ĆWICZENIE 1/2 WIDZENIE MASZYNOWE

Wstęp teoretyczny

Karta akwizycji obrazu Frame-Grabber typu Matrix Vision.

Poniżej pod są podstawowe informacje dotyczące sposobu działania karty VFG (Video Frame Grabber). Wszystkie funkcje i typy do obsługi VFG zawarte są w bibliotece:
mvIMPACT_NET.dll

Tablica LUT (Look-Up Table)
Programowalna tablica o wymiarze 3 x 256 bajtów.
Z reguły dostępna "software'owo", często ten typ jest predefiniowany w bibliotece VFG.

```
Pascal      LUT : array[1..3,0..255] of byte ;
C           byte  LUT[3][256] ;
            char  LUT[3][256] ;
```

Transformuje obraz według zadanej charakterystyki.
Można podstawić dowolne wartości typu byte pod elementy macierzy.

Jeśli LUT[1,n]<>LUT[2,n] itd. to odcienie szarości można iterpretować jako różne kolory.

Modyfikuje obraz "idący" na ekran (LUT Wyjściowy), lub zawartość pamięci (LUT wejśc.).

```
Obraz = R + G + B
Z pamięci pobrana wartość n typu byte.
Obraz = LUT[1,n]+LUT[2,n]+LUT[3,n]
```

Paletę kolorów zmienia się za pomocą funkcji

```
void presetPalette(Paletce* source, PalettePretype palettepretype)
```

należącej do klasy gBase. Parametr palettepretype jest predefiniowanym typem palety. (np. paleta szaroodcieniowa liniowa).

Przykładowy kod

```
Palette palette = new Palette(256, 3);
gBase.presetPalette(palette, PalettePresetType.pptGrayLinear);
```

Przykładowe fragmenty kodu źródłowego

Inicjalizacja karty VFG.

```
DeviceManager devMgr = new DeviceManager();
```

```
Device dev = devMgr.getDevice(0);

try
  {
   dev.open();
  }
catch( ImpactAcquireException e )
  {
   // this e.g. might happen if the same device is already opened in another process...
   Console.WriteLine("An error occurred while opening the device " + pDev.serial.read() +
   "(error code: " + e.Message + "). Press any key to end the application..." );
   Console.ReadLine();
   Environment.Exit(0);
  }
```

Ustawienia modu pracy

```
Settings settings = new Settings(ref dev);
settings.imageDestination.pixelFormat.write(TImageDestinationPixelFormat.idpfMono8);
```

Zakończenie pracy.

```
dev.close();
```

Stworzenie nowego okna zawierającego obiekt typu Image

```
Image image = new Image();
Window Win = new Window(image);
```

Zapis do bitmapy

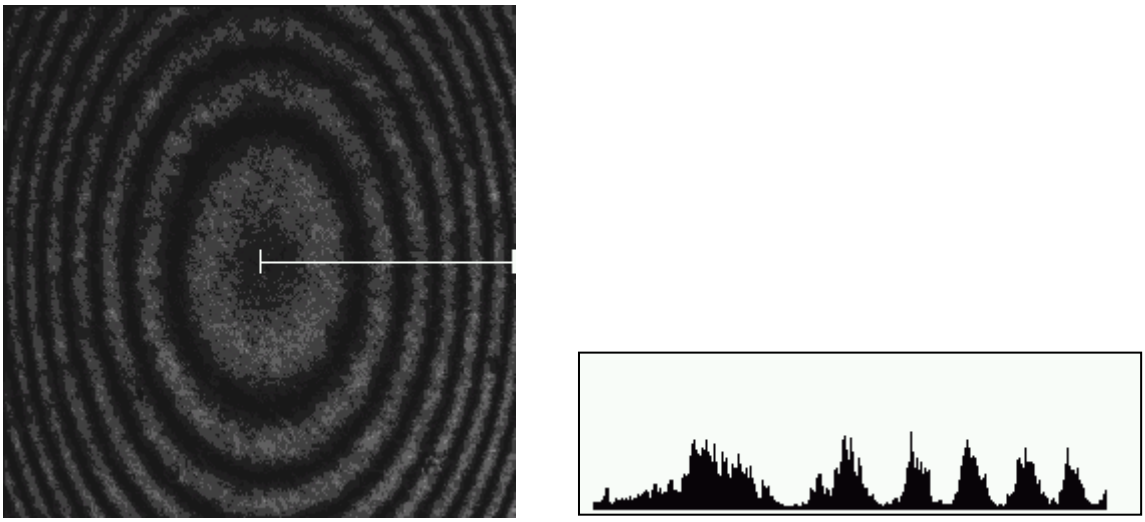
```
Palette palette = new Palette(256, 3);
Image image=new Image();

gBase.saveImage(image, "plik.bmp", FileFormat.ffBmp, ref palette);  //zapis BMP z paletą palette.
```

