  self.tab1_label_4)
        self.update_tab1_image.updateImage(img_no_camera)
        self.update_tab2_image.updateImage(img_no_camera)
        self.update_tab3_image.updateImage(img_no_camera)
        self.tab1_combobox.addItem(pleace_choice_camera)
        self.tab1_combobox_3.addItem(pleace_choice_camera)
        self.tab1_combobox_4.addItem(pleace_choice_camera)
        self.tab1_combobox_7.addItem(pleace_choice_voice)
        self.tab1_combobox.addItem(no_camera_)
        self.tab1_combobox_3.addItem(no_camera_)
        self.tab1_combobox_4.addItem(no_camera_)
        self.tab1_combobox_7.addItem(no_voice)
        self.tab1_combobox.addItems(cameralist_child)
        self.tab1_combobox_3.addItems(cameralist_child)
        self.tab1_combobox_4.addItems(cameralist_child)
        self.CreateButtons()
        with open("config/db.json","r",encoding='UTF-8') as
        ↪ dbfile_r:
            camera_voice_name = json.load(dbfile_r)
        self.tab1_text = camera_voice_name["hd_camera_name"]
        self.tab2_text = camera_voice_name["face_camera_name"]
        self.tab3_text = camera_voice_name["eye_camera_name"]
        self.tab4_text = camera_voice_name["hk_voice_name"]
        if len(cameralist_child) >= 1:
            for i in range(len(cameralist_child)):
                camera_judge[i] = 'close'
        else:
            print('no camera')
        p = pyaudio.PyAudio()
        info = p.get_host_api_info_by_index(0)
        numdevices = info.get('deviceCount')
        global voice_list
        voice_list = []
        for i in range(0, numdevices):
            if (p.get_device_info_by_host_api_device_index(0,
            ↪ i).get('maxInputChannels')) > 0:
                voice_list.append(p.get_device_info_by_host_a⌟
                ↪ pi_device_index(0,
                ↪ i).get('name'))
        self.tab1_combobox_7.addItems(voice_list)
        self.child_wind_judge = True
        if self.tab1_text in cameralist_child:
            self.tab1_combobox.setCurrentIndex(cameralist_chi⌟
            ↪ ld.index(self.tab1_text) +
            ↪ 2)
        if self.tab2_text in cameralist_child:
            self.tab1_combobox_3.setCurrentIndex(cameralist_c⌟
            ↪ hild.index(self.tab2_text) +
            ↪ 2)
        if self.tab3_text in cameralist_child:
            self.tab1_combobox_4.setCurrentIndex(cameralist_child.index(self.tab3_text) + 2)
        if self.tab4_text in voice_list:
            self.tab1_combobox_7.setCurrentIndex(voice_list.index(self.tab4_text) + 2)
    def tab1_combobox_setting(self):  #tab1中的下拉列表选择摄像头
        # 获取当前下拉列表选中的文本
        tab1_camera_text = self.tab1_combobox.currentText()
        self.tab1_text = tab1_camera_text
        #尝试查询当前文本所在摄像头列表中第几项，查询不到则显示没有摄像头的图片。
        try:
            tab1_camera_index = cameralist_child.index(tab1_camera_text)
            tab1_judge = True
        except:
            self.update_tab1_image.updateImage(img_no_camera)
            tab1_judge = False
        # 获取摄像头的名字后显示画面
        # 启动一个线程，显示摄像头捕获的画面
        #如果查询到文本所在该列表中，并且该摄像头没有打开，就启动线程显示该摄像头的画面
        if tab1_judge:
            if camera_judge[tab1_camera_index] == 'close':
                camera_judge[tab1_camera_index] = 'open'
                threading._start_new_thread(self.open_camera, (tab1_camera_index,))
        else:
           self.update_tab1_image.updateImage(img_no_camera)
    def tab2_combobox_setting(self):
        tab2_camera_text = self.tab1_combobox_3.currentText()
        self.tab2_text = tab2_camera_text
        try:
            tab2_camera_index = cameralist_child.index(tab2_camera_text)
            tab2_judge = True
```

```python
        except:
            self.update_tab2_image.updateImage(img_no_camera)
            tab2_judge = False
        # 获取摄像头的名字后显示画面
        # 启动一个线程，显示摄像头捕获的画面
        if tab2_judge:
            # 如果选择的摄像头没有打开
            if camera_judge[tab2_camera_index] == 'close':
                camera_judge[tab2_camera_index] = 'open'
                threading._start_new_thread(self.open_camera_2, (tab2_camera_index,))
        else:
            self.update_tab2_image.updateImage(img_no_camera)
    def open_camera(self,camera_index):
        #传入摄像头的序号，判断摄像头是否开启的代码应该在调用该函数的函数中。
        #根据摄像头的序号，打开摄像头，然后把摄像头的画面传入其他函数?
        #只有三个要显示的标签，判断要在那个标签上显示，然后更新图片。
        # cap[camera_index] = cv2.VideoCapture(camera_index,cv2.CAP_DSHOW)
        cap[camera_index] = cv2.VideoCapture(camera_index)
        print(cap[camera_index].get(3))
        camera_judge[camera_index] = 'open'
        cap[camera_index].set(3,hd_width)
        cap[camera_index].set(4,hd_width)
        cap[camera_index].set(5,30)
        while self.child_wind_judge:
            self.ret, frame[camera_index] = cap[camera_index].read()
            if self.ret:
                #判断需要在哪几个标签上显示  判断方式：获取三个下拉列表当前显示的值，确定该值在摄像头列表中所占的序号，序号与camera_index相同则更新图片。
                frame[camera_index] = cv2.cvtColor(frame[camera_index], cv2.COLOR_BGR2RGB)
                if (self.tab1_combobox.currentText() != pleace_choice_camera) and (self.tab1_combobox.currentText() != no_camera_):
                    if camera_index == cameralist_child.index(self.tab1_combobox.currentText()):
                        self.update_tab1_image.updateImage(frame[camera_index])
                    else:
                        pass
                else:
                    pass
                if (self.tab1_combobox_3.currentText() != pleace_choice_camera) and (self.tab1_combobox_3.currentText() != no_camera_):
                    if camera_index == cameralist_child.index(self.tab1_combobox_3.currentText()):
                        self.update_tab2_image.updateImage(frame[camera_index])
                    else:
                        pass
                else:
                    pass
                if (self.tab1_combobox_4.currentText() != pleace_choice_camera) and (self.tab1_combobox_4.currentText() != no_camera_):
                    if camera_index == cameralist_child.index(self.tab1_combobox_4.currentText()):
                        self.update_tab3_image.updateImage(frame[camera_index])
                    else:
                        pass
                else:
                    pass
                if (cameralist_child[camera_index] != self.tab1_combobox.currentText()) & (cameralist_child[camera_index] != self.tab1_combobox_3.currentText()) & (cameralist_child[camera_index] != self.tab1_combobox_4.currentText()):
                    camera_judge[camera_index] = 'close'
                    break
            if self.child_wind_judge == False: #子窗口如果关闭就停止循环
                break
    def open_camera_2(self,camera_index):
        # cap2[camera_index] = cv2.VideoCapture(camera_index,cv2.CAP_DSHOW)
        cap2[camera_index] = cv2.VideoCapture(camera_index)
        camera_judge[camera_index] = 'open'
        cap2[camera_index].set(3,hd_width)
        cap2[camera_index].set(4,hd_width)
        cap2[camera_index].set(5,30.0)
        while self.child_wind_judge:
            self.ret_2,frame2[camera_index] = cap2[camera_index].read()
            if self.ret_2:
                frame2[camera_index] = cv2.cvtColor(frame2[camera_index], cv2.COLOR_BGR2RGB)
                if (self.tab1_combobox.currentText() != pleace_choice_camera) and (self.tab1_combobox.currentText() != no_camera_):
                    if camera_index == cameralist_child.index(self.tab1_combobox.currentText()):
                        self.update_tab1_image.updateImage(frame2[camera_index])
                    else:
                        pass
                else:
                    pass
                if (self.tab1_combobox_3.currentText() != pleace_choice_camera) and (self.tab1_combobox_3.currentText() != no_camera_):
                    if camera_index == cameralist_child.index(self.tab1_combobox_3.currentText()):
                        self.update_tab2_image.updateImage(frame2[camera_index])
                    else:
                        pass
                else:
                    pass
                if (self.tab1_combobox_4.currentText() != pleace_choice_camera) and (self.tab1_combobox_4.currentText() != no_camera_):
                    if camera_index == cameralist_child.index(self.tab1_combobox_4.currentText()):
                        self.update_tab3_image.updateImage(frame2[camera_index])
                    else:
                        pass
                else:
                    pass
                if (cameralist_child[camera_index] != self.tab1_combobox.currentText()) & (cameralist_child[camera_index] != self.tab1_combobox_3.currentText()) & (cameralist_child[camera_index] != self.tab1_combobox_4.currentText()):
                    camera_judge[camera_index] = 'close'
                    break
                else:
                    pass
                if self.child_wind_judge == False: #子窗口如果关闭就停止循环
                    break
    def open_camera_3(self,camera_index):
        # cap3[camera_index] =
        ↪  cv2.VideoCapture(camera_index,cv2.CAP_DSHOW)
        cap3[camera_index] = cv2.VideoCapture(camera_index)
        camera_judge[camera_index] = 'open'
        cap3[camera_index].set(3,hd_width)
        cap3[camera_index].set(4,hd_width)
        cap3[camera_index].set(5,30.0)
        while self.child_wind_judge:
            self.ret_3, frame3[camera_index] =
            ↪  cap3[camera_index].read()
            if self.ret_3:
                frame3[camera_index] =
                ↪  cv2.cvtColor(frame3[camera_index],
                ↪  cv2.COLOR_BGR2RGB)
                if (self.tab1_combobox.currentText() !=
                ↪  pleace_choice_camera) and
                ↪  (self.tab1_combobox.currentText() !=
                ↪  no_camera_):
                    if camera_index == cameralist_child.index⌟
                    ↪  (self.tab1_combobox.currentText()):
                        self.update_tab1_image.updateImage(fr⌟
                        ↪  ame3[camera_index])
                    else:
                        pass
                else:
                    pass
                if (self.tab1_combobox_3.currentText() !=
                ↪  pleace_choice_camera) and
                ↪  (self.tab1_combobox_3.currentText() !=
                ↪  no_camera_):
                    if camera_index == cameralist_child.index⌟
                    ↪  (self.tab1_combobox_3.currentText()):
                        self.update_tab2_image.updateImage(fr⌟
                        ↪  ame3[camera_index])
                    else:
                        pass
                else:
                    pass
                if (self.tab1_combobox_4.currentText() != pleace_choice_camera) and (self.tab1_combobox_4.currentText() != no_camera_):
                    if camera_index == cameralist_child.index(self.tab1_combobox_4.currentText()):
                        self.update_tab3_image.updateImage(frame3[camera_index])
                    else:
                        pass
                else:
                    pass
                if (cameralist_child[camera_index] != self.tab1_combobox.currentText()) & (cameralist_child[camera_index] != self.tab1_combobox_3.currentText()) & (cameralist_child[camera_index] != self.tab1_combobox_4.currentText()):
                    camera_judge[camera_index] = 'close'
                    break
                else:
                    pass
                if self.child_wind_judge == False: #子窗口如果关闭就停止循环
                    break
    def tab3_combobox_setting(self):
        tab3_camera_text = self.tab1_combobox_4.currentText()
        self.tab3_text = tab3_camera_text
        try:
            tab3_camera_index = cameralist_child.index(tab3_camera_text)
            tab3_judge = True
        except:
            self.update_tab3_image.updateImage(img_no_camera)
            tab3_judge = False
        # 获取摄像头的名字后显示画面
        # 启动一个线程，显示摄像头捕获的画面
        if tab3_judge:
            if camera_judge[tab3_camera_index] == 'close':
                camera_judge[tab3_camera_index] = 'open'
                threading._start_new_thread(self.open_camera_⌟
                ↪  3,
                ↪  (tab3_camera_index,))
        else:
            self.update_tab3_image.updateImage(img_no_camera)
    def tab4_voice_setting(self):
        tab4_voice_text = self.tab1_combobox_7.currentText()
        self.tab4_text = tab4_voice_text
        try:
            voice_index = voice_list.index(self.tab4_text)
            tab4_judge = True
        except:
            tab4_judge =False
        try:
            self.sub_thread.vo_judge = False
        except:
            pass
        time.sleep(0.2)
        if tab4_judge:
            self.sub_thread = UpdateVolume()
            self.sub_thread.vo_judge = True
            self.sub_thread.voice_index_thread = voice_index
            self.sub_thread.update_data.connect(self.unnn)
            self.sub_thread.start()
        else:
            print('无声音设备')
        print('voice setting')
    def unnn(self,data):
        self.progressBar.setValue(int(data))
    def save_camera_json(self):
        # 用来关闭上面所有的线程，改写 json 配置文件。
        # tab1_text、tab2_text、tab3_text分别是行为、面部、
        ↪  眼部的摄像头名字，初始内容为读取json文件得到的。
        with open("config/db.json","r",encoding='UTF-8') as
        ↪  dbfile_r:
            camera_voice_name = json.load(dbfile_r)
        camera_voice_name["hd_camera_name"] = self.tab1_text
        camera_voice_name["face_camera_name"] = self.tab2_text
        camera_voice_name["eye_camera_name"] = self.tab3_text
        camera_voice_name["hk_voice_name"] = self.tab4_text
        with open("config/db.json","w",encoding='UTF-8') as dbfile:
            json.dump(camera_voice_name,dbfile)
        self.child_wind_judge = False
        try:
            self.sub_thread.vo_judge = False
            print('success')
        except:
            print('defalt')
        self.close()# 关闭该窗口
    def closeEvent(self, event):
        print('关闭窗口')
        self.child_wind_judge = False
        try:
            self.sub_thread.vo_judge = False
            self.child_wind_judge = False
        except:
            pass
#应该是改变摄像头选项时修改配置文件还是在点确认是关闭呢?点确认时关闭，先把选好的信息存起来，点确认时修改。
    def CreateButtons(self):
        # print("createbuttons")
        self.tab1_combobox.currentIndexChanged.connect(self.tab1_combobox_setting)
        self.tab1_combobox_3.currentIndexChanged.connect(self.tab2_combobox_setting)
        self.tab1_combobox_4.currentIndexChanged.connect(self.tab3_combobox_setting)
        self.tab1_combobox_7.currentIndexChanged.connect(self.tab4_voice_setting)
        self.OK.clicked.connect(self.save_camera_json)
```

1.14 CvPyGui/main.py

