HOCHSCHULE
RAVENSBURG-WEINGARTEN
UNIVERSITY
OF APPLIED SCIENCES

A Visual GUI Tool for Replacing Post-it Note in Image, Video and Live Stream with Customized Image

Ravensburg-Weingarten University

Electrical Engineering and Information Technology

Bachelor Project Report
Period: 01.07.2020 - 30.09.2020

Supervisor:
Prof. Dr. Markus Pfeil

Author:
Siyi Dai
Matrikel-Nr.:
29245

September 16, 2020

Contents

1. Introduction

Definition of Tasks

The project tasks are focused on developing graphical user interfaces (GUI) for replacing the post-it note area in image, video and live stream with loaded image using OpenCV-Python library.

GUI is a form of user interface that allows users to interact with electronic devices through graphical icons, instead of text-based user interfaces, typed command labels or text navigation.

OpenCV-Python is a library of Python bindings designed to solve computer vision problems. As an interdisciplinary scientific field, computer vision deals with how computers can gain high-level understanding from digital images or videos.

2. Environment Specification

The work environment is composed of two sides. The hardware environment which presents the machines or components being used while running code. The software environment which represents the advanced programming interfaces, integrated development environments, editors, technologies and tools being used.

Hardware Environment

The table describes the used computer for work while two BenQ Corporation 32" monitors were used.

PC	HP ZBook 15 G5 (3AX13AV)
CPU	Intel(R) Core(TM) i7-8850H CPU @ 2.60GHz
RAM	32GB
OS	Ubuntu 18.04.3 LTS
Graphics Card	NVIDIA Quadro P1000 Mobile

Table 2.1: Characteristics of used computer

Software Environment

Python 3.6.10 :: Anaconda, Inc. *Python* is an interpreted, high-level, general-purpose programming language. The internship tasks are all coded in *Python 3.6.10* within help of *Anaconda*, which is a *Python* and *R* distribution including the core python language, 100+ Python libraries and package manager *conda*.

Visual Studio Code *Visual Studio Code* is a free source-code editor made by *Microsoft* for *Windows*, *Linux*, *macOS*. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git.

PyQt5 *PyQt5* is a GUI develop application provided by *Python*. It is cross-platform GUI toolkit, a set of python bindings for *Qt v5*. An interactive desktop application can be developed with much easier because of the tools and simplicity provided by this library.

Modules and Classes: While the designation of Exporter and Visualizer, various modules and classes from *PyQt5* have been used.

Modules	Classes
Qt Core	*QTimer, Qt*
Qt gui	*QPixmap, QColor, QImage*
Qt Widgets	*QMainWindow, QColorDialog, QDialog*

Table 2.2: Modules and Classes being used for developing UI tools

Qt Creator *Qt Creator* includes a code editor and integrates *Qt Designer* for designing and building GUIs from *Qt widgets*. After layout designing and objects naming in *Qt Creator*, a form implementation could be generated from reading ui file by *PyQt5 UI code generator 5.9.2*.

OpenCV-Python *OpenCV-Python*is the Python API of OpenCV.

Functions: While the designation of post-it replacer, various functions from *cv2*, version 4.1.2, have been used.

Functions	Usages
cv2.VideoCapture()	get a video capture object for the camera or file path
cv2.boundingRect()	calculates and returns the minimal up-right bounding rectangle for the specified point set or non-zero pixels of gray-scale image
cv2.findContours()	retrieves contours from the binary image using the algorithm [Suzuki85]
cv2.CHAIN_APPROX_SIMPLE	compresses horizontal, vertical, and diagonal segments and leaves only their end points
cv2.RETR_TREE	retrieves all of the contours and reconstructs a full hierarchy of nested contours
cv2.cvtColor()	convert an image from one color space to another
cv2.COLOR_RGB2BGR	convert from RGB to BGR color spaces
cv2.COLOR_BGR2RGB	convert from BGR to RGB color spaces
cv2.inRange()	Checks if array elements lie between the elements of two other arrays

Table 2.3: Modules and Classes being used for developing UI tools

3. Approach and Method

Read Whether in picture mode, video mode or camera mode, the input stream needs to be read frame by frame, which means that it is resolved to several pictures, and detecting the target object in accordance with the input color range.

After the picture is read, it is stored in the form of a matrix, such as a 1280x720 color picture. If read according to RGB three channels, the picture becomes a 1280x720x3 matrix, and the three values at each position are the RGB channels values.

Mask The role of the mask is to limit the color range by setting the lower and upper limitation of RGB value. The colors between the lower and upper limits will be reserved as white, and the colors in the other ranges are all converted to black, so that a binary value with only black and white is obtained.

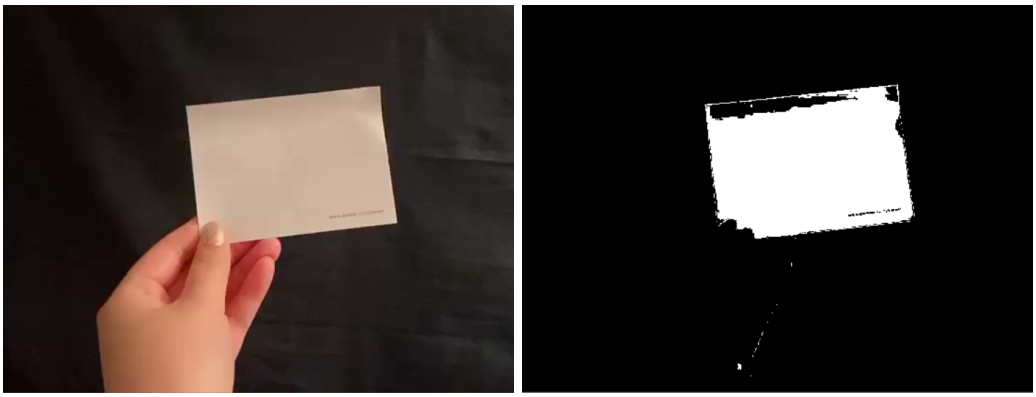

Figure 3.1: The mask obtained with the yellow post-it note in the video and the original image

Post-it note within the color range will be retained and the other colors will be ignored. This is also the reason that the color of post-it notes and the background color are required to be as different as possible.

Determine the Boundary After obtaining the mask, we need to determine the boundary of the white area in the mask. This process is implemented by the function *cv2.findContours()*.

This function will group all the continuous white pixels into one area and use the smallest rectangle bounding box to frame these discrete areas, which creates a lot of unconnected areas in the mask.

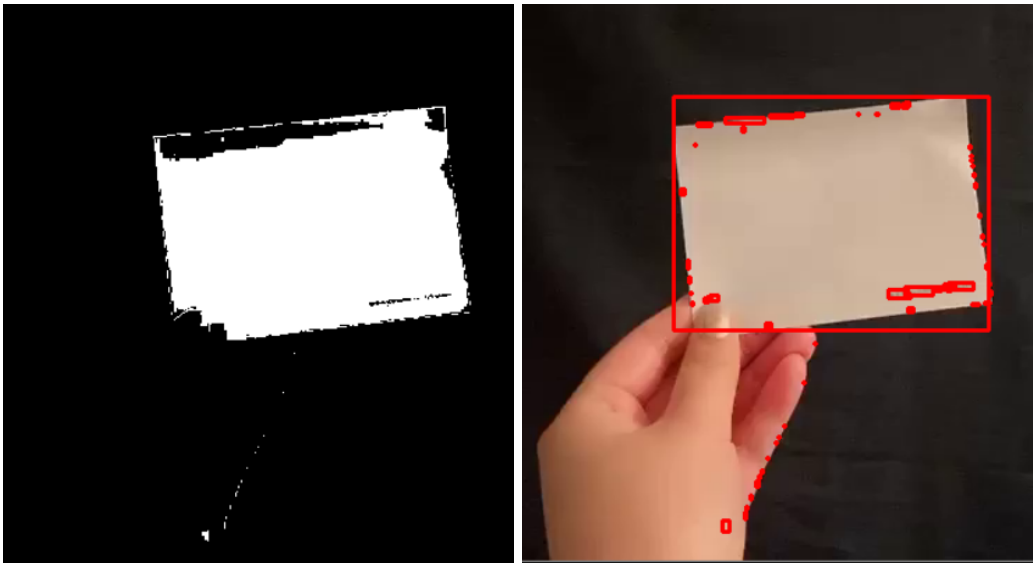

Figure 3.2: Determine the boundary of the white area in the mask with function *cv2.findContours()*

Many discrete rectangular areas will be detected. The function *cv2.findContours()* will return the coordinates of the upper left corner and the lower right corner of all rectangular boxes.

