Outline

1. The Direct Linear Transformation Algorithm
2. Error Functions
3. Transformation Invariance and Normalization
4. Iterative Minimization Methods
5. Robust Estimation

Objects to estimate from data

- a 2D projectivity
- a 3D to 2D projection (camera)
- the Fundamental Matrix
- the Trifocal Tensor

Data:
- $N$ pairs $x_n, x'_n$ of corresponding points in two images ($n = 1, \ldots, N$)

Note: The Trifocal Tensor represents a relation between three images and thus requires $N$ triples of corresponding points $x_n, x'_n, x''_n$ in three images ($n = 1, \ldots, N$).
Outline

1. The Direct Linear Transformation Algorithm
2. Error Functions
3. Transformation Invariance and Normalization
4. Iterative Minimization Methods
5. Robust Estimation

From Corresponding Points to Linear Equations (1/2)

Inhomogeneous coordinates:

\[ x'_n = \hat{x}'_n := Hx_n, \quad n = 1, \ldots, N \]
\[ = \begin{pmatrix} x_n^T & 0^T & 0^T \\ 0^T & x_n^T & 0^T \\ 0^T & 0^T & x_n^T \end{pmatrix} h, \quad h := \text{vect}(H) := \begin{pmatrix} H_{1,1} \\ H_{1,2} \\ H_{1,3} \\ \vdots \\ H_{3,3} \end{pmatrix} \]

Homogeneous coordinates:

\[ x'_{n,i} : x'_{n,j} = \hat{x}'_{n,i} : \hat{x}'_{n,j}, \quad \forall i, j \in \{1, 2, 3\}, i \neq j \]
\[ x'_{n,i} \hat{x}'_{n,j} - x'_{n,j} \hat{x}'_{n,i} = 0, \quad \text{and one equation is linear dependent} \]
\[ \rightarrow x'_n = \begin{pmatrix} 0^T & -x'_{n,3}x_n^T & x'_{n,2}x_n^T \\ x'_{n,3}x_n^T & 0^T & -x'_{n,1}x_n^T \end{pmatrix} h \]
\[ =: A(x_n, x'_n) \]
From Corresponding Points to Linear Equations (2/2)

\[ A(x_n, x'_n)h \overset{!}{=} 0, \quad n = 1, \ldots, N \]

\[
\begin{pmatrix}
A(x_1, x'_1) \\
A(x_2, x'_2) \\
\vdots \\
A(x_N, x'_N)
\end{pmatrix} h = 0
\]

\[ =: A(x, x') \]

- to estimate a general projectivity we need 4 points (8 equations, 8 dof)
- we are looking for non-trivial solutions \( h \neq 0 \).

More than 4 Points & Noise: Overdetermined

- For \( N > 4 \) points and **exact coordinates**, the system \( Ah = 0 \) still has rank 8 and a non-trivial solution \( h \neq 0 \).
- But for \( N > 4 \) points and **noisy coordinates**, the system \( Ah = 0 \) is overdetermined and (in general) has only the trivial solution \( h = 0 \).

Relax the objective \( Ah = 0 \) to

\[
\arg\min_{h: \|h\| = 1} \|Ah\| = \arg\min_h \frac{\|Ah\|}{\|h\|} = (\text{normed}) \text{ eigenvector to smallest eigenvalue}
\]

and solve via SVD:

\[ A^T A = USU^T, \quad S = \text{diag}(s_1, \ldots, s_9), \quad s_i \geq s_{i+1} \forall i, \quad UU^T = I \]

\[ h := U_9. \]
Degenerate Configurations: Underdetermined

- If three of the four points are collinear (in both images), $A$ will have rank < 8 and thus $h$ underdetermined, and thus there is no unique solution for $h$.

**Degenerate Configuration:**
Corresponding points that do not uniquely determine a transformation (in a particular class of transformations).

