郑 州 大 学

生 产 实 习 报 告

专　业 软件工程

班　级 软工四班

学　号 202124100508

姓　名 关晗

计算机与人工智能学院

2024 年 7 月

实习时间	2024 年 6 月 27 日至 7 月 6 日	实习地点	3 楼机房
实习单位	东方瑞通（郑州）咨询服务有限公司		
指导教师	梁鹏		
项目名称	VisionVoyage-基于鱼眼相机与其他感知技术的自动驾驶仿真系统		

实习目的：

1. 提高团队协作能力：在团队中分工合作，使用版本控制工具（如 Git），在实验过程中，采用敏捷开发模式（如 Scrum），学习快速迭代、持续反馈和改进的方法，提升学生在动态环境下开发人工智能项目的能力，提升团队协作和沟通能力。

2.了解在人工智能领域的应用： 通过具体实验，熟悉 Python 在人工智能中的广泛应用，包括数据处理、模型训练、结果评估等。

3. 掌握基本的人工智能算法和模型： 学习并实现常见的人工智能算法，如线性回归、决策树、神经网络等，理解它们的原理和应用场景。

4. 提升 Python 编程能力： 强化对 Python 语言的掌握，尤其是使用 Python 进行数据分析、模型构建和优化的能力，熟悉相关的库和工具（如 NumPy、Pandas、Scikit-learn、TensorFlow、Keras 等）。

5. 熟悉人工智能项目的开发流程： 从数据预处理、模型选择与训练、模型评估与优化，到结果解释与应用，全面了解人工智能项目的完整开发流程。

6. 培养解决实际问题的能力： 通过实验项目中的实际问题，学会分析问题、设计解决方案、实施并验证效果，提升在实际场景中解决问题的能力。

7. 关注人工智能领域的前沿技术和发展趋势： 了解当前人工智能领域的最新研究成果和技术发展趋势，培养创新意识和行业敏锐度。

8. 提升职业素养和专业能力： 通过严格的实验规范和要求，培养职业素养，提升在未来工作中的专业能力和素质。

9. 激发创新意识： 在实验过程中探索新方法、新技术，提出创新性的解决方案，培养独立思考和创新能力。

项目简介：

VisionVoyage：基于鱼眼相机与其他感知技术的自动驾驶仿真系统

VisionVoyage 是一款先进的自动驾驶仿真系统，利用鱼眼相机和多种感知技术，旨在提供全面的仿真环境和多样的应用场景。该项目采用了多种先进的技术和框架，确保系统的高效性、稳定性和易用性。

关键技术和框架：

语言和界面：

GUI 界面： 采用 PyQt6 和 PySide6 开发，提供简洁、直观的用户界面。这两个框架都是 Qt 框架的 Python 绑定，Qt 是一个跨平台的 C++ 框架，专注于开发图形用户界面应用程序。

后端： 采用 Python 3.8 和 C++17 实现，遵循 PEP8 规范，确保代码的规范性和可维护性。

开发环境及框架：

Unreal Engine 4： 从 Epic 官方 GitHub 仓库克隆源码，使用 clang++-10 编译生成 UE4Editor。支持主流操作系统平台，如 Linux、Windows、macOS。本项目实际使用的是 Ubuntu 20.04.6 LTS Desktop，并结合 CUDA 11.8 和 CUDNN 8.6.0 进行 GPU 加速。

资源整合： 使用 QRC（Qt Resource Collection）整合本地资源，提升系统的稳定性和性能。

感知技术：

语义分割： 使用 MMSegmentation 作为语义分割框架，结合 Mask2Former 和 Seg2Former 等深度学习模型，实现对图像的高效处理和分析。

鱼眼相机应用： 鱼眼相机通过压缩和扭曲大角度范围内的光线，提供 180° 视角的环境数据。通过拼接多个相机图像，去除扭曲和变形，实现 360° 全景影像功能。在泊车场景中，结合超声波雷达，实现停车线识别和障碍物检测。在自动驾驶系统中，鱼眼相机还用于车道线识别、交通信号灯识别和路况检测，提高行车安全性。

用户平台/环境：

操作系统： 推荐使用 Ubuntu 18.04/20.04 LTS Desktop，兼容 Windows 10/11。采用 C/S 架构，并采用前后端分离方案，提升系统的扩展性和维护性。

技术挑战和创新：

尽管鱼眼相机具有广泛的应用前景，但公开发布的鱼眼数据集非常有限，除了 WoodScape 数据集外，几乎没有其他数据集提供真正的语义分割注释。此外，目前针对鱼眼摄像头，还没有一个量产的、基于多摄像头组合的语义分割算法，大部分工作都集中在独立解决个别任务上。现有的编码器是共享的，但解码器之间没有协同作用。同时，现有的数据集主要是为了特定任务而设计，并没有为所有任务提供同时注释。鱼眼图像的畸变问题，使得传统的语义分割模型难以取得良好效果，需要设计特定的处理模块来应对这些挑战。

应用场景：

通过结合计算机视觉和深度学习技术，VisionVoyage 实现了对鱼眼相机获取的图像进行实时处理和分析，使车辆能够更准确地理解并适应不同的交通场景。鱼眼相机还用于实现车辆周围的盲区监测，提高行车安全性。通过将鱼眼相机获取的图像与高精度地图数据进行融合，还可以实现更精准的定位和路径规划，为自动驾驶系统提供更可靠的导航和控制支持。

