Computer Vision: Lecture 4

2023-11-09

Camera Matrix Cheat Sheet

- Camera (projection) matrix $\mathtt{P} = \mathtt{K}(\mathtt{R} \mid \mathbf{T}) = (\mathtt{P}_{3,3} \mid \mathtt{P}_4)$
  - $\mathtt{R}$ is rotation matrix, $\mathtt{K} = \begin{pmatrix} \gamma f & sf & x_0 \\ 0 & f & y_0 \\ 0 & 0 & 1 \end{pmatrix}$ is upper triangular calibration matrix
  - *Focal length* f and *principal point* (x_0, y_0)
  - Advice: always scale $\mathtt{P}$ such that $\|\mathtt{P}_{3,1:3}\| = 1$ and $\det(\mathtt{P}_{3,3}) > 0$ (why?)
- Image point $\mathbf{x} \sim \mathtt{P}\mathbf{X} \iff \lambda\mathbf{x} = \mathtt{P}\mathbf{X}$ for some $\lambda \neq 0$ $\quad (\mathbf{X} \in \mathbb{P}^3)$
  - Transform to camera coordinate system (CCS): $\mathbf{X}' = \mathtt{R}\mathbf{X} + \mathbf{T}$ $\quad (\mathbf{X} \in \mathbb{R}^3)$
  - Plus $\mathtt{K}$ and π: $\lambda\mathbf{x} = \mathtt{K}(\mathtt{R}\mathbf{X} + \mathbf{T}) = \mathtt{K}\mathtt{R}(\mathbf{X} - \mathbf{C})$ $\quad (\mathbf{X} \in \mathbb{R}^3)$
  - Camera center $\mathbf{C}$ is origin in CCS: defining relation $\mathbf{0} = \mathtt{R}\mathbf{C} + \mathbf{T}$
  - How can we compute camera center given $\mathtt{P}$? Hint: what is $\mathtt{K}\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$?
- Transform 3D direction $\mathbf{d}$ to CCS: $\mathbf{d}' = \mathtt{R}\mathbf{d}$
  - Principal (or optical) axis $\mathbf{e}_z = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ (z-axis) in CCS
  - Principal axis $\mathbf{a}$ in WCS: defining relation $\mathbf{e}_z = \mathtt{R}\,\mathbf{a}$
  - How can we compute principal axis given $\mathtt{P}$? Hint: what is $\mathtt{P}_{3,1:3}$?
  - Principal point: image of the principical axis $\begin{pmatrix} x_0 \\ y_0 \\ 1 \end{pmatrix} \sim \mathtt{P}_{3,3}\,\mathbf{a}$
- 3D line in WCS generated by image point $\mathbf{x}$: $\mathbf{X}(\lambda) = \lambda\mathtt{P}_{3,3}^{-1}\mathbf{x} - \mathtt{P}_{3,3}^{-1}\mathtt{P}_4$
- 2D points on line $\mathbf{l}$ are projections of 3D points on plane $\Pi = \mathtt{P}^\top\mathbf{l}$

Today's Lecture

Direct Linear Transform (DLT)

- Camera pose estimation (Szelisky, Sec. 11.2.1)
- Triangulation (Szelisky, Sec. 11.2.4)
- Homography estimation and panoramas (Szelisky, Sec. 8.2.3)

Keypoint detection and Matching (Szelisky, Sec. 7.1)

- Good feature points (Szelisky, Sec. 7.1.1)
- Invariance
- Feature descriptors - SIFT (Szelisky, Sec. 7.1.2)
- Matching

Direct Linear Transformation — DLT

Last Time: Simplified DLT $\lambda_i \mathbf{x}_i = \mathtt{P}\mathbf{X}_i$

$$\underbrace{\begin{pmatrix} \mathbf{X}_1^\top & 0 & 0 & -x_1 & 0 & 0 & \dots \\ 0 & \mathbf{X}_1^\top & 0 & -y_1 & 0 & 0 & \dots \\ 0 & 0 & \mathbf{X}_1^\top & -1 & 0 & 0 & \dots \\ \mathbf{X}_2^\top & 0 & 0 & 0 & -x_2 & 0 & \dots \\ 0 & \mathbf{X}_2^\top & 0 & 0 & -y_2 & 0 & \dots \\ 0 & 0 & \mathbf{X}_2^\top & 0 & -1 & 0 & \dots \\ \mathbf{X}_3^\top & 0 & 0 & 0 & 0 & -x_3 & \dots \\ 0 & \mathbf{X}_3^\top & 0 & 0 & 0 & -y_3 & \dots \\ 0 & 0 & \mathbf{X}_3^\top & 0 & 0 & -1 & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}}_{=\mathtt{M}} \underbrace{\begin{pmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \\ \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \vdots \end{pmatrix}}_{=\mathbf{v}} = \mathbf{0}.$$

- $3N \times (12 + N)$ matrix $\mathtt{M}$
- Is there a more efficient approach when N is large?

Anti-symmetrization

- Given relation between $\mathbf{x} \in \mathbb{R}^k$ and $\mathbf{y} \in \mathbb{R}^k$

$$\mathbf{x} = \lambda \mathbf{y} \qquad \lambda \neq 0$$

  k equations

- Cross-multiply equations

$$x_i = \lambda y_i \implies x_i \lambda y_j = x_j \lambda y_i \iff x_i y_j - x_j y_i = 0$$

- $k-1$ independent equations (e.g. choose $j = i+1$)

$$\begin{aligned} 0 &= x_1 y_2 - x_2 y_1 \\ 0 &= x_2 y_3 - x_3 y_2 \\ &\vdots \\ 0 &= x_{k-1} y_k - x_k y_{k-1} \end{aligned}$$

- $k = 3$: anti-symmetrization is essentially cross product $\mathbf{x} \times \mathbf{y}$
  - Only 2 equations are linearly independent

Anti-symmetrization

- Given relation between $\mathbf{x} \in \mathbb{R}^k$ and $\mathbf{y} \in \mathbb{R}^k$

$$\mathbf{x} = \lambda \mathbf{y} \qquad \lambda \neq 0$$

  k equations

- Cross-multiply equations

$$x_i = \lambda y_i \implies x_i \lambda y_j = x_j \lambda y_i \iff x_i y_j - x_j y_i = 0$$

- $k - 1$ independent equations (e.g. choose $j = i + 1$)

$$\begin{aligned} 0 &= x_1 y_2 - x_2 y_1 \\ 0 &= x_2 y_3 - x_3 y_2 \\ &\vdots \\ 0 &= x_{k-1} y_k - x_k y_{k-1} \end{aligned}$$

- $k = 3$: anti-symmetrization is essentially cross product $\mathbf{x} \times \mathbf{y}$
  - Only 2 equations are linearly independent

Anti-symmetrization

- Given relation between $\mathbf{x} \in \mathbb{R}^k$ and $\mathbf{y} \in \mathbb{R}^k$

$$\mathbf{x} = \lambda \mathbf{y} \qquad \lambda \neq 0$$

  k equations

- Cross-multiply equations

$$x_i = \lambda y_i \implies x_i \lambda y_j = x_j \lambda y_i \iff x_i y_j - x_j y_i = 0$$

- $k - 1$ independent equations (e.g. choose $j = i + 1$)

$$\begin{aligned} 0 &= x_1 y_2 - x_2 y_1 \\ 0 &= x_2 y_3 - x_3 y_2 \\ &\vdots \\ 0 &= x_{k-1} y_k - x_k y_{k-1} \end{aligned}$$

- $k = 3$: anti-symmetrization is essentially cross product $\mathbf{x} \times \mathbf{y}$
  - Only 2 equations are linearly independent

Direct Linear Transformation – DLT

- Apply on $\lambda_i \mathbf{x}_i = \mathtt{P}\mathbf{X}_i$

$$\mathbf{x}_i \times (\mathtt{P}\mathbf{X}_i) = \mathbf{0}$$

- Provides two equations, e.g.

$$0 = P_{11}X_i + P_{12}Y_i + P_{13}Z_i + P_{14} - x_i(P_{31}X_i + P_{32}Y_i + P_{33}Z_i + P_{34})$$
$$0 = P_{21}X_i + P_{22}Y_i + P_{23}Z_i + P_{24} - y_i(P_{31}X_i + P_{32}Y_i + P_{33}Z_i + P_{34})$$

- Provide two rows in matrix $\mathtt{M}$

$$\begin{pmatrix} X_i & Y_i & Z_i & 1 & 0 & 0 & 0 & 0 & -x_iX_i & -x_iY_i & -x_iZ_i & -x_i \\ 0 & 0 & 0 & 0 & X_i & Y_i & Z_i & 1 & -y_iX_i & -y_iY_i & -y_iZ_i & -y_i \end{pmatrix}$$

Direct Linear Transformation – DLT

- Apply on $\lambda_i \mathbf{x}_i = \mathtt{P}\mathbf{X}_i$

$$\mathbf{x}_i \times (\mathtt{P}\mathbf{X}_i) = \mathbf{0}$$

- Provides two equations, e.g.

$$0 = P_{11}X_i + P_{12}Y_i + P_{13}Z_i + P_{14} - x_i(P_{31}X_i + P_{32}Y_i + P_{33}Z_i + P_{34})$$
$$0 = P_{21}X_i + P_{22}Y_i + P_{23}Z_i + P_{24} - y_i(P_{31}X_i + P_{32}Y_i + P_{33}Z_i + P_{34})$$

- Provide two rows in matrix $\mathtt{M}$

$$\begin{pmatrix} X_i & Y_i & Z_i & 1 & 0 & 0 & 0 & 0 & -x_iX_i & -x_iY_i & -x_iZ_i & -x_i \\ 0 & 0 & 0 & 0 & X_i & Y_i & Z_i & 1 & -y_iX_i & -y_iY_i & -y_iZ_i & -y_i \end{pmatrix}$$

