Wybrane narzędzia do projektowania i budowy interfejsów wizyjnych

1. Image Processing Toolbox i Image Acquisition Toolbox dla systemu MATLAB
2. Biblioteka OpenCV
3. Biblioteka LTI-lib i Impresario
4. TouchLess SDK

Image Processing Toolbox (IPT)

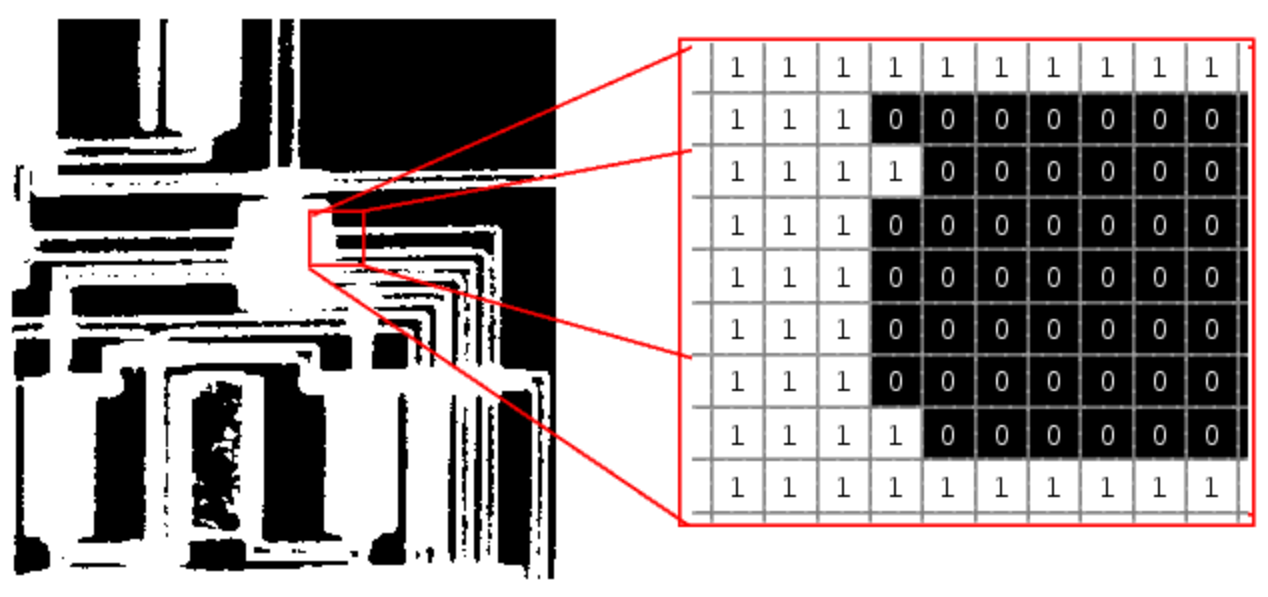
1. Reprezentacja obrazów w systemie MATLAB
2. Wprowadzanie i zapisywanie obrazów
3. Wyświetlanie obrazów
4. Operacje na obrazach

Reprezentacja obrazów w systemie MATLAB

Typy obrazów:

- Binarny – macierz logiczna, zawierająca jedynie wartości 0 i 1, interpretowane jako kolor czarny i biały, odpowiednio
- Indeksowy – macierz, której poszczególne elementy są indeksami do mapy kolorów (macierz `m`×3 klasy `double`, `m` – liczba kolorów w obrazie)
- Intensywnościowy – macierz, której elementy określają wartość intensywności danego piksela
- Truecolor (obraz RGB) – macierz `m`×`n`×3, której wartości określają nasycenie składowych R, G i B barwy

Obraz binarny



Obraz indeksowy

- Macierz obrazu może być typu: `logical`, `uint8`, `uint16`, `single` lub `double`
- Dla macierzy typu `single` i `double` wartości należą do zakresu [1, `p`], dla macierzy typu `logical`, `uint8` i `uint16` do zakresu [0, `p`-1]

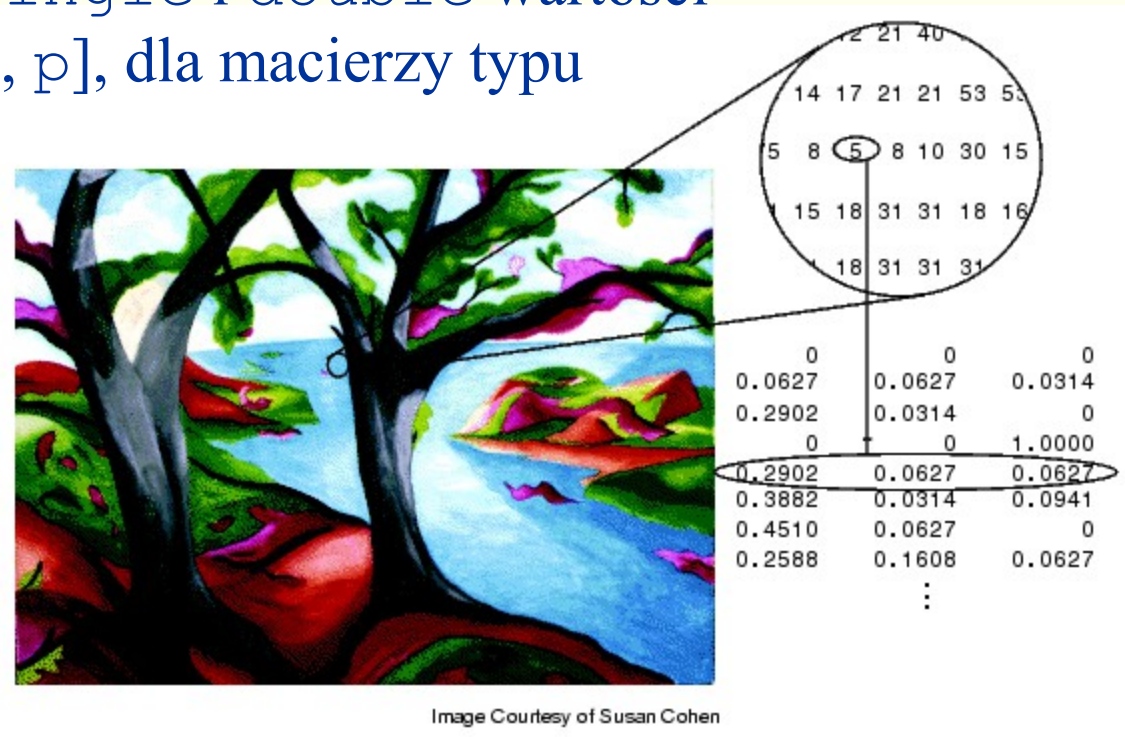
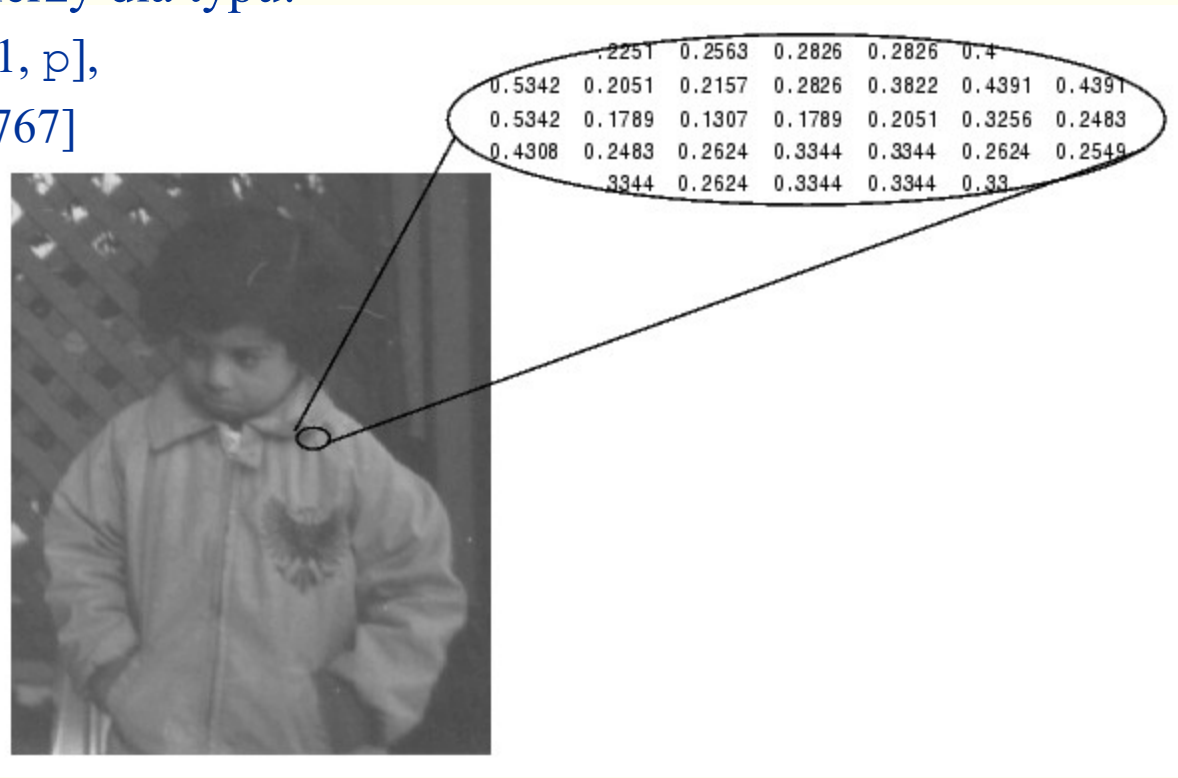