---

### Direct Linear Transformation Algorithm (DLT)

1: **procedure**

   `EST-2D-PROJECTIVITY-DLT(x_1, x'_1, x_2, x'_2, \ldots, x_N, x'_N \in \mathbb{P}^2)`

2: $A := \begin{pmatrix} A(x_1, x'_1) \\ A(x_2, x'_2) \\ \vdots \\ A(x_N, x'_N) \end{pmatrix} = \begin{pmatrix} 0^T & -x'_{1,3}x_1^T & x'_{1,2}x_1^T \\ x'_{1,3}x_1^T & 0^T & -x'_{1,1}x_1^T \\ x'_{1,2}x_1^T & 0^T & x'_{2,1}x_2^T \\ \vdots & \vdots & \vdots \\ x'_{N,3}x_N^T & 0^T & -x'_{N,1}x_N^T \end{pmatrix}$

3: $(U, S) := \text{SVD}(A^T A)$

4: $h := U_9,$

5: **return** $H := \begin{pmatrix} h_{1:3} \\ h_{4:6} \\ h_{7:9} \end{pmatrix}$

Note: Do not use this unnormalized version of DLT, but the one in section 3.
Outline

1. The Direct Linear Transformation Algorithm
2. Error Functions
3. Transformation Invariance and Normalization
4. Iterative Minimization Methods
5. Robust Estimation

Algebraic Distance

- the loss minimized by DLT, represented as distance between
  - $x'$: point in 2nd image
  - $\hat{x}' := Hx$: estimated position of $x'$ by $H$

$$
\ell_{\text{alg}}(H; x, x') := ||A(x', x)h||^2
= ||\begin{pmatrix} 0^T & -x'_3 x^T & x'_2 x^T \\ x'_3 x^T & 0^T & -x'_1 x^T \end{pmatrix} h||^2
= ||\begin{pmatrix} -x'_3 \hat{x}'_2 + x'_2 \hat{x}'_3 \\ x'_3 \hat{x}'_1 - x'_1 \hat{x}'_3 \end{pmatrix}||^2
= d_{\text{alg}}(x', \hat{x}')$$

with

$$
d_{\text{alg}}(x, y) := \sqrt{a_1^2 + a_2^2}, \quad (a_1, a_2, a_3)^T = x \times y$$
Geometric Distances: Transfer Errors

Transfer Error in One Image (2nd image):

\[ \ell_{\text{trans1}}(H; x, x') := d(x', Hx)^2 = d(x', \hat{x}')^2 \]

with Euclidean distance in inhomogeneous coordinates

\[ d(x, y) := \sqrt{(x_1/x_3 - y_1/y_3)^2 + (x_2/x_3 - y_2/y_3)^2} \]

\[ = \sqrt{1/(x_3y_3)} \cdot d_{\text{alg}}(x, y) \]

- DLT/algebraic error equals geometric error for affine transformations \((x_3 = y_3 = 1)\)

Symmetric Transfer Error:

\[ \ell_{\text{strans}}(H; x, x') := d(x, H^{-1}x')^2 + d(x', Hx)^2 \]

\[ = d(x, \hat{x})^2 + d(x', \hat{x}')^2, \quad \hat{x} := H^{-1}x' \]

Transfer Errors: Probabilistic Interpretation

Assume

- measurements \(x_n\) in the 1st image are noise-free,
- measurements \(x'_n\) in the 2nd image are distributed Gaussian around true values \(Hx_n\):

\[ p(x'_n \mid Hx_n, \sigma^2) = \frac{1}{2\pi\sigma^2} e^{-d(x'_n, Hx_n)^2/(2\sigma^2)} \]

log-likelihood for Transfer Error in One Image:

\[ p(H \mid x_{1:N}, x'_{1:N}) = \frac{p(x_{1:N}, x'_{1:N} \mid H)p(H)}{p(x_{1:N}, x'_{1:N})} \]

Bayes

\[ \propto p(x_{1:N}, x'_{1:N} \mid H)p(H) \propto p(x'_{1:N} \mid H, x_{1:N})p(H) \]

\[ = p(H) \prod_{n=1}^{N} p(x'_n \mid H, x_n) \propto \prod_{n=1}^{N} p(x'_n \mid H, x_n) \]

\[ \log p(H \mid x_{1:N}, x'_{1:N}) \propto - \sum_{n=1}^{N} d(x'_n, Hx_n)^2 \]

= transfer error
Reprojection Error

- additionally to projectivity $H$, also find noise-free / perfectly matching pairs $\hat{x}, \hat{x}':$

$$\text{minimize } \ell_{\text{rep}}(H, \hat{x}_1, \hat{x}_1', \ldots, \hat{x}_N, \hat{x}_N') := \sum_{n=1}^{N} d(x_n, \hat{x}_n)^2 + d(x'_n, \hat{x}'_n)^2$$

w.r.t.

$$\hat{x}'_n = H\hat{x}_n, \quad n = 1, \ldots, N$$

over

$$H, \hat{x}_1, \hat{x}_1', \ldots, \hat{x}_N, \hat{x}_N'$$

**Reprojection Error:**

$$\ell_{\text{rep}}(H, \hat{x}, \hat{x}'; x, x') := d(x, \hat{x})^2 + d(x', \hat{x}')^2,$$

with $\hat{x}' = H\hat{x}$

- analogue probabilistic interpretation:
  - measurements $x, x'$ are Gaussian around true values $\hat{x}, \hat{x}'$

---

**Outline**

1. The Direct Linear Transformation Algorithm
2. Error Functions
3. Transformation Invariance and Normalization
4. Iterative Minimization Methods
5. Robust Estimation
Are Solutions Invariant under Transformations?

