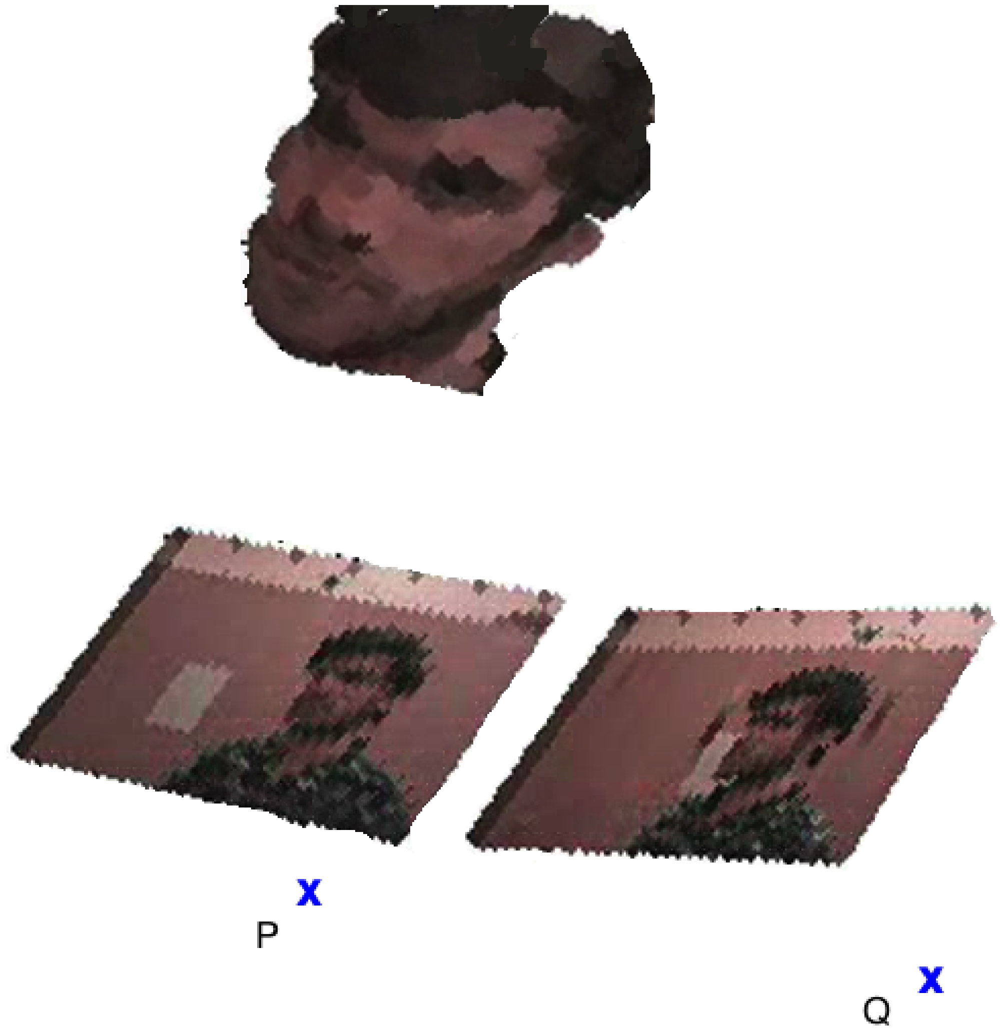
Epipolar Geometry and its application for the construction of state-of-the-art sensors.

Karel Zimmermann

Czech Technical University in Prague
Faculty of Electrical Engineering, Department of Cybernetics
Center for Machine Perception
http://cmp.felk.cvut.cz/~zimmerk, zimmerk@fel.cvut.cz

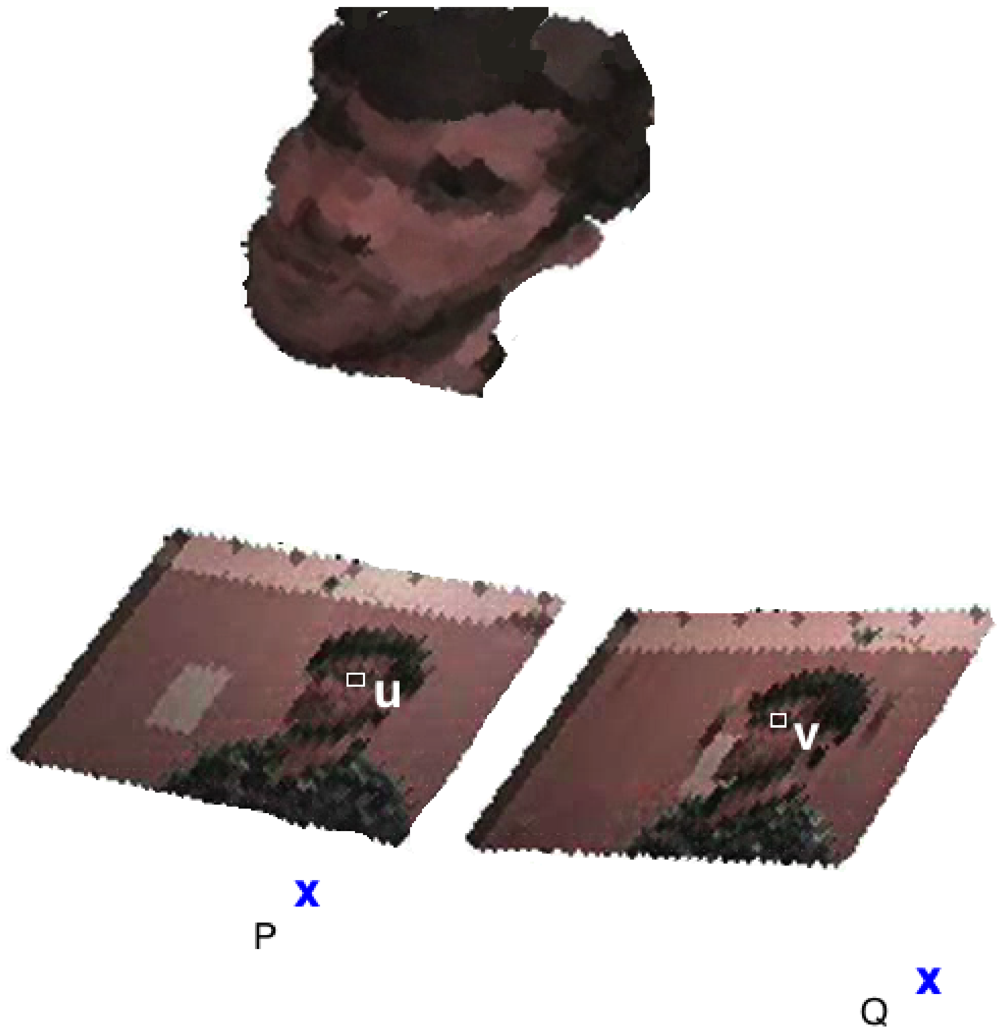
Motivation

- You are given two images of an object captured by two cameras P and Q from different view-points.



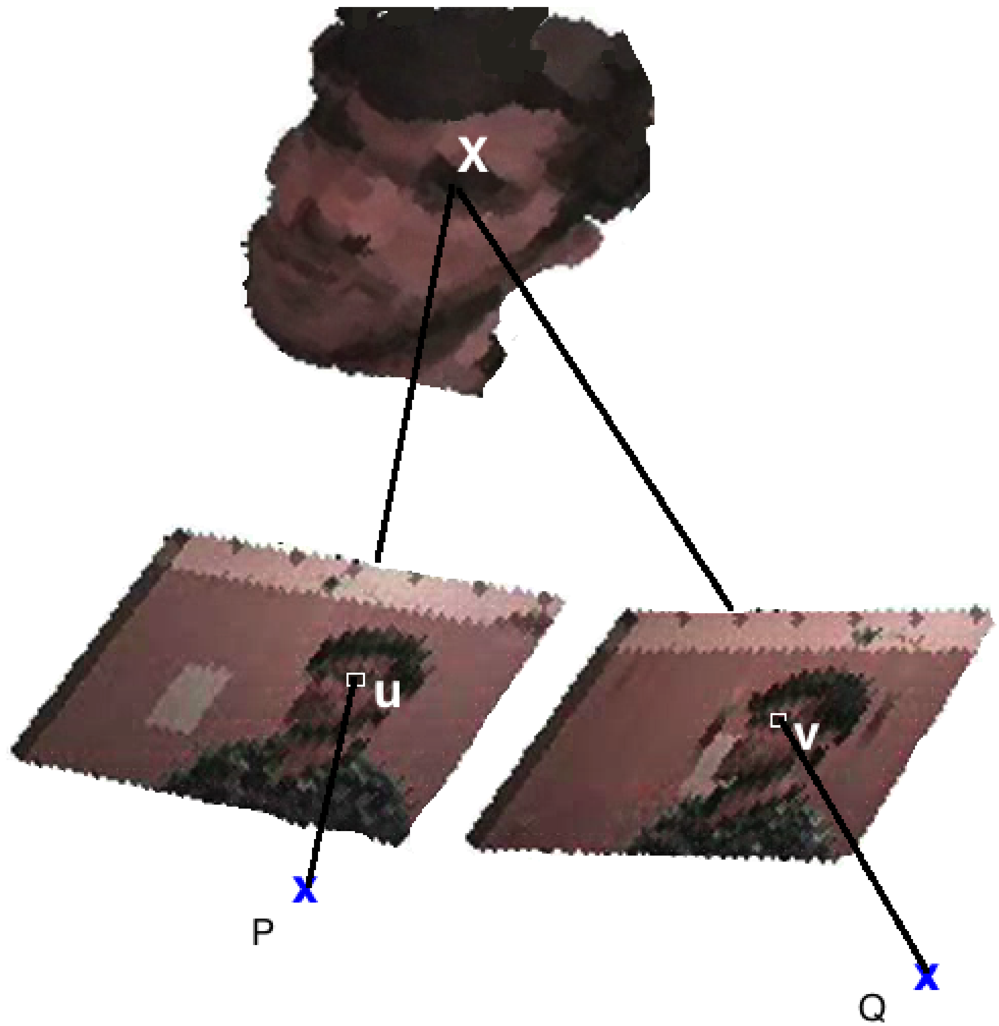
Motivation

- Given pair of corresponding pixels $(\mathbf{u}, \mathbf{v})$ (i.e. pixels corresponding to the same unknown 3D point $\mathbf{X}$ on the object), you can easily compute $\mathbf{X}$.



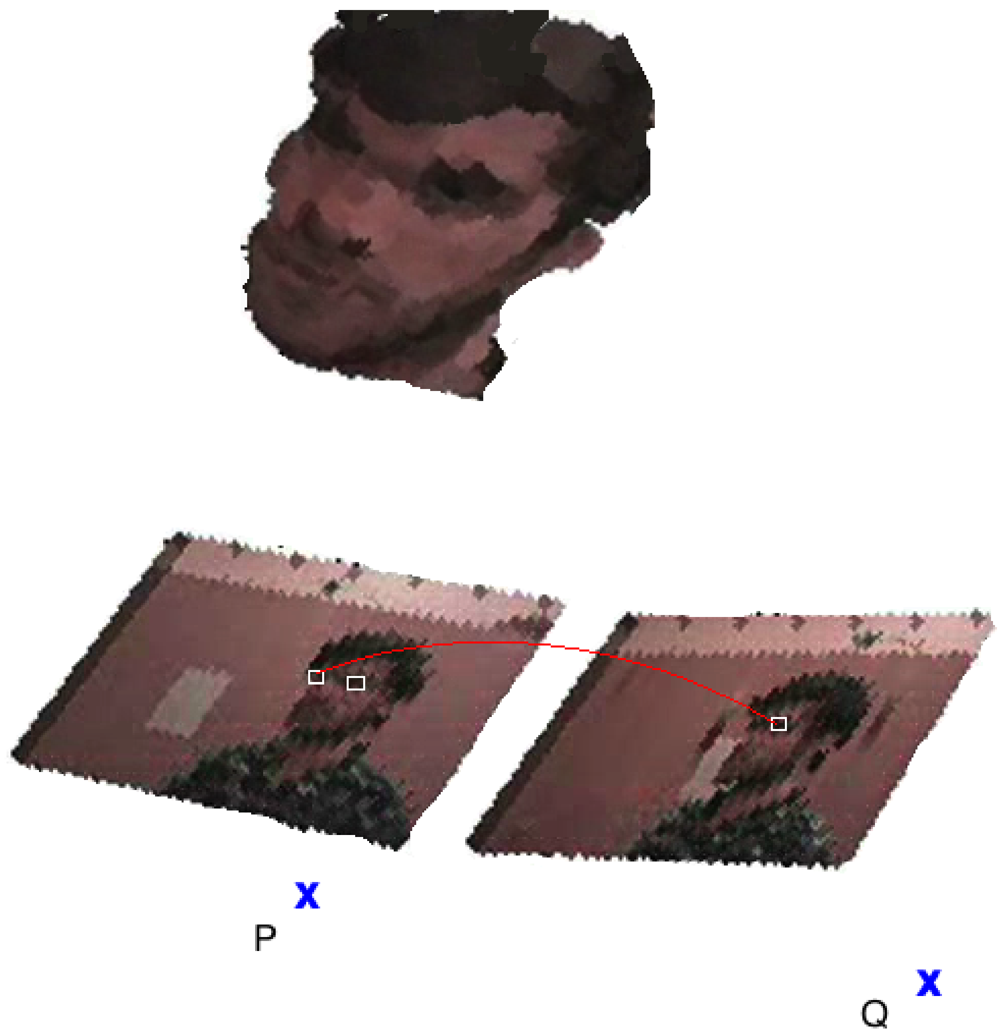
Motivation

- Given pair of corresponding pixels $(\mathbf{u}, \mathbf{v})$ (i.e. pixels corresponding to the same unknown 3D point $\mathbf{X}$ on the object), you can easily compute $\mathbf{X}$.



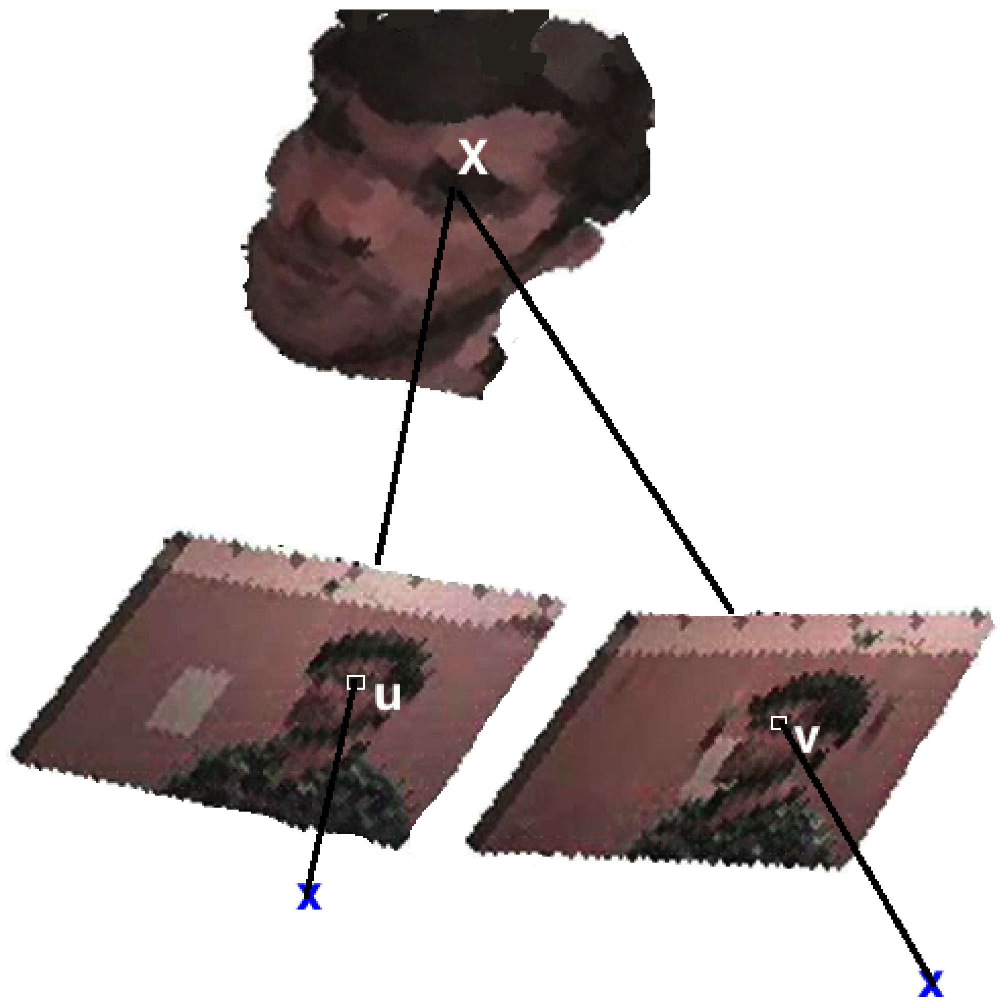
Motivation

- The only problem is, that you do not have the correspondence $(\mathbf{u}, \mathbf{v})$ and naïve matching of pixel neighbourhoods does not work.



