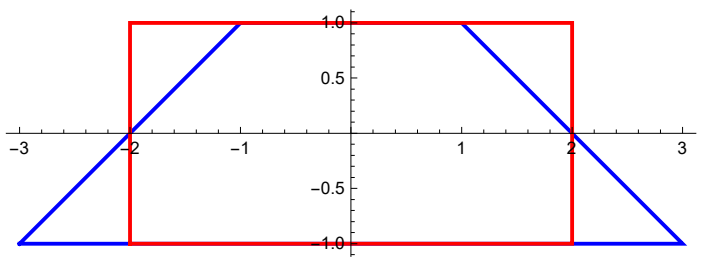
Tražimo projektivno preslikavanje koje slika trapez u pravougaonik. Ovde možete da vidite primer sa časa sa svim međurezultatima (za testiranje i traženje grešaka).

Ulazni podaci: koordinate 4 tacke (a,b,c,d) i koordinate njihovih slika (ap,bp,cp,dp).

Prvo ide naivni algoritam koji daje matricu projektivnog preslikavanja koje a,b,c,d slika u ap, bp, cp, dp.

In[10]:=
```
a = {-3, -1, 1};
b = {3, -1, 1};
c = {1, 1, 1};
d = {-1, 1, 1};
ap = {-2, -1, 1};
bp = {2, -1, 1};
cp = {2, 1, 1};
dp = {-2, 1, 1};
original = ShowPolygon[{a, b, c, d, a}, Blue];
slika = ShowPolygon[{ap, bp, cp, dp, ap}, Red];
Show[original, slika, Axes → True]
```

Out[20]=



Naivni algoritam - zadatae su tačke a, b, c, d i ap, bp, cp, dp. Odrediti matricu projektivnog preslikavanja koje slika jedne u druge.

In[21]:=
```
p = 3 Projective4pts[{a, b, c, d}, {ap, bp, cp, dp}];
p // MatrixForm
```

Out[22]//MatrixForm=

$$\begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & -1 \\ 0 & -1 & 2 \end{pmatrix}$$

p je tražena matrica preslikavanja dobijena naivnim algoritmom

DLT algoritam na 6 tačaka

In[23]:=
```
e = {1, 2, 3};
f = {-8, -2, 1};
ep = p.e
fp = p.f
```

Out[25]= {2, 1, 4}

Out[26]= {-16, -5, 4}

Za proveru DLP algoritma dodajemo još dve tačke: e i f, i njihove slike ep i fp.
Sada je ulaz {a,b,c,d,e,f} i {ap,bp,cp,dp,ep,fp}, a izlaz je opet matrica projektivnog preslikavanja

In[27]:=
```
dlp = ProjectiveDLP[{a, b, c, d, e, f}, {ap, bp, cp, dp, ep, fp}];
dlp // MatrixForm
pdlp = Round[dlp, 10^(-5)] // N;
pdlp // MatrixForm
pdlp / pdlp[[1]][[1]] * p[[1]][[1]] // MatrixForm
```

Out[28]//MatrixForm=

$$\begin{pmatrix} 0.534522 & 8.72795 \times 10^{-16} & -8.25587 \times 10^{-16} \\ 1.88689 \times 10^{-16} & 0.534522 & -0.267261 \\ -1.45181 \times 10^{-16} & -0.267261 & 0.534522 \end{pmatrix}$$

Matrica preslikavanja dobijena DLP algoritmom

Out[30]//MatrixForm=

$$\begin{pmatrix} 0.53452 & 0. & 0. \\ 0. & 0.53452 & -0.26726 \\ 0. & -0.26726 & 0.53452 \end{pmatrix}$$

Kada zaokružimo na npr. 5 decimala, dobijemo da matrica preslikavanja pdlp izgleda ovako

Out[31]//MatrixForm=

$$\begin{pmatrix} 2. & 0. & 0. \\ 0. & 2. & -1. \\ 0. & -1. & 2. \end{pmatrix}$$

Posto smo slike tačaka e i f pronašli množenjem matricom p, trebalo bi da se rezultat pri naivnom DLT algoritmu poklapaju, tj. da su matrice p i pdlt proporcionalne (Proporcionalne matrice zadaju isto projektivno preslikavanje).