Przykładowy dostęp do danych obrazu

```
image.prepareReadWriteAccess();  //przygotowanie obiektu Image do dostępu read+write
int value;
for (int x = 1; x < image.width; x++)
   for (int y = 1; y < image.height; y++)
{
value = image.getPixel(x,y);   //odczyt jasności piksela 0-255
image.setPixel(x,y,value+10);  //zapis jasności piksela 0-255
}
image.releaseAccess();  //zwolnienie obiektu Image
```

Densytogram

Densytogram jest rozkładem natężenia wzdłuż zadanego przekroju.



Densytogram jest obiektem typu Profile i jest obliczany za pomocą funkcji

void calcLineProfile(Profile* dest, Image* source, Line* line)

należącej do klasy gBase.

Aby w oknie pojawiła się linia, wdłuż której liczony jest densytogram, należy użyć obiektu typu LineOfInterest, który tworzony jest za pomocą konstruktora

LineOfInterest (Window *window, Line *line)

Przykładowy kod:

```
Image image=new Image();
Window Win=new Window(image); //okno z obrazem z którego będzie liczony densytogram
Window WinProf = new Window(profile);  //okno, w którym wyświetli się densytogram

//pamiętamy, że rozdzielczość obrazu to 576 linii, 768 kolumn.
LineOfInterest  loi = new LineOfInterest(Win, new Line(new Point(50, 576 / 2), new Point(768 - 50, 576/2)));
profile = gBase.calcLineProfile(image, new Line(loi.line.p0, loi.line.p1));
profWin.buffer = profile;
```

Histogram

Z punktu widzenia statystyki jest to rozkład gęstości prawdopodobieństwa występowania pixeli o określonej jasności.

Histogram jest obiektem typu Histogram i jest obliczany za pomocą funkcji

void calcHistogram(Histogram* dest, Image* source)

należącej do klasy gBase.

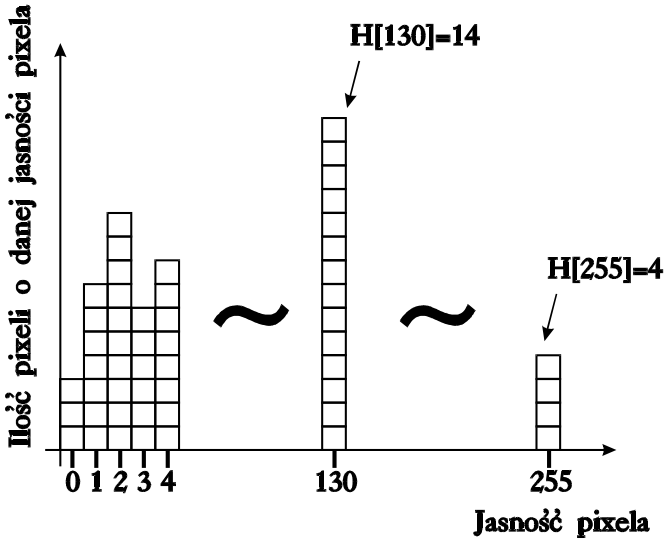
Aby obliczyć histogram z zaznaczonego obszaru, należy użyć obiektu typu AreaOfInterest, który tworzony jest za pomocą konstruktora:

AreaOfInterest (Window *window, Rectangle *rect)

