Geometry for Computer Vision

Lecture 8

Rolling shutter and push-broom cameras: geometry and calibration

Per-Erik Forssén

Overview

- What is a rolling shutter camera?
- Geometric modelling
- Readout time calibration

What is wrong?

What is wrong?

- Hand held ⇒
  non-smooth
  camera path
- Geometric distortions
  (wobble)
- HTC desire
  (Q2 2010)

Rolling shutter rectification

To correct the video, both effects need to be considered:

- Camera Motion
- Geometric Distortion

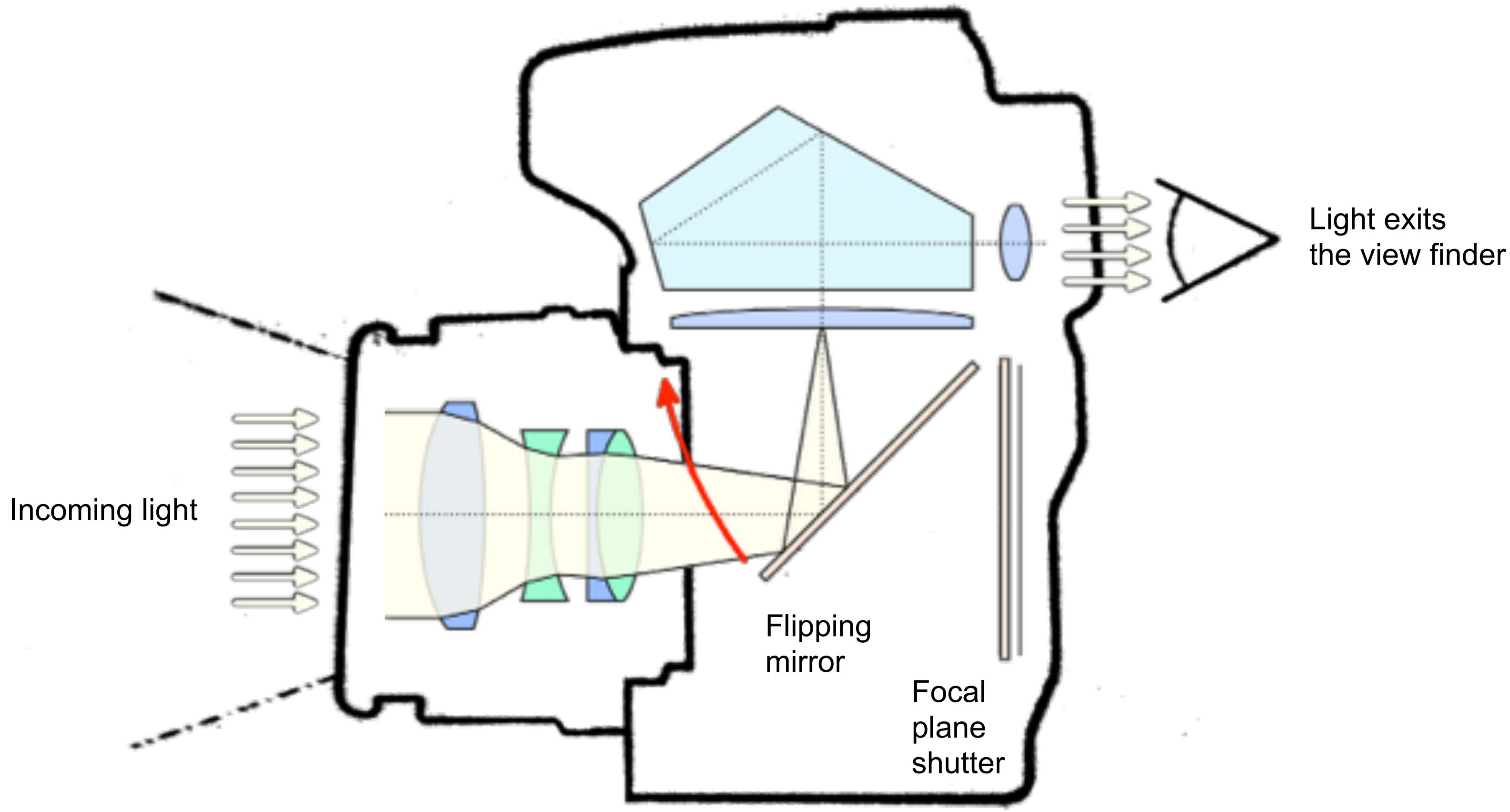
Some current cameras

€3000

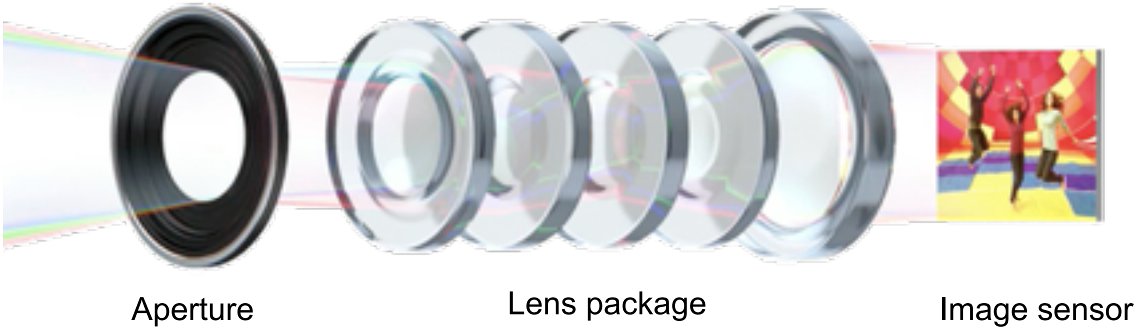
€120

€10

Single lens reflex camera



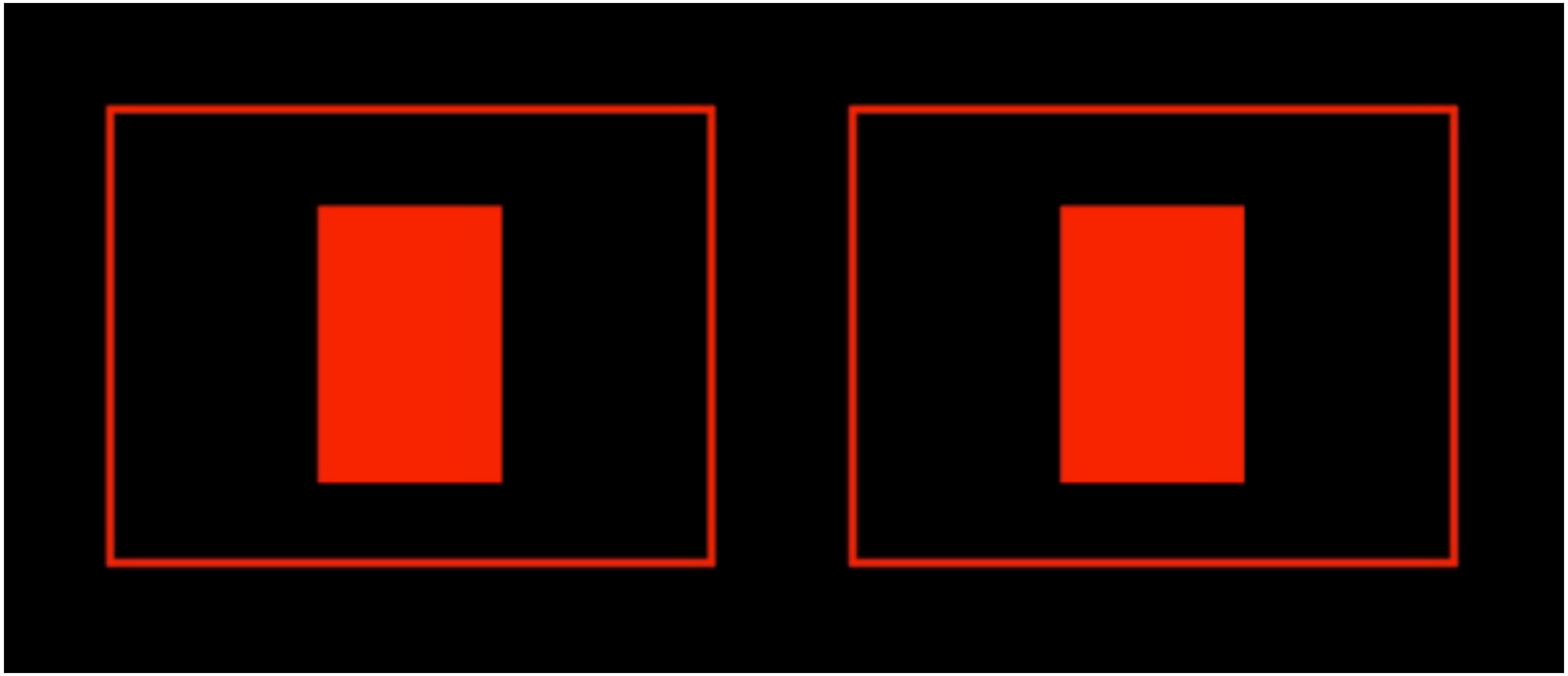
Wafer camera



- The shutter is electronic instead of mechanical!

What is a rolling shutter?

