ЛАБОРАТОРНАЯ РАБОТА №5
ВЫСОКОУРОВНЕВАЯ РАБОТА С ПЕРИФЕРИЙНЫМИ УСТРОЙСТВАМИ

Цели работы

1. Ознакомиться с программированием периферийных устройств на примере ввода данных с Web-камеры с использованием библиотеки OpenCV.

1. БИБЛИОТЕКА OPENCV

Библиотека OpenCV предоставляет высокоуровневый интерфейс для разработки прикладных программ в области машинного зрения. OpenCV включает следующие группы функций: работа с растровыми изображениями и их форматами, работа с видеоданными и их форматами, интерфейс для работы с камерами, машинное обучение с помощью нейронных сетей и других моделей, реализация упрощенного оконного интерфейса, операции над векторами и матрицами и другие.

2. ВЫВОД ВИДЕОПОТОКА С КАМЕРЫ НА ЭКРАН С ПОМОЩЬЮ БИБЛИОТЕКИ OPENCV

Рассмотрим последовательность деклараций и действий в программе, которая вводит изображение с камеры или из файла и показывает его в окне.

1. Создается поток ввода видеоданных с первой камеры (нумерация начинается с нуля). Результат операции – указатель на дескриптор обрабатываемого потока видеоданных.

```
CvCapture *capture = cvCreateCameraCapture(0);
if (!capture) return 0;
```

2. Запускается циклическая обработка потока видеоданных.

```
while(1) {
```

3. На каждой итерации извлекается очередной кадр (изображение) из открытого потока ввода видеоданных. Результат операции – указатель на дескриптор растрового изображения, которое будет содержать один кадр из потока видеоданных.

```
IplImage *frame = cvQueryFrame(capture);
if(!frame) break;
```

4. Текущий кадр выводится в окно с именем test (при первом вызове создается окно с таким именем):

```
cvShowImage("test", frame);
```

5. В течение 33 миллисекунд ожидается нажатие клавиши пользователем (если клавиша не была нажата, вызов срабатывает как задержка на 33 мс, что обеспечивает частоту показа примерно 30 кадров в сек):

```
char c = cvWaitKey(33);
```

6. Если пользователь нажал клавишу Esc, выйти из цикла обработки и показа кадров видеопотока:

```
if(c == 27) break;
```

7. Перейти к следующей итерации цикла.

```
}
```

8. В эту точку программы можно попасть в двух случаях: во-первых, когда в видеопотоке больше нет кадров (конец видео); во-вторых, когда пользователь нажал клавишу Esc. Здесь производится удаление потока ввода видеоданных, освобождение занятых им ресурсов, а также удаление окна, в которое выводились кадры потока видеоданных:

```
cvReleaseCapture(&capture);
cvDestroyWindow("test");
```

Пример программы ввода изображения с камеры и показа в окне:

```
#include <opencv2/highgui/highgui.hpp>
int main(int argc,char *argv[])
{ CvCapture *capture = cvCreateCameraCapture(0);
  if (!capture) return 0;
```

```
  while(1) {
    IplImage *frame = cvQueryFrame(capture);
    if(!frame) break;
    cvShowImage("test", frame);
    char c = cvWaitKey(33);
    if(c == 27) break;
  }
  cvReleaseCapture(&capture);
  cvDestroyWindow("test");
}
```

Пример команды компиляции программы с использованием OpenCV:

```
gcc -o prog prog.c –lopencv_core –lopencv_highgui
```

3. ПРЕОБРАЗОВАНИЕ ИЗОБРАЖЕНИЙ С ПОМОЩЬЮ БИБЛИОТЕКИ OPENCV

Библиотека OpenCV предоставляет набор функций, позволяющих выполнять различные преобразования изображений, например, сглаживание, морфологические преобразования (сужение и расширение), пороговые преобразования и другие. Следующий пример демонстрирует создание копии изображения `frame`, выполнение над ним библиотечной операции простого сглаживания и вывод на экран в окне:

```
IplImage *image = cvCloneImage(frame);
cvSmooth(frame,image,CV_BLUR,3,3);
cvShowImage(“smooth”,image);
```

Кроме того, OpenCV предоставляет прямой доступ к данным изображения, что позволяет производить редактирование изображения «вручную». Следующие поля дескриптора изображения (структуры `IplImage`) необходимы для корректного доступа к данным изображения:

- nChannels – число цветовых каналов,

- depth – глубина цвета в битах,
- width – ширина изображения в пикселях,
- height – высота изображения в пикселях,
- widthStep – расстояние между данными соседних пикселей по вертикали в байтах,
- imageData – указатель на данные изображения.

Данные изображения представляют собой массив пикселей с числом строк height и числом столбцов width, начинающийся с адреса imageData. Объем данных для одного пикселя в битах определен как nChannels*2^depth.

Далее приводится пример, в котором происходит обнуление данных красного и синего каналов изображения (nChannels: 3, depth: 8):

```
for (y=0; y<image->height; y++) {
  uchar *ptr = (uchar*)(image->imageData +
                        y*image->widthStep);
  for (x=0; x<image->width; x++) {
    ptr[3*x  ] = 0; // Blue
    ptr[3*x+2] = 0; // Red
  }
}
```

4. ЗАДАНИЕ К ЛАБОРАТОРНОЙ РАБОТЕ

1. Реализовать программу с использованием OpenCV, которая получает поток видеоданных с камеры и выводит его на экран.
2. Выполнить произвольное преобразование изображения.
3. Измерить количество кадров, обрабатываемое программой в секунду. Оценить долю времени, затрачиваемого процессором на обработку (ввод, преобразование, показ) видеоданных, получаемых с камеры.
4. Составить отчет по лабораторной работе. Отчет должен содержать следующее:

- Титульный лист.
- Цель лабораторной работы.
- Полный компилируемый листинг реализованной программы и команды для ее компиляции.
- Оценку скорости обработки видео (кадров в секунду) и долю времени, затрачиваемого процессором на обработку (ввод, показ) видеоданных.
- Вывод по результатам лабораторной работы.

5. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое OpenCV? Какие задачи решает OpenCV?
2. Как с помощью OpenCV получить изображение с видеокамеры и вывести его на экран?
3. Какие способы преобразования изображений можно использовать, применяя OpenCV.