Multiple View Geometry: Solution Sheet 2

Prof. Dr. Daniel Cremers, Simon Weber, Tarun Yenamandra
Computer Vision Group, TU Munich
`Wednesdays 16:00-18:15 at Hrsaal 2, "Interims I" (5620.01.102), and on RBG Live`

Exercise: May 18th, 2022

Part I: Theory

1. Groups and inclusions:
   Groups

   (a) $SO(n)$: special orthogonal group
   (b) $O(n)$: orthogonal group
   (c) $GL(n)$: general linear group
   (d) $SL(n)$: special linear group
   (e) $SE(n)$: special euclidean group (In particular, $SE(3)$ represents the rigid-body motions in $\mathbb{R}^3$)
   (f) $E(n)$: euclidean group
   (g) $A(n)$: affine group

   Inclusions

   (a) $SO(n) \subset O(n) \subset GL(n)$
   (b) $SE(n) \subset E(n) \subset A(n) \subset GL(n+1)$

2. $\lambda_a = \frac{(\lambda_a v_a)^\top v_b}{\langle v_a, v_b \rangle} = \frac{v_a^\top A^\top v_b}{\langle v_a, v_b \rangle} = \frac{v_a^\top A v_b}{\langle v_a, v_b \rangle} = \frac{v_a^\top (\lambda_b v_b)}{\langle v_a, v_b \rangle} = \lambda_b$

3. Let V be the orthonormal matrix (i.e. $V^\top = V^{-1}$) given by the eigenvectors, and Σ the diagonal matrix containing the eigenvalues:

$$V = \begin{pmatrix} | & & | \\ v_1 & \cdots & v_n \\ | & & | \end{pmatrix} \quad \text{and} \quad \Sigma = \begin{pmatrix} \lambda_1 & 0 & \ddots \\ 0 & \ddots & 0 \\ \ddots & 0 & \lambda_n \end{pmatrix}.$$

   As V is a basis, we can express x as a linear combination of the eigenvectors $x = V\alpha$, for some $\alpha \in \mathbb{R}^n$. For $||x|| = 1$ we have $\sum_i \alpha_i^2 = \alpha^\top \alpha = x^\top V V^\top x = x^\top x = 1$. This gives

$$\begin{aligned} x^\top A x &= x^\top V \Sigma V^{-1} x \\ &= \alpha^\top V^\top V \Sigma V^\top V \alpha \\ &= \alpha^\top \Sigma \alpha = \sum_i \alpha_i^2 \lambda_i \end{aligned}$$

Considering $\sum_i \alpha_i^2 = 1$, we can conclude that this expression is minimized iff only the α_i corresponding to the smallest eigenvalue(s) are non-zero. If $\lambda_{n-1} \gneq \lambda_n$, there exist only two solutions ($\alpha_n = \pm 1$), otherwise infinitely many.

For maximisation, only the the α_i corresponding to the largest eigenvalue(s) can be non-zero.

4. We show that: $x \in \text{kernel}(A) \Leftrightarrow x \in \text{kernel}(A^\top A)$.

   "$\Rightarrow$": Let $x \in \text{kernel}(A)$

   $$A^\top \underbrace{Ax}_{=0} = A^\top 0 = 0 \quad \Rightarrow x \in \text{kernel}(A^\top A)$$

   "$\Leftarrow$": Let $x \in \text{kernel}(A^\top A)$

   $$0 = x^\top \underbrace{A^\top Ax}_{=0} = \langle Ax, Ax \rangle = ||Ax||^2 \quad \Rightarrow Ax = 0 \quad \Rightarrow x \in \text{kernel}(A)$$

5. Singular Value Decomposition (SVD)

   Note: There exist multiple slightly different definitions of the SVD. Depending on the convention used, we might have $S \in \mathbb{R}^{m \times n}$, $S \in \mathbb{R}^{n \times n}$, or $S \in \mathbb{R}^{p \times p}$ where $p = \mathrm{rank}(A)$. In the lecture the third option was presented, for which S is invertible (no zeros on the diagonal). In the following, we present the results for the first option, since that is the one that Matlab's `svd` function returns by default.

   (a) $A \in \mathbb{R}^{m \times n}$, $U \in \mathbb{R}^{m \times m}$, $S \in \mathbb{R}^{m \times n}$, $V \in \mathbb{R}^{n \times n}$

   (b) Similarities and differences between SVD and EVD:

      i. Both are matrix diagonalization techniques.

      ii. The SVD can be applied to matrices $A \in \mathbb{R}^{m \times n}$ with $m \neq n$, whereas the EVD is only applicable to quadratic matrices ($A \in \mathbb{R}^{m \times n}$ with $m = n$).

   (c) Relationship between U, S, V and the eigenvalues and eigenvectors of $A^\top A$ and $AA^\top$:

      i. $A^\top A$: The columns of V are eigenvectors; the squares of the diagonal elements of S are eigenvalues.

      ii. $AA^\top$: The columns of U are eigenvectors; the squares of the diagonal elements of S are eigenvalues (possibly filled up with zeros).

   (d) Entries in S:

      i. S is a diagonal matrix. The elements along the diagonal are the *singular values* of A.

      ii. The number of non-zero singular values gives us the *rank* of the matrix A.