```
import wave
import cv2, os
```

```python
from PyQt5.QtCore import Qt,QTimer
from PyQt5.QtWidgets import (QMainWindow,QMessageBox,)
import threading
import time
from CvPyGui import ImageCvQtContainer
from CvPyGui import child_CameraVoice_setting
from CvPyGui import Child_File_Setting
from CvPyGui.ui import gui
from datetime import datetime
import pygame
import pygame.camera
import pyaudio

import imutils
from imutils import face_utils
import dlib
import time
import numpy as np
from keras.models import load_model
from keras.preprocessing.image import img_to_array

from PIL import Image, ImageDraw, ImageFont

from mediapipe import solutions
from mediapipe.framework.formats import landmark_pb2

# STEP 1: Import the necessary modules.
import mediapipe as mp
from mediapipe.tasks import python
from mediapipe.tasks.python import vision

# STEP 2: Create an PoseLandmarker object.
base_options = python.BaseOptions(model_asset_path='pose_landmarker.task')
options = vision.PoseLandmarkerOptions(
    base_options=base_options,
    output_segmentation_masks=True)
detector = vision.PoseLandmarker.create_from_options(options)
```

```python
Ui_MainWindow = gui.Ui_MainWindow
import json
img_no_camera = cv2.imread('config/image/no_camera.jpg', cv2.IMREAD_COLOR)
# img_no_voice = cv2.imread('config/image/no_voice.jpg', cv2.IMREAD_COLOR)
img_no_voice = cv2.imread('config/image/no_camera.jpg', cv2.IMREAD_COLOR)
img_voice=cv2.imread('config/image/voice.jpg',cv2.IMREAD_COLOR)


def preprocess_face(face):
    face = cv2.cvtColor(face, cv2.COLOR_BGR2GRAY)  # Convert to grayscale
    face = cv2.resize(face, (64, 64))  # Resize to the required size
    face = face.astype("float") / 255.0  # Normalize pixel values
    face = img_to_array(face)
    face = np.expand_dims(face, axis=0)
    return face

def cv2PutText(img, text, postion, textColor=(0, 255, 0), textSize=30):
    if (isinstance(img,np.ndarray)):
        img = Image.fromarray(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))

    draw = ImageDraw.Draw(img)
    font = ImageFont.truetype("font/simsun.ttc", textSize, encoding="utf-8")
    draw.text(postion, text, textColor, font=font)
    img = cv2.cvtColor(np.asarray(img), cv2.COLOR_RGB2BGR)
    return img

def draw_landmarks_on_image(rgb_image, detection_result):
  pose_landmarks_list = detection_result.pose_landmarks
  annotated_image = np.copy(rgb_image)

  # Loop through the detected poses to visualize.
  for idx in range(len(pose_landmarks_list)):
    pose_landmarks = pose_landmarks_list[idx]

    # Draw the pose landmarks.
    pose_landmarks_proto = landmark_pb2.NormalizedLandmarkList()
    pose_landmarks_proto.landmark.extend([
      landmark_pb2.NormalizedLandmark(x=landmark.x, y=landmark.y, z=landmark.z) for landmark in pose_landmarks
    ])
    solutions.drawing_utils.draw_landmarks(
      annotated_image,
      pose_landmarks_proto,
      solutions.pose.POSE_CONNECTIONS,
      solutions.drawing_styles.get_default_pose_landmarks_style())
  return annotated_image

class MyApp(QMainWindow, Ui_MainWindow, threading.Thread):
    filter_count = 0
    def __init__(self,chunk=1024, rate=16000):
        super().__init__()
        threading.Thread.__init__(self)
        QMainWindow.__init__(self)
        Ui_MainWindow.__init__(self)
        self.setupUi(self)
        self.initUI()
        self.hd_camera=-1
        self.voice_index=-1
        self.eye_camera=-1
        self.face_camera=-1
        self.hd_width = 1920
        self.hd_height = 1080
        self.face_width = 1280
        self.eye_width = 1280
        self.face_height = 1920
        self.eye_height = 720
        self.hd_fps = 30.0
        self.face_fps = 10.0
        self.eye_fps = 10.0
        self.CHUNK = chunk
        self.FORMAT = pyaudio.paInt16
        self.CHANNELS = 1
        self.RATE = rate
        self._win1_running = True
        self._win2_running = True
        self._win3_running = True
        self._voice_get = True
        self._begin = False
        self._frames = []
        self._init_running = True
        self._start_running = True
        self.record_judge = False
        self.recording_time = '0'
        self.recording_location = 'record/'
        self.recording_information = self.recording_time + '
        ↪   ' + self.recording_location
    def initUI(self):
        self.original1_image = ImageCvQtContainer.Image(
            'camera1', self.original_frame_lbl)
        self.original2_image = ImageCvQtContainer.Image(
            'camera2', self.processed_frame_lbl)
        self.eye_image =
        ↪   ImageCvQtContainer.Image('eye_camera',
        ↪   self.eye_lbl)
        self.voice_image =
        ↪   ImageCvQtContainer.Image('voice_mic',
        ↪   self.voice_lbl)
        self.setBackground()
        self.createButtons()
        self.Child_File_Setting()
    def r_json(self):
        # 读取配置文件
        with open('config/db.json',encoding='utf-8',mode='r')
        ↪   as rf:
        data = rf.read().encode()
        configuration = json.loads(data)
        print('configuation')
        print(configuration)
        self.hd_cream_name=configuration['hd_camera_name']
        self.face_camera_name=configuration['face_camera_name']
        self.eye_camera_name=configuration['eye_camera_name']
        self.vocie_name=configuration['voice_name']
        self.hk_voice_name=configuration['hk_voice_name']
        self.bluetooth_voice_name=configuration['bluetooth_voice_name']
        self.save_path=configuration['save_path']
        self.hd_width=configuration['hd_width']
        self.hd_height=configuration['hd_height']
        self.hd_fps=configuration['hd_fps']
        self.face_width=configuration['face_width']
        self.face_height=configuration['face_height']
        self.face_fps=configuration['face_fps']
        self.eye_width=configuration['eye_width']
        self.eye_height=configuration['eye_height']
        self.eye_fps=configuration['eye_fps']
        self.voice_format = configuration['voice_format']
        self.video_format = configuration['video_format']
        self.recorfing_location = self.save_path
    def initfrom(self):
        if self._init_running:
        self._init_running=False
        pygame.init()
        pygame.camera.init()
        cameralist = pygame.camera.list_cameras()
        print(cameralist)
        p = pyaudio.PyAudio()
        info = p.get_host_api_info_by_index(0)
        numdevices = info.get('deviceCount')
        voice_list = []
        for i in range(0, numdevices):
```

