Cvičení 2

Aritmetické operace

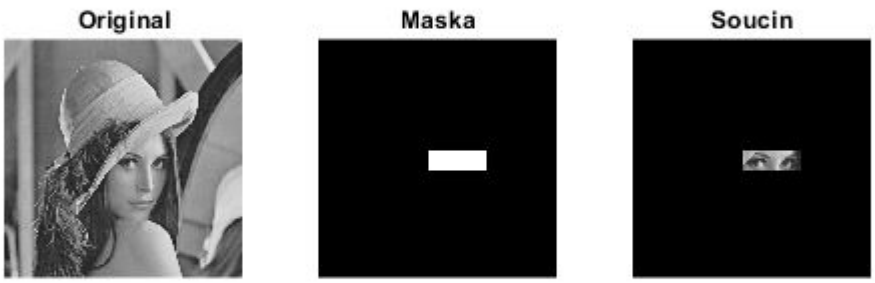
Jak už víme, obrázky jsou reprezentované maticí, kde každý prvek matice představuje buď intenzitu pixelu nebo index do nějaké palety. S obrázky tedy můžeme pracovat jako s maticemi pomocí aritmetických operací. Viz minulé cvičení.

Rozdíl

```
I = imread('lena_gray.png');
I2 = bitand(I,254);
d = I-I2;

figure
subplot(1,3,1)
imshow(I)
title('Obrazek 1')
subplot(1,3,2)
imshow(I2)
title('Obrazek 2')
subplot(1,3,3)
imshow(d,[])
title('Rozdil')
```

Součin

```
maska = uint8(zeros(size(I)));
maska(241:282,236:363) = 1;

I2 = maska.*I;
figure
subplot(1,3,1)
imshow(I)
title('Original')
subplot(1,3,2)
imshow(maska,[])
title('Maska')
subplot(1,3,3)
imshow(I2)
title('Soucin')
```



Logické operace and, or, xor, not

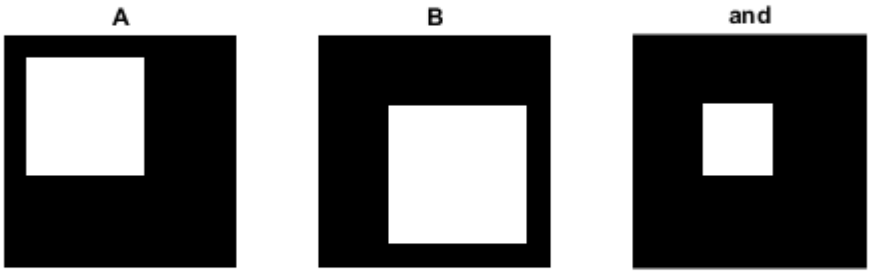
Kromě aritmetických operací můžeme používat i jiné operace známe z matematiky. Například černobílé obrázky můžeme chápat tak, že jednotlivé hodnoty pixelu jsou logické hodnoty. Jedničky většinou vymezují nějakou oblast. Dále jsou uvedeny logické operace nad dvěma obrázky.

```
% vytvoreni 2 obrazku
% false vytvori ctvercovou matici samych false hodnot.
A = false(500);
B = false(500);

% nastaveni nekterych pixelu na true
```

```
A(50:300, 50:300) = true;
B(150:450, 150:450) = true;
```

AND

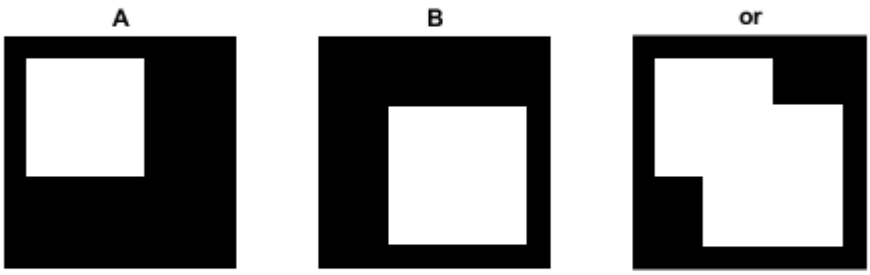
```
figure
subplot(1,3,1)
imshow(A)
title('A')
subplot(1,3,2)
imshow(B)
title('B')
subplot(1,3,3)
imshow(and(A,B))
title('and')
```