Find the Biggest Bounding Box we calculate the area of all rectangular boxes according to the width and height of the rectangular boxes, and choose the largest one.

```
147         try:
148             # find biggest bounding box
149             biggest, index = 0, 0
150             rect = list()
151             for i in range(len(contours)):
152                 x, y, w, h = cv2.boundingRect(contours[i])
153                 rect.append(((x, y), (x + w, y + h), (w, h)))
154                 if w * h > biggest:
155                     biggest = w * h
156                     index = i
```

Figure 3.3: The function used for finding the biggest bounding box among all white area

Transform For better visualization, the replacement area will be half-sized to the middle of the post-it, which is approached by the following function.

```
158             # get the bounding box in captured image
159             height = rect[index][1][1]- rect[index][0][1]
160             width = rect[index][1][0] - rect[index][0][0]
161             trans_pt1 = (int(rect[index][0][0]+width/4), int(rect[index][0][1]+height/4))
162             trans_pt2 = (int(rect[index][1][0]-width/4), int(rect[index][1][1]-height/4))
```

Figure 3.4: The function used for finding the half-sized bounding box in the middle of the post-it

Resize and Replace After obtaining the width and height of the largest rectangular box, that is, w and h, the selected replacement image will be read in RGB format and resize it into a matrix of (w/2, h/2, 3), replacing the pixels in the largest rectangular box in the current frame.

Since the detected rectangle is a rectangle, the picture can only be replaced in the positive direction, and the oblique post-it note will replace the smallest bounding rectangle, which is also in the positive direction.

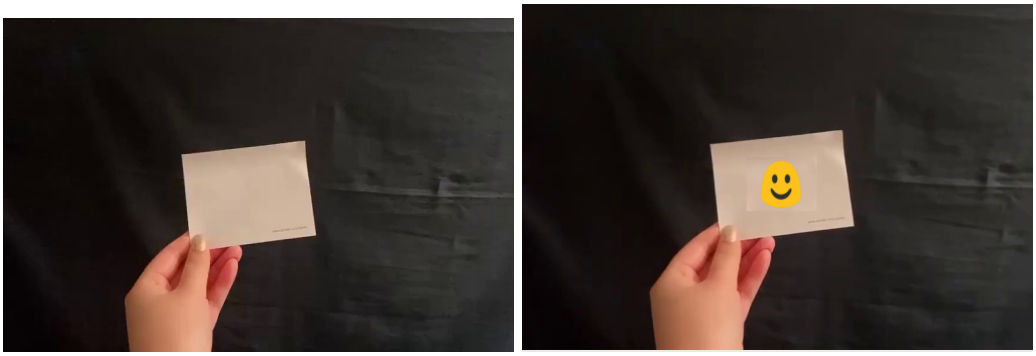

Figure 3.5: The function used for finding the half-sized bounding box in the middle of the post-it

4. Post-it Replacer

Project Structure

A graphical user interfaces (GUI) for replacing the post-it note area in image, video and live stream with loaded image using OpenCV-Python library has been successfully developed in this project.

The project folder is divided as:

- **dialogs** folder for storing IO dialog related scripts
- **ui and ui_py** folders for storing the *.ui* file and the *.py* file generated from reading *.ui* file by *PyQt5 UI code generator 5.9.2*
- **windows** folder for storing mainwindow related script
- **post_it_replace.py** executable file for starting the GUI

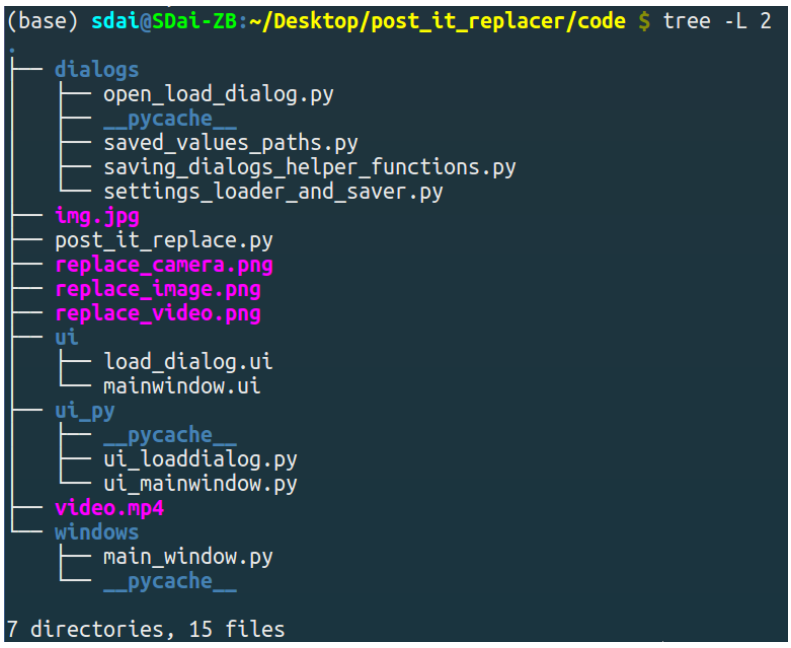

Figure 4.1: The project structure in level 2

Dialogs

With an IO dialog, the user can conveniently select the original image or video with post-it to be replaced with as well as the replace image.

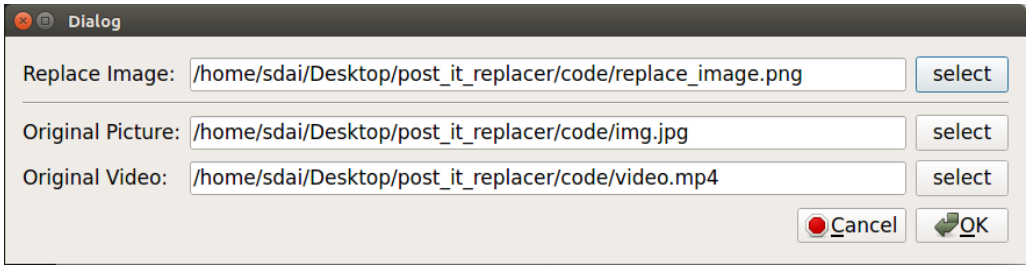

Figure 4.2: Dialog for loading replace image, original picture and original video