```
            if (p.get_device_info_by_host_api_device_index(0, i).get('maxInputChannels')) > 0:
                voice_list.append(p.get_device_info_by_host_api_device_index(0, i).get('name'))
        print(voice_list)
        try:
            self.r_json()
        except:
            self.statusbar.showMessage('未找到配置文件',5000)
        try:
            self.hd_camera=cameralist.index(self.hd_cream_name)
        except:
            self.hd_camera=-1
        try:
            self.face_camera=cameralist.index(self.face_camera_name)
        except:
            self.face_camera=-1
        try:
            self.eye_camera=cameralist.index(self.eye_camera_name)
        except:
            self.eye_camera=-1
        try:
            if self.hk_voice_name in voice_list:
                self.voice_index = voice_list.index(self.hk_voice_name)
            else:
                if self.vocie_name in voice_list and self.bluetooth_voice_name in voice_list:
                    self.voice_index=voice_list.index(self.vocie_name)
                    self.CHANNELS=2
                elif self.vocie_name in voice_list and self.bluetooth_voice_name not in voice_list:
                    self.voice_index=voice_list.index(self.vocie_name)
                    self.CHANNELS=2
                elif self.vocie_name not in voice_list and self.bluetooth_voice_name in voice_list:
                    self.voice_index=voice_list.index(self.bluetooth_voice_name)
                else:
                    self.voice_index = -1
        except:
            self.voice_index=-1
        print(self.hd_camera,'行为设备 index')
        print(self.face_camera,'面部设备 index')
        print(self.eye_camera,'眼部设备 index')
        print(self.voice_index,'音频设备 index')
        if os.path.exists(self.save_path) is False:
            os.makedirs(self.save_path)
        if self.hd_camera == -1 and self.face_camera == -1 and self.eye_camera ==-1 and self.voice_index ==-1:
            self.statusbar.showMessage('未检测到任何可用设备', 5000)
        elif self.hd_camera != -1 and self.face_camera == -1 and self.eye_camera ==-1 and self.voice_index ==-1:
            self.statusbar.showMessage('未检测到面部、眼部、音频设备',5000)
        elif self.hd_camera == -1 and self.face_camera != -1 and self.eye_camera ==-1 and self.voice_index ==-1:
            self.statusbar.showMessage('未检测到行为、眼部、音频设备', 5000)
```

```python
            elif self.hd_camera == -1 and self.face_camera == -1 and self.eye_camera !=-1 and self.voice_index ==-1:
                self.statusbar.showMessage('未检测到行为、面部、音频设备', 5000)
            elif self.hd_camera == -1 and self.face_camera == -1 and self.eye_camera ==-1 and self.voice_index !=-1:
                self.statusbar.showMessage('未检测到行为、面部、眼部设备', 5000)
            elif self.hd_camera != -1 and self.face_camera != -1 and self.eye_camera ==-1 and self.voice_index ==-1:
                self.statusbar.showMessage('未检测到眼部、音频设备', 5000)
            elif self.hd_camera != -1 and self.face_camera == -1 and self.eye_camera !=-1 and self.voice_index ==-1:
                self.statusbar.showMessage('未检测到面部、音频设备', 5000)
            elif self.hd_camera != -1 and self.face_camera == -1 and self.eye_camera ==-1 and self.voice_index !=-1:
                self.statusbar.showMessage('未检测到面部、眼部设备', 5000)
            elif self.hd_camera == -1 and self.face_camera != -1 and self.eye_camera !=-1 and self.voice_index ==-1:
                self.statusbar.showMessage('未检测到行为、音频设备', 5000)
            elif self.hd_camera == -1 and self.face_camera != -1 and self.eye_camera ==-1 and self.voice_index !=-1:
                self.statusbar.showMessage('未检测到行为、眼部设备', 5000)
            elif self.hd_camera == -1 and self.face_camera == -1 and self.eye_camera !=-1 and self.voice_index !=-1:
                self.statusbar.showMessage('未检测到行为、面部设备', 5000)
            elif self.hd_camera != -1 and self.face_camera != -1 and self.eye_camera ==-1 and self.voice_index !=-1:
                self.statusbar.showMessage('未检测到眼部设备', 5000)
            elif self.hd_camera != -1 and self.face_camera == -1 and self.eye_camera !=-1 and self.voice_index !=-1:
                self.statusbar.showMessage('未检测到面部设备', 5000)
            elif self.hd_camera == -1 and self.face_camera != -1 and self.eye_camera !=-1 and self.voice_index !=-1:
                self.statusbar.showMessage('未检测到行为设备', 5000)
            elif self.hd_camera != -1 and self.face_camera != -1 and self.eye_camera !=-1 and self.voice_index ==-1:
                self.statusbar.showMessage('未检测到音频设备', 5000)
            else:
                self.statusbar.showMessage('所有设备准备就绪', 5000)
            self.start()
            # self.win_4_upimage()
        else:
            pass
    def child_CameraVoice_setting(self):#点击此选项时不能处于以下状态：　录制视频中。

        if self._init_running == False:
            QMessageBox.information(self,'警告','进程未中止，请中止进程再点击此选项！')
        else:
        # 结束录制
            self.endRe()
            self.f_p()
            self.Child_window = child_CameraVoice_setting.Child_Window()
            self.Child_window.setWindowModality(Qt.ApplicationModal)
            self.Child_window.show()
    def Child_File_Setting(self):
        # 因为打开软件时便会调用该函数，所以不能调用下面两个方法
        self.child_file_window = Child_File_Setting.child_file_setting()
        self.child_file_window.setWindowModality(Qt.ApplicationModal)
        self.child_file_window.show()
    def start(self):
        threading._start_new_thread(self.win_1,())
        # threading._start_new_thread(self.win_2,())
        # threading._start_new_thread(self.win_3,())
    def win_1(self):
        if self.hd_camera != -1:
            self._win1_running=True
            self.cap1 = 0
            # self.cap1 = cv2.VideoCapture(self.hd_camera,cv2.CAP_DSHOW)
            # self.cap1 = cv2.VideoCapture(self.hd_camera)
            self.cap1 = cv2.VideoCapture("elder_dance.avi")
            # self.cap1.set(3,2560)
            # self.cap1.set(4,1440)
            # self.cap1.set(5,self.hd_fps)
            print(self.cap1.get(3))
            print(self.cap1.get(4))
            self.cap1_3 = self.cap1.get(3)  # 宽
            self.cap1_4 = self.cap1.get(4)  # 高
            self.hd_width = self.cap1_3
            self.hd_height = self.cap1_4
            with open("config/db.json", "r", encoding='UTF-8') as dbfile_r_hd:
                jsonfile_hd = json.load(dbfile_r_hd)
            jsonfile_hd['hd_width'] = self.hd_width
            jsonfile_hd['hd_height'] = self.hd_height
            if self.face_camera_name == self.hd_cream_name:
```

```
            self.face_width = self.cap1_3
            self.face_height = self.cap1_4
            jsonfile_hd['face_width'] = self.face_width
            jsonfile_hd['face_height'] = self.face_height
        if self.hd_cream_name == self.eye_camera_name:
            self.eye_width = self.cap1_3
            self.eye_height = self.cap1_4
            jsonfile_hd['eye_width'] = self.eye_width
            jsonfile_hd['eye_height'] = self.eye_height
            print('toooooo')
        with open("config/db.json", "w", encoding='UTF-8') as dbfile_hd:
            json.dump(jsonfile_hd, dbfile_hd)
        self.cap1.set(3, self.hd_width)
        self.cap1.set(4,self.hd_width)

        while self._win1_running:
                ret, frame = self.cap1.read()
                if ret:
                    frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
                    self.original1_image.updateImage(frame)

                    mp_image = mp.Image(image_format=mp.ImageFormat.SRGB, data=frame)

                    # STEP 4: Detect pose landmarks from the input image.
                    detection_result = detector.detect(mp_image)

                    blank_image = frame.copy()
                    blank_image.fill(0)
                    # STEP 5: Process the detection result. In this case, visualize it.
                    annotated_image = draw_landmarks_on_image(frame, detection_result)
                    blank_image = draw_landmarks_on_image(blank_image, detection_result)
                    self.eye_image.updateImage(annotated_image)
                    self.original2_image.updateImage(blank_image)

                    visualized_mask = None
                    if detection_result.segmentation_masks is not None:
                        rgb = cv2.cvtColor(detection_result.segmentation_masks[0].numpy_view(), cv2.COLOR_GRAY2RGB)
                        rgb = rgb * 255
                        rgb = rgb.astype(np.uint8)
                        visualized_mask = rgb
                    else:
                        black = np.zeros(frame.shape, dtype=np.uint8)
                        visualized_mask = black

                    self.voice_image.updateImage(visualized_mask)

                        # ff.write(str(id) + "\n" + np.array2string(shape) + "\n")


        # if self.face_camera_name == self.hd_cream_name and self.hd_cream_name == self.eye_camera_name:
        #     while self._win1_running:
        #         ret, frame = self.cap1.read()
        #         if ret:
```

```
#             frame = cv2.cvtColor(frame,
↪  cv2.COLOR_BGR2RGB)
#
↪  self.original1_image.updateImage(frame)
#             # self.eye_image.updateImage(frame)
#             #
↪  self.original2_image.updateImage(frame)
# if self.hd_cream_name == self.face_camera_name
↪  and self.hd_cream_name !=
↪  self.eye_camera_name:
#     while self._win1_running:
#         ret, frame = self.cap1.read()
#         if ret:
#             frame = cv2.cvtColor(frame,
↪  cv2.COLOR_BGR2RGB)
#
↪  self.original1_image.updateImage(frame)
#
↪  self.original2_image.updateImage(frame)
# if self.hd_cream_name != self.face_camera_name
↪  and self.hd_cream_name ==
↪  self.eye_camera_name:
#     while self._win1_running:
#         ret, frame = self.cap1.read()
#         if ret:
#             frame = cv2.cvtColor(frame,
↪  cv2.COLOR_BGR2RGB)
#
↪  self.original1_image.updateImage(frame)
#             self.eye_image.updateImage(frame)
# if self.hd_cream_name != self.face_camera_name
↪  and self.hd_cream_name !=
↪  self.eye_camera_name:
#     while self._win1_running:
#         ret, frame = self.cap1.read()
#         if ret:
#             frame = cv2.cvtColor(frame,
↪  cv2.COLOR_BGR2RGB)
```