VisionVoyage 致力于推动自动驾驶技术的发展，通过提供一个强大的仿真环境，帮助开发者进行测试和验证，从而加速自动驾驶技术的商业化进程。

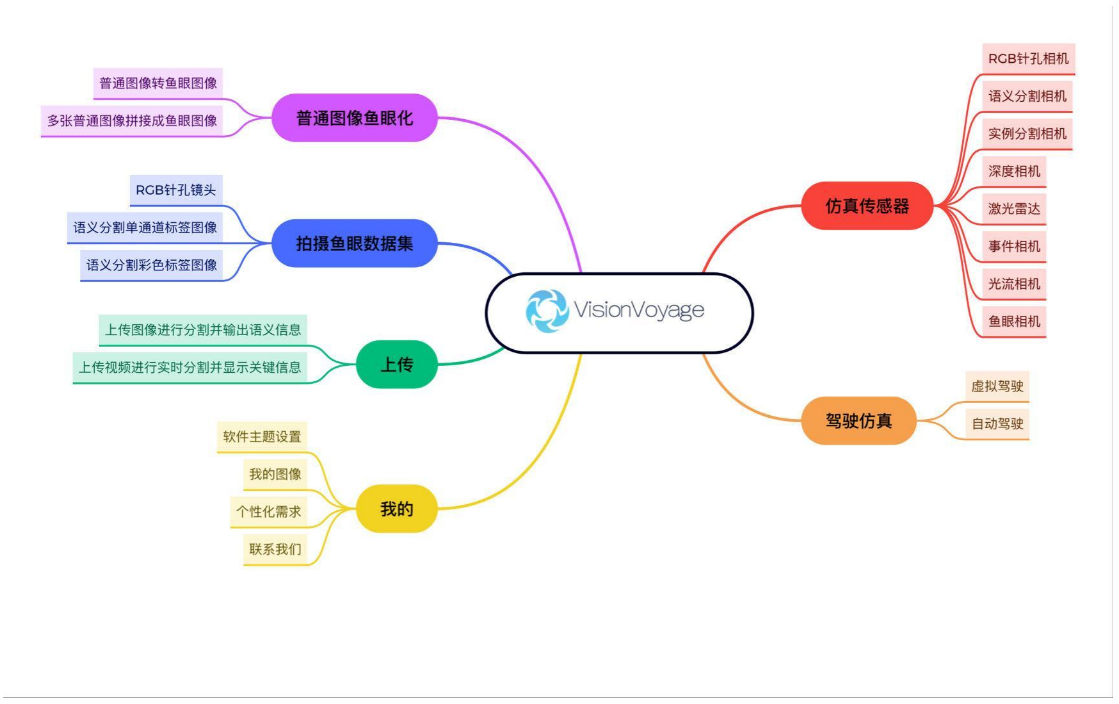
个人承担工作：

个人承担工作：

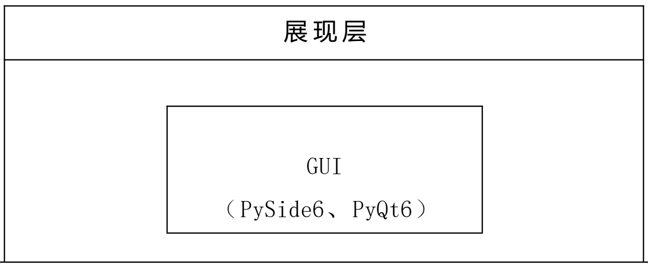
①软件运行等待窗口，加载界面

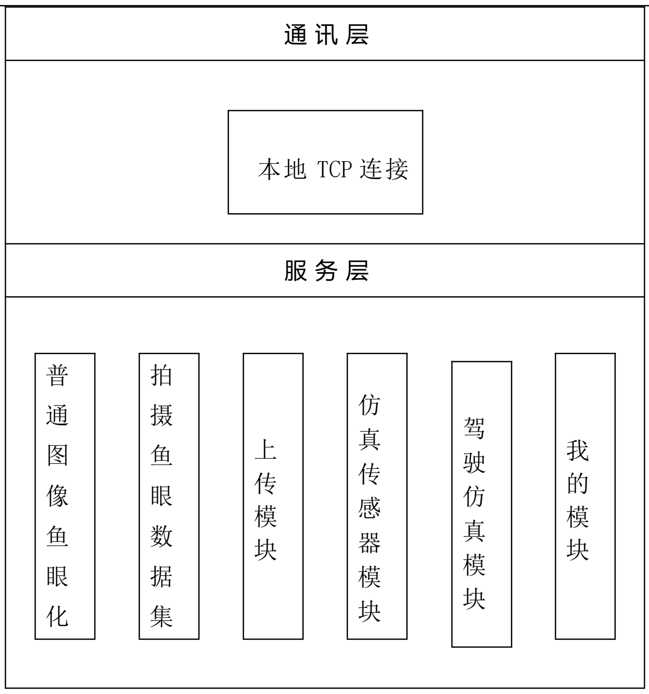
②利用 MMsegmentation 语义分割框架识别图片（部分）

功能图：

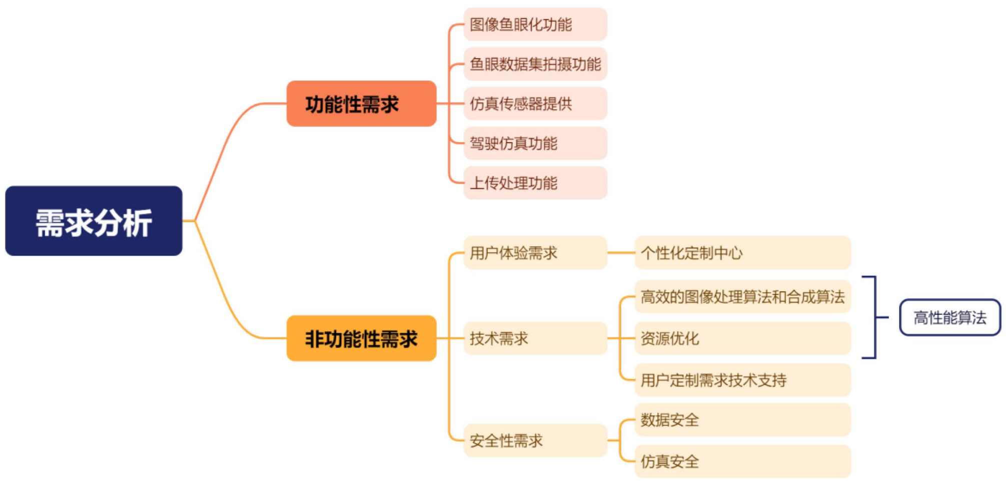


逻辑结构图：





需求分析图：



重要代码以及重要接口：

```
#!/home/m0rtzz/Program_Files/anaconda3/envs/py38/bin/python3


# ////////////////////////////////////////////////////////////////
#
# @file main.py
# @brief 程序入口文件
# @author M0rtzz E-mail: m0rtzz@outlook.com
# @version 4.0
# @PROJECT MADE WITH: Qt Designer and PySide6
#
# ///////////////////////////////////////////////////////////////

from scripts.alipay import AlipayPayment
from widgets import *
from modules import *
import subprocess
import warnings
import time
import sys
import cv2
import os

# NOTE: 禁止输出错误信息
sys.stderr = open('/dev/null', 'w')

# NOTE: 禁止指定警告输出
warnings.filterwarnings("ignore", category=DeprecationWarning)

os.environ["QT_FONT_DPI"] = "96"  # FIX Problem for High DPI and Scale
above 100%

# SET AS GLOBAL WIDGETS
widgets = None
```

```
class MainWindow(QMainWindow):
    def __init__(self):
        QMainWindow.__init__(self)
        self.dark_theme_enabled = True
        self.file_names = set()
        self.file_paths = set()
        self.fisheye_directory = ’./images/my_images/fisheye_dataset’
        self.common_directory = ’./images/my_images/other_sensors’
        self.normal_directory                                          =
’./images/my_images/fisheye_transformation/normal2fisheye’
        self.cubemap_directory                                         =
’./images/my_images/fisheye_transformation/cubemap2fisheye’
        self.sem_seg_directory = ’./images/my_images/sem_seg/output’
        self.is_plus = False

        # SET AS GLOBAL WIDGETS
        self.ui = Ui_MainWindow()
        self.ui.setupUi(self)
        global widgets
        widgets = self.ui

        # USE CUSTOM TITLE BAR | USE AS ″False″ FOR MAC OR LINUX
        Settings.ENABLE_CUSTOM_TITLE_BAR = True
        # Settings.ENABLE_CUSTOM_TITLE_BAR = False

        # APP NAME
        title = ″VisionVoyage - Modern GUI″
        # description = ″VisionVoyage - 一款基于鱼眼相机与其他感知技术
的自动驾驶仿真系统。″
```

```
        # # APPLY TEXTS
        self.setWindowTitle(title)
        # widgets.titleRightInfo.setText(description)

self.ui.line_edit_filenames.textChanged.connect(self.handleLineEditCh
ange)

        # TOGGLE MENU
        widgets.toggleButton.clicked.connect(lambda:
UIFunctions.toggleMenu(self, True))

        # SET UI DEFINITIONS
        UIFunctions.uiDefinitions(self)

        # BUTTON
        widgets.btn_home.clicked.connect(self.buttonClick)
        widgets.btn_widgets.clicked.connect(self.buttonClick)
        widgets.btn_image.clicked.connect(self.buttonClick)
        widgets.btn_simulation.clicked.connect(self.buttonClick)
        widgets.btn_personal_center.clicked.connect(self.buttonClick)
        widgets.btn_adjustments.clicked.connect(self.buttonClick)
        widgets.btn_my_image.clicked.connect(self.buttonClick)
        widgets.btn_open_dir.clicked.connect(self.buttonClick)
        widgets.btn_fisheye_one2one.clicked.connect(self.buttonClick)

widgets.btn_fisheye_five2one.clicked.connect(self.buttonClick)

widgets.btn_segmentation_image.clicked.connect(self.buttonClick)
widgets.btn_segmentation_video.clicked.connect(self.buttonClick)
        widgets.btn_get_fisheye.clicked.connect(self.buttonClick)
        widgets.btn_get_common.clicked.connect(self.buttonClick)
        widgets.btn_raw_to_platte.clicked.connect(self.buttonClick)
        widgets.btn_start_server.clicked.connect(self.buttonClick)

widgets.btn_generate_traffic.clicked.connect(self.buttonClick)
        widgets.btn_manual_control.clicked.connect(self.buttonClick)

widgets.btn_automatic_control.clicked.connect(self.buttonClick)
        widgets.btn_send_mail.clicked.connect(self.buttonClick)
        widgets.btn_print.clicked.connect(self.buttonClick)
        widgets.btn_unlock.clicked.connect(self.buttonClick)

        # EXTRA LEFT BOX
        def openCloseLeftBox():
            UIFunctions.toggleLeftBox(self, True)

        widgets.btn_personal_center.clicked.connect(openCloseLeftBox)
        widgets.extraCloseColumnBtn.clicked.connect(openCloseLeftBox)

        # EXTRA RIGHT BOX
        def openCloseRightBox():
            UIFunctions.toggleRightBox(self, True)

        widgets.settingsTopBtn.clicked.connect(openCloseRightBox)

        # SHOW APP
        self.show()

        # SET CUSTOM THEME
        use_custom_theme = True
        # theme_file = "./themes/light.qss"
        theme_file = "./themes/dark.qss"

        # SET THEME AND HACKS
        if use_custom_theme:
            # LOAD AND APPLY STYLE
            UIFunctions.theme(self, theme_file, True)

            # SET HACKS
            AppFunctions.setThemeHack(self)

        # SET HOME PAGE AND SELECT MENU
        widgets.stackedWidget.setCurrentWidget(widgets.home)

widgets.btn_home.setStyleSheet(UIFunctions.selectMenu(widgets.btn_hom
e.styleSheet()))

    # BUTTONS CLICK
    # Post here your functions for clicked buttons
    def setTableFontColor(self, table_widget):
        row_count = table_widget.rowCount()
        column_count = table_widget.columnCount()

        # 遍历所有单元格
        for row in range(row_count):
            for col in range(column_count):
                item = table_widget.item(row, col)
                if item is not None:
                    if self.dark_theme_enabled == True:  # 是黑色主题
                        color = QColor(255, 255, 255)  # 白色
                        item.setForeground(color)
                    else:
                        color = QColor(0, 0, 0)  # 黑色
                        item.setForeground(color)

    def clearColumn(self, widget, column_index):
        # 获取表格的行数
        row_count = widget.rowCount()

        # 清空指定列的每个单元格
        for row in range(row_count):
            item = widget.item(row, column_index)
            if item is not None:
                widget.takeItem(row, column_index)

    def onItemClickedSemSeg(self, item):
        column = item.column()
        row = item.row()
        table_widget = item.tableWidget()
        file_name = table_widget.item(row, column).text()
        # 遍历 output 目录下的所有子目录
        for root, dirs, files in os.walk(self.sem_seg_directory):
            for name in files:
                if name == file_name:
                    file_path = os.path.join(root, name)
                    if os.path.exists(file_path):
                        if file_name.endswith(".mp4"):

self.openVideoWithDefaultPlayer(file_path)
                        else:
                            cv2.namedWindow(file_name,
cv2.WINDOW_NORMAL | cv2.WINDOW_GUI_NORMAL)
                            cv2.resizeWindow(file_name, 800, 600)
                            mat = cv2.imread(file_path)
                            cv2.imshow(file_name, mat)
                        return
        print("File not found:", file_name)

    def onItemClickedFisheye(self, item):
        column = item.column()
        row = item.row()
        table_widget = item.tableWidget()
        file_name = table_widget.item(row, column).text()

        # 遍历 fisheye_dataset 目录下的所有子目录
        for root, dirs, files in os.walk(self.fisheye_directory):
            for name in files:
                if name == file_name:  # 如果找到了匹配的文件名
                    file_path = os.path.join(root, name)
                    if os.path.exists(file_path):  # 检查文件是否存在
                        cv2.namedWindow(file_name, cv2.WINDOW_NORMAL |
cv2.WINDOW_GUI_NORMAL)
                        cv2.resizeWindow(file_name, 800, 600)  # 设置
```

```
窗口大小为 800x600
                        mat = cv2.imread(file_path)
                        cv2.imshow(file_name, mat)
                        return  # 找到文件后就返回，不再继续搜索
        print("File not found:", file_name)  # 如果没有找到文件，打印
信息

    def onItemClickedCommon(self, item):
        column = item.column()
        row = item.row()
        table_widget = item.tableWidget()
        file_name = table_widget.item(row, column).text()

        for root, dirs, files in os.walk(self.common_directory):
            if file_name in files:
                file_path = os.path.join(root, file_name)
                cv2.namedWindow(file_name,     cv2.WINDOW_NORMAL     |
cv2.WINDOW_GUI_NORMAL)
                cv2.resizeWindow(file_name, 800, 600)  # 设置窗口大小
为 800x600
                mat = cv2.imread(file_path)
                cv2.imshow(file_name, mat)

    def onItemClickedPT(self, item):
        column = item.column()
        row = item.row()
        table_widget = item.tableWidget()
        file_name = table_widget.item(row, column).text()
        file_path = os.path.join(self.normal_directory, file_name)
        if os.path.exists(file_path):  # 检查文件是否存在
            # file_dir = os.path.dirname(file_path)
            #self.openVisualDirectory(file_dir)  # 打开文件所在文件夹
            cv2.namedWindow(file_name,       cv2.WINDOW_NORMAL        |
cv2.WINDOW_GUI_NORMAL)
            cv2.resizeWindow(file_name, 800, 600)  # 设置窗口大小为
800x600
            mat = cv2.imread(file_path)
            cv2.imshow(file_name, mat)

    def onItemClickedCubemap(self, item):
        column = item.column()
        row = item.row()
        table_widget = item.tableWidget()
        file_name = table_widget.item(row, column).text()
        file_path = os.path.join(self.cubemap_directory, file_name)
        if os.path.exists(file_path):  # 检查文件是否存在
            # file_dir = os.path.dirname(file_path)
            #self.openVisualDirectory(file_dir)  # 打开文件所在文件夹
            cv2.namedWindow(file_name,       cv2.WINDOW_NORMAL        |
cv2.WINDOW_GUI_NORMAL)
            cv2.resizeWindow(file_name, 800, 600)  # 设置窗口大小为
800x600
            mat = cv2.imread(file_path)
            cv2.imshow(file_name, mat)

    def handleLineEditChange(self, text):
        if not text:  # 如果 LineEdit 内容为空
            self.file_names.clear()  # 清空 file_names 集合
            self.file_paths.clear()  # 清空 file_paths 集合

    def openVideoWithDefaultPlayer(self, video_path):
        if sys.platform.startswith('linux'):  # Linux
            subprocess.run(['xdg-open', video_path])
        elif sys.platform == 'darwin':  # MacOS
            subprocess.run(['open', video_path])
        elif sys.platform == 'win32':  # Windows
            subprocess.run(['start', '', video_path], shell=True)
        else:
            print("Unsupported operating system")

    def openVisualDirectory(self, directory):
        if sys.platform.startswith('linux'):  # Linux 或类 Unix 系统
            os.system(f"xdg-open {directory}")
        elif sys.platform == 'darwin':  # macOS
            os.system(f"open {directory}")
        elif sys.platform == 'win32':  # Windows
            os.system(f"start {directory}")
        else:
            print("Unsupported platform")

    def openImage(self, folder_path, flag=False):
        # 检查文件夹路径是否存在
        if not os.path.isdir(folder_path):
            print("指定的文件夹不存在")
            return

        # 支持的图片格式列表
        image_extensions = ['.png', '.jpg', '.jpeg', '.gif', '.bmp',
'.tiff']

        if not flag:  # 如果不需要查找最新更改的文件
            # 查找文件夹中第一张图片
            first_image = None
            for file in sorted(os.listdir(folder_path)):
                if    any(file.lower().endswith(ext)    for    ext    in
image_extensions):
                    first_image = file
                    break

            # 如果没有找到图片
            if not first_image:
                print("在指定的文件夹中没有找到图片")
                return

            # 获取完整的文件路径
            image_path = os.path.join(folder_path, first_image)
        else:  # 如果需要查找最新更改的文件
            # 查找文件夹中最新更改的图片
            latest_image = None
            latest_time = 0
            for file in sorted(os.listdir(folder_path)):
                if    any(file.lower().endswith(ext)    for    ext    in
image_extensions):
                    file_path = os.path.join(folder_path, file)
                    file_time = os.path.getmtime(file_path)
                    if file_time > latest_time:
                        latest_time = file_time
                        latest_image = file

            # 如果没有找到图片
            if not latest_image:
                print("在指定的文件夹中没有找到图片")
                return

            # 获取完整的文件路径
            image_path = os.path.join(folder_path, latest_image)

        # 根据不同的操作系统打开图片
        try:
            if sys.platform.startswith('win32'):  # Windows
                os.startfile(image_path)
            elif sys.platform.startswith('darwin'):  # macOS
                subprocess.run(['open', image_path])
            else:  # Linux 或类 Unix 系统
                subprocess.run(['xdg-open', image_path])
        except Exception as e:
            print(f"打开图片时出现错误：{e}")

    def becomePlus(self, phone_number):
        app_private_key_path = "./certs/app_private_key.pem"
        alipay_public_key_path = "./certs/alipay_public_key.pem"
        background_url = 'https://jsd.cdn.zzko.cn/gh/M0rtzz/ImageHosting@master/images/Year:2024/Month:03/Day:15/22:26:14_background.png'
        alipay_payment      =      AlipayPayment(app_private_key_path,
alipay_public_key_path)
        qr_code_url,            out_trade_no_with_time              =
alipay_payment.createOrder()
        qr_cv = alipay_payment.generateQrCode(qr_code_url)
        alipay_payment.displayQrCodeOnBackground(qr_cv,
background_url)

        # 检查支付状态
        alipay_payment.checkPaymentStatus(out_trade_no_with_time,
phone_number)

    def closeWindowByTitle(self, window_title):
        if sys.platform.startswith('win32'):  # Windows
            command = f'taskkill /F /FI "WINDOWTITLE eq {window_title}"'
            os.system(command)
        elif sys.platform.startswith('darwin'):  # macOS
            command = f"osascript -e 'quit app \"{window_title}\"'"
            os.system(command)
        elif sys.platform.startswith('linux'):  # Linux 或类 Unix 系统
            command = f'wmctrl -c "{window_title}"'
            os.system(command)
        else:
            print("Unsupported platform")

    def findEncFile(self, directory, phone_number) -> bool:
        for root, _, files in os.walk(directory):
            for file in files:
                if file.endswith(phone_number + ".enc"):
                    terminal_command = "./scripts/encryptor.out false "
+ phone_number
                    output = subprocess.check_output(terminal_command,
shell=True, text=True)
                    if 'TRADE_SUCCES' in output:
                        print("已成为 VisionVoyage Plus")
                        return True
        return False

    def paymentCodeSegment(self):
        phone_number, ok_pressed = QInputDialog.getText(None, "请输入
手机号", "请输入输入您支付宝绑定的手机号:")
        if ok_pressed:
            phone_number = str(phone_number)

        while True:
            existed_and_is_plus     =     self.findEncFile('./private',
phone_number)
            if existed_and_is_plus:
                self.is_plus = True
                break
            else:
                self.becomePlus(phone_number)
                self.closeWindowByTitle("请在三分钟内完成支付")
                time.sleep(5)

    def buttonClick(self):
