Technische Universität München
Fakultät für Informatik
Prof. Dr. Nassir Navab, Dr.
Selim Ben Himane

SoSe 2007
Exercise Sheet 1
April 24, 2007

Exercises in 3D Computer Vision

Use Matlab and write m-Files to solve this exercises.

This exercice sheet can be done in a team of max. 3 students.

Short remainder on projective geometry
We call $\mathbb{P}^n$ the projective space of dimension n. For any point in $\mathbb{P}^n$, we define its corresponding vector in the homogeneous coordinates:

$$\mathbf{x}_h = [x_1, x_2, \ldots, x_{n+1}]^\top \in \mathbb{R}^{n+1}$$

We introduce on $\mathbb{P}^n$ a new equivalence operator $\equiv$ such that: $\forall \mathbf{x}_h, \mathbf{y}_h \in \mathbb{R}^{n+1}$, we have

$$\mathbf{x}_h \equiv \mathbf{y}_h \Leftrightarrow \exists \alpha \in \mathbb{R}^* \text{ where } \mathbf{x}_h = \alpha \mathbf{y}_h$$

If $x_{n+1} = 0$, then $\mathbf{x}_h$ is a *point at the infinity*. If $x_{n+1} \neq 0$, then we can obtain the Euclidian coordinates $\mathbf{x}_e \in \mathbb{R}^n$ from the homogenenous coordinates $\mathbf{x}_h$.

For $\mathbf{x}_h = [x_1, x_2, \ldots, x_{n+1}]^\top$ and $x_{n+1} \neq 0$, then $\mathbf{x}_e = \left[\frac{x_1}{x_{n+1}}, \frac{x_2}{x_{n+1}}, \ldots, \frac{x_n}{x_{n+1}}\right]^\top$.

Exercise 1 (P) Projective 2D Transformations (40 Pts)

- $\mathbf{A}_1$ is a 3×3 homogeneous matrix that scales a factor of 2 in the x direction and a factor of 0.5 in the y direction. What is the general form of this matrix? Provide two numerical examples, one normalized where $\mathbf{A}(3,3) = 1$.
- $\mathbf{A}_2$ is a 2D rotation. It rotates a 2D points by an angle of $\frac{\pi}{3}$ clockwise. Give the matrix $\mathbf{A}_2$.
- Plot the points $\mathbf{p}_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $\mathbf{p}_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$, $\mathbf{p}_3 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, $\mathbf{p}_4 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, $\mathbf{p}_5 = \begin{bmatrix} 0.5 \\ 1.5 \end{bmatrix}$, $\mathbf{p}_6 = \begin{bmatrix} 0 \\ -1 \end{bmatrix}$ and $\mathbf{p}_7 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$ as blue crosses; draw the polygon $\overline{\mathbf{p}_1\mathbf{p}_2\mathbf{p}_4\mathbf{p}_5\mathbf{p}_3\mathbf{p}_1\mathbf{p}_4\mathbf{p}_3\mathbf{p}_2}$ in red; draw the lines $\overline{\mathbf{p}_1\mathbf{p}_2}$ and $\overline{\mathbf{p}_1\mathbf{p}_3}$ in blue - these are your axes. Rotate the points $\mathbf{p}_i$, where $i \in \{1, \ldots, 7\}$, with the matrix $\mathbf{A}_2$ and draw them as blue circles; draw the rotated lines as dotted lines. The original points and line and the rotated line must be in the same window.

  Reminder: Concatenate all points into one matrix in order to use the following property:

$$[\mathbf{A}_2\mathbf{p}_1, \mathbf{A}_2\mathbf{p}_2, \ldots, \mathbf{A}_2\mathbf{p}_7] = \mathbf{A}_2 [\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_7]$$

- Do the same as above with matrix $\mathbf{A}_1$ in another figure window.

- Change the points to homogeneous coordinates. Do the same with the matrix $\mathbf{A}_3$ that performs a translation of $\mathbf{t} = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$.
- $\mathbf{A}_4$ performs an affine projection. Take a look at the reminder above to create a transformation $\mathbf{A}_4$, that changes the unit vector of x-axis $[1,0]^\top$ to $[2,1]^\top$ and the unit vector of y-axis $[0,1]^\top$ to $[0.5, 0.8]^\top$. Confirm your result repeating the drawing steps above.
- Apply the projective transformation defined by the matrix $\mathbf{A}_5 = \begin{bmatrix} 1.02 & 0.01 & 0 \\ 0 & 1.03 & 0 \\ 0.4 & 0.1 & 1 \end{bmatrix}$ to the points and visualize it as above in a 2D plot.
- $\mathbf{A}_6$ is the matrix that performs first the rotation of $\mathbf{A}_2$, then the translation of $\mathbf{A}_3$. Define this matrix. Draw the points and lines like in the questions before.
- $\mathbf{A}_7$ is the matrix that performs first the scaling of $\mathbf{A}_1$, then the rotation of $\mathbf{A}_2$, then the translation of $\mathbf{A}_3$ and finally the projective transformation of $\mathbf{A}_5$. Visualize the transformation as in the tasks above.