OR

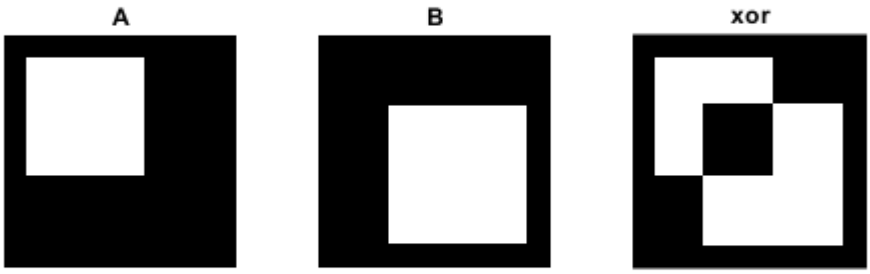
```
figure
subplot(1,3,1)
imshow(A)
title('A')
subplot(1,3,2)
imshow(B)
title('B')
subplot(1,3,3)
imshow(or(A,B))
```

```
title('or')
```



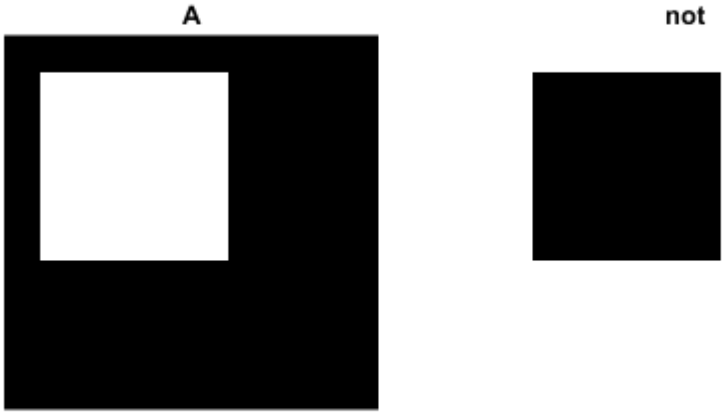
XOR

```
figure
subplot(1,3,1)
imshow(A)
title('A')
subplot(1,3,2)
imshow(B)
title('B')
subplot(1,3,3)
imshow(xor(A,B))
title('xor')
```



NOT

```
figure
subplot(1,2,1)
imshow(A)
title('A')
subplot(1,2,2)
imshow(not(A))
title('not')
```



Geometrické transformace

Velkou skupinou operací, které můžeme s obrázky provádět jsou operace geometrické. Jsou to různé transformace - otočení, posunutí, změna měřítka. Tyto operace nepracují přímo s hodnotami jednotlivých pixelů. Mění jejich polohu v obraze - transformují prostor vstupního obrazu (souřadnice pixelu (w,z)) na prostor výstupního (nové souřadnice pixelu jsou (x,y)). Konkrétní hodnoty pixelů se pak dopočítávají pomocí interpolace. Funkce pro převod souřadnic se nazývají geometrické transformace - transformační funkce. Z praktických důvodů se nepočítají nové souřadnice pro každý pixel vstupního obrazu, ale naopak pro každý pixel výstupního obrazu se počítá, kde by pixel ležel ve vstupním a z této informace se dopočítává hodnota pixelu. Místo transformační funkce se používá funkce k ní inverzní - zpětná transformační funkce. V matlabu se definuje tzv. transformační struktura (t-forma) a ta se aplikuje na obrázek.

```
tform = maketform(typ, parametry, ...)
```