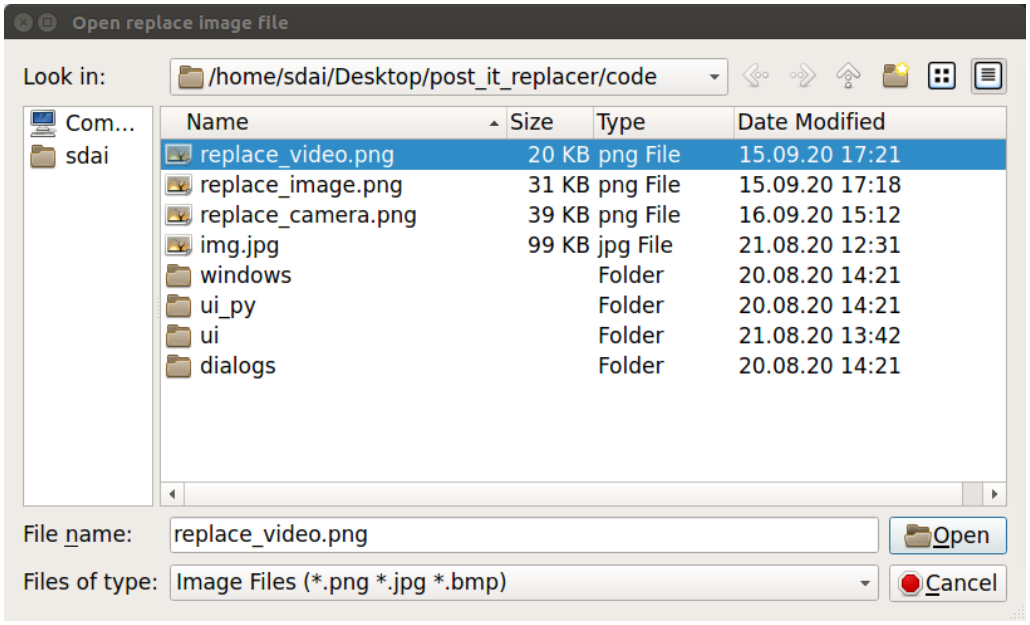

Figure 4.3: Select image or video from system file path

Code

- open_load_dialog.py
- saved_values_paths.py
- saving_dialogs_helper_functions.py
- settings_loader_and_saver.py
- load_dialog.ui
- ui_loaddialog.py

```python
#open_load_dialog.py

from PyQt5.QtWidgets import QDialog
from ui_py.ui_loaddialog import Ui_LoadDialog

from dialogs.saved_values_paths import SavedValuesConstants
from dialogs.settings_loader_and_saver import SettingsLoaderAndSaver
from dialogs.saving_dialogs_helper_functions import (
    load_value_and_initialize_field,
    start_file_dialog,
)
import os


class OpenLoadDialog(QDialog):
    def __init__(self):
        super(OpenLoadDialog, self).__init__()
        self.ui = Ui_LoadDialog()
        self.ui.setupUi(self)

        self.settings_loader = SettingsLoaderAndSaver(SavedValuesConstants.LoaderDialog.SETTING_NAME)

        self.__initialize_fields_and_values()

    def __initialize_fields_and_values(self):

        load_value_and_initialize_field(

self.settings_loader.read(SavedValuesConstants.LoaderDialog.REPLACE_IMAGE_PATH),
            self.ui.replace_image_line_edit,
            self.ui.select_replace_image_button.pressed,
            self.on_select_replace_image_button_down,
        )
        load_value_and_initialize_field(

self.settings_loader.read(SavedValuesConstants.LoaderDialog.ORIGINAL_PICTURE_PATH),
            self.ui.original_picture_line_edit,
            self.ui.select_original_picture_button.pressed,
            self.on_select_original_picture_button_down,
        )
        load_value_and_initialize_field(

self.settings_loader.read(SavedValuesConstants.LoaderDialog.ORGINAL_VIDEO_PATH),
            self.ui.original_video_line_edit,
            self.ui.select_original_video_button.pressed,
            self.on_select_original_video_button_down,
        )

    def on_select_replace_image_button_down(self):
        start_file_dialog(
            self,
            "Open replace image file",
            "/home",
            self.replace_image_file_path(),
            "Image Files (*.png *.jpg *.bmp)",
            self.ui.replace_image_line_edit,
        )

    def on_select_original_picture_button_down(self):
        start_file_dialog(
```

```
58              self,
59              "Open original picture file",
60              "/home",
61              self.original_picture_file_path(),
62              "Image Files (*.png *.jpg *.bmp)",
63              self.ui.original_picture_line_edit,
64          )
65
66      def on_select_original_video_button_down(self):
67          start_file_dialog(
68              self,
69              "Open original video file",
70              "/home",
71              self.original_picture_file_path(),
72              "Video files (*.mp4 *.mwv *.mov)",
73              self.ui.original_video_line_edit,
74          )
75
76      def replace_image_file_path(self):
77          return self.ui.replace_image_line_edit.text()
78
79      def original_picture_file_path(self):
80          return self.ui.original_picture_line_edit.text()
81
82      def original_video_file_path(self):
83          return self.ui.original_video_line_edit.text()
84
85      def save_values(self):
86          self.settings_loader.write(
87              SavedValuesConstants.LoaderDialog.REPLACE_IMAGE_PATH,
    self.replace_image_file_path(),
88          )
89          self.settings_loader.write(
90              SavedValuesConstants.LoaderDialog.ORIGINAL_PICTURE_PATH,
    self.original_picture_file_path(),
91          )
92          self.settings_loader.write(
93              SavedValuesConstants.LoaderDialog.ORGINAL_VIDEO_PATH,
    self.original_video_file_path(),
94          )
95
96      def exec_(self):
97          if not super(OpenLoadDialog, self).exec_():
98              return False
99
100         self.save_values()
101         return True
102
```

```python
#saved_values_paths.py

from PyQt5.QtGui import QColor


class SavedValuesConstants:
    ORGANIZATION = "RWU - EI"
    APPLICATION = "Post-it Replacer"

    class LoaderDialog:
        SETTING_NAME = "Load dialog"
        REPLACE_IMAGE_PATH = "replace image path"
        ORIGINAL_PICTURE_PATH = "original picture path"
        ORGINAL_VIDEO_PATH = "original video path"

    class SettingsColorPicker:
        CUSTOMIZED_COLOR_POST_IT = QColor(209, 181, 162)
        CUSTOMIZED_COLOR_POST_IT_UPPER = QColor(189, 161, 144)
        CUSTOMIZED_COLOR_POST_IT_LOWER = QColor(229, 201, 182)

```

```python
#saving_dialogs_helper_functions.py

from PyQt5.QtWidgets import QFileDialog


def load_value_and_initialize_field(default_value, text_target, action,
result_function):
    if default_value is not None:
        text_target.setText(default_value)

    action.connect(result_function)


def start_file_dialog(who, caption, default_value, loaded_default_value, file_types,
text_line):
    file_path = QFileDialog.getOpenFileName(
        who, caption, default_value if not loaded_default_value else
loaded_default_value, file_types,
    )
    text_line.setText(file_path[0])

```

```python
#settings_loader_and_saver.py

from dialogs.saved_values_paths import SavedValuesConstants
from PyQt5.QtCore import QSettings

SPECIAL_SAVER_SEPARATOR = " - "


class SettingsLoaderAndSaver:
    def __init__(
        self, name, organization=SavedValuesConstants.ORGANIZATION,
application=SavedValuesConstants.APPLICATION,
    ):
        self.organization = organization
        self.application = application
        self.name = name

    def read(self, path):
        settings_reader = QSettings(self.organization, self.application)
        return settings_reader.value(self.name + SPECIAL_SAVER_SEPARATOR + path)

    def write(self, path, value):
        settings_writer = QSettings(self.organization, self.application)
        settings_writer.setValue(self.name + SPECIAL_SAVER_SEPARATOR + path, value)

```

```
#load_dialog.ui

<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>LoadDialog</class>
 <widget class="QDialog" name="LoadDialog">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>712</width>
    <height>157</height>
   </rect>
  </property>
  <property name="sizePolicy">
   <sizepolicy hsizetype="Preferred" vsizetype="Preferred">
    <horstretch>0</horstretch>
    <verstretch>0</verstretch>
   </sizepolicy>
  </property>
  <property name="windowTitle">
   <string>Dialog</string>
  </property>
  <layout class="QGridLayout" name="gridLayout_2">
   <item row="1" column="0">
    <widget class="QDialogButtonBox" name="buttonBox">
     <property name="orientation">
      <enum>Qt::Horizontal</enum>
     </property>
     <property name="standardButtons">
      <set>QDialogButtonBox::Cancel|QDialogButtonBox::Ok</set>
     </property>
    </widget>
   </item>
   <item row="0" column="0">
    <layout class="QVBoxLayout" name="verticalLayout">
     <item>
      <layout class="QGridLayout" name="gridLayout">
       <item row="3" column="2">
        <widget class="QPushButton" name="select_original_video_button">
         <property name="text">
          <string>select</string>
         </property>
        </widget>
       </item>
       <item row="2" column="1">
        <widget class="QLineEdit" name="original_picture_line_edit">
         <property name="text">
          <string/>
         </property>
        </widget>
       </item>
       <item row="0" column="1">
        <widget class="QLineEdit" name="replace_image_line_edit"/>
       </item>
       <item row="2" column="2">
        <widget class="QPushButton" name="select_original_picture_button">
         <property name="text">
          <string>select</string>
         </property>
        </widget>
```

```
      </item>
      <item row="2" column="0">
       <widget class="QLabel" name="label_2">
        <property name="text">
         <string>Original Picture:</string>
        </property>
       </widget>
      </item>
      <item row="0" column="2">
       <widget class="QPushButton" name="select_replace_image_button">
        <property name="text">
         <string>select</string>
        </property>
       </widget>
      </item>
      <item row="3" column="1">
       <widget class="QLineEdit" name="original_video_line_edit"/>
      </item>
      <item row="1" column="0" colspan="3">
       <widget class="Line" name="line_2">
        <property name="orientation">
         <enum>Qt::Horizontal</enum>
        </property>
       </widget>
      </item>
      <item row="3" column="0">
       <widget class="QLabel" name="label_8">
        <property name="text">
         <string>Original Video:</string>
        </property>
       </widget>
      </item>
      <item row="0" column="0">
       <widget class="QLabel" name="label_5">
        <property name="text">
         <string>Replace Image:</string>
        </property>
       </widget>
      </item>
     </layout>
    </item>
   </layout>
  </item>
 </layout>
</widget>
<resources/>
<connections>
 <connection>
  <sender>buttonBox</sender>
  <signal>accepted()</signal>
  <receiver>LoadDialog</receiver>
  <slot>accept()</slot>
  <hints>
   <hint type="sourcelabel">
    <x>248</x>
    <y>254</y>
   </hint>
   <hint type="destinationlabel">
    <x>157</x>
    <y>274</y>
   </hint>
```

```
  </hints>
 </connection>
 <connection>
  <sender>buttonBox</sender>
  <signal>rejected()</signal>
  <receiver>LoadDialog</receiver>
  <slot>reject()</slot>
  <hints>
   <hint type="sourcelabel">
    <x>316</x>
    <y>260</y>
   </hint>
   <hint type="destinationlabel">
    <x>286</x>
    <y>274</y>
   </hint>
  </hints>
 </connection>
</connections>
</ui>