        def handle_btn_home(btn):
            widgets.stackedWidget.setCurrentWidget(widgets.home)
            UIFunctions.resetStyle(self, widget=btn.objectName())

btn.setStyleSheet(UIFunctions.selectMenu(btn.styleSheet()))

        def handle_btn_widgets(btn):
            widgets.stackedWidget.setCurrentWidget(widgets.widgets)
            UIFunctions.resetStyle(self, widget=btn.objectName())

btn.setStyleSheet(UIFunctions.selectMenu(btn.styleSheet()))

        def handle_btn_image(btn):
            if not self.is_plus:
                self.paymentCodeSegment()

widgets.stackedWidget.setCurrentWidget(widgets.image_page)
            UIFunctions.resetStyle(self, widget=btn.objectName())

btn.setStyleSheet(UIFunctions.selectMenu(btn.styleSheet()))

        def handle_btn_simulation(btn):
            if not self.is_plus:
                self.paymentCodeSegment()

widgets.stackedWidget.setCurrentWidget(widgets.simulation_page)
            UIFunctions.resetStyle(self, widget=btn.objectName())

btn.setStyleSheet(UIFunctions.selectMenu(btn.styleSheet()))

        def handle_btn_my_image(btn):
            self.openVisualDirectory("./images/my_images")

        def handle_btn_open_dir(btn):
            self.ui.line_edit_filenames.clear()
            self.file_paths.clear()
            file_dialog = QFileDialog()
            file_dialog.setFileMode(QFileDialog.ExistingFiles)
            options = QFileDialog.Options()
            options |= QFileDialog.DontUseNativeDialog
            filter = "图像文件 (*.png *.jpg *.bmp *.jpeg);;视频文件
(*.mp4 *.avi *.mov *.mkv)"
            files, _ = file_dialog.getOpenFileNames(
                None, "选择图片或视频", "./images/my_images", filter,
options=options)
            if files:
                self.file_names.update([os.path.basename(file)     for
file in files])
                self.ui.line_edit_filenames.setText(",
".join(self.file_names))
                self.file_paths.update([os.path.relpath(file,    "./")
for file in files])

        def handle_btn_fisheye_one2one(btn):
            terminal_command   =   "./scripts/PT2fisheye.out   "   +   "
".join(self.file_paths)
            os.system(terminal_command)
            directory                                                    =
'./images/my_images/fisheye_transformation/normal2fisheye'
            table_widget                                                 =
widgets.table_widget_transform_upload_result
            table_widget.itemClicked.connect(self.onItemClickedPT)
            png_files = [file for file in os.listdir(directory) if
file.endswith(".png")]
            self.clearColumn(table_widget, 0)
            for index, file in enumerate(png_files):
                item = QTableWidgetItem(file)
                table_widget.setItem(index, 0, item)

self.openImage('./images/my_images/fisheye_transformation/normal2fish
eye', flag=True)

        def handle_btn_fisheye_five2one(btn):
            terminal_command = "./scripts/cubemap2fisheye.out " + "
".join(self.file_paths)
            os.system(terminal_command)
            directory                                                    =
'./images/my_images/fisheye_transformation/cubemap2fisheye'
            table_widget                                                 =
widgets.table_widget_transform_upload_result

table_widget.itemClicked.connect(self.onItemClickedCubemap)
            png_files = [file for file in os.listdir(directory) if
file.endswith(".png")]
            self.clearColumn(table_widget, 1)
            for index, file in enumerate(png_files):
                item = QTableWidgetItem(file)
                table_widget.setItem(index, 1, item)
```

```
self.openImage('./images/my_images/fisheye_transformation/cubemap2fis
heye', flag=True)

        def handle_btn_segmentation_image(btn):
            terminal_command       =       "./scripts/sem_seg_image.py
--image_paths " + " ".join(self.file_paths)
            os.system(terminal_command)
            base_directory = './images/my_images/sem_seg/output'
            table_widget                                              =
widgets.table_widget_transform_upload_result

table_widget.itemClicked.connect(self.onItemClickedSemSeg)
            sub_dirs = ['images', 'videos']
            media_files = []
            for sub_dir in sub_dirs:
                dir_path = os.path.join(base_directory, sub_dir)
                if os.path.exists(dir_path):
                    media_files.extend([os.path.join(sub_dir, file)
                                        for          file          in
os.listdir(dir_path) if
                                        file.endswith(".png")       or
file.endswith(".mp4")])
            self.clearColumn(table_widget, 2)
            for index, file in enumerate(media_files):
                file = os.path.basename(file)
                item = QTableWidgetItem(file)
                table_widget.setItem(index, 2, item)

        def handle_btn_segmentation_video(btn):
            terminal_command       =       ″./scripts/sem_seg_video.py
--weights ./scripts/weights/video.pt --source ″ + \
                               ″ ″.join(self.file_paths)
            os.system(terminal_command)
            base_directory = './images/my_images/sem_seg/output'
            table_widget                                             =
widgets.table_widget_transform_upload_result

table_widget.itemClicked.connect(self.onItemClickedSemSeg)
            sub_dirs = ['images', 'videos']
            media_files = []
            for sub_dir in sub_dirs:
                dir_path = os.path.join(base_directory, sub_dir)
                if os.path.exists(dir_path):
                    media_files.extend([os.path.join(sub_dir, file)
                                        for        file          in
os.listdir(dir_path) if
                                        file.endswith(″.png″)      or
file.endswith(″.mp4″)])
            self.clearColumn(table_widget, 2)
            for index, file in enumerate(media_files):
                file = os.path.basename(file)
                item = QTableWidgetItem(file)
                table_widget.setItem(index, 2, item)

        def handle_btn_raw_to_platte(btn):
            os.system(″./scripts/change_index.out″)
            os.system(″./scripts/gray2color.out″)
```

```
self.openImage('./images/my_images/fisheye_dataset/semantic_segmentat
ion_CityScapesPalette')

        def handle_btn_start_server(btn):
            subprocess.Popen(['gnome-terminal',             '--title',
'VisionVoyage Server状态终端',
                              '--',                                'sh',
'./scripts/VisionVoyageServer.sh', "-quality-level=low"])

        def handle_btn_generate_traffic(btn):
            subprocess.Popen(['gnome-terminal', '--title', '交通初始化
',
                              '--', './scripts/generate_traffic.py'])

        def handle_btn_manual_control(btn):
            subprocess.Popen(['gnome-terminal', '--title', '虚拟驾驶',
                              '--', './scripts/manual_control.py'])

        def handle_btn_automatic_control(btn):
            subprocess.Popen(['gnome-terminal', '--title', '自动驾驶',
                              '--', './scripts/automatic_control.py'])

        def handle_btn_get_fisheye(btn):
            os.system("./scripts/dataset_main.py")
            base_directory = './images/my_images/fisheye_dataset'
            table_widget = widgets.table_widget_get_image

table_widget.itemClicked.connect(self.onItemClickedFisheye)
            sub_dirs    =    ['rgb',    'semantic_segmentation_raw',
'semantic_segmentation_CityScapesPalette']
            png_files = []
            for sub_dir in sub_dirs:
                dir_path = os.path.join(base_directory, sub_dir)
                if os.path.exists(dir_path):
                    png_files.extend([os.path.join(sub_dir, file)
                                      for file in os.listdir(dir_path)
if file.endswith(".png")])
            self.clearColumn(table_widget, 0)
            for index, file in enumerate(png_files):
                file = os.path.basename(file)
                item = QTableWidgetItem(file)
                table_widget.setItem(index, 0, item)

        def handle_btn_get_common(btn):
            os.system("./scripts/manual_control_gbuffer.py")
            directory = './images/my_images/other_sensors'
            table_widget = widgets.table_widget_get_image

table_widget.itemClicked.connect(self.onItemClickedCommon)
            png_files = [os.path.basename(os.path.join(dp, f)) for dp,
dn, filenames in os.walk(directory)
                         for f in filenames if f.endswith(".png")]
            self.clearColumn(table_widget, 1)
            for index, file in enumerate(png_files):
                item = QTableWidgetItem(file)
                table_widget.setItem(index, 1, item)

        def handle_btn_adjustments(btn):
            self.dark_theme_enabled = not self.dark_theme_enabled
            if self.dark_theme_enabled:
                theme_file = "./themes/dark.qss"
                # 设置字体为颜色为白色
                widgets.btn_fisheye_one2one.setStyleSheet("color:
#FFFFFF;")
                widgets.btn_fisheye_five2one.setStyleSheet("color:
#FFFFFF;")
                widgets.btn_segmentation_image.setStyleSheet("color:
#FFFFFF;")
                widgets.btn_segmentation_video.setStyleSheet("color:
#FFFFFF;")
                widgets.btn_get_fisheye.setStyleSheet("color:
#FFFFFF;")
                widgets.btn_get_common.setStyleSheet("color:
#FFFFFF;")
                widgets.btn_raw_to_platte.setStyleSheet("color:
#FFFFFF;")
                widgets.btn_generate_traffic.setStyleSheet("color:
#FFFFFF;")
                widgets.btn_manual_control.setStyleSheet("color:
#FFFFFF;")
                widgets.btn_automatic_control.setStyleSheet("color:
#FFFFFF;")

widgets.line_edit_operation_help.setStyleSheet("color: #FFFFFF;")
                #widgets.table_widget_get_image.setStyleSheet("color:
#FFFFFF;")
```

```
self.setTableFontColor(widgets.table_widget_get_image)


self.setTableFontColor(widgets.table_widget_operation_help)
            else:
                theme_file = "./themes/light.qss"
                widgets.btn_fisheye_one2one.setStyleSheet("color:
#000000;")
                widgets.btn_fisheye_five2one.setStyleSheet("color:
#000000;")
                widgets.btn_segmentation_image.setStyleSheet("color:
#000000;")
                widgets.btn_segmentation_video.setStyleSheet("color:
#000000;")
                widgets.btn_get_fisheye.setStyleSheet("color:
#000000;")
                widgets.btn_get_common.setStyleSheet("color:
#000000;")
                widgets.btn_raw_to_platte.setStyleSheet("color:
#000000;")
                widgets.btn_raw_to_platte.setStyleSheet("color:
#000000;")
                widgets.btn_generate_traffic.setStyleSheet("color:
#000000;")
                widgets.btn_manual_control.setStyleSheet("color:
#000000;")
                widgets.btn_automatic_control.setStyleSheet("color:
#000000;")


widgets.line_edit_operation_help.setStyleSheet("color: #000000;")
```

```
                #widgets.table_widget_get_image.setStyleSheet("color: #000000;")

self.setTableFontColor(widgets.table_widget_get_image)

self.setTableFontColor(widgets.table_widget_operation_help)

            UIFunctions.theme(self,                    file=theme_file, useCustomTheme=True)
            AppFunctions.setThemeHack(self)

        def handle_btn_send_mail(btn):
            import webbrowser
            to_email = "m0rtzz@outlook.com"
            webbrowser.open("mailto:" + to_email, new=1)

        def handle_btn_print(btn):
            file_dialog = QFileDialog()
            file_dialog.setFileMode(QFileDialog.ExistingFiles)
            options = QFileDialog.Options()
            options |= QFileDialog.DontUseNativeDialog
            filter = "文件 (*.txt *.pdf)"
            files, _ = file_dialog.getOpenFileNames(
                None, "选择文本或 PDF 文件", "./logs", filter, options=options)
            terminal_command_1 = "./scripts/txt2pdf.out " + " ".join(files)
            os.system(terminal_command_1)
            print_files = set(file.replace(".txt", ".pdf") if
file.endswith(
                ".txt") else file.replace(".pdf", ".pdf") for file in
files)
            terminal_command_2 = "pdftk " + " ".join(print_files) + " cat
output - | lpr"
            os.system(terminal_command_2)

        def handle_btn_unlock(btn):
            self.paymentCodeSegment()

        btn_actions = {
            "btn_home": handle_btn_home,
            "btn_widgets": handle_btn_widgets,
            "btn_image": handle_btn_image,
            "btn_simulation": handle_btn_simulation,
            "btn_my_image": handle_btn_my_image,
            "btn_open_dir": handle_btn_open_dir,
            "btn_fisheye_one2one": handle_btn_fisheye_one2one,
            "btn_fisheye_five2one": handle_btn_fisheye_five2one,
            "btn_segmentation_image": handle_btn_segmentation_image,
            "btn_segmentation_video": handle_btn_segmentation_video,
            "btn_raw_to_platte": handle_btn_raw_to_platte,
            "btn_start_server": handle_btn_start_server,
            "btn_generate_traffic": handle_btn_generate_traffic,
            "btn_manual_control": handle_btn_manual_control,
            "btn_automatic_control": handle_btn_automatic_control,
            "btn_get_fisheye": handle_btn_get_fisheye,
            "btn_get_common": handle_btn_get_common,
            "btn_adjustments": handle_btn_adjustments,
            ″btn_send_mail″: handle_btn_send_mail,
            ″btn_print″: handle_btn_print,
            ″btn_unlock″: handle_btn_unlock
        }

        btn = self.sender()
        btn_name = btn.objectName()

        action = btn_actions.get(btn_name)
        if action:
            action(btn)

    def resizeEvent(self, event):
        # Update Size Grips
        UIFunctions.resize_grips(self)

    # MOUSE CLICK EVENTS
    def mousePressEvent(self, event):
        # SET DRAG POS WINDOW
        self.dragPos = event.globalPos()



if __name__ == ″__main__″:
    app = QApplication(sys.argv)
    app.setWindowIcon(QIcon(″./images/icons/icon.ico″))
    os.system(″sl -e″)
    window = MainWindow()
    print(’’
_____
\ \    / / ___     | |     __       ___    _ __      ___       o O O |_   _|
___
 \ \/\/ / / -_)    | |    / _|    / _ \  |'  \   / -_)    o        | |
/ _ \
  \_/\_/  \___|    _|_|_    \__|_    \___/  |_|_|_|  \___|    TS__[O]
_|_|_   \___/
_|"""""|_|"""""|_|"""""|_|"""""|_|"""""|_|"""""|_|"""""|
{======|_|"""""|_|"""""|
"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'./o--000'"`-0
-0-'"`-0-0-'
''')


    print('''
 __    __     _                 _                      __    __           _
_            __ _
 \ \ / /   (_)      ___      (_)      ___    _ _    \ \ / /   ___    | ||
|  __ _    / _` |   ___
  \ V /    | |    (_-<     | |    / _ \  | ' \    \ V /   / _ \    \_,
| / _` |   \__, |  / -_)
  _\_/_    _|_|_    /__/_    _|_|_    \___/  |_||_|    _\_/_    \___/
_|__/  \__,_|   |___/   \___|
_|   """"|_|"""""|_|"""""|_|"""""|_|"""""|_|"""""|_|   """"|_|"""""|_|
""""|_|"""""|_|"""""|_|"""""|
"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-
0-'"`-0-0-'"`-0-0-'"`-0-0-'
''')
    sys.exit(app.exec())
```

