

Modelowanie matematyczne

Dokumentacja projektu końcowego

Tytuł: **Wybrane algorytmy przetwarzania obrazów**

Autor: Dawid Bitner, Daniel Broczkowski, Damian Kwaśniok

Kierunek: Informatyka, studia 2 stopnia (sem.II)

Cel zadania:

Celem zadania było zaprezentowanie wybranych algorytmów przetwarzania obrazów dostępnych w programie Mathematica.

Opis:

W ramach projektu zostały zaprezentowane następujące algorytmy:

1. Metoda odsumowania obrazu poprzez zastosowanie nielokalnego filtru średnich

Nielokalny filtr średnich opiera się na średniej liczonej nie na lokalnym obszarze obrazu, lecz na średniej wszystkich pikseli na obrazie. Skutkuje to większą przejrzystością po filtrowaniu i mniejszą utratą szczegółów obrazu w porównaniu z filtrami lokalnymi.

Wartości pikseli są obliczane ze wzoru:

$$u(p) = \frac{1}{C(p)} \int_{\Omega} v(q) f(p, q) dq$$

gdzie:

- Ω – obszar obrazu
- p i q – piksele obrazu
- $f(p, q)$ – funkcja ważenia określająca jak ściśle są powiązane punkty na obrazie, najczęściej używana jest funkcja ważenia Gaussa:

$$f(p, q) = e^{-\frac{|B(q)-B(p)|^2}{h^2}}, \text{ gdzie:}$$

- h - parametr filtrowania (najczęściej odchylenie standardowe)
- $B(p)$ – lokalna średnia wartość pikseli obrazu w otoczeniu p
- Całka jest liczona $\forall q \in \Omega$
- $C(p)$ – współczynnik normalizacyjny wyznaczany z wzoru:

$$C(p) = \int_{\Omega} f(p, q) dq$$

Efekt zastosowania tego filtra jest widoczny gołym okiem:



2. Alorytm konwersji obrazu na grafikę wektorową

Wektoryzacja obrazu, czyli konwersja grafika rastrowej na wektorową ma zastosowanie przy tworzeniu clip-artów, projektowaniu grafik i różnych produktów.

Przekonwertowane zdjęcie traci niektóre szczegóły, wygładza często powierzchnie obiektów.

Algorytm stosuje filtr przepływu krzywizny, który w swoim działaniu wykorzystuje równanie różniczkowe:

$$\delta_t f = \kappa |\nabla f|$$

z krzywizną konturu: $\kappa = \nabla \cdot \frac{\nabla f}{|\nabla f|}$ dla każdego kanału obrazu f .

Filtr przepływu krzywizny wygładza obraz przy jednoczesnym zachowaniu krawędzi. Skutecznie rozkłada krzywiznę wzdłuż konturu, zaokrąglając w ten sposób rogi i zmniejszając euklidesową długość konturu.

Następnie z grafiki z pomocą metody DominantColors wyszukiwane jest n najbardziej dominujących kolorów z minimalną odległością kolorów = d .

Algorytm ten przetestowano dla danych $n = 32$ i odległości $d = 0.05$.

Została również przygotowana procedura Manipulate sterująca parametrami n oraz d .

Przykładowe otrzymane wyniki:

filtr przepływu krzywizny



dominujące kolory

|minimalna odległość między kolor

{ , , , , , , , , , , , , , , , , , }

grafika wektorowa



[zmieniaj](#)

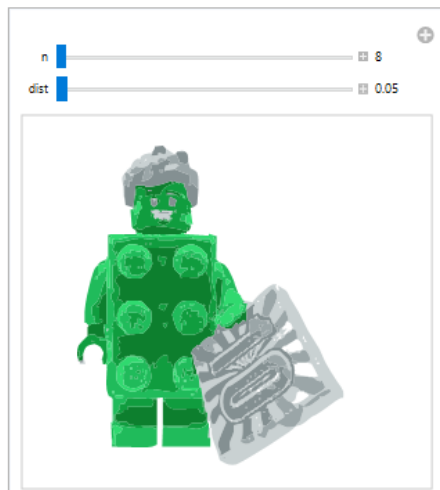
dominujące kolory

|minimalna odległość między kolorami

grafika wektorowa

[wygląd

[wygląd



3. Metoda zmiany nasycenia barw z palety RGB na obrazach

Wbudowana w Mathematica funkcja **ImageApply** pozwala na zastosowanie na obrazie różnorodnych filtrów, które można również samemu zdefiniować.