<u>typ</u>: `'affine'`, `'projective'`, `'custom'`, `'box'`, `'composite'`

`'custom'` pro uživatelsky definované transformační funkce

```
tform = maketform('custom', ndim_in, ndim_out, dopredna_fce, zpetna_fce, data)
```

Příklad:

```
(x,y) = T{(w,z)} = (3w,2z)
(w,z) = T^(-1){(x,y)} = (x/3, y/2)
```

```
% dopredna_fce
dopredna_fce = @(wz,tdata) [3*wz(:,1),2*wz(:,2)]
```

```
dopredna_fce = function_handle with value:
    @(wz,tdata)[3*wz(:,1),2*wz(:,2)]
```

```
% wz predstavuje souradnice bodu, ktere se budou transformovat

% zpetna_fce
zpetna_fce = @(xy,tdata) [xy(:,1)/3,xy(:,2)/2]
```

```
zpetna_fce = function_handle with value:
    @(xy,tdata)[xy(:,1)/3,xy(:,2)/2]
```

```
% transformacni struktura
tform1 = maketform('custom',2,2,dopredna_fce, zpetna_fce,[])
```

```
tform1 = struct with fields:
       ndims_in: 2
      ndims_out: 2
    forward_fcn: @(wz,tdata)[3*wz(:,1),2*wz(:,2)]
    inverse_fcn: @(xy,tdata)[xy(:,1)/3,xy(:,2)/2]
          tdata: []
```

Aplikace transformace

XY = tformfwd(WZ,tform) WZ = tforminv(XY,tform)

```
% matice prestavuje dva body (1, 1) a (3, 2)
WZ = [1 1;
      3 2];

XY = tformfwd(WZ,tform1)
```

```
XY = 2×2
     3     2
     9     4
```

```
% pro kontrolu vratime body do puvodniho prostoru
WZ2 = tforminv(XY,tform1)
```

```
WZ2 = 2×2
     1     1
     3     2
```

ÚKOL 1

Vytvořte transformační strukturu `tform2` *pro funkce*

$$(x, y) = T\{(w, z)\} = (w + 0.4z, z)$$

$$(w, z) = T^{-1}\{(x, y)\} = (x - 0.4y, y)$$

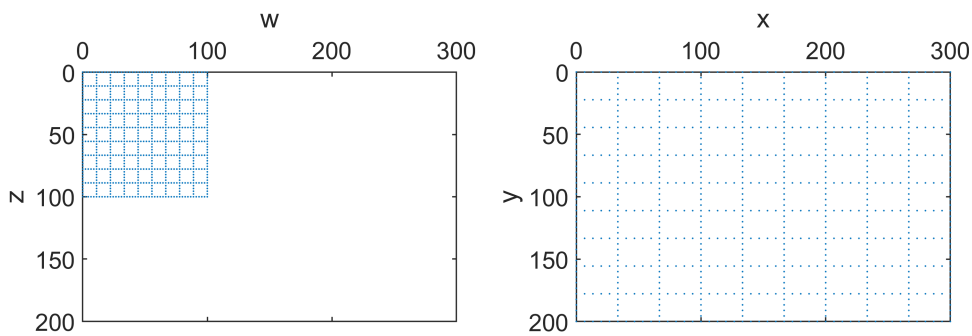
Vyzkoušejte na bodech WZ.

```
WZ = [1 1;
      3 2];
```

Vizualizace transformace

Používají se přiložené pomocné funkce. Jak fungují není podstatné.

```
vistform(tform1,pointgrid([0 0; 100 100]));
```




```
% figure, vistform(tform2,pointgrid([0 0; 100 100]));
```

Afinní transformace

Speciálním případem geometrických tranformací, jsou transformace afinní. Znáte je z dřivějšího studia - algebra, případně geometrie. Pokud ne, tak důležité je vědět, že jsou to tranformace, které se dají zapsat pomocí matice ve tvaru

$$\begin{bmatrix} a_{11} & a_{12} & 0 \\ a_{21} & a_{22} & 0 \\ a_{31} & a_{32} & 1 \end{bmatrix}$$

Případně transponovaný tvar. Nové souřadnice pro bod (w, z) se získají vynásobením vektorem [w, z, 0].