②对图片进行语义分割：

```
#!/home/m0rtzz/Program_Files/anaconda3/envs/mmsegmentation/bin/python
3

import argparse
import os
import cv2
from mmseg.apis import inference_model, init_model, show_result_pyplot

# NOTE: 禁止指定警告输出
import warnings
warnings.filterwarnings("ignore",          message="torch.meshgrid*",
category=UserWarning)

# 从命令行获取图片路径参数
parser = argparse.ArgumentParser(description='Image Segmentation')
parser.add_argument('--image_paths',  nargs='+',  help='input  image
paths')
args = parser.parse_args()

normal_config_path                                                  =
'scripts/configs/mask2former/mask2former_swin-t_8xb2-90k_normal.py'
normal_checkpoint_path = 'scripts/weights/normal.pth'

fisheye_config_path                                                 =
'scripts/configs/mask2former/CBAM4_swin-t_fisheye-mean.py'
fisheye_checkpoint_path = 'scripts/weights/fisheye.pth'

config_path = ''
```

```
checkpoint_path = ''
output_dir = 'images/my_images/sem_seg/output/images'



def assignConfigByResolution(img_path):
    img = cv2.imread(img_path)
    height, width, _ = img.shape

    resolution_paths = {
        (2048, 1024): (normal_config_path, normal_checkpoint_path),
        (1280, 966): (fisheye_config_path, fisheye_checkpoint_path),
    }

    default_paths = (normal_config_path, normal_checkpoint_path)

    # 查找匹配的分辨率路径，如果没有匹配则使用默认路径
    config_path,   checkpoint_path   =   resolution_paths.get((width,
height), default_paths)

    return config_path, checkpoint_path



if __name__ == '__main__':
    # 遍历输入图片路径列表
    for img_path in args.image_paths:
        if img_path.endswith('.png') or img_path.endswith('.jpg'):
            # 构建输出文件路径
            config_path,               checkpoint_path               =
assignConfigByResolution(img_path)
```

```
            filename = os.path.basename(img_path)
            output_path = os.path.join(output_dir, filename)

            # 从配置文件和权重文件构建模型
            model    =    init_model(config_path,    checkpoint_path,
device=’cuda:0’)

            # 推理给定图像
            result = inference_model(model, img_path)

            # 保存可视化结果
            show_result_pyplot(model,  img_path,  result,  show=False,
save_dir=output_dir, out_file=output_path)

            cv2.namedWindow(″Press  ESC  to  exit″,  cv2.WINDOW_NORMAL  |
cv2.WINDOW_GUI_NORMAL)
            cv2.resizeWindow(″Press ESC to exit″, 800, 600)  # 设置窗
口大小为 800x600
            mat = cv2.imread(output_path)
            cv2.imshow(″Press ESC to exit″, mat)
            while True:
                if cv2.waitKey(1) == 27:  # 按下 esc 键的 ASCII 码为 27
                    cv2.destroyWindow(″Press ESC to exit”)
                    break
```