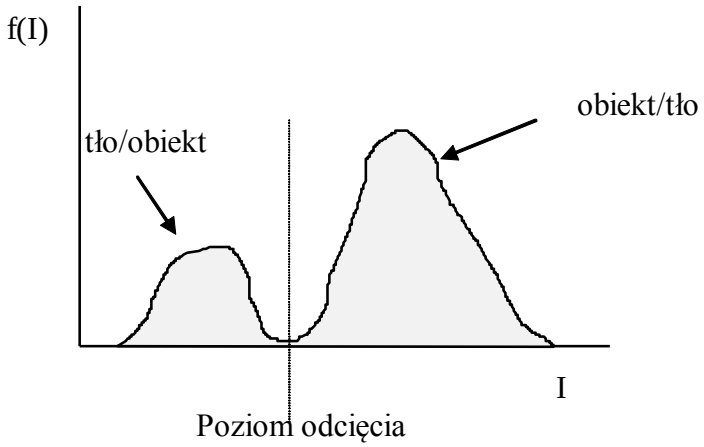
Przykładowy kod:

```
Histogram hist;
```

```
Image image=new Image();
Window Win=new Window(image); //okno z obrazem z którego będzie liczony histogram
Window WinHist = New Window(hist); //okno histogramu
aoi = new AreaOfInterest(Win, new Rectangle(1,1,image.width-2,image.height-2));
hist = gBase.calcHistogram(image);
WinHist.buffer = hist;
```



Przy ustalaniu poziomu odcięcia na podstawie histogramu przyjmuje się go w minimum rozkładu:



Podstawowe problemy przy wyborze poziomu odcięcia na podstawie histogramu:

- szeroka “dolina” funkcji utrdnia znalezienie optimum
- w histogramie występuje wiele min /max ze względu na charakter obiektów
- wysokie szumy obrazu uniemożliwiają znalezienie min
- zmienne tło t(x,y) modyfikuje histogram

W celu wyeliminowania tych problemów stosuje się technikę scattergramu (patrz dalej).

Filtracja obrazu metodą splotu:

Splot funkcji definiujemy jako:

$$f(x)*g(x)=\int_{-\infty}^{+\infty} f(u)g(x-u)du$$

co w wypadku filtracji numerycznej polega na zastosowaniu funkcji odpowiedzi impulsowej PSF[g(x)][1] do wszytstkich punktów funkcji f(x) i akumulację wyniku w każdym punkcie. W zastosowniach cyfrowych stosuje się zamianę całki dwuwymiarowej na dyskretną sumację.

$$F(x,y)=f(x,y)*g(x,y)=\sum_i\sum_j f(i,j)g(x-i,y-j)$$

gdzie ***g*** jest maską splotu przestrzennego. Gdy splot jet wykonywany na całym obrazie staramy się ograniczyć wymiar maski w celu minimalizacji czasu obliczeń. Typowo stosuje się maski 3x3, ze względów wydajnościowych maksymalnie 15x15.

Dla wygody stosuje się wstępnie przetworzoną maskę

$$h(x,y)=g(-x,-y)$$

wtedy splot przyjmuje postać:

$$F(x,y)=\sum_i\sum_j f(x+i,y+j)h(i,j)$$

Jest to de facto uwzględnienie z odpowiednią wagą punktów wpływu punktów otoczenia na punkt podlegający filtracji. Aby przez proces filtracji nie zmieniać średniej intensywności w obrazie wynik filtracji mnoży się przez czynnik normujący. W zależności od celu procesu filtracji stosuje się różne typu filtrów. Kilka przykładowych podajemy poniżej:

$$\frac{1}{10}\begin{bmatrix}1&1&1\\1&2&1\\1&1&1\end{bmatrix}$$

Figure A filtr gaussowski wygładzający (dolnoprzepustowy)

$$\frac{1}{16}\begin{bmatrix}1&2&1\\2&4&2\\1&2&1\end{bmatrix}$$

Figure B filtr gaussowski dokładniej oddający ksształt krzywej gaussa

$$\begin{bmatrix}-1&-1&-1\\-1&9&-1\\-1&-1&-1\end{bmatrix}$$

Figure C Filtr górnoprzepustowy

$$\begin{bmatrix}-1&-1&-1\\-1&8&-1\\-1&-1&-1\end{bmatrix}$$

Figure D Filtr laplasowski

Filtr erozyjny:

[1] ang. *Point Spread Function* PSF

Punkt środkowy zastępowany jest przez maksymalną wartość spośród punktów sąsiednich.

