Dmytro Matviichuk 240545 Termin: pt TN 17:15-20:15

Konrad Kieda 250997

**Urządzenia peryferyjne**

**Temat: Obsługa kamery USB**

Kod źródłowy(GitHub ): https://github.com/dmatviichuk/Urzadzenia\_Peryferyjne

**1. Cel ćwiczenia**

Celem ćwiczenia było napisanie programu, który wykryje urządzenia typu kamera, podłączone do komputera, by następnie móc wyświetlić i przetwarzać w aplikacji obrazy, przechwytywane z dwóch, wybranych przez użytkownika urządzeń. Również za pomocą aplikacji powinno dać się zmieniać opcje kamery (nasycenie, kontrast, jaskrawość) oraz aplikacja powinna umożliwiać zapis filmu w postaci AVI oraz robienie zdjęć w dowolnym formacie.

**2. Wstęp**

Kamera cyfrowa jest urządzeniem rejestrującym obraz i dźwięk na nośniku cyfrowym. Kamery mogą występować w różnej postaci, gdy współpracują z komputerem najczęściej komunikują się z nim za pomocą portu USB lub są jego wbudowaną częścią. W naszym programie sprawdzaliśmy czy obraz się zmienia, jest to swego rodzaju wstęp do wykorzystywania kamer w przemyśle.

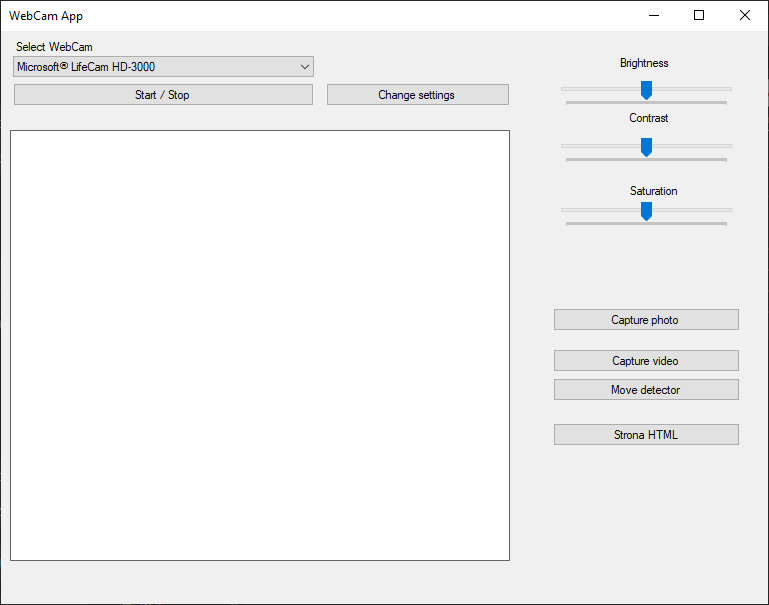
**3. Infromancja o programie oraz fragmenty kodu**Program został napisany w technologii C#, z wykorzystaniem open source’owego framework’u AForge.NET, zaprojektowanego specjalnie z myślą o przetwarzaniu obrazów, sieciach neuronowych, robotyce i tym podobnych gałęziach nauk technicznych. *Biblioteki które zostały użyte w naszej aplikacji:*  
using Aforge;  
using AForge.Video; //przetwarzanie video  
using Aforge.Video.FFMPEG;  
using Aforge.Video.VFW;  
using Aforge.Video.DirectShow;  
using AForge.Imaging.Filters; //przetwarzanie obrazu, filtry

*Funkcja która odpowiada za wykrycie kamer:*  
private void MyForm\_Load(object sender, EventArgs e) {  
 // get the collection of video input devices  
 videoDevices = new FilterInfoCollection(FilterCategory.VideoInputDevice)  
 // list these devices in the combobox  
 foreach (FilterInfo device in videoDevices)  
 {  
 comboBoxDevices.Items.Add(device.Name);  
 }  
 videoSource = new VideoCaptureDevice();  
 comboBoxDevices.SelectedIndex = 0; // default selected item will be the first device  
 }

*Funkcja przechwytująca i wyświetlająca klatki kamery:*void videoSource\_NewFrame(object sender, NewFrameEventArgs eventArgs)  
{ pictureBoxOutput.Image = (Bitmap)eventArgs.Frame.Clone();  
 BrightnessCorrection br = new BrightnessCorrection(brightness1);  
 ContrastCorrection cr = new ContrastCorrection(contrast1);  
 SaturationCorrection sr = new SaturationCorrection(saturation1);   
...//}