Direct Linear Transformation – DLT

- Apply on $\lambda_i \mathbf{x}_i = \mathtt{P}\mathbf{X}_i$

$$\mathbf{x}_i \times (\mathtt{P}\mathbf{X}_i) = \mathbf{0}$$

- Provides two equations, e.g.

$$0 = P_{11}X_i + P_{12}Y_i + P_{13}Z_i + P_{14} - x_i(P_{31}X_i + P_{32}Y_i + P_{33}Z_i + P_{34})$$
$$0 = P_{21}X_i + P_{22}Y_i + P_{23}Z_i + P_{24} - y_i(P_{31}X_i + P_{32}Y_i + P_{33}Z_i + P_{34})$$

- Provide two rows in matrix $\mathtt{M}$

$$\begin{pmatrix} X_i & Y_i & Z_i & 1 & 0 & 0 & 0 & 0 & -x_iX_i & -x_iY_i & -x_iZ_i & -x_i \\ 0 & 0 & 0 & 0 & X_i & Y_i & Z_i & 1 & -y_iX_i & -y_iY_i & -y_iZ_i & -y_i \end{pmatrix}$$

Direct Linear Transformation – DLT

Standard DLT: Camera resectioning

For $N \geq 6$ correspondences $\mathbf{x}_i \leftrightarrow \mathbf{X}_i$ solve

$$\underbrace{\begin{pmatrix} & & & & & & \vdots & & & & & \\ X_i & Y_i & Z_i & 1 & 0 & 0 & 0 & 0 & -x_iX_i & -x_iY_i & -x_iZ_i & -x_i \\ 0 & 0 & 0 & 0 & X_i & Y_i & Z_i & 1 & -y_iX_i & -y_iY_i & -y_iZ_i & -y_i \\ & & & & & & \vdots & & & & & \end{pmatrix}}_{=\mathtt{M}} \underbrace{\begin{pmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{pmatrix}}_{=\mathbf{v}} = \mathbf{0}$$

to determine the camera matrix $\mathtt{P}$.

- $2N \times 12$ matrix $\mathtt{M}$
- Centering transformation to improve numerical stability
- Use RQ-decomposition to obtain $\mathtt{P} = \mathtt{K}(\mathtt{R} \mid \mathbf{T})$
  - If required and if you don't know $\mathtt{K}$

Improving the Accuracy of Floating Point Computations

- Recall

$$\mathtt{M} = \begin{pmatrix} & & & & & & \vdots & & & & & \\ X_i & Y_i & Z_i & 1 & 0 & 0 & 0 & 0 & -x_iX_i & -x_iY_i & -x_iZ_i & -x_i \\ 0 & 0 & 0 & 0 & X_i & Y_i & Z_i & 1 & -y_iX_i & -y_iY_i & -y_iZ_i & -y_i \\ & & & & & & \vdots & & & & & \end{pmatrix}$$

- Contains 1s, coordinate values (X_i, x_i, etc) and products (e.g. x_iX_i)
  - $x_i = 1000$ pixel and $X_i = 100\,m$: $x_iX_i = 100000 \gg 1$
  - Ideally all non-zero values have similar magnitude

Normalizing Transformation

Transform image points $\mathbf{x}'_i = \mathtt{T}_1\mathbf{x}$ and 3D points $\mathbf{X}'_i = \mathtt{T}_2\mathbf{X}_i$ such that

$$\frac{1}{N}\sum \mathbf{x}'_i = \mathbf{0} \qquad \frac{1}{N}\sum \|\mathbf{x}'_i\|^2 = 1 \qquad \frac{1}{N}\sum \mathbf{X}'_i = \mathbf{0} \qquad \frac{1}{N}\sum \|\mathbf{X}'_i\|^2 = 1$$

$\mathtt{T}_1$ and $\mathtt{T}_2$ are scalings + translations, e.g. $\mathtt{T}_1 = \begin{pmatrix} s_x & 0 & -s_x\bar{x} \\ 0 & s_y & -s_y\bar{y} \\ 0 & 0 & 1 \end{pmatrix}$

- Solve the homogeneous linear least squares system and transform back to the original coordinate system
- $\mathtt{T}_1$ usually not needed if you work with normalized image points $\tilde{\mathbf{x}}_i = \mathtt{K}^{-1}\mathbf{x}_i$

Improving the Accuracy of Floating Point Computations

- Recall

$$\mathtt{M} = \begin{pmatrix} & & & & & & \vdots & & & & & \\ X_i & Y_i & Z_i & 1 & 0 & 0 & 0 & 0 & -x_iX_i & -x_iY_i & -x_iZ_i & -x_i \\ 0 & 0 & 0 & 0 & X_i & Y_i & Z_i & 1 & -y_iX_i & -y_iY_i & -y_iZ_i & -y_i \\ & & & & & & \vdots & & & & & \end{pmatrix}$$

- Contains 1s, coordinate values (X_i, x_i, etc) and products (e.g. x_iX_i)
  - $x_i = 1000$ pixel and $X_i = 100\,m$: $x_iX_i = 100000 \gg 1$
  - Ideally all non-zero values have similar magnitude

Normalizing Transformation

Transform image points $\mathbf{x}'_i = \mathtt{T}_1\mathbf{x}$ and 3D points $\mathbf{X}'_i = \mathtt{T}_2\mathbf{X}_i$ such that

$$\frac{1}{N}\sum \mathbf{x}'_i = \mathbf{0} \qquad \frac{1}{N}\sum \|\mathbf{x}'_i\|^2 = 1 \qquad \frac{1}{N}\sum \mathbf{X}'_i = \mathbf{0} \qquad \frac{1}{N}\sum \|\mathbf{X}'_i\|^2 = 1$$

$\mathtt{T}_1$ and $\mathtt{T}_2$ are scalings + translations, e.g. $\mathtt{T}_1 = \begin{pmatrix} s_x & 0 & -s_x\bar{x} \\ 0 & s_y & -s_y\bar{y} \\ 0 & 0 & 1 \end{pmatrix}$

- Solve the homogeneous linear least squares system and transform back to the original coordinate system
- $\mathtt{T}_1$ usually not needed if you work with normalized image points $\tilde{\mathbf{x}}_i = \mathtt{K}^{-1}\mathbf{x}_i$

Improving the Accuracy of Floating Point Computations

- Recall

$$\mathtt{M} = \begin{pmatrix} & & & & & & \vdots & & & & & \\ X_i & Y_i & Z_i & 1 & 0 & 0 & 0 & 0 & -x_iX_i & -x_iY_i & -x_iZ_i & -x_i \\ 0 & 0 & 0 & 0 & X_i & Y_i & Z_i & 1 & -y_iX_i & -y_iY_i & -y_iZ_i & -y_i \\ & & & & & & \vdots & & & & & \end{pmatrix}$$

- Contains 1s, coordinate values (X_i, x_i, etc) and products (e.g. x_iX_i)
  - $x_i = 1000$ pixel and $X_i = 100\,m$: $x_iX_i = 100000 \gg 1$
  - Ideally all non-zero values have similar magnitude

Normalizing Transformation

Transform image points $\mathbf{x}'_i = \mathtt{T}_1\mathbf{x}$ and 3D points $\mathbf{X}'_i = \mathtt{T}_2\mathbf{X}_i$ such that

$$\frac{1}{N}\sum \mathbf{x}'_i = \mathbf{0} \qquad \frac{1}{N}\sum \|\mathbf{x}'_i\|^2 = 1 \qquad \frac{1}{N}\sum \mathbf{X}'_i = \mathbf{0} \qquad \frac{1}{N}\sum \|\mathbf{X}'_i\|^2 = 1$$

$\mathtt{T}_1$ and $\mathtt{T}_2$ are scalings + translations, e.g. $\mathtt{T}_1 = \begin{pmatrix} s_x & 0 & -s_x\bar{x} \\ 0 & s_y & -s_y\bar{y} \\ 0 & 0 & 1 \end{pmatrix}$

- Solve the homogeneous linear least squares system and transform back to the original coordinate system
- $\mathtt{T}_1$ usually not needed if you work with normalized image points $\tilde{\mathbf{x}}_i = \mathtt{K}^{-1}\mathbf{x}_i$

Direct Linear Transformation – DLT

Homography estimation

Given point correspondences $\mathbf{x}_i = (x_i, y_i, 1)^\top \leftrightarrow \mathbf{u}_i = (u_i, v_i, 1)^\top$. The initial relations are $\alpha_i \mathbf{u}_i = \mathtt{H}\mathbf{x}_i$, i.e. $\mathbf{u}_i \times (\mathtt{H}\mathbf{x}_i) = 0$. Solve

$$\underbrace{\begin{pmatrix} & & & & \vdots & & & & \\ x_i & y_i & 1 & 0 & 0 & 0 & -u_i x_i & -u_i y_i & -u_i \\ 0 & 0 & 0 & x_i & y_i & 1 & -v_i x_i & -v_i y_i & -v_i \\ & & & & \vdots & & & & \end{pmatrix}}_{=\mathtt{M}} \underbrace{\begin{pmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \mathbf{h}_3 \end{pmatrix}}_{=\mathbf{v}} = \mathbf{0}$$

Homography $\mathtt{H} = \begin{pmatrix} \mathbf{h}_1^\top \\ \mathbf{h}_2^\top \\ \mathbf{h}_3^\top \end{pmatrix}$.