```
            # self.original1_image.updateImage(frame)

            # self.original1_image.updateImage(img_no_camera)
            # if self.face_camera_name == self.hd_cream_name:
            #     self.original2_image.updateImage(img_no_camera)
            # if self.hd_cream_name == self.eye_camera_name:
            #     self.eye_image.updateImage(img_no_camera)
        else:
            self.original1_image.updateImage(img_no_camera)
    def win_2(self):
        if (self.face_camera != -1) and (self.face_camera_name != self.hd_cream_name):#脸部摄像头存在并且不和行为摄像头重合
            self._win2_running=True
            self.cap2 = 0
            self.cap2 = cv2.VideoCapture(self.face_camera,cv2.CAP_DSHOW)
            self.cap2.set(3,2560)
            self.cap2.set(4,1440)
            self.cap2.set(5,self.face_fps)
            print('cap2'+'_'+str(self.cap2.get(3)))
            print('cap2'+'_'+str(self.cap2.get(4)))
            print('cap2'+'_'+str(self.cap2.get(5)))
            self.cap2_3 = self.cap2.get(3)  # 宽
            self.cap2_4 = self.cap2.get(4)  # 高
            self.face_width = self.cap2_3
            self.face_height = self.cap2_4
            with open("config/db.json", "r", encoding='UTF-8') as dbfile_r_face:
                jsonfile_face = json.load(dbfile_r_face)
            jsonfile_face['face_width'] = self.face_width
            jsonfile_face['face_height'] = self.face_height
            if self.face_camera_name == self.eye_camera_name:
                self.eye_width = self.cap2_3
                self.eye_height = self.cap2_4
                jsonfile_face['eye_width'] = self.eye_width
                jsonfile_face['eye_height'] = self.eye_height
            with open("config/db.json", "w", encoding='UTF-8') as dbfile_face:
                json.dump(jsonfile_face, dbfile_face)
            self.cap2.set(3, self.face_width)
            self.cap2.set(4,self.face_width)
            if self.face_camera_name == self.eye_camera_name:
                while self._win2_running:
                    ret2,frame2 = self.cap2.read()
                    if ret2:
                        frame2 = cv2.cvtColor(frame2, cv2.COLOR_BGR2RGB)
                        self.original2_image.updateImage(frame2)
                        self.eye_image.updateImage(frame2)
            else:
                while self._win2_running:
                    ret2, frame2 = self.cap2.read()
                    if ret2:
                        frame2 = cv2.cvtColor(frame2, cv2.COLOR_BGR2RGB)
                        self.original2_image.updateImage(frame2)
            self.original2_image.updateImage(img_no_camera)
            if self.face_camera_name == self.eye_camera_name:
                self.eye_image.updateImage(img_no_camera)
        else:
            self.original2_image.updateImage(img_no_camera)
    def win_3(self):
        if (self.eye_camera != -1) and (self.eye_camera_name != self.face_camera_name) and (self.eye_camera_name != self.hd_cream_name):#不和脸部、行为重合
            self._win3_running=True
            self.cap3 = 0
            self.cap3 = cv2.VideoCapture(self.eye_camera,cv2.CAP_DSHOW)
            self.cap3.set(3,2560)
            self.cap3.set(4,1440)
            self.cap3.set(5,self.eye_fps)
            print('cap3'+'_'+str(self.cap3.get(3)))
            print('cap3'+'_'+str(self.cap3.get(4)))
            print('cap3'+'_'+str(self.cap3.get(5)))
            self.cap3_3 = self.cap3.get(3)  # 宽
            self.cap3_4 = self.cap3.get(4)  # 高
            self.eye_width = self.cap3_3
            self.eye_height = self.cap3_4
            with open("config/db.json", "r", encoding='UTF-8')
            ↪   as dbfile_r_eye:
                jsonfile_eye = json.load(dbfile_r_eye)
            jsonfile_eye['face_width'] = self.eye_width
            jsonfile_eye['face_height'] = self.eye_height
            with open("config/db.json", "w", encoding='UTF-8')
            ↪   as dbfile_eye:
                json.dump(jsonfile_eye, dbfile_eye)
            self.cap3.set(3, self.eye_width)
            self.cap3.set(4,self.eye_width)
            while self._win3_running:
                ret3, frame3 = self.cap3.read()
                if ret3:
                    frame3 = cv2.cvtColor(frame3,
                    ↪   cv2.COLOR_BGR2RGB)
                    self.eye_image.updateImage(frame3)
            self.eye_image.updateImage(img_no_camera)
        else:
            self.eye_image.updateImage(img_no_camera)
    def save_hd_video(self):
        if self.hd_camera != -1:
            fn = self.lineEdit.text()
            name=self.save_path+"/behavior_" + fn + "_" +
            ↪   datetime.now().strftime('%Y%m%d%H%M%S') +
            ↪   self.video_format
            fourcc = cv2.VideoWriter_fourcc(*'XVID')
            out = cv2.VideoWriter(name, fourcc,
            ↪   int(self.hd_fps), (int(self.hd_width),
            ↪   int(self.hd_height)))
            while self._begin:
                ret,frame=self.cap1.read()
                out.write(frame)
    def save_face_video(self):
        if self.face_camera != -1:
            fn = self.lineEdit.text()
            name=self.save_path+"/face_" + fn + "_" + datetime.now().strftime('%Y%m%d%H%M%S') + self.video_format
            fourcc = cv2.VideoWriter_fourcc(*'XVID')
            out = cv2.VideoWriter(name, fourcc, int(self.face_fps), (int(self.face_width), int(self.face_height)))
            if self.face_camera_name != self.hd_cream_name:
                while self._begin:
                    ret,frame=self.cap2.read()
                    out.write(frame)
            else:
                while self._begin:
                    ret,frame=self.cap1.read()
                    out.write(frame)
    def save_eye_video(self):
        if self.eye_camera != -1:
            fn = self.lineEdit.text()
            name=self.save_path+"/eye_" + fn + "_" + datetime.now().strftime('%Y%m%d%H%M%S') + self.video_format
            fourcc = cv2.VideoWriter_fourcc(*'XVID')
            out = cv2.VideoWriter(name, fourcc, int(self.eye_fps), (int(self.eye_width), int(self.eye_height)))
            if (self.eye_camera_name != self.hd_cream_name) & (self.eye_camera_name != self.face_camera_name):
                while self._begin:
                    ret,frame=self.cap3.read()
                    out.write(frame)
            elif (self.eye_camera_name != self.hd_cream_name) & (self.eye_camera_name == self.face_camera_name):
                while self._begin:
                    ret,frame=self.cap2.read()
                    out.write(frame)
            else:
                while self._begin:
                    ret,frame=self.cap1.read()
                    out.write(frame)
    def win_4_upimage(self):
        if self.voice_index != -1:
            self.voice_image.updateImage(img_voice)
        else:
            pass
    def win_4(self):
        self._voice_get = True
        self._frames=[]
        p=pyaudio.PyAudio()
        print(self.voice_index)
        stream=p.open(format=self.FORMAT,
                      rate=self.RATE,
                      channels=self.CHANNELS,
                      input=True,
                      frames_per_buffer=self.CHUNK,
                      input_device_index=self.voice_index)
        while self._voice_get:
            data = stream.read(self.CHUNK)
            self._frames.append(data)
        stream.stop_stream()
        stream.close()
        p.terminate()
    def voice_save(self):
        p = pyaudio.PyAudio()
        fn = self.lineEdit.text()
        name = self.save_path+"/vioce_" + fn + "_" +
        ↪   datetime.now().strftime('%Y%m%d%H%M%S') +
        ↪   self.voice_format
        wf = wave.open(name, 'wb')
        wf.setnchannels(self.CHANNELS)
        wf.setsampwidth(p.get_sample_size(self.FORMAT))
        wf.setframerate(self.RATE)
        wf.writeframes(b''.join(self._frames))
        wf.close()
```

```python
            print("Saved")
    def updatemessage(self):
        # 只有在点击开始录制时才会显示该消息，点击进程结束时停止显示。
        with open("config/db.json", "r", encoding='UTF-8') as dbfile_r:
            jsonfile = json.load(dbfile_r)
        self.save_path = jsonfile['save_path']
        self.time_end = time.time()
        m, s = divmod(int(self.time_end-self.time_start), 60)
        h, m = divmod(m, 60)
        self.recording_time = str(h)+'时'+str(m)+'分'+str(s)+'秒'
        self.statusbar.showMessage('当前样本编号：' + self.lineEdit.text() + '   保存位置：' + self.save_path + '   录制时间：' + self.recording_time)
        if self._start_running == True:
            self.qttimer.stop()
    def startRe(self):
        print(self.voice_index)
        if self._start_running:
            self._start_running =False
            self._begin = True
            with open("config/db.json", "r", encoding='UTF-8') as dbfile_r:
                jsonfile = json.load(dbfile_r)
            self.voice_format = jsonfile["voice_format"]
            self.video_format = jsonfile["video_format"]
            #需要显示的有：病人ID，当前保存路径，已经录制多长时间，在加一个录制中。
            #已录制多长时间需要动态显示，每秒更新一次状态栏，只在录制视频的时候显示。
            #什么情况下会录制视频？_begin == True ,上面写了，不等于-1就能录制视频。
            if self.voice_index != -1 or self.eye_camera != -1 or self.hd_camera != -1 or self.face_camera != -1:#self.voice_index! =-1，则一定会录制音频。如果有其他设备不为-1则会录制视频，只要有录制情况就显示该消息
                self.time_start = time.time()
                self.record_judge = True #表示是否在录制中。该变量只与qtimer有关，该变量为True则一定有QTimer（）,当然线程终止也会赋值为False，表示结束录制了。
                self.qttimer = QTimer()
                self.qttimer.start(200)
                self.qttimer.timeout.connect(self.updatemessage)
            threading._start_new_thread(self.save_hd_video, ())
            threading._start_new_thread(self.save_face_video, ())
            threading._start_new_thread(self.save_eye_video, ())
            if self.voice_index != -1:
                threading._start_new_thread(self.win_4, ())
        else:
            pass
    def endRe(self):
        if self._start_running==False:
            if self.record_judge == True:# 只有点击
                self.qttimer.stop()
                self.record_judge = False
                self.statusbar.showMessage('录制结束，文件已保存至：' + self.save_path + '录制时长：' + self.recording_time)
            self._start_running=True
            self._begin =False
            if self.voice_index != -1:
                self._voice_get = False
                self.voice_save()
```