Motivation

- This lesson is about
  - how to get 3D points from images captured by known cameras and
  - how to use this knowledge to built state-of-the-art depth sensors.



Outline

- Epipolar geometry
  - Epipolar line, essential and fundamental matrix
  - L_2 estimation of the essential matrix
- Depth sensors: Stereo, Kinect and RealSense
- Depth from a single camera and the robust estimation of the essential matrix (RANSAC).

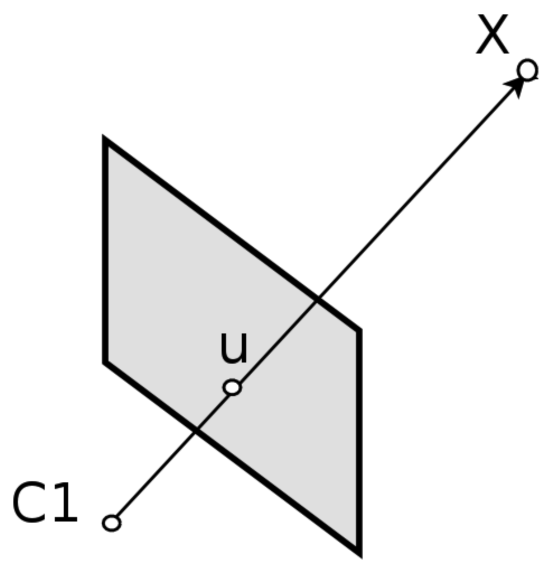
Projection of the 3D point to a single camera

- You are given 3×4 camera matrix $\mathtt{P} = \begin{bmatrix} \mathbf{p}_1^\top \\ \mathbf{p}_2^\top \\ \mathbf{p}_3^\top \end{bmatrix}$
- 3D point with homogeneous coordinates $\mathbf{X}$ projects on pixel $\mathbf{u}$

Projection of the 3D point to a single camera

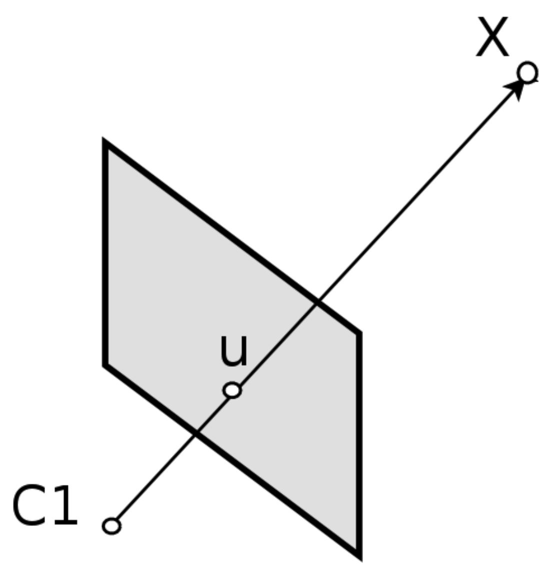
- You are given 3×4 camera matrix $\mathtt{P} = \begin{bmatrix} \mathbf{p}_1^\top \\ \mathbf{p}_2^\top \\ \mathbf{p}_3^\top \end{bmatrix}$
- 3D point with homogeneous coordinates $\mathbf{X}$ projects on pixel $\mathbf{u}$

$$u_1 = \frac{\mathbf{p}_1^\top \mathbf{X}}{\mathbf{p}_3^\top \mathbf{X}}, \qquad u_2 = \frac{\mathbf{p}_2^\top \mathbf{X}}{\mathbf{p}_3^\top \mathbf{X}}$$



Projection of the 3D point to a single camera

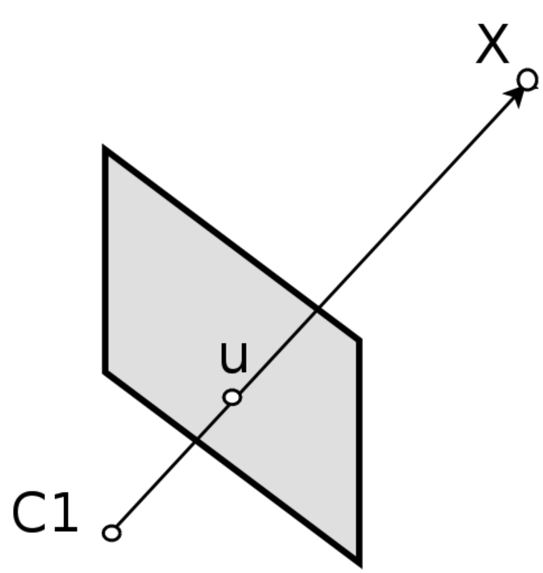
- What if $\mathbf{u}$ is known? Which $\mathbf{X}$ correspond to $\mathbf{u}$?



Projection of the 3D point to a single camera

- What if $\mathbf{u}$ is known? Which $\mathbf{X}$ correspond to $\mathbf{u}$?
- All 3D points corresponding to pixel $\mathbf{u}$ lies in 1D linear subspace (ray) of 3D space (2 linear equations with 3 unknowns):

$$\begin{array}{c} u_1\mathbf{p}_3^\top\mathbf{X} = \mathbf{p}_1^\top\mathbf{X}, \\ u_2\mathbf{p}_3^\top\mathbf{X} = \mathbf{p}_2^\top\mathbf{X} \end{array} \Rightarrow \begin{bmatrix} u_1\mathbf{p}_3^\top - \mathbf{p}_1^\top \\ u_2\mathbf{p}_3^\top - \mathbf{p}_2^\top \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \mathbf{0}$$

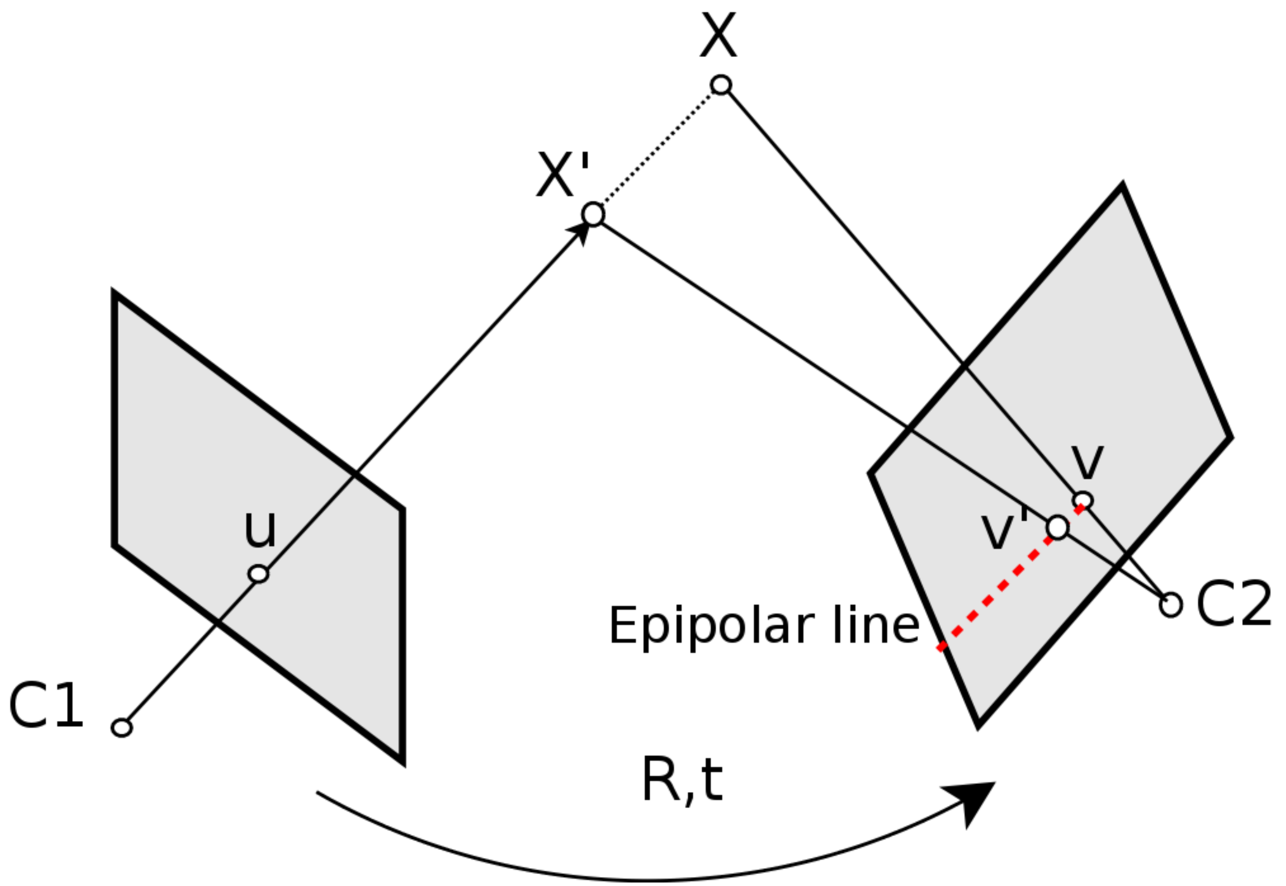


Fundamental matrix

- Projection of the ray from $\mathbf{u}$ into a second camera is called epipolar line

$$\{\mathbf{v} \mid \mathbf{u}^\top \mathtt{F}\,\mathbf{v} = 0\},$$

- where matrix $\mathtt{F} = \mathtt{K}^\top(\mathtt{R} \times \mathbf{t})\mathtt{K}$ is called fundamental matrix.



Essential matrix

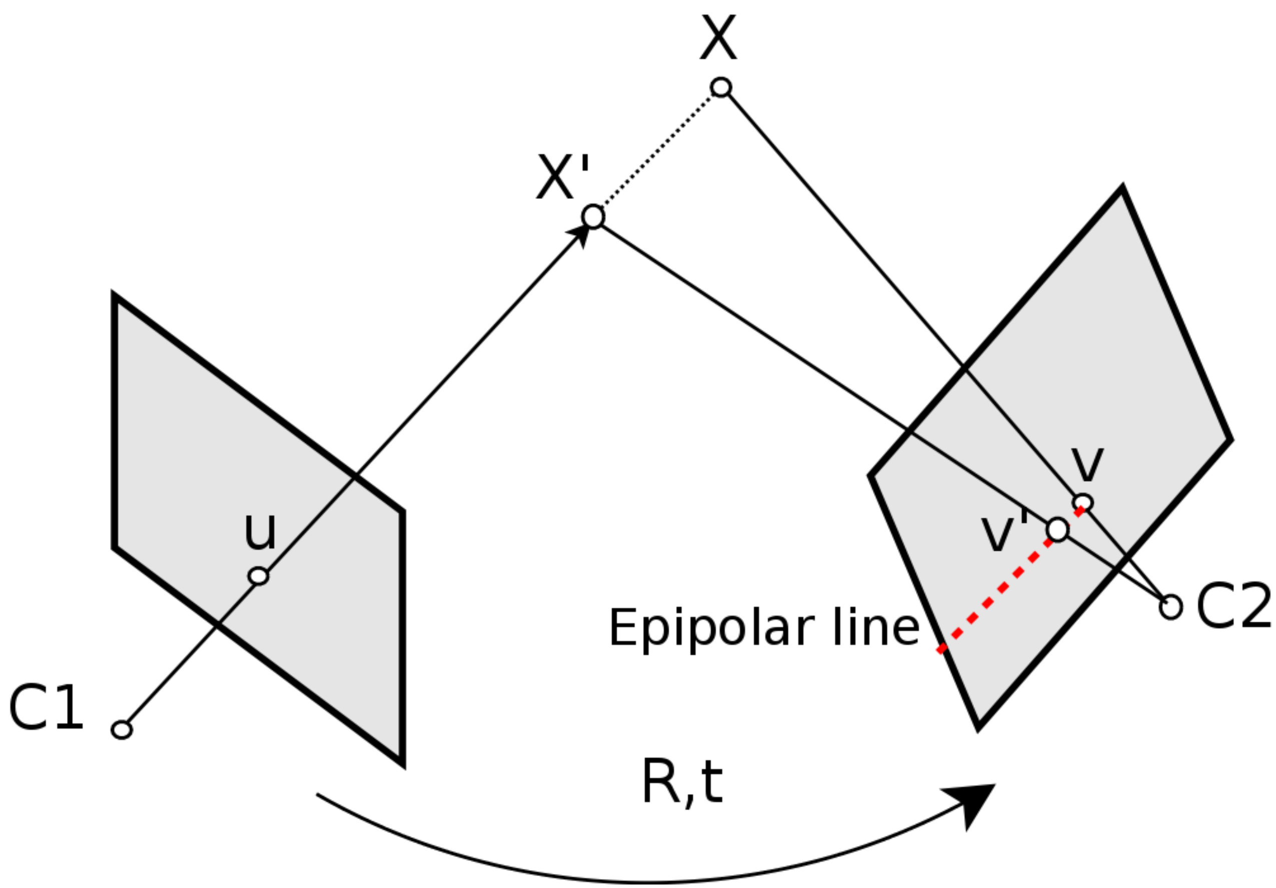
- We assume that K is known (i.e. the camera is calibrated).

Essential matrix

- We assume that $\mathtt{K}$ is known (i.e. the camera is calibrated).
- We normalize coordinates $\mathbf{u_n} = \mathtt{K}^{-1}\mathbf{u}$, $\mathbf{v_n} = \mathtt{K}^{-1}\mathbf{v}$ and pretend that $\mathtt{K}$ is identity.

Essential matrix

- We assume that $\mathtt{K}$ is known (i.e. the camera is calibrated).
- We normalize coordinates $\mathbf{u_n} = \mathtt{K}^{-1}\mathbf{u}$, $\mathbf{v_n} = \mathtt{K}^{-1}\mathbf{v}$ and pretend that $\mathtt{K}$ is identity.
- Epipolar line wrt normalized coordinates is $\{\mathbf{v_n} \mid \mathbf{u_n}^\top \mathtt{E}\, \mathbf{v_n} = 0\}$, where matrix $\mathtt{E} = \mathtt{R} \times \mathbf{t}$ is called essential matrix.



What is the essential matrix good for?

- **Important result 1:**
  - If camera motion is **known** (e.g. stereo), then
  - all possible correspondences of point $\mathbf{u}$ lie on the epipolar line (i.e. either $\{\mathbf{v} \mid \mathbf{u}^\top \mathbf{F}\, \mathbf{v} = 0\}$ or $\{\mathbf{v_n} \mid \mathbf{u_n}^\top \mathbf{E}\, \mathbf{v_n} = 0\}$).

What is the essential matrix good for?

- **Important result 1:**
  - If camera motion is **known** (e.g. stereo), then
  - all possible correspondences of point $\mathbf{u}$ lie on the epipolar line (i.e. either $\{\mathbf{v} \mid \mathbf{u}^\top \mathbf{F}\, \mathbf{v} = 0\}$ or $\{\mathbf{v_n} \mid \mathbf{u_n}^\top \mathbf{E}\, \mathbf{v_n} = 0\}$).
- **Important result 2:**
  - If camera motion is **unknown** (e.g. motion of a single camera), then
  - the essential matrix determines relative position of cameras (i.e. motion), since there exist unique decomposition $\mathtt{E} = \mathtt{R} \times \mathbf{t}$.

What is the essential matrix good for?

- **Important result 1:**
  - If camera motion is **known** (e.g. stereo), then
  - all possible correspondences of point $\mathbf{u}$ lie on the epipolar line (i.e. either $\{\mathbf{v} \mid \mathbf{u}^\top \mathbf{F}\,\mathbf{v} = 0\}$ or $\{\mathbf{v_n} \mid \mathbf{u_n}^\top \mathtt{E}\,\mathbf{v_n} = 0\}$).
- **Important result 2:**
  - If camera motion is **unknown** (e.g. motion of a single camera), then
  - the essential matrix determines relative position of cameras (i.e. motion), since there exist unique decomposition $\mathtt{E} = \mathtt{R} \times \mathtt{t}$.
- From now on, we drop the index n in normalized coordinates.
- How do we obtain the essential/fundamental matrix?