Image Courtesy of Susan Cohen

Obraz intensywnościowy

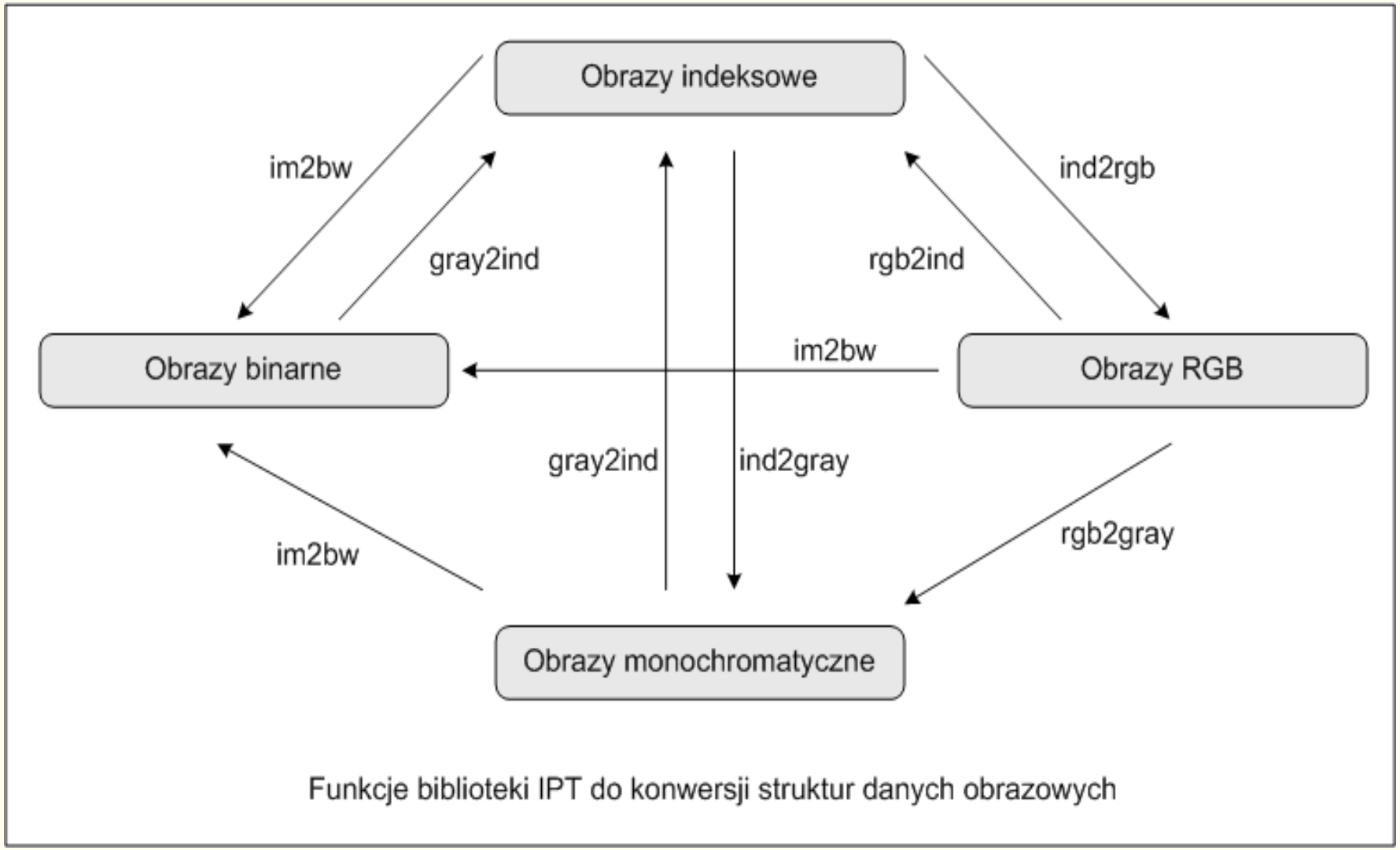
- Macierz obrazu może być typu: `uint8`, `uint16`, `int16`, `single` lub `double`
- Zakres wartości w macierzy dla typu:
  - `single` i `double` [1, p],
  - `int16`: [-32768, 32767]
  - `uint8`: [0, 255],
  - `uint16`: [0, 65535].



Obraz truecolor

- Odpowiadające sobie wartości w macierzach `R`, `G` i `B` określają kolor danego piksela
- Macierz obrazu może być typu: `uint8`, `uint16`, `single` lub `double`
- Dla macierzy typu `single` i `double` wartości należą do zakresu [0, 1], dla macierzy typu `uint8` i `uint16` do zakresu [0, 255]

Konwersja między typami obrazów



Funkcje biblioteki IPT do konwersji struktur danych obrazowych

Konwersja pomiędzy klasami obrazów

- Funkcje `im2uint8`, `im2uint16`, `im2int16`, `im2single`, `im2double` automatycznie dokonują przeskalowania danych każdego typu obrazu
- Funkcja `mat2gray` pozwala na utworzenie obrazu intensywnościowego z danych zawartych w macierzy poprzez przeskalowanie danych
- Konwersja obrazów indeksowych nie zawsze jest możliwa. Jeśli liczba kolorów obrazu konwertowanego jest większa niż zakres wartości macierzy docelowej, konieczne jest najpierw zredukowanie liczby kolorów za pomocą funkcji `imapprox`
- **Przykład:**

```
RGB2 = im2uint8(RGB1); %konwersja obrazu RGB
%typu double (zakres wartości 0..1)
%do typu uint8 (zakres wartości 0..255)
```

Wczytywanie i zapisywanie obrazów

- Funkcja `imread` służy do wczytywania obrazów z pliku
  - Obrazy zapisane w formacie 8 bitów na piksel są wczytywane do macierzy klasy `uint8`
  - Pliki obsługujące format 16 bitowy (PNG, TIFF) wczytywane są do macierzy klasy `uint16`
  - **Przykład:**

```
RGB = imread('football.jpg');
[X, map] = imread('trees.tif');
```

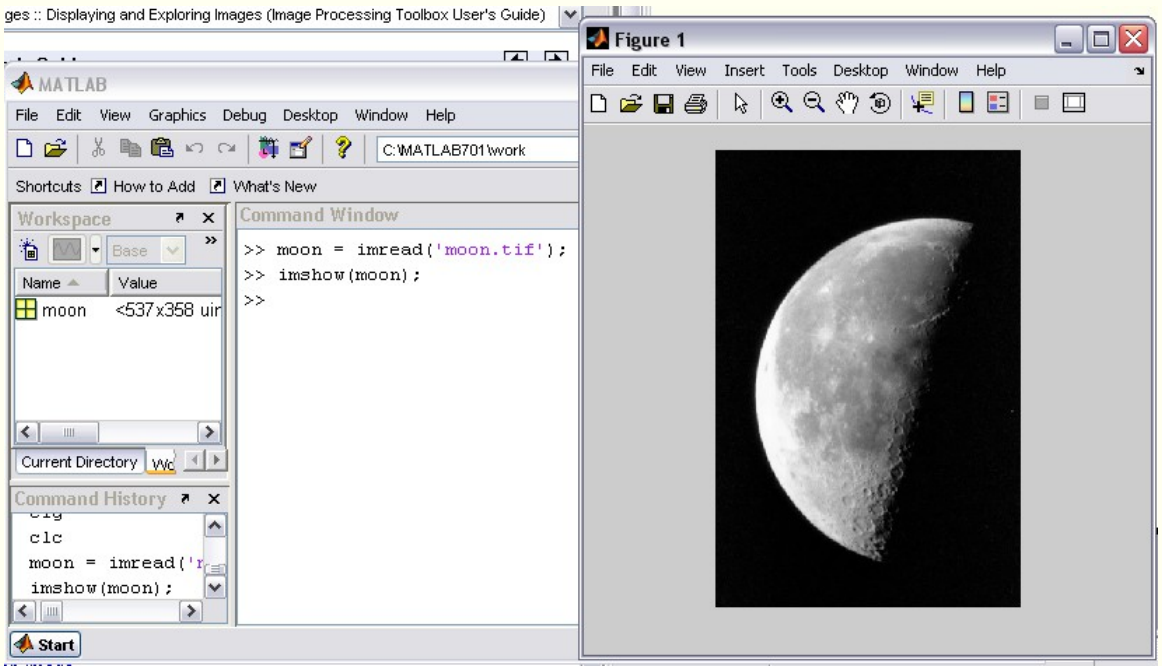
- Funkcja `imwrite` służy do zapisywania obrazu do pliku zgodnie z zadanym formatem
  - **Przykład:**

```
BW = imread('text.png');
imwrite(BW,'test.tif');
```