Matrice poredimo tako što podelimo pdlt matricu sa pdlt[1,1], zatim pomožimo sa p[1,1]. Vidimo da se posle ovoga matrice poklapaju. (Isto može da se uradi i za bilo koji drugi nenula element matrice)

```
(*Međurezultati za DLT algoritam*)
orig = {a, b, c, d, e, f};
images = {ap, bp, cp, dp, ep, fp};
TwoEquations[a, ap] // MatrixForm
A = Flatten[MapThread[TwoEquations, {orig, images}], 1];
A // MatrixForm
SingularValueDecomposition[A][[1]] // MatrixForm
SingularValueDecomposition[A][[2]] // MatrixForm
SingularValueDecomposition[A][[3]] // MatrixForm
v = SingularValueDecomposition[A][[3]];
last = (Transpose[v])[[9]]
Partition[last, 3] // MatrixForm
```

Hajde da vidimo šta su međurezultati u dlt algoritmu:
Ulaz su tačke a,b,c,d,e,f i njihove slike ap, bp, cp, dp, ep, fp

Out[81]//MatrixForm=

$$\begin{pmatrix} 0. & 0. & 0. & 3. & 1. & -1. & 3. & 1. & -1. \\ -3. & -1. & 1. & 0. & 0. & 0. & -6. & -2. & 2. \end{pmatrix}$$

Svaka tačka i njena slika zadaju dve vrste matrice A. Npr. ove dve vrste se dobiju za tačke a i ap.

Out[83]//MatrixForm=

$$\begin{pmatrix} 0. & 0. & 0. & 3. & 1. & -1. & 3. & 1. & -1. \\ -3. & -1. & 1. & 0. & 0. & 0. & -6. & -2. & 2. \\ 0. & 0. & 0. & -3. & 1. & -1. & -3. & 1. & -1. \\ 3. & -1. & 1. & 0. & 0. & 0. & -6. & 2. & -2. \\ 0. & 0. & 0. & -1. & -1. & -1. & 1. & 1. & 1. \\ 1. & 1. & 1. & 0. & 0. & 0. & -2. & -2. & -2. \\ 0. & 0. & 0. & 1. & -1. & -1. & -1. & 1. & 1. \\ -1. & 1. & 1. & 0. & 0. & 0. & -2. & 2. & 2. \\ 0. & 0. & 0. & -4. & -8. & -12. & 1. & 2. & 3. \\ 4. & 8. & 12. & 0. & 0. & 0. & -2. & -4. & -6. \\ 0. & 0. & 0. & 32. & 8. & -4. & 40. & 10. & -5. \\ -32. & -8. & 4. & 0. & 0. & 0. & -128. & -32. & 16. \end{pmatrix}$$

Evo kako izgleda cela matrica. Po dve vrste redom odgovaraju tačkama a i ap, b i bp, ... , f i fp

SVD dekompozicija razbija A na prozvod ove tri matrice:

Out[84]//MatrixForm=

$$\begin{pmatrix} -0.0236091 & -0.0905041 & 0.00633263 & -0.0411228 & -0.0938672 & -0.0815149 & 0.210577 & -0.0629792 & 0.629243 & -0.0806813 & -0.283794 & 0.665403 \\ 0.0490722 & -0.0285939 & -0.0342383 & -0.0129952 & 0.240159 & 0.195615 & 0.0452732 & 0.802403 & -0.279529 & -0.186835 & -0.293389 & 0.23339 \\ 0.0183797 & 0.0723046 & 0.00441422 & -0.0121826 & -0.288282 & 0.0637628 & 0.888905 & -0.0148142 & -0.237255 & -0.13719 & 0.203678 & -0.0112786 \\ 0.030428 & 0.0233452 & 0.102578 & -0.0214468 & -0.903143 & 0.00116443 & -0.254303 & 0.29686 & 0.0151368 & 0.0854509 & -0.0898353 & -0.0590013 \\ -0.00682899 & 0.0331671 & -0.0735986 & -0.0813739 & 0.02259 & 0.210174 & -0.0361316 & 0.086815 & -0.142042 & 0.705372 & 0.450917 & 0.460692 \\ 0.0130418 & 0.00995811 & 0.158915 & -0.036659 & -0.0452828 & -0.439248 & -0.0185948 & -0.302469 & -0.600195 & 0.152266 & -0.457408 & 0.306054 \\ 0.00529871 & -0.0208977 & -0.0577835 & -0.0951599 & -0.120387 & 0.206814 & -0.307483 & -0.22376 & -0.267488 & -0.629965 & 0.380107 & 0.41518 \\ 0.012733 & -0.00609417 & -0.025178 & -0.0478335 & -0.0567632 & 0.806833 & 0.0189168 & -0.333799 & -0.0757228 & 0.111794 & -0.457276 & -0.0597051 \\ -0.0053116 & 0.149067 & -0.406748 & -0.885272 & -0.00492791 & -0.107164 & 0.00708224 & 0.0124495 & 0.0261438 & -0.000260945 & -0.0754049 & -0.102554 \\ 0.00817 & 0.0455855 & 0.887531 & -0.414004 & 0.113677 & 0.0904982 & 0.0101088 & 0.0395966 & 0.0716295 & -0.0215873 & 0.0977274 & -0.0285449 \\ -0.297685 & -0.934172 & -0.0122958 & -0.145192 & -0.0378601 & -0.00626809 & 0.0446518 & 0.0108999 & -0.074046 & 0.0363989 & 0.0384876 & -0.0748359 \\ 0.952183 & -0.294226 & -0.0172162 & -0.0451196 & 0.00909316 & -0.021121 & 0.00922236 & -0.0385034 & 0.0201889 & 0.0240931 & 0.031295 & -0.0195498 \end{pmatrix}$$