个人遇到的主要问题及解决方法：

1. 路径和文件名错误

问题

QCoreApplication.setLibraryPaths() 和 qInitResources() 所使用的路径可能不正确，导致资源无法正确加载。

os.path.abspath(__file__) 获取的路径在某些情况下可能不正确，特别是在打包或跨平台使用时。

解决方法

确保路径的正确性，使用相对路径来确保跨平台兼容。

使用环境变量或配置文件来动态设置路径。

```
current_dir = os.path.dirname(os.path.abspath(__file__))
QCoreApplication.setLibraryPaths([os.path.join(current_dir, "lib")])
```

2. 模块和库导入问题

问题

需要导入的模块和库可能在路径中找不到，导致 ImportError。

解决方法

确保所有需要的模块和库都在 PYTHONPATH 中。

使用虚拟环境并安装所需依赖项。

```
try:
    from sensors.fisheye_capture import FisheyeCapture
    from backends.common.cli_parser import create_parser, add_logging_arguments
    import backends.common.resources
    from utilities.logger import get_logger
```

```
except ImportError as e:
    print(f"Error importing modules: {e}")
    sys.exit(1)
```

3. 未处理的异常

问题

代码中未处理的异常可能导致程序崩溃。

解决方法

在关键部分添加异常处理逻辑，确保异常能够被捕获并处理。

```
try:
    # 关键代码段
except Exception as e:
    print(f"An error occurred: {e}")
    sys.exit(1)
```

4. 资源初始化问题

问题

qInitResources() 函数可能无法正确初始化所有资源，导致后续使用资源时出错。

解决方法

确保资源文件路径正确，并且资源文件已经被正确加载。

```
try:
    qInitResources()
except Exception as e:
    print(f"Resource initialization failed: {e}")
    sys.exit(1)
```

