INF 2064 - Visão Computacional e Realidade Aumentada
Prova Final de 2008.2

Aluno(a):_____matrícula:_____

Para fazer a prova favor observar o seguinte:

1. A prova é individual.
2. A prova é para ser feita em casa e com consulta livros, manuais e material de referência na internet.
3. A prova é entregue por e-mail, através de um arquivo Word e sua resposta também deve ser digital (arquivo Word ou pdf) entregue também por e-mail.
4. Durante o período de provas os alunos devem se abster de conversar sobre ela, qualquer que seja o laço pessoal entre eles.
5. Mantenha sua solução em segredo até a data de entrega.
6. A prova deverá ser entregue na quinta-feira, dia 27 de novembro, e deverá ser devolvida (estar na minha caixa postal) antes das 7h da 4ª feira, dia 3 de dezembro.
7. Após esta hora as provas não serão mais aceitas.

1. A relação entre as coordenadas (X,Y,Z) de um ponto em 3D e as coordenadas de seu *pixel* (u,v) pode ser descrita por uma matriz de projeção de câmera **P.**
   a. Pode a matriz de câmera incorporar as distorções de lente? Explique brevemente.

      Não. A distorção radial é uma deformação não-linear. A matriz de projeção é linear no espaço projetivo.

   b. Liste os parâmetros intrísicos (internos) e extrinsicos(externos) de um modelo de câmera.

      No modelo *pin-hole* podemos ter os seguintes parâmetros internos:

      f_x = distância focal medida em tamanho do *pixel* em x
      f_y = distância focal medida em tamanho do *pixel* em y
      γ = distorção dos eixos xy da câmera
      (o_x,o_y) = centro óptico da câmera na imagem
      k_1 e k_2 = coeficientes de distorção radial

      e os seguintes parâmetros externos:

      (t_x, t_y, t_z), 3 gl que definem a posição do centro de projeção da câmera
      3 gl que definem a rotação da câmera em relação aos eixos do mundo.

   c. Mostre como a matriz **P** pode ser escrita em termos do produto de duas matrizes, uma que contenha só os parâmetros intrínsicos e outra que contenha os parâmetros extrínsicos.

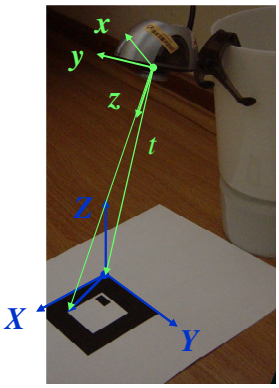
      Resposta:
      $\mathbf{p} = \mathbf{K}[\mathbf{R} \quad \mathbf{t}]\mathbf{P}$

$$\begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} -f_x & \gamma & o_x \\ 0 & -f_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \left[\begin{bmatrix} (\mathbf{i}_w \cdot \mathbf{i}_c) & (\mathbf{j}_w \cdot \mathbf{i}_c) & (\mathbf{k}_w \cdot \mathbf{i}_c) \\ (\mathbf{i}_w \cdot \mathbf{j}_c) & (\mathbf{j}_w \cdot \mathbf{j}_c) & (\mathbf{k}_w \cdot \mathbf{j}_c) \\ (\mathbf{i}_w \cdot \mathbf{k}_c) & (\mathbf{j}_w \cdot \mathbf{k}_c) & (\mathbf{k}_w \cdot \mathbf{k}_c) \end{bmatrix} \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix} \right] \begin{bmatrix} \begin{pmatrix} X_w \\ Y_w \\ Z_w \end{pmatrix} \\ 1 \end{bmatrix}$$

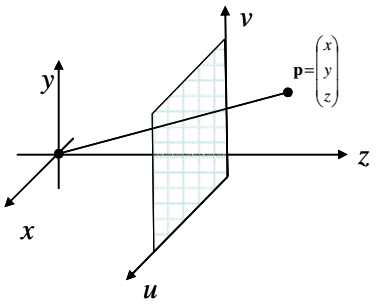
      os vetores $\mathbf{i}_c, \mathbf{j}_c, \mathbf{k}_c$ são os unitários do sistema de eixos da câmera, definidos através da rotação da câmera em relação aos eixos do mundo.