Wyświetlanie obrazów funkcja `imshow`

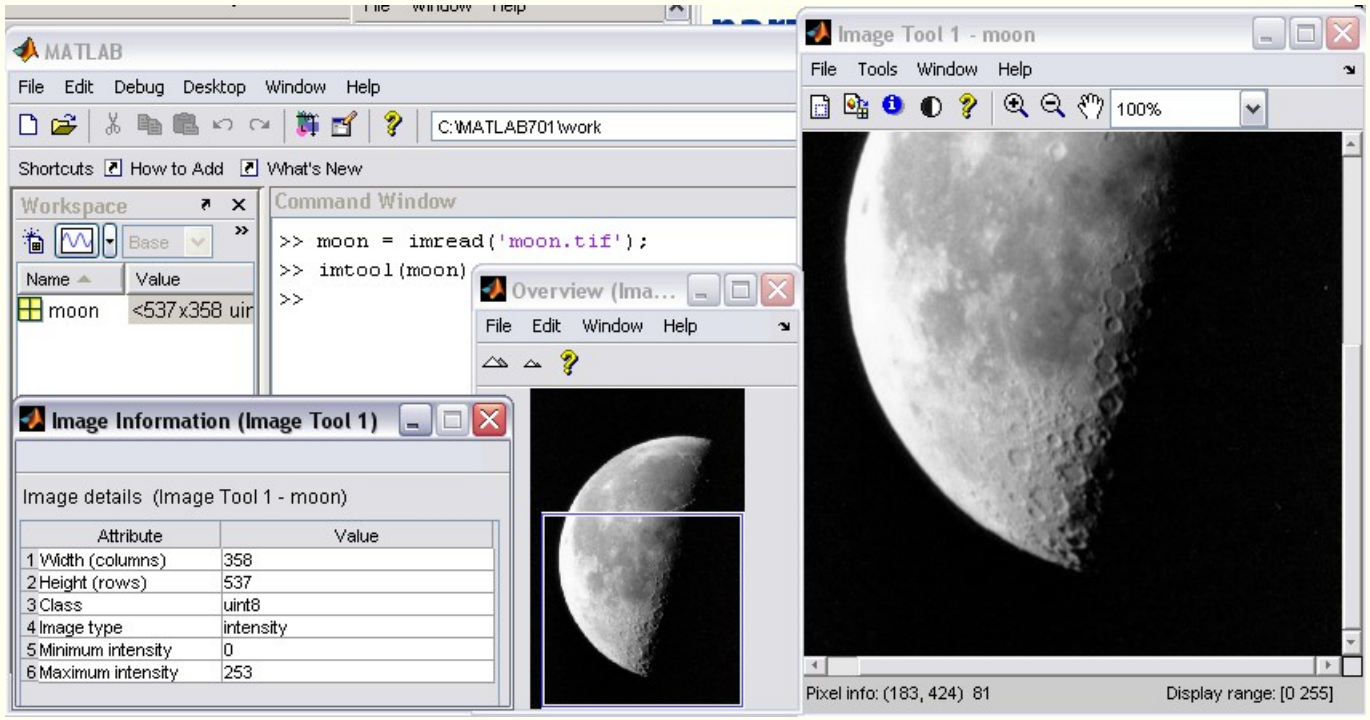
- Funkcja `imshow` – wyświetla obraz wskazany jako zmienna zawierająca dane obrazowe lub plik
  - **Przykład:**

```
moon = imread('moon.tif');
imshow(moon);
```



Wyświetlanie obrazów narzędzie `imtool`

- Narzędzie `imtool` – pozwala wyświetlić obraz oraz uzyskać informacje o poszczególnych pikselach i innych danych związanych z obrazem



Funkcje IPT

- Wczytywanie, zapisywanie i wyświetlanie obrazów – funkcje do importowania, eksportowania i wyświetlania obrazów oraz konwersji pomiędzy formatami obrazów
- Interaktywne narzędzia do pracy z obrazami
- Transformacje przestrzenne
- Analiza obrazów i statystyki
- Arytmetyka obrazów – funkcje służące do wykonywania operacji arytmetycznych na obrazach takich jak dodawanie, odejmowanie, mnożenie i dzielenie
- Poprawa jakości obrazów
- Filtracja liniowa i transformacje
- Operacje morfologiczne
- Funkcje do definiowania obszaru zainteresowania (ROI) i przetwarzania blokowego
- Funkcje do pracy z kolorem

Image Acquisition Toolbox (IAT)

1. Przechwytywanie obrazów z kamery
2. Podgląd strumienia wideo
3. Praca z przechwyconymi obrazami

Przygotowanie do akwizycji obrazów

Krok 1: Zainstalowanie i skonfigurowanie kamery

Kamera powinna być zainstalowana i skonfigurowana przed uruchomieniem systemu MATLAB

Krok 2: Utworzenie obiektu do akwizycji wideo

```
vid = videoinput('winvideo');
```

Krok 3: Podgląd strumienia video

```
preview(vid);
```

Pobranie obrazu

I. Przechwycenie pojedynczej klatki

```
obraz = getsnapshot(vid);
```

II. Przechwytywanie sekwencji klatek

```
figure; %utworzenie okna graficznego
set(gcf,'doublebuffer','on') %zapewnia płynne wyświetlanie
start(vid) %rozpoczęcie akwizycji
while (vid.FramesAcquired <= 100),
   obrazy = getdata(vid, 2);
   %wyznaczenie obrazu różnicowego dla 2 ostatnio pobranych
   %klatek
   diff_im = imabsdiff(data(:,:,:,1), data(:,:,:,2));
   imshow(diff_im); %wyświetlenie obrazu różnicowego
end;
stop(vid); %zatrzymanie akwizycji
```

Parametry przechwytywania

- Wyzwolenie akwizycji powoduje rozpoczęcie przechwytywania do bufora
- Funkcja `getdata` pobiera obraz z bufora do przestrzeni roboczej i zwalnia po nich miejsce w buforze
- Liczba dostępnych w buforze klatek pamiętana jest w polu `vid.FramesAvailable`
- Można ustalić liczbę klatek, jakie należy przechwycić wpisując daną wartość do pola `vid.FramesPerTrigger`
- W celu uniknięcia przepełnienia pamięci można zastosować funkcję `flushdata` usuwającą z pamięci wszystkie lub wskazane klatki

Biblioteka OpenCV

1. Charakterystyka
2. Struktura
3. Zawartość
4. Przydatne struktury
5. Praca z sekwencjami wideo

Charakterystyka biblioteki OpenCV

- OpenCV jest biblioteką funkcji implementujących metody wizji komputerowej
- Jest kompatybilna z Intel® Image Processing Library, która implementuje operacje niskiego poziomu na obrazach cyfrowych
- Implementuje algorytmy obejmujące kalibrację kamery, detekcję cech, śledzenie obiektów, analizę kształtu, analizę ruchu, rekonstrukcję 3D, segmentację obrazów i rozpoznawanie obiektów
- Jest wysoce funkcjonalna i bardzo wydajna (zoptymalizowana dla procesorów Intel® i technologii MMX™)

Jest wieloplatformowa, można z niej korzystać w systemach Windows, Linux, Android iOS, Mac OS (od wersji 2.1 także na procesorach 64-bitowych) w językach C, C++, Python i Java

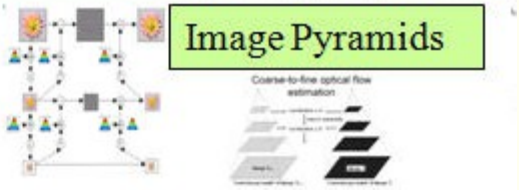
Jest darmowa zarówno do zastosowań komercyjnych jak i niekomercyjnych, dostępna pod adresem: http://opencv.org/