Compute essential matrix by minimizing L2-norm

- Let us assume that we have several correct correspondences.

Compute essential matrix by minimizing L2-norm

- Let us assume that we have several correct correspondences.
- Essential matrix $\mathtt{E}$ is just a solution of (overdetermined) homogeneous system of linear equations.

Compute essential matrix by minimizing L2-norm

- Let us assume that we have several correct correspondences.
- Essential matrix $\mathtt{E}$ is just a solution of (overdetermined) homogeneous system of linear equations.
- For each correspondence pair $\mathbf{u}, \mathbf{v}$, the following holds:

$$\mathbf{u}^\top \mathtt{E}\,\mathbf{v} = \mathbf{u}^\top \begin{bmatrix} \mathbf{e}_1^\top \\ \mathbf{e}_2^\top \\ \mathbf{e}_3^\top \end{bmatrix} \mathbf{v} = \mathbf{u}^\top \begin{bmatrix} \mathbf{e}_1^\top \mathbf{v} \\ \mathbf{e}_2^\top \mathbf{v} \\ \mathbf{e}_3^\top \mathbf{v} \end{bmatrix} = [u_1 \mathbf{e}_1^\top \mathbf{v} + u_2 \mathbf{e}_2^\top \mathbf{v} + u_3 \mathbf{e}_3^\top \mathbf{v}] =$$

Compute essential matrix by minimizing L2-norm

- Let us assume that we have several correct correspondences.
- Essential matrix $\mathtt{E}$ is just a solution of (overdetermined) homogeneous system of linear equations.
- For each correspondence pair $\mathbf{u}, \mathbf{v}$, the following holds:

$$\mathbf{u}^\top \mathbf{E}\,\mathbf{v} = \mathbf{u}^\top \begin{bmatrix} \mathbf{e}_1^\top \\ \mathbf{e}_2^\top \\ \mathbf{e}_3^\top \end{bmatrix} \mathbf{v} = \mathbf{u}^\top \begin{bmatrix} \mathbf{e}_1^\top \mathbf{v} \\ \mathbf{e}_2^\top \mathbf{v} \\ \mathbf{e}_3^\top \mathbf{v} \end{bmatrix} = [u_1 \mathbf{e}_1^\top \mathbf{v} + u_2 \mathbf{e}_2^\top \mathbf{v} + u_3 \mathbf{e}_3^\top \mathbf{v}] =$$

$$= [u_1 \mathbf{v}^\top \, u_2 \mathbf{v}^\top \, u_3 \mathbf{v}^\top] \begin{bmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \\ \mathbf{e}_3 \end{bmatrix} = 0$$

- It must hold for all correspondece pairs $\mathbf{u}_i$, $\mathbf{v}_i$, therefore:

$$\begin{bmatrix} u_{11}\mathbf{v}_1^\top & u_{12}\mathbf{v}_1^\top & u_{13}\mathbf{v}_1^\top \\ u_{21}\mathbf{v}_2^\top & u_{22}\mathbf{v}_2^\top & u_{23}\mathbf{v}_2^\top \\ & \vdots\vdots\vdots & \end{bmatrix} \begin{bmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \\ \mathbf{e}_3 \end{bmatrix} = \mathbf{0}$$

Compute essential matrix by minimizing L2-norm

- It is just homogeneous set of linear equations:

$$\underbrace{\begin{bmatrix} u_{11}\mathbf{v}_1^\top & u_{12}\mathbf{v}_1^\top & u_{13}\mathbf{v}_1^\top \\ u_{21}\mathbf{v}_2^\top & u_{22}\mathbf{v}_2^\top & u_{23}\mathbf{v}_2^\top \\ & \vdots\vdots\vdots & \end{bmatrix}}_{\mathtt{A}} \underbrace{\begin{bmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \\ \mathbf{e}_3 \end{bmatrix}}_{\mathbf{e}} = \mathbf{0}$$

- We want to avoid trivial solution $\mathbf{e}_1 = \mathbf{e}_2 = \mathbf{e}_3 = \mathbf{0}$,
- therefore the following optimization task (constrained LSQ) is solved:

$$\arg\min_{\mathbf{e}} \|\mathtt{A}\mathbf{e}\| \ \text{ subject to } \ \|\mathbf{e}\| = 1$$

- the solution is singular vector of matrix $\mathtt{A}$ corresponding to the smallest singular value (can be found via SVD or eigenvectors/eigenvalues of $\mathtt{A}\mathtt{A}^\top$)

Compute essential matrix by minimizing L2-norm

- The same is valid for the estimation of the fundamental matrix from not normalized coordinates.

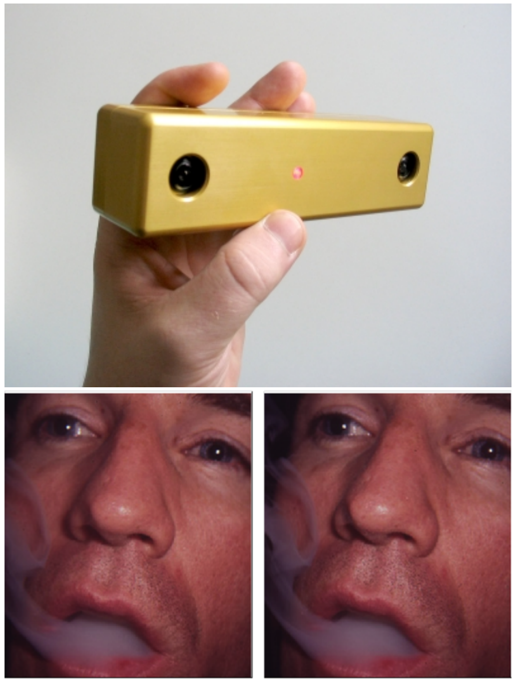
Compute essential matrix by minimizing L2-norm

- The same is valid for the estimation of the fundamental matrix from not normalized coordinates.
- L_2-norm works only in a controlled environment (e.g. offline stereo calibration).

Compute essential matrix by minimizing L2-norm

- The same is valid for the estimation of the fundamental matrix from not normalized coordinates.
- L_2-norm works only in a controlled environment (e.g. offline stereo calibration).
- I will show how essential/fundamental matrix allows to estimate correspondences in state-of-the-art depth (3D) sensors.

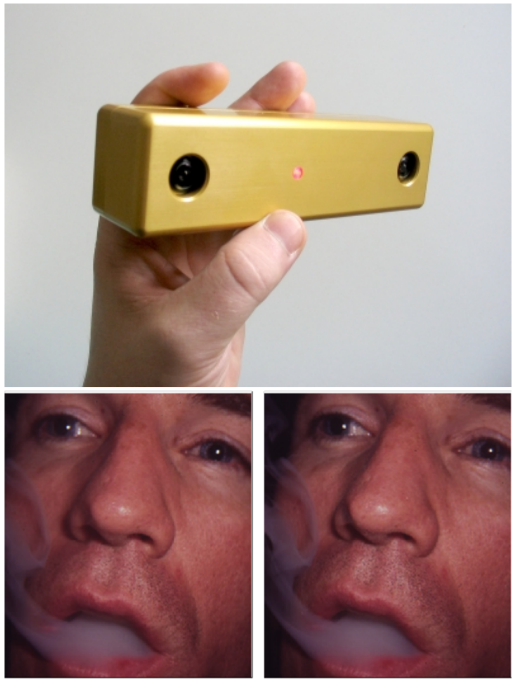
Stereo



- Pair of cameras mounted on a rigid body, which provides depth (3D points) of the scene (simulates human binocular vision).
- Relative position of cameras fixed

[0] Courtesy of prof.Boris Flach for original stereo images and depth images

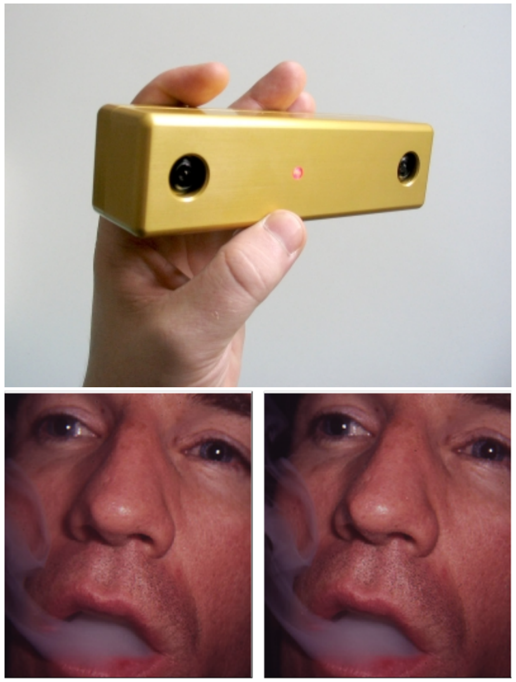
Stereo



- Pair of cameras mounted on a rigid body, which provides depth (3D points) of the scene (simulates human binocular vision).
- Relative position of cameras fixed
- **offline**: fundamental matrix estimated from known correspondences.

[0] Courtesy of prof.Boris Flach for original stereo images and depth images

Stereo

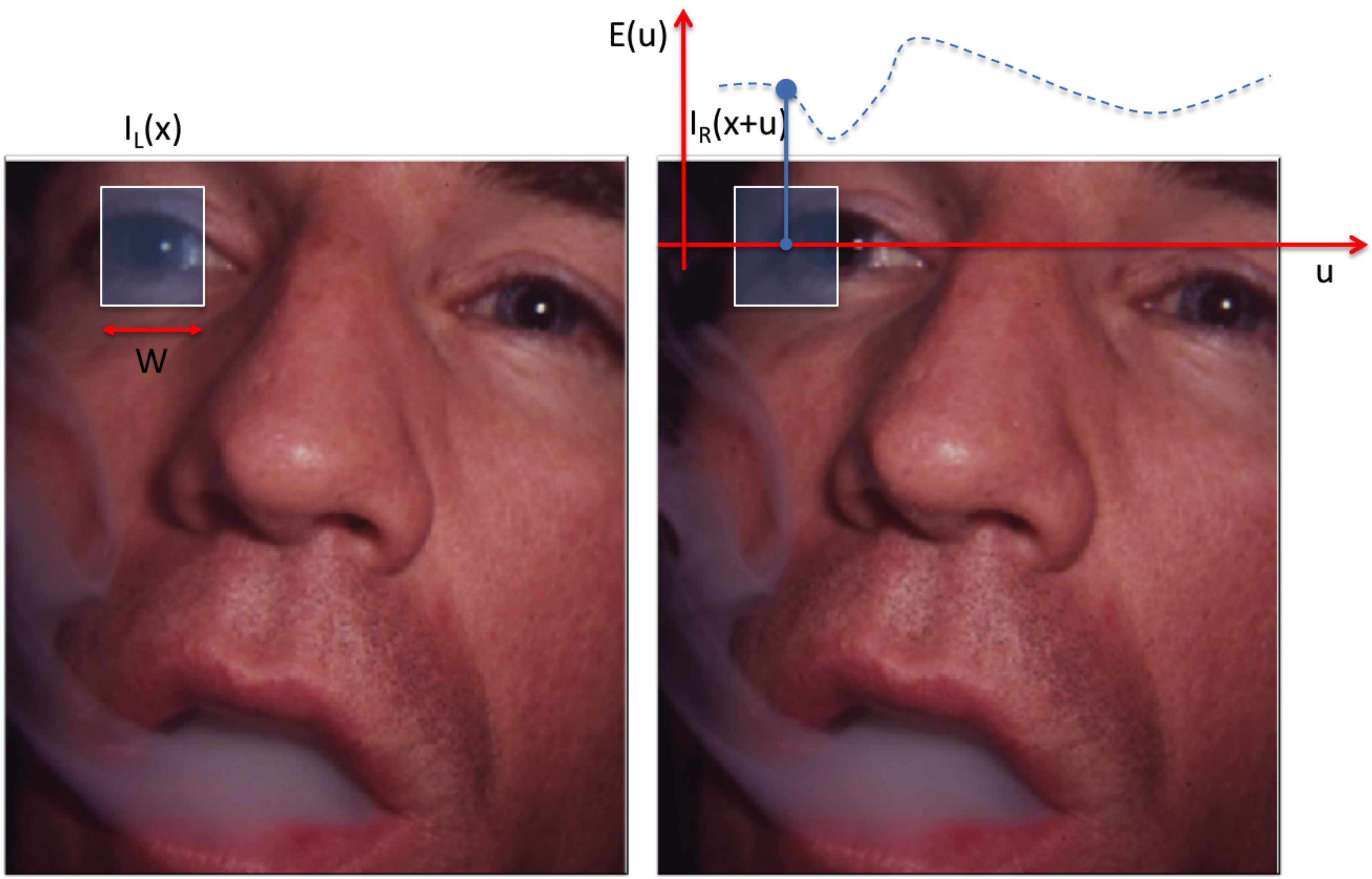


- Pair of cameras mounted on a rigid body, which provides depth (3D points) of the scene (simulates human binocular vision).
- Relative position of cameras fixed
- **offline**: fundamental matrix estimated from known correspondences.
- **online**: correspondences searched along epipolar lines.