```

```python
#ui_loaddialog.py

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

# Form implementation generated from reading ui file 'ui/load_dialog.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_LoadDialog(object):
    def setupUi(self, LoadDialog):
        LoadDialog.setObjectName("LoadDialog")
        LoadDialog.resize(712, 157)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Preferred, QtWidgets.QSizePolicy.Preferred)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(LoadDialog.sizePolicy().hasHeightForWidth())
        LoadDialog.setSizePolicy(sizePolicy)
        self.gridLayout_2 = QtWidgets.QGridLayout(LoadDialog)
        self.gridLayout_2.setObjectName("gridLayout_2")
        self.buttonBox = QtWidgets.QDialogButtonBox(LoadDialog)
        self.buttonBox.setOrientation(QtCore.Qt.Horizontal)
        self.buttonBox.setStandardButtons(QtWidgets.QDialogButtonBox.Cancel | QtWidgets.QDialogButtonBox.Ok)
        self.buttonBox.setObjectName("buttonBox")
        self.gridLayout_2.addWidget(self.buttonBox, 1, 0, 1, 1)
        self.verticalLayout = QtWidgets.QVBoxLayout()
        self.verticalLayout.setObjectName("verticalLayout")
        self.gridLayout = QtWidgets.QGridLayout()
        self.gridLayout.setObjectName("gridLayout")
        self.select_original_video_button = QtWidgets.QPushButton(LoadDialog)

self.select_original_video_button.setObjectName("select_original_video_button")
        self.gridLayout.addWidget(self.select_original_video_button, 3, 2, 1, 1)
        self.original_picture_line_edit = QtWidgets.QLineEdit(LoadDialog)
        self.original_picture_line_edit.setText("")
        self.original_picture_line_edit.setObjectName("original_picture_line_edit")
        self.gridLayout.addWidget(self.original_picture_line_edit, 2, 1, 1, 1)
        self.replace_image_line_edit = QtWidgets.QLineEdit(LoadDialog)
        self.replace_image_line_edit.setObjectName("replace_image_line_edit")
        self.gridLayout.addWidget(self.replace_image_line_edit, 0, 1, 1, 1)
        self.select_original_picture_button = QtWidgets.QPushButton(LoadDialog)

self.select_original_picture_button.setObjectName("select_original_picture_button")
        self.gridLayout.addWidget(self.select_original_picture_button, 2, 2, 1, 1)
        self.label_2 = QtWidgets.QLabel(LoadDialog)
        self.label_2.setObjectName("label_2")
        self.gridLayout.addWidget(self.label_2, 2, 0, 1, 1)
        self.select_replace_image_button = QtWidgets.QPushButton(LoadDialog)

self.select_replace_image_button.setObjectName("select_replace_image_button")
        self.gridLayout.addWidget(self.select_replace_image_button, 0, 2, 1, 1)
        self.original_video_line_edit = QtWidgets.QLineEdit(LoadDialog)
        self.original_video_line_edit.setObjectName("original_video_line_edit")
        self.gridLayout.addWidget(self.original_video_line_edit, 3, 1, 1, 1)
        self.line_2 = QtWidgets.QFrame(LoadDialog)
        self.line_2.setFrameShape(QtWidgets.QFrame.HLine)
```

```
        self.line_2.setFrameShadow(QtWidgets.QFrame.Sunken)
        self.line_2.setObjectName("line_2")
        self.gridLayout.addWidget(self.line_2, 1, 0, 1, 3)
        self.label_8 = QtWidgets.QLabel(LoadDialog)
        self.label_8.setObjectName("label_8")
        self.gridLayout.addWidget(self.label_8, 3, 0, 1, 1)
        self.label_5 = QtWidgets.QLabel(LoadDialog)
        self.label_5.setObjectName("label_5")
        self.gridLayout.addWidget(self.label_5, 0, 0, 1, 1)
        self.verticalLayout.addLayout(self.gridLayout)
        self.gridLayout_2.addLayout(self.verticalLayout, 0, 0, 1, 1)

        self.retranslateUi(LoadDialog)
        self.buttonBox.accepted.connect(LoadDialog.accept)
        self.buttonBox.rejected.connect(LoadDialog.reject)
        QtCore.QMetaObject.connectSlotsByName(LoadDialog)

    def retranslateUi(self, LoadDialog):
        _translate = QtCore.QCoreApplication.translate
        LoadDialog.setWindowTitle(_translate("LoadDialog", "Dialog"))
        self.select_original_video_button.setText(_translate("LoadDialog",
"select"))
        self.select_original_picture_button.setText(_translate("LoadDialog",
"select"))
        self.label_2.setText(_translate("LoadDialog", "Original Picture:"))
        self.select_replace_image_button.setText(_translate("LoadDialog", "select"))
        self.label_8.setText(_translate("LoadDialog", "Original Video:"))
        self.label_5.setText(_translate("LoadDialog", "Replace Image:"))


if __name__ == "__main__":
    import sys

    app = QtWidgets.QApplication(sys.argv)
    LoadDialog = QtWidgets.QDialog()
    ui = Ui_LoadDialog()
    ui.setupUi(LoadDialog)
    LoadDialog.show()
    sys.exit(app.exec_())

```

Color Picker

A color picker has been added with the main window of GUI for user to pick the exact color of post-it note to replace in order to have more precise detection.

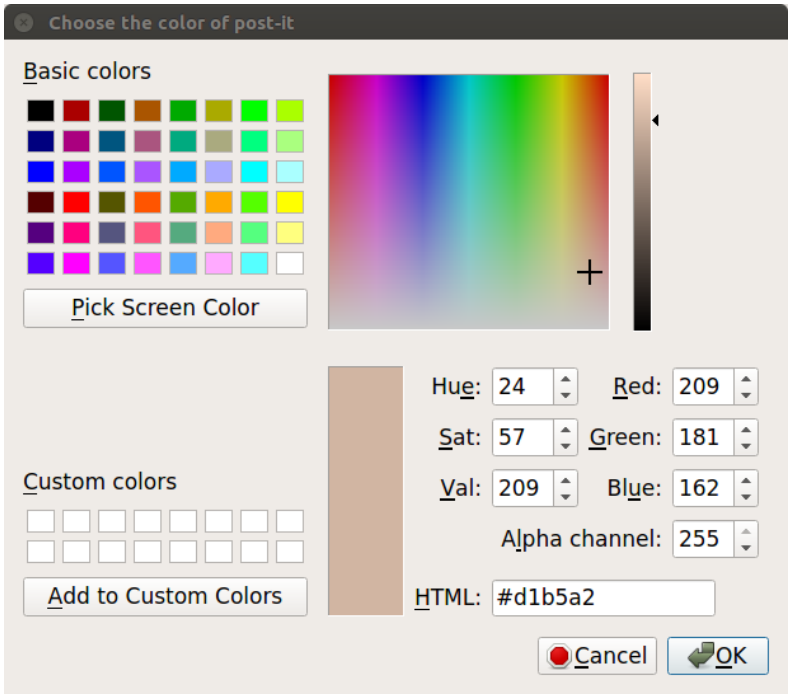

Figure 4.4: A color picker for more precise detection of the color of post-it

MainWindow

The main window of post-it replacer has three groups: replacement result groupbox, replace image groupbox and original source groupbox, which aimed at showing each part of the resources and the replacement result at the same time.

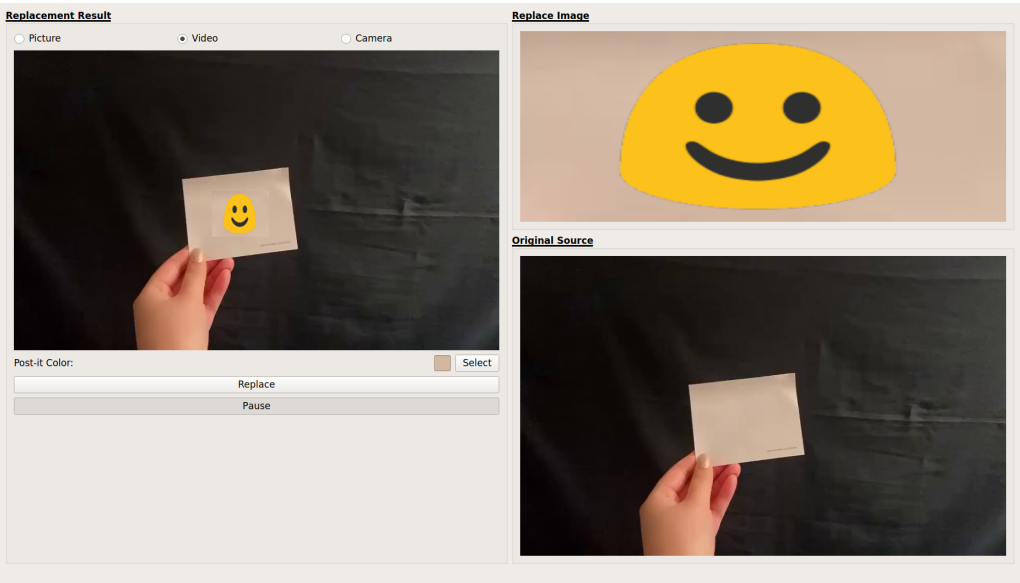

Figure 4.5: Main window showing original source, replace image and replacement result at the same time in the video mode