Exercise 2 (P) Duality between lines and points (20 Pts)

- Draw the line $\mathbf{l}_1 = \begin{bmatrix} 2 \\ 1 \\ -0.5 \end{bmatrix}$ and the point $\mathbf{p}_1 = \begin{bmatrix} 2 \\ 1 \\ -0.5 \end{bmatrix}$ in blue and the point $\mathbf{p}_2 = \begin{bmatrix} 0.5 \\ 1 \\ 1 \end{bmatrix}$ and the line $\mathbf{l}_2 = \begin{bmatrix} 0.5 \\ 1 \\ 1 \end{bmatrix}$, in red into Figure 1. Add the axes in black.
  Reminder: A 2D point with homogeneous coordinates $\mathbf{p} = [x, y, w]^\top$ belongs to a line $\mathbf{l} = [a, b, c]^\top$ if it verifies:
  $$ax + by + cw = 0$$
  For drawing it, you may want to set $w = 1$ (Euclidean coordinates), transform the equation above to the form:
  $$y = mx + n$$
  and use sample points for x in a certain interval. In this exercise, we suggest to take the interval $x \in [-5, 5]$.
- Draw $\mathbf{p}_1$, $\mathbf{p}_2$ as points and $\mathbf{p}_1 \times \mathbf{p}_2$ as a line into Figure 2 and $\mathbf{l}_1$, $\mathbf{l}_2$ as lines and $\mathbf{l}_1 \times \mathbf{l}_2$ as a point into Figure 3.

Exercise 3 (P) Projective 2D Transformations for images (40 Pts)

In this exercise, we define the function transImage = transformImage(I, $\mathbf{M}$) that transforms the image I with the homogeneous (3×3) projection matrix $\mathbf{M}$.

- Create a function $\mathbf{p}_e = homo2euclid\,(\mathbf{p}_h)$ that transforms a point from homogeneous coordinates $\mathbf{p}_h = [x, y, w]^\top \in \mathbb{R}^3$ to Euclidean coordinates $\mathbf{p}_e \in \mathbb{R}^2$ (normalize by w).
- Create the function transImage = transformImageSimple(I,M) that projects each point into the resulting image. Disregard zero and negative values.
  Hint: Find out the new boundaries of the TransImage before creating it.

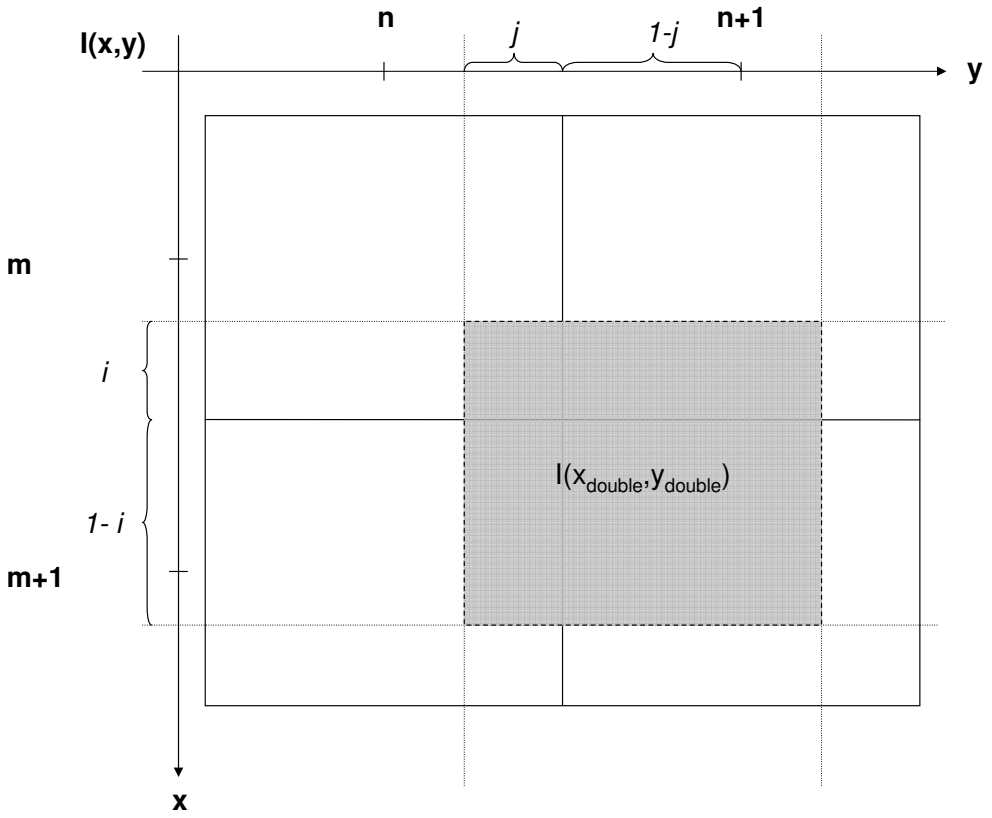

Figure 1: Bilinear interpolation. Each pixel has the size 1.

- Try the matrices $\mathbf{A}_1$ to $\mathbf{A}_6$ with this function. Change $\mathbf{A}_5$ to $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0.001 & 0.003 & 1 \end{bmatrix}$ before in order to have a nice projective projection. Are you satisfied with the result? Why was the result predictable ?
- Create function transImage = transformImageReverse(I,M) that takes each point in the resulting image, transforms back to the original image takes the closest point in the original image. Use the inverse matrix of the matrix $\mathbf{M}$.
- Change the last function: instead of taking the closest point take the 4 closest points and weigh them according to the area that is covered by the pixel that has a real (not discrete) value.
- ***(Bonus*** List some others interpolations techniques. Explain them briefly.