- Given corresponding points \( x_n, x'_n \), a method such as DLT will find a projectivity \( H \).
- Now assume
  - the first image is transformed by projectivity \( T \),
  - the second image is transformed by projectivity \( T' \)
before we apply the estimation method.
- Corresponding points now will be \( \tilde{x}_n := Tx_n, \tilde{x}'_n := T'x'_n \).
- Let \( \tilde{H} \) be the projectivity estimated by the method applied to \( \tilde{x}_n, \tilde{x}'_n \).
- Is it guaranteed that \( H \) and \( \tilde{H} \) are “the same” (equivalent) ?

\[
\tilde{H} \overset{?}{=} T'HT^{-1}
\]

- This may depend on the class of projectivities allowed for \( T, T' \).
  - at least invariance under similarities would be useful !

DLT is not Invariant under Similarities

- If \( T' \) is a similarity transformation with scale factor \( s \) and \( T \) any projectivity, then one can show

\[
||\tilde{A}\tilde{h}|| = s||Ah||
\]

- But solutions \( H \) and \( \tilde{H} \) will not be equivalent nevertheless, as DLT minimizes under constraint \( ||h|| = 1 \)
  - and this constraint is not scaled with \( s \) !
- So DLT is not invariant under similarity transforms.

Note: \( \tilde{A} := A(\tilde{x}, \tilde{x}'), \tilde{h} := \text{vect}(\tilde{H}) \)
Transfer/Reprojection Errors are Invariant under Similarities

- If $T'$ is Euclidean:
  \[
  d(\tilde{x}'_n, \tilde{H}\tilde{x}_n)^2 = d(T'x'_n, T'HT^{-1}Tx_n)^2 \\
  = x'_n T'T'HT^{-1}Tx_n = x'_n Hx_n = d(x'_n, Hx_n)^2
  \]
- If $T'$ is a similarity with scale factor $s$:
  \[
  d(\tilde{x}'_n, \tilde{H}\tilde{x}_n)^2 = d(T'x'_n, T'HT^{-1}Tx_n)^2 \\
  = x'_n T'T'HT^{-1}Tx_n = x'_n s^2 Hx_n = s^2 d(x'_n, Hx_n)^2
  \]

- Error is just scaled, so attains minimum at same position.
  $\Rightarrow$ Transfer/Reprojection Errors are invariant under similarities.

DLT with Normalization

- Image coordinates of corresponding points are usually finite:
  \[
  x = (x_1, x_2, 1)^T,
  \]
  thus have different scale (100, 100, 1) when measured in pixels.
- Therefore, entries in $A(x, x')$ will have largely different scale:
  \[
  A(x, x') = \begin{pmatrix}
  0^T & -x'_3 x^T & x'_2 x^T \\
  x'_3 x^T & 0^T & -x'_1 x^T \\
  0^T & x^T & x'_2 x^T \\
  -x'_3 x^T & 0^T & -x'_1 x^T \\
  \end{pmatrix} = \begin{pmatrix}
  0^T & -x^T & x'_2 x^T \\
  x^T & 0^T & -x'_1 x^T \\
  \end{pmatrix}
  \]
  - some in 100s ($x^T$), some in 10.000s ($x'_2 x^T, -x'_1 x^T$)
DLT with Normalization

▶ normalize $x_i$:

$$\tilde{x}_i := \text{normalize}(x_i) := \left(\frac{x_n - \mu(x_i)}{\tau(x_i)/\sqrt{2}}\right)_{n=1,\ldots,N},$$

with

$$\mu(x_i) := \frac{1}{N} \sum_{n=1}^{N} x_n \quad \text{centroid/mean}$$

$$\tau(x_i) := \frac{1}{N} \sum_{n=1}^{N} d(x_n - \mu(x_i), 0) \quad \text{avg. distance to centroid}$$