- $2N \times 9$ matrix $\mathtt{M}$
- $N \geq 4$ point correspondences required ($2N$ equations, 8 parameters)

Direct Linear Transformation – DLT

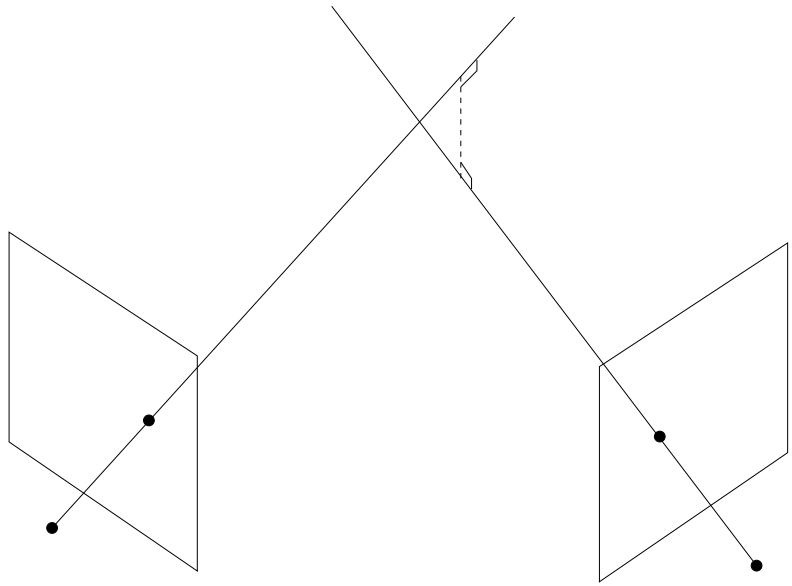
Triangulation

For given camera matrices P_i and image points $\mathbf{x}_i$ compute the 3D scene point $\mathbf{X}$, i.e. solve

$$\lambda_1 \mathbf{x}_1 = \mathrm{P}_1 \mathbf{X} \qquad \cdots \qquad \lambda_N \mathbf{x}_N = \mathrm{P}_N \mathbf{X}$$

for $\mathbf{X} \sim (X_1, X_2, X_3, X_4)^\top$.

- $3N$ equations, $3 + N$ unknowns $\implies 3N \geq 3 + N \implies N \geq 2$ image points



Direct Linear Transformation – DLT

Triangulation: simplified DLT

Find the least squares solution of

$$\begin{aligned}\lambda_1 \mathbf{x}_1 &= \mathtt{P}_1 \mathbf{X} \\ \lambda_2 \mathbf{x}_2 &= \mathtt{P}_2 \mathbf{X} \\ &\vdots\end{aligned}$$

In matrix form:

$$\begin{pmatrix} \mathtt{P}_1 & -\mathbf{x}_1 & 0 & \dots \\ \mathtt{P}_2 & 0 & -\mathbf{x}_2 & \dots \\ \vdots & \vdots & \vdots & \end{pmatrix} \begin{pmatrix} \mathbf{X} \\ \lambda_1 \\ \lambda_2 \\ \vdots \end{pmatrix} = \mathbf{0}$$

- SVD of a $3N \times (4+N)$ matrix
- Homogeneous method. Inhomogeneous method fixes $X_4 = 1$.

Direct Linear Transformation – DLT

Triangulation: standard DLT

Find the least squares solution of

$$\mathbf{x}_1 \times (\mathrm{P}_1\mathbf{X}) = 0 \qquad \cdots \qquad \mathbf{x}_N \times (\mathrm{P}_N\mathbf{X}) = 0$$

In matrix form:

$$\begin{pmatrix} & \vdots & & \\ P_{11} - x_i P_{31} & P_{12} - x_i P_{32} & P_{13} - x_i P_{33} & P_{14} - x_i P_{34} \\ P_{21} - y_i P_{31} & P_{22} - y_i P_{32} & P_{23} - y_i P_{33} & P_{24} - y_i P_{34} \\ & \vdots & & \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{pmatrix} = \mathbf{0}$$

- SVD of a $2N \times 4$ matrix
- Homogeneous method. Inhomogeneous method fixes $X_4 = 1$.

Direct Linear Transformation – DLT

Discussion

- Is the standard or the simplified DLT preferable?
- Will the answers be the same? Why? Why not?
- Inhomogeneous method: fix the scale by fixing an element, e.g.
  - $X_4 = 1$ (triangulation)
  - $P_{34} = 1$ (camera resectioning)
  - $H_{33} = 1$ (homography estimation)

  What are pros and cons?

Taxonomy

	3D points	Camera matrices	Image points
Camera resectioning & pose estimation	Known	Unknown	Known
Triangulation	Unknown	Known	Known
Homography	Unknown but planar	Unknown	Known
Epipolar geometry, relative pose & SfM†	Unknown	Unknown	Known

†: coming up next week

Panoramas

- Panoramas are a special instance of homographies, when camera centers coincide
- W.l.o.g. $\mathtt{P}_1 = (\mathtt{I} \mid \mathbf{0})$ and $\mathtt{P}_2 = (\mathtt{R} \mid \mathbf{0})$
- Normalized image point (ray) $\tilde{\mathbf{x}}$ in camera 1 corresponds to $\mathtt{R}\tilde{\mathbf{x}}$ in camera 2

$$\tilde{\mathbf{y}} = \mathtt{R}\tilde{\mathbf{x}}$$

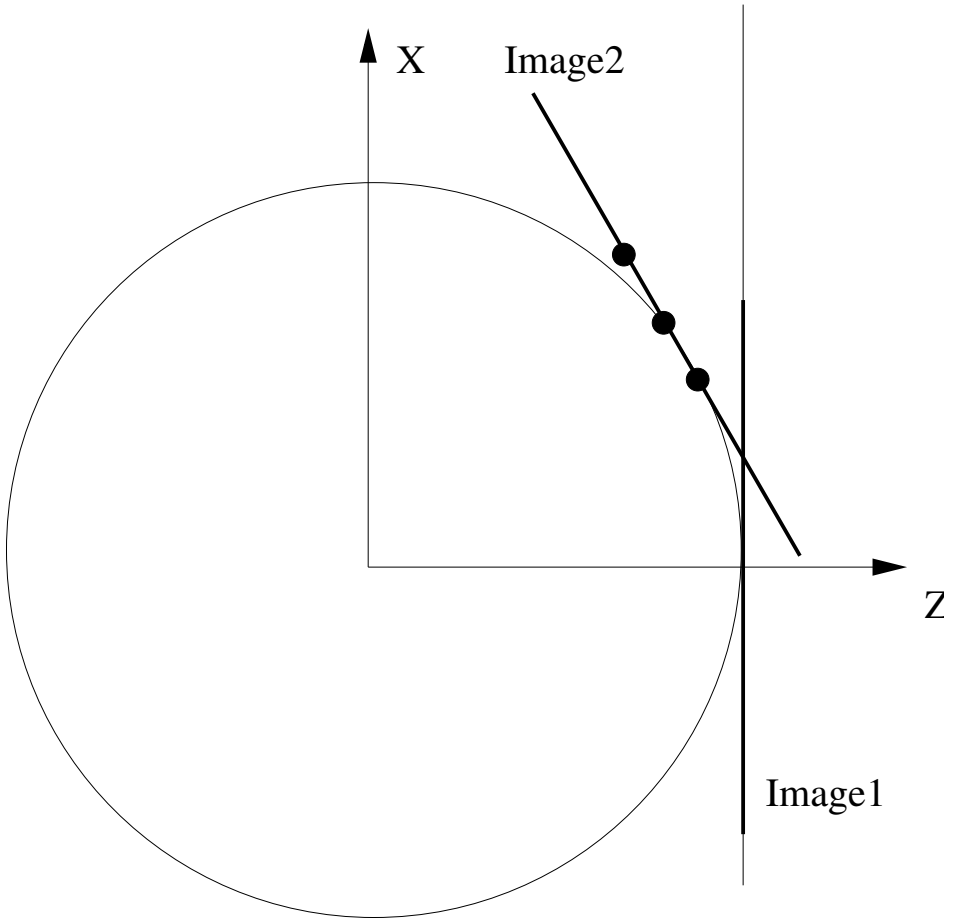
- With calibration matrices

$$\mathbf{y} = \underbrace{\mathtt{K}_2 \, \mathtt{R} \, \mathtt{K}_1^{-1}}_{=\mathtt{H}} \mathbf{x}$$

- $\mathtt{K}_2\mathtt{R}\mathtt{K}_1^{-1}$ is called a "conjugate rotation"

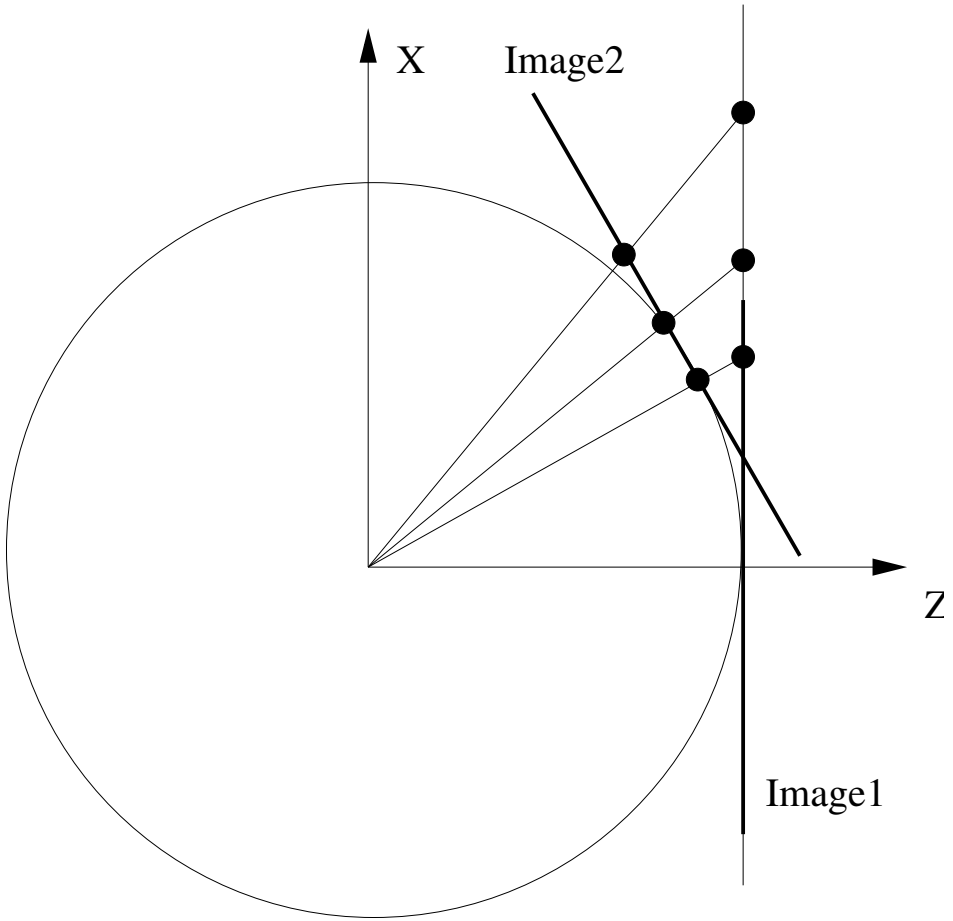
Panoramas

For calibrated cameras:



Panoramas

For calibrated cameras:

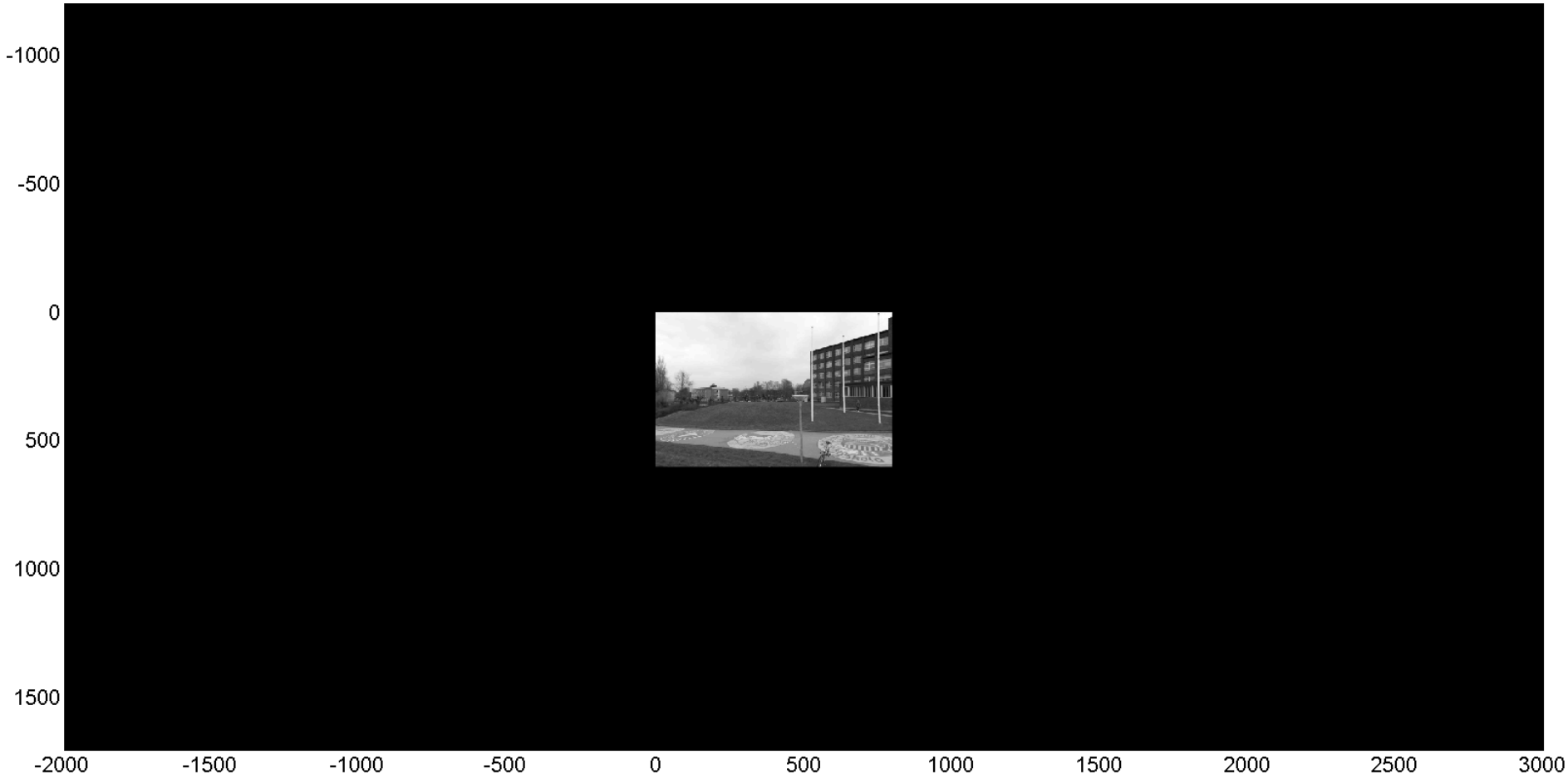


Panoramas

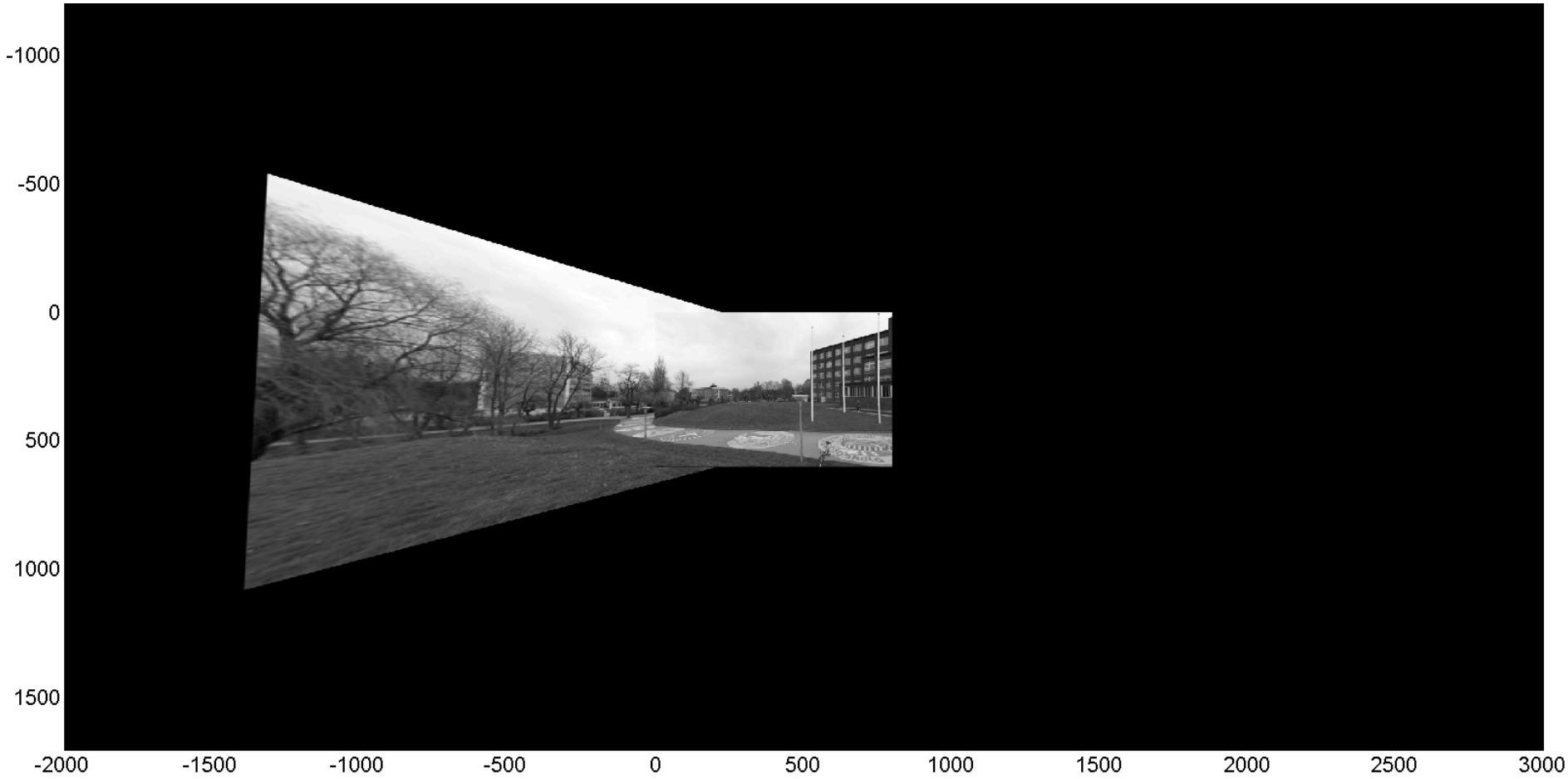





Panoramas

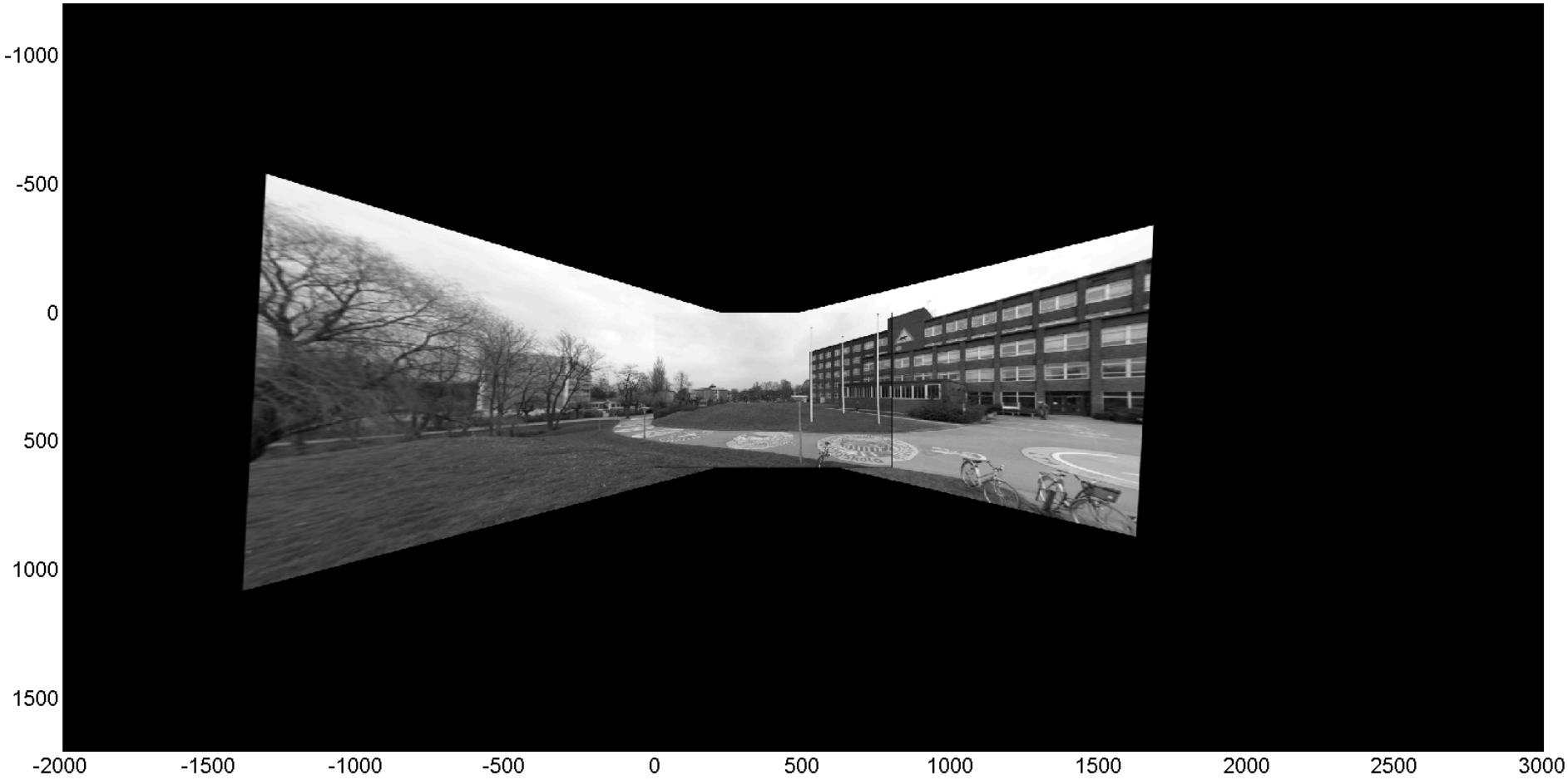

-1000
-500
0
500
1000
1500
-2000
-1500
-1000
-500
0
500
1000
1500
2000
2500
3000