[0] Courtesy of prof.Boris Flach for original stereo images and depth images

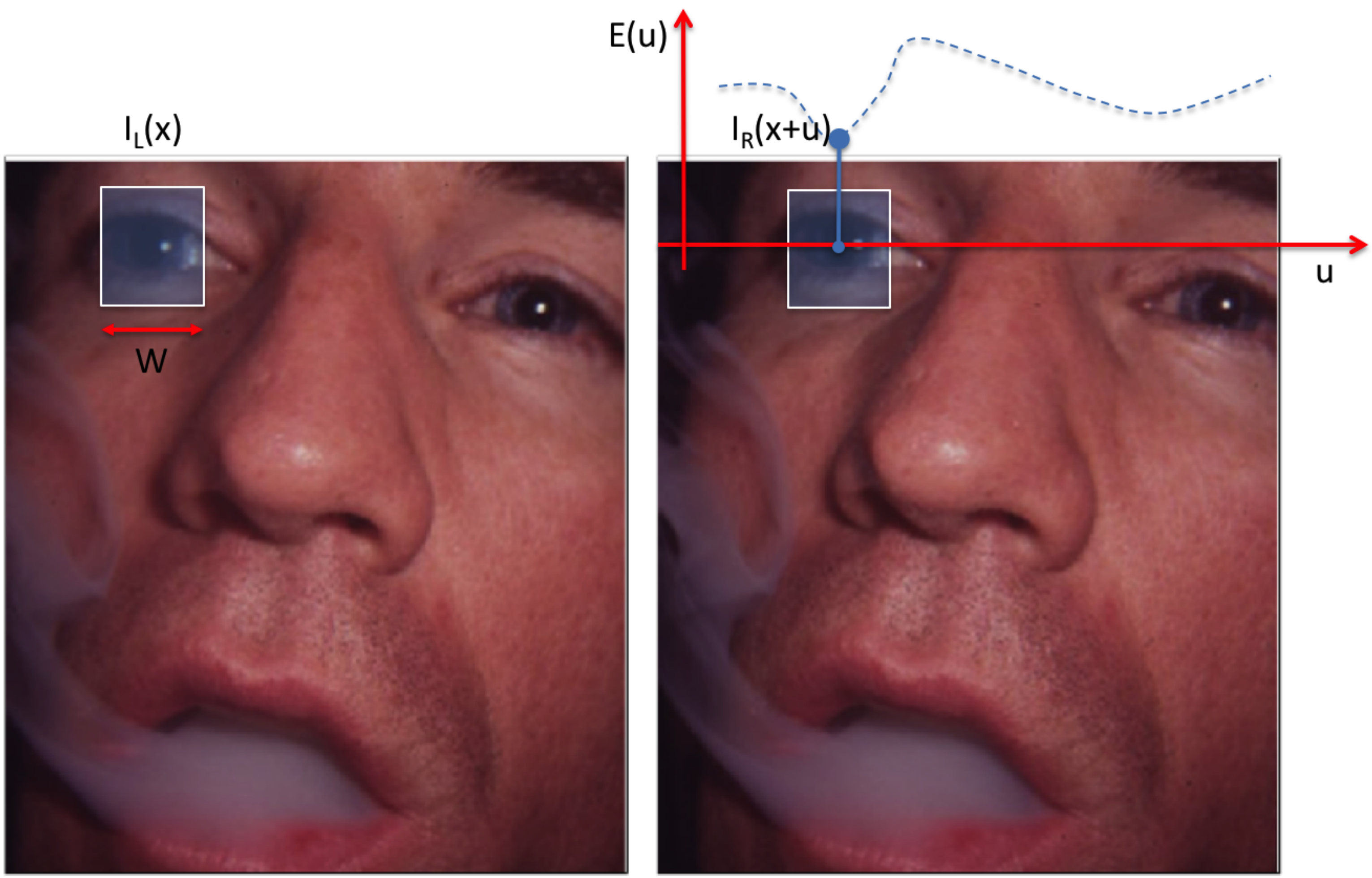
Stereo

Block-matching energy function: $E(u) = \sum_{x \in W} (I_L(x) - I_R(x+u))^2$



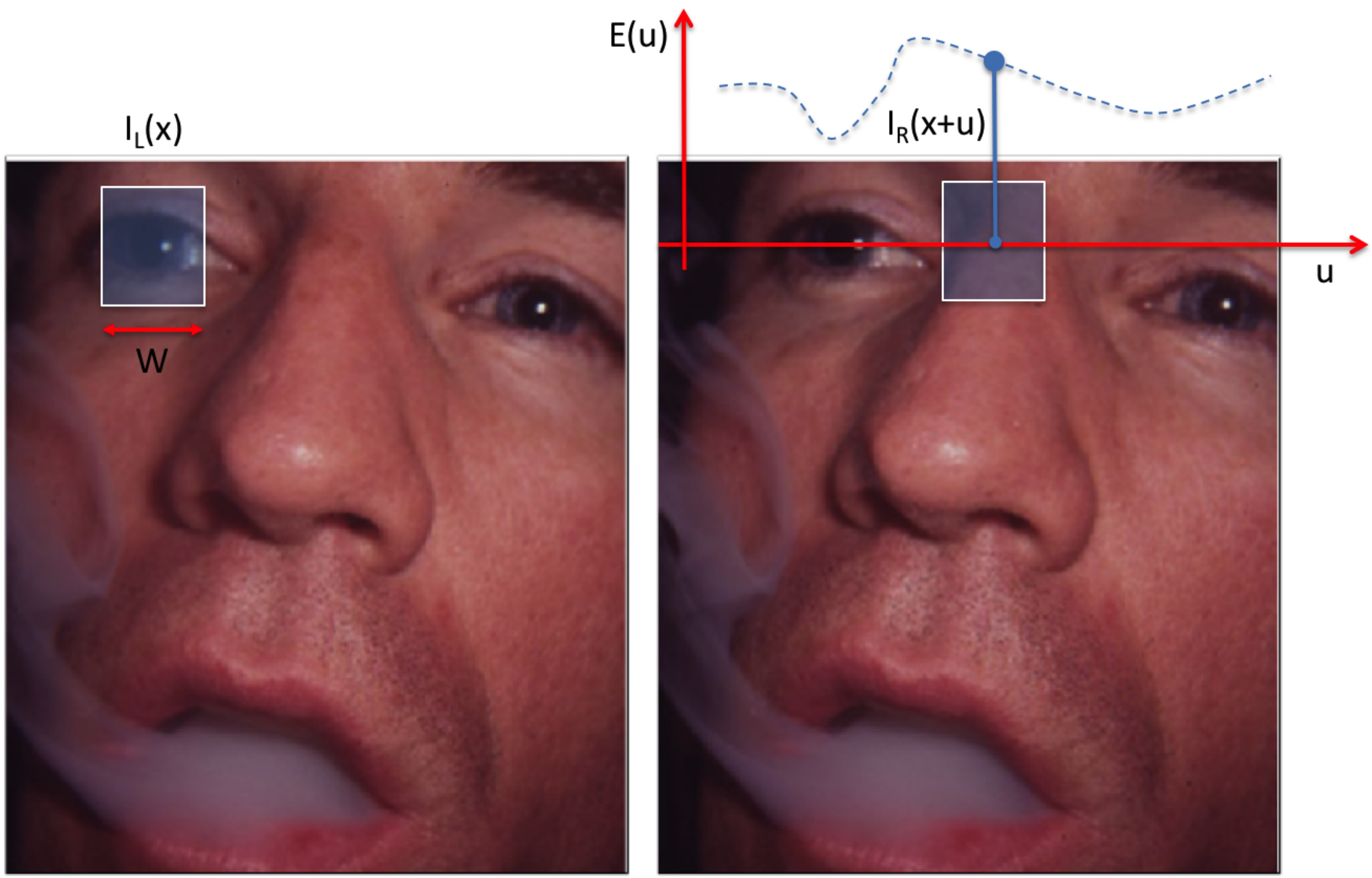
Stereo

Block-matching energy function: $E(u) = \sum_{x \in W} (I_L(x) - I_R(x+u))^2$



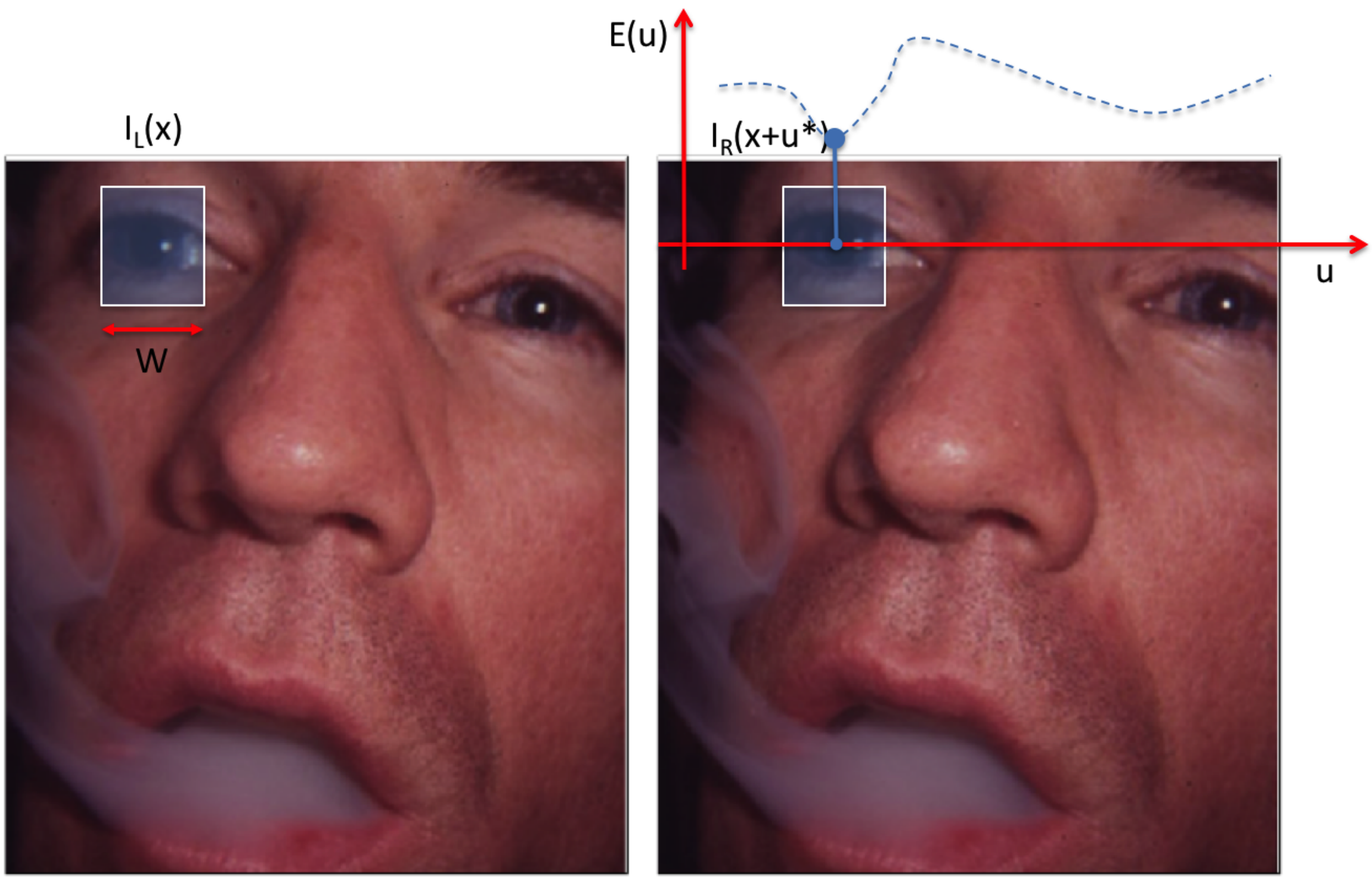
Stereo

Block-matching energy function: $E(u) = \sum_{x \in W} (I_L(x) - I_R(x+u))^2$



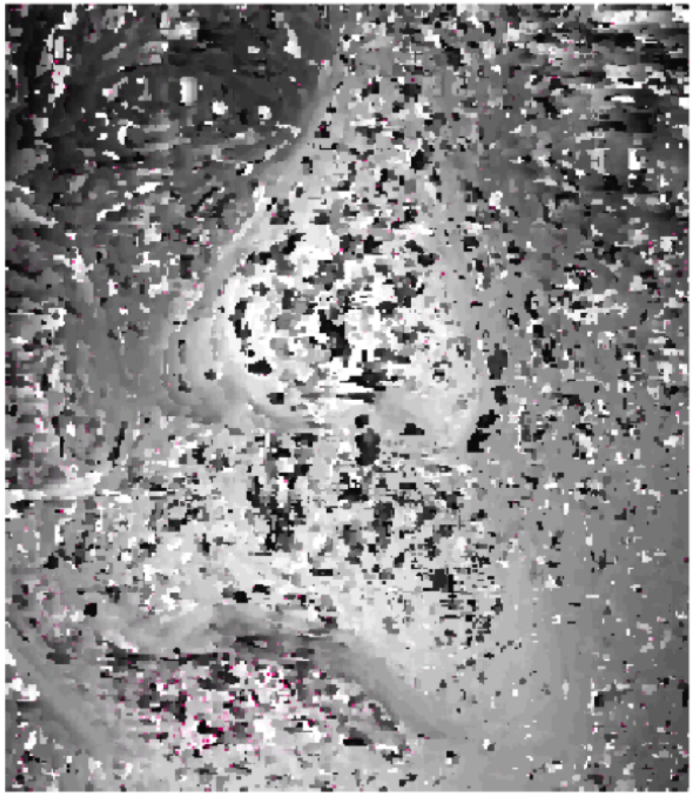
Stereo

Correspondence for each pixel estimated separately: $u^* = \arg\min_u E(u)$



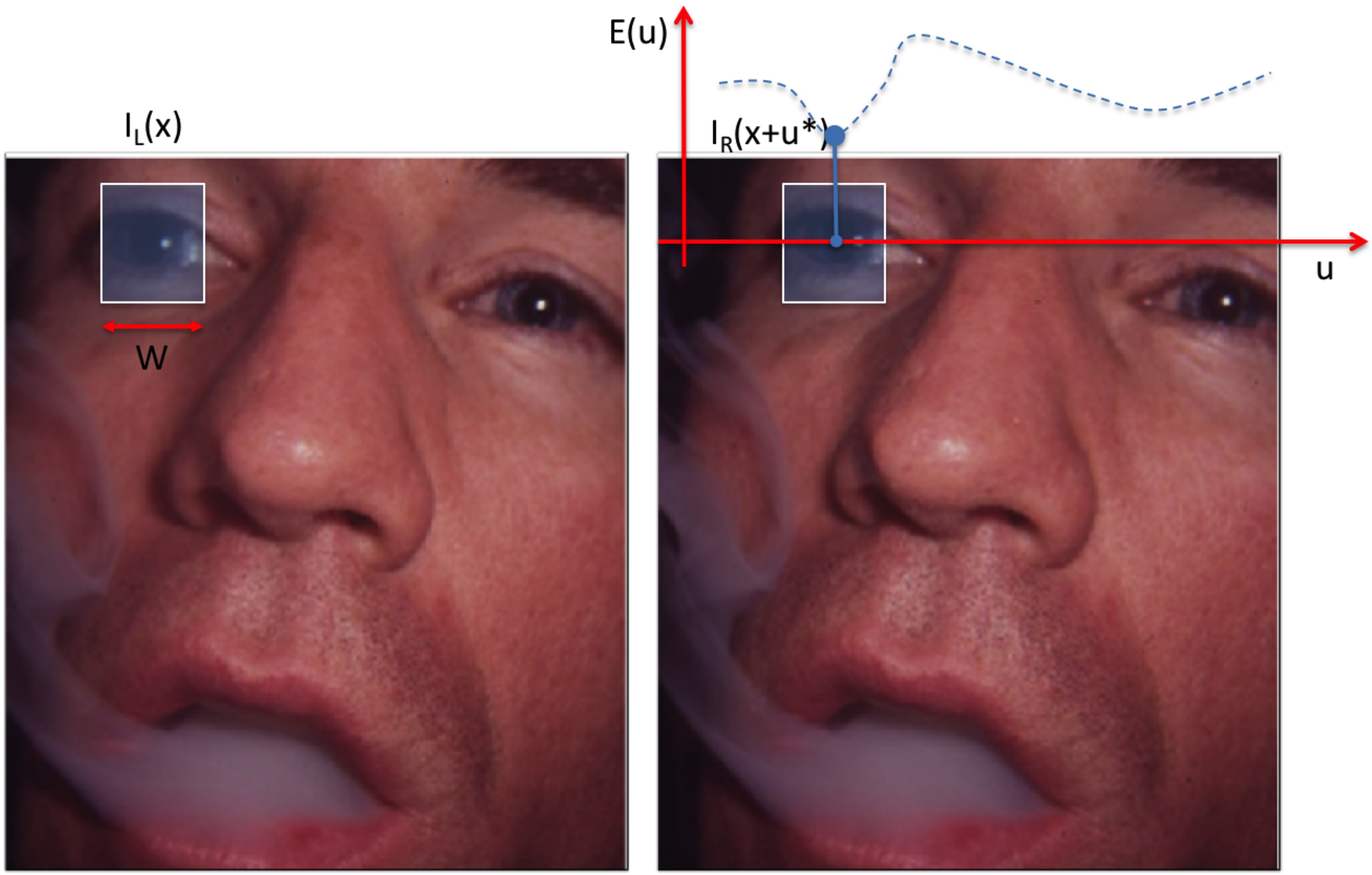
Stereo

Correspondence for each pixel estimated separately: $u^* = \arg\min_u E(u)$



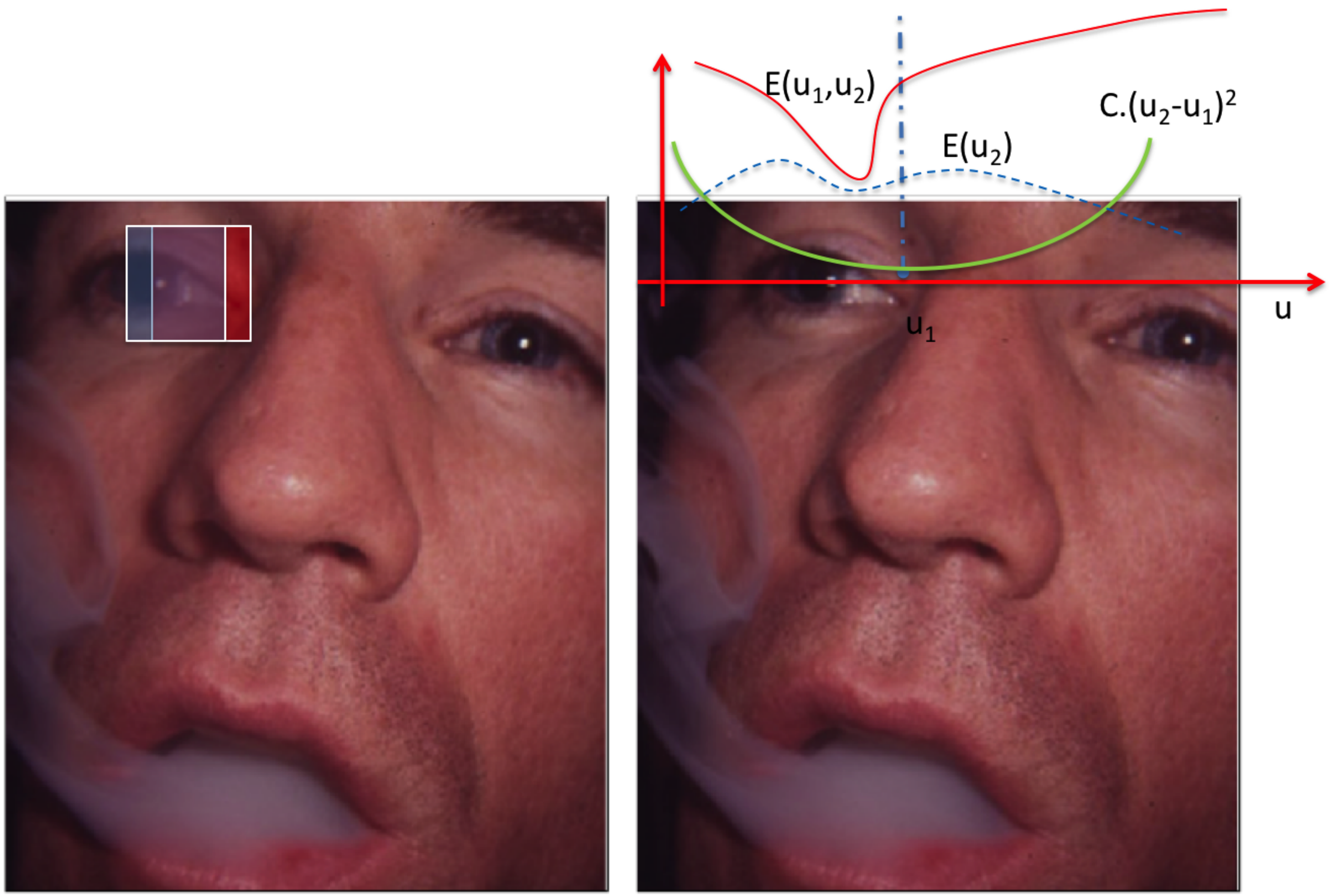
Stereo

How can we improve the result?