```python
                self.statusbar.showMessage('录制结束，文件已保存至：'+self.save_path+' 录制时长：'+self.recording_time)
        else:
            pass
    def closeEvent(self,event):
        # 关闭所有线程，防止不能退出
        # 如果没有运行进程终止，则不能退出
        if self._init_running == False:
            QMessageBox.information(self,'警告','进程未中止，请中止进程再关闭窗口！')
            event.ignore()
        print('closeevent')
    def stop(self):
        if self.record_judge == True:
            QMessageBox.information(self,'警告','正在录制中，请结束录制后再点击停用视频设备！')
        else:
            self._win1_running = False
            self._win2_running = False
            self._win3_running = False
            self._init_running = True
            print('stop','所有通道均已断开')
            self.voice_image.updateImage(img_no_voice)
            self.statusbar.showMessage('所有通道均已断开')
    def f_p(self):
        #点进程终止的时候录音进程可能不会终止，而视频录制会结束，必须再点结束录制才能结束---------先不管它，假设先用不到它。--------还是先解决它吧。
        #进程中止前判断是否在录制中，若在录制中，需要点击录制结束
        if self.record_judge == True:
            QMessageBox.information(self,'警告','正在录制中，请结束录制后再点击进程中止！')
        else:
            self._win1_running = False
            self._win2_running = False
            self._win3_running = False
            self._init_running = True
            if self.record_judge == True:
                self._begin = False
                self._start_running = True
                self._voice_get = False
                self.voice_save()
                self.qttimer.stop()
                self.record_judge = False
            self._voice_get = True
            self._begin = False
            print('stop', '所有进程均已中止')
            self.statusbar.showMessage('所有进程均已中止')
            self.voice_image.updateImage(img_no_voice)
    def about(self):
        QMessageBox.about(self, " 面部表型特征提取系统",
                                  "<p> 山东省精神卫生中心 </p>")
    def updateImages(self):
        self.calculateProcessed()
        self.calculateOriginal()
    def createButtons(self):
        self.initButton.clicked.connect(self.initfrom)
        self.pushButton.clicked.connect(self.stop)
        self.shotButton.clicked.connect(self.f_p)
        self.actionAbout.triggered.connect(self.about)
        self.startButton.clicked.connect(self.startRe)
        self.endButton.clicked.connect(self.endRe)
        self.action12.triggered.connect(self.child_CameraVoice_setting)    #子窗口配置摄像头和麦克风
        self.actionse.triggered.connect(self.Child_File_Setting)#选择文件路径
    def setBackground(self):
        cv_img_rgb = img_no_camera
        self.original1_image.updateImage(cv_img_rgb)
        self.original2_image.updateImage(cv_img_rgb)
        self.eye_image.updateImage(cv_img_rgb)
        self.voice_image.updateImage(img_no_voice)
```

1.15 CvPyGui/ui/__init__.py

```

```

1.16 CvPyGui/ui/child_file.py

```
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file
↪   'child_file.ui'
#
# Created by: PyQt5 UI code generator 5.15.10
#
# WARNING: Any manual changes made to this file will be lost
↪   when pyuic5 is
# run again.  Do not edit this file unless you know what you
↪   are doing.


from PyQt5 import QtCore, QtGui, QtWidgets


class Ui_Form(object):
    def setupUi(self, Form):
        Form.setObjectName("Form")
        Form.resize(385, 269)
        self.file_ok = QtWidgets.QPushButton(Form)
        self.file_ok.setGeometry(QtCore.QRect(290, 230, 75,
        ↪   23))
        self.file_ok.setObjectName("file_ok")
        self.label = QtWidgets.QLabel(Form)
        self.label.setGeometry(QtCore.QRect(10, 100, 101, 16))
        self.label.setObjectName("label")
        self.layoutWidget = QtWidgets.QWidget(Form)
        self.layoutWidget.setGeometry(QtCore.QRect(14, 5, 361,
        ↪   71))
        self.layoutWidget.setObjectName("layoutWidget")
        self.verticalLayout = QtWidgets.QVBoxLayout(self.layoutWidget)
        self.verticalLayout.setContentsMargins(0, 0, 0, 0)
        self.verticalLayout.setObjectName("verticalLayout")
        self.horizontalLayout_2 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_2.setObjectName("horizontalLayout_2")
        self.show_file_path_label = QtWidgets.QLabel(self.layoutWidget)
        self.show_file_path_label.setStyleSheet("")
        self.show_file_path_label.setObjectName("show_file_path_label")
        self.horizontalLayout_2.addWidget(self.show_file_path_label)
        spacerItem = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Minimum, QtWidgets.QSizePolicy.Expanding)
        self.horizontalLayout_2.addItem(spacerItem)
        self.verticalLayout.addLayout(self.horizontalLayout_2)
        self.horizontalLayout = QtWidgets.QHBoxLayout()
        self.horizontalLayout.setObjectName("horizontalLayout")
        self.file_path_label = QtWidgets.QLabel(self.layoutWidget)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Minimum, QtWidgets.QSizePolicy.Preferred)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.file_path_label.sizePolicy().hasHeightForWidth())
        self.file_path_label.setSizePolicy(sizePolicy)
        self.file_path_label.setMinimumSize(QtCore.QSize(0, 20))
        self.file_path_label.setObjectName("file_path_label")
        self.horizontalLayout.addWidget(self.file_path_label)
        self.choice_file_path_button = QtWidgets.QPushButton(self.layoutWidget)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Fixed, QtWidgets.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.choice_file_path_button.sizePolicy().hasHeightForWidth())
        self.choice_file_path_button.setSizePolicy(sizePolicy)
        self.choice_file_path_button.setObjectName("choice_file_path_button")
        self.horizontalLayout.addWidget(self.choice_file_path_button)
        self.verticalLayout.addLayout(self.horizontalLayout)
        self.layoutWidget1 = QtWidgets.QWidget(Form)
        self.layoutWidget1.setGeometry(QtCore.QRect(40, 180, 291, 22))
        self.layoutWidget1.setObjectName("layoutWidget1")
        self.horizontalLayout_3 = QtWidgets.QHBoxLayout(self.layoutWidget1)
        self.horizontalLayout_3.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout_3.setObjectName("horizontalLayout_3")
        self.label_3 = QtWidgets.QLabel(self.layoutWidget1)
        self.label_3.setObjectName("label_3")
        self.horizontalLayout_3.addWidget(self.label_3)
        self.radioButton_mp3 = QtWidgets.QRadioButton(self.layoutWidget1)
        self.radioButton_mp3.setEnabled(False)
        self.radioButton_mp3.setObjectName("radioButton_mp3")
        self.horizontalLayout_3.addWidget(self.radioButton_mp3)
        self.radioButton_wav = QtWidgets.QRadioButton(self.layoutWidget1)
        self.radioButton_wav.setEnabled(False)
        self.radioButton_wav.setObjectName("radioButton_wav")
        self.horizontalLayout_3.addWidget(self.radioButton_wav)
        self.layoutWidget2 = QtWidgets.QWidget(Form)
        self.layoutWidget2.setGeometry(QtCore.QRect(40, 130, 291, 22))
        self.layoutWidget2.setObjectName("layoutWidget2")
        self.horizontalLayout_4 = QtWidgets.QHBoxLayout(self.layoutWidget2)
        self.horizontalLayout_4.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout_4.setObjectName("horizontalLayout_4")
        self.label_2 = QtWidgets.QLabel(self.layoutWidget2)
        self.label_2.setObjectName("label_2")
        self.horizontalLayout_4.addWidget(self.label_2)
        self.radioButton_mp4 = QtWidgets.QRadioButton(self.layoutWidget2)
        self.radioButton_mp4.setObjectName("radioButton_mp4")
        self.horizontalLayout_4.addWidget(self.radioButton_mp4)
        self.radioButton_avi = QtWidgets.QRadioButton(self.layoutWidget2)
        self.radioButton_avi.setObjectName("radioButton_avi")
        self.horizontalLayout_4.addWidget(self.radioButton_avi)

        self.retranslateUi(Form)
        QtCore.QMetaObject.connectSlotsByName(Form)

    def retranslateUi(self, Form):
        _translate = QtCore.QCoreApplication.translate
        Form.setWindowTitle(_translate("Form", "初始化配置"))
        self.file_ok.setText(_translate("Form", "确认"))
        self.label.setText(_translate("Form", "音视频格式选择"))
        self.show_file_path_label.setText(_translate("Form", "当前文件保存路径:"))
        self.file_path_label.setText(_translate("Form", "C:"))
        self.choice_file_path_button.setText(_translate("Form", "选择文件保存路径"))
        self.label_3.setText(_translate("Form", "音频格式"))
        self.radioButton_mp3.setText(_translate("Form", "mp3"))
        self.radioButton_wav.setText(_translate("Form", "wav"))
        self.label_2.setText(_translate("Form", "视频格式"))
        self.radioButton_mp4.setText(_translate("Form", "mp4"))
        self.radioButton_avi.setText(_translate("Form", "avi"))


