Complex numbers and image analysis

Henrik Skov Midtiby

Maersk McKinney Moller Institute
hemi@mmmi.sdu.dk

2015-05-28

What is a complex number?

Complex numbers let us solve certain equations like

$$x^2 = -4$$

by introducing the number i, with the property $i^2 = -1$.

Matlab code example

```
>> x = 2i
>> x^2
x =
   0.0000 + 2.0000i
ans =
    -4

>> x = 1 + 1i
>> x^2
x =
   1.0000 + 1.0000i
ans =
   0.0000 + 2.0000i
```

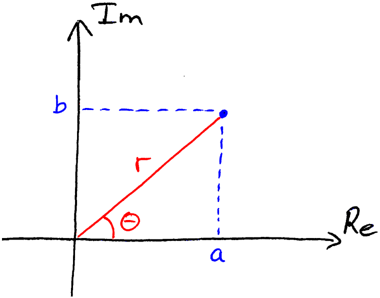
Dual representation

Cartesian form

$$z = a + ib$$

Polar form

$$z = r \cdot e^{i\theta}$$



Eulers identity

Eulers identity

$$e^{i\theta} = \cos(\theta) + i\sin(\theta)$$

describes how to calculate complex exponentials.

Oscillating exponentials

$$e^{ikt} \qquad k \in \mathbb{R}$$

Complex exponentials "oscillate" and are simpler to do calculations with than sine and cosine.

Matlab code example

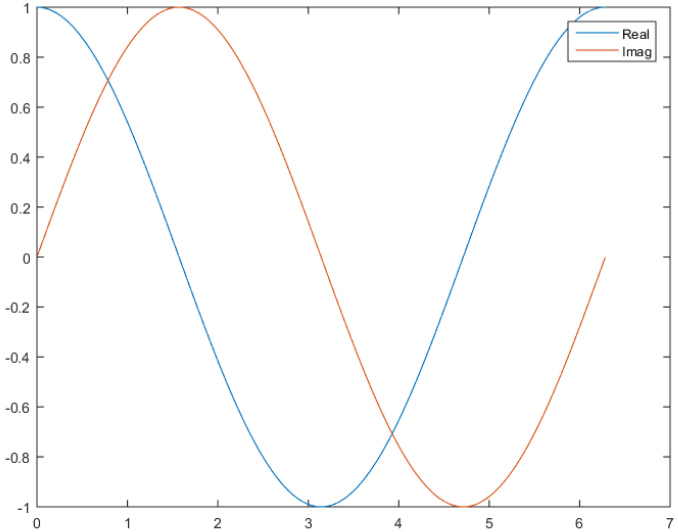
To plot the oscillating function

$$f(t) = e^{it}$$

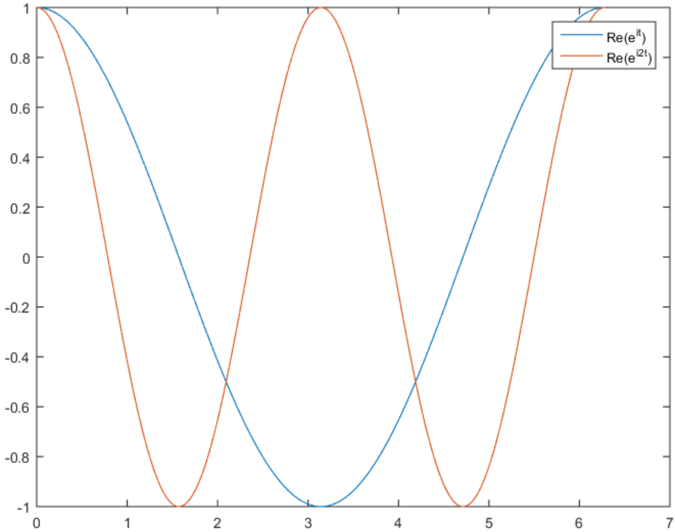
The following code is used

```
x = linspace(0, 2*pi, 1000);
x = x(2:end);
ex = exp(1i * x);
plot(x, real(ex)); hold on;
plot(x, imag(ex)); hold off;
legend('Real', 'Imag')
```