▶ afterwards:

$$\mu(\tilde{x}_i) = 0, \quad \tau(\tilde{x}_i) = \sqrt{2}$$

▶ Normalization is a similarity transform:

$$T := T_{\text{norm}}(x_i) := \begin{pmatrix} \sqrt{2}/\tau(x_i) & -\mu(x_i)\sqrt{2}/\tau(x_i) \\ 0 & 1 \end{pmatrix}$$

DLT with Normalization / Algorithm

1: **procedure**

   EST-2D-PROJECTIVITY-DLTN($x_1, x'_1, x_2, x'_2, \ldots, x_N, x'_N \in \mathbb{P}^2$)

2: $T := T_{\text{norm}}(x_i) := \begin{pmatrix} \sqrt{2}/\tau(x_i) & -\mu(x_i)\sqrt{2}/\tau(x_i) \\ 0 & 1 \end{pmatrix}$

3: $T' := T_{\text{norm}}(x'_i) := \begin{pmatrix} \sqrt{2}/\tau(x'_i) & -\mu(x'_i)\sqrt{2}/\tau(x'_i) \\ 0 & 1 \end{pmatrix}$

4: $\tilde{x}_n := T x_n \quad \forall n = 1, \ldots, N \quad \triangleright \text{normalize } x_n$

5: $\tilde{x}'_n := T' x'_n \quad \forall n = 1, \ldots, N \quad \triangleright \text{normalize } x'_n$

6: $\tilde{H} := \text{est-2d-projectivity-dlt}(\tilde{x}_1, \tilde{x}'_1, \tilde{x}_2, \tilde{x}'_2, \ldots, \tilde{x}_N, \tilde{x}'_N)$

7: $H := T'^{-1}\tilde{H} T \quad \triangleright \text{unnormalize } \tilde{H}$

8: **return** $H$
Outline

1. The Direct Linear Transformation Algorithm
2. Error Functions
3. Transformation Invariance and Normalization
4. Iterative Minimization Methods
5. Robust Estimation

Types of Problems

- The transformation estimation problem for the intersection distance/loss can be cast into a single linear system of equations (DLTn).
- The transformation estimation problem for the transfer distance/loss as well as for the reconstruction loss is more complicated and has to be handled by an explicit iterative minimization procedure.
Minimization Objectives $f : \mathbb{R}^M \rightarrow \mathbb{R}$

a) transfer distance in one image:

$$\text{minimize } f(H) := \sum_{n=1}^{N} d(x'_n, Hx_n)^2$$

b) symmetric transfer distance:

$$\text{minimize } f(H) := \sum_{n=1}^{N} d(x'_n, Hx_n)^2 + d(x_n, H^{-1}x'_n)^2$$

c) reconstruction loss:

$$\text{minimize } f(H, \hat{x}_{1:N}) := \sum_{n=1}^{N} d(x_n, \hat{x}_n)^2 + d(x'_n, H\hat{x}_n)^2$$

$\triangleright x_n, x'_n$ are constants, $H, \hat{x}_{1:N}$ variables

$\triangleright$ a), b) have $M := 9$ parameters / variables

$\triangleright$ as $H$ as only 8 dof, the objective is slightly overparametrized

$\triangleright$ c) has $M := 2N + 9$ parameters / variables

$\triangleright$ allowing only finite points for $\hat{x}_n$

Objectives of type $f = e^T e \ (1/3)$

All three objectives $f$ are $L_2$ norms of (parametrized) vectors, i.e. can be written as

$$f(x) = e(x)^T e(x), \quad h : \mathbb{R}^M \rightarrow \mathbb{R}^N$$

a) transfer distance in one image:

$$\text{minimize } f(H) := \sum_{n=1}^{N} d(x'_n, Hx_n)^2$$

$$= e(H)^T e(H),$$

$$e(H) := \begin{pmatrix} x'_{1,1}/x'_{1,3} - (Hx_1)_1/(Hx_1)_3 \\
 x'_{1,2}/x'_{1,3} - (Hx_1)_2/(Hx_1)_3 \\
 \vdots \\
 x'_{N,1}/x'_{N,3} - (Hx_N)_1/(Hx_N)_3 \\
 x'_{N,2}/x'_{N,3} - (Hx_N)_2/(Hx_N)_3 \end{pmatrix}$$
Objectives of type $f = e^T e$ (2/3)

b) symmetric transfer distance:

$$
\text{minimize } f(H) := \sum_{n=1}^{N} d(x'_n, Hx_n)^2 + d(x_n, H^{-1}x'_n)^2 = e(H)^T e(H),
$$

$$
e(H) := \begin{pmatrix}
\frac{x'_{1,1}/x'_{1,3} - (Hx_1)_{1}/(Hx_1)_{3}}{x'_{1,2}/x'_{1,3} - (Hx_1)_{2}/(Hx_1)_{3}} \\
\vdots \\
\frac{x'_{N,1}/x'_{N,3} - (Hx_N)_{1}/(Hx_N)_{3}}{x'_{N,2}/x'_{N,3} - (Hx_N)_{2}/(Hx_N)_{3}}
\end{pmatrix}
$$

Objectives of type $f = e^T e$ (3/3)

c) reconstruction loss:

$$
\text{minimize } f(H, \hat{x}_{1:N}) := \sum_{n=1}^{N} d(x_n, \hat{x}_n)^2 + d(x'_n, H\hat{x}_n)^2 = e(H)^T e(H),
$$

$$
e(H) := \begin{pmatrix}
\frac{x'_{1,1}/x'_{1,3} - (H\hat{x}_1)_{1}/(H\hat{x}_1)_{3}}{x'_{1,2}/x'_{1,3} - (H\hat{x}_1)_{2}/(H\hat{x}_1)_{3}} \\
\vdots \\
\frac{x'_{N,1}/x'_{N,3} - (H\hat{x}_N)_{1}/(H\hat{x}_N)_{3}}{x'_{N,2}/x'_{N,3} - (H\hat{x}_N)_{2}/(H\hat{x}_N)_{3}} \\
x_{1,1}/x_{1,3} - \hat{x}_{1,1} \\
x_{1,2}/x_{1,3} - \hat{x}_{1,2} \\
\vdots \\
x_{N,1}/x_{N,3} - \hat{x}_{N,1} \\
x_{N,2}/x_{N,3} - \hat{x}_{N,2}
\end{pmatrix}
$$
Minimizing $f(I)$: Gradient Descent

To minimize $f : \mathbb{R}^M \rightarrow \mathbb{R}$ over $x \in \mathbb{R}^M$ **Gradient Descent**

1. starts at a random **starting point** $x_0 \in \mathbb{R}^M$
   
   $t := 0, \quad x(t) := x_0$

2. computes as **descent direction** $d(t)$ at $x(t)$ — direction where $f$ decreases —
   the gradient of $f$:
   
   $$d(t) := -g(t) := -\nabla_x f|_{x(t)} := -\frac{\partial f(y)}{\partial x_m}(x(t))_{m=1,\ldots,M}$$

3. moves into the descent direction:
   
   $$x(t+1) := x(t) + d$$

Beware:

- $f$ decreases only in the neighborhood of $x(t)$
- A full gradient step may be too large and **not** leading to a decrease!

Minimizing $f(I)$: Gradient Descent w. Steplength Control

To minimize $f : \mathbb{R}^M \rightarrow \mathbb{R}$ over $x \in \mathbb{R}^M$ **Gradient Descent**

1. starts at a random **starting point** $x_0 \in \mathbb{R}^M$
   
   $t := 0, \quad x(t) := x_0$

2. computes as **descent direction** $d(t)$ at $x(t)$ — direction where $f$ decreases —
   the gradient of $f$:
   
   $$d(t) := -g(t) := -\nabla_x f|_{x(t)} := -\frac{\partial f(y)}{\partial x_m}(x(t))_{m=1,\ldots,M}$$

3. finds a steplength $\alpha \in \mathbb{R}^+$ so that $f$ actually decreases:
   
   $$\alpha := \max\{\alpha := 2^{-k} \mid k = 0, 1, 2, \ldots, f(x + \alpha d) < f(x)\}$$

4. moves a step into the descent direction:
   
   $$x(t+1) := x(t) + \alpha d$$
Minimizing $f$ (I): Gradient Descent / Algorithm

1: procedure MIN-GD($f : \mathbb{R}^M \to \mathbb{R}$, $x_0 \in \mathbb{R}^M$, $\nabla_x f : \mathbb{R}^M \to \mathbb{R}^M$, $\epsilon \in \mathbb{R}^+$)
2: $x := x_0$
3: do
4: $d := -\nabla_x f|_x$
5: $\alpha := 1$
6: while $f(x + \alpha d) \geq f(x)$ do
7: $\alpha := \alpha / 2$
8: $x := x + \alpha d$
9: while $||d|| > \epsilon$
10: return $x$