Przykładowym filtrem może być operacja zerująca wartości nasycenia koloru czerwonego i zwiększeniu wartości nasycenia koloru niebieskiego. Opisuje to funkcja:

$$f(\{r, g, b\}) = \{0, g, b^2\}$$

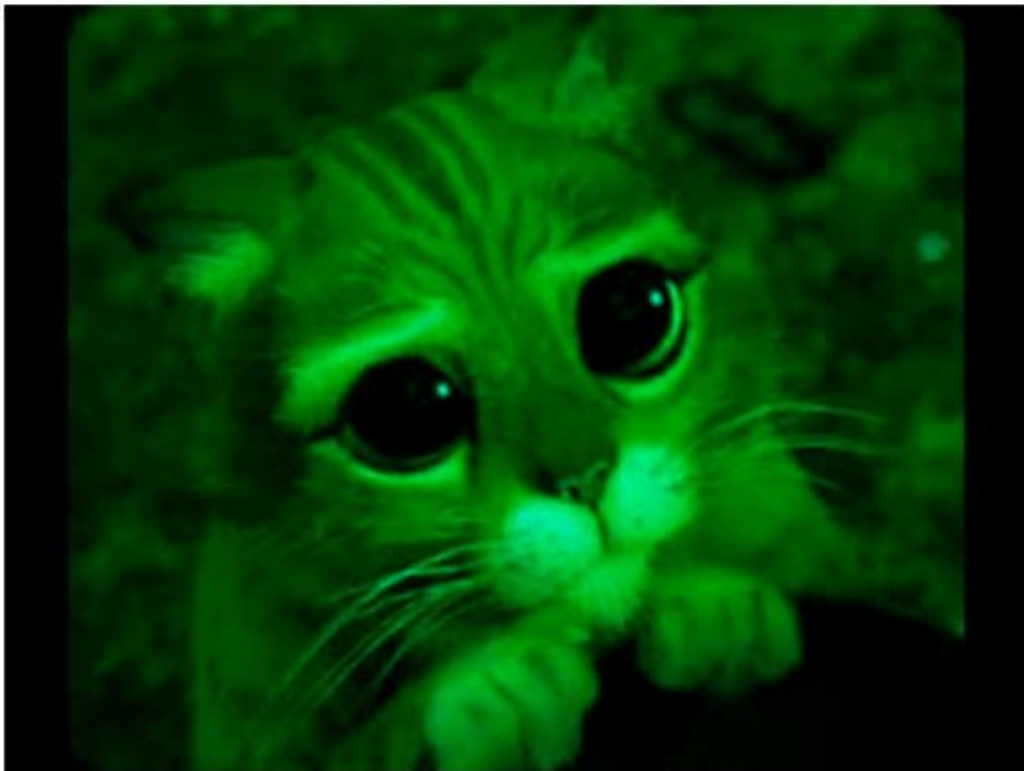
Efekt zastosowania takiego przekształcenia jest następujący:



```
blueBoostRedDrop[{r_, g_, b_}] := {0, g, b^2};
```

```
ImageApply[blueBoostRedDrop, kotek]
```

[zastosuj do obrazu](#)



4. Prezentacja domyślnych filtrów wbudowanych w Mathematica

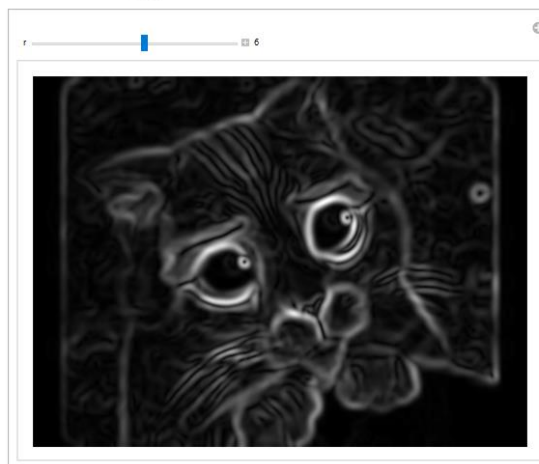
a) Filtr gradientu

Filtr gradientu służy do wykrywania krawędzi na obrazie. Wartości pikseli są obliczane jako norma euklidesowa gradientu g w pozycji piksela, aproksymowaną przy użyciu dyskretnych pochodnych Gaussa w każdym wymiarze:

$$\sqrt{\left(\frac{di}{dx}\right)^2 + \left(\frac{dy}{dy}\right)^2}$$

Funkcja ta przyjmuje jeden argument r , który jest promieniem próbki. Została przygotowana instrukcja Manipulate sterująca wartością tego parametru.

```
Manipulate[GradientFilter[kotek, r] // ImageAdjust  
  {zmieniaj, filtr gradientu, popraw obraz  
    , {r, 1, 10, 1, Appearance -> "Labeled"}}  
  {wygląd}
```



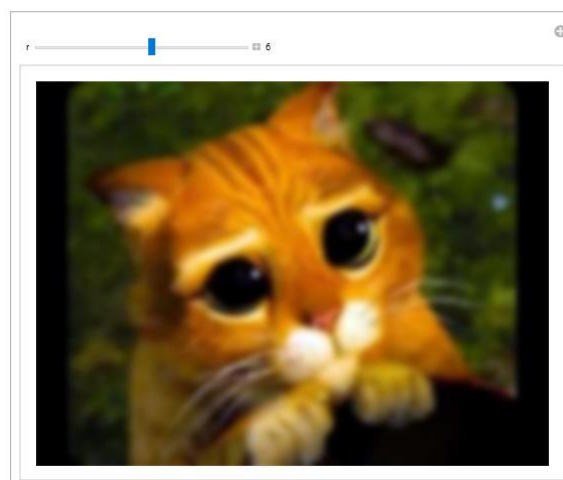
b) Filtr Gaussa

Filtr Gaussa służy to rozmywania obrazów. Przyjmuje jeden parametr r , który odpowiada za promień próbki. Wartości pikseli obliczane są ze wzoru:

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}},$$

gdzie odchylenie standardowe $\sigma = \frac{r}{2}$

```
Manipulate[GaussianFilter[kotek, r] // ImageAdjust  
  {zmieniaj, filtr Gaussa, popraw obraz  
    , {r, 1, 10, 1, Appearance -> "Labeled"}}  
  {wygląd}
```



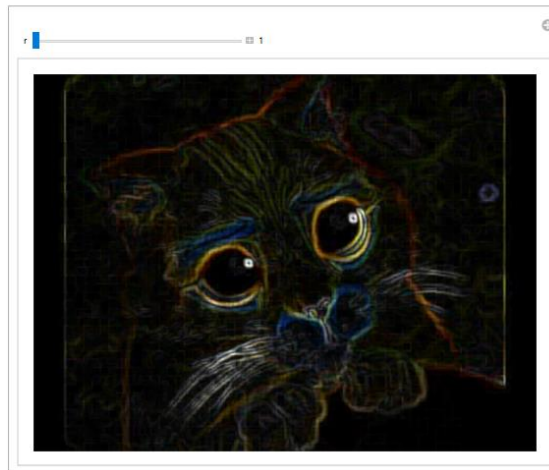
c) Filtr zakresu

Filtr zakresu służy do wykrywania lokalnych skoków w danych, gdzie wielkość lokalnego sąsiedztwa zależy od wartości promienia r .

Wartości obliczane są jako różnica między maksymalną i minimalną wartością w danym sąsiedztwie.