**4. Działanie aplikacji**

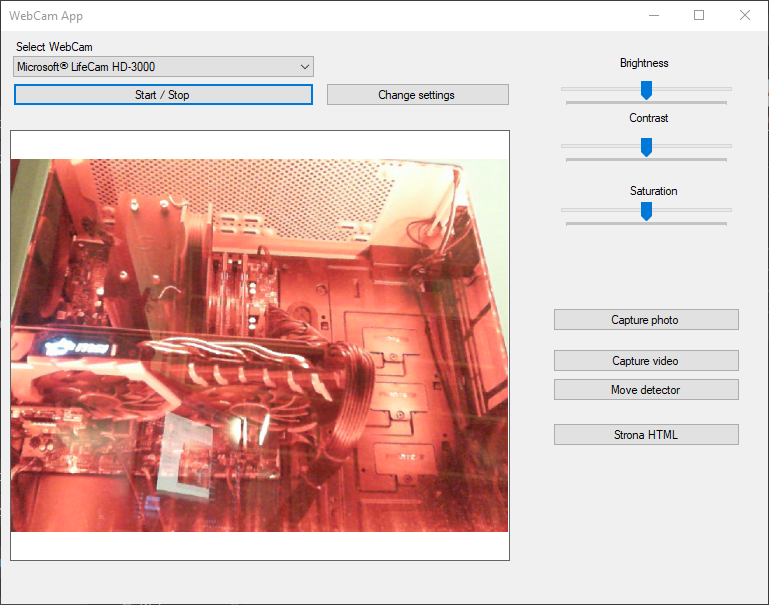
4.1 Tak wygłąda aplikcja po otwarciu:



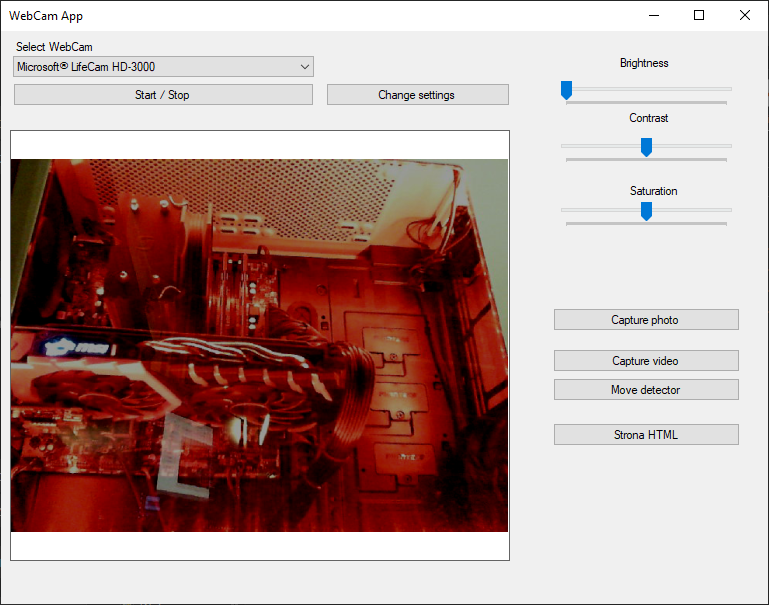
4.2 Program autamatycznie wykrywa wszystkie kamery, wybieramy kamerę a potem klikamy *Start.*