Out[85]//MatrixForm=

$$\begin{pmatrix} 143.493 & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 33.8496 & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 17.4673 & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 14.3459 & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 6.20127 & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 4.03816 & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 2.69879 & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 1.38249 & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \end{pmatrix}$$

Out[86]//MatrixForm=

$$\begin{pmatrix} -0.212503 & 0.288613 & 0.268822 & -0.0157794 & -0.524843 & -0.196022 & -0.441281 & -0.0685748 & 0.534522 \\ -0.0530048 & 0.0805801 & 0.418117 & -0.209197 & 0.225375 & 0.263427 & 0.0802004 & -0.803425 & 8.72795 \times 10^{-16} \\ 0.0279598 & -0.0186491 & 0.617359 & -0.367174 & 0.102474 & 0.387766 & -0.0187171 & 0.567193 & -8.25587 \times 10^{-16} \\ -0.067031 & -0.916769 & 0.071854 & -0.0840413 & -0.121192 & -0.0522799 & -0.335635 & -0.112891 & 1.88689 \times 10^{-16} \\ -0.0163261 & -0.256912 & 0.188795 & 0.421295 & -0.0883383 & 0.0922277 & 0.646087 & 0.0338189 & 0.534522 \\ 0.00878952 & 0.0577205 & 0.289157 & 0.797011 & 0.111352 & 0.225798 & -0.377748 & 0.0157281 & -0.267261 \\ -0.937152 & -0.00173267 & -0.0662691 & 0.0148457 & 0.322129 & -0.0454958 & 0.0305244 & 0.101793 & -1.45181 \times 10^{-16} \\ -0.233702 & 0.0075377 & -0.237636 & -0.0273046 & -0.632718 & 0.628842 & 0.132233 & -0.0584904 & -0.267261 \\ 0.116373 & 0.000928152 & -0.431856 & -0.0206623 & 0.352498 & 0.531115 & -0.327564 & 0.0133748 & 0.534522 \end{pmatrix}$$

Nas zanima samo treća matrica iz SVD dekompozicije

Out[88]= $\{0.534522, 8.72795 \times 10^{-16}, -8.25587 \times 10^{-16}, 1.88689 \times 10^{-16}, 0.534522, -0.267261, -1.45181 \times 10^{-16}, -0.267261, 0.534522\}$

Tačnije samo njena poslednja kolona

Out[89]//MatrixForm=

$$\begin{pmatrix} 0.534522 & 8.72795 \times 10^{-16} & -8.25587 \times 10^{-16} \\ 1.88689 \times 10^{-16} & 0.534522 & -0.267261 \\ -1.45181 \times 10^{-16} & -0.267261 & 0.534522 \end{pmatrix}$$

Kada napravimo matricu od prethodnog vektora (kolone), to je matrica traženog ptojektivnog preslikavanja (poklapa se sa rezultatom na prethodnoj strani pre zaokruživanja)

Važna napomena:
Proverite, ovde može da se desi da je rešenje poslednja vrsta, a ne poslednja kolona u zavisnosti od programa u kome radite i od paketa koji koristite za SVD dekompoziciju. U Wolframu je poslednja kolona, a ako koristite Python i biblioteku numpy, rešenje je poslednja vrsta.

```
In[90]:= ndlp = ProjectiveDLPNorm[orig, images];
         ndlp // MatrixForm
         pndlp = Round[ndlp, 10^-5] // N;
         pndlp // MatrixForm
         pndlp / pndlp[[1]][[1]] * p[[1]][[1]] // MatrixForm
```