OpenCV Overview: > 500 functions

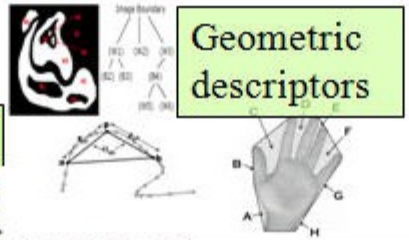
opencv.willowgarage.com

General Image Processing Functions

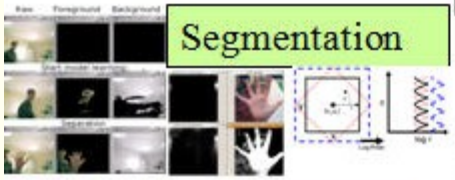
Image Pyramids



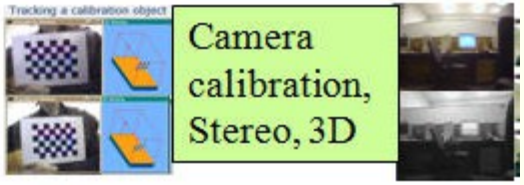
Geometric descriptors



Segmentation

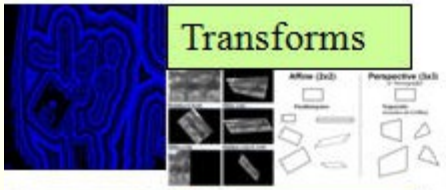


Camera calibration, Stereo, 3D

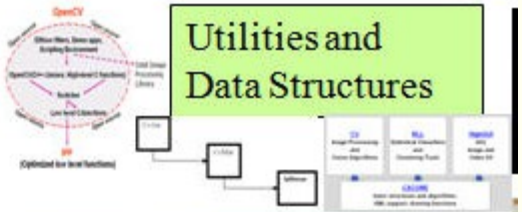


Features

Transforms



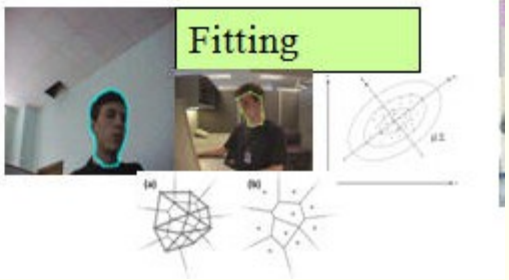
Utilities and Data Structures



Tracking

Machine Learning:
- Detection,
- Recognition

Fitting

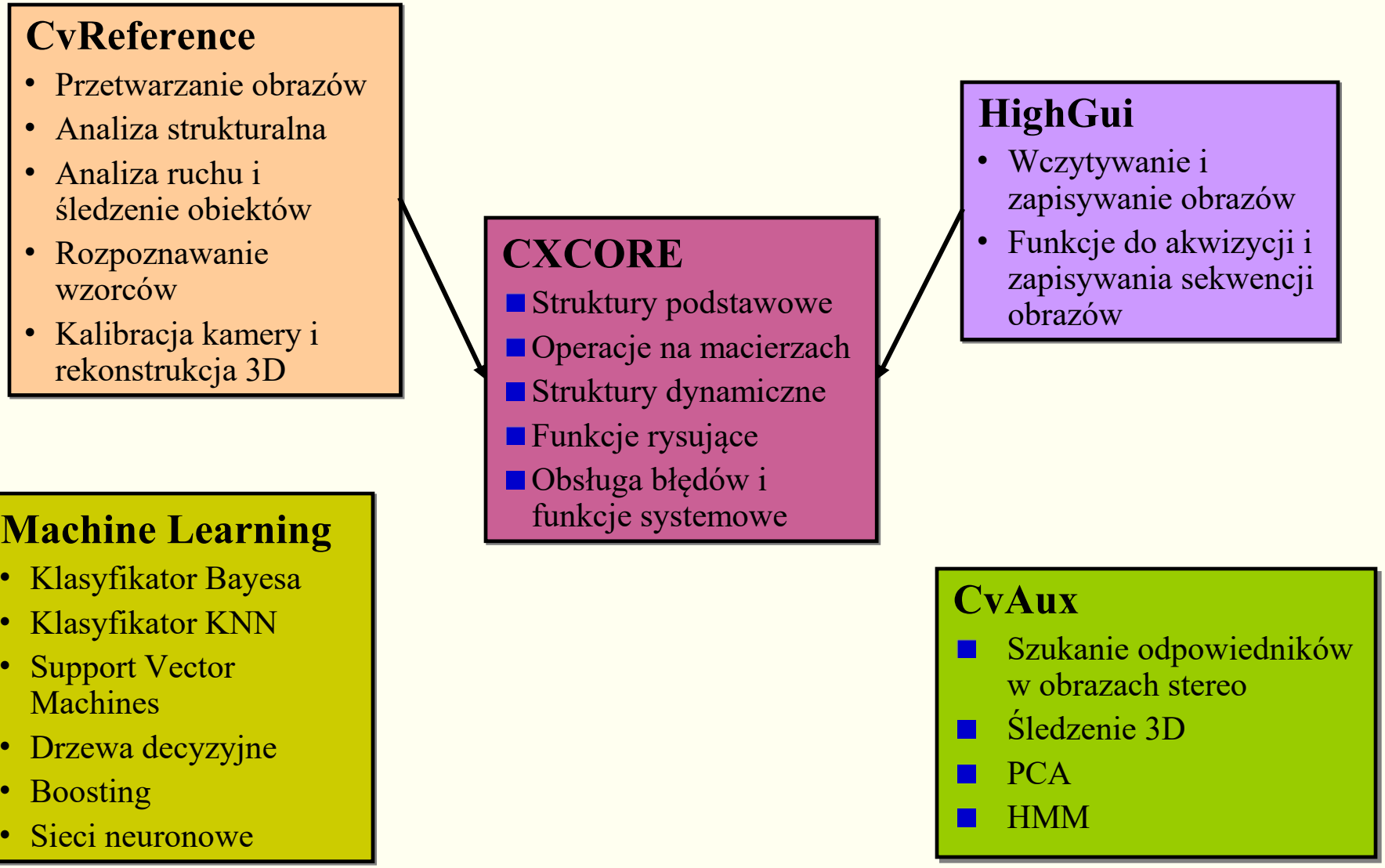


Matrix Math

Dokumentacja

- G. Bradski, *Learning OpenCV*, Editio Cantor Verlaq, 2008
- R. Laganière, *OpenCV 2 Computer Vision Application Programming Cookbook*, 2011
- Strona OpenCV: https://opencv.org/
  https://sourceforge.net/projects/opencvlibrary/
- Dokumentacja on-line: http://docs.opencv.org/

Struktura biblioteki w wersji 2.1



Struktura (v. 2.2)

video
- Analiza strumienia video (estymacja ruchu, śledzenie obiektów)
- Współpraca z sensorem Kinect

calib3d
- Przetwarzanie obrazów z wielu kamer
- Kalibracja kamer
- Estymacja położenia obiektów
- Poszukiwanie odpowiedników w obrazach stereo
- Rekonstrukcja 3D

features3d
- Detekcja punktów
- Tworzenie i porównywanie deskryptorów

objectdetect
- Detekcja obiektów i instancji predefiniowanych klas (twarzy, oczu, osób, samochodów)

imgproc
- Wstępna obróbka obrazów
- Filtracja obrazów
- Transformacje geometryczne
- Konwersje przestrzeni barw
- Histogramy

core
- Struktury podstawowe
- Operacje na macierzach
- Struktury dynamiczne
- Funkcje rysujące
- Obsługa błędów i funkcje systemowe

highgui
- Wczytywanie i zapisywanie obrazów
- Funkcje do akwizycji i zapisywania sekwencji obrazów

ml
- Klasyfikatory
- Regresja
- Grupowanie danych

gpu
- Algorytmy z różnych modułów przeznaczone do użycia na procesorze graficznym

Moduły pomocnicze
- Wrappery do języków Java, Python
- Wrapper do systemu MATLAB
- inne

Porównanie wersji

v. 2.0 (wrzesień 2009)

- Dodano interfejs obiektowy

v. 2.1 (kwiecień 2010)

- Dodano mechanizm obsługi błędów za pomocą wyjątków
- Możliwość zastosowania dla systemów 64-bitowych

v. 2.2 (grudzień 2010)

- Nowy podział na moduły
- Możliwość użycia dla systemu Android
- Dodano moduł opencv_gpu umożliwiający wykonywanie obliczeń na procesorze graficznym

v. 2.3 (lipiec 2011)

- Udoskonalono dokumentację
- Zastąpiono pakiet LAPACK (Linear Algebra PACKage) własną implementacją funkcji z zakresu algebry liniowej

Porównanie wersji

v. 2.4 (maj 2012)

- Moduł GPU uzupełniono o wsparcie dla pakietu CUDA 4.1 i CUDA 4.2
- Dodano API do przechowywania struktur danych OpenCV w łańcuchach znakowych i ich odczytywania
- Utworzono **opencv2.framework** dla iOS
- Wprowadzono udoskonalenia do modułu GPU

v. 3.0 (sierpień 2014)

- Przedefiniowane API
- Znacznie poprawiona wydajność na CPU
- Przyspieszenie działania na GPU