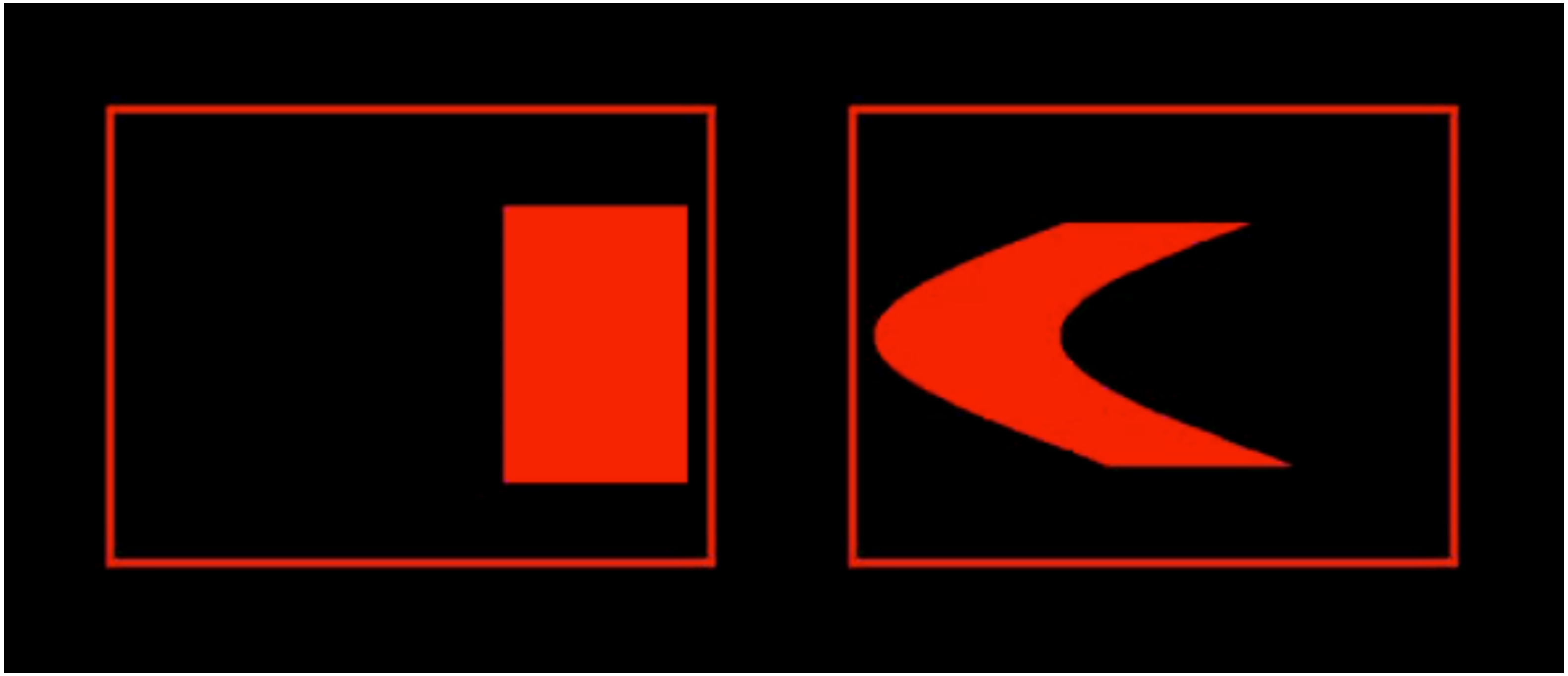
- With a rolling shutter camera, rows are exposed sequentially



Static Scene **Captured Image**

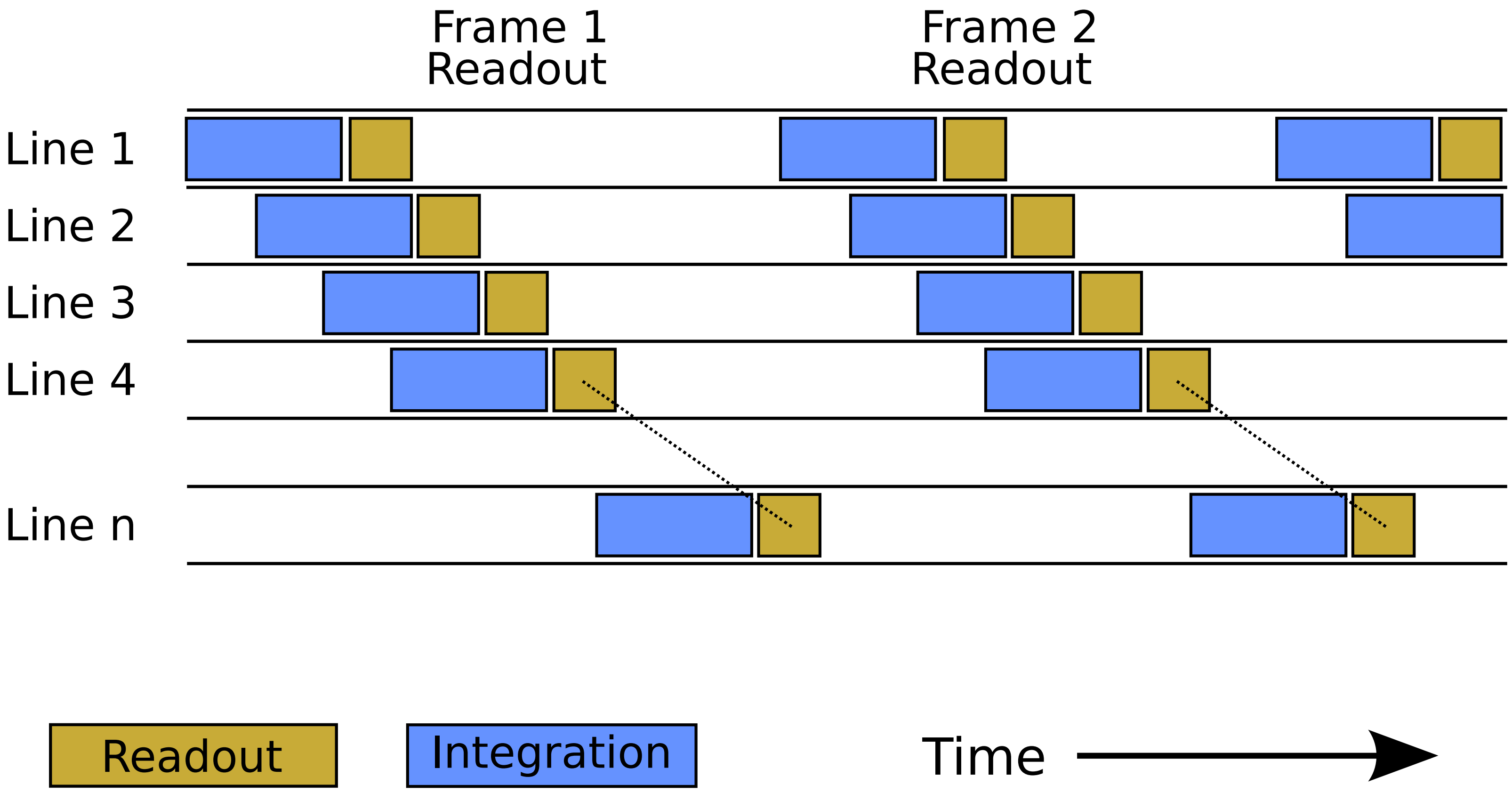
What is a rolling shutter?