Panoramas



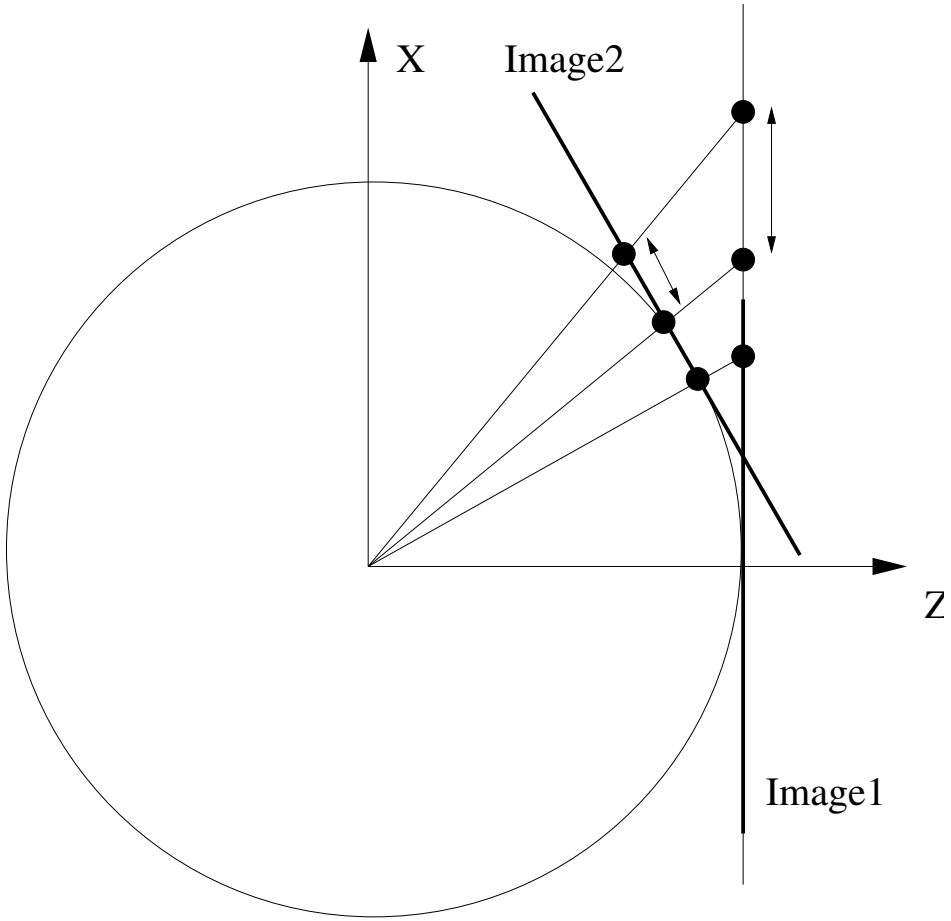
Points are transformed to the first image.

Panoramas


-1000
-500
0
500
1000
1500
-2000
-1500
-1000
-500
0
500
1000
1500
2000
2500
3000

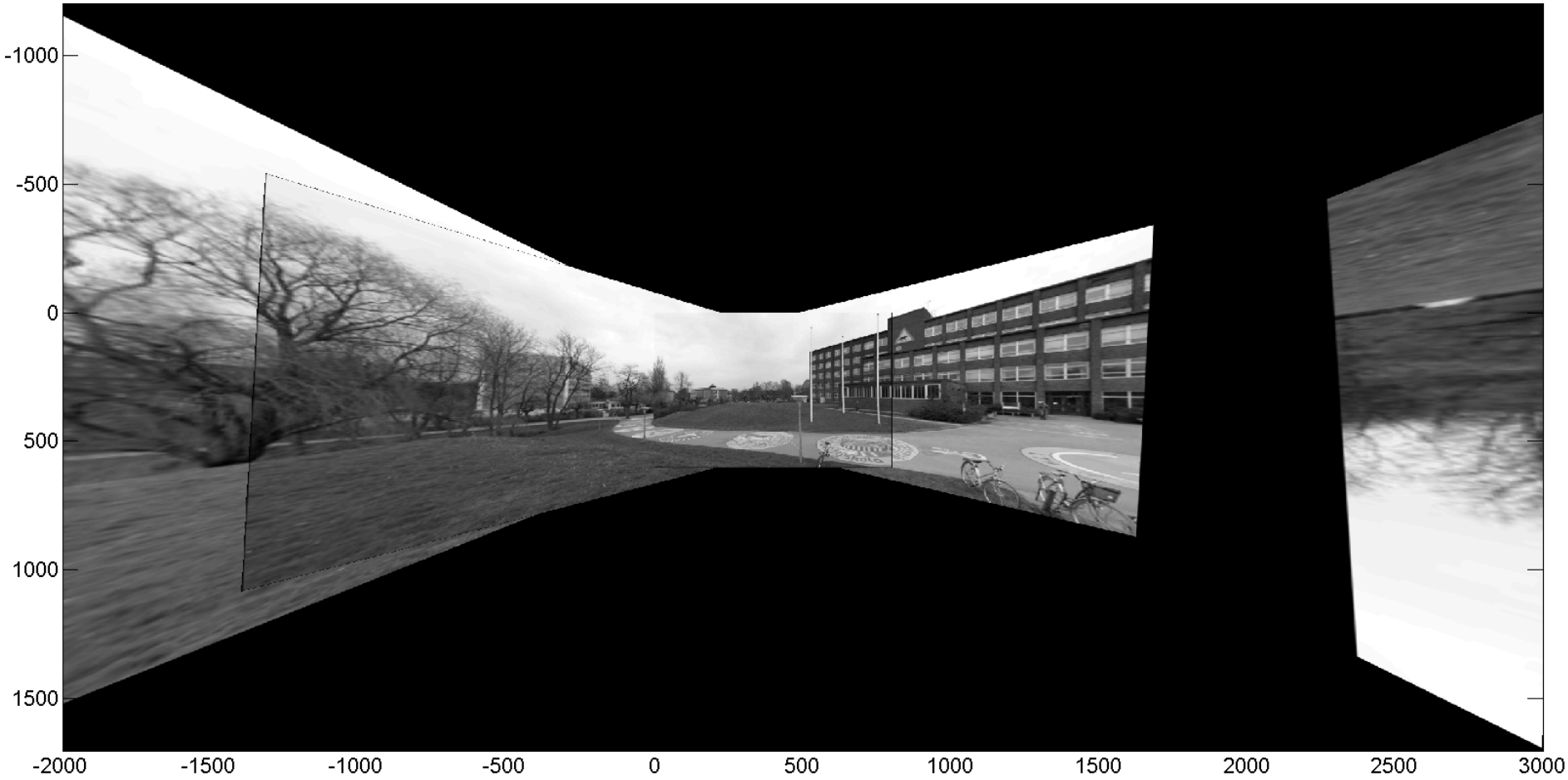
Panoramas

For calibrated cameras:



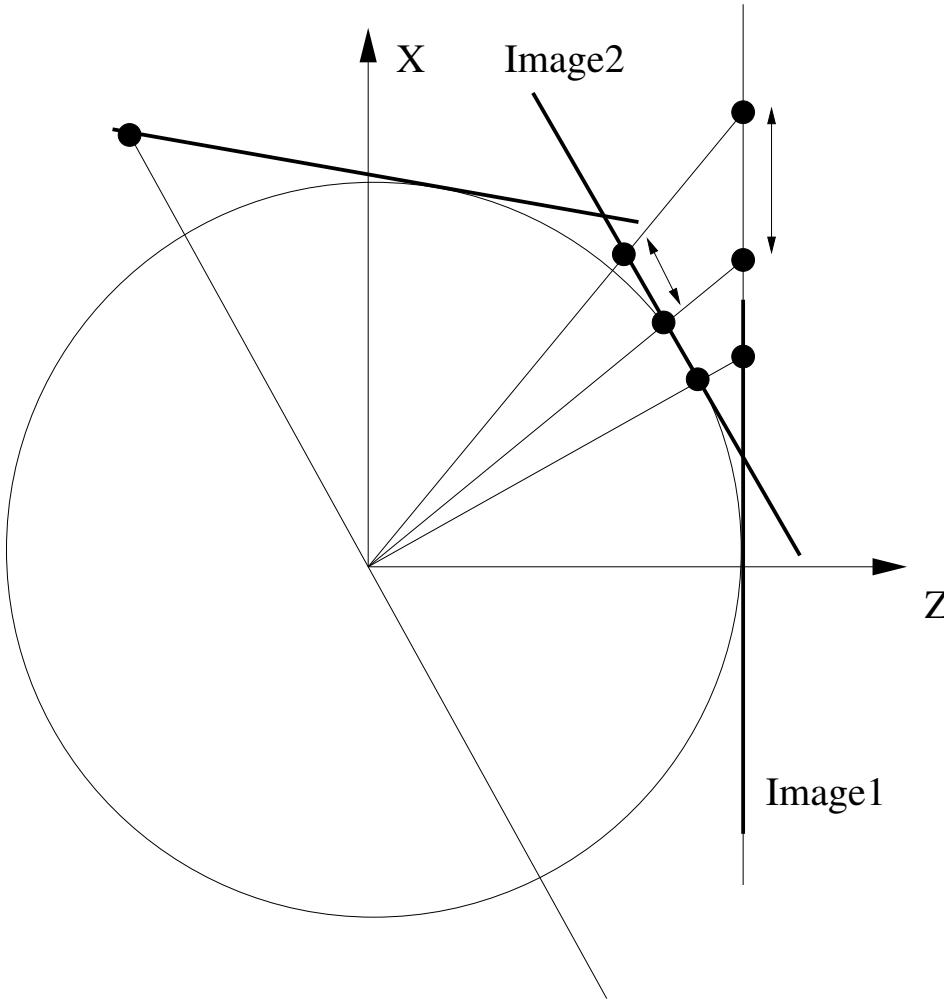
Distances are not preserved. Points close to the x-axis tend to infinity.

Panoramas


-1000
-500
0
500
1000
1500
-2000
-1500
-1000
-500
0
500
1000
1500
2000
2500
3000

Panoramas

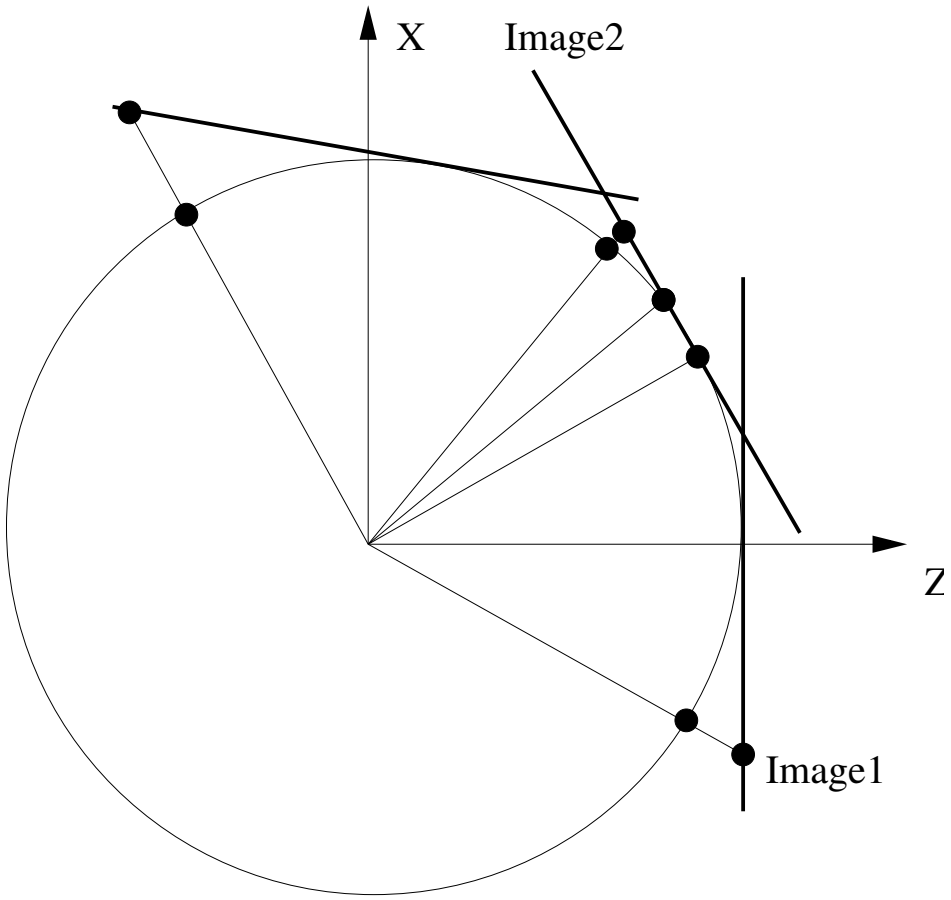
For calibrated cameras:



Cannot transfer all points into the first image.

Panoramas

For calibrated cameras:



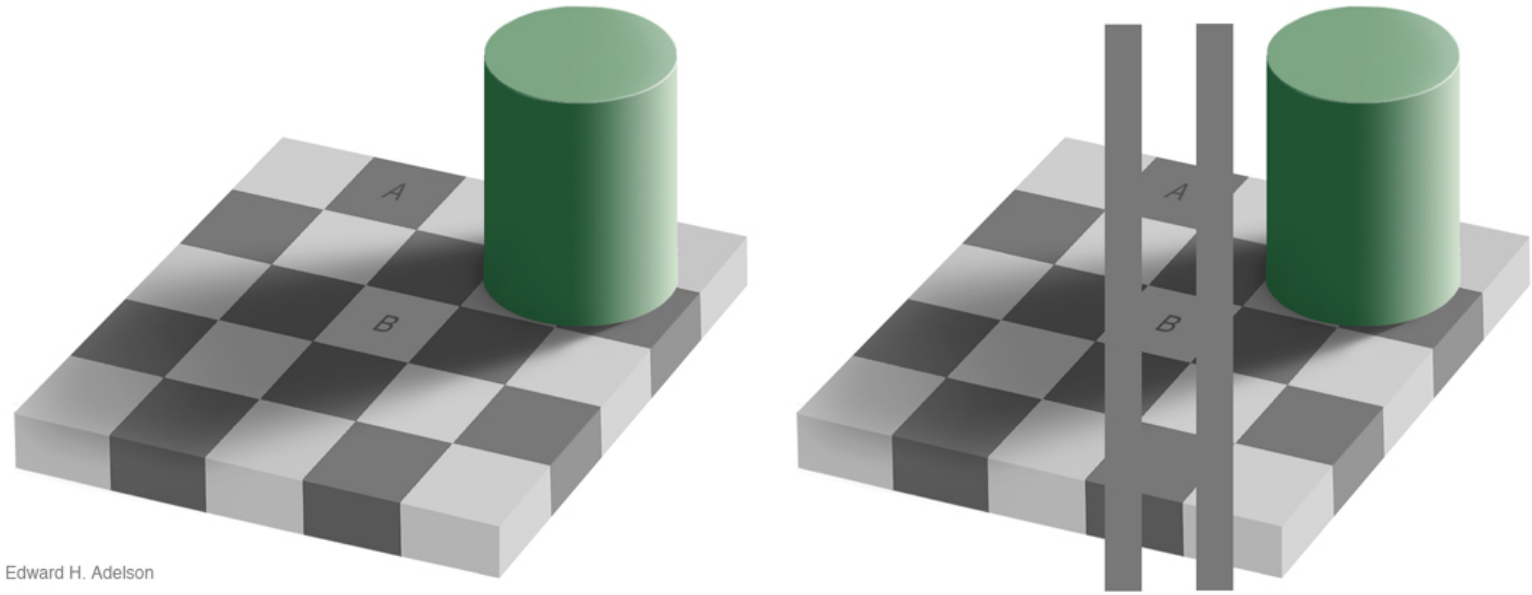
Project onto a cylinder instead.

Panoramas

For calibrated cameras:

Distances are roughly preserved. Lines may not appear straight.

