Prof. Dr.-Ing. A. Bruhn
Institute for Visualization and Interactive Systems
Department Intelligent Systems
University of Stuttgart

Example Solution for Homework Assignment 2

Problem 2.1 (Cooccurrence Matrices) 8 Points

The cooccurrence matrix is set up by counting pairs of grey values (i, j) that occur along the vector $\vec{d} = (-1, -1)^\top$. Precisely, if we denote the image with f, the (i, j) entry of the cooccurrence matrix gets incremented for all (k, l) such that $f_{k,l} = i$ and $f_{k-1,l-1} = j$.

0	3	2	1
1	1	3	2
0	0	2	1
3	0	1	0

The cooccurrence matrix is given by:

j: 0	1	2	3	i
1	1	1	0	0
2	0	0	1	1
0	1	1	0	2
0	0	0	1	3

(a) The most frequent configuration is $(1, 0)$ with $p_{1,0} = \frac{2}{9}$.

The contrast can be computed as follows:

$$
\begin{aligned}
\sum_{i,j} (i-j)^2 \, p_{i,j} \quad = \quad & (0-0)^2 \cdot \frac{1}{9} + (0-1)^2 \cdot \frac{1}{9} + (0-2)^2 \cdot \frac{1}{9} + (0-3)^2 \cdot \frac{0}{9} \\
& + (1-0)^2 \cdot \frac{2}{9} + (1-1)^2 \cdot \frac{0}{9} + (1-2)^2 \cdot \frac{0}{9} + (1-3)^2 \cdot \frac{1}{9} \\
& + (2-0)^2 \cdot \frac{0}{9} + (2-1)^2 \cdot \frac{1}{9} + (2-2)^2 \cdot \frac{1}{9} + (2-3)^2 \cdot \frac{0}{9} \\
& + (3-0)^2 \cdot \frac{0}{9} + (3-1)^2 \cdot \frac{0}{9} + (3-2)^2 \cdot \frac{0}{9} + (3-3)^2 \cdot \frac{1}{9} \\
= \quad & \frac{1}{9} (0+1+4+0 \; + \; 2+0+0+4 \; + \; 0+1+0+0 \; + \; +0+0+0+0) \\
= \quad & \frac{12}{9}
\end{aligned}
$$

(b) If we choose $\mathbf{d} = (1,1)^\top$, each entry (i,j) of the old cooccurrence matrix becomes an entry (j,i) of the new one: by setting $(\tilde{k}, \tilde{j}) := (k-1, l-1)$, we see that

$$\exists (k,l) : (f_{k,l} = i) \wedge (f_{k-1,l-1} = j) \;\Leftrightarrow\; \exists (\tilde{k}, \tilde{l}) : (f_{\tilde{k},\tilde{l}} = j) \wedge (f_{\tilde{k}+1,\tilde{l}+1} = i)$$

In fact, for each pair of grey values, the roles of the two partners are exchanged:

0	3	2	1
1	1	3	2
0	0	2	1
3	0	1	0

The cooccurrence matrix is given by:

j				
0	1	2	3	i
1	2	0	0	0
1	0	1	0	1
1	0	1	0	2
0	1	0	1	3

Thus, the new cooccurrence matrix is the transposed of the old one. That means that the highest probability is reached for $(0,1)$ with $p_{0,1} = \frac{2}{9}$. On the other hand, the contrast is symmetric with respect to i and j, so that the new cooccurrence matrix will have the same contrast as the old one.