Stereo

Energy with horizontal smoothness term: $E(u_1, u_2) = E(u_2) + C \cdot (u_2 - u_1)^2$

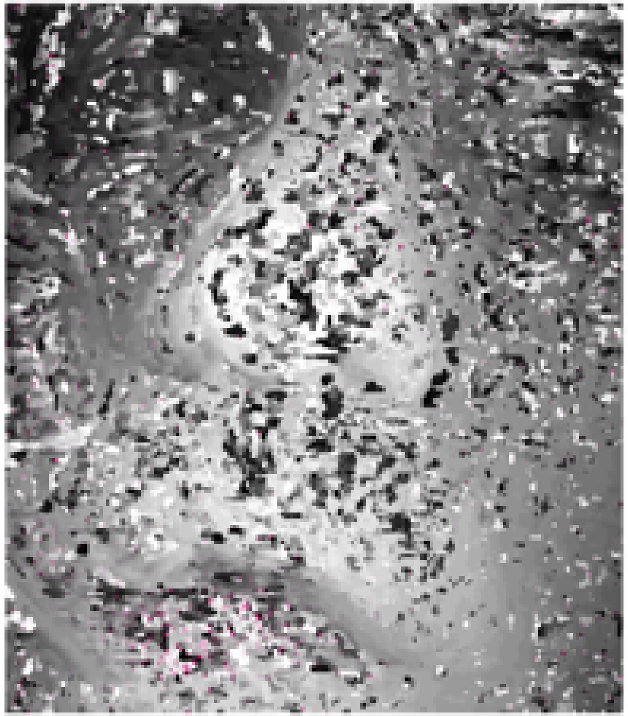


Stereo

Dynamic programming solves each line of N pixels separately:

$$U^* = \arg\min_{U \in \mathcal{R}^{\mathcal{N}}} \sum_{i=1}^{N-1} E(u_i, u_{i+1})$$

Image



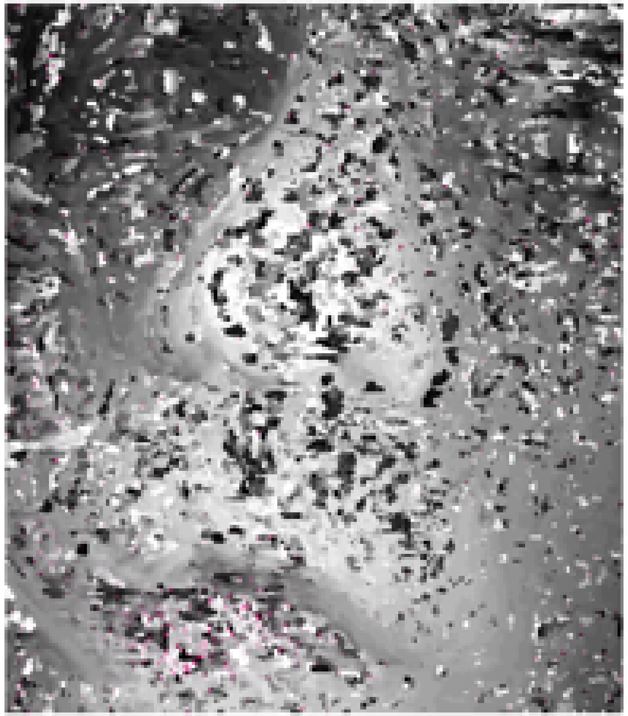
Block matching

Dynamic programming

Stereo

What else can we do?

Image

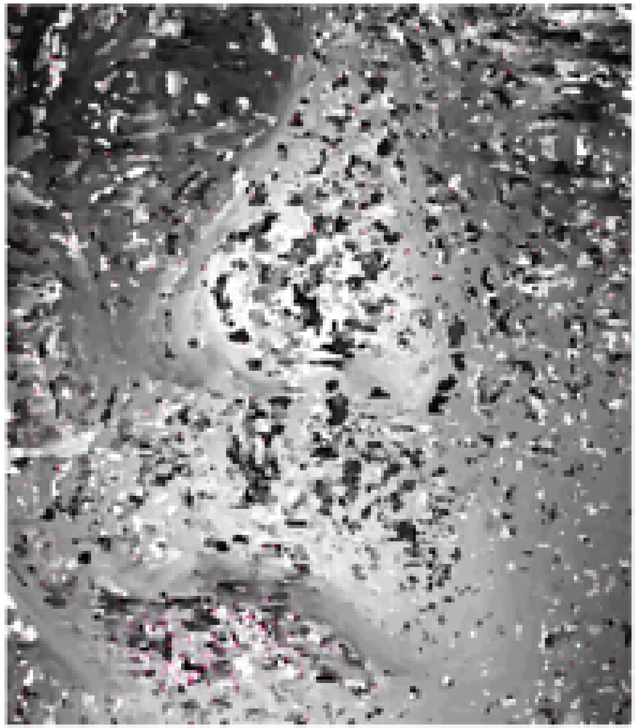


Block matching

Dynamic programming

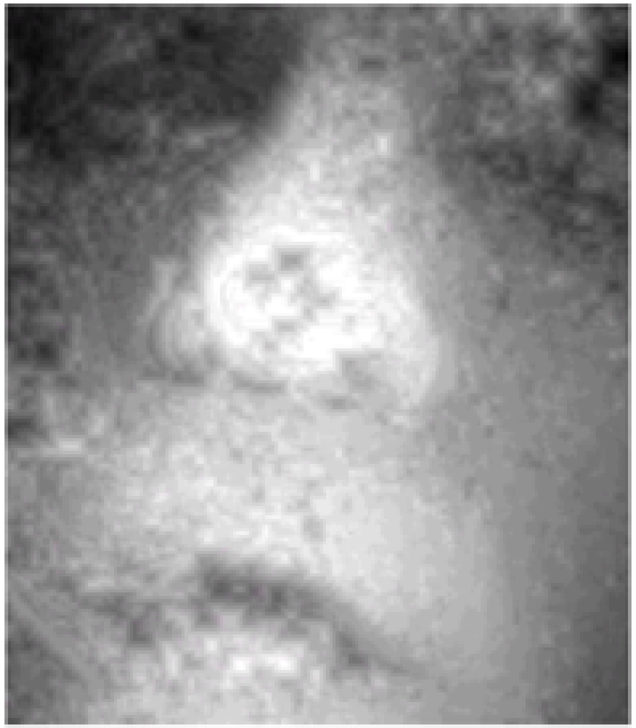
Stereo

Enforce also vertical smoothness $\Rightarrow$ graph energy minimization (computationally demanding optimization solved on specialized chips).



Block matching

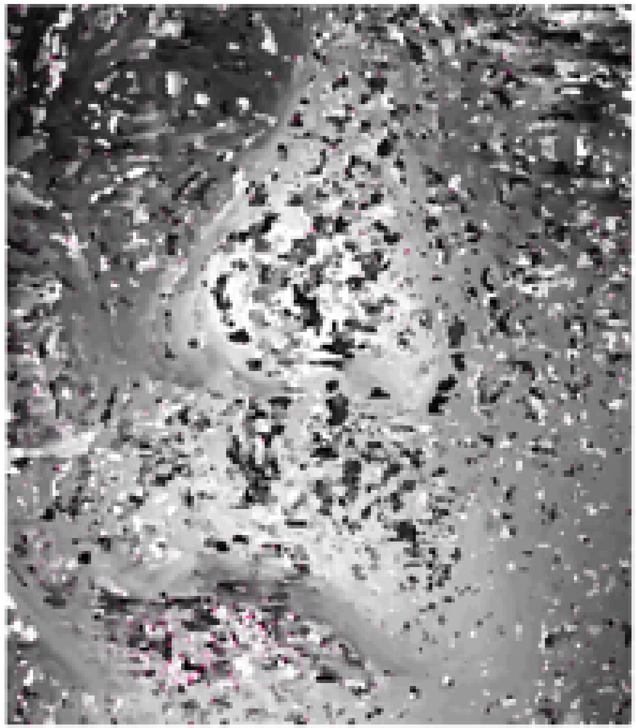
Dynamic programming



(Min,+) solution

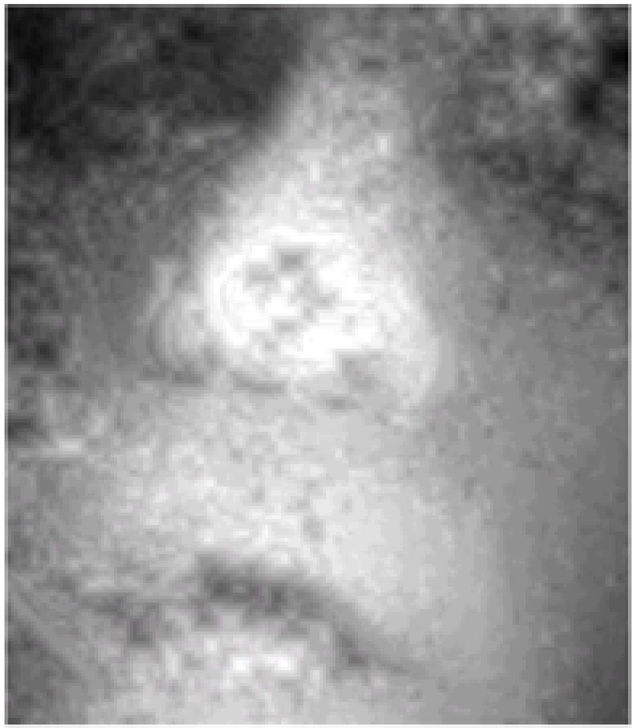
Stereo

Enforce also vertical smoothness $\Rightarrow$ graph energy minimization (computationally demanding optimization solved on specialized chips).



Block matching

Dynamic programming



(Min,+) solution

- **Limitation:** usually works only on sufficiently rich patterns and sufficiently smooth depths.

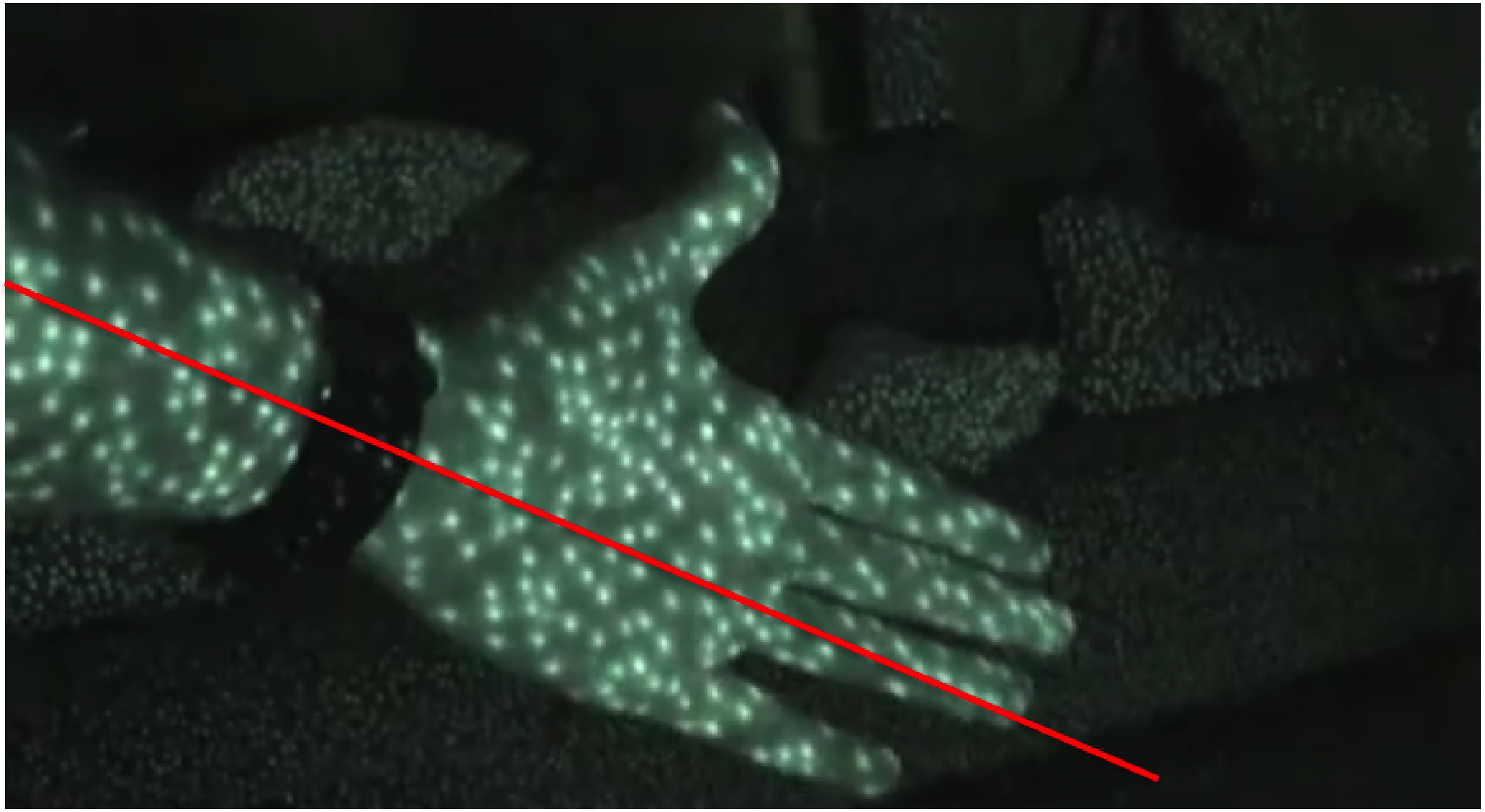
Stereo competition

- Do you have your own idea how to estimate the depth from stereo images?
- http://vision/middlebury.edu/stereo/data/2014/

Kinect (structured-light approach)

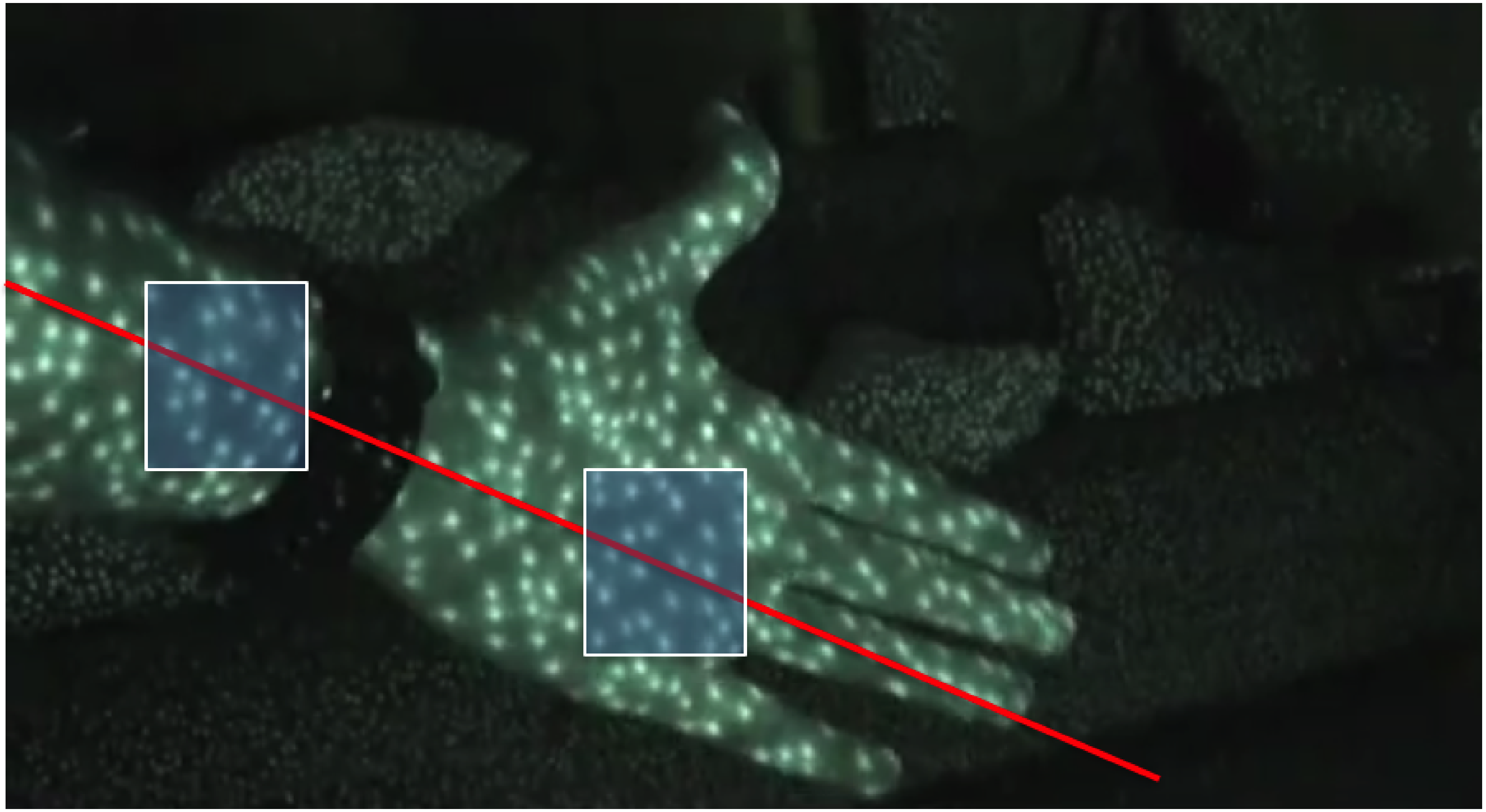
- **Stereo** looks at the same object two-times and estimates the correspondence from two passive RGB images.
- **Kinect** avoids ambiguity by actively projecting a unique IR pattern on the surface and search for its known appearance in the IR camera.