Filtr antyerozyjny:
Punkt środkowy zastępowany jest przez minimalną wartość spośród punktów sąsiednich.

Filtry wykrywające krawędzie:

0°	**45°**
$\begin{bmatrix} -1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & 1 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & -1 & 1 \end{bmatrix}$
Figure E Prewitta 0 stopni	*Figure F Prewitta 45 stopni*
$\begin{bmatrix} -3 & -3 & 5 \\ -3 & 0 & 5 \\ -3 & 3 & 5 \end{bmatrix}$	$\begin{bmatrix} -3 & 5 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & -3 \end{bmatrix}$
Figure G Kirsha 0 stopni	*Figure H Kirsha 45 stopni*
$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{bmatrix}$
Figure I Robinsona 0 stopni	*Figure J Robinsona 45 stopni*

W celu przyspieszenia obliczeń czasami stosuje się zamianę filtrów 2-D na dwa filtry 1-D:

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} = \frac{1}{4}\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}\frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$

Zamiana filtru 2-D na dwa filtry 1-D

Filtracji obrazu łatwo dokonać za pomocą funkcji

void convolve(Image* dest, Image* source, StockKernel kernel)

należącej do klasy gBase. Parametr kernel jest predefiniowaną maską splotu.

Przykład (filtr wykrywający krawędzie 0°)

```
Image image=new Image();
gBase.convolve(image,image, StockKernel.skVerEdge);
```

Binaryzacja obrazu.

Obraz pobierany z kamery CCD akwizowny jest przez VFG w sposób zależny od ustawionego modu karty i sposobu zaprogramowania LUT wejściowego. W większości przypadków pobierany jest obraz szaroodcieniowy o ustalonej dynamice jasności, który dopiero później poddawany jest dalszej obróbce. Dla podstawowych zadań numerycznej analizy obrazów takich jak analiza kształtów, detekcja krawędzi, rozpoznawanie liter, znajdowanie środka ciężkości obiektu obraz taki niesie z sobą zbyt wiele informacji utrudniając osiągnięcie zamierzonego celu. W takich wypadkach przed właściwą analizą obrazu stosuje się obróbkę wstępną mającą na celu segmentację pobranego obrazu, czyli podzielenie obrazu na obszary o jednorodnych wartościach wybranego parametru (np. intensywności). W szczególności segmentacja obrazu mająca na celu oddzielenia obiektów od tła używana jest do binaryzacji obrazu, tj. zamiany obrazu szaroodcieniowego na obraz binarny. Dokonuje się tego poprzez ustawienie poziomu odcięcia (*treshold level*) i zamianę pikseli o intensywności poniżej poziomu na **0** a pikseli powyżej poziomu na **1.** W wyniku operacji otrzymujemy obraz gotowy do analizy, którego jakość zależy od trafności doboru wartości poziomu odcięcia.
Jednym z fundamentalnych problemów jest ustalenie właściwego poziomu odcięcia dokonuje się tego między innymi poprzez analizę histogramu lub scattergramu obrazu.