- With a rolling shutter camera, rows are exposed sequentially

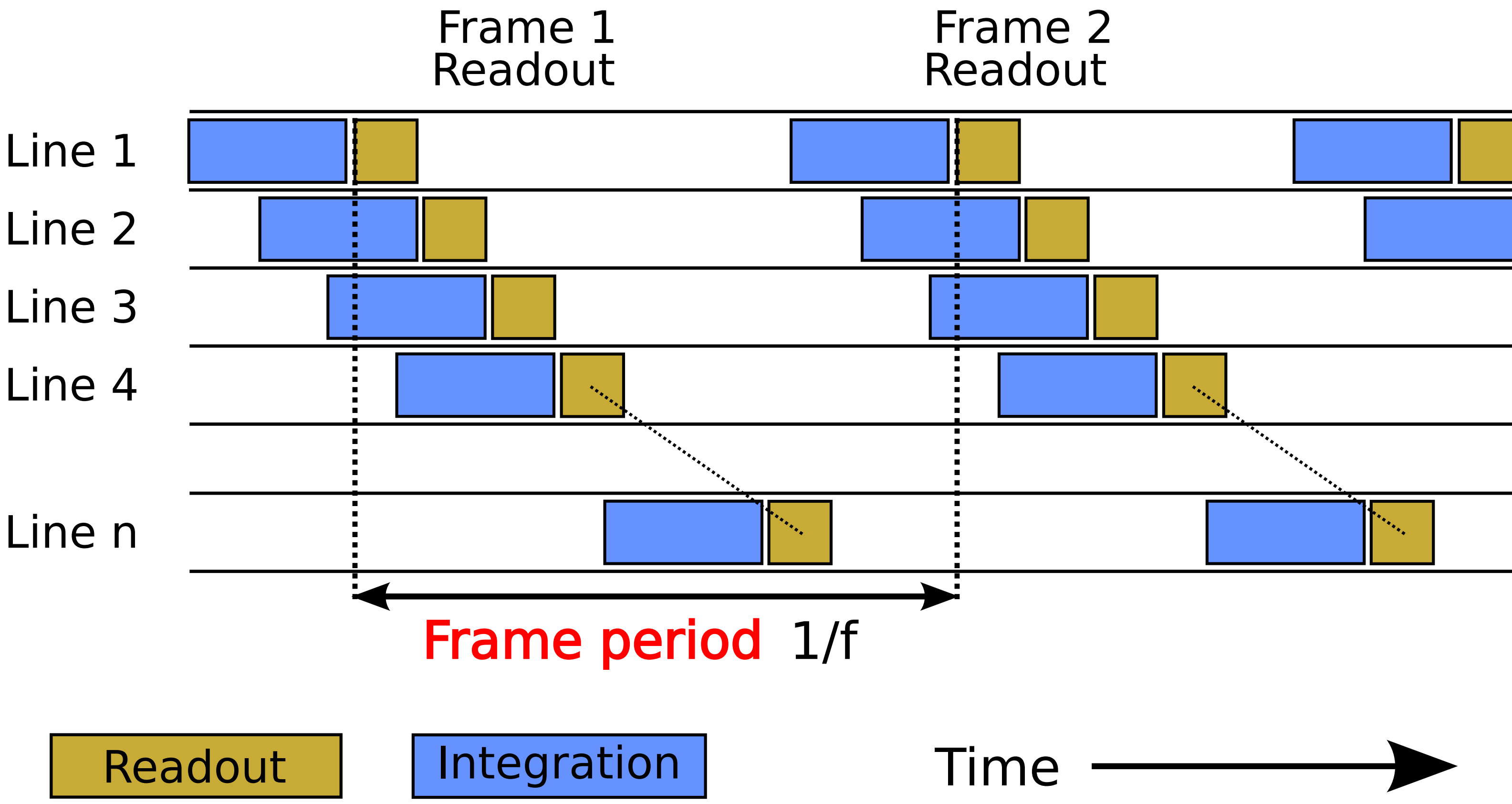


Dynamic Scene **Captured Image**

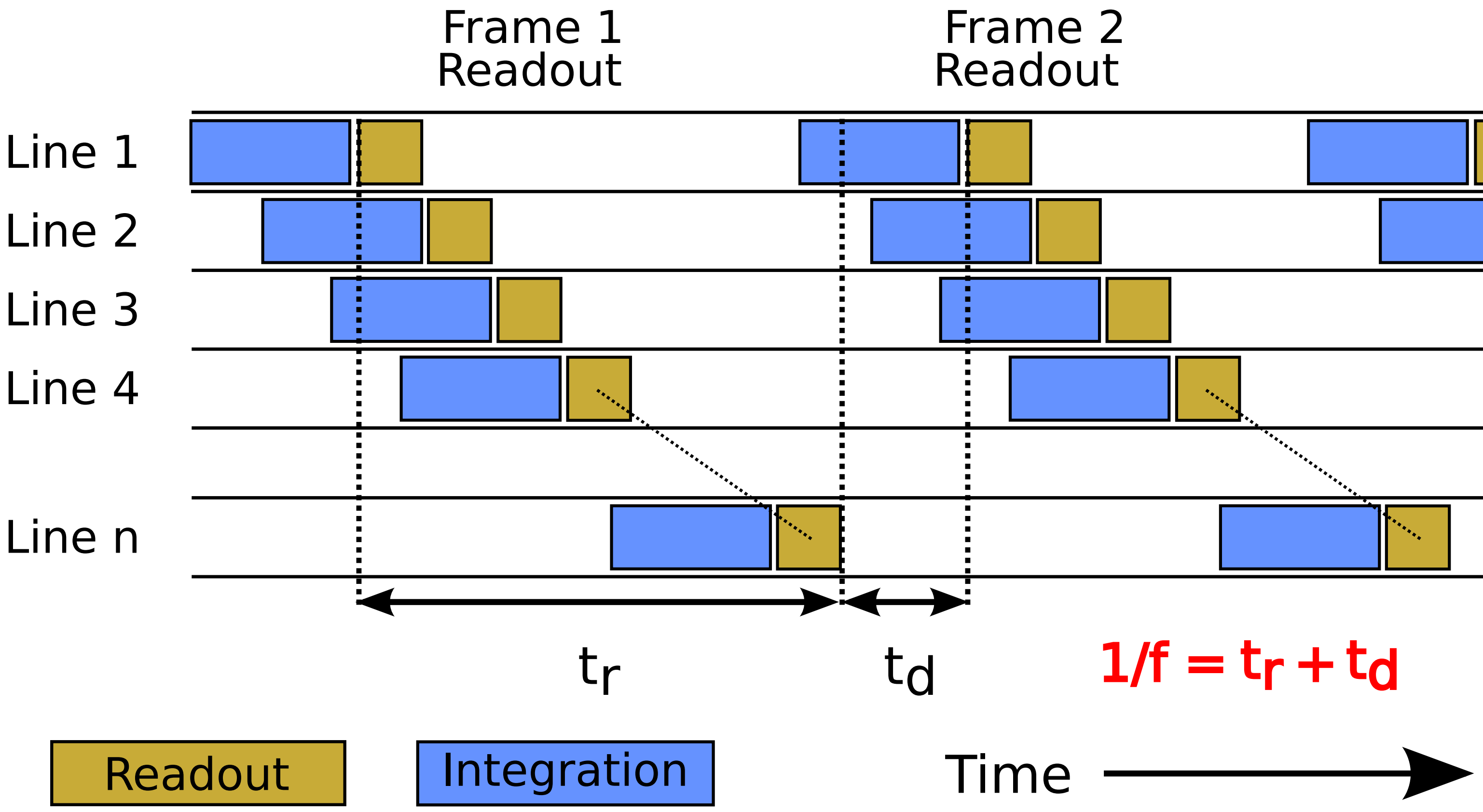
What is a rolling shutter?



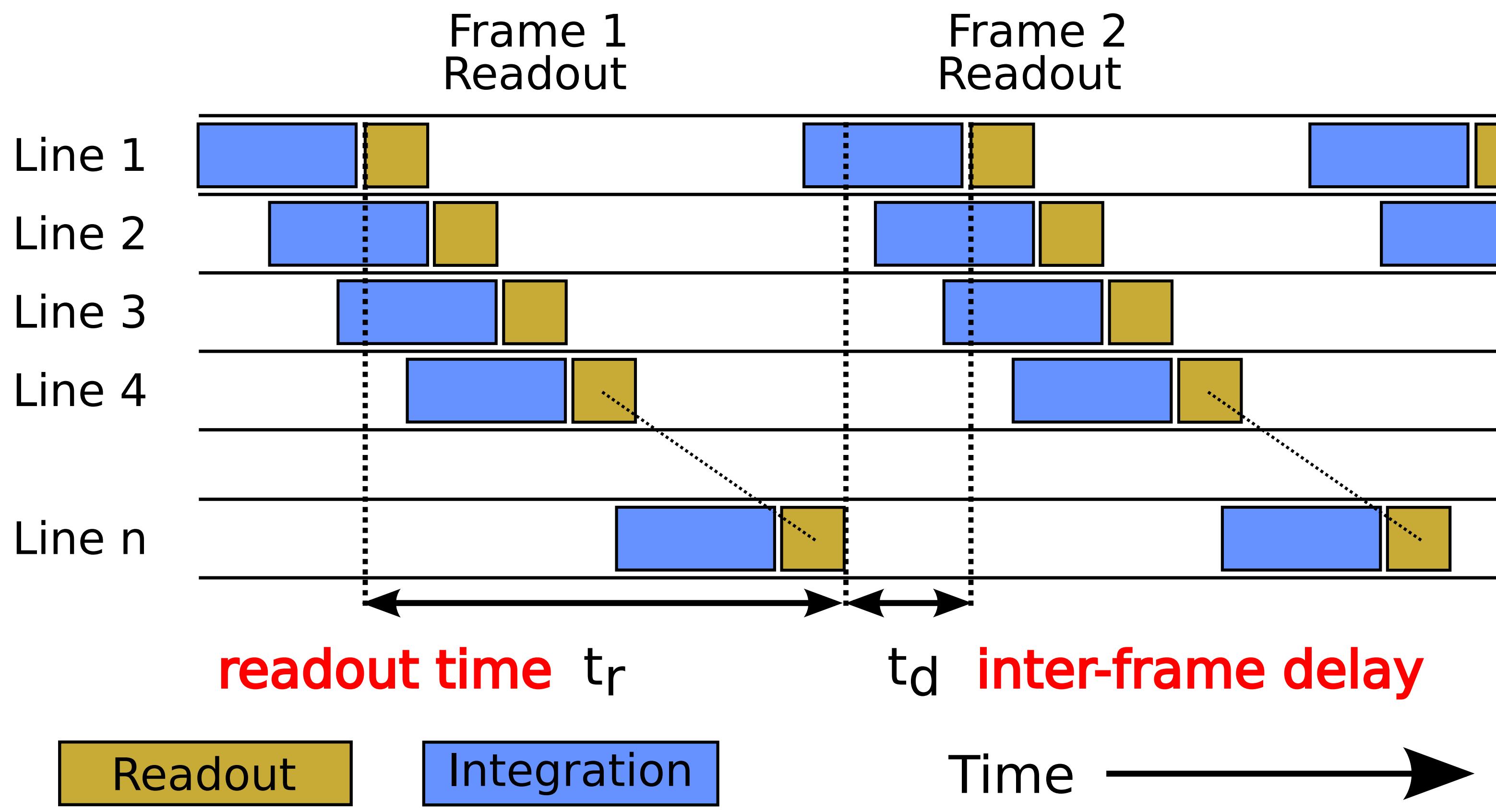
What is a rolling shutter?