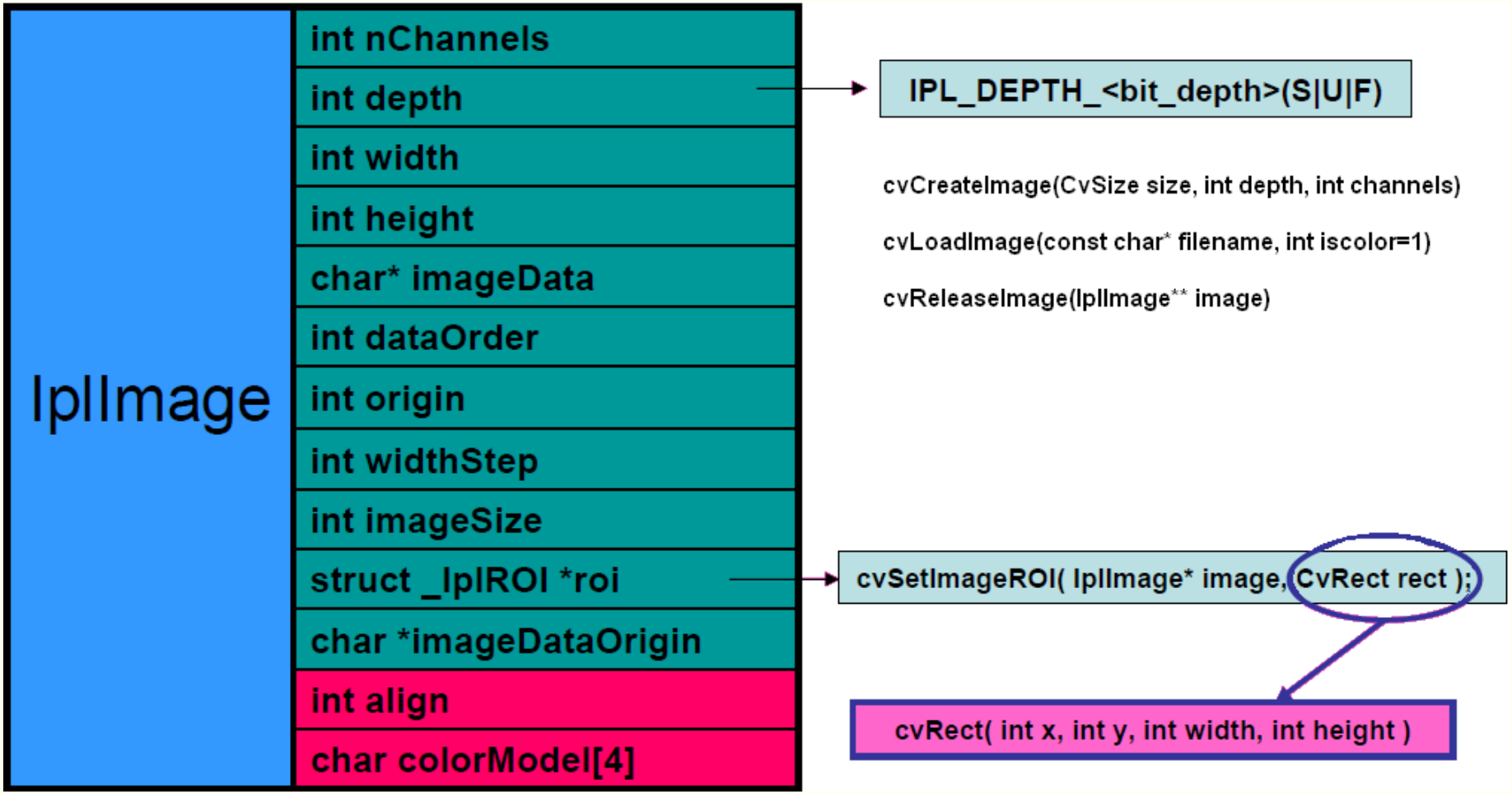
v. 3.3 (sierpień 2017)

- Poprawiony moduł uczenia hierarchicznego (Deep Learning - poszerzona wersja uczenia maszynowego oparta na uczeniu reprezentacji danych)
- Wiele optymalizacji

v. 4.0 (listopad 2018)

Wersje release: https://opencv.org/releases.html

Struktura do przechowywania obrazów - IplImage



Klasa Mat

```
class CV_EXPORTS Mat {
  public:
    // ... Wiele metod ...
    /*! Pola bitowe: (magic signature, continuity flag, depth, number of channels*/
    int flags;
    int dims; //! Wymiar macierzy, >= 2
    //! Liczba wierszy i kolumn lub (-1, -1) dla macierzy o wymiarze > 2
    int rows, cols;
    uchar* data; //! Wskaźnik na dane
    int* refcount; //! Wskaźnik do licznika referencji (= NULL dla macierzy wskazującej
                   //! na dane przydzielone przez użytkownika
    // inne metody i pola
    ...
};
```

Tworzenie macierzy - przykłady

Przykład 1: Utworzenie macierzy 100×60, 15 kanałów, elementy – 8-bitowe liczby bez znaku

```
Cv::Mat M;
M.create(100, 60, CV_8UC(15))
```

Przykład 2: Tworzenie macierzy

```
Int sz[] = {100, 100, 100};
Cv::Mat duzySzescian(3, sz, CV_8U, Scalar::all(0));
```

Przykład 3: Tworzenie macierzy dla danych pochodzących z bufora

```
void process_frame(const unsigned char* pixels, int width, int height, int step) {
    cv:Mat img(height, width, CV_8U, pixels, step);
    cv::GaussianBlur(img, img, cv::Size(7, 7), 1.5, 1.5);
}
```

Przykład 4: Tworzenie macierzy za pomocą specjalnych funkcji

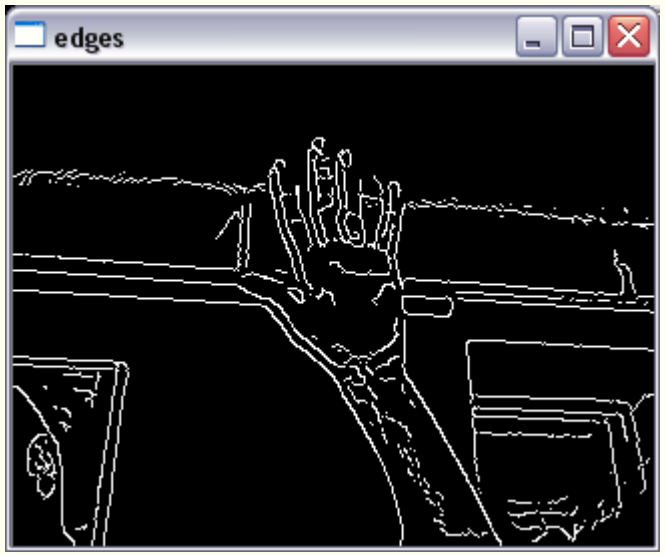
```
M += Mat::eye(M.rows, M.cols, CV_64F);
```

Dostęp do elementów macierzy

```
...
//rozdzielenie obrazu na składowe przestrzeni barw
vector<Mat> planes;
split(img_yuv, planes);
//Metoda 1. Przekształcenie składowej Y za pomocą iteratora
MatIterator_<uchar> it = planes[0].begin<uchar>(), it_end = planes[0].end<uchar>();
for (; it != it_end; ++it) {
    double v = *it*1.7 + rand()%21-10;
    *it = saturate_cast<uchar>(v*v/255.);
}
//Metoda 2. Przekształcenie I składowej chromatycznej z użyciem wskaźnika
//Metoda 3. Przekształcenie II składowej chromatycznej z użyciem operatora at
for (int y = 0; y < img_yuv.rows; y++) {
    uchar* Uptr = planes[1].ptr<uchar>(y);
    for (int x = 0; x < img_yuv.cols; x++) {
        Uptr[x] = saturate_cast<uchar>((Uptr[x]-128)/2 + 128);
        uchar& Vxy = planes[2].at<uchar>(y, x);
        Vxy = saturate_cast<uchar>((Vxy-128)/2 + 128);
    }
}
merge(planes, img_yuv);
...
```

Przykładowy program

```
#include „highgui.h”
#include „cv.h”
using namespace cv;
int main()
{
    VideoCapture cap(0);
    if (!cap.isOpened()) return –1;
    Mat edges;
    namedWindow(„edges”, 1);
    for (;;) {
        Mat frame;
        cap >> frame;
        cvtColor(frame, edges, CV_BGR2GRAY);
        GaussianBlur(edges, edges, Size(7, 7), 1.5, 1.5);
        Canny(edges, edges, 0, 30, 3);
        imshow(„edges”, edges);
        if (waitKey(30) >= 0) break;
    }
    return 0;
}
```



Przykładowy program

Wydzielanie krawędzi z użyciem operatora Sobela

void cvSobel(const CvArr* src, CvArr* dst, int xorder, int yorder, int aperture_size=3)

```
#include ″cv.h″
#include ″highgui.h″
int main(int argc, char** argv)
{
    char *nazwa_pliku = ″clown.bmp″;
  IplImage *obraz = cvLoadImage(nazwa_pliku, 0);
  cvNamedWindow(″Obraz oryginalny″);
  cvShowImage(″Obraz oryginalny″, obraz);
  IplImage *obraz_kraw = cvCreateImage(cvGetSize(obraz), IPL_DEPTH_16S, 1);
  cvSobel(obraz, obraz_kraw, 0, 1);
  cvNamedWindow(″Krawedzie″);
  cvShowImage(″Krawedzie″, obraz_kraw);
  cvWaitKey(0);
  cvReleaseImage(&obraz);
  cvReleaseImage(&obraz_kraw);
  cvDestroyWindow(″Obraz oryginalny″);
  cvDestroyWindow ″Krawedzie″);
}
```