V matlabu se transformační struktury vytváří pomocí

```
tform = maketform('affine',T)
```

T představuje transformační matici.

Případně pomocí

`tform = affine2d(T)`

```
%afinni transformace odpovidajici tform1
T1 = [3 0 0;
      0 2 0;
      0 0 1];

tform3 = maketform('affine',T1);
WZ = [1 1; 3 2];
XY = tformfwd(WZ,tform3)
```

```
XY = 2×2
     3     2
     9     4
```

```
WZ2 = tforminv(XY,tform3)
```

```
WZ2 = 2×2
     1     1
     3     2
```

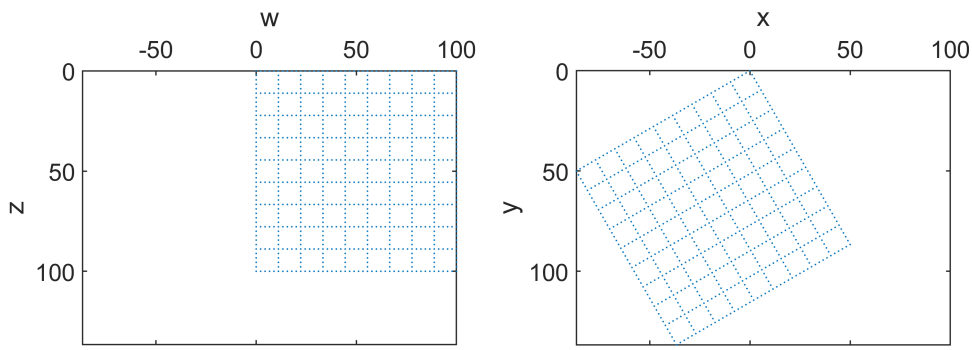
Úkol 2

Jak vypadá afinní matice odpovídající `tform2`.

```
% T2 = ..... doplnte
% tform4 = maketform('affine',T2);

% porovnejte s vysledky pro tform2
% WZ = [1 1; 3 2];
% XY = tformfwd(WZ,tform4)
% WZ2 = tforminv(XY,tform4)
```

Vizualizace otočení

```
uhel = pi/3;
T = [cos(uhel) sin(uhel) 0;
    -sin(uhel) cos(uhel) 0;
    0 0 1];
tform5 = maketform('affine',T);

vistform(tform5,pointgrid([0 0; 100 100]));
```



Aplikace transformace na obrázky

Zatím jsme transformaci aplikovali jen na jednotlivé body a počítali nové souřadnice bodu. My je ale chceme aplikovat na obrázky, tedy nejen počítat nové souřadnice bodu. K tomu slouží funkce `imtransform()` (případně `imwarp()`), která bere jako vstup obrazek `f` a transformační strukturu. Jak bylo řečeno, výsledné hodnoty pixelu se dopočítávají pomocí interpolace. Jakou interpolaci má matlab použít, můžeme zadat jako třetí parametr.

```
% Vice informaci:
% help imtransform

% g = imtransform(f,tform);
% nebo
% g = imwarp(f,tform);

f = imread('lena_gray.png');

g1 = imtransform(f,tform1,'bilinear');
%g2 = imtransform(f,tform2);
%g3 = imtransform(f,tform3);
%g4 = imtransform(f,tform4);
g5 = imtransform(f,tform5);

figure
imshow(g1);
axis on;
```


```
%figure, imshow(g2);
```

Interpolace

Zde je možné porovnat výsledky pro různé interpolační metody.

```
g5a = imtransform(f(300:500,300:500),tform5, 'nearest');
g5b = imtransform(f(300:500,300:500),tform5, 'bilinear');
g5c = imtransform(f(300:500,300:500),tform5, 'bicubic');

figure
subplot(1,3,1), imshow(g5a);
subplot(1,3,2), imshow(g5b);
subplot(1,3,3), imshow(g5c);
```