W celu prezentacji została przygotowana instrukcja Manipulate sterująca parametrem r :

```
Manipulate[RangeFilter[kotek, r] // ImageAdjust  
[zmieniaj] [filtr zakresu] [popraw obraz  
, {r, 1, 10, 1, Appearance -> "Labeled"}]  
[wygląd]
```



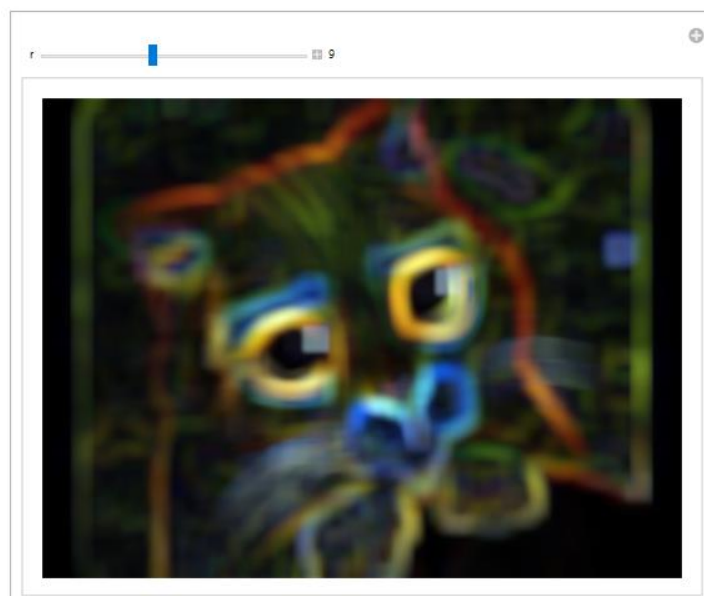
d) Filtr odchylenia standardowego

Filtr odchylenia standardowego służy do wykrywania lokalnego rozproszenia danych w danym otoczeniu wyznaczanym przez promień r .

Wartość danego piksela jest obliczana jako odchylenie standardowe punktów w wyznaczonym sąsiedztwie.

W celu prezentacji została przygotowana instrukcja Manipulate sterująca parametrem r :

```
Manipulate[StandardDeviationFilter[kotek, r] // ImageAdjust  
[zmieniaj] [filtr odchylenia standardowego] [popraw obraz  
, {r, 1, 20, 1, Appearance -> "Labeled"}]  
[wygląd]
```



5. Prezentacja efektów graficznych dostępnych w Mathematica

a) Efekt „Charcoal” – szkic

```
ImageEffect[kotek, "Charcoal"]
```

[zastosuj efekt do obrazu](#)



b) Efekt „Embossing” – wytłoczenie

```
ImageEffect[kotek, "Embossing"]
```

[zastosuj efekt do obrazu](#)



c) Efekt „OilPainting” – farba olejna

```
ImageEffect[kotek, "OilPainting"]
```

[zastosuj efekt do obrazu](#)



d) Efekt „Posterization” – posteryzacja

```
ImageEffect[kotek, "Posterization"]
```

[zastosuj efekt do obrazu](#)



e) Efekt „Solarization” – solaryzacja

```
ImageEffect[kotek, "Solarization"]
```

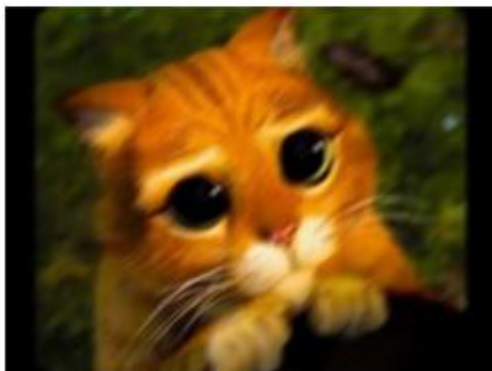
[zastosuj efekt do obrazu](#)



f) Efekt „MotionBlur” – rozmycie ruchu

```
ImageEffect[kotek, "MotionBlur"]
```

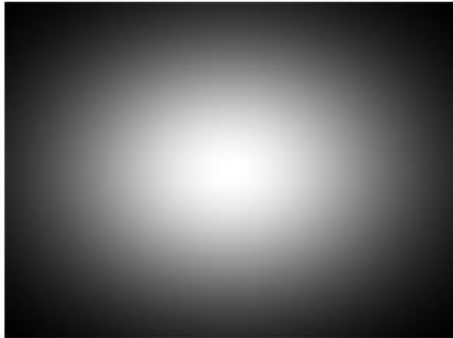
[zastosuj efekt do obrazu](#)