What is a rolling shutter?

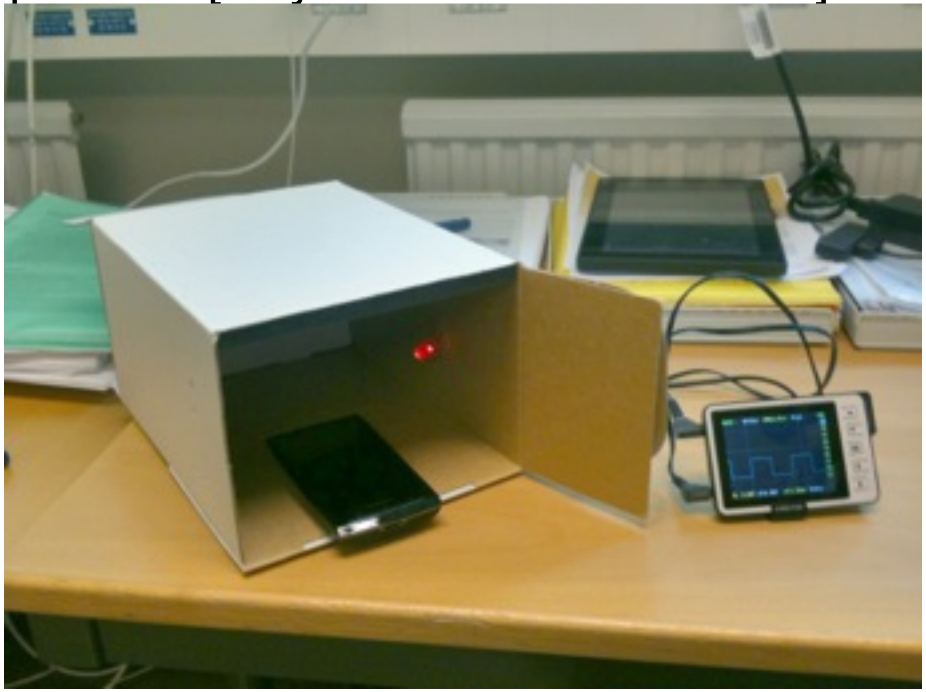


What is a rolling shutter?



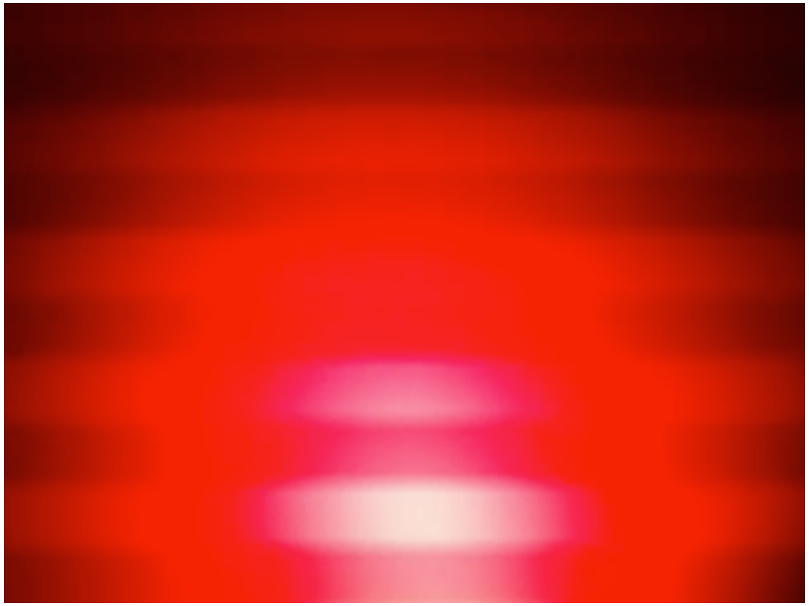
Sensor readout times

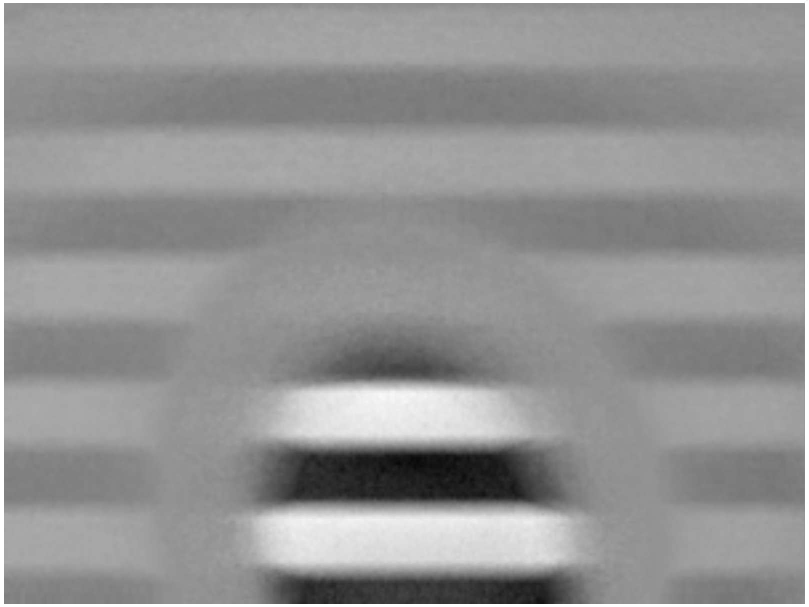
We obtain the readout time as $t_r = N_r/(Tf_o)$ by imaging a flashing LED with known frequency f_o and measuring the imaged period T [Geyer et al. OmniVis 2005]



Sensor readout times

We obtain the readout time as $t_r = N_r/(Tf_o)$ by imaging a flashing LED with known frequency f_o and measuring the imaged period T [Ringaby & Forssén IJCV 2012]





Sensor readout times

Device	framerate	Released	readout
GoProHD Hero	59.94fps	Fall 2009	16.22 msec
Kinect RGB	30fps	Nov 2010	26.11 msec
Kinect NIR	29.97fps	Nov 2010	30.55 msec
iPhone 4s	30fps	Oct 2011	22.08 msec
AR drone v2	30fps	June 2012	24 msec




Summary of the RS situation

- Rolling shutter cameras are everywhere
- A rolling shutter degrades all kinds of geometric computer vision
- A mechanical shutter solves the RS problem
- The readout time is a new camera parameter, that determines the rolling shutter speed.

The pin-hole camera

Recap:

The camera projection operator P has the explicit form:

$$\mathbf{P} = \mathbf{K}\left[\mathbf{R}^T | -\mathbf{R}^T\mathbf{d}\right] = \mathbf{K}\mathbf{R}^T\left[\mathbf{I} | -\mathbf{d}\right]$$

K is the 3x3 intrinsic camera matrix

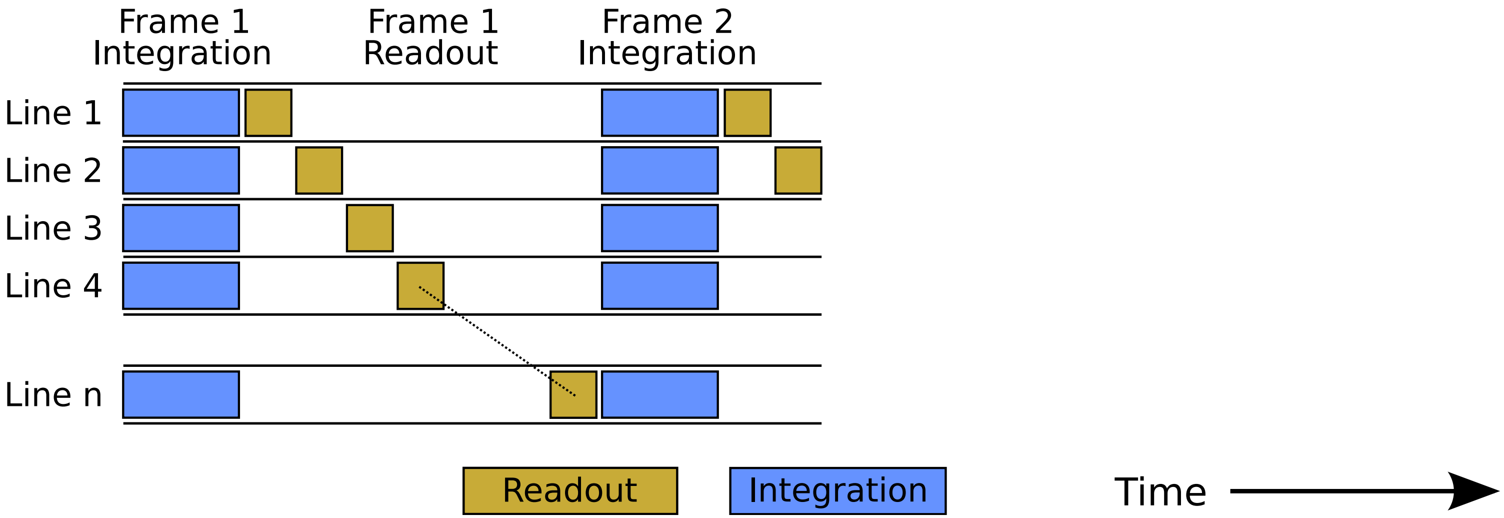
d is a translation of the origin, and R is a 3D rotation