2. Uma biblioteca de calibração fornece as matrizes K3x3 e [R t]3x4. Determine, a partir delas as matrizes LookAt e Projection do OpenGL. Suponha que o sistema de calibração de câmera se baseie nos eixos os mostrados abaixo.

Visão



$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{bmatrix} \begin{bmatrix} r_0 & r_1 & r_2 \\ r_3 & r_4 & r_5 \\ r_6 & r_7 & r_8 \end{bmatrix} & \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix} \end{bmatrix} \begin{bmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \\ 1 \end{bmatrix}$$

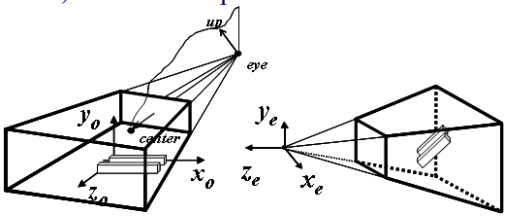


$$\begin{pmatrix} su \\ sv \\ s \end{pmatrix} = \begin{bmatrix} k_0 & k_1 & k_2 \\ 0 & k_4 & k_5 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

Resposta:

a) Obtendo a LookAt

A matrix LookAt do OpenGL se baseia na transformação entre o sistema dos objetos (*world coord*) e os do olho que são ambos sistemas destros (seguem a "sistemas da mão direita").



No caso proposto o sistema global também é destro e se ambos tiverem na mesma posição a relação entre eles é trivial:

$$\begin{bmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \\ 1 \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix}$$

O sistema da câmera é sinistro e tem o eixo z invertido em relação ao que seria um sistema destro. Com isto a relação ente os sistemas do **eye** (OpenGL) e o da Câmera pode ser escrita como:

$$\begin{bmatrix} x_e \\ y_e \\ z_e \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

A matriz de parâmetros extrínsecos pode ser re-escrita acrescentando-se uma linha inócua (1=1):

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} r_0 & r_1 & r_2 \\ r_3 & r_4 & r_5 \\ r_6 & r_7 & r_8 \end{bmatrix} & \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \\ 1 \end{bmatrix}$$

Substituindo as coordenadas do sistema de calibração pelas do OpenGL temos:

$$\begin{bmatrix} x_e \\ y_e \\ z_e \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \begin{bmatrix} r_0 & r_1 & r_2 \\ r_3 & r_4 & r_5 \\ r_6 & r_7 & r_8 \end{bmatrix} & \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \\ 1 \end{bmatrix}$$

ou:

$$\begin{bmatrix} x_e \\ y_e \\ z_e \\ 1 \end{bmatrix} = \begin{bmatrix} r_0 & r_1 & r_2 & t_x \\ r_3 & r_4 & r_5 & t_y \\ -r_6 & -r_7 & -r_8 & -t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix}$$

Esta matriz 4×4 é a matriz LookAt do OpenGL.

Note que, se o sistema do mundo fosse também sinistro teríamos:

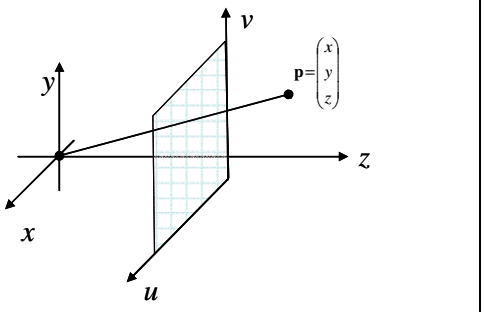
$$\begin{bmatrix} x_e \\ y_e \\ z_e \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \begin{bmatrix} r_0 & r_1 & r_2 \\ r_3 & r_4 & r_5 \\ r_6 & r_7 & r_8 \end{bmatrix} & \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \\ 1 \end{bmatrix}$$

ou:

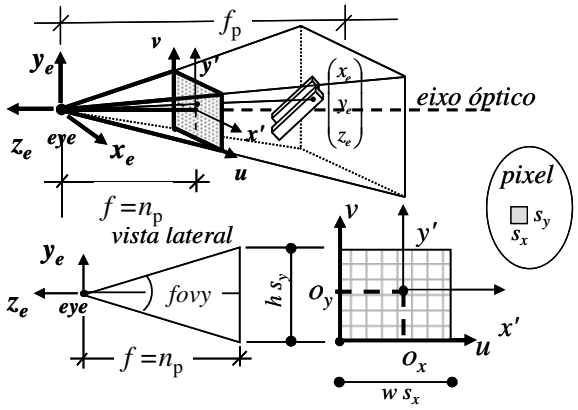
$$\begin{bmatrix} x_e \\ y_e \\ z_e \\ 1 \end{bmatrix} = \begin{bmatrix} r_0 & r_1 & -r_2 & t_x \\ r_3 & r_4 & -r_5 & t_y \\ -r_6 & -r_7 & +r_8 & -t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix}$$

Esta relação ocorre com freqüência em sistemas multi-câmera quando os sistemas delas são sinistros e uma das câmeras assume o papel de sistema global.

b) Matriz de Projeção (intrínsica)

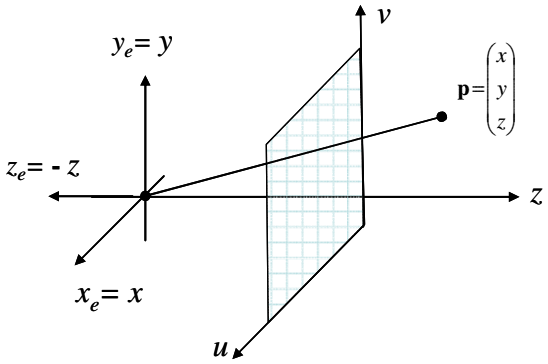


$$\begin{pmatrix} su \\ sv \\ s \end{pmatrix} = \begin{bmatrix} k_0 & k_1 & k_2 \\ 0 & k_4 & k_5 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$



$$\begin{bmatrix} wx_n \\ wy_n \\ wz_n \\ w \end{bmatrix} = \left[\mathbf{P}_{ogl}\right] \begin{bmatrix} x_e \\ y_e \\ z_e \\ 1 \end{bmatrix}$$

Primeiramente relacionar o sistema de eixos do ***eye*** do OpenGL com o sistema sinistro da câmera.



$$\begin{pmatrix} su \\ sv \\ s \end{pmatrix} = \begin{bmatrix} k_0 & k_1 & k_2 \\ 0 & k_4 & k_5 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{bmatrix} k_0 & k_1 & k_2 \\ 0 & k_4 & k_5 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{pmatrix} x_e \\ y_e \\ z_e \end{pmatrix}$$

ou

$$\begin{pmatrix} su \\ sv \\ s \end{pmatrix} = \begin{bmatrix} k_0 & k_1 & -k_2 \\ 0 & k_4 & -k_5 \\ 0 & 0 & -1 \end{bmatrix} \begin{pmatrix} x_e \\ y_e \\ z_e \end{pmatrix}$$

No lado esquerdo das equações de projeção a dificuldade é um pouco maior.
Primeiramente no sistema da câmera temos *u* e *v* que variam no intervalo: [0,*w-1*]×[0,*h-1*] enquanto no OpenGL a matriz de projeção está baseada em coordenadas de normalizadas, de acordo com:

$$\begin{bmatrix} wx_n \\ wy_n \\ wz_n \\ w \end{bmatrix} = \left[\mathbf{P}_{ogl}\right] \begin{bmatrix} x_e \\ y_e \\ z_e \\ 1 \end{bmatrix} = \begin{bmatrix} p_0 & p_4 & p_8 & p_{12} \\ p_1 & p_5 & p_9 & p_{13} \\ p_2 & p_6 & p_{10} & p_{14} \\ p_3 & p_7 & p_{11} & p_{15} \end{bmatrix} \begin{bmatrix} x_e \\ y_e \\ z_e \\ 1 \end{bmatrix}$$

que variam de [-1,1] ×[-1,1] ×[-1,1]. Note também que no OpenGL a linearização da matriz é por colunas e não linhas.