Matlab transformace

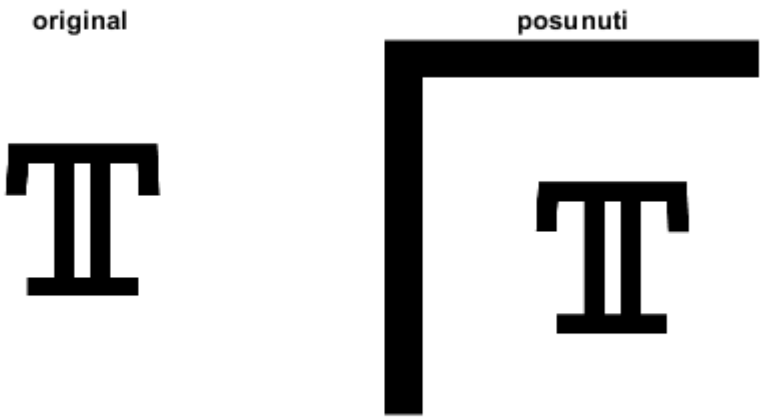
Některé geometrické tranformace jsou v matlabu implementovány přímo a není potřeba používat transformační struktury. Například je to posunutí, rotace, změna měřítka..

posun: `imtranslate()`

otočení: `imrotate()`

změna měřítka: `imresize()`

```
T = imread('t.png');
T2 = imtranslate(T, [50,50]);

figure
subplot(1,2,1)
imshow(T)
title('original')
subplot(1,2,2)
imshow(T2)
title('posunuti')
```




```
T2 = imrotate(T, 30);

figure
subplot(1,2,1)
imshow(T)
title('original')
subplot(1,2,2)
imshow(T2)
title('otoceni')
```