if __name__ == "__main__":
    import sys
    app = QtWidgets.QApplication(sys.argv)
    Form = QtWidgets.QWidget()
    ui = Ui_Form()
    ui.setupUi(Form)
    Form.show()
    sys.exit(app.exec_())
```

1.17 CvPyGui/ui/child_file.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>Form</class>
 <widget class="QWidget" name="Form">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>385</width>
    <height>269</height>
   </rect>
  </property>
  <property name="windowTitle">
   <string>初始化配置</string>
  </property>
  <widget class="QPushButton" name="file_ok">
```

```
    <property name="geometry">
     <rect>
      <x>290</x>
      <y>230</y>
      <width>75</width>
      <height>23</height>
     </rect>
    </property>
    <property name="text">
     <string>确认</string>
    </property>
   </widget>
   <widget class="QLabel" name="label">
    <property name="geometry">
     <rect>
      <x>10</x>
      <y>100</y>
      <width>101</width>
      <height>16</height>
     </rect>
    </property>
    <property name="text">
     <string>音视频格式选择</string>
    </property>
   </widget>
   <widget class="QWidget" name="layoutWidget">
    <property name="geometry">
     <rect>
      <x>14</x>
      <y>5</y>
      <width>361</width>
      <height>71</height>
     </rect>
    </property>
    <layout class="QVBoxLayout" name="verticalLayout">
     <item>
      <layout class="QHBoxLayout" name="horizontalLayout_2">
       <item>
        <widget class="QLabel" name="show_file_path_label">
         <property name="styleSheet">
          <string notr="true"/>
         </property>
         <property name="text">
          <string>当前文件保存路径:</string>
         </property>
        </widget>
       </item>
       <item>
        <spacer name="horizontalSpacer">
         <property name="orientation">
          <enum>Qt::Orientation::Horizontal</enum>
         </property>
         <property name="sizeHint" stdset="0">
          <size>
           <width>40</width>
           <height>20</height>
          </size>
         </property>
        </spacer>
       </item>
      </layout>
     </item>
     <item>
      <layout class="QHBoxLayout" name="horizontalLayout">
       <item>
        <widget class="QLabel" name="file_path_label">
         <property name="sizePolicy">
          <sizepolicy hsizetype="Minimum"
           ↪  vsizetype="Preferred">
           <horstretch>0</horstretch>
           <verstretch>0</verstretch>
          </sizepolicy>
         </property>
         <property name="minimumSize">
          <size>
           <width>0</width>
           <height>20</height>
          </size>
         </property>
         <property name="text">
          <string>C:</string>
         </property>
        </widget>
       </item>
       <item>
        <widget class="QPushButton" name="choice_file_path_button">
         <property name="sizePolicy">
          <sizepolicy hsizetype="Fixed" vsizetype="Fixed">
           <horstretch>0</horstretch>
           <verstretch>0</verstretch>
          </sizepolicy>
         </property>
         <property name="text">
          <string>选择文件保存路径</string>
         </property>
        </widget>
       </item>
      </layout>
     </item>
    </layout>
   </widget>
   <widget class="QWidget" name="layoutWidget">
    <property name="geometry">
     <rect>
      <x>40</x>
      <y>180</y>
      <width>291</width>
      <height>22</height>
     </rect>
    </property>
    <layout class="QHBoxLayout" name="horizontalLayout_3">
     <item>
      <widget class="QLabel" name="label_3">
       <property name="text">
        <string>音频格式</string>
       </property>
      </widget>
     </item>
     <item>
      <widget class="QRadioButton" name="radioButton_mp3">
       <property name="enabled">
        <bool>false</bool>
       </property>
       <property name="text">
        <string>mp3</string>
       </property>
      </widget>
     </item>
     <item>
      <widget class="QRadioButton" name="radioButton_wav">
       <property name="enabled">
        <bool>false</bool>
       </property>
       <property name="text">
        <string>wav</string>
       </property>
      </widget>
     </item>
    </layout>
   </widget>
   <widget class="QWidget" name="layoutWidget">
    <property name="geometry">
     <rect>
      <x>40</x>
      <y>130</y>
      <width>291</width>
      <height>22</height>
     </rect>
    </property>
    <layout class="QHBoxLayout" name="horizontalLayout_4">
     <item>
      <widget class="QLabel" name="label_2">
       <property name="text">
        <string>视频格式</string>
       </property>
      </widget>
     </item>
     <item>
      <widget class="QRadioButton" name="radioButton_mp4">
       <property name="text">
        <string>mp4</string>
       </property>
      </widget>
     </item>
     <item>
      <widget class="QRadioButton" name="radioButton_avi">
       <property name="text">
        <string>avi</string>
       </property>
      </widget>
     </item>
    </layout>
   </widget>
  </widget>
  <resources/>
  <connections/>
 </ui>
```