Minimizing $f$ (II): Newton

The Newton algorithm computes a better descent direction:

- approximate $f$ by the quadratic Taylor expansion at $x^{(t)}$:

$$f(x + d) \approx \tilde{f}(d) := f(x^{(t)}) + \nabla_x f|_{x^{(t)}}^T d + \frac{1}{2} d^T \nabla^2_x f|_{x^{(t)}} d$$

$$= f(x^{(t)}) + g_{x^{(t)}}^T d + \frac{1}{2} d^T H_{x^{(t)}} d$$

where

$$\nabla^2_x f|_x := H_x := \left( \frac{\partial^2 f}{\partial x_m \partial x_k} \right)_{m,k=1,...,M} \text{ Hessian of } f$$

- the approximation attains its minimum at

$$0 \equiv \nabla_d \tilde{f}(d) = g_{x^{(t)}} + H_{x^{(t)}} d$$

$$H_{x^{(t)}} d = - g_{x^{(t)}}$$

- solve this linear system of equations to find descent direction
Minimizing $f$ (II): Newton / Algorithm

1: procedure MIN-NEWTON($f : \mathbb{R}^M \rightarrow \mathbb{R}, x_0 \in \mathbb{R}^M$, $\nabla_x f : \mathbb{R}^M \rightarrow \mathbb{R}^M, \nabla^2_x f : \mathbb{R}^M \rightarrow \mathbb{R}^{M \times M}, \epsilon \in \mathbb{R}^+$)
2: $x := x_0$
3: do
4: $g := \nabla_x f|_x$
5: $H := \nabla^2_x f|_x$
6: $d := \text{solve}_d(Hd = -g)$
7: $\alpha := 1$
8: while $f(x + \alpha d) \geq f(x)$ do
9: $\alpha := \alpha/2$
10: $x := x + \alpha d$
11: while $||d|| \geq \epsilon$
12: return $x$

Gauss-Newton is

- a specialization of the Newton algorithm
- for objectives of type $f(x) = e(x)^T e(x)$
- that approximates the Hessian:

$$
\nabla_x f|_x = 2\nabla_x e|_x^T e(x)
$$

$$
\nabla^2_x f|_x = 2\nabla_x e|_x^T \nabla_x e|_x + 2\nabla^2_x e|_x^T e(x)
$$

Now approximate $e$ by a linear Taylor expansion, i.e.

$$
\nabla^2_x e|_x \approx 0
$$

$$
\approx \nabla^2_x f|_x \approx 2\nabla_x e|_x^T \nabla_x e|_x
$$
Minimizing $f = e^T e$ (I): Gauss-Newton / Algorithm

1: **procedure** MIN-GAUSS-NEWTON($f : \mathbb{R}^M \to \mathbb{R}$, $x_0 \in \mathbb{R}^M$, $\nabla_x e : \mathbb{R}^M \to \mathbb{R}^{N \times M}$, $\epsilon \in \mathbb{R}^+$)
2: \hspace{1em} $x := x_0$
3: \hspace{1em} **do**
4: \hspace{2em} $J := \nabla_x e \big|_x$
5: \hspace{2em} $g := J^T e(x)$
6: \hspace{2em} $H := J^T J$
7: \hspace{2em} $d := \text{solve}_d (Hd = -g)$
8: \hspace{2em} $\alpha := 1$
9: \hspace{2em} **while** $f(x + \alpha d) \geq f(x)$ **do**
10: \hspace{3em} $\alpha := \alpha/2$
11: \hspace{2em} $x := x + \alpha d$
12: \hspace{2em} **while** $||d|| > \epsilon$
13: \hspace{2em} **return** $x$

Minimizing $f = e^T e$ (II): Levenberg-Marquardt
Outline

1. The Direct Linear Transformation Algorithm
2. Error Functions
3. Transformation Invariance and Normalization
4. Iterative Minimization Methods
5. Robust Estimation
Outline

1. The Direct Linear Transformation Algorithm
2. Error Functions
3. Transformation Invariance and Normalization
4. Iterative Minimization Methods
5. Robust Estimation
Further Readings

- [HZ04, ch. 4].
- For iterative estimation methods in CV see [HZ04, appendix 6].
- You may also read [HZ04, ch. 5] which will not be covered in the lecture explicitly.
References

Richard Hartley and Andrew Zisserman.  
*Multiple view geometry in computer vision.*  