5. 线程问题

问题

如果 FisheyeCapture 或其他组件在单独的线程中运行，可能会出现线程同步问题。

解决方法

确保所有线程操作是线程安全的，使用线程锁或其他同步机制。

```
import threading


capture_lock = threading.Lock()


def start_capture():
    with capture_lock:
        capture = FisheyeCapture(args.device)
        capture.start()
```

6. 平台相关问题

问题

在不同操作系统上，库和资源的路径及权限可能会有所不同，导致不兼容问题。

解决方法

使用平台检测代码，并根据不同平台设置不同的路径或库。

```
import platform


if platform.system() == "Windows":
    # Windows 相关设置
elif platform.system() == "Linux":
    # Linux 相关设置
```

7. 性能问题

问题

在处理高分辨率视频或复杂的计算任务时，可能会出现性能瓶颈。

解决方法

优化代码，使用并行计算或硬件加速。

```
import multiprocessing


def process_frame(frame):
    # 帧处理代码
    pass


with multiprocessing.Pool(processes=4) as pool:
    results = pool.map(process_frame, frames)
```

8. 依赖项版本问题

问题

依赖的第三方库版本不兼容，导致功能异常或缺失。

解决方法

使用 requirements.txt 或 Pipfile 锁定依赖项版本，并定期更新。

```
# requirements.txt
PyQt6==6.2.0
some_other_library==1.4.2
```

9. 用户权限问题

问题

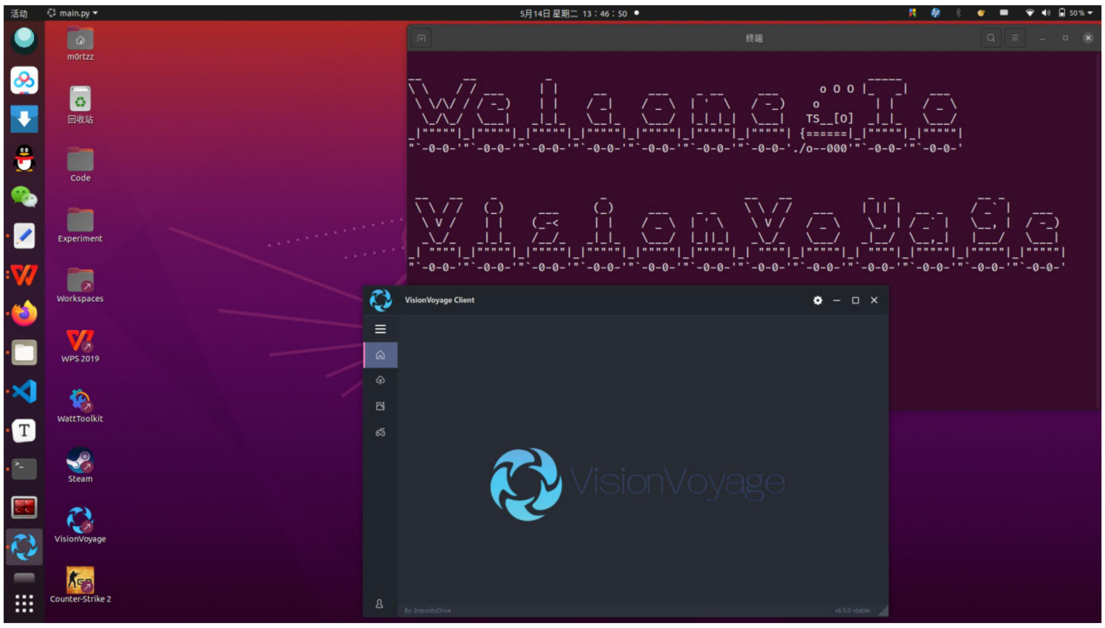
在某些操作系统上，访问某些资源（如摄像头、GPU）需要特定权限。

解决方法

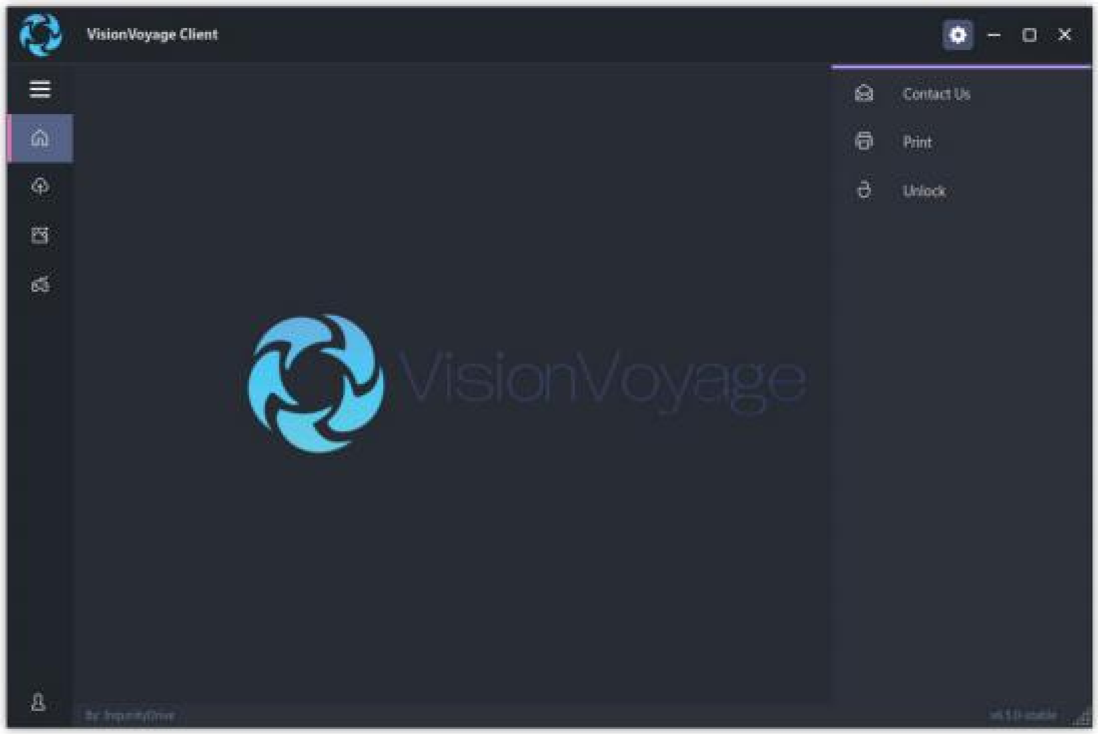
提前检查并申请所需权限。

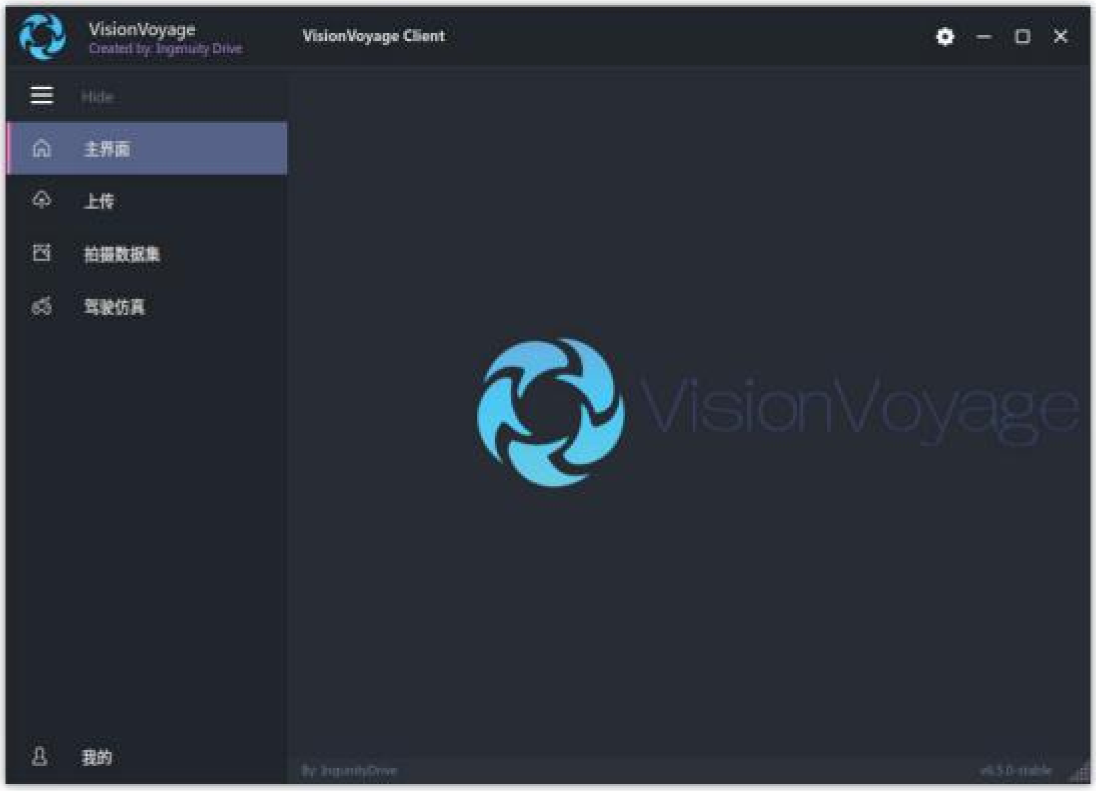
```
import os


if not os.access("/dev/video0", os.R_OK):
    print("No read access to the camera device.")
    sys.exit(1)
```