Problem 2.2 (Lucas and Kanade) — 8 Points

Applying Cramer's rule leads us to the following results:

$$u = \frac{\det\begin{pmatrix} b_1 & a_{12} \\ b_2 & a_{22} \end{pmatrix}}{\det\begin{pmatrix} a_{11} & a_{12} \\ a_{12} & a_{22} \end{pmatrix}} = \frac{b_1 a_{22} - b_2 a_{12}}{a_{11}a_{22} - a_{12}^2}$$

$$v = \frac{\det\begin{pmatrix} a_{11} & b_1 \\ a_{12} & b_2 \end{pmatrix}}{\det\begin{pmatrix} a_{11} & a_{12} \\ a_{12} & a_{22} \end{pmatrix}} = \frac{b_2 a_{11} - b_1 a_{12}}{a_{11}a_{22} - a_{12}^2}$$

If we replace the abbreviations again, this gives

$$u = \frac{-\int_{B_\rho} f_x f_z \, dxdy \cdot \int_{B_\rho} f_y^2 \, dxdy + \int_{B_\rho} f_y f_z \, dxdy \cdot \int_{B_\rho} f_x f_y \, dxdy}{\int_{B_\rho} f_x^2 \, dxdy \cdot \int_{B_\rho} f_y^2 \, dxdy - \left(\int_{B_\rho} f_x f_y \, dxdy\right)^2}$$

$$v = \frac{-\int_{B_\rho} f_y f_z \, dxdy \cdot \int_{B_\rho} f_x^2 \, dxdy + \int_{B_\rho} f_x f_z \, dxdy \cdot \int_{B_\rho} f_x f_y \, dxdy}{\int_{B_\rho} f_x^2 \, dxdy \cdot \int_{B_\rho} f_y^2 \, dxdy - \left(\int_{B_\rho} f_x f_y \, dxdy\right)^2}$$

Problem 2.3 (Lucas and Kanade) 8 Points

The missing code in `create_eq_systems` reads:

```
w5 = 1.0 / ht;  df_dz = w5 * (f2[i][j] - f1[i][j]);
:
dxz[i][j] = df_dx * df_dz;
dyz[i][j] = df_dy * df_dz;
:
gauss_conv (rho, nx, ny, hx, hy, 5.0, 0, dxz);
gauss_conv (rho, nx, ny, hx, hy, 5.0, 0, dyz);
```

and in `lucas_kanade`:

```
trace = dxx[i][j] + dyy[i][j];

det = dxx[i][j] * dyy[i][j] - dxy[i][j] * dxy[i][j];

if (trace<=eps)
  {
  /* nothing can be said */
  u[i][j] = 0.0;
  v[i][j] = 0.0;
  c[i][j] = 0.0;
  }
else if (det<=eps)
  {
  /* we can only compute the normal flow */
  u[i][j] = -dxz[i][j] / trace;
  v[i][j] = -dyz[i][j] / trace;
  c[i][j] = 127.0;
  }
else
  {
  /* we can solve the system */
  u[i][j] = ( - dxz[i][j] * dyy[i][j] + dyz[i][j] * dxy[i][j] ) / det;
  v[i][j] = ( - dyz[i][j] * dxx[i][j] + dxz[i][j] * dxy[i][j] ) / det;
  c[i][j] = 255.0;
  }
```

Prof. Dr.-Ing. A. Bruhn
Institute for Visualization and Interactive Systems
Department Intelligent Systems
University of Stuttgart

Example Solution for Classroom Assignment 2

C 2.1 (Affine Lucas and Kanade)

In the lecture, we have written the flow variables u, v by means of an affine parametrisation based on the variables $a, \ldots, f$:

$$\vec{w} = \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} ax + by + c \\ dx + ey + f \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} x & y & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}}_{M} \cdot \underbrace{\begin{pmatrix} a \\ b \\ c \\ d \\ e \\ f \\ 1 \end{pmatrix}}_{\vec{p}}$$