Zawartość biblioteki

- Wyznaczanie cech
  - Detekcja krawędzi (operator Sobela, laplasjan, operator Canny'ego) i punktów narożnych
  - Detekcja linii (transformacja Hough'a)
- Statystyki
  - Wartość średnia, minimalna i maksymalna, odchylenie standardowe,
  - Momenty (centralne, znormalizowane, Hu)
- Morfologia
  - Erozja, dylatacja, opening, closing, gradient, top-hat
- Funkcja odległości
- Progowanie
- Wypełnianie obiektów
- Histogramy wielowymiarowe
- Analiza strukturalna (aproksymacja konturów)
- Rozpoznawanie obiektów (PCA, HMM)
- Rekonstrukcja 3D

Dostęp do pikseli

Niech:

- img – obraz,
- w, k – numer wiesza i kolumny dla piksela, którego wartość w kanale n-tym chcemy odczytać/zapisać
- lw, lk – liczba wierszy i kolumn obrazu img

Sposoby dostępu:

```
CvScalar s;
s = cvGet2D(img, w, k);
int value = s.val[n];
s.val[n] = 123;
cvSet2D(img, w, k, s);
```

```
int height = img->height;
int width = img->width;
int step = img->widthStep/sizeof(float);
int channels = img->nChannels;
TYPE *data = (TYPE *)img->imageData;
data[w*step+k*channels+n];
data =123;
```

```
value=((TYPE *)(img->imageData+w*img->widthStep))[k*img->nChannels+n];
((TYPE *)(img->imageData+w*img->widthStep))[k*img->nChannels+n]=123;
```

Inne przydatne struktury

- **CvPoint**, **CvPoint2D32f**, **CvPoint3D32f** – struktury do przechowywania informacji o punktach
- **CvSize**, **CvSize2D32f** – struktury do przechowywania wymiarów prostokąta
- **CvRect** – prostokąt z offsetem

  CvRect r = cvrect(int x, int y, int width, int height);
- **CvMat** – struktura do przechowywania macierzy, obsługiwana podobnie jak IplImage

  CvMat * cvCreateMat(int rows, int cols, int type);

  cvReleaseMat(CvMat** mat);
- **CvMatND** – wielowymiarowa wersja CvMat
- **CvScalar** – kontener dla 1, 2, 3 lub 4 wartości double

  CvScalar s = cvScalar(double val0, double val1, double val2, double val3);
- **CvSparseMat** – n-wymiarowa macierz rzadka

Struktury dynamiczne

- **CvSeq** – wektor dynamiczny (listy, kolejki, stosy)

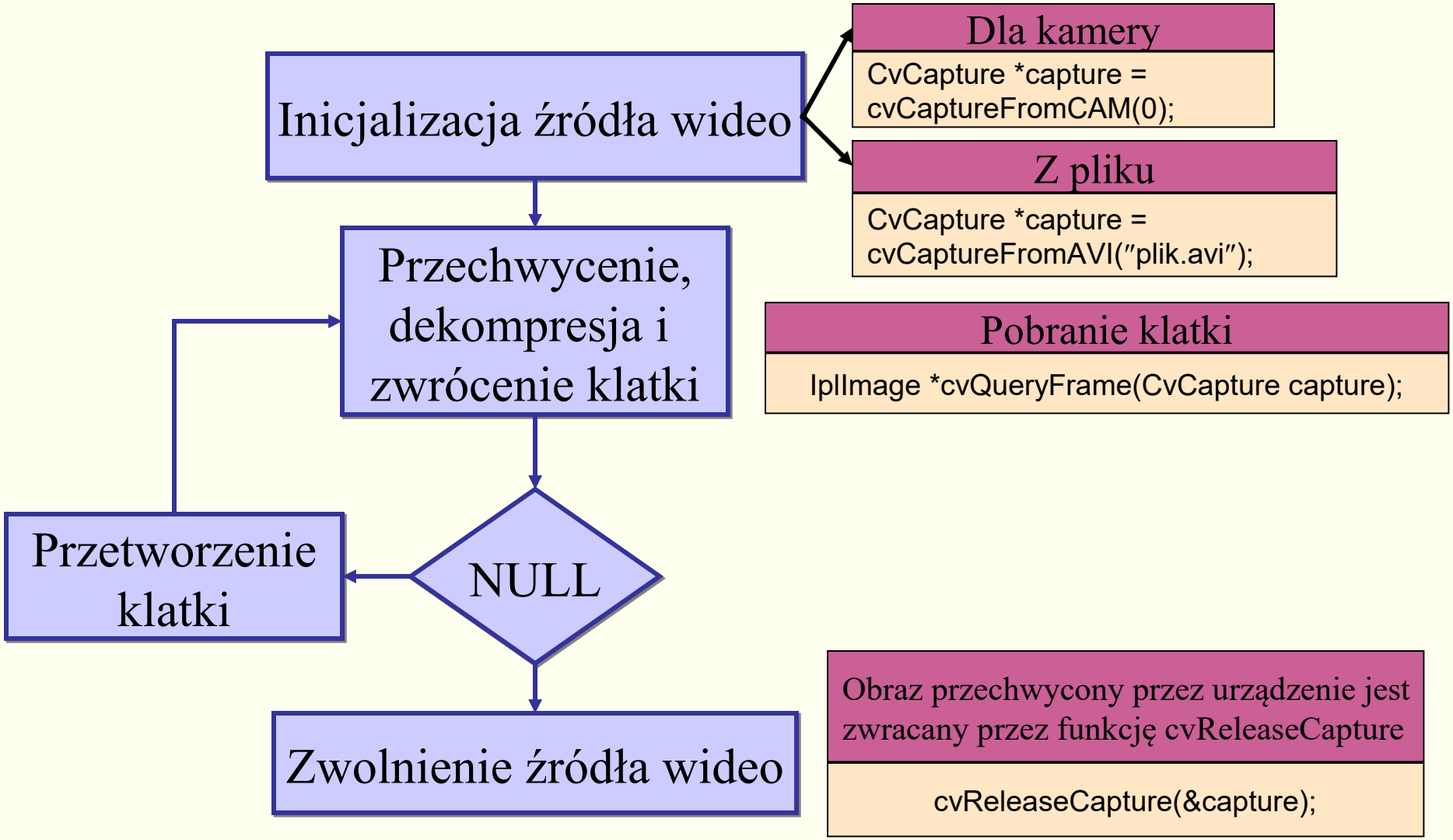
  cvCreateSeq(int seq_flags, int header_size, int elem_size,
  CvMemStorage storage);

- **CvMemStorage** – dynamiczny magazyn pamięci (sekwencje, kontury, grafy)

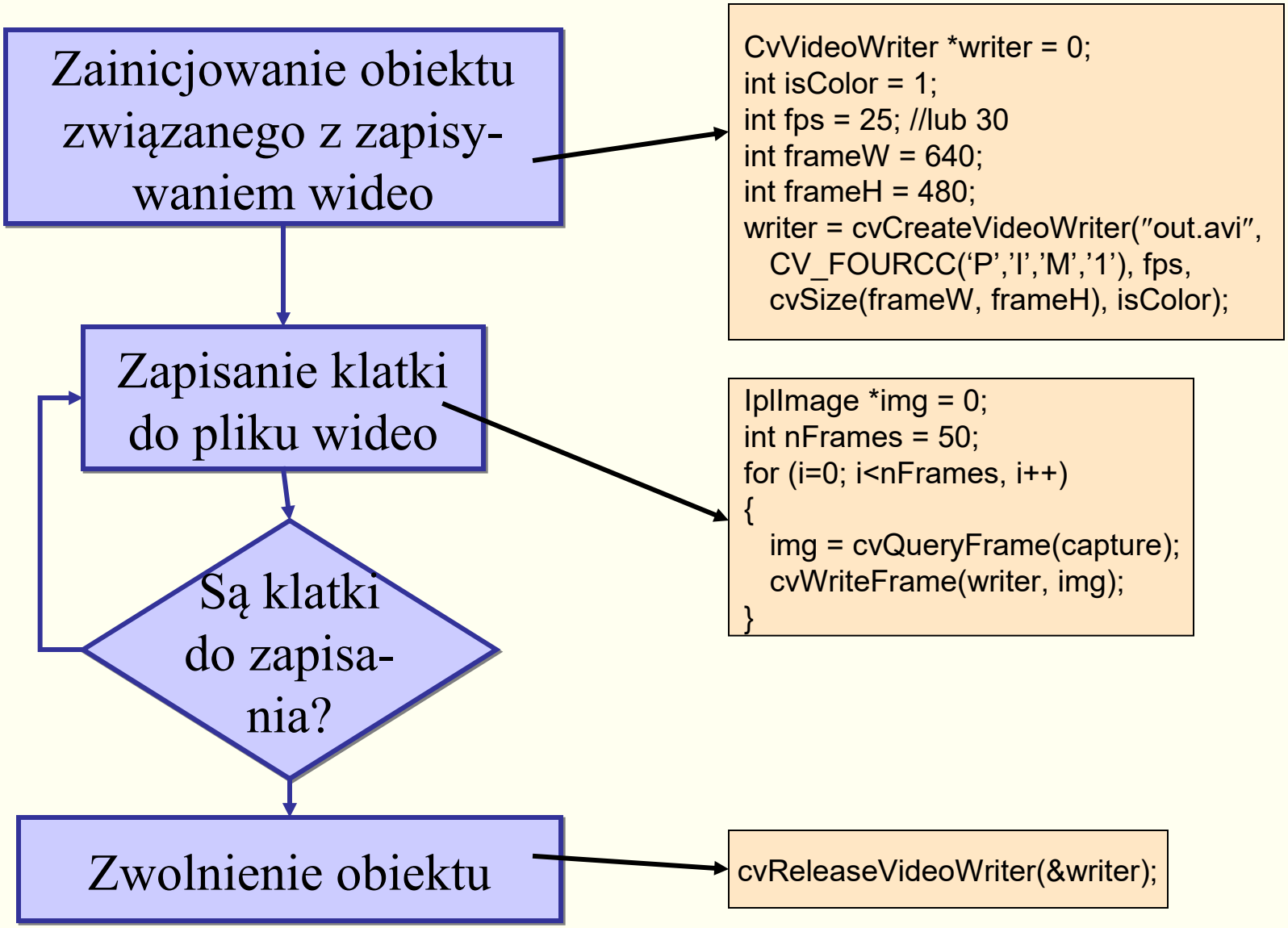
  cvCreateMemStorage(int block_size = 0);

  cvClearMemStorage(CvMemStorage *storage)