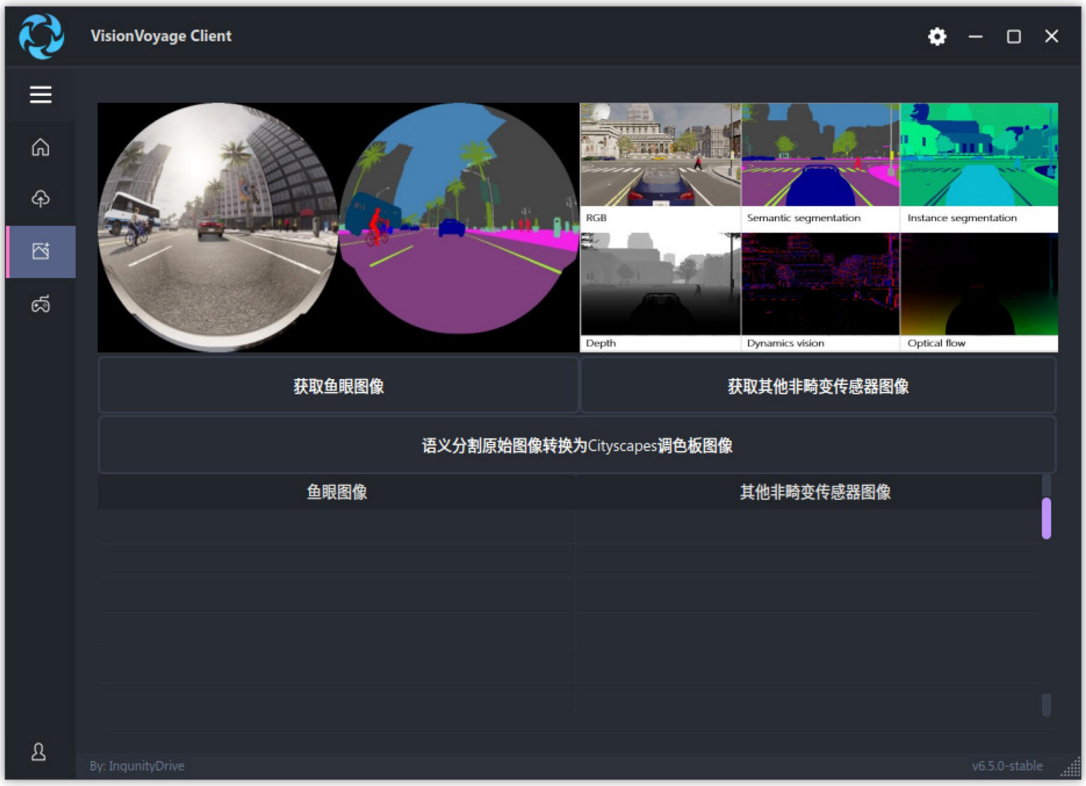
项目效果图

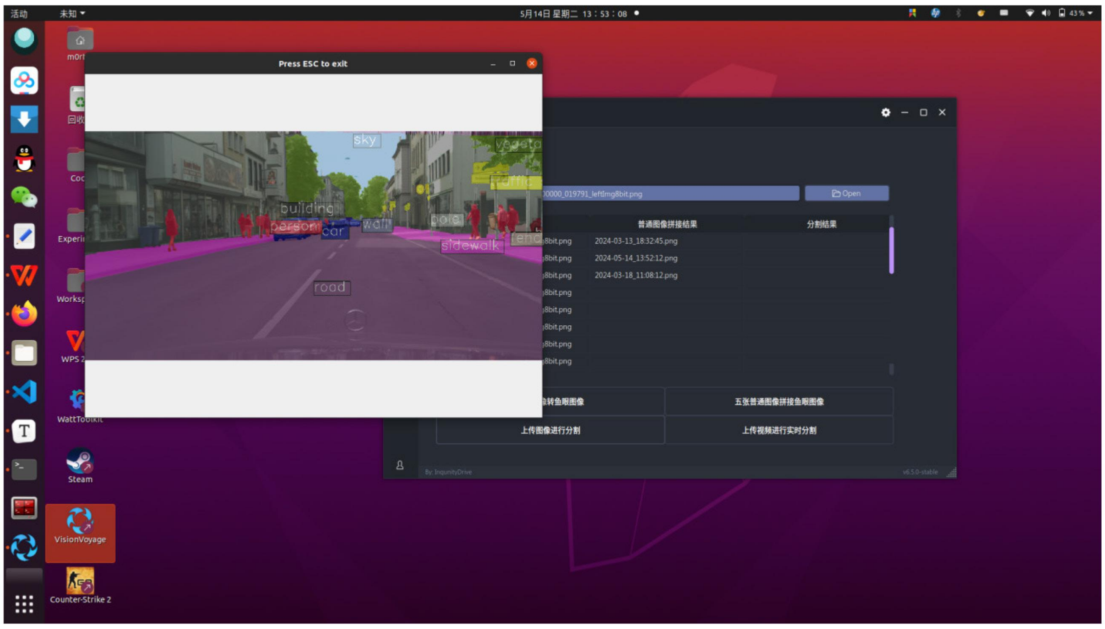


```
__      __        _                                         _____
\ \    / /___    | |    __    ___   _ __    ___     o O O |_   _|   ___
 \ \/\/ // -_)   | |   / _|  / _ \ | '  \  / -_)   o        | |    / _ \
  \_/\_/ \___|  _|_|_  \__|_ \___/ |_|_|_| \___|  TS__[O]  _|_|_   \___/
_|"""""|_|"""""|_|"""""|_|"""""|_|"""""|_|"""""|_|"""""| {======|_|"""""|_|"""""|
"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'./o--000'"`-0-0-'"`-0-0-'


__   __   _              _                   __   __           _  _              __ _
\ \ / /  (_)    ___     (_)    ___    _ _    \ \ / /   ___    | || |   __ _     / _` |   ___
 \ V /   | |   (_-<     | |   / _ \  | ' \    \ V /   / _ \    \_, |  / _` |    \__, |  / -_)
  \_/   _|_|_  /__/_   _|_|_  \___/  |_||_|   _\_/_   \___/   _|__/   \__,_|    |___/   \___|
_| """"|_|"""""|_|"""""|_|"""""|_|"""""|_|"""""|_| """"|_|"""""|_| """"|_|"""""|_|"""""|_|"""""|
"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'
```


VisionVoyage Client
Contact Us
Print
Unlock
VisionVoyage
By: IngenuityDrive
v6.5.0-stable

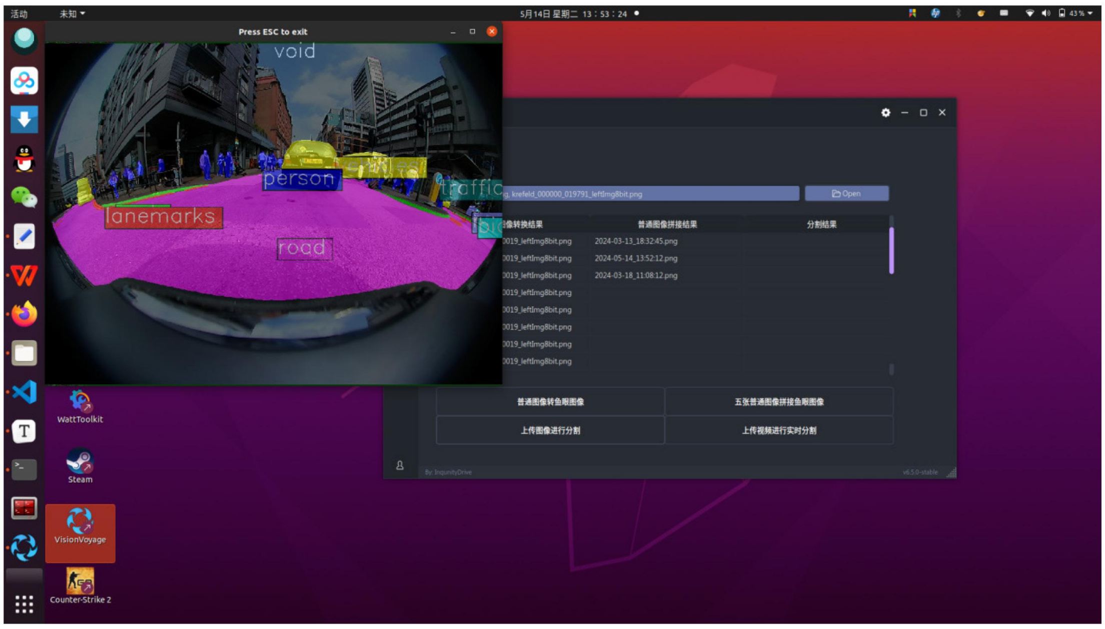

VisionVoyage
Created by: Ingenuity Drive
VisionVoyage Client
Hide
主界面
上传
拍摄数据集
驾驶仿真
我的
VisionVoyage
By: IngenuityDrive
v6.5.0-stable