6. Zastosowanie masek w przetwarzaniu grafik w Mathematica

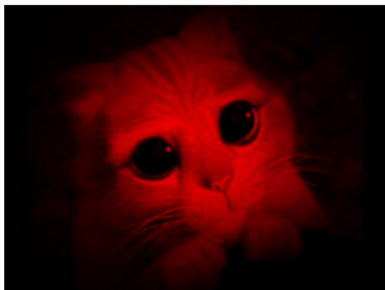
Funkcjonalność przetwarzania obrazu w Mathematica umożliwia stosowanie masek, na przykład w celu tworzenia efektów winietowych. W tym celu należy wygenerować maskę. Można wybrać na przykład proste cieniowanie w rogach korzystając z maski Gaussa. Następnie wygenerowaną maskę należy przemnożyć przez wartości pikseli obrazu.

```
myMask = ImageResize[ImageAdjust@Image[GaussianMatrix[{240, 130}]], ImageDimensions[kotek]]
ImageMultiply[kotek, myMask]
```



Można również zastosować kolorowe maski:

```
angryKitkuMask = ImageResize[ImageMultiply[ImageAdjust@Image[GaussianMatrix[{240, 130}]], Red], ImageDimensions[kotek]];
ImageMultiply[kotek, angryKitkuMask]
```

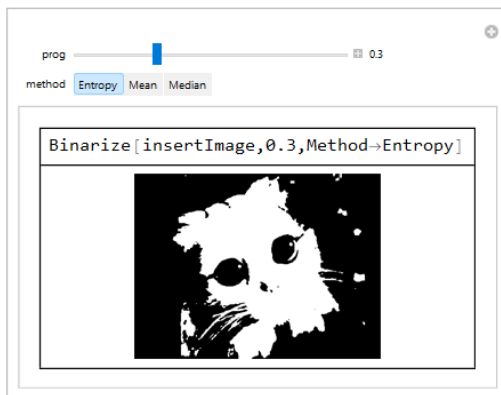


7. Progowanie (binaryzacja) obrazu

Algorytm binaryzacji polega na konwersji obrazu do skali szarości, a następnie tworzeniu obrazu, w którym każdy piksel ma wartość 0 lub 1. Przypisywanie wartości binarnych może odbywać się za pomocą analizy różnych danych statystycznych takich jak na przykład entropia, średnia czy mediana.

W celu prezentacji algorytmu została przygotowana instrukcja Manipulate sterująca wartością progową oraz metodą statystyczną użytą do wyznaczenia wartości binarnych:

```
Manipulate[Grid[{{Row[{"Binarize[insertImage,", prog, ",Method->", method, ""]}},  
  {Binarize[kotek, prog, Method->method]}], Frame->All, Spacings->{1, 1}},  
  {{prog, .6, "prog"}, 0, 1, .1, Appearance->"Labeled"}, {{method, "Cluster", "method"}, {"Entropy", "Mean", "Median"}},  
  TrackedSymbols->{method, prog}]
```



8. Metody poprawiania niedoświetlonych zdjęć

TODO: Pod linkiem poniżej wytłumaczenie, jakoś trzeba streścić, opisać 😊

<https://www.wolfram.com/language/12/new-in-image-processing/arithmetic-for-fast-algorithm-prototyping.html?product=language>

Literatura:

- <https://reference.wolfram.com/language/example/ImageDenoising>
- <https://www.wolfram.com/language/12/new-in-image-processing/convert-images-to-vector-graphics.html?product=language>
- <https://reference.wolfram.com/language/guide/ImageFilteringAndNeighborhoodProcessing.html>
- <https://reference.wolfram.com/language/ref/ImageEffect.html>
- <https://reference.wolfram.com/language/ref/Binarize.html>
- <https://www.wolfram.com/language/12/new-in-image-processing/arithmetic-for-fast-algorithm-prototyping.html?product=language>

Załącznik:

— Plik z programem (MM_Bitner_Broczkowski_Kwasniok_projekt.nb)