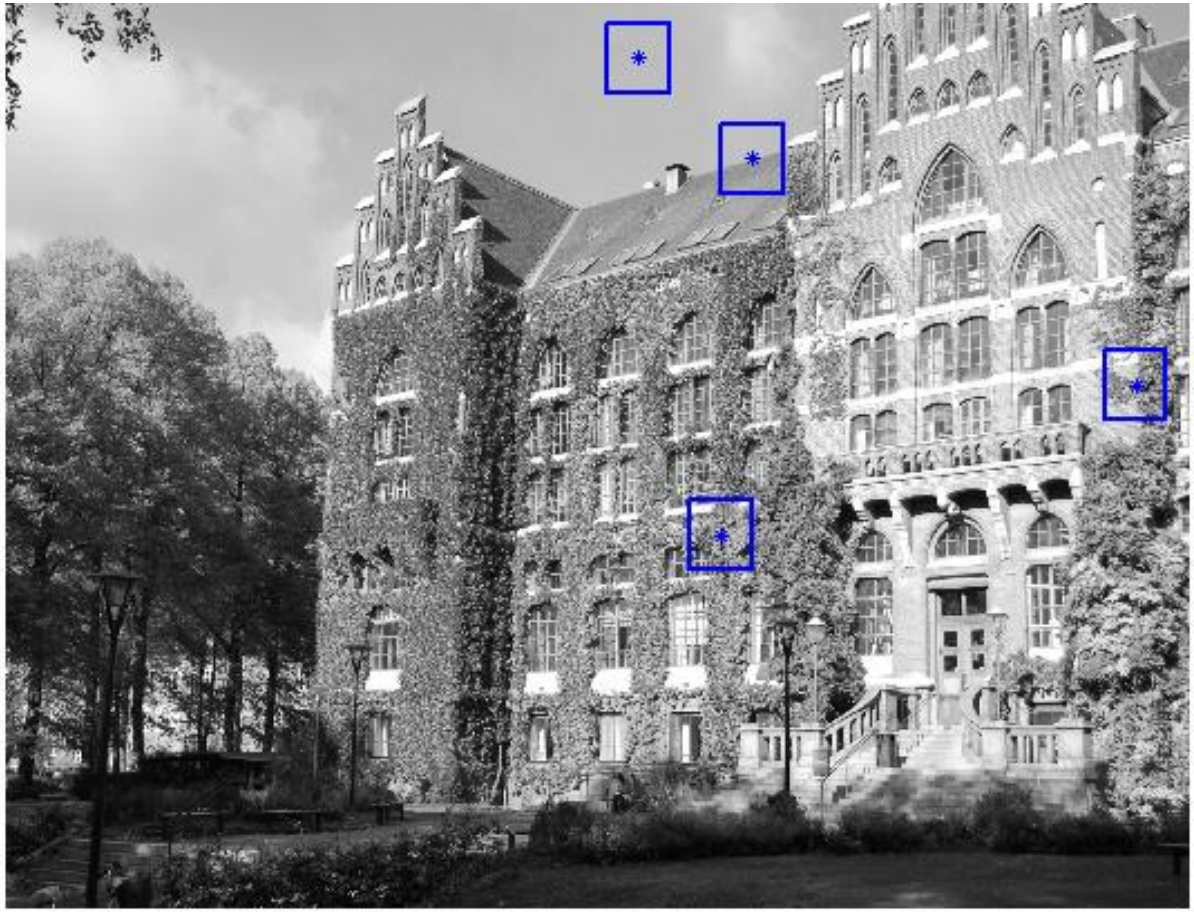
What is a good point?

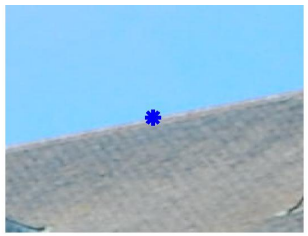


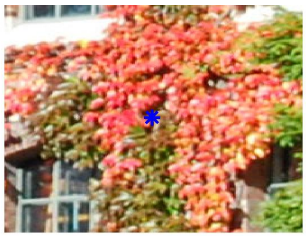
Color alone does not seem reliable. Large contrast changes more reliable.

What is a good point?

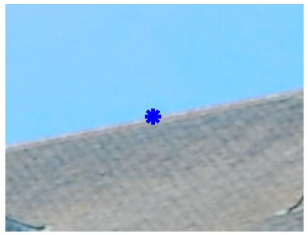
What is a good point?







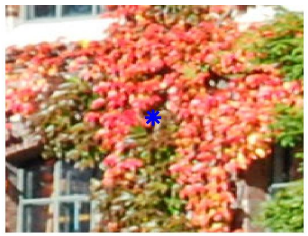
What is a good point?



Can only be localized in one direction.

No contrast, impossible to localize.

Plenty of texture, might work.



Plenty of texture, might work.

Invariance

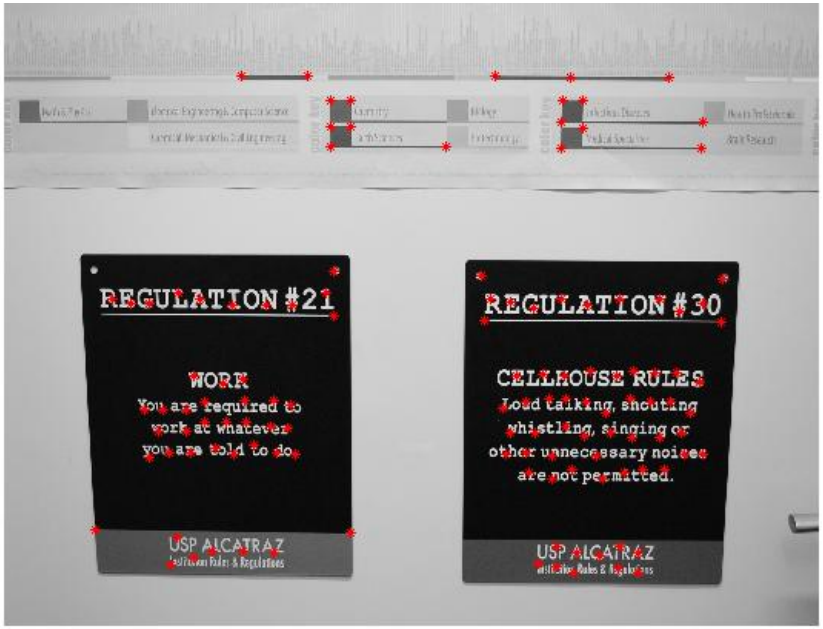
- Image points will appear different due to perspective effects.
- Would like feature detectors that give the same response regardless of view point.

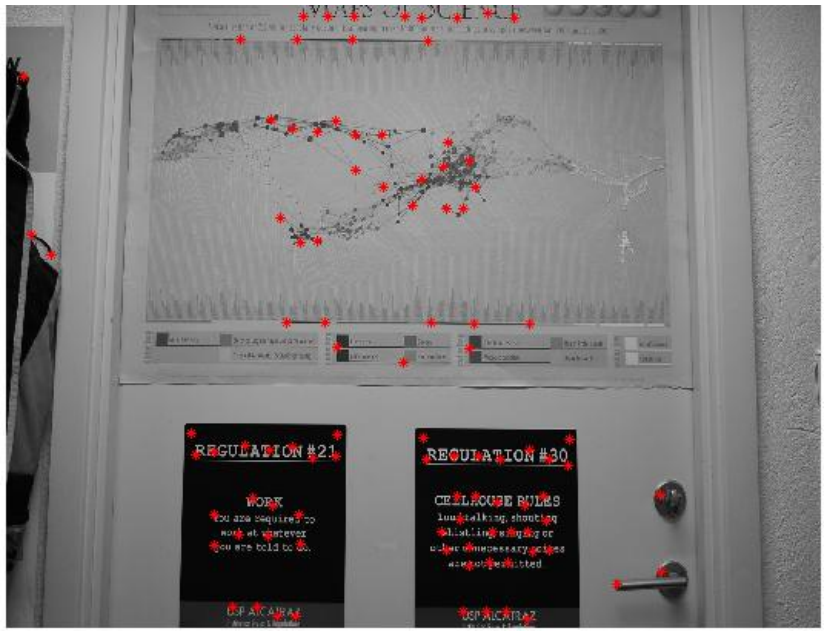
Invariance

The same feature points may not be detected if the scale changes.

Invariance

The same feature points may not be detected if the scale changes.

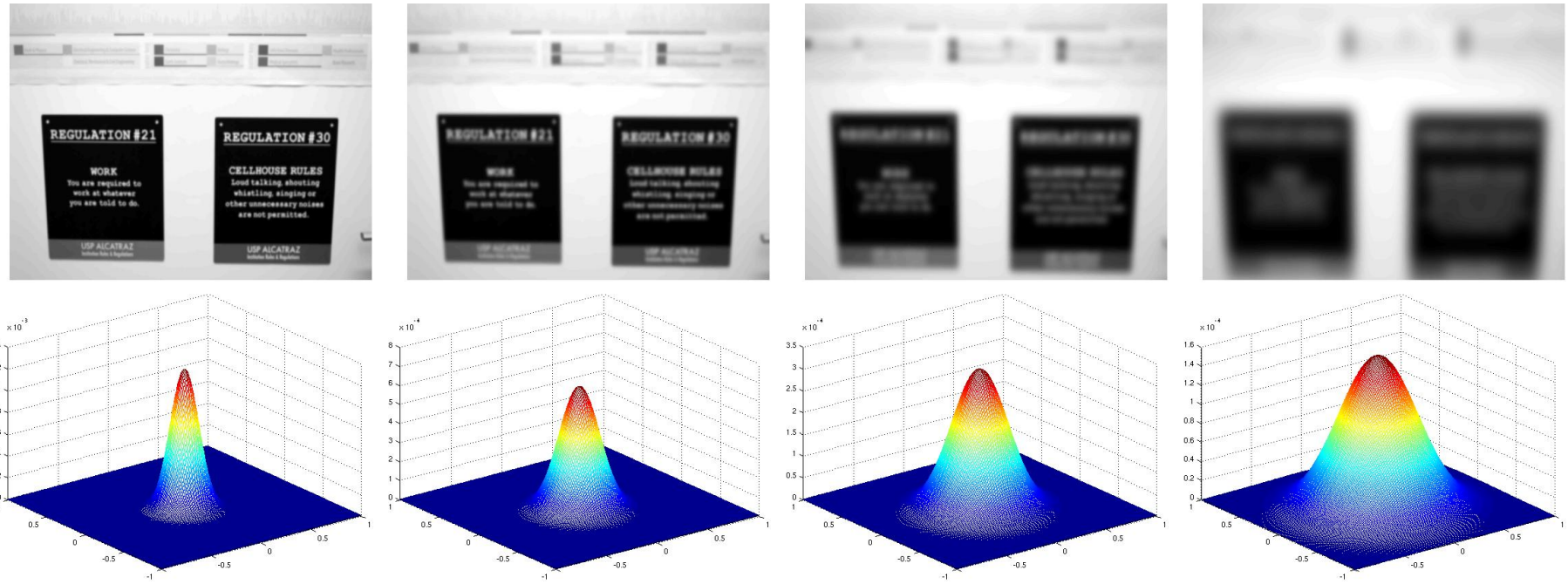




Options:

- Extract features on different scales (different window size).
- Compute points that are not just local maximum in the image, but local maximum in the scale space as well (SIFT).