4.3 Przykłady działania programu:

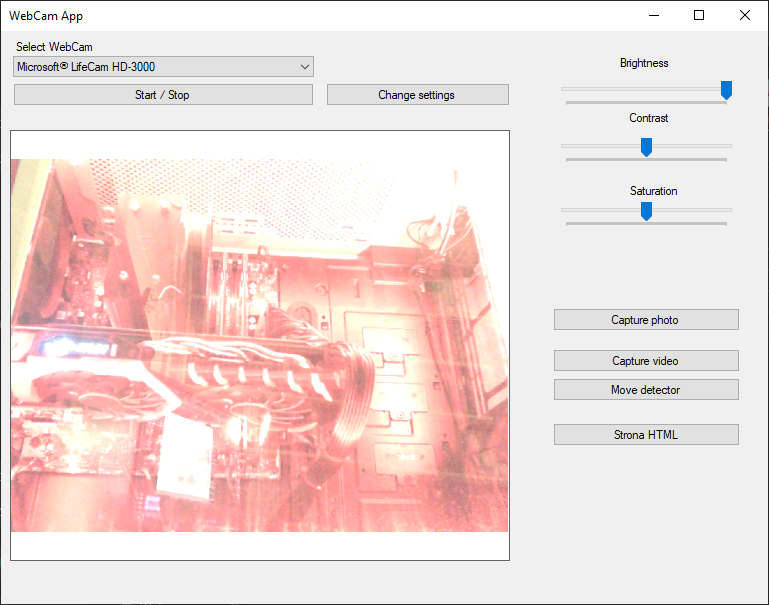
* Obraz bez modyfikatorów



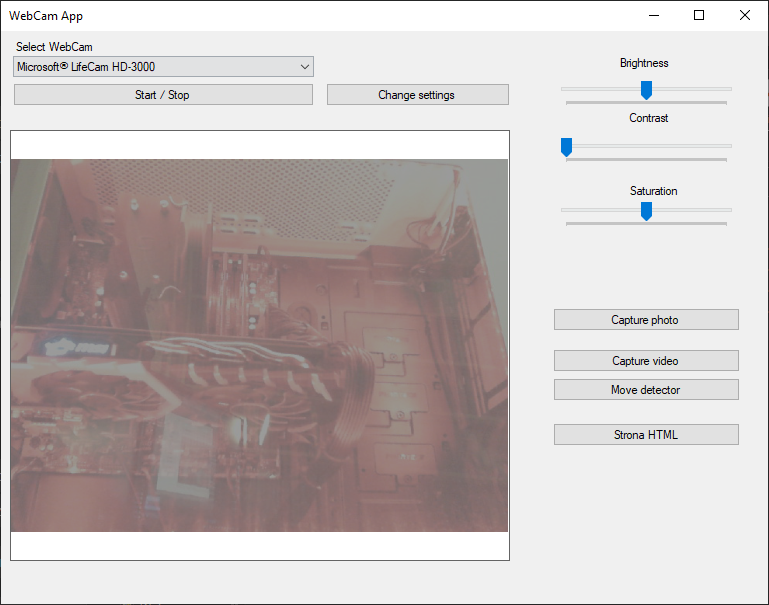
* Minimalna jasność



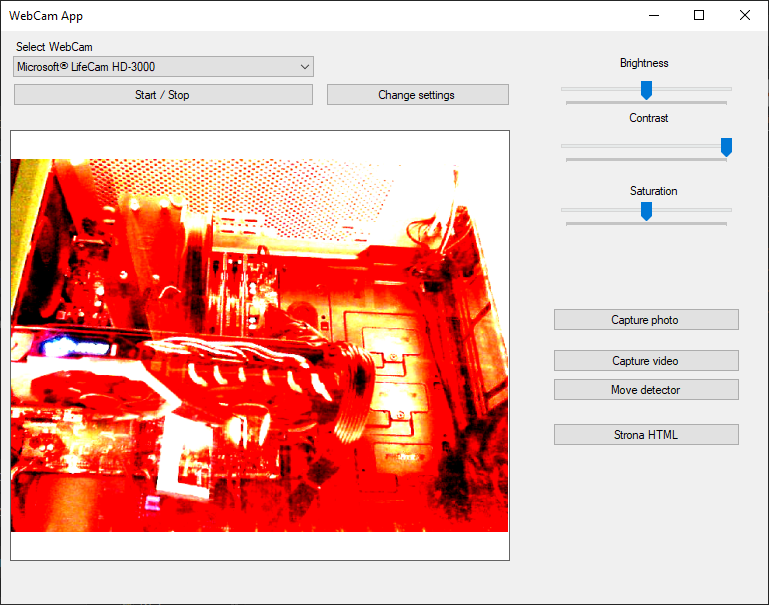
* Maksymalna jasność



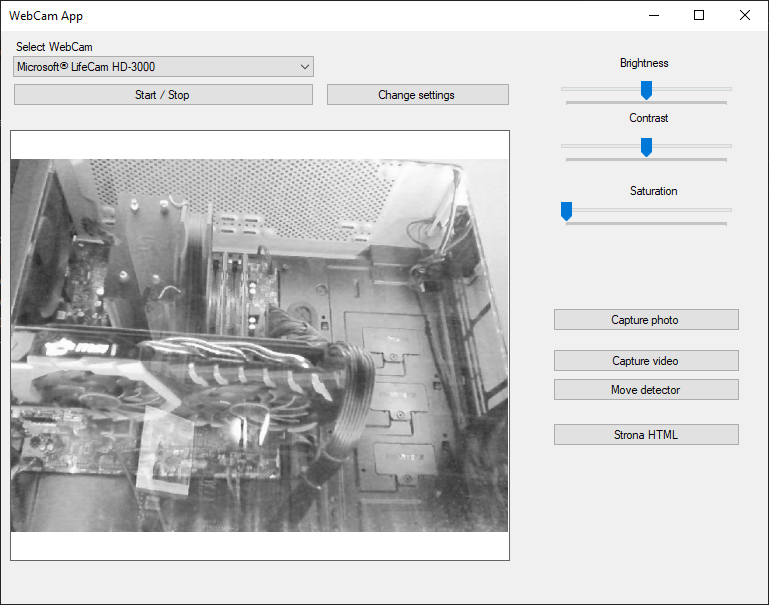
* Minimalny kontrast



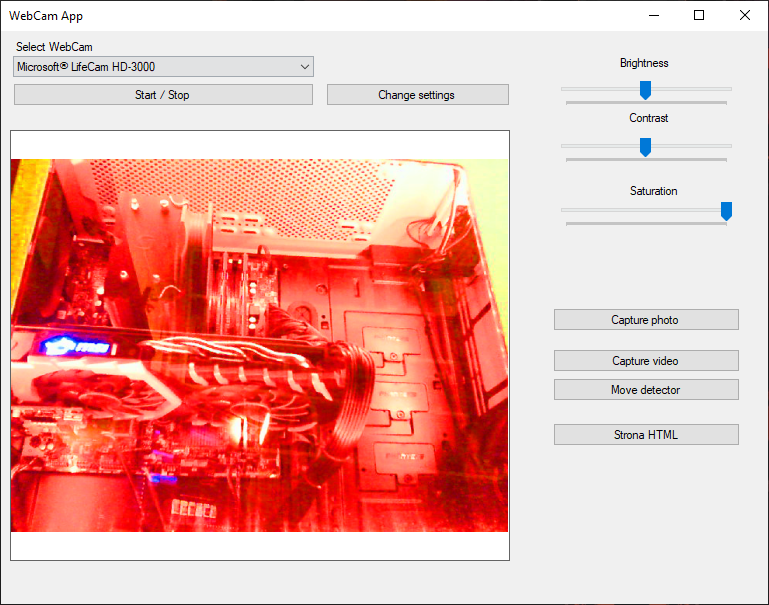
* Maksymalny kontrast



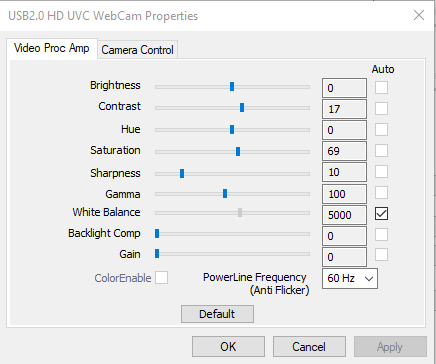
* Minimalna saturacja



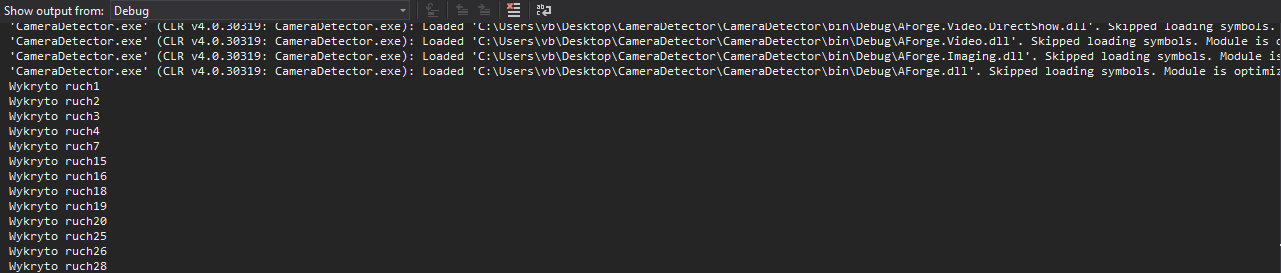
* Maksymalna saturacja



4.4 Po naciśnięciu przycisku *Change settings* otwierają się ustawienia zaawansowane:



4.5 Po naciśnięciu przycisku *Move detector* w consoli można zobaczyć czy jest ruch na obrazku czy nie:



4.6 Po naciśnięciu przycisku *Capture photo* program robi zdjęcie  
 Po naciśnięciu przycisku *Capture Video* program nagrywa wideo AVI



4.7 Również program generuje stronę html z odświeżanym automatycznie obrazem z kamery.

**5. Wnioski**

Podczas laboratoriów nauczyliśmy się podstawowej obsługi kamery USB.  
Opanowaliśmy w podstawowym stopniu bibliotekę AForge.NET która w znaczącym stopniu ułatwia wykorzystywanie podstawowych funkcji kamery USB. Udało nam się zaimplementować algorytm wykrywający ruch przez porównywanie zdjęć piksel po pikselu. Jednak algorytm zawsze wykrywa ruch ponieważ kamerka za każdym razem pobierała inne kolory. Efekt ten uwidacznia się przy małej ilości światła oraz bardzo niskiej rozdzielczości.