Oscillating function



Two oscillating functions



Representations of a periodic function

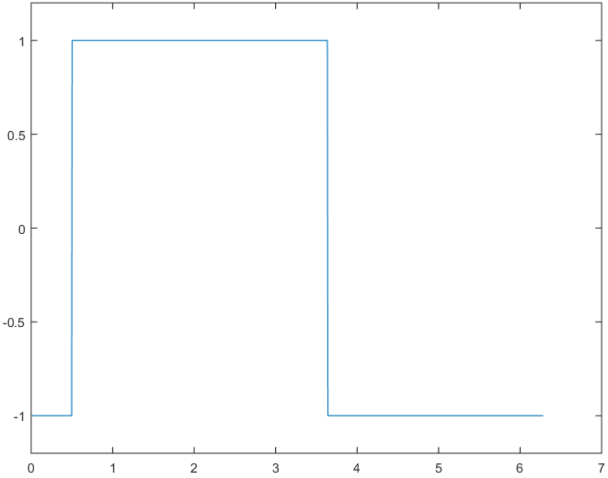
A periodic function, $f(t)$, with period 2π can be represented af a sum of oscillating functions with different frequencies.
Sine and cosine based

$$\begin{aligned} f(t) = a_1 \cdot \sin(1t) + a_2 \cdot \sin(2t) + a_3 \cdot \sin(3t) + \ldots \\ b_0 + b_1 \cdot \cos(1t) + b_2 \cdot \cos(2t) + \ldots \end{aligned}$$

Based on complex exponentials

$$\begin{aligned} f(t) = c_1 \cdot e^{1ti} + c_2 \cdot e^{2ti} + c_3 \cdot e^{3ti} + \ldots \\ c_0 + c_{-1} \cdot e^{-1ti} + c_{-2} \cdot e^{-2ti} + c_{-3} \cdot e^{-3ti} + \ldots \end{aligned}$$

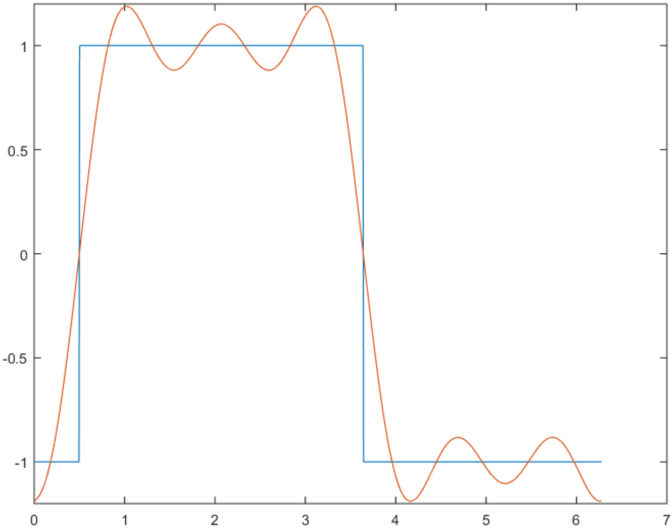
Example of a periodic function



Question: How to locate the center of the peak?

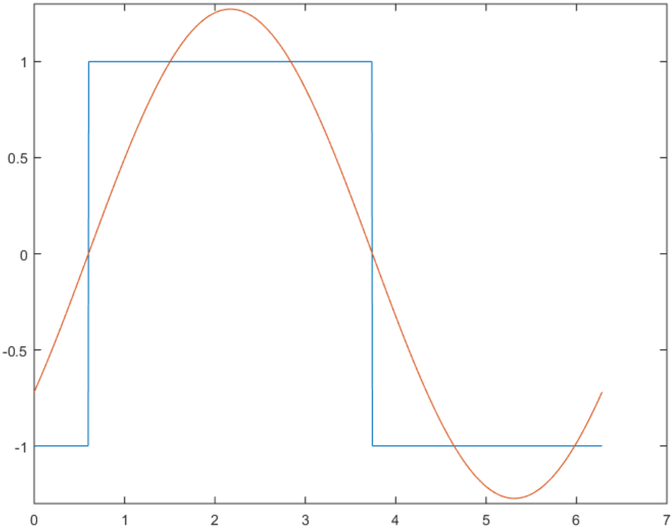
Alternative representation

Fourier series with five terms



Alternative representation

Fourier series with one term



Orthogonal functions

The complex exponentials e^{nti} and e^{mti} are orthogonal with respect to the inner product

$$\langle f(t), g(t) \rangle = \int_0^{2\pi} f(t) \cdot (g(t))^* \, dt$$

if and only if $n \neq m$.

$$\begin{aligned} \langle e^{nti}, e^{mti} \rangle &= \int_0^{2\pi} e^{nti} \cdot (e^{mti})^* \, dt \\ &= \int_0^{2\pi} e^{(n-m)ti} \, dt \end{aligned}$$

For example

$$\langle e^{2ti}, e^{2ti} \rangle = 2\pi \qquad \langle e^{2ti}, e^{3ti} \rangle = 0$$

How to determine c_n?