A relação entre as coordenadas normalizadas e (u,v) é a seguinte:

$$x_n = \frac{u}{2w} - 1 \text{ e } y_n = \frac{v}{2h} - 1$$

A profundidade não é computada no processo de calibração, e a relação entre as coordenadas normalizadas e (u,v) tem que ser acrescida da identidade ($z_n = z_n$). Ou seja:

$$\begin{bmatrix} wx_n \\ wy_n \\ wz_n \\ w \end{bmatrix} = \begin{bmatrix} \frac{2}{w} & 0 & 0 & -1 \\ 0 & \frac{2}{h} & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} su \\ sv \\ sz_n \\ s \end{bmatrix}$$

Como o *pipeline* de *rendering* do OpenGL se baseia no algortimo de *ZBuffer* temos que fornecer na matriz de projeção uma relação entre as profundidades normalizadas e no sistema do ***eye***. Esta relação pode ser definida a partir das distâncias dos planos *near* e *far*, n_p e f_p, respectivamente por por:

$$z_c = -\frac{f_p + n_p}{f_p - n_p} z_e - \frac{2 f_p n_p}{f_p - n_p}, \; w = -z_e \text{ e } z_n = \frac{z_c}{w}$$

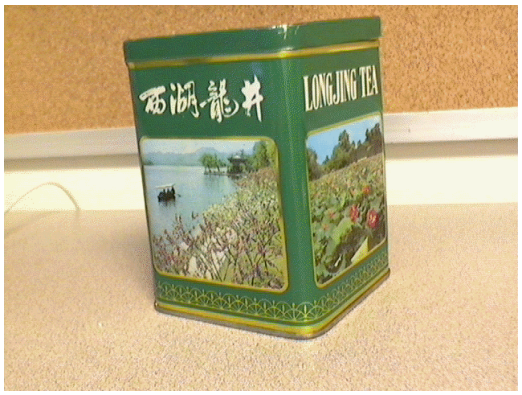
como mostramos na derivação do algoritmo de *rendering* do OpenGL.
Tudo isto resulta em:

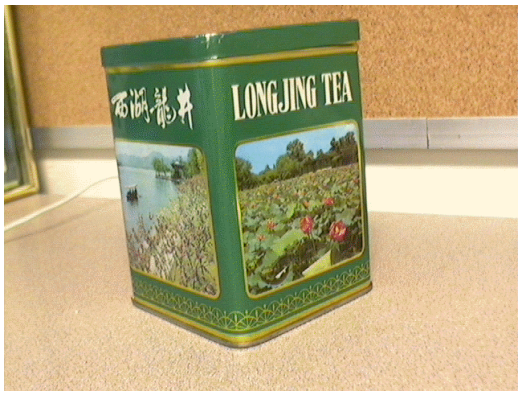
$$\begin{bmatrix} wx_n \\ wy_n \\ wz_n \\ w \end{bmatrix} = \begin{bmatrix} \frac{2}{w} & 0 & 0 & -1 \\ 0 & \frac{2}{h} & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} k_0 & k_1 & -k_2 & 0 \\ 0 & k_4 & -k_5 & 0 \\ 0 & 0 & \frac{-(f_p + n_p)}{f_p - n_p} & \frac{-2 f_p n_p}{f_p - n_p} \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} x_e \\ y_e \\ z_e \\ 1 \end{bmatrix}$$

ou

$$[\mathbf{P}_{OGL}] = \begin{bmatrix} \frac{2k_o}{w} & \frac{2k_1}{w} & 1 - \frac{2k_2}{w} & 0 \\ 0 & \frac{2k_4}{h} & 1 - \frac{2k_5}{h} & 0 \\ 0 & 0 & -\frac{f_p + n_p}{f_p - n_p} & -\frac{2 f_p n_p}{f_p - n_p} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

3. No site http://research.microsoft.com/~zhang/Calib/, Zhang fornece duas imagens de uma lata de chá com pontos de correspondência. Utilize estas imagens e calcule (utilizando softwares como o MathLab) a matriz Fundamental que relaciona as retas epipolares destas imagens utilizando o algoritmo de 8 pontos. Lembrem que ele tem um problema de estabilidade numérica que pode ser resolvido normalizando a imagem. Escolha um ponto em cada imagem e determine, na outra a sua reta epipolar.