Registrace

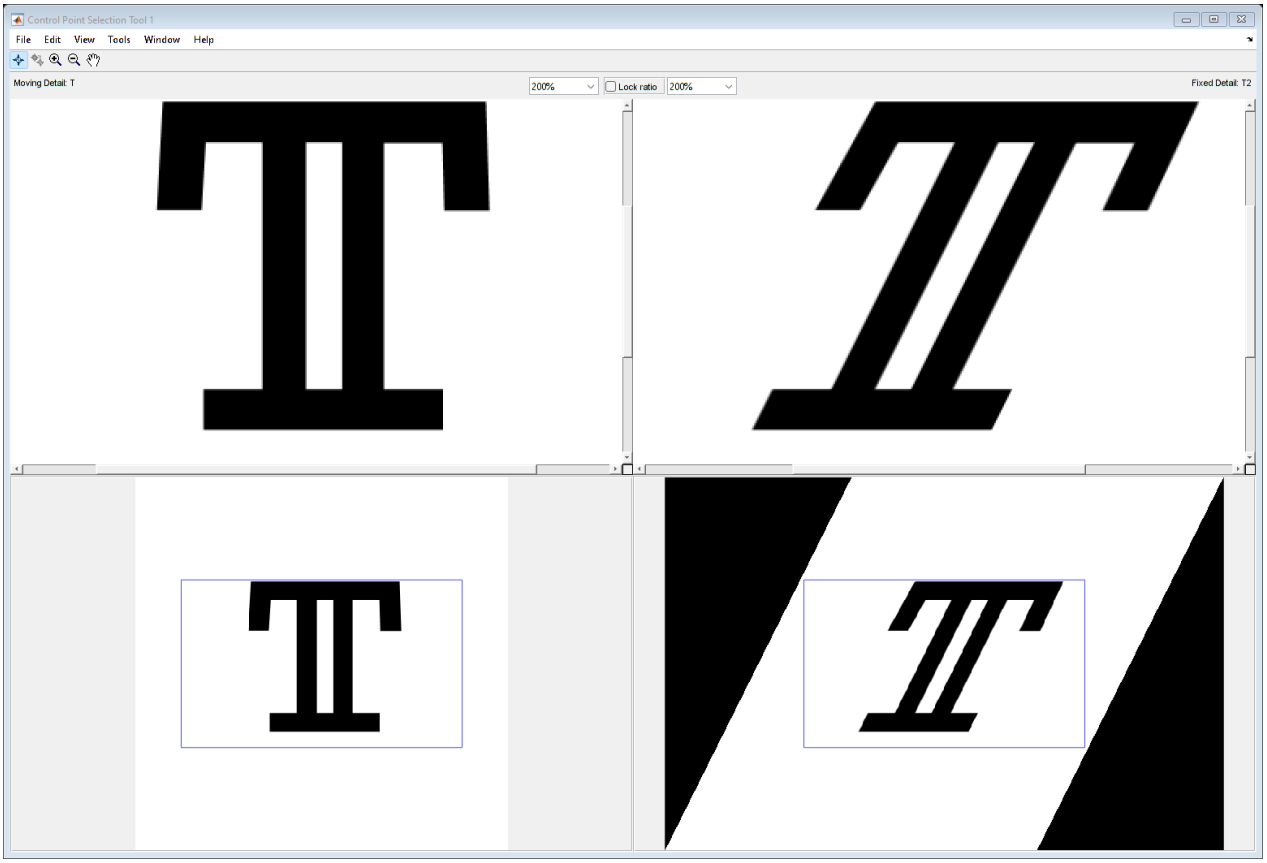
Při zpracování obrazu se geometrické transformace používají i v jiných případech. Máme například dva obrazy zachycující stejný objekt a naším úkolem je namapovat tyto dva obrázky na sebe. Jinými slovy, obrázky se od sebe liší jen nějakou transformací a naším úkolem je zjistit, jaká transformace to byla. Tu aplikovat na první obraz a tím získat obrazy, které si prostorově odpovídají. Prakticky se to dělá tak, že v obrazech najdeme body, které si odpovídají - kontrolní body (podle toho, jakou budeme hledat transformaci, tolik bodů najdeme) - a ze známých souřadnic se vypočítají koeficienty transformační funkce. Ta se poté aplikuje na všechny body. Tomuto postupu se říká Registrace.

```
% Nacteni obrazku a vytvoreni noveho pomoci afinni trnasformace
T = imread('t.png');
tform = affine2d([1 0 0;
                  -0.5 1 0;
                  0 0 1]);

T2 = imwarp(T,tform);

figure
subplot(1,2,1)
imshow(T)
subplot(1,2,2)
imshow(T2)
```

Výběr kontrolních bodů

Otevře se nové okno s nástrojem pro výběr kontrolních bodů. Jelikož hledáme afinní transformaci, stačí nám tři body, které neleží na jedné přímce (jsou to body v obecné poloze). Vždy se vybere bod v jednom obrázku pomocí myši a hned k němu odpovídající bod ve druhém obrázku. Po výběru kontrolních bodů - export points to workspace (názvy nechte, jak jsou navržené).

```
cpselect(T,T2);
```



Hledání transformace

Pomocí funkce `fitgeotrans()`.

```
tform2 = fitgeotrans(movingPoints,fixedPoints,'affine')
```

