Ćwiczenia 4a: Obrazy i wykresy, cd.
Notatki

Funkcja image

```
clear

load earth
image(X); colormap(map)
axis image

load mandrill
%figure('color','k')
image(X)
colormap(map)
axis off        % Remove axis ticks and numbers
axis image      % Set aspect ratio to obtain square pixels

figure
ax(1) = subplot(1,2,1);
rgb = imread('ngc6543a.jpg');
image(rgb); title('RGB image')
ax(2) = subplot(122);
im = mean(rgb,3);
image(im); title('Intensity Heat Map')
colormap(hot(256))
linkaxes(ax,'xy')
axis(ax,'image')
```

Funkcja colormap

```
figure; image(X); colormap(map);          % mandrill

colormap(gray)
colormap(cool)
colormap(gray(2))
colormap(gray(32))
colormap(gray(220))
colormap(cool(220))
colormap(bone(220))
colormap(autumn(220))

doc colormap
doc image
```

Funkcja imagesc

Działa jak image, ale przed rysowaniem przeskalowuje obrazek tak, żeby wykorzystać całą mapę kolorów.

```
colormap(gray(512))
imagesc(X); axis image
image(X); axis image
```

Rysunki trójwymiarowe

mesh, meshc, meshz	Mesh plots
peaks	Example function of two variables
surf, surfc	3-D shaded surface plot
surfl	Surface plot with colormap-based lighting
meshgrid	Generate X and Y arrays for 3-D plots
view	Viewpoint specification

```
colormap('default')
z = peaks
mesh(z)
surf(z)
```

```
[X,Y] = meshgrid(-3:.125:3);
Z = peaks(X,Y);
meshc(X,Y,Z);
axis([-3 3 -3 3 -10 5])
```

Ćwiczenia 4b: Sterowanie programem, skrypty, funkcje Notatki

Funkcja find

```
x = 3*rand(1,7)
ind = find(x > 1)
y = x(ind)

A = rand(3)
find(A>0.5)
[i,j,x] = find(A>0.5)

for n=1:length(x)
     disp(A(i(n), j(n)))
end
```

Pętla for

```
n = 10
x = []
for i = 1:n
    x = [x, i^2]
end

n = 10
x = []
for i = n:-1:1
    x = [i^2, x]
end
```

Pętla while

```
a = 1e9
n = 0
while 2^n <= a
    n = n + 1 ;
end
n
```

Można to policzyć szybciej przy użyciu funkcji log2

```
  [f,n] = log2(a)
```

Wyrażenie if

```
if n < 0
    parity = 0 ;
elseif rem(n,2) == 0
    parity = 2 ;
else
    parity = 1 ;
end
```

```
% a program to evolve a wave in excitable media
size_of_lattice = 400;
half_size = floor(size_of_lattice/2);
max_time = 500;
every_time = 1;
prob = 0.5;
refr_time = 9;

lattice = zeros(size_of_lattice+2);
newlattice = zeros(size_of_lattice+2);

% random initial state
lattice = floor((refr_time+2)*rand(size_of_lattice+2));
% middle site set
%lattice(20,35)=refr_time+1;

imagesc(refr_time+1-lattice);

for time=2:max_time
  %warunki brzegowe
  lattice(1,:) = lattice(size_of_lattice+1,:);
  lattice(size_of_lattice+2,:) = lattice(2,:);
  lattice(:,1) = lattice(:,size_of_lattice+1);
  lattice(:,size_of_lattice+2) = lattice(:,2);

  for n=2:size_of_lattice+1
    for m=2:size_of_lattice+1
      if (lattice(n,m)>0)
          newlattice(n,m) = lattice(n,m)-1;
      elseif ( lattice(n-1,m) == refr_time+1 || ...
               lattice(n+1,m) == refr_time+1 || ...
               lattice(n,m-1) == refr_time+1 || ...
               lattice(n,m+1) == refr_time+1)
          newlattice(n,m) = refr_time+1;
      end % if
    end % for m
  end % for n

  lattice = newlattice;

  if mod(time,every_time) == 0
    imagesc(refr_time+1-lattice);
    drawnow
  end % if
end % for time
```

```
% automaty komórkowe 1D
nr_of_cells=101;
max_time=200;
bias=0.78;
prob=0.9;

universe = zeros(nr_of_cells+2,max_time);
%universe(2:nr_of_cells+1,1) = floor(2*rand(nr_of_cells,1));
%universe(1+find(rand(nr_of_cells,1)>bias),1)=1;
universe((nr_of_cells+1)/2)=1;

colormap(gray);

for time=2:max_time
    %universe(1+find(rand(nr_of_cells,1)<prob),time) = 1;
    for n=2:nr_of_cells+1
        if (universe(n-1,time-1) == 0 && ...
                universe(n ,time-1) == 0 && ...
                universe(n+1,time-1) == 0 )
            universe(n,time) = 0;
        end
        if (universe(n-1,time-1) == 0 && ...
                universe(n ,time-1) == 0 && ...
                universe(n+1,time-1) == 1 )
            universe(n,time) = 1;
        end
        if (universe(n-1,time-1) == 0 && ...
                universe(n ,time-1) == 1 && ...
                universe(n+1,time-1) == 0 )
            universe(n,time) = 1;
        end
        if (universe(n-1,time-1) == 0 && ...
                universe(n ,time-1) == 1 && ...
                universe(n+1,time-1) == 1 )
            universe(n,time) = 1;
        end
```

```
       if (universe(n-1,time-1) == 1 && ...
                universe(n ,time-1) == 0 && ...
                universe(n+1,time-1) == 0 )
            universe(n,time) = 1;
        end
        if (universe(n-1,time-1) == 1 && ...
                universe(n ,time-1) == 0 && ...
                universe(n+1,time-1) == 1 )
            universe(n,time) = 0;
        end
        if (universe(n-1,time-1) == 1 && ...
                universe(n ,time-1) == 1 && ...
                universe(n+1,time-1) == 0 )
            universe(n,time) = 0;
        end
        if (universe(n-1,time-1) == 1 && ...
                universe(n ,time-1) == 1 && ...
                universe(n+1,time-1) == 1 )
            universe(n,time) = 0;
        end
    end
end
imagesc(1-universe')
```

Zadanie domowe

Napisać skrypt, który symuluje wybrany automat komórkowy (1D albo 2D). Można posłużyć się powyższymi przykładami jako punktem wyjścia. Wyniki symulacji powinny być zapisywane w postaci filmu. Raport będzie ciekawszy jeżeli opiszecie Państwo wasze doświadczenia z symulacji różnych wariantów automatów (np. różne reguły 1D, albo różne parametry w 2D, itp.).