Resposta:

Pontos de correspondência dados

```
(0.2613,0.2344)<->(0.2047,0.2427)
(0.4883,0.1781)<->(0.3531,0.2016)
(0.7223,0.1505)<->(0.6656,0.1667)
(0.5000,0.3000)<->(0.3652,0.3203)
(0.5824,0.1828)<->(0.4414,0.2057)
(0.2973,0.4943)<->(0.2352,0.4932)
(0.3438,0.6969)<->(0.2730,0.6891)
(0.5414,0.7370)<->(0.4105,0.7542)
(0.5113,0.3563)<->(0.3770,0.3760)
(0.3098,0.2073)<->(0.2352,0.2208)
(0.5863,0.2750)<->(0.4484,0.2979)
(0.7309,0.2271)<->(0.6801,0.2458)
(0.7363,0.6130)<->(0.6816,0.6500)
(0.6164,0.7208)<->(0.4844,0.7495)
(0.7316,0.3021)<->(0.6793,0.3250)
(0.5898,0.3688)<->(0.4516,0.3917)
```

Pontos de correspondência normalizados

```
(-1.0000,-0.5252)<->(-0.9529,-0.6282)
(-0.1687,-0.7313)<->(-0.3431,-0.7972)
(0.6884,-0.8324)<->(0.9406,-0.9406)
(-0.1257,-0.2849)<->(-0.2934,-0.3094)
(0.1762,-0.7141)<->(0.0196,-0.7801)
(-0.8684,0.4268)<->(-0.8277,0.4009)
(-0.6981,1.1689)<->(-0.6720,1.2054)
(0.0259,1.3158)<->(-0.1072,1.4728)
(-0.0842,-0.0788)<->(-0.2452,-0.0805)
(-0.8226,-0.6245)<->(-0.8277,-0.7181)
(0.1905,-0.3764)<->(0.0484,-0.4014)
(0.7199,-0.5520)<->(1.0000,-0.6154)
(0.7399,0.8617)<->(1.0064,1.0449)
(0.3007,1.2567)<->(0.1961,1.4535)
(0.7228,-0.2772)<->(0.9968,-0.2902)
(0.2034,-0.0330)<->(0.0613,-0.0163)
```

Pontos de correspondência em coordenadas da janela [0,w]×[0,h]

```
(  84, 184)<->(  66, 182)
( 156, 197)<->( 113, 192)
```

```
( 231, 204)<->( 213, 200)
( 160, 168)<->( 117, 163)
( 186, 196)<->( 141, 191)
(  95, 121)<->(  75, 122)
( 110,  73)<->(  87,  75)
( 173,  63)<->( 131,  59)
( 164, 155)<->( 121, 150)
(  99, 190)<->(  75, 187)
( 188, 174)<->( 144, 169)
( 234, 186)<->( 218, 181)
( 236,  93)<->( 218,  84)
( 197,  67)<->( 155,  60)
( 234, 168)<->( 217, 162)
( 189, 152)<->( 145, 146)
```

Matriz de normalização de coordenadas da esquerda [R_left]:

```
3.6630 0.0000 -1.9573
0.0000 3.6630 -1.3838
0.0000 0.0000  1.0000
```

Matriz de normalização de coordenadas da direita [R_right]:

```
4.1079 0.0000 -1.7937
0.0000 4.1079 -1.6252
0.0000 0.0000  1.0000
```