Kinect



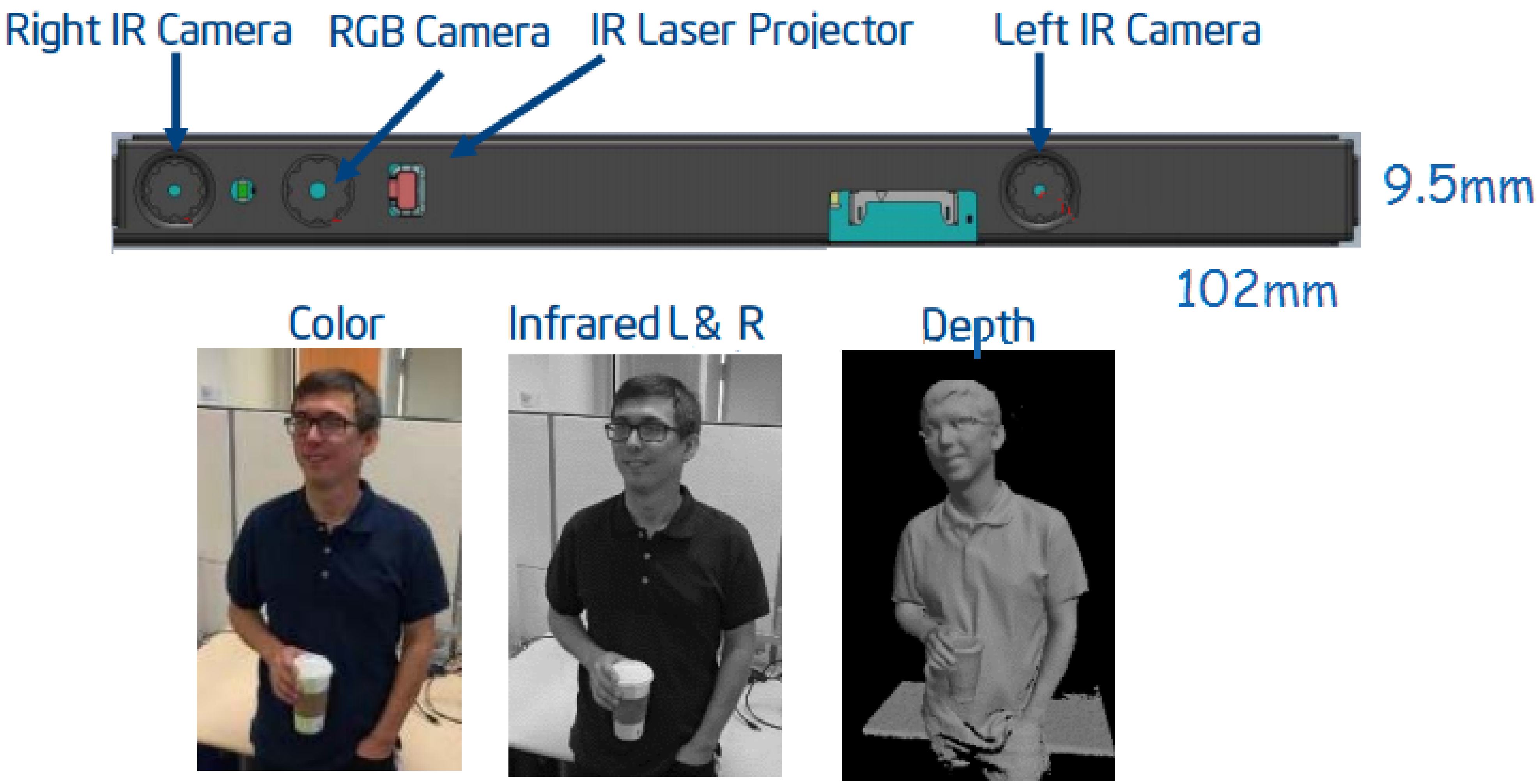
- Since camera-projector relative position is known, correspondence between projected pixel and observed pixel lies again on epipolar lines.

Kinect



- Unique IR speckle-pattern: no two sub-windows with the same pattern
- Energy along epipolar line has only one strong minimum.
- Kinect fusion: http://research.microsoft.com/en-us/projects/surfacerecon/
- **Limitation:** works only indoor.

RealSense



- Hybrid approach one IR projector and two IR cameras.
- Combines advantages of stereo and structured light approach. So far best solution for robotics.

Depth from a single camera

- Is it possible to get the 3D points from a single camera?

Depth from a single camera

- Is it possible to get the 3D points from a single camera?
- Theoretically yes (if scene is static and the camera moves around sufficiently).

Depth from a single camera

- Is it possible to get the 3D points from a single camera?
- Theoretically yes (if scene is static and the camera moves around sufficiently).
- We have also two cameras. Main difference is that they have been captured in different times and the relative motion (i.e. epipolar geometry) is unknown.

Depth from a single camera

- Is it possible to get the 3D points from a single camera?
- Theoretically yes (if scene is static and the camera moves around sufficiently).
- We have also two cameras. Main difference is that they have been captured in different times and the relative motion (i.e. epipolar geometry) is unknown.
- The second part of this lecture is about how to estimate **online** both the relative motion of the camera and the 3D model of the world from captured images.

Depth from a single camera

- Is it possible to get the 3D points from a single camera?
- Theoretically yes (if scene is static and the camera moves around sufficiently).
- We have also two cameras. Main difference is that they have been captured in different times and the relative motion (i.e. epipolar geometry) is unknown.
- The second part of this lecture is about how to estimate **online** both the relative motion of the camera and the 3D model of the world from captured images.
- We assume, that at least the camera intrinsic parameters $\mathtt{K}$ has been calibrated **offline**.

Algorithm at glance

1. Get image I_k.
2. Estimate tentative correspondences between I_{k-1} and I_k .
3. Find correct correspondences and robustly estimate essential matrix $\mathtt{E}$
4. Decompose $\mathtt{E}$ into $\mathtt{R}_k$ and $\mathbf{t}_k$.
5. Compute 3D model (points X).
6. Rescale t_k according to relative scale r.
7. $k = k + 1$

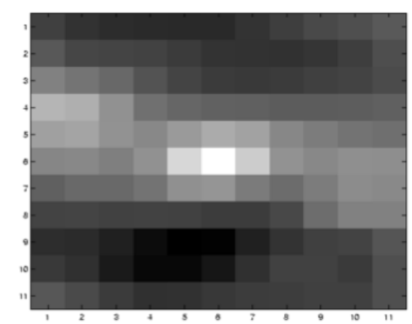
Feature point detection

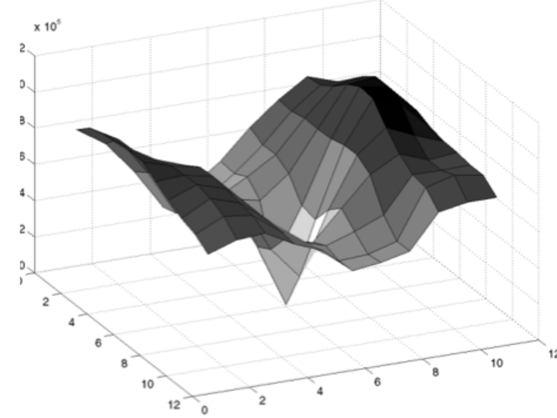
- Which points are suitable?

Feature point detection

- Feature points must be well distinguishable from its neighbourhood.

$$E(u,v) = \sum_{x,y} \Big(I(x+u, y+v) - I(x,y) \Big)^2 \approx [u\ v]\ \mathtt{M} \begin{bmatrix} u \\ v \end{bmatrix}$$



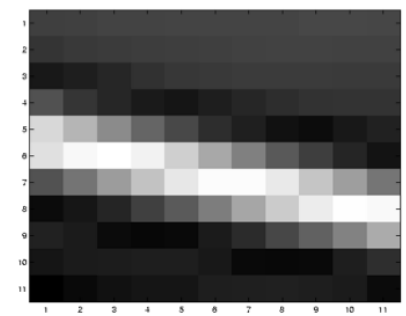


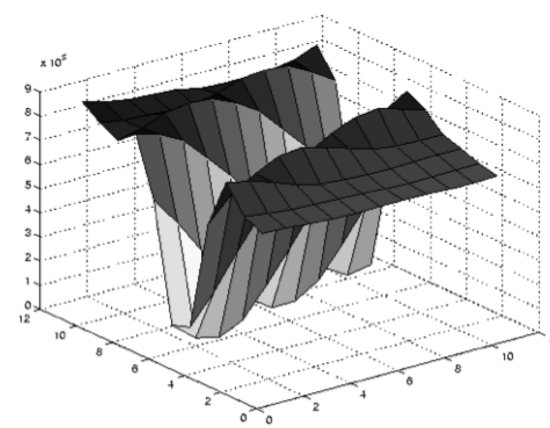
λ_1 and λ_2 are large

Feature point detection

- Feature points must be well distinguishable from its neighbourhood.

$$E(u,v) = \sum_{x,y} \Big(I(x+u, y+v) - I(x,y) \Big)^2 \approx [u\ v]\ \mathtt{M} \begin{bmatrix} u \\ v \end{bmatrix}$$



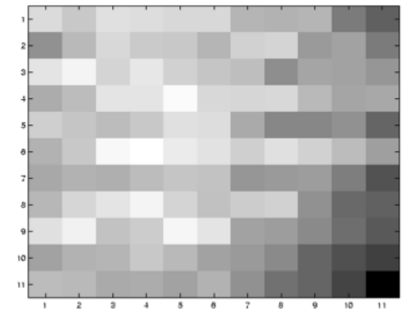


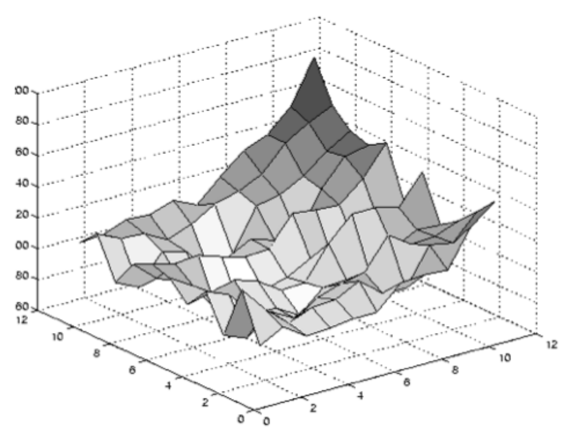
large λ_1, small λ_2

Feature point detection

- Feature points must be well distinguishable from its neighbourhood.

$$E(u,v) = \sum_{x,y} \Big(I(x+u, y+v) - I(x,y)\Big)^2 \approx [u\ v]\ \mathtt{M} \begin{bmatrix} u \\ v \end{bmatrix}$$





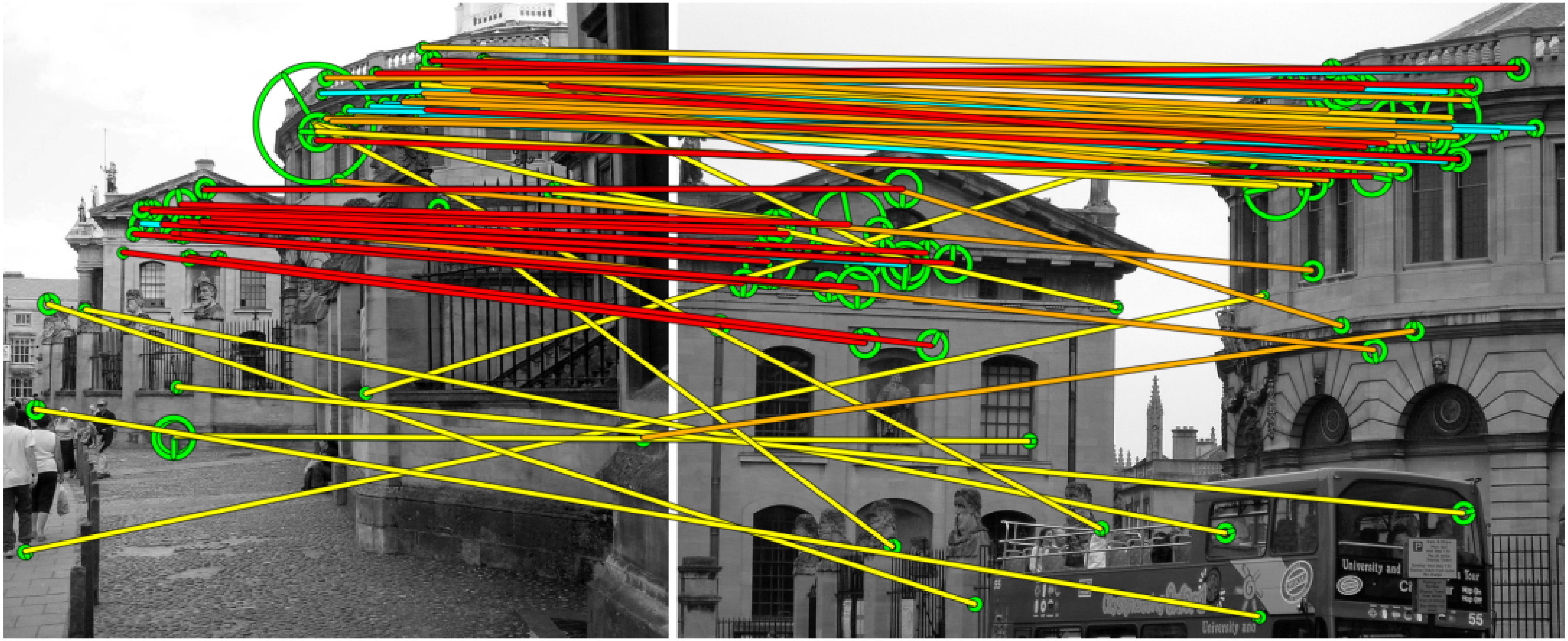
small λ_1, small λ_2

Estimate tentative correspondences

- Estimate tentative correspondences by matching pixel neighbourhoods.
- Matching pixels: Tracking - for high temporal resolution

  OpenCV Lucas-Kanade tracker
- Matching invariant descriptors: Detection - for high spatial resolution

  OpenCV: SIFT, SURF etc ...