Rolling shutter model

For a moving camera, projection in frame k becomes:

$$\mathbf{x}_k \sim \mathbf{K}\mathbf{R}_k^T [\mathbf{I}| - \mathbf{d}_k]\, \mathbf{X}$$

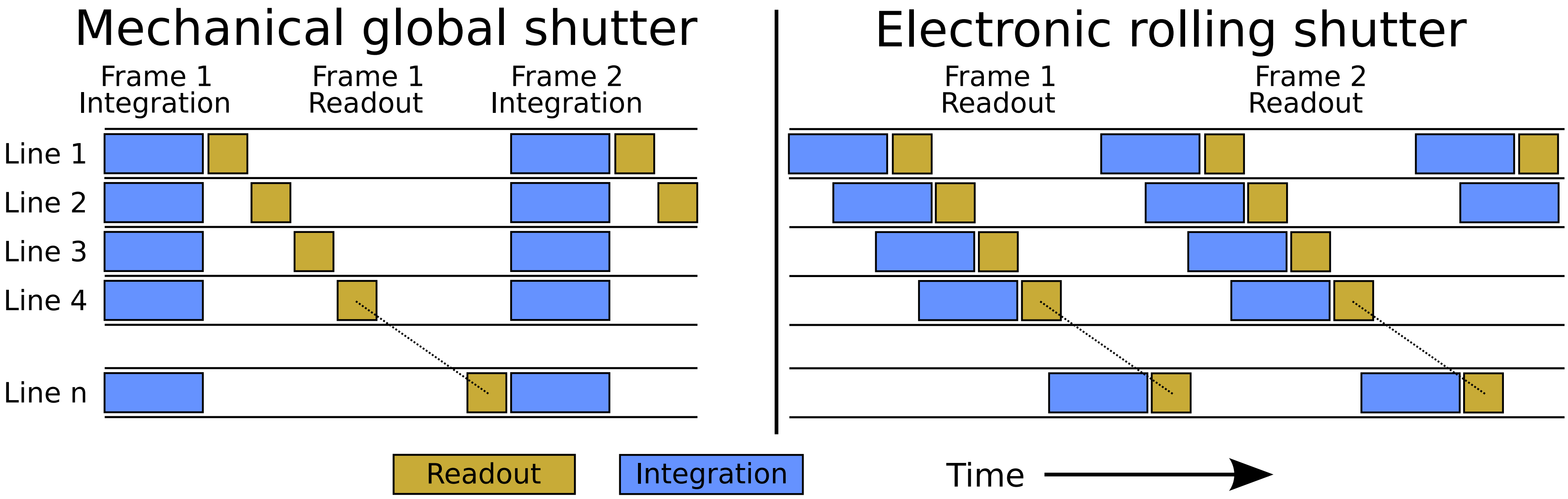
Mechanical global shutter



Rolling shutter model

For a moving camera, projection in frame k becomes:

$$\mathbf{x}_k \sim \mathbf{K}\mathbf{R}_k^T [\mathbf{I}| - \mathbf{d}_k]\, \mathbf{X} \qquad \mathbf{x}_k \sim \mathbf{K}\mathbf{R}(t_{\mathbf{x}})^T [\mathbf{I}| - \mathbf{d}(t_{\mathbf{x}})]\mathbf{X}$$



Rolling shutter model

For a moving camera, projection in frame k becomes:

$$\mathbf{x}_k \sim \mathbf{K}\mathbf{R}_k^T[\mathbf{I}|-\mathbf{d}_k]\,\mathbf{X} \qquad \mathbf{x}_k \sim \mathbf{K}\mathbf{R}(t_{\mathbf{x}})^T[\mathbf{I}|-\mathbf{d}(t_{\mathbf{x}})]\mathbf{X}$$

One pose per line! For tractability, we need to parameterise **d** and **R**

The simplest option is to assume linear changes according to the image row index.

$$\mathbf{d}(t_{\mathbf{x}}) = \mathbf{d}_0(1-\lambda) + \mathbf{d}_1\lambda \quad \lambda = (t_{\mathbf{x}} - t_0)/(t_1 - t_0)$$

…and similarly with SLeRP for the rotation (lecture 7).

More advanced modelling requires the use of splines (see lecture 7)

Rolling shutter model

For a moving camera, projection in frame k becomes:

$$\mathbf{x}_k \sim \mathbf{K}\mathbf{R}_k^T[\mathbf{I}|-\mathbf{d}_k]\,\mathbf{X} \qquad \mathbf{x}_k \sim \mathbf{K}\mathbf{R}(t_\mathbf{x})^T[\mathbf{I}|-\mathbf{d}(t_\mathbf{x})]\mathbf{X}$$

One pose per line! For tractability, we need to parameterise **d** and **R**

The simplest option is to assume linear changes according to the image row index.

$$\mathbf{d}(t_\mathbf{x}) = \mathbf{d}_0(1-\lambda) + \mathbf{d}_1\lambda \quad \lambda = (t_\mathbf{x} - t_0)/(t_1 - t_0)$$

In practise one can compute the projection time from the row index:

$$t_\mathbf{x} - t_0 = x_2/N_r \cdot t_r/T$$

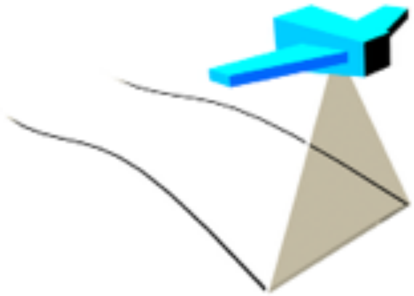
Push-Broom Sensors

A push broom

Push-Broom Sensors

A push broom

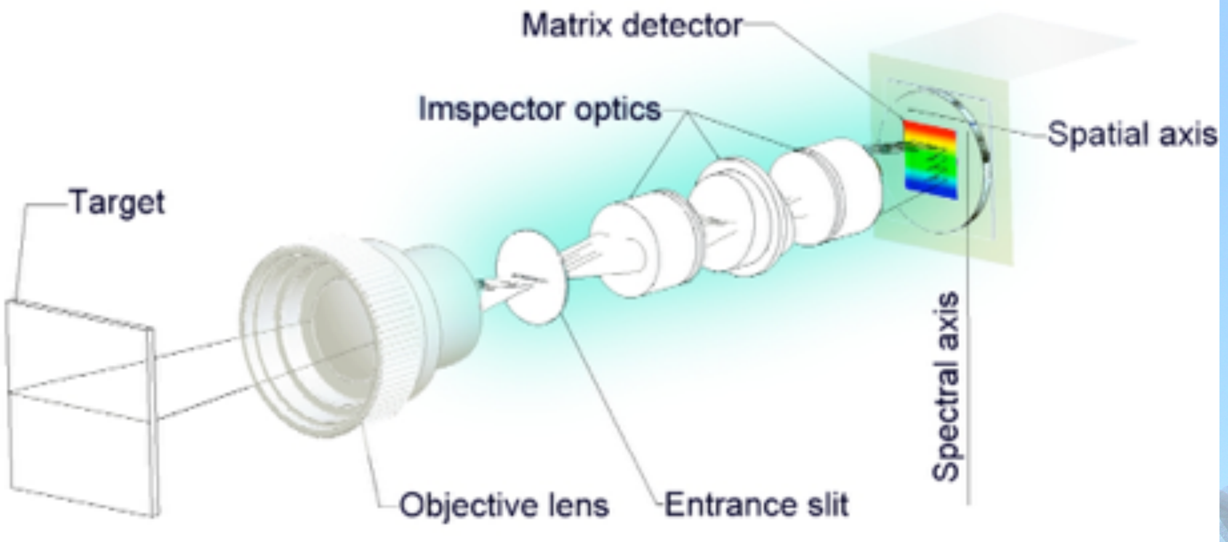
=



A push-broom sensor is a 1D image sensor that acquires 2D images by moving.