The orthogonal property of the complex exponentials allows us to calculate c_n as follows:

$$
\begin{aligned}
f(t) &= c_{n-1} \cdot e^{(n-1)ti} + c_n \cdot e^{nti} + c_{n+1} \cdot e^{(n+1)ti} \\
\langle e^{nti}, f(t) \rangle &= c_{n-1} \cdot \langle e^{nti}, e^{(n-1)ti} \rangle + \\
&\quad c_n \cdot \langle e^{nti}, e^{nti} \rangle + c_{n+1} \cdot \langle e^{nti}, e^{(n+1)ti} \rangle \\
&= c_n \cdot \langle e^{nti}, e^{nti} \rangle \\
c_n &= \frac{\langle e^{nti}, f(t) \rangle}{\langle e^{nti}, e^{nti} \rangle}
\end{aligned}
$$

Matlab code example

```
x = linspace(0, 2*pi, 1001);
x = x(1:(end - 1));
fx = sign(sin(x - 0.6));

ex1 = exp(1i * x );
a1 = fx * ex1' / 1000;
exm1 = exp(-1i * x );
am1 = fx * exm1' / 1000;

peaklocation = -angle(a1)
peaklocation = angle(am1)

plot(x, fx); hold on;
plot(x, real(a1 * ex1 + am1 * exm1))
```

Determined peak location

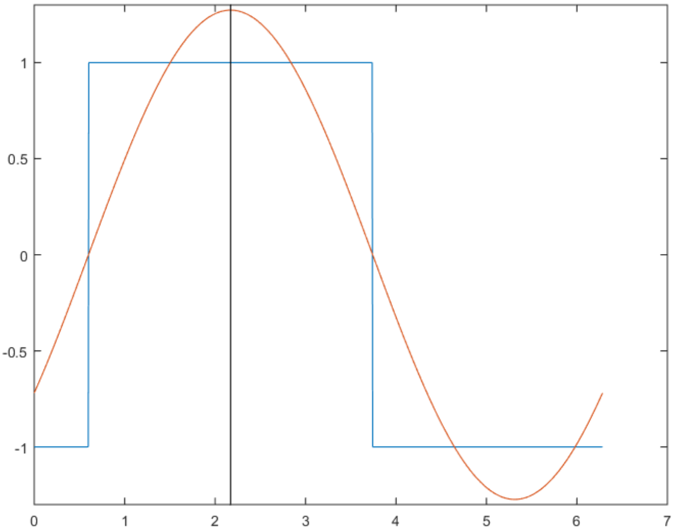
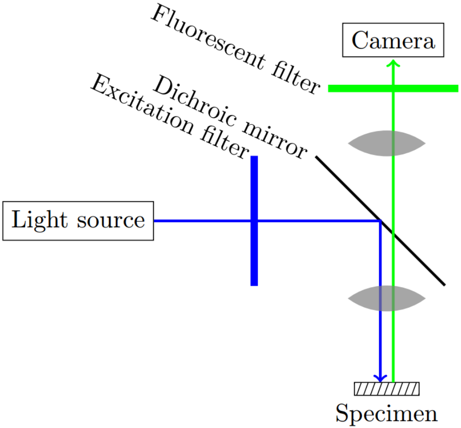
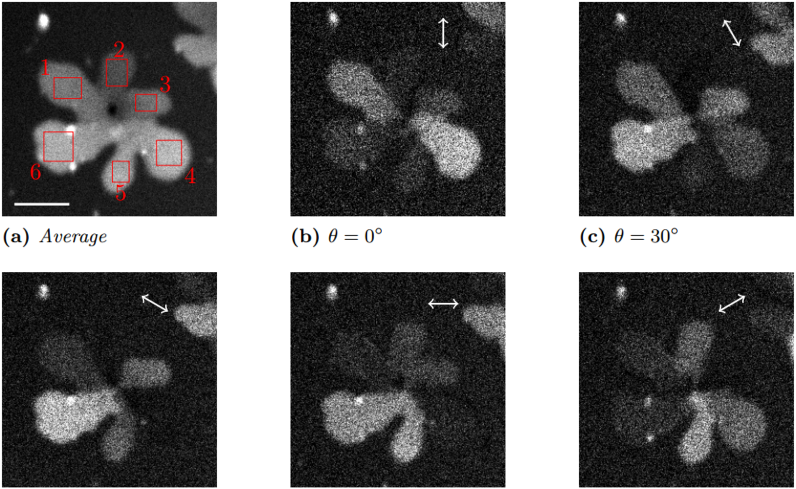