Code

- mainwindow.ui
- ui_mainwindow.py
- main_window.py
- post_it_replace.py

```
#mainwindow.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>904</width>
    <height>629</height>
   </rect>
  </property>
  <property name="sizePolicy">
   <sizepolicy hsizetype="MinimumExpanding" vsizetype="MinimumExpanding">
    <horstretch>0</horstretch>
    <verstretch>0</verstretch>
   </sizepolicy>
  </property>
  <property name="windowTitle">
   <string>Post-it Replace</string>
  </property>
  <widget class="QWidget" name="centralwidget">
   <layout class="QGridLayout" name="gridLayout_10">
    <property name="sizeConstraint">
     <enum>QLayout::SetNoConstraint</enum>
    </property>
    <item row="0" column="1">
     <layout class="QVBoxLayout" name="verticalLayout_5">
      <item>
       <layout class="QHBoxLayout" name="horizontalLayout_3">
        <property name="sizeConstraint">
         <enum>QLayout::SetNoConstraint</enum>
        </property>
        <item>
         <widget class="QGroupBox" name="replace_image_groupbox">
          <property name="sizePolicy">
           <sizepolicy hsizetype="Expanding" vsizetype="Expanding">
            <horstretch>0</horstretch>
            <verstretch>0</verstretch>
           </sizepolicy>
          </property>
          <property name="font">
           <font>
            <weight>75</weight>
            <bold>true</bold>
            <underline>true</underline>
           </font>
          </property>
          <property name="title">
           <string>Replace Image</string>
          </property>
          <layout class="QGridLayout" name="gridLayout_2">
           <item row="0" column="0">
            <widget class="QLabel" name="replace_image_label">
             <property name="sizePolicy">
              <sizepolicy hsizetype="Expanding" vsizetype="Expanding">
               <horstretch>0</horstretch>
               <verstretch>0</verstretch>
              </sizepolicy>
```

```
            </property>
            <property name="minimumSize">
             <size>
              <width>420</width>
              <height>240</height>
             </size>
            </property>
            <property name="layoutDirection">
             <enum>Qt::LeftToRight</enum>
            </property>
            <property name="text">
             <string/>
            </property>
            <property name="scaledContents">
             <bool>true</bool>
            </property>
           </widget>
          </item>
         </layout>
        </widget>
       </item>
      </layout>
     </item>
     <item>
      <widget class="QGroupBox" name="original_source_groupbox">
       <property name="font">
        <font>
         <weight>75</weight>
         <bold>true</bold>
         <underline>true</underline>
        </font>
       </property>
       <property name="title">
        <string>Original Source</string>
       </property>
       <layout class="QGridLayout" name="gridLayout_11">
        <item row="0" column="0">
         <layout class="QVBoxLayout" name="verticalLayout_4">
          <item>
           <widget class="QLabel" name="original_source_label">
            <property name="sizePolicy">
             <sizepolicy hsizetype="Expanding" vsizetype="Expanding">
              <horstretch>0</horstretch>
              <verstretch>0</verstretch>
             </sizepolicy>
            </property>
            <property name="minimumSize">
             <size>
              <width>400</width>
              <height>240</height>
             </size>
            </property>
            <property name="text">
             <string/>
            </property>
            <property name="scaledContents">
             <bool>true</bool>
            </property>
           </widget>
          </item>
         </layout>
```

```
        </item>
       </layout>
      </widget>
     </item>
    </layout>
   </item>
   <item row="0" column="0">
    <widget class="QGroupBox" name="replacement_result_groupbox">
     <property name="sizePolicy">
      <sizepolicy hsizetype="Preferred" vsizetype="Expanding">
       <horstretch>0</horstretch>
       <verstretch>0</verstretch>
      </sizepolicy>
     </property>
     <property name="font">
      <font>
       <weight>75</weight>
       <bold>true</bold>
       <underline>true</underline>
      </font>
     </property>
     <property name="title">
      <string>Replacement Result</string>
     </property>
     <property name="alignment">
      <set>Qt::AlignLeading|Qt::AlignLeft|Qt::AlignTop</set>
     </property>
     <layout class="QGridLayout" name="gridLayout_8">
      <item row="2" column="0">
       <widget class="QPushButton" name="pushButton_pause">
        <property name="font">
         <font>
          <weight>50</weight>
          <bold>false</bold>
          <underline>false</underline>
          <kerning>false</kerning>
         </font>
        </property>
        <property name="text">
         <string>Pause</string>
        </property>
        <property name="checkable">
         <bool>true</bool>
        </property>
       </widget>
      </item>
      <item row="1" column="0">
       <widget class="QPushButton" name="pushButton_replace">
        <property name="font">
         <font>
          <weight>50</weight>
          <bold>false</bold>
          <underline>false</underline>
         </font>
        </property>
        <property name="text">
         <string>Replace</string>
        </property>
        <property name="checkable">
         <bool>false</bool>
        </property>
```

```
        </widget>
       </item>
       <item row="0" column="0">
        <layout class="QVBoxLayout" name="verticalLayout_7">
         <item>
          <layout class="QHBoxLayout" name="horizontalLayout_2">
           <item>
            <widget class="QRadioButton" name="radioButton_picture">
             <property name="font">
              <font>
               <weight>50</weight>
               <bold>false</bold>
               <underline>false</underline>
              </font>
             </property>
             <property name="text">
              <string>Picture</string>
             </property>
             <property name="checked">
              <bool>true</bool>
             </property>
            </widget>
           </item>
           <item>
            <widget class="QRadioButton" name="radioButton_video">
             <property name="font">
              <font>
               <weight>50</weight>
               <bold>false</bold>
               <underline>false</underline>
              </font>
             </property>
             <property name="text">
              <string>Video</string>
             </property>
            </widget>
           </item>
           <item>
            <widget class="QRadioButton" name="radioButton_camera">
             <property name="sizePolicy">
              <sizepolicy hsizetype="Minimum" vsizetype="Fixed">
               <horstretch>0</horstretch>
               <verstretch>0</verstretch>
              </sizepolicy>
             </property>
             <property name="font">
              <font>
               <weight>50</weight>
               <bold>false</bold>
               <underline>false</underline>
              </font>
             </property>
             <property name="text">
              <string>Camera</string>
             </property>
            </widget>
           </item>
          </layout>
         </item>
         <item>
          <layout class="QGridLayout" name="gridLayout_9">
```

```
            <item row="3" column="0">
             <widget class="QLabel" name="replacement_result_label">
              <property name="sizePolicy">
               <sizepolicy hsizetype="Expanding" vsizetype="Expanding">
                <horstretch>0</horstretch>
                <verstretch>0</verstretch>
               </sizepolicy>
              </property>
              <property name="minimumSize">
               <size>
                <width>400</width>
                <height>240</height>
               </size>
              </property>
              <property name="text">
               <string/>
              </property>
              <property name="scaledContents">
               <bool>true</bool>
              </property>
             </widget>
            </item>
            <item row="4" column="0">
             <layout class="QHBoxLayout" name="horizontalLayout_4">
              <item>
               <widget class="QLabel" name="post_it_color_label">
                <property name="sizePolicy">
                 <sizepolicy hsizetype="Minimum" vsizetype="Fixed">
                  <horstretch>0</horstretch>
                  <verstretch>0</verstretch>
                 </sizepolicy>
                </property>
                <property name="font">
                 <font>
                  <weight>50</weight>
                  <bold>false</bold>
                  <underline>false</underline>
                 </font>
                </property>
                <property name="text">
                 <string>Post-it Color:</string>
                </property>
               </widget>
              </item>
              <item>
               <spacer name="horizontalSpacer">
                <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>
               <widget class="QLineEdit" name="post_it_lineEdit">
                <property name="sizePolicy">
                 <sizepolicy hsizetype="Fixed" vsizetype="Fixed">
```

```
                <horstretch>0</horstretch>
                <verstretch>0</verstretch>
               </sizepolicy>
              </property>
              <property name="maximumSize">
               <size>
                <width>30</width>
                <height>16777215</height>
               </size>
              </property>
             </widget>
            </item>
            <item>
             <widget class="QPushButton" name="post_it_color_pushButton">
              <property name="sizePolicy">
               <sizepolicy hsizetype="Fixed" vsizetype="Fixed">
                <horstretch>0</horstretch>
                <verstretch>0</verstretch>
               </sizepolicy>
              </property>
              <property name="font">
               <font>
                <weight>50</weight>
                <bold>false</bold>
                <underline>false</underline>
               </font>
              </property>
              <property name="text">
               <string>Select</string>
              </property>
             </widget>
            </item>
           </layout>
          </item>
         </layout>
        </item>
       </layout>
      </item>
      <item row="3" column="0">
       <spacer name="verticalSpacer">
        <property name="orientation">
         <enum>Qt::Vertical</enum>
        </property>
        <property name="sizeType">
         <enum>QSizePolicy::MinimumExpanding</enum>
        </property>
        <property name="sizeHint" stdset="0">
         <size>
          <width>20</width>
          <height>40</height>
         </size>
        </property>
       </spacer>
      </item>
     </layout>
    </widget>
   </item>
  </layout>
 </widget>
 <widget class="QMenuBar" name="menubar">
  <property name="geometry">
```

```
    <rect>
     <x>0</x>
     <y>0</y>
     <width>904</width>
     <height>25</height>
    </rect>
   </property>
   <widget class="QMenu" name="menuMain">
    <property name="title">
     <string>File</string>
    </property>
    <addaction name="separator"/>
    <addaction name="actionLoader"/>
   </widget>
   <addaction name="menuMain"/>
  </widget>
  <widget class="QStatusBar" name="statusbar"/>
  <action name="actionLoader">
   <property name="text">
    <string>Load...</string>
   </property>
  </action>
 </widget>
 <resources/>
 <connections/>
</ui>

