Dokumentacja praktyki zawodowej

Jakub Cichowicz 38625

Zadanie 1.

Mechanizm śledzenia położenia twarzy zrealizowany został za pomocą języka Python z wykorzystaniem biblioteki OpenCV. Program pozyskuje obraz z kamery i na pojedynczej klatce obrazu poszukuje twarzy za pomocą klasyfikatora kaskad Haar’a. Algorytm poszukuje na obrazie cech Haar’a, które zawierają informacje o zmianie wartości kontrastu pomiędzy prostokątnymi grupami pikseli. Zmiany te są wykorzystywane do określenia relatywnie jasnych i ciemnych obszarów. Dwie lub trzy sąsiadujące grupy o podobnej wariancji kontrastu tworzą cechę Haar’a. Wykryta twarz zaznaczana jest na obrazie czerwonym prostokątem, a następnie obliczany jest środek tego prostokąta, którego współrzędne konwertowane są do układu odniesienia z punktem 0,0 na środku obrazu. Współrzędne wykrytej twarzy porównywane są z poprzednią ramką i na tej podstawie wyznaczane są wartości przesunięć poziomych, pionowych oraz wartości euklidesowej. Wartości te prezentowane są na osobnych wykresach za pomocą biblioteki pyqtygraph.

Zadanie 2.

Mechanizm śledzenia punktów charakterystycznych twarzy został zrealizowany za pomocą języka Python z wykorzystaniem biblioteki Dlib. Pierwszym krokiem programu jest wykrycie twarzy za pomocą detektora wbudowanego w bibliotekę. Detektor opiera się o histogramy zorientowanych gradientów w połączeniu z klasyfikatorem liniowym. Następnie w wyznaczonym obszarze poszukiwane jest 68 punktów charakterystycznych twarzy za pomocą predyktora działającego na zasadzie drzew regresji, który zawarty jest w bibliotece Dlib. Kolejnym krokiem jest zaznaczenie obrysu twarzy za pomocą prostokąta w celu utworzenia układu odniesienia dla 40 punktów charakterystycznych znajdujących się wewnątrz obrysu twarzy. Punkty te śledzone są na obrazie i zapisywane w pliku tekstowym wraz z sygnaturą czasową. Część zadania z zapisywaniem nagrania wideo nie została wykonana ze względu na problemy z kodekiem, przez co odtworzenie zapisanego pliku wideo było niemożliwe.

Użyte zasoby: Python 3.6.8, OpenCV, Numpy, PyQt5, PyQtyGraph, CMake, Dlib