Praca z sekwencjami wideo



Zapisywanie sekwencji wideo



Dostępne kodeki dla wideo

Kodek	Wartość parametru dla funkcji cvCrateVideoWriter
MPEG-1	CV_FOURCC('P','I','M','1')
motion-jpeg	CV_FOURCC('M','J','P','G')
MPEG-4.2	CV_FOURCC('M','P','4','2')
MPEG-4.3	CV_FOURCC('D','I','V','3')
MPEG-4	CV_FOURCC('D','I','V','X')
H263	CV_FOURCC('I','2','6','3')
FLV1	CV_FOURCC('F','L','V','1')
Kod -1 spowoduje otwarcie okna wyboru kodeka (w Windows)	

Analiza ruchu i śledzenie obiektów

- Odejmowanie tła
- Mean-shift i Cam-shift
- Wzorce ruchu
- Przepływ optyczny
- Aktywne kontury
- Estymatory

Biblioteka Point Cloud Library (PCL)

Charakterystyka

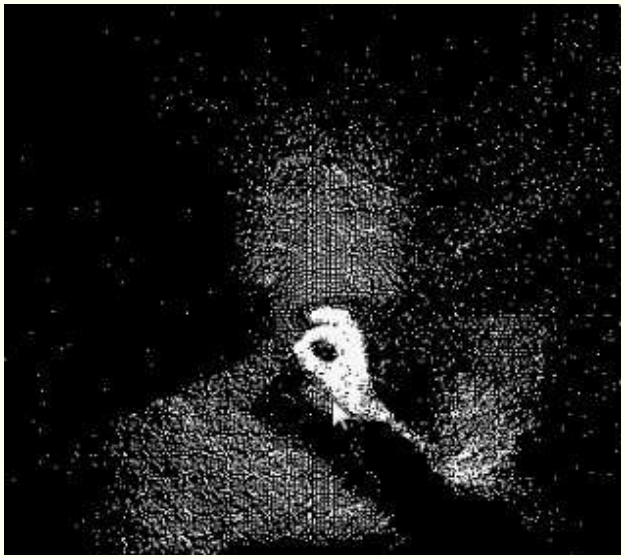
(http://pointclouds.org/)

- Zawiera klasy do przetwarzania chmur punktów
- Tworzona w ramach projektu otwartego realizowanego przez społeczność programistów
- Zawiera implementacje algorytmów do filtracji, wyznaczania cech, rekonstrukcji powierzchni, dopasowywania modelu, segmentacji i inne
- Dostępna na licencji BSD
- Projekt jest wspierany finansowo przez Willow Garage, Nvidia i Google.

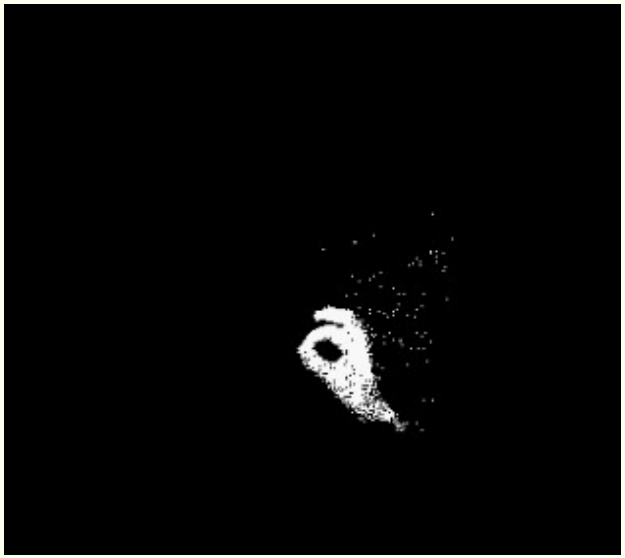
Ważniejsze klasy biblioteki PCL

- **PassThroughFilter** – filtracja chmury punktów
- **StatisticalOutlierRemoval** – usuwanie punktów izolowanych
- **VoxelGrid** – redukcja liczby punktów chmury
- **NormalEstimation** – wyznaczanie wektorów normalnych do powierzchni rozpiętej na chmurze
- **PFHEstimation**, **FPFHEstimation** – budowa deskryptorów lokalnych PFH i FPFH
- **VFHEstimation** – budowa deskryptora globalnego VFH