Image of experimental setup



Polarization angles



(a) *Average* (b) $\theta = 0°$ (c) $\theta = 30°$

(d) $\theta = 60°$ (e) $\theta = 90°$ (f) $\theta = 120°$

Matlab code example

Load image stack

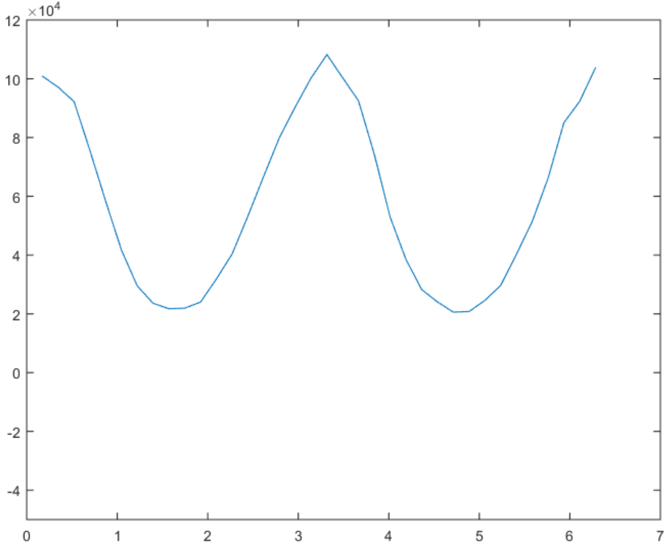
```
imagestack = [];
for k=1:36
    filename = sprintf('img/%02d.png', k);
    img = imread(filename);
    imagestack(:, :, k) = img;
end
```

Matlab code example

Plot intensity variations of a region in the image sequence

```
vals = [];
for k = 1:36
    temp = imagestack(181:207, 162:193, k);
    vals(k) = sum(temp(:));
end
plot(vals);
```

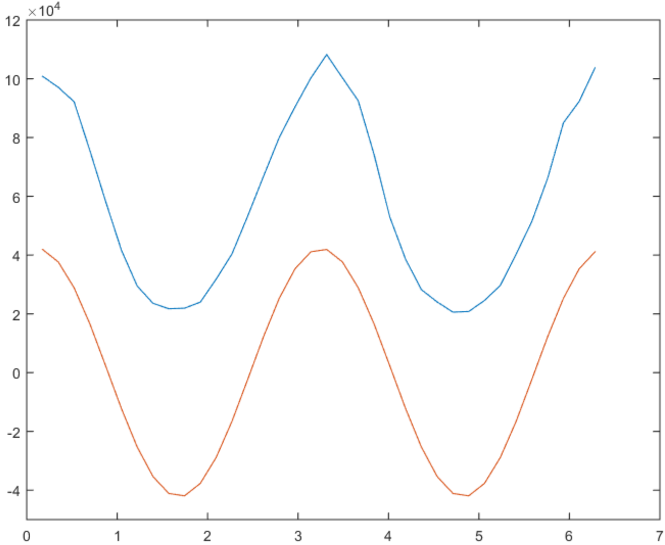
Intensity variations



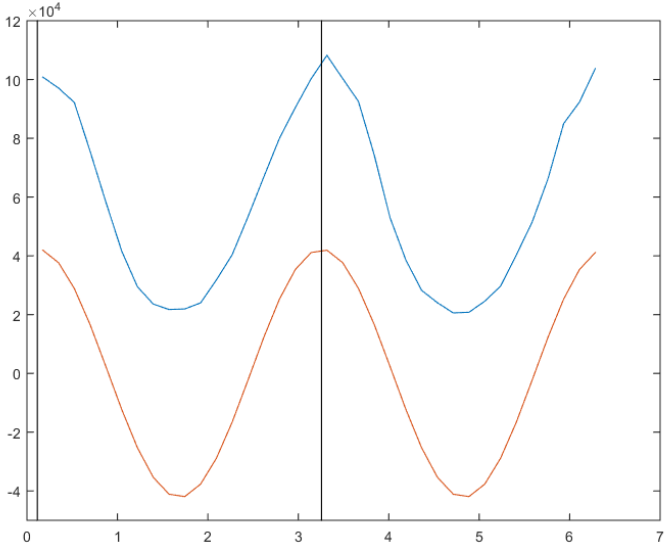
Matlab code example

```
x = (1:36) * 2 * pi / 36

ex2 = exp(2i * x );
a2 = vals * ex2' / 36
exm2 = exp(-2i * x );
am2 = vals * exm2' / 36

peaklocation = -angle(a2) / 2

plot(x, real(a2 * ex2 + am2 * exm2))
```