Let us recall the data constraint from the standard Lucas-Kanade-method (that is the same as for the Horn-and-Schunck-method) and replace the estimate vector: $\vec{w} = M\vec{p}$. This gives

$$\begin{aligned} 0 &= \vec{w}^\top J \vec{w} \\ &= \vec{w}^\top \nabla_3 f \nabla_3 f^\top \vec{w} \\ &= \vec{p}^\top M^\top \nabla_3 f \nabla_3 f^\top M \vec{p} \\ &= \vec{p}^\top \left(M^\top \nabla_3 f \right) \left(\nabla_3 f^\top M \right) \vec{p} \\ &= \vec{p}^\top \underbrace{\left(M^\top \nabla_3 f \right)}_{\vec{r}} \underbrace{\left(M^\top \nabla_3 f \right)^\top}_{\vec{r}^\top} \vec{p} \\ &= \vec{p}^\top \underbrace{\left(\vec{r}\, \vec{r}^\top \right)}_{J_{\text{affine}}} \vec{p} \\ &= \vec{p}^\top J_{\text{affine}}\, \vec{p} \end{aligned}$$

By now, we have rewritten the data term such that the flow has an affine parametrisation. Therefore, we have replaced the estimates u, v (the variables that have to be computed) by the affine parameters a, b, c, d, e, f. This transforms the 3×3 quadratic form into a 7×7 quadratic form. Thus, for computing the optical flow with this affine method, we only have to increase the dimensionality compared to the standard LK-method.

It remains to compute the vector $\vec{r}$ and the affine tensor $J_{\text{affine}} = \vec{r}\,\vec{r}^\top$:

$$\mathbf{r} = M^\top \cdot \nabla_3 f = \begin{pmatrix} x & 0 & 0 \\ y & 0 & 0 \\ 1 & 0 & 0 \\ 0 & x & 0 \\ 0 & y & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix} = \begin{pmatrix} xf_x \\ yf_x \\ f_x \\ xf_y \\ yf_y \\ f_y \\ f_z \end{pmatrix}$$

We can recheck this result by computing

$$\begin{aligned} \mathbf{r}^\top \mathbf{p} &= \begin{pmatrix} xf_x \\ yf_x \\ f_x \\ xf_y \\ yf_y \\ f_y \\ f_z \end{pmatrix}^\top \cdot \begin{pmatrix} a \\ b \\ c \\ d \\ e \\ f \\ 1 \end{pmatrix} \\ &= axf_x + byf_x + cf_x + dxf_y + eyf_y + ff_y + f_z \\ &= (ax + by + c)f_x + (dx + ey + f)f_y + f_z \\ &= uf_x + vf_y + f_z \end{aligned}$$

The final tensor J_0 is given by

$$\begin{aligned} J_0 = \mathbf{r} \cdot \mathbf{r}^\top &= \begin{pmatrix} xf_x \\ yf_x \\ f_x \\ xf_y \\ yf_y \\ f_y \\ f_z \end{pmatrix} \cdot \begin{pmatrix} xf_x \\ yf_x \\ f_x \\ xf_y \\ yf_y \\ f_y \\ f_z \end{pmatrix}^\top \\ &= \begin{pmatrix} x^2f_x^2 & xyf_x^2 & xf_x^2 & x^2f_xf_y & xyf_xf_y & xf_xf_y & xf_xf_z \\ xyf_x^2 & y^2f_x^2 & yf_x^2 & xyf_xf_y & y^2f_xf_y & yf_xf_y & yf_xf_z \\ xf_x^2 & yf_x^2 & f_x^2 & xf_xf_y & yf_xf_y & f_xf_y & f_xf_z \\ x^2f_xf_y & xyf_xf_y & xf_xf_y & x^2f_y^2 & xyf_y^2 & xf_y^2 & xf_yf_z \\ xyf_xf_y & y^2f_xf_y & yf_xf_y & xyf_y^2 & y^2f_y^2 & yf_y^2 & yf_yf_z \\ xf_xf_y & yf_xf_y & f_xf_y & xf_y^2 & yf_y^2 & f_y^2 & f_yf_z \\ xf_xf_z & yf_xf_z & f_xf_z & xf_yf_z & yf_yf_z & f_yf_z & f_z^2 \end{pmatrix} \end{aligned}$$