[A]

```
0.9529 0.6282 -1.0000 0.5005 0.3300 -0.5252 -0.9529 -0.6282 1.0000
0.0579 0.1345 -0.1687 0.2509 0.5830 -0.7313 -0.3431 -0.7972 1.0000
0.6476 -0.6475 0.6884 -0.7830 0.7830 -0.8324 0.9406 -0.9406 1.0000
0.0369 0.0389 -0.1257 0.0836 0.0881 -0.2849 -0.2934 -0.3094 1.0000
0.0034 -0.1374 0.1762 -0.0140 0.5571 -0.7141 0.0196 -0.7801 1.0000
0.7187 -0.3481 -0.8684 -0.3532 0.1711 0.4268 -0.8277 0.4009 1.0000
0.4691 -0.8414 -0.6981 -0.7856 1.4090 1.1689 -0.6720 1.2054 1.0000
-0.0028 0.0382 0.0259 -0.1411 1.9379 1.3158 -0.1072 1.4728 1.0000
0.0207 0.0068 -0.0842 0.0193 0.0063 -0.0788 -0.2452 -0.0805 1.0000
0.6808 0.5907 -0.8226 0.5169 0.4484 -0.6245 -0.8277 -0.7181 1.0000
0.0092 -0.0765 0.1905 -0.0182 0.1511 -0.3764 0.0484 -0.4014 1.0000
0.7199 -0.4430 0.7199 -0.5520 0.3397 -0.5520 1.0000 -0.6154 1.0000
0.7447 0.7732 0.7399 0.8673 0.9004 0.8617 1.0064 1.0449 1.0000
0.0590 0.4370 0.3007 0.2464 1.8266 1.2567 0.1961 1.4535 1.0000
0.7204 -0.2097 0.7228 -0.2763 0.0804 -0.2772 0.9968 -0.2902 1.0000
0.0125 -0.0033 0.2034 -0.0020 0.0005 -0.0330 0.0613 -0.0163 1.0000
```

Matriz Fundamental obtida pela Solução Ax=0, [F0]:

```
-0.0136  0.0176 0.0540
 0.0238 -0.0006 0.7431
-0.0132 -0.6661 0.0061
```

Se calcularmos a distância média quadrática dos pontos fornecidos às retas epipolares com [F0] temos **0.000508**

Decompondo [F0] em [U][D][V]t temos:

[U]

```
-0.9971 -0.0712  0.0282
 0.0721 -0.9969  0.0318
-0.0259 -0.0338 -0.9991
```

[D]

```
0.0156 0.0000 0.0000
0.0000 0.7455 0.0000
0.0000 0.0000 0.6664
```

[V]

```
 0.9993 -0.0300 0.0204
-0.0195  0.0293 0.9994
-0.0306 -0.9991 0.0287
```

Anulando o menor valor de [D] para forçar o rank([F])=2, temos:

[D]

```
0.0000 0.0000 0.0000
```

```
0.0000 0.7455 0.0000
0.0000 0.0000 0.6664
```
e a Matriz Fundamental [F]
```
 0.0020 0.0173 0.0536
 0.0227 -0.0006 0.7432
-0.0128 -0.6662 0.0060
```
A distância média quadrática dos pontos fornecidos às retas epipolares com [F] resulta em
0.002064

[F] em coordenadas [0,w]x[0,h] é dada por:
```
 0.0000 -0.0000  0.0027
-0.0000 -0.0000 -0.0375
 0.0004  0.0336  0.9987
```

Retas epipolares:

Pontos correspondentes de teste: `(95,121) <-> (75,122)`

O ponto a esquerda `(95,121)` gera a linha `a=0.002373 b=-0.037827 c=5.105315` a direita.
Esta linha passa por `0.0024*75-0.0378*122+5.1053=0.0000` que e' o correspondente do ponto da esquerda.

O ponto a direta `(75,122)` gera a linha `a=-0.000046 b=0.033426 c=-3.371923` a esquerda.
Esta linha passa por `-0.000046*95+0.0334*121-3.3719=0.0000` que e' o correspondente do ponto da direita.