



计算机视觉（本科）实验报告

作业名称 制作小短片视频（HW1）

姓 名 杨逍宇

学 号 3220105453

电子邮箱 1877114891@qq.com

联系电话 13518290755

导 师 潘纲



2024 年 11 月 17 日

1 已实现的功能简述及运行简要说明

1.1 已实现的功能简述:

- (1). 程序运行之后, 会将当前计算机摄像头内容实时显示, 并录制下来, 关闭后保存为视频文件。
- (2). 在摄像头显示与生成的视频中, 在画面的底部加一行 Hello OpenCV 的字幕。
- (3). 在摄像头显示与生成的视频中, 在画面右上角加上当前时间的显示。
- (4). 在摄像头显示与生成的视频中, 在画面左上角一直显示以我的个人小照片的 logo, logo 下面显示我的姓名 yxy。

1.2 运行简要说明:

- (1). 其中可执行文件 hw_1.exe 在目录 build 下, 在 ubuntu 环境下可直接运行。
- (2). 运行程序后, 摄像头内容被实时显示并录制下来, 按下'q' 键可以退出程序。程序退出后, 输出文件会保存在.asset 文件目录下。

2 开发与运行环境

本实验使用的软件和工具如下:

- 开发环境: Visual Studio Code on Ubuntu22.04
- 编程语言: C++
- 库: OpenCV 4.7.0
- 构建工具: CMake

3 算法实现步骤

本实验的算法步骤如下:

- (1). 初始化视频捕获对象和视频写入对象。
- (2). 读取图标图像并调整其尺寸。
- (3). 进入视频录制循环:

- (a) 从摄像头捕获一帧图像。
 - (b) 检查帧是否为空，若为空则退出循环。
 - (c) 获取当前时间并显示在帧的右上角。
 - (d) 在帧的底部添加字幕。
 - (e) 在帧的左上角添加图标和用户名。
 - (f) 将处理后的帧写入视频文件。
 - (g) 显示处理后的帧。
 - (h) 检查用户是否按下'q' 键，若按下则退出循环。
- (4). 释放摄像头和视频写入器资源。
- (5). 打印输出文件名，提示用户生成的视频文件位置。

4 实现要点

- 使用 `cv::VideoCapture` 对象捕获视频帧。

```
1 // 初始化摄像头
2 cv::VideoCapture cap(0);
3 if (!cap.isOpened()) {
4     std::cerr << "Error: Could not open camera" << std::endl;
5     return -1;
6 }
```

- 使用 `cv::VideoWriter` 对象写入视频文件。

```
1 // 创建视频写入器
2 int frame_width = static_cast<int>(cap.get(cv::CAP_PROP_FRAME_
    WIDTH));
3 int frame_height = static_cast<int>(cap.get(cv::CAP_PROP_FRAME_
    HEIGHT));
4 cv::Size frameSize(frame_width, frame_height); // 定义
    frameSize变量
5 int fourcc = cv::VideoWriter::fourcc('m', 'p', '4', 'v'); // 使用
    'mp4v'编解码器
6 cv::VideoWriter out(output_file, fourcc, 20.0, frameSize);
```

- 使用 `cv::putText` 在帧上添加时间、字幕和用户名。

```
1 // 添加时间显示 (右上角)
2 now = time(0);
3 ltm = localtime(&now);
4 strftime(current_time, sizeof(current_time), "%H:%M:%S", ltm);
5 cv::putText(frame, current_time, cv::Point(frame.cols - 150, 20),
6           font, 1, cv::Scalar(255, 255, 255), 2);
7
8 // 添加字幕 (底部)
9 std::string subtitle = "Hello OpenCV";
10 cv::putText(frame, subtitle, cv::Point(50, frame.rows - 30), font,
11           1, cv::Scalar(255, 255, 255), 2);
```

- 使用 `cv::Mat::copyTo` 将图标复制到帧的指定位置。

```
1 // 添加图标和用户名 (左上角)
2 if (!logo.empty()) {
3     logo.copyTo(frame(cv::Rect(0, 0, logo.cols, logo.rows)));
4 }
5 cv::putText(frame, username, cv::Point(10, logo.rows + 30), font,
6           1, cv::Scalar(255, 255, 255), 2);
```

- 使用 `cv::imshow` 显示处理后的帧，并使用 `cv::waitKey` 检查用户输入。

```
1 // 将帧写入视频文件
2 out.write(frame);
3
4 // 显示帧
5 cv::imshow("Camera", frame);
6 if (cv::waitKey(1) == 'q') { // 按下 'q' 键退出录制
7     break;
8 }
```

5 实验结果及分析

实验结果如下图所示：



Figure 1: 处理后的视频帧

从图中可以看到，视频帧的右上角显示了当前时间，底部添加了字幕，左上角添加了图标和用户名。实验结果表明，算法能够正确地处理视频帧并添加所需的元素。

6 结论与心得体会

通过本次实验，我学到了以下几点：

- 熟悉了 OpenCV 库的基本使用方法，特别是视频捕获和处理功能。
- 掌握了如何在视频帧上添加文本和图像。
- 了解了如何使用 CMake 构建 C++ 项目。
- 提高了调试和解决问题的能力。