1.18 CvPyGui/ui/child_test.py

```python
from PyQt5 import QtCore, QtGui, QtWidgets
class Ui_Form(object):
    def setupUi(self, Form):
        Form.setObjectName("Form")
        Form.resize(400, 310)
        self.tabWidget = QtWidgets.QTabWidget(Form)
        self.tabWidget.setGeometry(QtCore.QRect(10, 10, 381, 261))
        self.tabWidget.setObjectName("tabWidget")
        self.tab = QtWidgets.QWidget()
        self.tab.setObjectName("tab")
        self.layoutWidget = QtWidgets.QWidget(self.tab)
        self.layoutWidget.setGeometry(QtCore.QRect(0, 10, 371, 211))
        self.layoutWidget.setObjectName("layoutWidget")
        self.horizontalLayout_2 = QtWidgets.QHBoxLayout(self.layoutWidget)
        self.horizontalLayout_2.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout_2.setObjectName("horizontalLayout_2")
        spacerItem = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_2.addItem(spacerItem)
        self.verticalLayout_2 = QtWidgets.QVBoxLayout()
        self.verticalLayout_2.setObjectName("verticalLayout_2")
        self.verticalLayout = QtWidgets.QVBoxLayout()
        self.verticalLayout.setObjectName("verticalLayout")
        self.horizontalLayout = QtWidgets.QHBoxLayout()
        self.horizontalLayout.setObjectName("horizontalLayout")
        spacerItem1 = QtWidgets.QSpacerItem(30, 20, QtWidgets.QSizePolicy.Maximum, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout.addItem(spacerItem1)
        self.tab1_combobox = QtWidgets.QComboBox(self.layoutWidget)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.MinimumExpanding, QtWidgets.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.tab1_combobox.sizePolicy().hasHeightForWidth())
        self.tab1_combobox.setSizePolicy(sizePolicy)
        self.tab1_combobox.setObjectName("tab1_combobox")
        self.horizontalLayout.addWidget(self.tab1_combobox)
        spacerItem2 = QtWidgets.QSpacerItem(30, 20, QtWidgets.QSizePolicy.Maximum, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout.addItem(spacerItem2)
        self.verticalLayout.addLayout(self.horizontalLayout)
        spacerItem3 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.verticalLayout.addItem(spacerItem3)
        self.verticalLayout_2.addLayout(self.verticalLayout)
        self.tab1_label = QtWidgets.QLabel(self.layoutWidget)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Ignored, QtWidgets.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.tab1_label.sizePolicy().hasHeightForWidth())
        self.tab1_label.setSizePolicy(sizePolicy)
        self.tab1_label.setMinimumSize(QtCore.QSize(256, 144))
        self.tab1_label.setMaximumSize(QtCore.QSize(256, 144))
        self.tab1_label.setAlignment(QtCore.Qt.AlignCenter)
        self.tab1_label.setObjectName("tab1_label")
        self.verticalLayout_2.addWidget(self.tab1_label)
        self.horizontalLayout_2.addLayout(self.verticalLayout_2)
        spacerItem4 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_2.addItem(spacerItem4)
        self.tabWidget.addTab(self.tab, "")
        self.tab_2 = QtWidgets.QWidget()
        self.tab_2.setObjectName("tab_2")
        self.layoutWidget_2 = QtWidgets.QWidget(self.tab_2)
        self.layoutWidget_2.setGeometry(QtCore.QRect(0, 10, 371, 211))
        self.layoutWidget_2.setObjectName("layoutWidget_2")
        self.horizontalLayout_7 = QtWidgets.QHBoxLayout(self.layoutWidget_2)
        self.horizontalLayout_7.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout_7.setObjectName("horizontalLayout_7")
        spacerItem5 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_7.addItem(spacerItem5)
        self.verticalLayout_5 = QtWidgets.QVBoxLayout()
        self.verticalLayout_5.setObjectName("verticalLayout_5")
        self.verticalLayout_6 = QtWidgets.QVBoxLayout()
        self.verticalLayout_6.setObjectName("verticalLayout_6")
        self.horizontalLayout_8 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_8.setObjectName("horizontalLayout_8")
        spacerItem6 = QtWidgets.QSpacerItem(30, 20, QtWidgets.QSizePolicy.Maximum, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_8.addItem(spacerItem6)
        self.tab1_combobox_3 = QtWidgets.QComboBox(self.layoutWidget_2)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.MinimumExpanding, QtWidgets.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.tab1_combobox_3.sizePolicy().hasHeightForWidth())
        self.tab1_combobox_3.setSizePolicy(sizePolicy)
        self.tab1_combobox_3.setObjectName("tab1_combobox_3")
        self.horizontalLayout_8.addWidget(self.tab1_combobox_3)
        spacerItem7 = QtWidgets.QSpacerItem(30, 20, QtWidgets.QSizePolicy.Maximum, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_8.addItem(spacerItem7)
        self.verticalLayout_6.addLayout(self.horizontalLayout_8)
        spacerItem8 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.verticalLayout_6.addItem(spacerItem8)
        self.verticalLayout_5.addLayout(self.verticalLayout_6)
        self.tab1_label_3 = QtWidgets.QLabel(self.layoutWidget_2)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Ignored, QtWidgets.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.tab1_label_3.sizePolicy().hasHeightForWidth())
        self.tab1_label_3.setSizePolicy(sizePolicy)
        self.tab1_label_3.setMinimumSize(QtCore.QSize(256, 144))
        self.tab1_label_3.setMaximumSize(QtCore.QSize(256, 144))
        self.tab1_label_3.setAlignment(QtCore.Qt.AlignCenter)
        self.tab1_label_3.setObjectName("tab1_label_3")
        self.verticalLayout_5.addWidget(self.tab1_label_3)
        self.horizontalLayout_7.addLayout(self.verticalLayout_5)
        spacerItem9 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_7.addItem(spacerItem9)
        # self.tabWidget.addTab(self.tab_2, "")
        self.tab_3 = QtWidgets.QWidget()
        self.tab_3.setObjectName("tab_3")
        self.layoutWidget_3 = QtWidgets.QWidget(self.tab_3)
        self.layoutWidget_3.setGeometry(QtCore.QRect(0, 10, 371, 211))
        self.layoutWidget_3.setObjectName("layoutWidget_3")
        self.horizontalLayout_4 = QtWidgets.QHBoxLayout(self.layoutWidget_3)
        self.horizontalLayout_4.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout_4.setObjectName("horizontalLayout_4")
        spacerItem10 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_4.addItem(spacerItem10)
        self.verticalLayout_7 = QtWidgets.QVBoxLayout()
        self.verticalLayout_7.setObjectName("verticalLayout_7")
        self.verticalLayout_8 = QtWidgets.QVBoxLayout()
        self.verticalLayout_8.setObjectName("verticalLayout_8")
        self.horizontalLayout_9 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_9.setObjectName("horizontalLayout_9")
        spacerItem11 = QtWidgets.QSpacerItem(30, 20, QtWidgets.QSizePolicy.Maximum, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_9.addItem(spacerItem11)
        self.tab1_combobox_4 = QtWidgets.QComboBox(self.layoutWidget_3)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.MinimumExpanding, QtWidgets.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.tab1_combobox_4.sizePolicy().hasHeightForWidth())
        self.tab1_combobox_4.setSizePolicy(sizePolicy)
        self.tab1_combobox_4.setObjectName("tab1_combobox_4")
        self.horizontalLayout_9.addWidget(self.tab1_combobox_4)
        spacerItem12 = QtWidgets.QSpacerItem(30, 20, QtWidgets.QSizePolicy.Maximum, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_9.addItem(spacerItem12)
        self.verticalLayout_8.addLayout(self.horizontalLayout_9)
        spacerItem13 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.verticalLayout_8.addItem(spacerItem13)
        self.verticalLayout_7.addLayout(self.verticalLayout_8)
        self.tab1_label_4 = QtWidgets.QLabel(self.layoutWidget_3)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Ignored, QtWidgets.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.tab1_label_4.sizePolicy().hasHeightForWidth())
        self.tab1_label_4.setSizePolicy(sizePolicy)
        self.tab1_label_4.setMinimumSize(QtCore.QSize(256, 144))
        self.tab1_label_4.setMaximumSize(QtCore.QSize(256, 144))
        self.tab1_label_4.setAlignment(QtCore.Qt.AlignCenter)
        self.tab1_label_4.setObjectName("tab1_label_4")
        self.verticalLayout_7.addWidget(self.tab1_label_4)
        self.horizontalLayout_4.addLayout(self.verticalLayout_7)
        spacerItem14 = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_4.addItem(spacerItem14)
        # self.tabWidget.addTab(self.tab_3, "")
        self.tab_4 = QtWidgets.QWidget()
        self.tab_4.setObjectName("tab_4")
        self.layoutWidget1 = QtWidgets.QWidget(self.tab_4)
        self.layoutWidget1.setGeometry(QtCore.QRect(0, 10, 371, 211))
        self.layoutWidget1.setObjectName("layoutWidget1")
        self.horizontalLayout_13 = QtWidgets.QHBoxLayout(self.layoutWidget1)
        self.horizontalLayout_13.setContentsMargins(0, 0, 0, 0)
        self.horizontalLayout_13.setObjectName("horizontalLayout_13")
        spacerItem15 = QtWidgets.QSpacerItem(40, 40, QtWidgets.QSizePolicy.Expanding, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_13.addItem(spacerItem15)
        self.verticalLayout_13 = QtWidgets.QVBoxLayout()
        self.verticalLayout_13.setObjectName("verticalLayout_13")
        self.verticalLayout_14 = QtWidgets.QVBoxLayout()
        self.verticalLayout_14.setObjectName("verticalLayout_14")
        self.horizontalLayout_14 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_14.setObjectName("horizontalLayout_14")
        spacerItem16 = QtWidgets.QSpacerItem(31, 20, QtWidgets.QSizePolicy.Maximum, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_14.addItem(spacerItem16)
        self.tab1_combobox_7 = QtWidgets.QComboBox(self.layoutWidget1)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.MinimumExpanding, QtWidgets.QSizePolicy.Fixed)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.tab1_combobox_7.sizePolicy().hasHeightForWidth())
        self.tab1_combobox_7.setSizePolicy(sizePolicy)
        self.tab1_combobox_7.setMinimumSize(QtCore.QSize(160, 0))
        self.tab1_combobox_7.setObjectName("tab1_combobox_7")
        self.horizontalLayout_14.addWidget(self.tab1_combobox_7)
        spacerItem17 = QtWidgets.QSpacerItem(31, 20, QtWidgets.QSizePolicy.Maximum, QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_14.addItem(spacerItem17)
        self.verticalLayout_14.addLayout(self.horizontalLayout_14)
        spacerItem18 = QtWidgets.QSpacerItem(256, 55,
        ↪  QtWidgets.QSizePolicy.Minimum,
        ↪  QtWidgets.QSizePolicy.Minimum)
        self.verticalLayout_14.addItem(spacerItem18)
        self.verticalLayout_13.addLayout(self.verticalLayout_⌟
        ↪  14)
        self.progressBar =
        ↪  QtWidgets.QProgressBar(self.layoutWidget1)
        self.progressBar.setProperty("value", 0)
        self.progressBar.setTextVisible(False)
        self.progressBar.setObjectName("progressBar")
        self.verticalLayout_13.addWidget(self.progressBar)
        spacerItem19 = QtWidgets.QSpacerItem(40, 60,
        ↪  QtWidgets.QSizePolicy.Expanding,
        ↪  QtWidgets.QSizePolicy.Minimum)
        self.verticalLayout_13.addItem(spacerItem19)
        self.horizontalLayout_13.addLayout(self.verticalLayou⌟
        ↪  t_13)
        spacerItem20 = QtWidgets.QSpacerItem(40, 20,
        ↪  QtWidgets.QSizePolicy.Expanding,
        ↪  QtWidgets.QSizePolicy.Minimum)
        self.horizontalLayout_13.addItem(spacerItem20)
        self.tabWidget.addTab(self.tab_4, "")
        self.OK = QtWidgets.QPushButton(Form)
        self.OK.setGeometry(QtCore.QRect(260, 280, 111, 23))
        self.OK.setObjectName("OK")
        self.retranslateUi(Form)
        self.tabWidget.setCurrentIndex(0)
        QtCore.QMetaObject.connectSlotsByName(Form)
    def retranslateUi(self, Form):
        _translate = QtCore.QCoreApplication.translate
        Form.setWindowTitle(_translate("Form",
        ↪  "音视频设备设置"))
        self.tab1_label.setText(_translate("Form",
        ↪  "TextLabel"))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self⌟
        ↪  .tab), _translate("Form",
        ↪  "视频源"))
```

```python
        self.tab1_label_3.setText(_translate("Form", "TextLabel"))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab_2), _translate("Form", "面部"))
        self.tab1_label_4.setText(_translate("Form", "TextLabel"))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab_3), _translate("Form", "眼睛"))
        self.tabWidget.setTabText(self.tabWidget.indexOf(self.tab_4), _translate("Form", "音频源"))
        self.OK.setText(_translate("Form", "确认"))
```

