 POLITECHNIKA WROCŁAWSKA

WYDZIAŁ INFORMATYKI I TELEKOMUNIKACJI

Urządzenia peryferyjne

**Laboratorium 2**

Obsługa kamery USB

Termin zajęć: Środa TP, 17:30

Autorzy:                               Prowadzący zajęcia:

Daria Jeżowska, 252731                         dr inż. Jarosław Mierzwa

**Cel ćwiczenia**

Celem ćwiczenia było stworzenie programu, który wykorzystał kamerę USB z następującymi założeniami:

1. Wylistuj urządzenia typu cap (kamery) i stwórz interfejs umożliwiający wybór po nazwie urządzenia (drivera) z którym chcesz się połączyć
2. Połącz się z wybranym urządzeniem i za pomocą odpowiednich komunikatów łączących się z driverami kamery - skonfiguruj ją.
3. Za pomocą programu powinno dać się zmieniać opcje kamery (rozdzielczość obrazu, nasycenie, kontrast, ew. zoom, sterowanie kamera etc.)
4. Zapisz obraz z kamery w dowolnym formacie (wskazany JPG)
5. Zapisz obraz z kamery w postaci filmu AVI
6. Rozbuduj program o:

* stwórz prosty detektor ruchu - poprzez analizę obrazu z kamery w czasie rzeczywistym (wystarczy sprawdzać zmiany koloru kilku punktów (pikseli), ćwiczenie można rozwinąć o najprostsze algorytmy wykrywające krawędzie etc.)

**Wstęp**

Kamera cyfrowa to urządzenie rejestrujące obraz i dźwięk oraz zapisujące sygnał audiowizualny w postaci cyfrowej. Najczęściej komunikuje się z komputerem za pomocą portu USB, a w laptopach jest to część wbudowana. W ostatnim czasie kamerki stały się podstawowym wyposażeniem każdego domu z komputerem i okazały się być niezbędne w naszej codzienności. Za ich pomocą prawie codziennie każdy się komunikuje z innymi ludźmi w pracy, na uczelni czy po prostu jako forma kontaktu z bliskimi. Historia kamerek internetowych sięga początku lat 90 – w 1993 roku pierwszy raz obraz z takiej kamery został umieszczony w internecie.

**Przebieg ćwiczenia**

Program został zrealizowany w języku *C#* z pomocą *Windows Forms* oraz biblioteki *AVICAP32.dll.* Ta biblioteka jest używana w celu przechwytywania obrazu wideo z rozszerzeniem *.avi* z kamerek internetowych oraz innych tego typu urządzeń. Jest to dosyć stara biblioteka i wiele nowych urządzeń nie jest przez nią obsługiwana. Abyśmy mogli jej użyć w efektywny sposób skorzystaliśmy z *wrappera* dla avicap32.dll. Poszczególne funkcjonalności uzyskuje się za pomocą funkcji SendMessage, która za argumenty przyjmuje *uchwyt* (HWND) z którego korzysta WIN32 API oraz odpowiednich wartości od których zależy co się może stać. Np. zmienna *WM\_CAP\_DRIVER\_CONNECT* przyjmująca wartość *0x40a* odpowiada za wychwycenie kamerki.

**Wykrycie kamery i wyświetlanie obrazu**

Funkcja odpowiedzialna za wykrycie kamery i wyświetlanie obrazu jest już w naszym wrapperze i wygląda następująco:

Obraz zawierający tekst

Opis wygenerowany automatycznie

Na początku tworzymy nasz *uchwyt* hHwnd, który stworzy nam nowe okienko w którym będziemy mogli wybrać urządzenie z którego chcemy skorzystać. *SendMessage* zwraca wartości 0 lub 1 i na podstawie tego możemy uzyskać różne funkcjonalności. W pierwszy ifie sprawdzamy czy istnieje urządzenie z którym możemy się połączyć, jeśli tak jest to wyświetlamy podgląd obrazu z kamerki. W przeciwnym wypadku niszczymy nasze powstałe wcześniej okienko. Dodatkowo w SetWindowPos możemy ustawić rozdzielczość obrazu.

**Robienie i zapisywanie zdjęć**

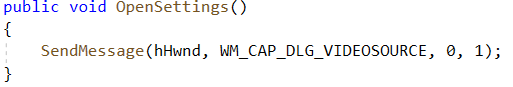
Funkcja odpowiedzialna za zapisanie obrazu wygląda następująco:

Obraz zawierający tekst

Opis wygenerowany automatycznie

Tworzymy obiekt *saveFileDialog*, co pozwala nam na otworzenie okna w którym wybieramy lokalizację dla pliku z rozszerzeniem *.jpg*, a następnie za pomocą SendMessage kopiujemy aktualny obraz z kamerki, a następnie przechowujemy go w zmiennej *data* i przerabiamy na bitmapę, a następnie zapisujemy.

**Otwieranie ustawień**



Za pomocą WM\_CAP\_DLG\_VIDEOSOURCE uruchamiamy okienko ustawień.

**Nagrywanie filmiku**

Obraz zawierający tekst

Opis wygenerowany automatycznie

W pliku *Form1.cs* są funkcje odpowiadające za m.in. za wywoływanie funkcji obiektu webCam, który został opisany wyżej przy pomocny kliknięcia odpowiedniego guzika.

Obraz zawierający tekst

Opis wygenerowany automatycznie