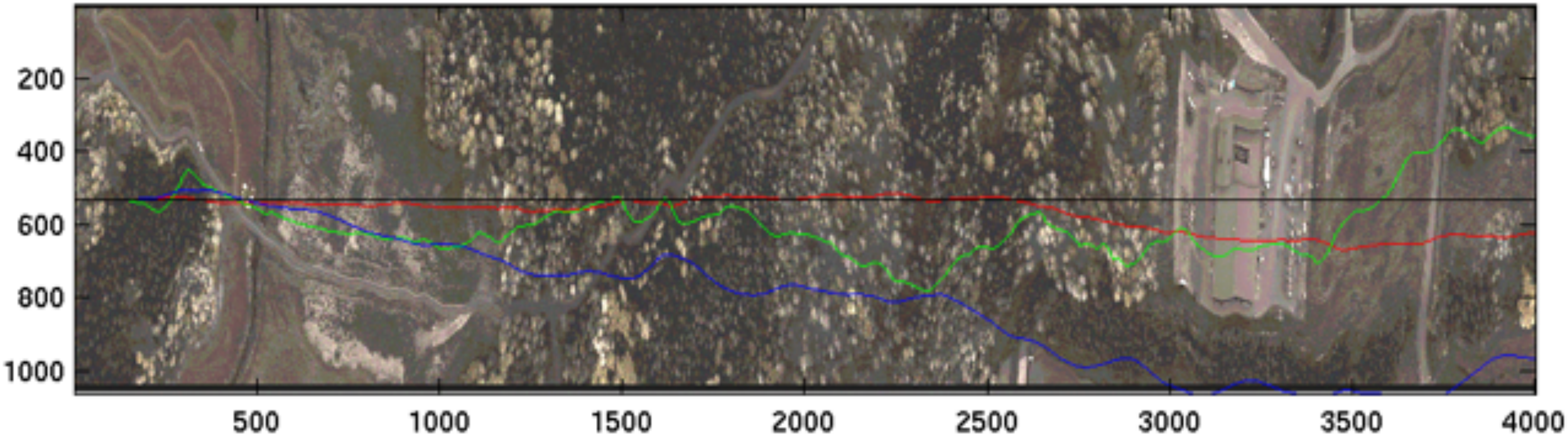
Push-Broom Sensors

Example: Imspec sensor used at FOI



Push-Broom Sensors

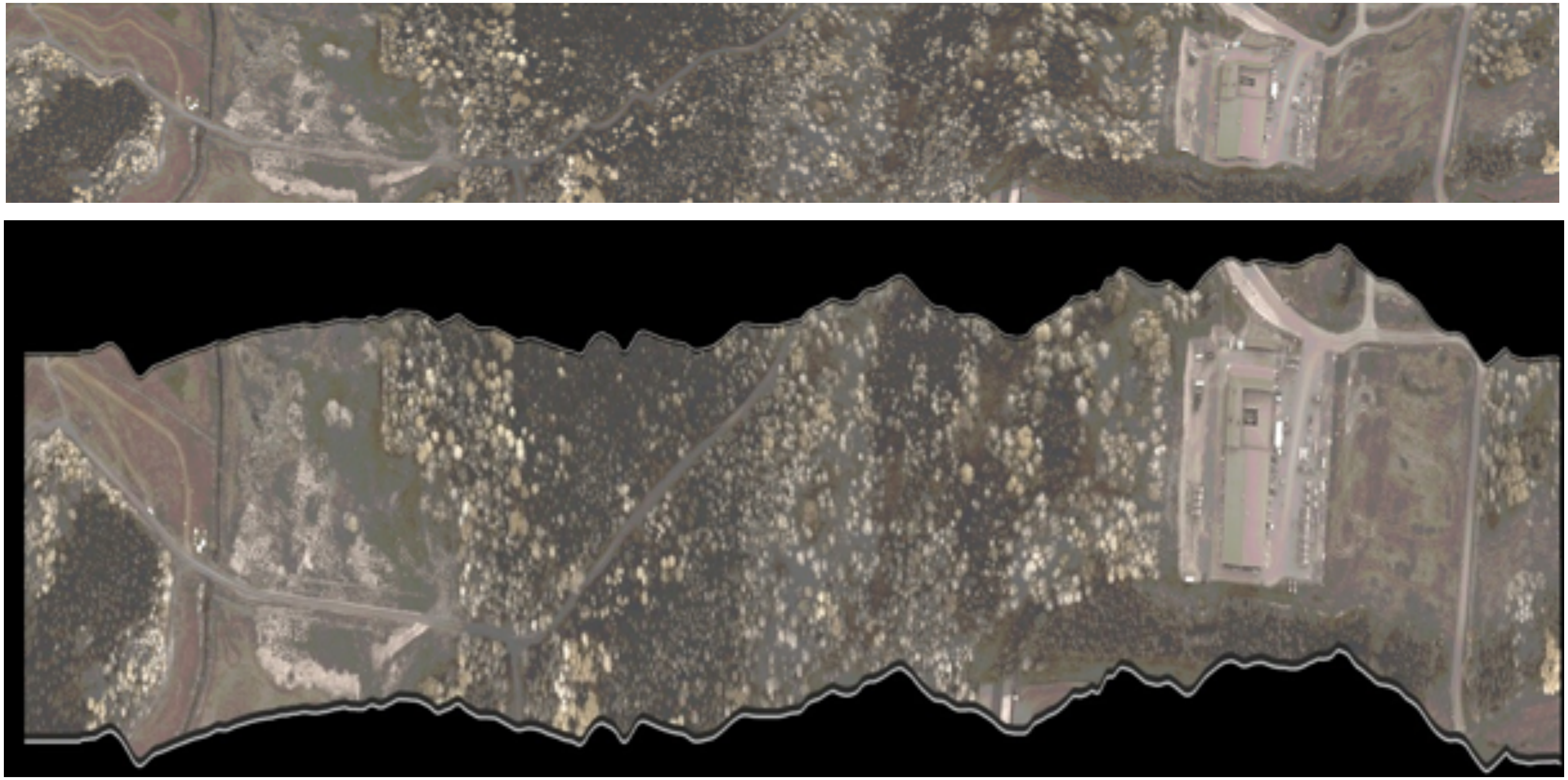
3 or ~60 output bands from sensor, registered with a 3dof gyro signal



Data from FOI Sensor systems

Push-Broom Sensors

Gyro based compensation (rotation only)



Push-broom camera model

- Push-broom geometry can be viewed as a special case of rolling shutter geometry
- With a PB sensor, only one (very long) image is acquired
- With a general RS video camera, many frames in sequence are captured.

Rolling shutter stereo

If we observe a point in two views, we can do triangulation (if motion is known)

$$\begin{aligned}\mathbf{x}_1 &\sim \mathbf{K}_1[\mathbf{R}(\tau_1)|\mathbf{t}(\tau_1)]\mathbf{X} \\ \mathbf{x}_2 &\sim \mathbf{K}_2[\mathbf{R}(\tau_2)|\mathbf{t}(\tau_2)]\mathbf{X}\end{aligned} \quad \Rightarrow \quad \begin{aligned}\mathbf{0} &\sim \mathbf{x}_1 \times \mathbf{P}_1\mathbf{X} \\ \mathbf{0} &\sim \mathbf{x}_2 \times \mathbf{P}_2\mathbf{X}\end{aligned}$$

Rolling shutter stereo

If we observe a point in two views, we can do triangulation (if motion is known)

$$\begin{aligned} \mathbf{x}_1 &\sim \mathbf{K}_1[\mathbf{R}(\tau_1)|\mathbf{t}(\tau_1)]\mathbf{X} \\ \mathbf{x}_2 &\sim \mathbf{K}_2[\mathbf{R}(\tau_2)|\mathbf{t}(\tau_2)]\mathbf{X} \end{aligned} \quad \Rightarrow \quad \begin{aligned} \mathbf{0} &\sim \mathbf{x}_1 \times \mathbf{P}_1\mathbf{X} \\ \mathbf{0} &\sim \mathbf{x}_2 \times \mathbf{P}_2\mathbf{X} \end{aligned}$$

3D SaM from a rolling-shutter image pair is possible, using bundle adjustment:
[**Ait-Aider&Berry ICCV09**]

$$J(\{\mathbf{X}_n\}_{n=1}^N, \mathbf{R}, \mathbf{t}) = \sum_{n=1}^{N} ||\mathbf{x}_{1,n} - \hat{\mathbf{x}}_{1,n}||^2 + ||\mathbf{x}_{2,n} - \hat{\mathbf{x}}_{2,n}||^2$$

Rolling shutter stereo

If we observe a point in two views, we can do triangulation (if motion is known)

$$\begin{aligned}\mathbf{x}_1 &\sim \mathbf{K}_1[\mathbf{R}(\tau_1)|\mathbf{t}(\tau_1)]\mathbf{X} \\ \mathbf{x}_2 &\sim \mathbf{K}_2[\mathbf{R}(\tau_2)|\mathbf{t}(\tau_2)]\mathbf{X}\end{aligned} \quad \Rightarrow \quad \begin{aligned}\mathbf{0} &\sim \mathbf{x}_1 \times \mathbf{P}_1\mathbf{X} \\ \mathbf{0} &\sim \mathbf{x}_2 \times \mathbf{P}_2\mathbf{X}\end{aligned}$$

Each correspondence gives us 4 equations (Why?)

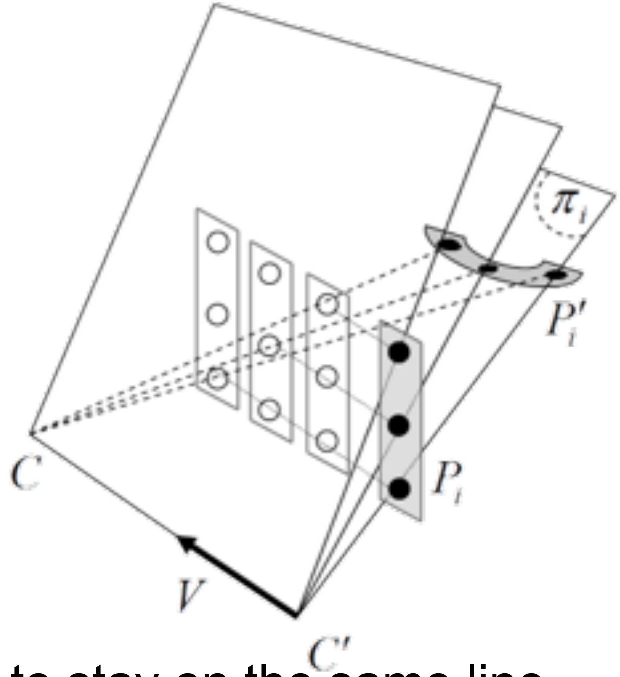
Assuming that **R**,**t** change linearly with time, we have 5+3N unknowns for N correspondences. Thus we need