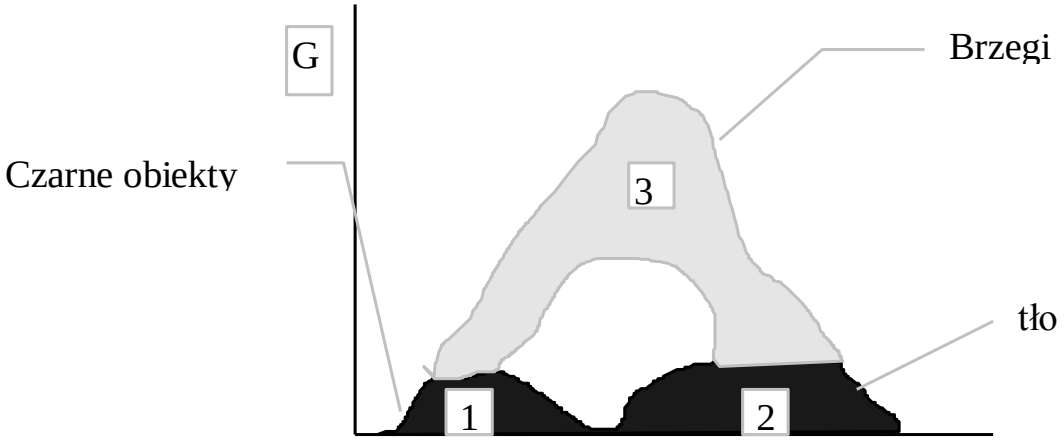
Automatycznej binaryzacji dokonuje się za pomocą funkcji

dynamicThreshold(Image* dest, Image* source)

należącej do klasy gBase.

Scattergram

Proces polega olega na znalezieniu w obrazie miejsc z max gradientu intensywności (brzegi). Analizuje się wartości intensywności w tych obszarach pomijając pozostałe tworząc mapę dwuwymiarową rozkładu gradientu intensywności w funkcji intensywności.



Metoda wykonania odcięcia na podstawie scattergramu jest następująca:

- należy zlokalizować piksele o dużym rozkładzie intensywności,
- określić histogram dla tych pikseli i ich sąsiedztwa,
- wtedy dwa główne piki są znacznie osłabione.

Zadania do wykonania (2 zajęcia po 4 h każde):

1) Zapis obrazu do pliku BMP.
2) Obliczenie i wyświetlenie densytogramu według zadanego przekroju wraz z zapisem do pliku. Zapis reprezentatywnego zdjęcia i jego densytogramu do sprawozdania.
3) Obliczenie i wyświetlenie histogramu na żywo z zaznaczonego obszaru wraz z zapisem do pliku. Zapis reprezentatywnego zdjęcia i jego histogramu do sprawozdania.
4) Złapanie przykładowego obrazu z przeplotem.
5) Ustawienie rejestrów LUT. Wykorzystanie palet predefiniowanych: negatyw, nieliniowe, obraz pseudokolorowy, tęcza.
6) Porównanie obrazków przy małym i dużym wzmocnieniu (manipulacja przesłoną albo filtrami szarymi). W sprawozdaniu zestawiamy powiększone fragmenty zdjęć przedstawiających ten sam obiekt (półcień) przy małym i dużym szumie. Uwaga - prowadzący powinien włączyć AutoGain kamery (pin 4 w górę).
7) Filtry splotowe. Filtracje górno i dolno przepustowe. Wykrywanie krawędzi obiektów. Zapis zdjęć obrazujących dobrze działanie danego filtra, obrazy wejściowe w wersji szaroodcieniowej i binarnej.
8) Wykonanie kombinacji erozji i dylatacji na tym samy obrazie - porównanie możliwości wykrywania krawędzi z filtrami z pkt. 7.
9) "Oko żaby" na żywo (odejmowanie dwóch kolejnych klatek). Wykrywanie poruszających się fragmentów obrazu.
10) Poprawa obrazu przez odejmowanie wolnozmiennego tła. Zapis tła w pierwszej kolejności, nastepnie odejmowanie go od użytecznego sygnału. Zademonstrować działanie algorytmu na żywo na odpowiednio dobranej scenie widzianej przez kamerę.
11) Scattergram. Określenie poziomu odcięcia na podstawie histogramu i scattergramu.
12) Binaryzacja automatyczna – sformułowanie wniosków co do poprawności odcięcia wg scattergramu, histogramu i automatycznego. Do sprawozdania zdjęcia z odcięciem tego samego obrazu szaroodcieniowego trzema metodami.
13) Inspekcja: wyszukiwanie brakującego/uszkodzonego elementu na płytce drukowanej poprzez odejmowanie obrazów.

Wszystkie etapy muszą być poparte zdjęciami własnoręcznie dobranych plansz/obiektów - co świadczy o zrozumieniu problemu.

Pracownia Informatyki Optycznej WF PW
Marzec 2015