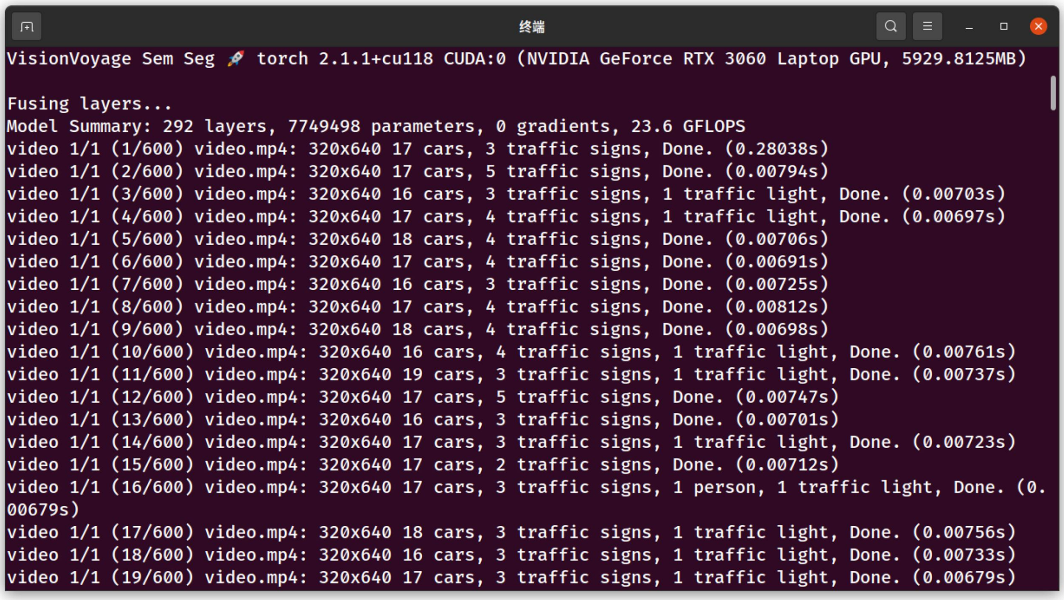




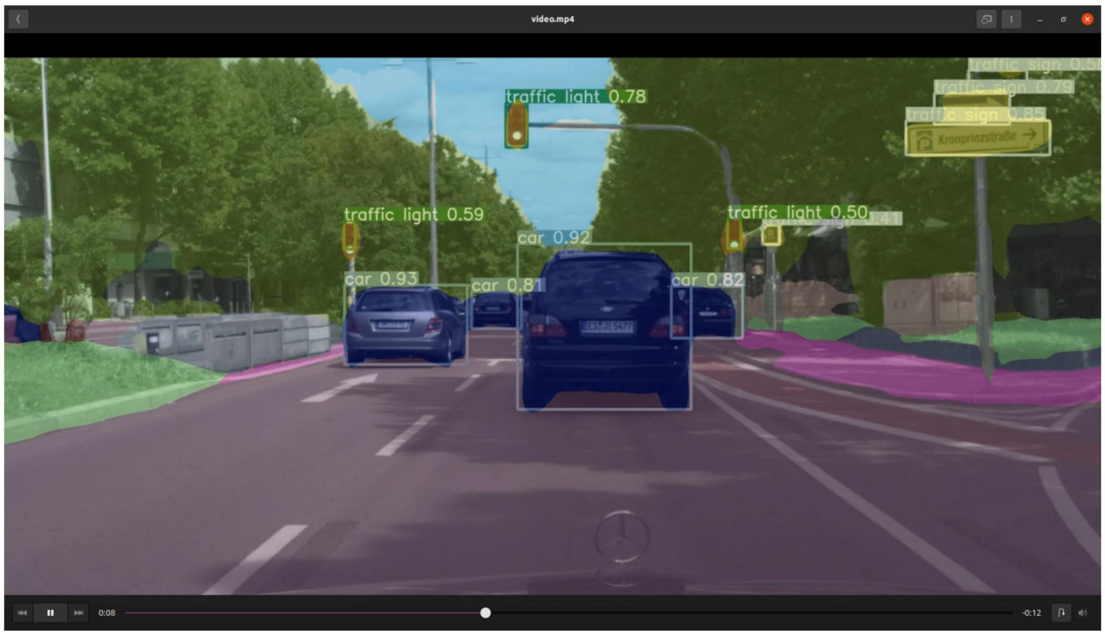
00000_019791_leftImg8bit.png

	普通图像拼接结果	分割结果
8bit.png	2024-03-13_18:32:45.png	
8bit.png	2024-05-14_13:52:12.png	
8bit.png	2024-03-18_11:08:12.png	
8bit.png		
8bit.png		
8bit.png		
8bit.png		
8bit.png		





```
VisionVoyage Sem Seg 🚀 torch 2.1.1+cu118 CUDA:0 (NVIDIA GeForce RTX 3060 Laptop GPU, 5929.8125MB)

Fusing layers...
Model Summary: 292 layers, 7749498 parameters, 0 gradients, 23.6 GFLOPS
video 1/1 (1/600) video.mp4: 320x640 17 cars, 3 traffic signs, Done. (0.28038s)
video 1/1 (2/600) video.mp4: 320x640 17 cars, 5 traffic signs, Done. (0.00794s)
video 1/1 (3/600) video.mp4: 320x640 16 cars, 3 traffic signs, 1 traffic light, Done. (0.00703s)
video 1/1 (4/600) video.mp4: 320x640 17 cars, 4 traffic signs, 1 traffic light, Done. (0.00697s)
video 1/1 (5/600) video.mp4: 320x640 18 cars, 4 traffic signs, Done. (0.00706s)
video 1/1 (6/600) video.mp4: 320x640 17 cars, 4 traffic signs, Done. (0.00691s)
video 1/1 (7/600) video.mp4: 320x640 16 cars, 3 traffic signs, Done. (0.00725s)
video 1/1 (8/600) video.mp4: 320x640 17 cars, 4 traffic signs, Done. (0.00812s)
video 1/1 (9/600) video.mp4: 320x640 18 cars, 4 traffic signs, Done. (0.00698s)
video 1/1 (10/600) video.mp4: 320x640 16 cars, 4 traffic signs, 1 traffic light, Done. (0.00761s)
video 1/1 (11/600) video.mp4: 320x640 19 cars, 3 traffic signs, 1 traffic light, Done. (0.00737s)
video 1/1 (12/600) video.mp4: 320x640 17 cars, 5 traffic signs, Done. (0.00747s)
video 1/1 (13/600) video.mp4: 320x640 16 cars, 3 traffic signs, Done. (0.00701s)
video 1/1 (14/600) video.mp4: 320x640 17 cars, 3 traffic signs, 1 traffic light, Done. (0.00723s)
video 1/1 (15/600) video.mp4: 320x640 17 cars, 2 traffic signs, Done. (0.00712s)
video 1/1 (16/600) video.mp4: 320x640 17 cars, 3 traffic signs, 1 person, 1 traffic light, Done. (0.
00679s)
video 1/1 (17/600) video.mp4: 320x640 18 cars, 3 traffic signs, 1 traffic light, Done. (0.00756s)
video 1/1 (18/600) video.mp4: 320x640 16 cars, 3 traffic signs, 1 traffic light, Done. (0.00733s)
video 1/1 (19/600) video.mp4: 320x640 17 cars, 3 traffic signs, 1 traffic light, Done. (0.00679s)
```



个人实习体会及收获：

在 VisionVoyage 项目开发的整个过程中，我们团队展现了卓越的协调能力和专业素养。每位成员都肩负起了明确的职责，通过设计、开发、测试和优化各个环节，逐步提升了项目的整体质量，确保项目按时达成预期目标。这个过程中，我们不仅深化了专业技术的理解，还掌握了敏捷开发的方法，涵盖团队合作、计划制定和文档撰写等关键环节。这些经验将为我们未来的职业生涯打下坚实基础。

在项目实施过程中，我们深刻体会到团队合作的力量。团队精神促使我们目标一致，形成了强大的凝聚力和战斗力。通过明确分工，我们将整体目标细化为每个成员的具体任务，确保了工作的高效推进。正是这种协同合作，使我们能够在项目中不断创新并实现这些创新，最终达成共识。

我们还认识到，团队中的多样性是推动项目进展的重要因素。我们学会了在不同观点中寻求共识，通过充分的讨论和协商，找到最佳的解决方案。在项目开发过程中，我们迎接了诸多挑战，每一位成员都展示了坚韧不拔的精神和创新的勇气。我们从中总结了宝贵的经验教训，为未来的个人成长和团队发展注入了新的动力。

通过这次项目的开发，我们深刻认识到，软件开发过程从来不会一帆风顺。每个成员都是一个独立的个体，有着各自的思维和判断。作为一个整体的团队，我们学会了包容，学会了在差异中寻求共识，通过讨论和协商获得最佳的解决方案。这次项目的成功不仅满足了用户的核心需求，也为我们积累了宝贵的经验，为未来的发展奠定了坚实的基础。

总体而言，VisionVoyage 项目的开发对我们每一位成员来说，都是一次宝贵的学习和成长经历。我们不仅提升了专业技能，还在团队合作、创新思维和问题解决能力方面取得了显著进步。我们相信，这次项目的成功经验将为我们的未来发展提供强大的动力和支持。

生产实习成绩汇总

生产实习分数	平时成绩（20分）		项目成绩（60分）		报告成绩（20分）		总分
	出勤情况 (10 分)	课堂表现 (10 分)	基本功能 (30 分)	答辩情况 (30 分)	报告内容 (10 分)	规范程度 (10 分)	

总体评价：

实习导师签名：

企业导师签名：