Proverimo sada modifikovani dlp algoritam. Ulaz je isti kao malopre, izlaz je matrica pndlp

Out[91]//MatrixForm=

$$\begin{pmatrix} 0.397003 & 4.24364\times10^{-16} & -5.55112\times10^{-16} \\ 3.44504\times10^{-16} & 0.397003 & -0.198501 \\ -1.16872\times10^{-16} & -0.198501 & 0.397003 \end{pmatrix}$$

Out[93]//MatrixForm=

$$\begin{pmatrix} 0.397 & 0. & 0. \\ 0. & 0.397 & -0.1985 \\ 0. & -0.1985 & 0.397 \end{pmatrix}$$

Zaokružimo opet na npr. 5 decimala

Out[94]//MatrixForm=

$$\begin{pmatrix} 2. & 0. & 0. \\ 0. & 2. & -1. \\ 0. & -1. & 2. \end{pmatrix}$$

Uporedimo opet sa matricom p tako što podelimo celu matricu pndlp elementom na mestu [1,1] i pomnožimo celu matricu sa p[1,1]. Vidmo da opet dobijamo isti rezultat.

Međurezultati za modifikovani DLT algoritam

Hajde da vidimo neke međurezultate u modifikovanom algoritmu:

```
(*Matrice koje normalizuju koordinate originala i slika*)
T = normMatrix[orig] // N;
T // MatrixForm
Tp = normMatrix[images] // N;
Tp // MatrixForm
noriginali = (T.#) & /@ orig
nslike = (Tp.#) & /@ images
m = ProjectiveDLP[noriginali, nslike];
m // MatrixForm
Inverse[Tp].m.T // MatrixForm // MatrixForm
```

Out[97]//MatrixForm=

$$\begin{pmatrix} 0.449686 & 0. & 0.574598 \\ 0. & 0.449686 & 0.0999301 \\ 0. & 0. & 1. \end{pmatrix}$$

Matrica koja normalizuje originale (a,b,c,d,e,f), tako što translira koordinatni početak u težište ovog skupa tačaka, zatim skalira tako da prosečno rastojanje tačaka od koordinatnog početka bude koren iz 2.

Out[99]//MatrixForm=

$$\begin{pmatrix} 0.61609 & 0. & 0.359386 \\ 0. & 0.61609 & 0.102682 \\ 0. & 0. & 1. \end{pmatrix}$$

Matrica koja normalizuje slike (ap,bp,cp,dp,ep,fp) na isti način.

Out[100]= {{-0.774459, -0.349755, 1.}, {1.92366, -0.349755, 1.}, {1.02428, 0.549616, 1.}, {0.124913, 0.549616, 1.}, {2.17348, 1.19916, 3.}, {-3.02289, -0.799441, 1.}}

Normalizovane koordinate originala (a,b,c,d,e,f) u novom koordinatnom sistemu

Out[101]= {{-0.872794, -0.513408, 1.}, {1.59157, -0.513408, 1.}, {1.59157, 0.718772, 1.}, {-0.872794, 0.718772, 1.}, {2.66972, 1.02682, 4.}, {-8.4199, -2.66972, 4.}}

Normalizovane koordinate slika (ap,bp,cp,dp,ep,fp) u novom koordinatnom sistemu (Ova dva koordinatna sistema se razlikuju)

Out[103]//MatrixForm=

$$\begin{pmatrix} 0.543912 & -0.158641 & -0.154001 \\ 4.17905\times10^{-16} & 0.498586 & -0.131354 \\ -1.87629\times10^{-16} & -0.441422 & 0.441114 \end{pmatrix}$$

Matrica preslikavanja dobijena dlt algoritmom za normalizovane koodinate tačaka

Out[104]//MatrixForm=

$$\begin{pmatrix} 0.397003 & 8.23765\times10^{-16} & -3.33067\times10^{-16} \\ 3.19092\times10^{-16} & 0.397003 & -0.198501 \\ -8.4374\times10^{-17} & -0.198501 & 0.397003 \end{pmatrix}$$

Vratimo nazad u polazni koordinatni sistem, dobijamo matricu preslikavanja u polaznom koordinatnom sistemu