```

```python
#ui_mainwindow.py

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

# Form implementation generated from reading ui file 'ui/mainwindow.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(904, 629)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.MinimumExpanding, QtWidgets.QSizePolicy.MinimumExpanding)
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(MainWindow.sizePolicy().hasHeightForWidth())
        MainWindow.setSizePolicy(sizePolicy)
        self.centralwidget = QtWidgets.QWidget(MainWindow)
        self.centralwidget.setObjectName("centralwidget")
        self.gridLayout_10 = QtWidgets.QGridLayout(self.centralwidget)
        self.gridLayout_10.setSizeConstraint(QtWidgets.QLayout.SetNoConstraint)
        self.gridLayout_10.setObjectName("gridLayout_10")
        self.verticalLayout_5 = QtWidgets.QVBoxLayout()
        self.verticalLayout_5.setObjectName("verticalLayout_5")
        self.horizontalLayout_3 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_3.setSizeConstraint(QtWidgets.QLayout.SetNoConstraint)
        self.horizontalLayout_3.setObjectName("horizontalLayout_3")
        self.replace_image_groupbox = QtWidgets.QGroupBox(self.centralwidget)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Expanding, QtWidgets
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.replace_image_groupbox.sizePolicy().hasHeig
        self.replace_image_groupbox.setSizePolicy(sizePolicy)
        font = QtGui.QFont()
        font.setBold(True)
        font.setUnderline(True)
        font.setWeight(75)
        self.replace_image_groupbox.setFont(font)
        self.replace_image_groupbox.setObjectName("replace_image_groupbox")
        self.gridLayout_2 = QtWidgets.QGridLayout(self.replace_image_groupbox)
        self.gridLayout_2.setObjectName("gridLayout_2")
        self.replace_image_label = QtWidgets.QLabel(self.replace_image_groupbox)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Expanding, QtWidgets
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.replace_image_label.sizePolicy().hasHeightF
        self.replace_image_label.setSizePolicy(sizePolicy)
        self.replace_image_label.setMinimumSize(QtCore.QSize(420, 240))
        self.replace_image_label.setLayoutDirection(QtCore.Qt.LeftToRight)
        self.replace_image_label.setText("")
        self.replace_image_label.setScaledContents(True)
        self.replace_image_label.setObjectName("replace_image_label")
        self.gridLayout_2.addWidget(self.replace_image_label, 0, 0, 1, 1)
        self.horizontalLayout_3.addWidget(self.replace_image_groupbox)
        self.verticalLayout_5.addLayout(self.horizontalLayout_3)
        self.original_source_groupbox = QtWidgets.QGroupBox(self.centralwidget)
```

```
        font = QtGui.QFont()
        font.setBold(True)
        font.setUnderline(True)
        font.setWeight(75)
        self.original_source_groupbox.setFont(font)
        self.original_source_groupbox.setObjectName("original_source_groupbox")
        self.gridLayout_11 = QtWidgets.QGridLayout(self.original_source_groupbox)
        self.gridLayout_11.setObjectName("gridLayout_11")
        self.verticalLayout_4 = QtWidgets.QVBoxLayout()
        self.verticalLayout_4.setObjectName("verticalLayout_4")
        self.original_source_label = QtWidgets.QLabel(self.original_source_groupbox)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Expanding, QtWidgets
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.original_source_label.sizePolicy().hasHeigh
        self.original_source_label.setSizePolicy(sizePolicy)
        self.original_source_label.setMinimumSize(QtCore.QSize(400, 240))
        self.original_source_label.setText("")
        self.original_source_label.setScaledContents(True)
        self.original_source_label.setObjectName("original_source_label")
        self.verticalLayout_4.addWidget(self.original_source_label)
        self.gridLayout_11.addLayout(self.verticalLayout_4, 0, 0, 1, 1)
        self.verticalLayout_5.addWidget(self.original_source_groupbox)
        self.gridLayout_10.addLayout(self.verticalLayout_5, 0, 1, 1, 1)
        self.replacement_result_groupbox = QtWidgets.QGroupBox(self.centralwidget)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Preferred, QtWidgets
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.replacement_result_groupbox.sizePolicy().ha
        self.replacement_result_groupbox.setSizePolicy(sizePolicy)
        font = QtGui.QFont()
        font.setBold(True)
        font.setUnderline(True)
        font.setWeight(75)
        self.replacement_result_groupbox.setFont(font)

self.replacement_result_groupbox.setAlignment(QtCore.Qt.AlignLeading|QtCore.Qt.AlignL
        self.replacement_result_groupbox.setObjectName("replacement_result_groupbox")
        self.gridLayout_8 = QtWidgets.QGridLayout(self.replacement_result_groupbox)
        self.gridLayout_8.setObjectName("gridLayout_8")
        self.pushButton_pause = QtWidgets.QPushButton(self.replacement_result_groupbo
        font = QtGui.QFont()
        font.setBold(False)
        font.setUnderline(False)
        font.setWeight(50)
        font.setKerning(False)
        self.pushButton_pause.setFont(font)
        self.pushButton_pause.setCheckable(True)
        self.pushButton_pause.setObjectName("pushButton_pause")
        self.gridLayout_8.addWidget(self.pushButton_pause, 2, 0, 1, 1)
        self.pushButton_replace = QtWidgets.QPushButton(self.replacement_result_group
        font = QtGui.QFont()
        font.setBold(False)
        font.setUnderline(False)
        font.setWeight(50)
        self.pushButton_replace.setFont(font)
        self.pushButton_replace.setCheckable(False)
        self.pushButton_replace.setObjectName("pushButton_replace")
        self.gridLayout_8.addWidget(self.pushButton_replace, 1, 0, 1, 1)
        self.verticalLayout_7 = QtWidgets.QVBoxLayout()
        self.verticalLayout_7.setObjectName("verticalLayout_7")
```

```
        self.horizontalLayout_2 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_2.setObjectName("horizontalLayout_2")
        self.radioButton_picture = QtWidgets.QRadioButton(self.replacement_result_gro
        font = QtGui.QFont()
        font.setBold(False)
        font.setUnderline(False)
        font.setWeight(50)
        self.radioButton_picture.setFont(font)
        self.radioButton_picture.setChecked(True)
        self.radioButton_picture.setObjectName("radioButton_picture")
        self.horizontalLayout_2.addWidget(self.radioButton_picture)
        self.radioButton_video = QtWidgets.QRadioButton(self.replacement_result_group
        font = QtGui.QFont()
        font.setBold(False)
        font.setUnderline(False)
        font.setWeight(50)
        self.radioButton_video.setFont(font)
        self.radioButton_video.setObjectName("radioButton_video")
        self.horizontalLayout_2.addWidget(self.radioButton_video)
        self.radioButton_camera = QtWidgets.QRadioButton(self.replacement_result_grou
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Minimum, QtWidgets.Q
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.radioButton_camera.sizePolicy().hasHeightFo
        self.radioButton_camera.setSizePolicy(sizePolicy)
        font = QtGui.QFont()
        font.setBold(False)
        font.setUnderline(False)
        font.setWeight(50)
        self.radioButton_camera.setFont(font)
        self.radioButton_camera.setObjectName("radioButton_camera")
        self.horizontalLayout_2.addWidget(self.radioButton_camera)
        self.verticalLayout_7.addLayout(self.horizontalLayout_2)
        self.gridLayout_9 = QtWidgets.QGridLayout()
        self.gridLayout_9.setObjectName("gridLayout_9")
        self.replacement_result_label = QtWidgets.QLabel(self.replacement_result_grou
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Expanding, QtWidgets
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.replacement_result_label.sizePolicy().hasHe
        self.replacement_result_label.setSizePolicy(sizePolicy)
        self.replacement_result_label.setMinimumSize(QtCore.QSize(400, 240))
        self.replacement_result_label.setText("")
        self.replacement_result_label.setScaledContents(True)