Algorithm at glance

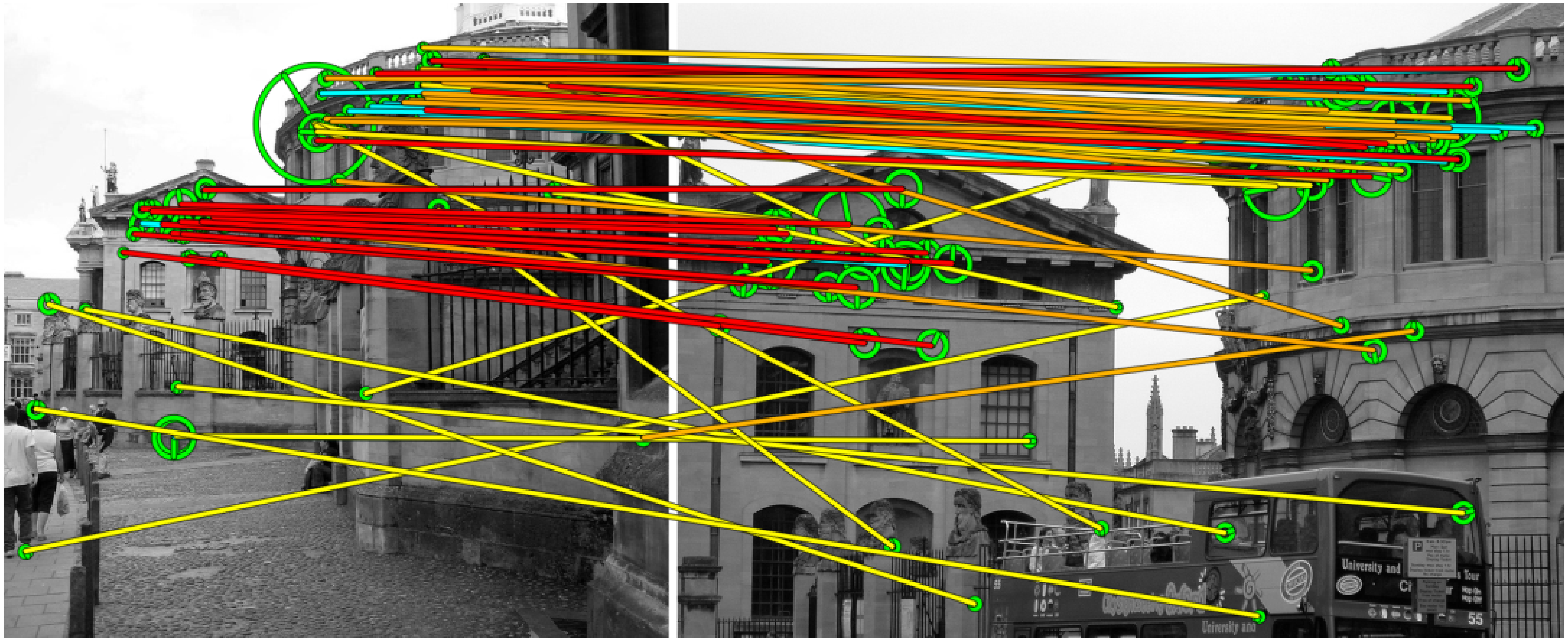
1. Get image I_k.
2. Estimate tentative correspondences between I_{k-1} and I_k.
3. Find correct correspondences and robustly estimate essential matrix $\mathtt{E}$
4. Decompose $\mathtt{E}$ into $\mathtt{R}_k$ and $\mathbf{t}_k$.
5. Compute 3D model (points X).
6. Rescale t_k according to relative scale r.
7. $k = k + 1$

Estimate essential matrix

- most of the tentative correspondences is **incorrect**,
- L_2-norm is very sensitive to such incorrect correspondence (i.e. ouliers).
- Direct minimization of the L_2-norm, yields poor essential matrix

$$\mathbf{e}^* = \arg\min_{\mathbf{e}} \|\mathtt{A}\mathbf{e}\|$$

$$\text{s.t. } \|\mathbf{e}\| = 1$$



Estimate essential matrix by minimizing box-penalty function

- We will use outlier-insensitive estimation which will find both:
  - the correct essential matrix and
  - the set of correct correspondences (i.e. inliers).

Estimate essential matrix by minimizing box-penalty function

- What makes the L_2-norm outlier-sensitive?

Estimate essential matrix by minimizing box-penalty function

- What makes the L_2-norm outlier-sensitive?

- L_2-norm:

$$\arg\min_{\mathbf{e}} \|\mathtt{A}\mathbf{e}\|$$

$$\text{s.t. } \|\mathbf{e}\| = 1$$

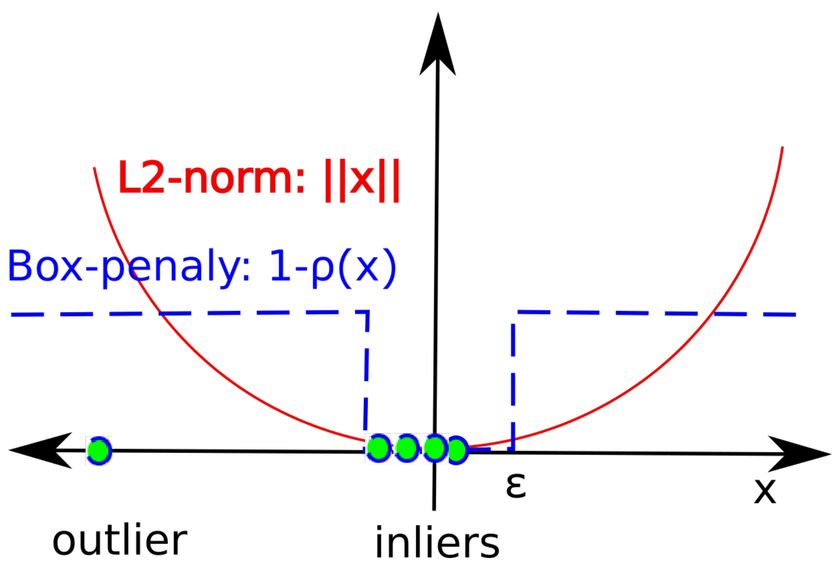
Estimate essential matrix by minimizing box-penalty function

- What makes the L_2-norm outlier-sensitive?
- L_2-norm:

$$\arg\min_{\mathbf{e}} \|\mathtt{A}\mathbf{e}\|$$
$$\text{s.t. } \|\mathbf{e}\| = 1$$

- Box-penalty:

$$\arg\min_{\mathbf{e}} 1 - \rho(\mathtt{A}\mathbf{e})$$
$$\text{s.t. } \|\mathbf{e}\| = 1$$



RANSAC algorithm

- We solve the following not-convex and not-differentiable optimization task:

$$\begin{aligned} \arg\min_{\mathbf{e}} \; & 1 - \rho(\mathtt{A}\mathbf{e}) = \arg\max_{\mathbf{e}} \; \rho(\mathtt{A}\mathbf{e}) \\ \text{s.t.} \; & \|\mathbf{e}\| = 1 \qquad\qquad \text{s.t.} \; \|\mathbf{e}\| = 1 \end{aligned}$$

RANSAC algorithm

- We solve the following not-convex and not-differentiable optimization task:

$$\arg\min_{\mathbf{e}} 1 - \rho(\mathtt{A}\mathbf{e}) = \arg\max_{\mathbf{e}} \rho(\mathtt{A}\mathbf{e})$$

$$\text{s.t. } \|\mathbf{e}\| = 1 \qquad\qquad \text{s.t. } \|\mathbf{e}\| = 1$$

- RANSAC (RAndom SAmple Consensus) algorithm:
  1. Randomly choose minimal subset of equations (rows) $\mathtt{B}$ from $\mathtt{A}$.
  2. Solve constrained LSQ problem by SVD decomposition:

$$\mathbf{e}^* = \arg\min_{\mathbf{e}} \|\mathtt{B}\mathbf{e}\|$$

$$\text{s.t. } \|\mathbf{e}\| = 1$$

  3. Estimate $\rho(\mathtt{A}\mathbf{e}^*)$ as the number of rows $\mathbf{a}_i^\top$ of $\mathtt{A}$ which satisfy $|\mathbf{a}_i^\top \mathbf{e}^*| < \epsilon$.
  4. If $\rho_{\max} > \rho(\mathbf{e}^*)$ then $\rho_{\max} = \rho(\mathbf{e}^*)$ and $\mathbf{e}_{\max} = \mathbf{e}^*$.
  5. Repeat from 1 until the optimum is found with sufficient probability.

RANSAC properties

- **Important result 3:** Let us denote
  - $N \ldots$ number of data points.
  - $w \ldots$ fraction of inliers.
  - $s \ldots$ size of the sample
  - $K \ldots$ number of trials.
  - $p \ldots$ probability to select uncontaminated samples at least once
- then

$$K = \frac{\log(1-p)}{\log(1-w^s)}$$

RANSAC properties

- **Important result 3:** Let us denote
  - $N \ldots$ number of data points.
  - $w \ldots$ fraction of inliers.
  - $s \ldots$ size of the sample
  - $K \ldots$ number of trials.
  - $p \ldots$ probability to select uncontaminated samples at least once
- then

$$K = \frac{\log(1-p)}{\log(1-w^s)}$$

- We search for 8 unknows ($\dim(\mathbf{e}) = 9$ minus scale) $\Rightarrow$ at least 8 correspondences needed $\Rightarrow$ $s = 8$ $\Rightarrow$ K grows fast with s.
- However you want to find only camera translation (3 DoFs) and rotation (3 DoFs) minus scale $\Rightarrow$ 5-point algorithm [Nister 2003].

Algorithm at glance

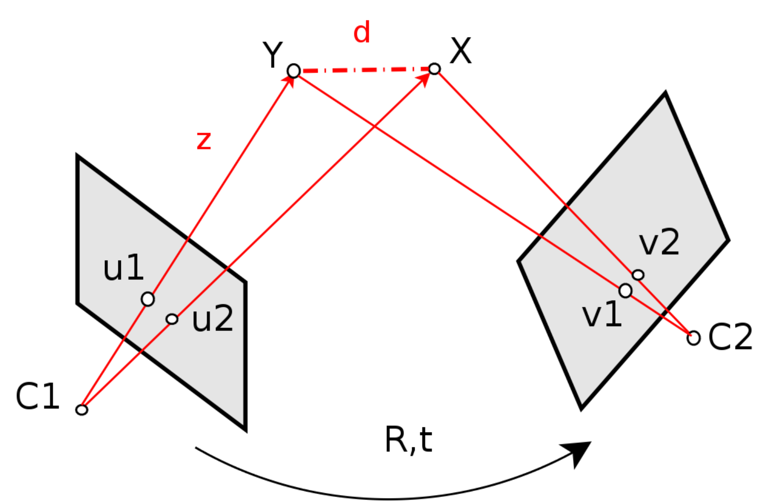
1. Get image I_k.
2. Estimate tentative correspondences between I_{k-1} and I_k .
3. Find correct correspondences and compute essential matrix $\mathtt{E}$.
4. Decompose $\mathtt{E}$ into $\mathtt{R}_k$ and $\mathbf{t}_k$.
5. Compute 3D model (points X).
6. Rescale t_k according to relative scale r.
7. $k = k + 1$

Decompose $\mathtt{E}$ into $\mathtt{R}$ and $\mathbf{t}$

- Once you find $\mathtt{E}$, you can estimate camera motion by SVD ($\mathtt{E} = \mathtt{U}\Sigma\mathtt{V}^\top$) as follows: $[\mathbf{t}]_\times = \mathtt{V}\mathtt{W}\Sigma\mathtt{V}^\top,\ \mathtt{R} = \mathtt{U}\mathtt{W}^{-1}\mathtt{V}^\top$, **but !!!**:

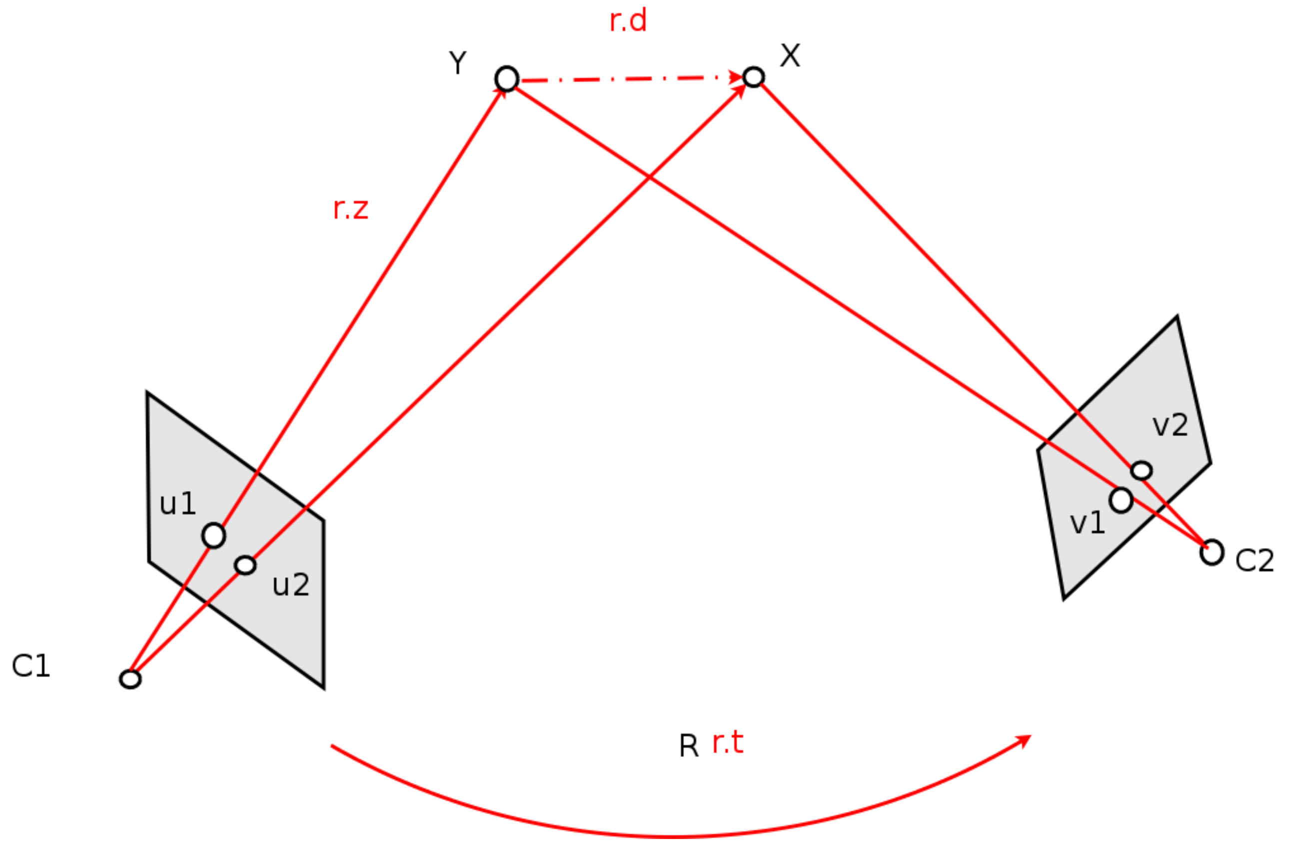
Decompose $\mathtt{E}$ into $\mathtt{R}$ and $\mathbf{t}$

- Once you find $\mathtt{E}$, you can estimate camera motion by SVD ($\mathtt{E} = \mathtt{U}\Sigma\mathtt{V}^\top$) as follows: $[\mathbf{t}]_\times = \mathtt{V}\mathtt{W}\Sigma\mathtt{V}^\top$, $\mathtt{R} = \mathtt{U}\mathtt{W}^{-1}\mathtt{V}^\top$, **but !!!**:
- Scale r is unknown (if $\|\mathtt{A} \cdot \mathbf{e}^*\| \approx 0$, then $\|\mathtt{A} \cdot (r\mathbf{e}^*)\| \approx 0$).



Decompose $\mathtt{E}$ into $\mathtt{R}$ and $\mathtt{t}$

- Once you find $\mathtt{E}$, you can estimate camera motion by SVD ($\mathtt{E} = \mathtt{U}\Sigma\mathtt{V}^\top$) as follows: $[\mathbf{t}]_\times = \mathtt{V}\mathtt{W}\Sigma\mathtt{V}^\top$, $\mathtt{R} = \mathtt{U}\mathtt{W}^{-1}\mathtt{V}^\top$, **but !!!**:
- Scale r is unknown (if $\|\mathtt{A}\cdot\mathbf{e}^*\| \approx 0$, then $\|\mathtt{A}\cdot(r\mathbf{e}^*)\| \approx 0$).