Second harmonics



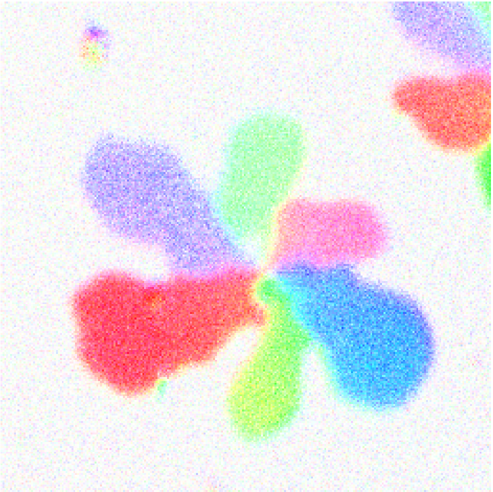
Located peaks



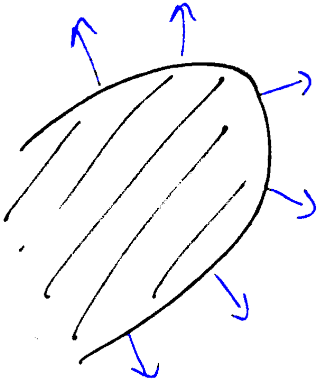
Matlab code example

```
phaseimage = imagestack(:, :, 1) * 0;
for k = 1:36
    phaseimage = phaseimage + ...
        imagestack(:, :, k) * exp(1i * k * 2 * 2*pi / 36);
end

absPhaseImage = abs(phaseimage);
absPhaseImage = absPhaseImage / max(absPhaseImage(:));
argPhaseImage = angle(phaseimage);

rgbimage = hsv2rgb((argPhaseImage + pi)/(2*pi), ...
        1+0*argPhaseImage, 1 + 0*argPhaseImage);
image(rgbimage);
```

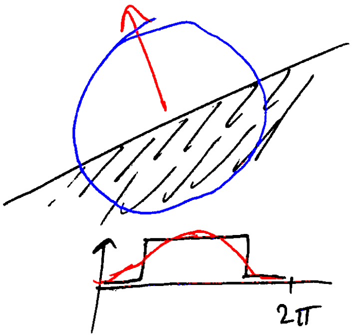
Located orientations



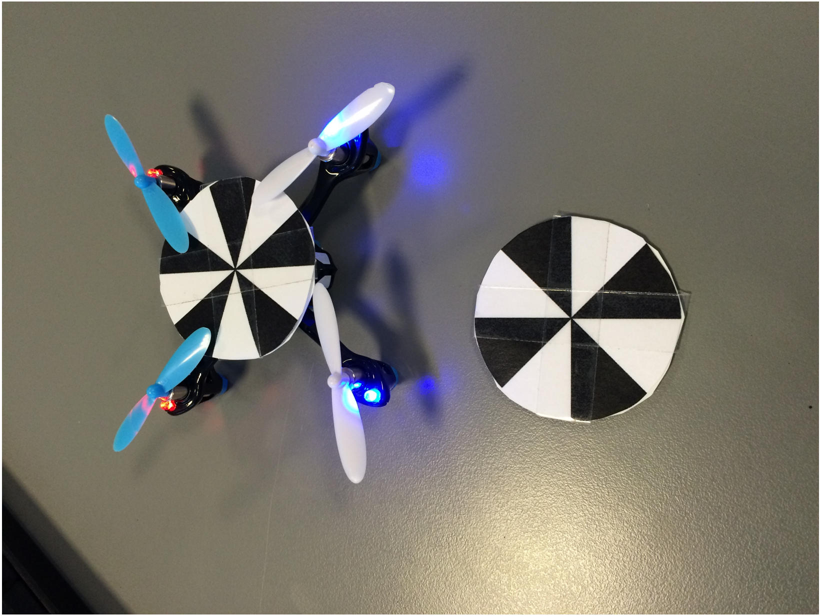
Building block of an algorithm from my phd