1.19 CvPyGui/ui/gui.py

```python
# -*- coding: utf-8 -*-
# Form implementation generated from reading ui file 'gui.ui'
#
# Created by: PyQt5 UI code generator 5.9.2
#
# WARNING! All changes made in this file will be lost!
from PyQt5 import QtCore, QtGui, QtWidgets
class Ui_MainWindow(object):
    def setupUi(self, MainWindow):
        MainWindow.setObjectName("MainWindow")
        MainWindow.resize(830, 578)
        icon = QtGui.QIcon()
        icon.addPixmap(QtGui.QPixmap("icon.ico"), QtGui.QIcon.Normal, QtGui.QIcon.Off)
        MainWindow.setWindowIcon(icon)
        self.centralwidget = QtWidgets.QWidget(MainWindow)
        self.centralwidget.setObjectName("centralwidget")
        self.verticalLayout_4 = QtWidgets.QVBoxLayout(self.centralwidget)
        self.verticalLayout_4.setObjectName("verticalLayout_4")
        self.h_img_lay = QtWidgets.QHBoxLayout()
        self.h_img_lay.setObjectName("h_img_lay")
        self.original_img_v_lay = QtWidgets.QVBoxLayout()
        self.original_img_v_lay.setObjectName("original_img_v⌟
        ↪  _lay")
        spacerItem = QtWidgets.QSpacerItem(20, 40,
        ↪  QtWidgets.QSizePolicy.Minimum,
        ↪  QtWidgets.QSizePolicy.Expanding)
        self.original_img_v_lay.addItem(spacerItem)
        self.original_name_lbl =
        ↪  QtWidgets.QLabel(self.centralwidget)
        self.original_name_lbl.setScaledContents(False)
        self.original_name_lbl.setAlignment(QtCore.Qt.AlignCe⌟
        ↪  nter)
        self.original_name_lbl.setObjectName("original_name_l⌟
        ↪  bl")
        self.original_img_v_lay.addWidget(self.original_name_⌟
        ↪  lbl)
        spacerItem1 = QtWidgets.QSpacerItem(20, 40,
        ↪  QtWidgets.QSizePolicy.Minimum,
        ↪  QtWidgets.QSizePolicy.Expanding)
        self.original_img_v_lay.addItem(spacerItem1)
        self.original_frame_lbl =
        ↪  QtWidgets.QLabel(self.centralwidget)
        self.original_frame_lbl.setMinimumSize(QtCore.QSize(4⌟
        ↪  00,
        ↪  240))
        self.original_frame_lbl.setAlignment(QtCore.Qt.AlignC⌟
        ↪  enter)
        self.original_frame_lbl.setObjectName("original_frame⌟
        ↪  _lbl")
        self.original_img_v_lay.addWidget(self.original_frame⌟
        ↪  _lbl)
        spacerItem2 = QtWidgets.QSpacerItem(20, 40,
        ↪  QtWidgets.QSizePolicy.Minimum,
        ↪  QtWidgets.QSizePolicy.Expanding)
        self.original_img_v_lay.addItem(spacerItem2)
        self.h_img_lay.addLayout(self.original_img_v_lay)
        self.processed_img_v_lay = QtWidgets.QVBoxLayout()
        self.processed_img_v_lay.setObjectName("processed_img⌟
        ↪  _v_lay")
        spacerItem3 = QtWidgets.QSpacerItem(20, 40,
        ↪  QtWidgets.QSizePolicy.Minimum,
        ↪  QtWidgets.QSizePolicy.Expanding)
        self.processed_img_v_lay.addItem(spacerItem3)
        self.processed_name_lbl =
        ↪  QtWidgets.QLabel(self.centralwidget)
        self.processed_name_lbl.setAlignment(QtCore.Qt.AlignC⌟
        ↪  enter)
        self.processed_name_lbl.setObjectName("processed_name⌟
        ↪  _lbl")
        self.processed_img_v_lay.addWidget(self.processed_nam⌟
        ↪  e_lbl)
        spacerItem4 = QtWidgets.QSpacerItem(20, 40,
        ↪  QtWidgets.QSizePolicy.Minimum,
        ↪  QtWidgets.QSizePolicy.Expanding)
        self.processed_img_v_lay.addItem(spacerItem4)
        self.processed_frame_lbl =
        ↪  QtWidgets.QLabel(self.centralwidget)
        self.processed_frame_lbl.setMinimumSize(QtCore.QSize(⌟
        ↪  400,
        ↪  240))
        self.processed_frame_lbl.setBaseSize(QtCore.QSize(0,
        ↪  0))
        self.processed_frame_lbl.setText("")
        self.processed_frame_lbl.setAlignment(QtCore.Qt.Align⌟
        ↪  Center)
        self.processed_frame_lbl.setObjectName("processed_fra⌟
        ↪  me_lbl")
        self.processed_img_v_lay.addWidget(self.processed_fra⌟
        ↪  me_lbl)
        spacerItem5 = QtWidgets.QSpacerItem(20, 40,
        ↪  QtWidgets.QSizePolicy.Minimum,
        ↪  QtWidgets.QSizePolicy.Expanding)
        self.processed_img_v_lay.addItem(spacerItem5)
        self.h_img_lay.addLayout(self.processed_img_v_lay)
        self.verticalLayout_4.addLayout(self.h_img_lay)
        self.horizontalLayout = QtWidgets.QHBoxLayout()
        self.horizontalLayout.setObjectName("horizontalLayout␣
        ↪  ")
        self.verticalLayout_2 = QtWidgets.QVBoxLayout()
        self.verticalLayout_2.setObjectName("verticalLayout_2␣
        ↪  ")
        self.label_3 = QtWidgets.QLabel(self.centralwidget)
        self.label_3.setAlignment(QtCore.Qt.AlignCenter)
        self.label_3.setObjectName("label_3")
        self.verticalLayout_2.addWidget(self.label_3)
        self.eye_lbl = QtWidgets.QLabel(self.centralwidget)
        self.eye_lbl.setMinimumSize(QtCore.QSize(400, 240))
        self.eye_lbl.setText("")
        self.eye_lbl.setAlignment(QtCore.Qt.AlignCenter)
        self.eye_lbl.setObjectName("eye_lbl")
        self.verticalLayout_2.addWidget(self.eye_lbl)
        self.horizontalLayout.addLayout(self.verticalLayout_2)
        self.verticalLayout = QtWidgets.QVBoxLayout()
        self.verticalLayout.setObjectName("verticalLayout")
        self.label_2 = QtWidgets.QLabel(self.centralwidget)
        self.label_2.setAlignment(QtCore.Qt.AlignCenter)
        self.label_2.setObjectName("label_2")
        self.verticalLayout.addWidget(self.label_2)
        self.voice_lbl = QtWidgets.QLabel(self.centralwidget)
        self.voice_lbl.setMinimumSize(QtCore.QSize(400, 240))
        self.voice_lbl.setText("")
        self.voice_lbl.setAlignment(QtCore.Qt.AlignCenter)
        self.voice_lbl.setObjectName("voice_lbl")
        self.verticalLayout.addWidget(self.voice_lbl)
        self.horizontalLayout.addLayout(self.verticalLayout)
        self.verticalLayout_4.addLayout(self.horizontalLayout)
        self.h_btn_lay = QtWidgets.QHBoxLayout()
        self.h_btn_lay.setObjectName("h_btn_lay")
        self.initButton =
        ↪  QtWidgets.QPushButton(self.centralwidget)
        self.initButton.setObjectName("initButton")
        self.h_btn_lay.addWidget(self.initButton)
        self.pushButton =
        ↪  QtWidgets.QPushButton(self.centralwidget)
        self.pushButton.setObjectName("pushButton")
        self.h_btn_lay.addWidget(self.pushButton)
        spacerItem6 = QtWidgets.QSpacerItem(40, 20,
        ↪  QtWidgets.QSizePolicy.Expanding,
        ↪  QtWidgets.QSizePolicy.Minimum)
        self.h_btn_lay.addItem(spacerItem6)
        self.label = QtWidgets.QLabel(self.centralwidget)
        self.label.setAlignment(QtCore.Qt.AlignRight|QtCore.Q⌋
        ↪  t.AlignTrailing|QtCore.Qt.AlignVCenter)
        self.label.setObjectName("label")
        self.h_btn_lay.addWidget(self.label)
        self.lineEdit =
        ↪  QtWidgets.QLineEdit(self.centralwidget)
        self.lineEdit.setObjectName("lineEdit")
        self.h_btn_lay.addWidget(self.lineEdit)
        self.startButton =
        ↪  QtWidgets.QPushButton(self.centralwidget)
        self.startButton.setObjectName("startButton")
        self.h_btn_lay.addWidget(self.startButton)
        self.endButton =
        ↪  QtWidgets.QPushButton(self.centralwidget)
        self.endButton.setObjectName("endButton")
        self.h_btn_lay.addWidget(self.endButton)
        self.shotButton =
        ↪  QtWidgets.QPushButton(self.centralwidget)
        self.shotButton.setObjectName("shotButton")
        self.h_btn_lay.addWidget(self.shotButton)
        self.verticalLayout_4.addLayout(self.h_btn_lay)
        self.v_filters_lay = QtWidgets.QVBoxLayout()
        self.v_filters_lay.setObjectName("v_filters_lay")
        self.verticalLayout_4.addLayout(self.v_filters_lay)
        MainWindow.setCentralWidget(self.centralwidget)
        self.menubar = QtWidgets.QMenuBar(MainWindow)
        self.menubar.setGeometry(QtCore.QRect(0, 0, 1035, 23))
        self.menubar.setObjectName("menubar")
        self.menuHelp = QtWidgets.QMenu(self.menubar)
        self.menuHelp.setObjectName("menuHelp")
        self.menu = QtWidgets.QMenu(self.menubar)
        self.menu.setObjectName("menu")
        self.menu_2 = QtWidgets.QMenu(self.menubar)
        self.menu_2.setObjectName("menu_2")
        MainWindow.setMenuBar(self.menubar)
        self.statusbar = QtWidgets.QStatusBar(MainWindow)
        self.statusbar.setMinimumSize(QtCore.QSize(500, 30))
        self.statusbar.setObjectName("statusbar")
        MainWindow.setStatusBar(self.statusbar)
        self.actionOpen_image = QtWidgets.QAction(MainWindow)
        self.actionOpen_image.setObjectName("actionOpen_image⌟
        ↪   ")
        self.actionSave_original_image =
        ↪   QtWidgets.QAction(MainWindow)
        self.actionSave_original_image.setObjectName("actionS⌟
        ↪   ave_original_image")
        self.actionSave_processed_image =
        ↪   QtWidgets.QAction(MainWindow)
        self.actionSave_processed_image.setObjectName("action⌟
        ↪   Save_processed_image")
        self.actionExit = QtWidgets.QAction(MainWindow)
        self.actionExit.setObjectName("actionExit")
        self.actionLicense = QtWidgets.QAction(MainWindow)
        self.actionLicense.setObjectName("actionLicense")
        self.actionAbout = QtWidgets.QAction(MainWindow)
        self.actionAbout.setObjectName("actionAbout")
        self.actionmanual = QtWidgets.QAction(MainWindow)
        self.actionmanual.setObjectName("actionmanual")
        self.action12 = QtWidgets.QAction(MainWindow)
        self.action12.setObjectName("action12")
        self.actionse = QtWidgets.QAction(MainWindow)
        self.actionse.setObjectName("actionse")
        self.menuHelp.addAction(self.actionmanual)
        self.menuHelp.addAction(self.actionAbout)
        self.menu_2.addAction(self.action12)
        self.menu_2.addAction(self.actionse)
        self.menubar.addAction(self.menu.menuAction())
        self.menubar.addAction(self.menu_2.menuAction())
        self.menubar.addAction(self.menuHelp.menuAction())
        self.retranslateUi(MainWindow)
        QtCore.QMetaObject.connectSlotsByName(MainWindow)
    def retranslateUi(self, MainWindow):
        _translate = QtCore.QCoreApplication.translate
        MainWindow.setWindowTitle(_translate("MainWindow", "肢体表型特征采集系统---山东省精神卫生中心"))
        self.original_name_lbl.setText(_translate("MainWindow", "原始图像"))
        self.processed_name_lbl.setText(_translate("MainWindow", "肢体特征"))
        self.label_3.setText(_translate("MainWindow", "标记图像"))
        self.label_2.setText(_translate("MainWindow", "肢体轮廓"))
        self.initButton.setText(_translate("MainWindow", "检测视频设备"))
        self.pushButton.setText(_translate("MainWindow", "停用视频设备"))
        self.label.setText(_translate("MainWindow", "样本编号:"))
        self.startButton.setText(_translate("MainWindow", "开始录制"))
        self.endButton.setText(_translate("MainWindow", "结束录制"))
        self.shotButton.setText(_translate("MainWindow", "进程中止"))
        self.menuHelp.setTitle(_translate("MainWindow", "帮助"))
        self.menu.setTitle(_translate("MainWindow", " 文件"))
        self.menu_2.setTitle(_translate("MainWindow", "设置"))
        self.actionOpen_image.setText(_translate("MainWindow", "Open image"))
        self.actionSave_original_image.setText(_translate("MainWindow", "Save original image"))
        self.actionSave_processed_image.setText(_translate("MainWindow", "Save processed image"))
        self.actionExit.setText(_translate("MainWindow", "Exit"))
        self.actionLicense.setText(_translate("MainWindow", "License"))
        self.actionAbout.setText(_translate("MainWindow", "关于"))
        self.actionmanual.setText(_translate("MainWindow", "操作说明"))
        self.action12.setText(_translate("MainWindow", "设备选择"))
        self.actionse.setText(_translate("MainWindow", "文件设置"))
```