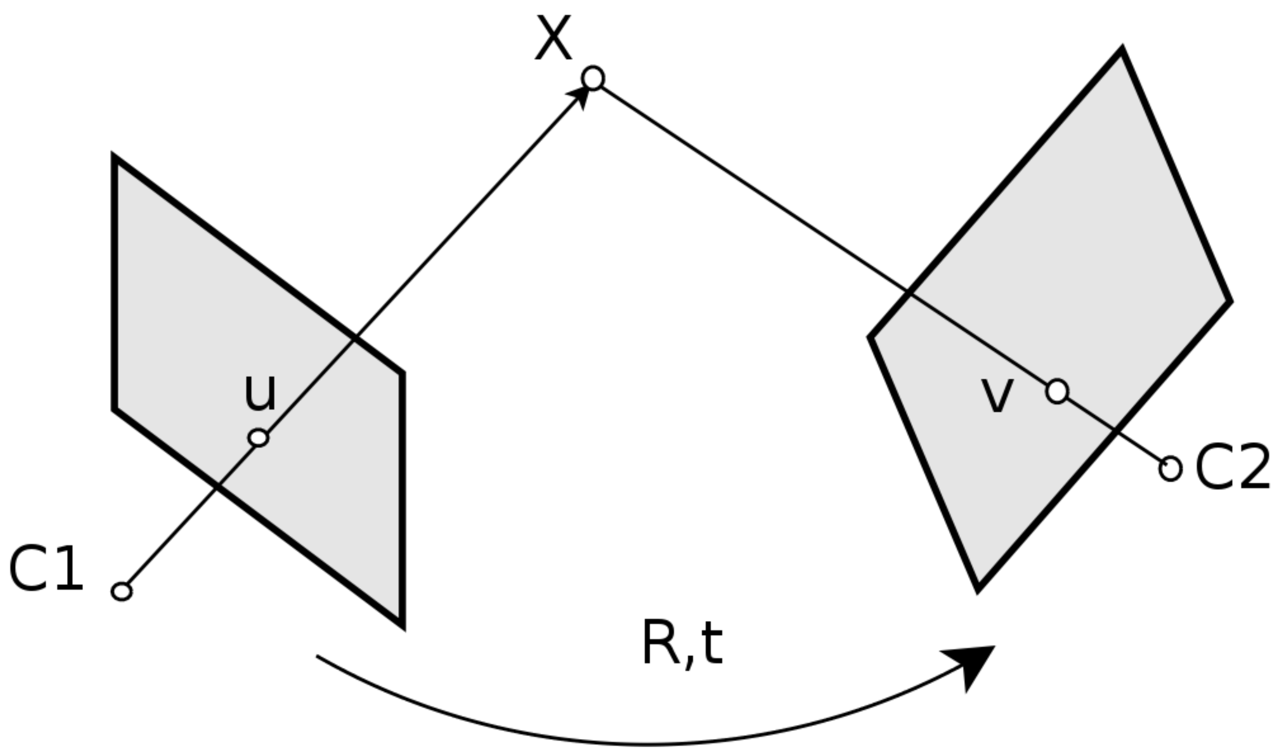
Algorithm at glance

1. Get image I_k.
2. Estimate tentative correspondences between I_{k-1} and I_k (either feature matching or tracking).
3. Find correct correspondences and compute essential matrix $\mathtt{E}$.
4. Decompose $\mathtt{E}$ into $\mathtt{R}_k$ and $\mathbf{t}_k$.
5. Compute 3D model (points X).
6. Rescale t_k according to relative scale r.
7. $k = k + 1$

Compute 3D model

- Scene point X is observed by two cameras `P` and `Q`.
- Let $\mathbf{u} = [u_1\ u_2]^\top$ and $\mathbf{v} = [v_1\ v_2]^\top$ are projections of X in `P` and `Q`,
- then

$$u_1 = \frac{\mathbf{p}_1^\top \mathbf{X}}{\mathbf{p}_3^\top \mathbf{X}} \quad \Rightarrow \quad u_1 \mathbf{p}_3^\top \mathbf{X} - \mathbf{p}_1^\top \mathbf{X} = 0$$



Compute 3D model

- Scene point X is observed by two cameras $\mathtt{P}$ and $\mathtt{Q}$.
- Let $\mathbf{u} = [u_1\ u_2]^\top$ and $\mathbf{v} = [v_1\ v_2]^\top$ be a correspondence pair (i.e. projections of X in $\mathtt{P}$ and $\mathtt{Q}$).
- Then

$$u_1 = \frac{\mathbf{p}_1^\top \mathbf{X}}{\mathbf{p}_3^\top \mathbf{X}} \quad \Rightarrow \quad u_1 \mathbf{p}_3^\top \mathbf{X} - \mathbf{p}_1^\top \mathbf{X} = 0$$

- and similarly ...

$$u_2 = \frac{\mathbf{p}_2^\top \mathbf{X}}{\mathbf{p}_3^\top \mathbf{X}} \quad \Rightarrow \quad u_2 \mathbf{p}_3^\top \mathbf{X} - \mathbf{p}_2^\top \mathbf{X} = 0$$

$$v_1 = \frac{\mathbf{q}_1^\top \mathbf{X}}{\mathbf{q}_3^\top \mathbf{X}} \quad \Rightarrow \quad v_1 \mathbf{q}_3^\top \mathbf{X} - \mathbf{q}_1^\top \mathbf{X} = 0$$

$$v_2 = \frac{\mathbf{q}_2^\top \mathbf{X}}{\mathbf{q}_3^\top \mathbf{X}} \quad \Rightarrow \quad v_2 \mathbf{q}_3^\top \mathbf{X} - \mathbf{q}_2^\top \mathbf{X} = 0$$

Compute 3D model

- Which is 4×4 homogeneous system of linear equations:

$$\begin{bmatrix} u_1 \mathbf{p}_3^\top - \mathbf{p}_1^\top \\ u_2 \mathbf{p}_3^\top - \mathbf{p}_2^\top \\ v_1 \mathbf{q}_3^\top - \mathbf{q}_1^\top \\ v_2 \mathbf{q}_3^\top - \mathbf{q}_2^\top \end{bmatrix} \mathbf{X} = \mathbf{0}$$

Algorithm at glance

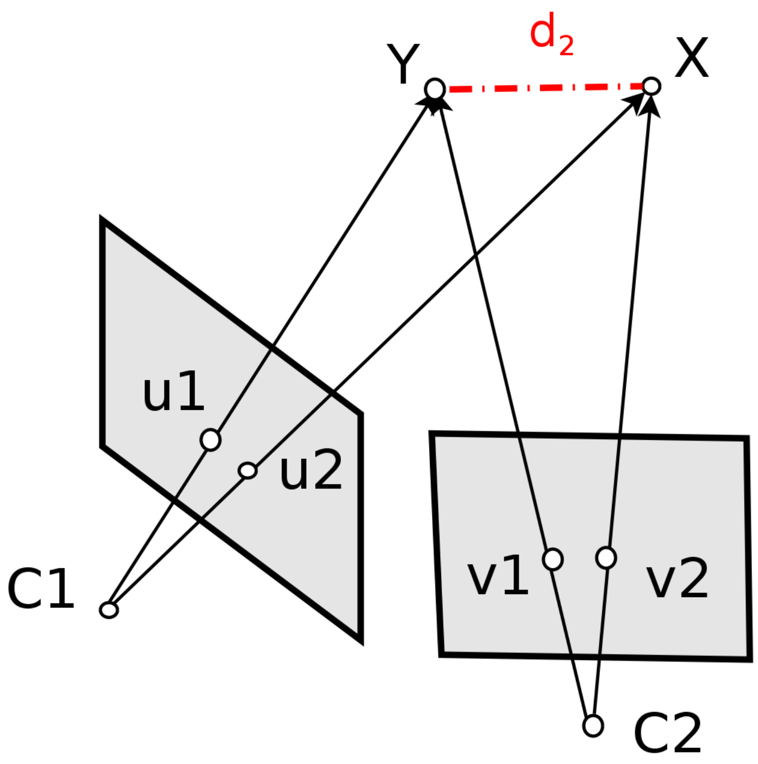
1. Get image I_k.
2. Compute correspondences between I_{k-1} and I_k (either feature matching or tracking).
3. Find correct correspondences and compute essential matrix $\mathtt{E}$.
4. Decompose $\mathtt{E}$ into $\mathtt{R}_k$ and $\mathbf{t}_k$.
5. Compute 3D model (points X).
6. Rescale t_k according to relative scale r.
7. $k = k + 1$

Estimating camera motion - relative scale

1. You cannot get absolute scale (without a calibration object).

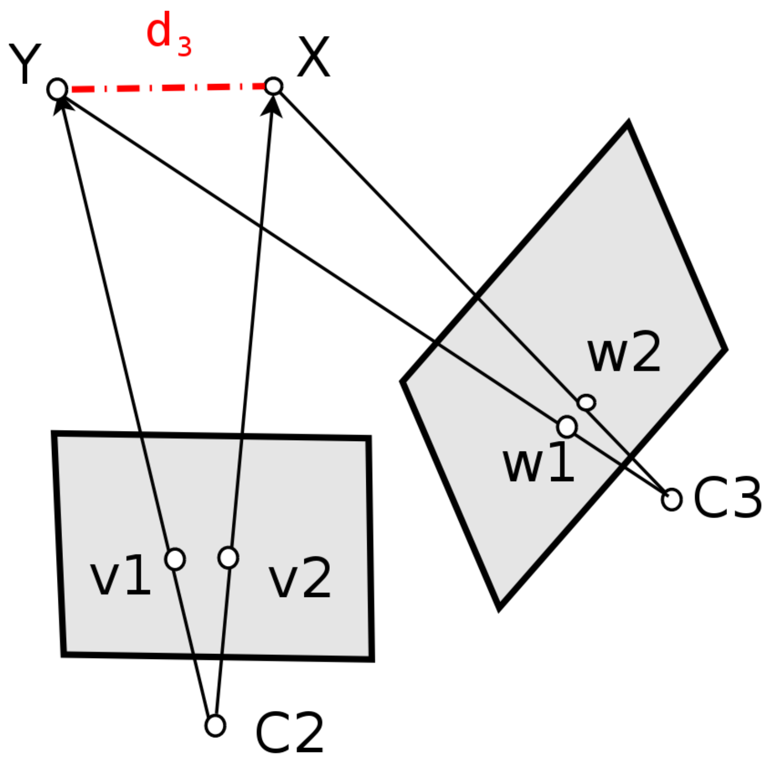
Estimating camera motion - relative scale

1. You cannot get absolute scale (without a calibration object).
2. If you estimate motion (and 3D model) from C_1, C_2



Estimating camera motion - relative scale

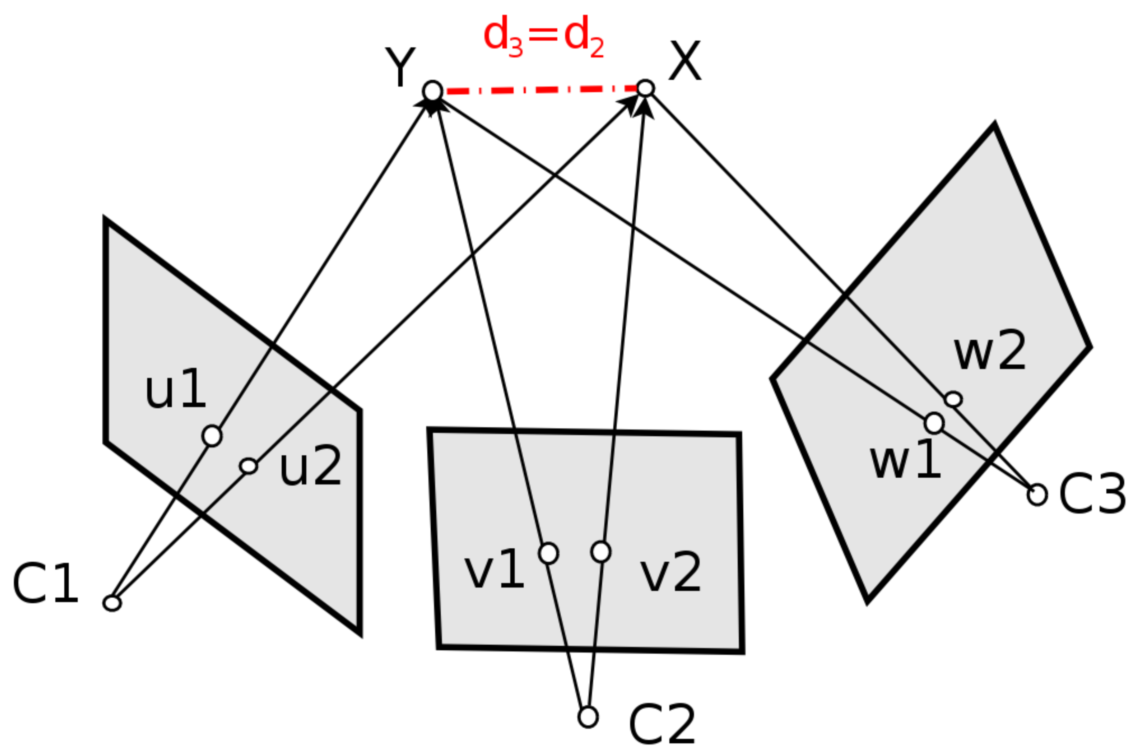
1. You cannot get absolute scale (without calibration object).
2. If you estimate motion (and 3D model) from C_1, C_2 and then from C_2, C_3 you can have completely different scale.



Estimating camera motion - relative scale

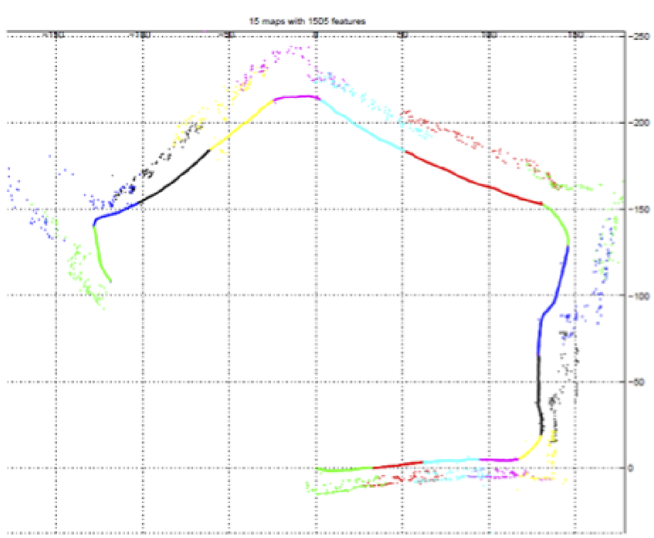
1. You cannot get absolute scale (without calibration object).
2. If you estimate motion (and 3D model) from C_1, C_2 and then from C_2, C_3 you can have completely different scale.
3. You want to keep the same relative scale r by rescaling $\mathbf{t}$ (and 3D)

$$r = \frac{d_k}{d_{k-1}} = \frac{\|X_k - Y_k\|}{\|X_{k-1} - Y_{k-1}\|}$$

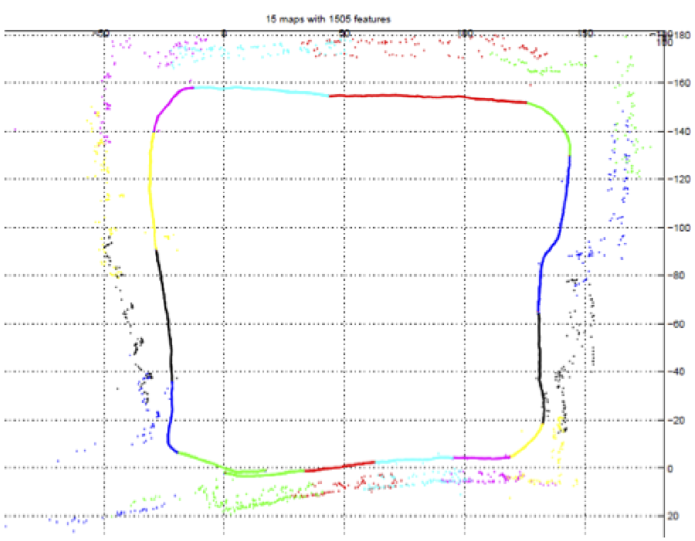


What we did not speak about.

- Result is usually improved by gradient descent of the reprojection error (bundle adjustment).
- Error accumulates over time $\Rightarrow$ drift $\Rightarrow$ loop-closure needed.
- Avoid motion estimation for small motions or pure rotation (keyframe detection)
- Single camera is usually fused with IMU (e.g. Google project Tango).
- Many papers about clever similarity measure for tentative correspondences.



Before loop closing



After loop closing