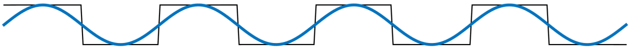
Fourier transform around points in an image



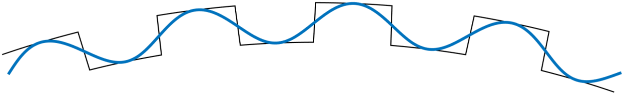
Marker localisation



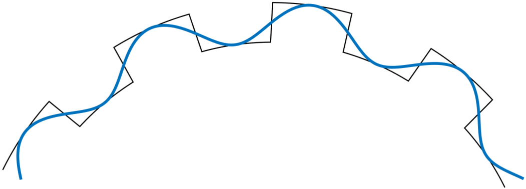
Bending space



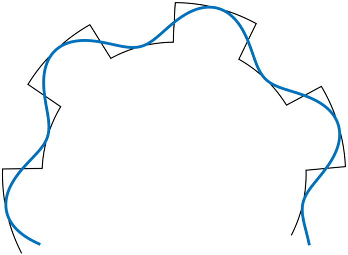
Bending space



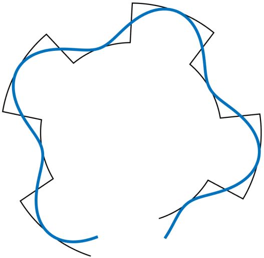
Bending space



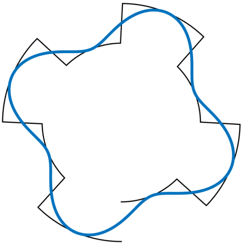
Bending space



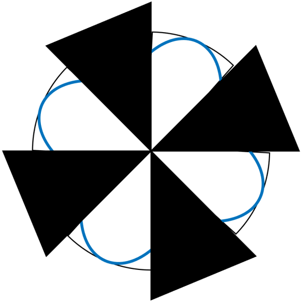
Bending space



Bending space



Final marker

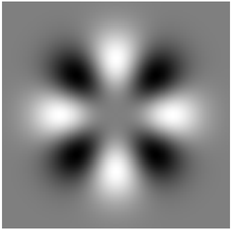


Matlab code example

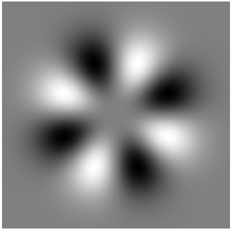
```
kernel = genSymDetectorKernel(4, 150);

function kernel = genSymDetectorKernel(order, kernelsize)

stepsize = 2 / (kernelsize-1);
temp1 = meshgrid(-1:stepsize:1);
kernel = temp1 + 1i*temp1';

magni = abs(kernel);
kernel = kernel.^order;
kernel = kernel.*exp(-8*magni.^2);
abs(kernel);

end
```