        self.replacement_result_label.setObjectName("replacement_result_label")
        self.gridLayout_9.addWidget(self.replacement_result_label, 3, 0, 1, 1)
        self.horizontalLayout_4 = QtWidgets.QHBoxLayout()
        self.horizontalLayout_4.setObjectName("horizontalLayout_4")
        self.post_it_color_label = QtWidgets.QLabel(self.replacement_result_groupbox)
        sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Minimum, QtWidgets.Q
        sizePolicy.setHorizontalStretch(0)
        sizePolicy.setVerticalStretch(0)
        sizePolicy.setHeightForWidth(self.post_it_color_label.sizePolicy().hasHeightF
        self.post_it_color_label.setSizePolicy(sizePolicy)
        font = QtGui.QFont()
        font.setBold(False)
        font.setUnderline(False)
        font.setWeight(50)
        self.post_it_color_label.setFont(font)
        self.post_it_color_label.setObjectName("post_it_color_label")
        self.horizontalLayout_4.addWidget(self.post_it_color_label)
```

```
182         spacerItem = QtWidgets.QSpacerItem(40, 20, QtWidgets.QSizePolicy.Expanding,
    QtWidgets.QSizePolicy.Minimum)
183         self.horizontalLayout_4.addItem(spacerItem)
184         self.post_it_lineEdit = QtWidgets.QLineEdit(self.replacement_result_groupbox)
185         sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Fixed, QtWidgets.QSi
186         sizePolicy.setHorizontalStretch(0)
187         sizePolicy.setVerticalStretch(0)
188         sizePolicy.setHeightForWidth(self.post_it_lineEdit.sizePolicy().hasHeightForW
189         self.post_it_lineEdit.setSizePolicy(sizePolicy)
190         self.post_it_lineEdit.setMaximumSize(QtCore.QSize(30, 16777215))
191         self.post_it_lineEdit.setObjectName("post_it_lineEdit")
192         self.horizontalLayout_4.addWidget(self.post_it_lineEdit)
193         self.post_it_color_pushButton = QtWidgets.QPushButton(self.replacement_result
194         sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Fixed, QtWidgets.QSi
195         sizePolicy.setHorizontalStretch(0)
196         sizePolicy.setVerticalStretch(0)
197         sizePolicy.setHeightForWidth(self.post_it_color_pushButton.sizePolicy().hasHe
198         self.post_it_color_pushButton.setSizePolicy(sizePolicy)
199         font = QtGui.QFont()
200         font.setBold(False)
201         font.setUnderline(False)
202         font.setWeight(50)
203         self.post_it_color_pushButton.setFont(font)
204         self.post_it_color_pushButton.setObjectName("post_it_color_pushButton")
205         self.horizontalLayout_4.addWidget(self.post_it_color_pushButton)
206         self.gridLayout_9.addLayout(self.horizontalLayout_4, 4, 0, 1, 1)
207         self.verticalLayout_7.addLayout(self.gridLayout_9)
208         self.gridLayout_8.addLayout(self.verticalLayout_7, 0, 0, 1, 1)
209         spacerItem1 = QtWidgets.QSpacerItem(20, 40, QtWidgets.QSizePolicy.Minimum,
    QtWidgets.QSizePolicy.MinimumExpanding)
210         self.gridLayout_8.addItem(spacerItem1, 3, 0, 1, 1)
211         self.gridLayout_10.addWidget(self.replacement_result_groupbox, 0, 0, 1, 1)
212         MainWindow.setCentralWidget(self.centralwidget)
213         self.menubar = QtWidgets.QMenuBar(MainWindow)
214         self.menubar.setGeometry(QtCore.QRect(0, 0, 904, 25))
215         self.menubar.setObjectName("menubar")
216         self.menuMain = QtWidgets.QMenu(self.menubar)
217         self.menuMain.setObjectName("menuMain")
218         MainWindow.setMenuBar(self.menubar)
219         self.statusbar = QtWidgets.QStatusBar(MainWindow)
220         self.statusbar.setObjectName("statusbar")
221         MainWindow.setStatusBar(self.statusbar)
222         self.actionLoader = QtWidgets.QAction(MainWindow)
223         self.actionLoader.setObjectName("actionLoader")
224         self.menuMain.addSeparator()
225         self.menuMain.addAction(self.actionLoader)
226         self.menubar.addAction(self.menuMain.menuAction())
227 
228         self.retranslateUi(MainWindow)
229         QtCore.QMetaObject.connectSlotsByName(MainWindow)
230 
231     def retranslateUi(self, MainWindow):
232         _translate = QtCore.QCoreApplication.translate
233         MainWindow.setWindowTitle(_translate("MainWindow", "Post-it Replace"))
234         self.replace_image_groupbox.setTitle(_translate("MainWindow", "Replace Image'
235         self.original_source_groupbox.setTitle(_translate("MainWindow", "Original Sou
236         self.replacement_result_groupbox.setTitle(_translate("MainWindow", "Replaceme
237         self.pushButton_pause.setText(_translate("MainWindow", "Pause"))
238         self.pushButton_replace.setText(_translate("MainWindow", "Replace"))
239         self.radioButton_picture.setText(_translate("MainWindow", "Picture"))
240         self.radioButton_video.setText(_translate("MainWindow", "Video"))
241         self.radioButton_camera.setText(_translate("MainWindow", "Camera"))
```

```
242         self.post_it_color_label.setText(_translate("MainWindow", "Post-it Color:"))
243         self.post_it_color_pushButton.setText(_translate("MainWindow", "Select"))
244         self.menuMain.setTitle(_translate("MainWindow", "File"))
245         self.actionLoader.setText(_translate("MainWindow", "Load..."))
246 
247 
248 if __name__ == "__main__":
249     import sys
250     app = QtWidgets.QApplication(sys.argv)
251     MainWindow = QtWidgets.QMainWindow()
252     ui = Ui_MainWindow()
253     ui.setupUi(MainWindow)
254     MainWindow.show()
255     sys.exit(app.exec_())
256 
257 
```

```python
#maim_window.py

from PyQt5.QtCore import Qt, QTimer
from PyQt5.QtGui import QPixmap, QColor, QImage
from PyQt5.QtWidgets import QMainWindow, QColorDialog
import os
import cv2
import numpy as np
from PIL import Image

from ui_py.ui_mainwindow import Ui_MainWindow
from dialogs.open_load_dialog import OpenLoadDialog
from dialogs.saved_values_paths import SavedValuesConstants

REPLACED_FILE_NAME = "savedImage.jpg"
COLOR_RANGE = 20


class MainWindow(QMainWindow):
    def __init__(self):
        super(MainWindow, self).__init__()

        # setup ui layout
        self.ui = Ui_MainWindow()
        self.ui.setupUi(self)

        # register event handlers
        self.ui.radioButton_camera.toggled.connect(self.__refresh_ui)
        self.ui.radioButton_picture.toggled.connect(self.__refresh_ui)
        self.ui.radioButton_video.toggled.connect(self.__refresh_ui)
        self.ui.actionLoader.triggered.connect(self.on_action_loader_triggered)
        self.ui.pushButton_replace.pressed.connect(self.show_replace_result)

self.ui.post_it_color_pushButton.pressed.connect(self.on_action_settings_color_picker

        self.post_it_color_rgb_value = np.array([])

        self.replace_image_path = None
        self.original_picture_path = None
        self.original_video_path = None

        self.__refresh_post_it_color_line_edit()

    def on_action_loader_triggered(self):
        dialog = OpenLoadDialog()
        if not dialog.exec_():
            return

        self.replace_image_path = dialog.replace_image_file_path()
        self.original_picture_path = dialog.original_picture_file_path()
        self.original_video_path = dialog.original_video_file_path()

        self.__refresh_ui()

    def on_action_settings_color_picker_triggered(self):
        title = "Choose the color of post-it"
        color = SavedValuesConstants.SettingsColorPicker.CUSTOMIZED_COLOR_POST_IT
        color_pick = QColorDialog.getColor(color, self, title, QColorDialog.ShowAlpha

        self.__store_post_it_color(color_pick)
        self.__update_post_it_color()
```

```
61          self.__refresh_post_it_color_line_edit()
62
63      def show_replace_result(self):
64          assert self.replace_image_path is not None
65          replace_img = self.__convert_rgb_to_bgr(self.replace_image_path)
66          replace_img = Image.fromarray(replace_img)
67
68          # check which radiobutton has been selected
69          if self.ui.radioButton_picture.isChecked():
70              self.ui.pushButton_pause.setEnabled(False)
71              self.on_picture_button_is_checked(replace_img)
72          if self.ui.radioButton_video.isChecked():
73              self.ui.pushButton_pause.setEnabled(True)
74              self.on_video_button_is_checked(replace_img)