Scale Selection

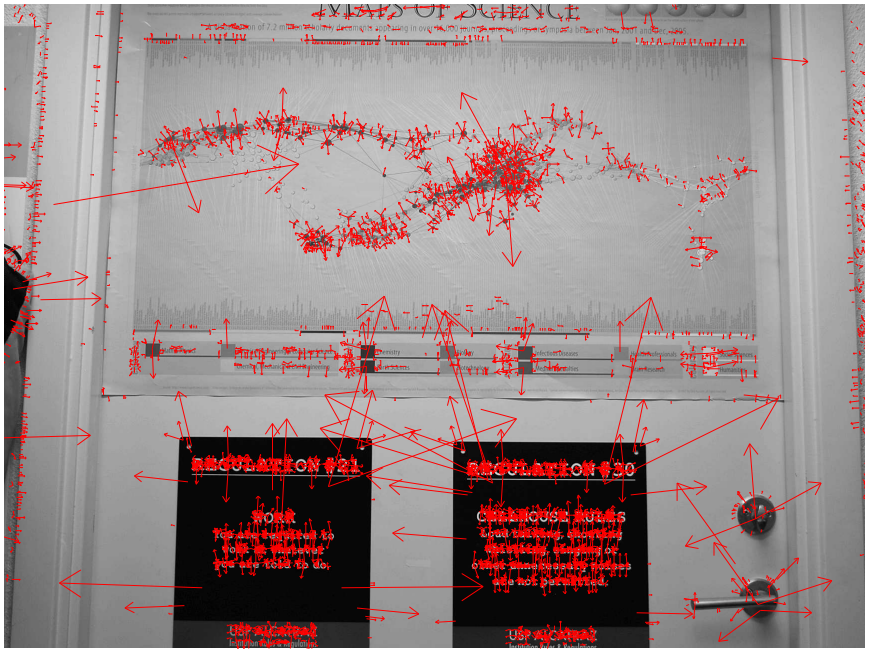


If we are looking for two big black signs, details on the finest scale might not be relevant.
Choose a σ such that the Gaussian

$$G(x, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{||x||^2}{2\sigma^2}}$$

has larger spatial support.

SIFT




REGULATION #21
WORK
You are required to work at whatever you are told to do.
REGULATION #30
CHILDHOUSE RULES

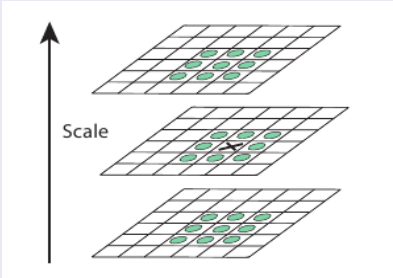
SIFT - Lowe, IJCV 2004

Keypoint detection - Rough Outline

- Compute difference of Gaussians

$$D(x, \sigma) = (G(x, k\sigma) - G(x, \sigma)) * I_0(x)$$

- Detect local maxima and minima of $D(x, \sigma)$ with respect to both x and σ.



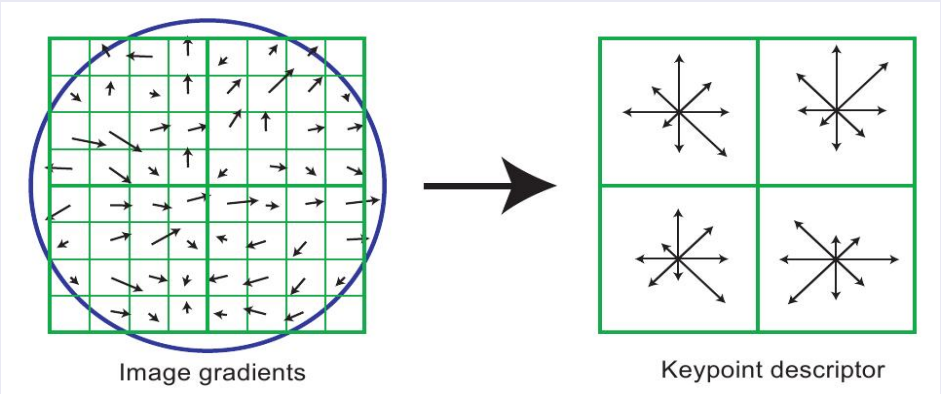
- Accurate keypoint localization (in image space) using Taylor expansion.

Keypoint description

Need to describe the appearance of the patch for matching to other images. The description needs to be robust to appearance changes.

SIFT

Computes gradients around the keypoint (at the appropriate scale and orientation).



The gradient magnitudes are down weighted using a Gaussian and added to a 8-bin histogram.

Keypoint matching

- Compare descriptors match the ones that have the most similar descriptors, according to some matching criterion, e.g.

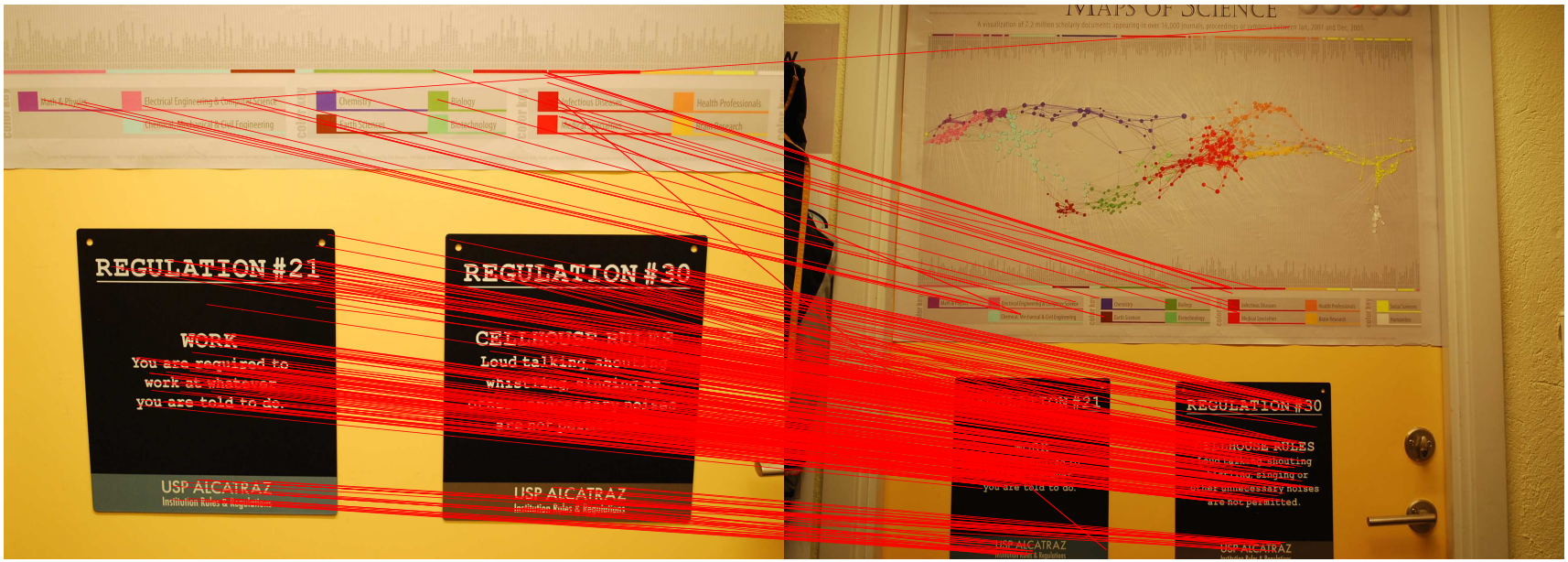
$$m_{ij} = ||d_i - d_j||$$

- Only include keypoints that have a relatively unique descriptor. Compare to the second best match. Use the point if

$$m_{ij} < cm_{ik}, \quad \forall k$$

where $c \in (0, 1)$ is some constant.

Keypoint matching



Discussion

- Why not just describe a keypoint with a patch of RGB values centred at the keypoint location?
- More invariances than scale needed? Pros and cons?
- Learned descriptors?

VLFeat

We rely on existing library for keypoint extraction and matching.
Main calls (in Matlab)

1. `addpath <path>/vlfeat-0.9.21/toolbox/`
   `vl_setup`
2. `[keypoints{i}, descriptors{i}] = vl_sift(single(rgb2gray(imgs{i})), "PeakThresh", 1);`
3. `[matches, scores] = vl_ubcmatch(descriptors{i}, descriptors{j});`

To do

- Work on assignment 2 / hand in assignment 1
- Lab session after this lecture: IDE-D2505-7, MT12–MTI4

More reading:

- Szeliski, Section 11.2 on Pose Estimation and Triangulation.
- Szeliski, Section 11.3 on Two-View Structure from Motion.