Marker localization

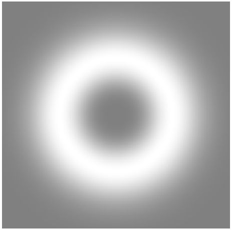
Real part of kernel



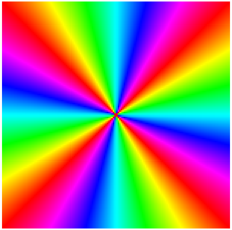
Imaginary part of kernel



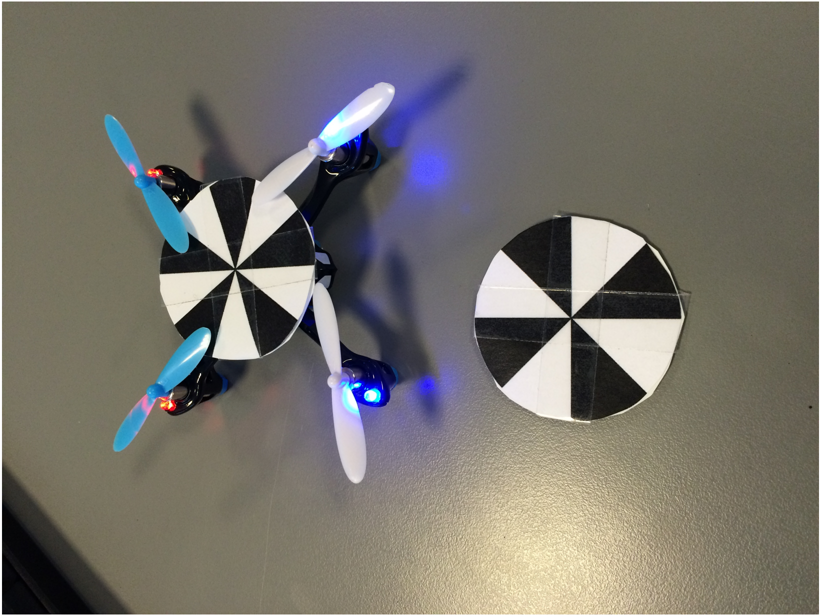
Absolute value of kernel



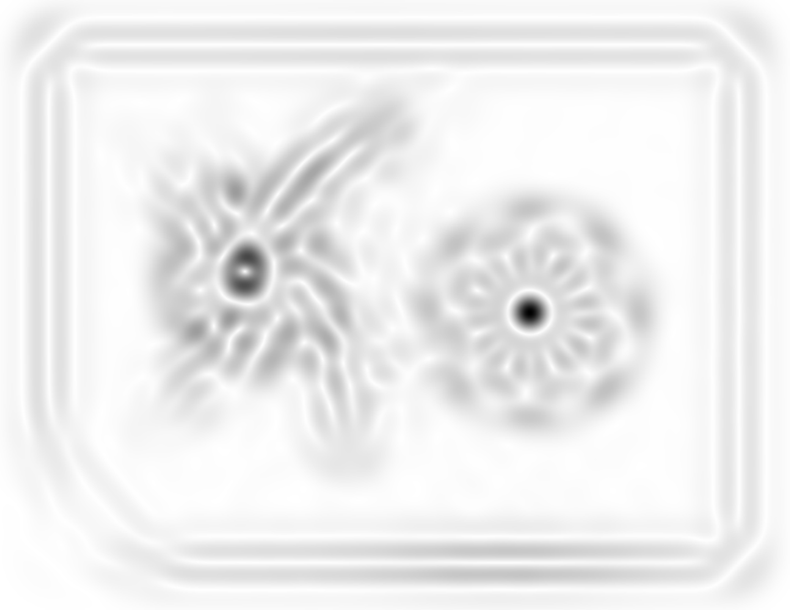
Argument of kernel



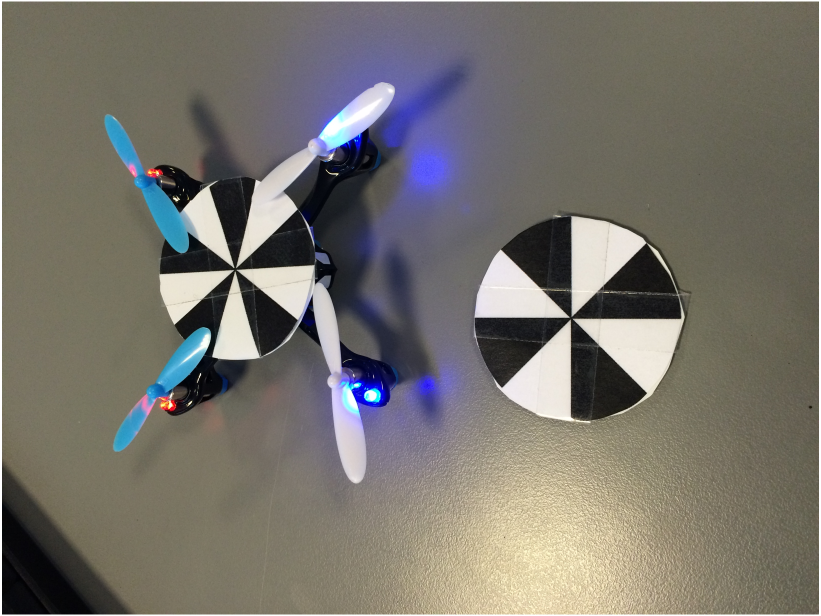
Input image



Magnitude response of convolution



Input image



Conclusion

Complex numbers can be used to detect repeating patterns in images.
The patterns might repeat over

- time
- orientation

Questions