75          if self.ui.radioButton_camera.isChecked():
76              self.ui.pushButton_pause.setEnabled(False)
77              self.on_camera_button_is_checked(replace_img)
78
79      def on_picture_button_is_checked(self, replace_img):
80          assert self.original_picture_path is not None
81          self.replace_frame = self.show_result_picture(self.original_picture_path, rep
82          self.__refresh_replaced_result(self.replace_frame)
83
84      def on_video_button_is_checked(self, replace_img):
85          assert self.original_video_path is not None
86          self.cap = cv2.VideoCapture(self.original_video_path)
87          self.replace_img = replace_img
88          self.timer_video = QTimer(self)
89          self.timer_video.timeout.connect(self.show_result_video)
90          self.timer_video.start(50)
91
92      def on_camera_button_is_checked(self, replace_img):
93          self.cap = cv2.VideoCapture(0)
94          self.replace_img = replace_img
95          self.timer_camera = QTimer(self)
96          self.timer_camera.timeout.connect(self.show_result_camera_stream)
97          self.timer_camera.start(5)
98
99      def show_result_picture(self, pic_path, replace_img):
100         # Picture, change the post-it part in pic
101         frame = self.__convert_rgb_to_bgr(pic_path)
102         replace_result = self.detect_and_replace(frame.copy(), replace_img)
103         return replace_result
104
105     def show_result_video(self):
106         # Video, change the post-it part in video
107         replace_img = self.replace_img
108         ret, frame = self.cap.read()
109
110         if frame is not None:
111             replace_result = self.detect_and_replace(frame.copy(), replace_img)
112
113             replace_result = self.__convert_bgr_to_rgb(replace_result)
114             frame = self.__convert_bgr_to_rgb(frame)
115
116             self.__refresh_original_video(frame)
117             self.__refresh_replaced_video(replace_result)
118
119             if self.ui.pushButton_pause.isChecked():
120                 self.timer_video.stop()
121                 return
```

```
122
123         else:
124             self.timer_video.stop()
125             return
126
127     def show_result_camera_stream(self):
128         # webcam, change the post-it part in camera stream
129         replace_img = self.replace_img
130         ret, frame = self.cap.read()
131
132         if frame is not None:
133             replace_result = self.detect_and_replace(frame.copy(), replace_img)
134
135             replace_result = self.__convert_bgr_to_rgb(replace_result)
136             frame = self.__convert_bgr_to_rgb(frame)
137
138             self.__refresh_original_video(frame)
139             self.__refresh_replaced_video(replace_result)
140
141         else:
142             self.timer_camera.stop()
143             return
144
145     def detect_and_replace(self, frame, replace_img):
146         contours, hierarchy = self.find_contours(frame)
147         try:
148             # find biggest bounding box
149             biggest, index = 0, 0
150             rect = list()
151             for i in range(len(contours)):
152                 x, y, w, h = cv2.boundingRect(contours[i])
153                 rect.append(((x, y), (x + w, y + h), (w, h)))
154                 if w * h > biggest:
155                     biggest = w * h
156                     index = i
157
158             # draw bounding box in captured image
159             height = rect[index][1][1]- rect[index][0][1]
160             width = rect[index][1][0] - rect[index][0][0]
161             trans_pt1 = (int(rect[index][0][0]+width/4), int(rect[index][0][1]+height
162             trans_pt2 = (int(rect[index][1][0]-width/4), int(rect[index][1][1]-height
163
164             # frame = cv2.rectangle(frame, trans_pt1, trans_pt2, (0, 0, 255), 2)
165             temp_replace = np.array(replace_img.resize((int(trans_pt2[0]-
trans_pt1[0]),int(trans_pt2[1]-trans_pt1[1]))))
166             frame[trans_pt1[1] : trans_pt2[1], trans_pt1[0] : trans_pt2[0]] = temp_re
167
168         except:
169             pass
170         return frame
171
172     def set_mask(self, frame):
173         lower = self.post_it_color_rgb_value - COLOR_RANGE
174         upper = self.post_it_color_rgb_value + COLOR_RANGE
175         return cv2.inRange(frame, lowerb=lower, upperb=upper)
176
177     def find_contours(self, frame):
178         mask = self.set_mask(frame)
179         return cv2.findContours(mask, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
180
181     @staticmethod
```

```
182     def __convert_rgb_to_bgr(rgb_img_path):
183         # open image with Image.open to avoid error from opencv
184         # convert rgb image to bgr for opencv
185         rgb_img = np.array(Image.open(rgb_img_path).convert("RGB"))
186         return cv2.cvtColor(rgb_img, cv2.COLOR_RGB2BGR)
187 
188     @staticmethod
189     def __convert_bgr_to_rgb(bgr_img):
190         return cv2.cvtColor(bgr_img, cv2.COLOR_BGR2RGB)
191 
192     @staticmethod
193     def __store_post_it_color(color_pick):
194         SavedValuesConstants.SettingsColorPicker.CUSTOMIZED_COLOR_POST_IT = color_pic
195 
196     def __update_post_it_color(self):
197         rgb_value = SavedValuesConstants.SettingsColorPicker.CUSTOMIZED_COLOR_POST_IT
198         self.post_it_color_rgb_value = np.array([rgb_value.blue(), rgb_value.green(),
    rgb_value.red()])
199 
200     def __refresh_ui(self):
201         self.__update_post_it_color()
202         self.__refresh_replace_image(self.replace_image_path)
203         self.__refresh_original_picture(self.original_picture_path)
204 
205     def __refresh_replace_image(self, replace_image_path):
206         pix_map = QPixmap(replace_image_path)
207         self.ui.replace_image_label.setPixmap(pix_map)
208 
209     def __refresh_original_picture(self, original_picture_path):
210         pix_map = QPixmap(original_picture_path)
211         self.ui.original_source_label.setPixmap(pix_map)
212 
213     def __refresh_replaced_result(self, frame):
214         frame = self.__convert_bgr_to_rgb(frame)
215         source_image = QImage(frame.data, frame.shape[1], frame.shape[0], QImage.Form
216         self.ui.replacement_result_label.setPixmap(QPixmap.fromImage(source_image))
217 
218     def __refresh_original_video(self, frame):
219         source_image = QImage(frame.data, frame.shape[1], frame.shape[0], QImage.Form
220         self.ui.original_source_label.setPixmap(QPixmap.fromImage(source_image))
221 
222     def __refresh_replaced_video(self, replace_result):
223         show_image = QImage(
224                 replace_result.data, replace_result.shape[1], replace_result.shape[0]
    QImage.Format_RGB888
225             )
226         self.ui.replacement_result_label.setPixmap(QPixmap.fromImage(show_image))
227 
228     def __refresh_post_it_color_line_edit(self):
229         self.ui.post_it_lineEdit.setStyleSheet(
230             "QLineEdit { background-color: %s}"
231             % SavedValuesConstants.SettingsColorPicker.CUSTOMIZED_COLOR_POST_IT.name(
232         )
233 
```

```python
#post_it_replace.py

from PyQt5.QtWidgets import QApplication
from windows.main_window import MainWindow


def launch_main_window():
    app = QApplication([])
    main_window = MainWindow()
    main_window.show()
    app.exec_()


if __name__ == "__main__":
    launch_main_window()

```

Bibliography

[1] Wikipedia - GUI, https://en.wikipedia.org/wiki/Graphical_user_interface

[2] Wikipedia - Qt (software),
https://en.wikipedia.org/wiki/Qt_(software)

[3] Wikipedia - Qt Creator,
https://en.wikipedia.org/wiki/Qt_Creator

[4] Python - Introduction to PyQt5,
https://www.geeksforgeeks.org/python-introduction-to-pyqt5/

[5] Wikipedia - Visual Studio Code,
https://en.wikipedia.org/wiki/Visual_Studio_Code

[6] Wikipedia - Computer Vision,
https://en.wikipedia.org/wiki/Computer_vision

[7] OpenCV-Python,
https://www.geeksforgeeks.org/python-opencv-cv2-imread-method/

[8] Computer Vision for Beginners: Part 4,
https:
//towardsdatascience.com/computer-vision-for-beginners-part-4-64a8d9856208

[9] opencv function docs,
https://docs.opencv.org/4.1.2/index.html