$$4N \geq 5 + 3N \quad \Rightarrow$$

$$N \geq 5$$

Rolling shutter stereo

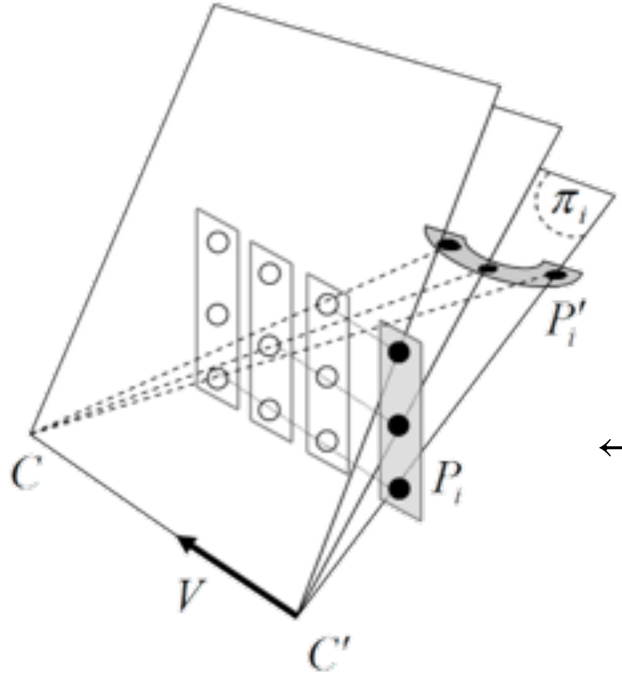
Degenerate motion [**Ait-Aider&Berry ICCV09**]:



Motion causes points to stay on the same line.

Rolling shutter stereo

Degenerate motion [**Ait-Aider&Berry ICCV09**]:



⟵Equivalent stationary structure

⟵Translating structure

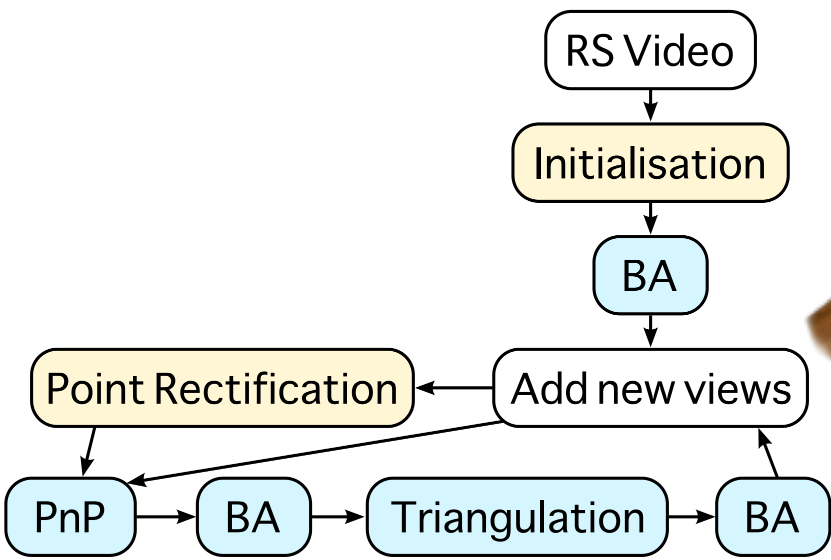
Illustration by Ait-Aider and Berry

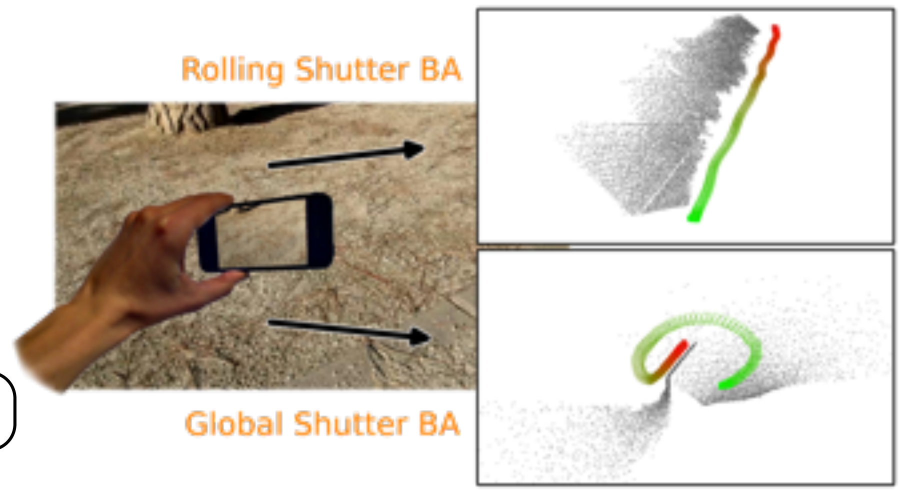
Rolling shutter stereo

- Under degeneracy, motion and structure can be interchanged freely.
- Special case: known motion, no degeneracy.
- If one of the cameras has a global shutter, both structure and motion can be determined [**Ait-Aider&Berry ICCV'09**]
- If multiple frames are used, rolling shutter structure from motion (SfM) is stable in practise [**Hedborg et al. CVPR'12**].

Structure from motion

For RS aware SfM, we define reprojection errors again using interpolated key poses (translations and rotations)





Cost Function, BA

Cameras: $\mathbf{C}_j, j = 1..J$. 3D points: $\mathbf{X}_k,\ k = 1..K$

$$\mathbf{C}_j = \mathbf{R}_j^T[\mathbf{I}| - \mathbf{d}_j]$$

$$\text{proj}(\mathbf{C}_j, \mathbf{X}_k) \text{ where proj} : (\Re^{3\times 4}, \Re^3) \to \Re^2$$

$$\min_{\mathbf{C},\mathbf{X}} \frac{1}{2} \sum_{j=1}^{J} \sum_{k\in\mathcal{V}_j} ||\mathbf{p}_{j,k} - \text{proj}(\mathbf{C}_j, \mathbf{X}_k)||_2^2$$

$\mathcal{V}_j$ is the set of visible points in camera j

Cost Function, RSBA

Cameras: $\mathbf{C}_j(y), j = 1..J$. 3D points: $\mathbf{X}_k,\ k = 1..K$

$$\mathbf{C}_j(y) = \mathbf{R}^T_{j,j+1}(y)[\mathbf{I}| - \mathbf{d}_{j,j+1}(y)]$$

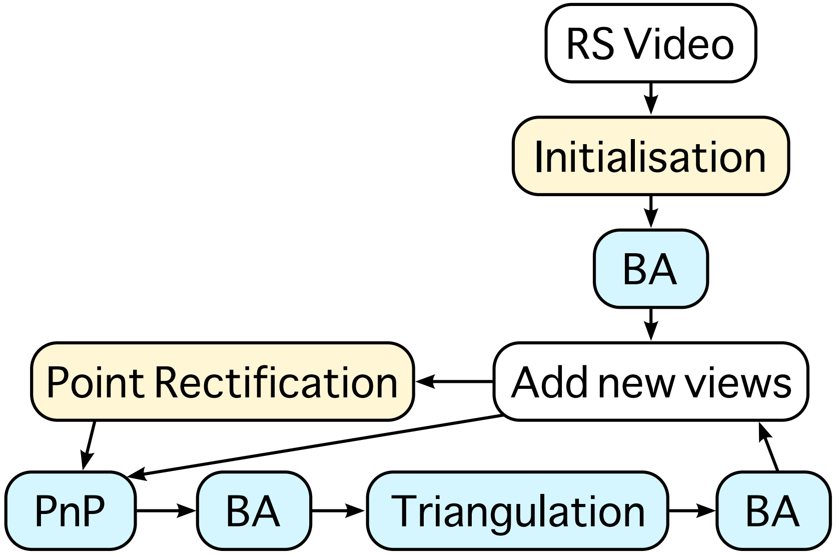
$$\text{proj}(\mathbf{C}_j(y), \mathbf{X}_k) \text{ where proj} : (\Re^{3\times4}, \Re^3) \to \Re^2$$

$$\min_{\mathbf{C},\mathbf{X}} \frac{1}{2} \sum_{j=1}^{J} \sum_{k \in \mathcal{V}_j} ||\mathbf{p}_{j,k} - \text{proj}(\mathbf{C}_j(y_k), \mathbf{X}_k)||_2^2$$

$\mathcal{V}_j$ is the set of visible points in camera j

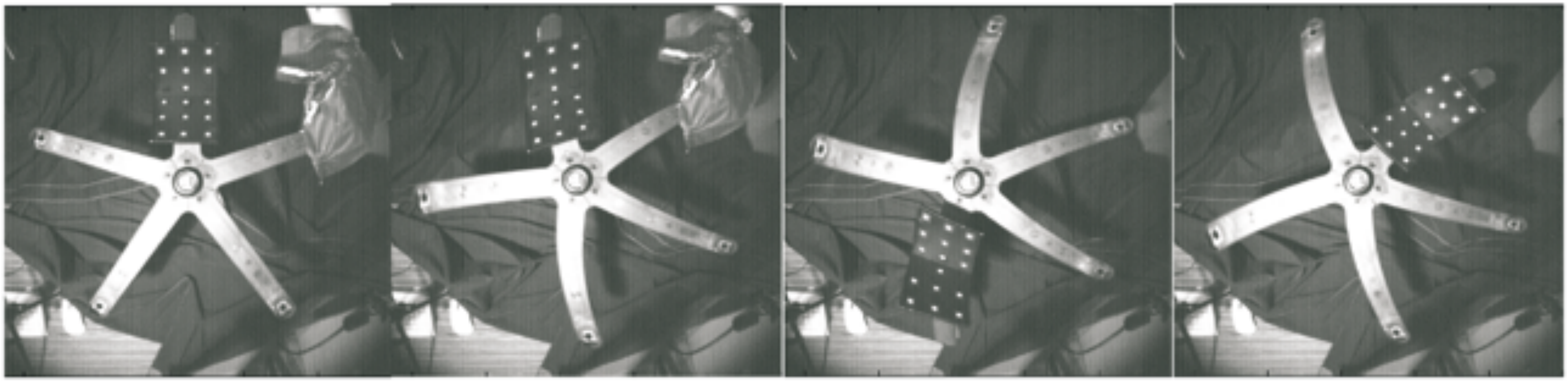
Structure from motion

In RSBA, we also need rolling-shutter aware versions of PnP and Triangulation.