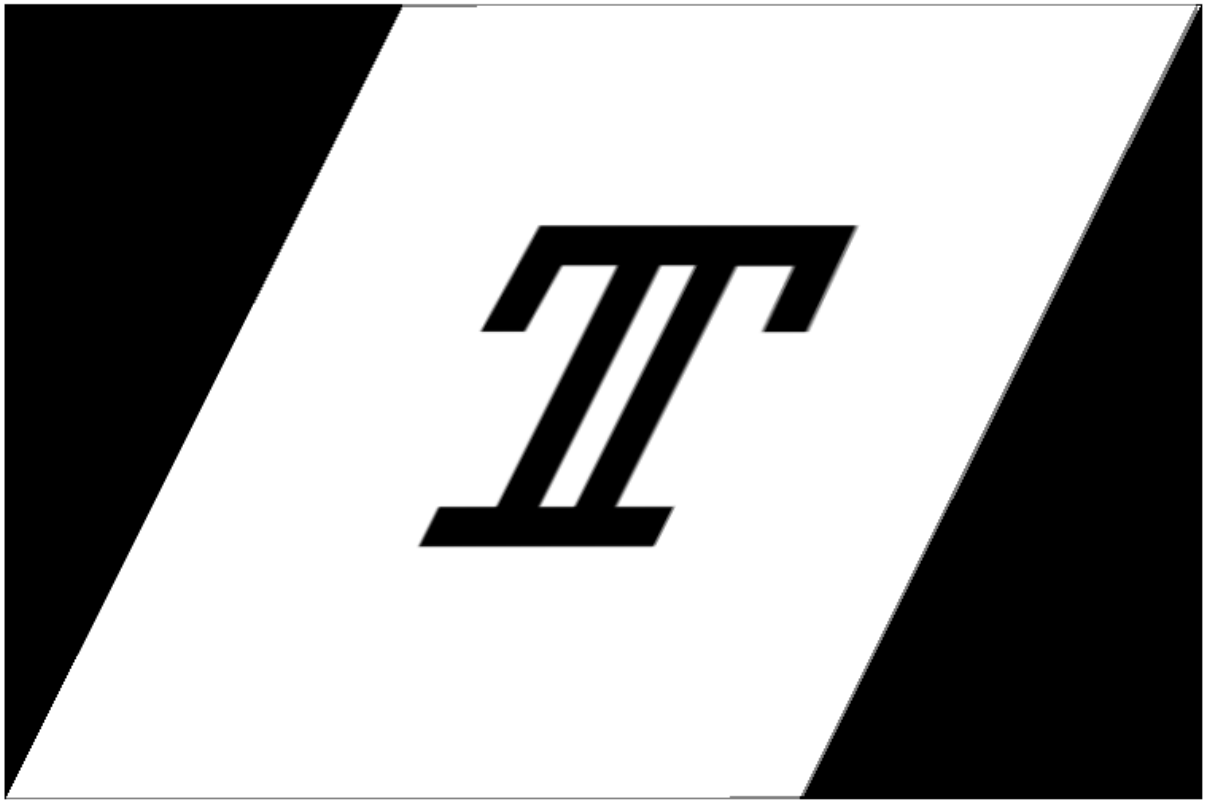
```
tform2 = 
  affine2d with properties:

                T: [3×3 double]
    Dimensionality: 2
```

Aplikace nalezené transformace

```
T3 = imwarp(T,tform2);
figure
subplot(1,2,1)
imshow(T2)
subplot(1,2,2)
imshow(T3)
```

Zobrazení obou obrázků v jednom.

```
figure, imshowpair(T2,T3,'blend');
```



Úkol 3

Naprogramujte funkci registrace (stačí popsat postup, jakým byste postupovali), která jako vstup bere dva obrázky, které obsahují černý obdélník (`obdelnik1.png` a `obdelnik2.png`). Předpokládejme, že obdélníky na obou obrázcích jsou ty samé, jen druhý obdelník je posunutý a deformovaný operací zvětšení/zmenšení v osách x a y. Funkce vrací 3 hodnoty – vektor posunutí a 2 koeficienty představující míru zvětšení/zmenšení v ose x a v ose y.

```
% Pro nasledujici matice by byl vystup posunuti = [2,1], zvetseni_x = 1.5,
% zvetseni_y = 2

M1 = [ 1 1 1 1 1 1;
       1 0 0 1 1 1;
       1 0 0 1 1 1;
       1 1 1 1 1 1;
       1 1 1 1 1 1;
       1 1 1 1 1 1];

M2 = [ 1 1 1 1 1 1;
       1 1 1 1 1 1;
       1 1 1 0 0 0;
       1 1 1 0 0 0;
       1 1 1 0 0 0;
```

```
        1 1 1 0 0 0];
```