Filtracja chmury punktów
PassThroughFilter

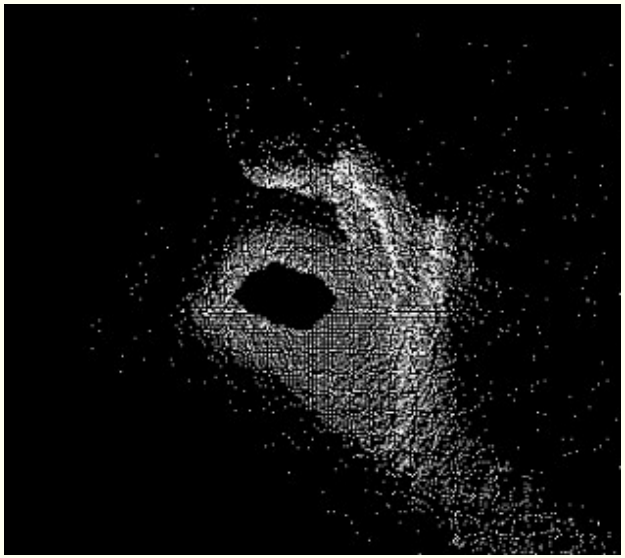


przed

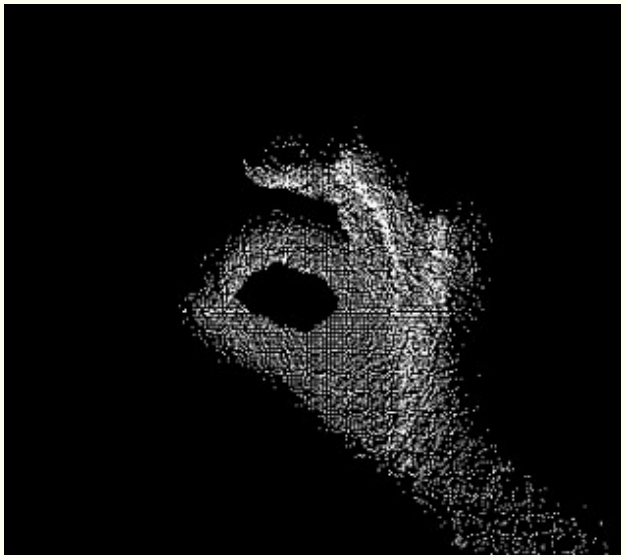


po

Usuwanie punktów izolowanych
StatisticalOutlierRemoval

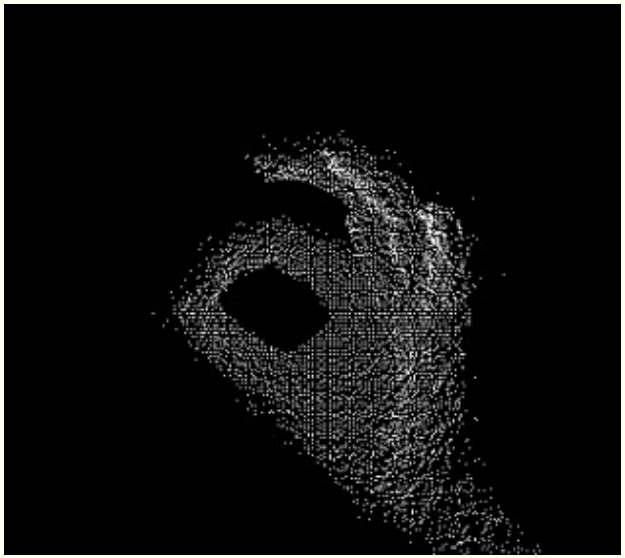


przed



po

Redukcja liczby punktów
VoxelGrid

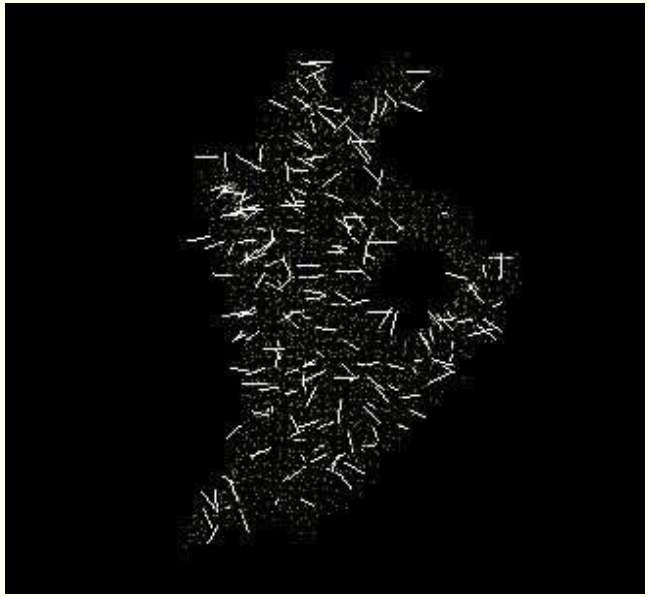


przed

po

Wyznaczanie wektorów normalnych do powierzchni rozpiętej na chmurze

NormalEstimation



Biblioteka LTI-Lib

1. Charakterystyka
2. Nakładka Impresario

Charakterystyka LTI-Lib

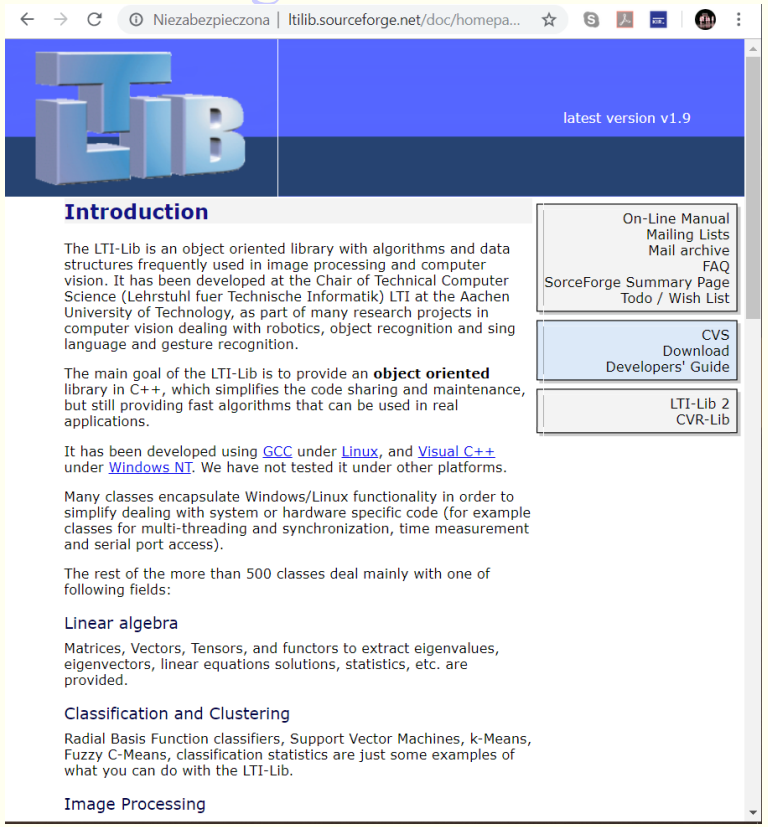
1. Dostępna na licencji LGPL
2. Jest biblioteką obiektową
3. Dostępna dla różnych platform (Linux, Windows)
4. Wykorzystuje język skryptowy *perl*, GTK (Gimp Tool Kit) do zadań związanych z wizualizacją, bibliotekę zlib (do kompresji danych) w niektórych funkcjach we/wy
5. Posiada interfejs do niektórych funkcji pakietu LAPACK (Linear Algebra PACKage)

Zawartość

- Algebra liniowa
  - Obliczanie wartości własnych i wektorów własnych
  - Rozwiązywanie równań liniowych
  - Statystyki, itp.
- Klasyfikacja i grupowanie
  - Klasyfikatory Radial Basis Function, Support Vector Machines, k-średnich
  - Statystyki klasyfikacyjne
- Przetwarzanie obrazów
  - Filtracja liniowa, segmentacja, falki
- Narzędzia do wizualizacji

Źródła dla LTI-Lib

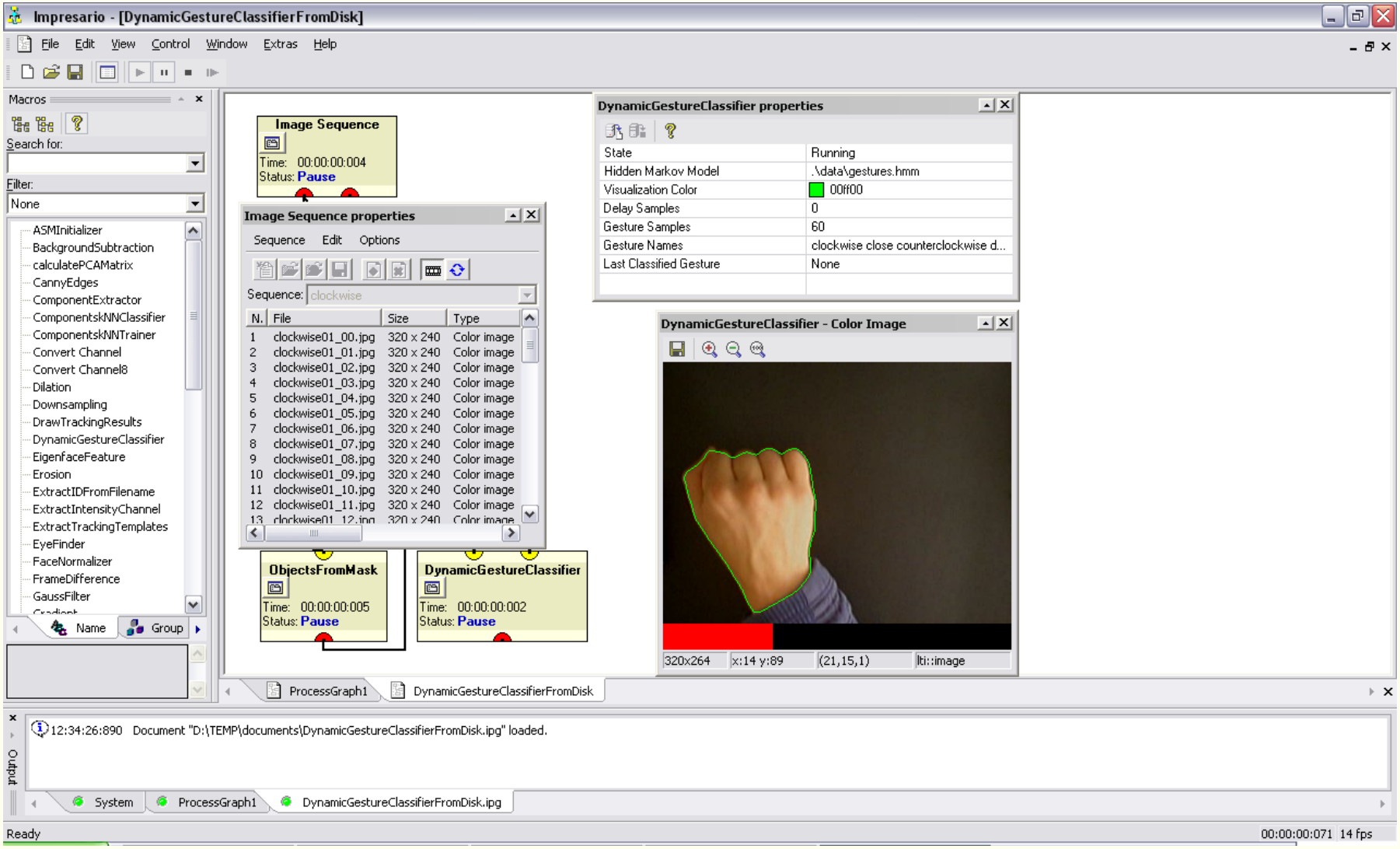
- Strona projektu: http://ltilib.sourceforge.net
- Dokumentacja on-line: http://ltilib.sourceforge.net/doc/html/index.shtml
- Wiki: http://ltilib.pdoerfler.com/wiki



Impresario

A Graphical User Interface for Rapid Prototyping of Image Processing System

shttp://www.techinfo.rwth-aachen.de/Software/Impresario



TouchLess: webcam multi-touch SDK

- Pozwala tworzyć aplikacje typu multi-touch z użyciem kamery
- Dostępny na stronie http://www.codeplex.com/touchless
- Darmowy
- Open-source