Rolling Shutter PnP

Simultaneous Object Pose and Velocity Computation Using a Single View from a Rolling Shutter Camera
Omar Ait-Aider, Nicolas Andreff, Jean Marc Lavest and Philippe Martinet, **ECCV 2006**



Non-linear least squares on the following cost function:

$$\min_{\mathbf{R},\mathbf{d},\mathbf{\Omega},\mathbf{v}} \sum_{k=1}^{K} ||\mathbf{x}_k, \mathrm{proj}(\mathbf{C}_j(t), \mathbf{X}_k)||^2$$

Rolling Shutter PnP

Parallel Tracking and Mapping on a camera phone
Georg Klein and David Murray, **ISMAR 2009**

- Ported PTAM (parallel tracking and mapping) to the CMOS camera of the Iphone 3G
- System Initialization: find planar structure and do homography estimation
- Then a rolling shutter aware perspective-n-point method

Rolling Shutter PnP

Parallel Tracking and Mapping on a camera phone
Georg Klein and David Murray, **ISMAR 2009**

Solve the velocity of the camera

Then compensate for the RS-distortion

$$\begin{bmatrix} \mathbf{J}_1 \\ \vdots \\ \mathbf{J}_n \end{bmatrix} \dot{\mu} = \begin{bmatrix} \dot{\mathbf{m}}_1 \\ \vdots \\ \dot{\mathbf{m}}_n \end{bmatrix}, \mathbf{m}'_i = \mathbf{m}_i - \mathbf{J}_i \dot{\mu} \delta t$$

Rolling shutter rectification

Rotation homography approximation:

$$\mathbf{x}_1 \sim \mathbf{K}\mathbf{R}(t_1)\mathbf{X} \qquad \qquad \mathbf{x}_1 \sim \mathbf{H}\mathbf{x}_2$$

$$\Rightarrow$$

$$\mathbf{x}_2 \sim \mathbf{K}\mathbf{R}(t_2)\mathbf{X} \qquad \qquad \mathbf{H} \sim \mathbf{K}\mathbf{R}(t_1)\mathbf{R}(t_2)^T\mathbf{K}^{-1}$$

Valid if the distance to imaged objects is large compared to the baseline

Rolling shutter rectification

Rotation homography approximation:

$$\mathbf{x}_1 \sim \mathbf{KR}(t_1)\mathbf{X} \qquad \Rightarrow \qquad \mathbf{x}_1 \sim \mathbf{H}\mathbf{x}_2$$

$$\mathbf{x}_2 \sim \mathbf{KR}(t_2)\mathbf{X} \qquad\qquad \mathbf{H} \sim \mathbf{KR}(t_1)\mathbf{R}(t_2)^T\mathbf{K}^{-1}$$

Valid if the distance to imaged objects is large compared to the baseline

For hand-held motion rotation is typically the dominant source of distortions

Rolling shutter rectification

Rotation homography approximation:

$$\mathbf{x}_1 \sim \mathbf{K}\mathbf{R}(t_1)\mathbf{X} \qquad \Rightarrow \qquad \mathbf{x}_1 \sim \mathbf{H}\mathbf{x}_2$$

$$\mathbf{x}_2 \sim \mathbf{K}\mathbf{R}(t_2)\mathbf{X} \qquad \qquad \mathbf{H} \sim \mathbf{K}\mathbf{R}(t_1)\mathbf{R}(t_2)^T\mathbf{K}^{-1}$$

Valid if the distance to imaged objects is large compared to the baseline

Allows estimation of rotations across a sequence of frames given correspondences, using BA (**see next discussion paper**)

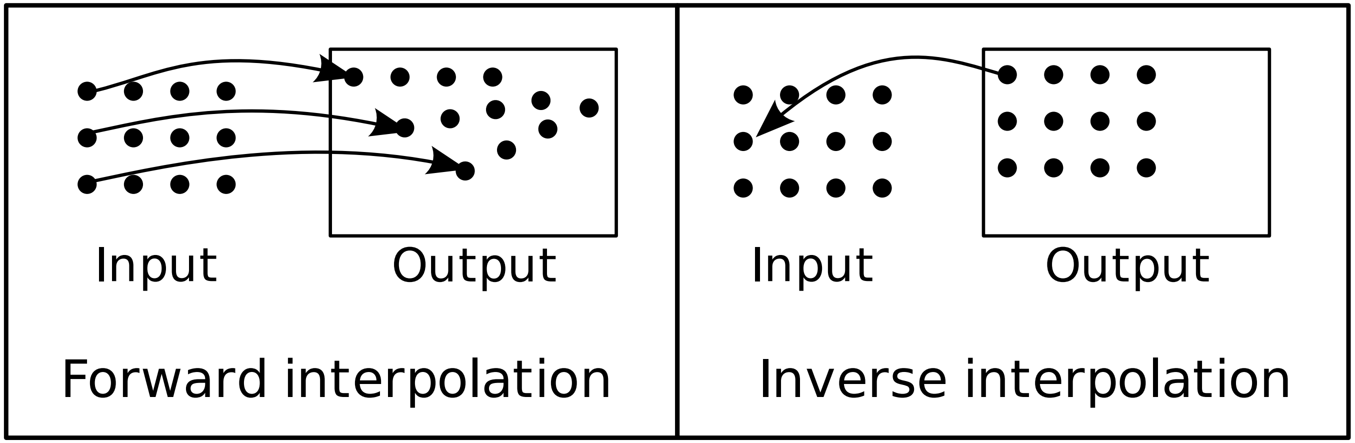
$$J = \sum_{k=1}^{K} d(\mathbf{x}_{1,k}, \mathbf{H}\mathbf{x}_{2,k})^2 + d(\mathbf{x}_{2,k}, \mathbf{H}^{-1}\mathbf{x}_{1,k})^2$$

Rolling shutter rectification

Once we know the rotations **R** and the intrinsics **K**, rectification from a single frame is possible

$$\mathbf{x}' \sim \mathbf{K}\mathbf{R}(t_{\text{mid}})\mathbf{R}(t_{\mathbf{x}})^T\mathbf{K}^{-1}\mathbf{x}$$

This is forward interpolation, which in this case is slightly more accurate than regular inverse interpolation



Rectification

Each line is rectified with a separate homography

$$\mathbf{x}' \sim \mathbf{K}\mathbf{R}(t_{\mathrm{mid}})\mathbf{R}(t_{\mathbf{x}})^T\mathbf{K}^{-1}\mathbf{x}$$

Original

Corrected

Readout calibration

Luc Oth, Paul Furgale, Laurent Kneip, Roland Siegwart, *Rolling Shutter Camera Calibration*, CVPR'13

Checkerboard calibration, using known intrinsics **K**.

$$J(t_r, \mathbf{R}_1, \mathbf{d}_1, \ldots, \mathbf{R}_N, \mathbf{d}_N) = \sum_n \sum_k d^2(\mathbf{x}_{k,n}, \mathbf{K}\mathbf{R}(t_{k,n})^T[\mathbf{I}| - \mathbf{d}(t_{k,n})]\mathbf{X}_k)$$

True reprojection error is used:

- Time at reprojection $t_{k,n}$ insead of
- Time at observation $t_{\mathbf{x}}$

Requires iteration, as camera pose R(t),d(t) now depends on t, and t depends on R(t),d(t) !

Readout calibration

Luc Oth, Paul Furgale, Laurent Kneip, Roland Siegwart, *Rolling Shutter Camera Calibration*, CVPR'13

Checkerboard calibration, using known intrinsics **K**.

$$J(t_r, \mathbf{R}_1, \mathbf{d}_1, \ldots, \mathbf{R}_N, \mathbf{d}_N) = \sum_n \sum_k d^2(\mathbf{x}_{k,n}, \mathbf{K}\mathbf{R}(t_{k,n})^T[\mathbf{I}| - \mathbf{d}(t_{k,n})]\mathbf{X}_k)$$

True reprojection error is used:

- Time at reprojection $t_{k,n}$ insead of
- Time at observation $t_{\mathbf{x}}$

Requires iteration, as camera pose R(t),d(t) now depends on t, and t depends on R(t),d(t) !

Also uses a split criterion, to iteratively add new poses where needed.

Papers to discuss next week…

E. Ringaby and P-.E. Forssén. *Efficient Video Rectification and Stabilisation for Cell-Phones*, IJCV'12