Event-based Robot Vision

Prof. Dr. Guillermo Gallego
Chair: Robotic Interactive Perception

guillermo.gallego@tu-berlin.de

http://www.guillermogallego.es

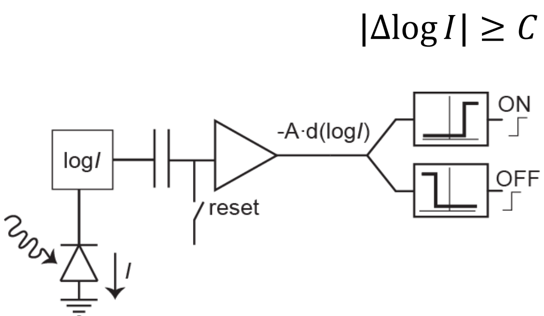
Case study

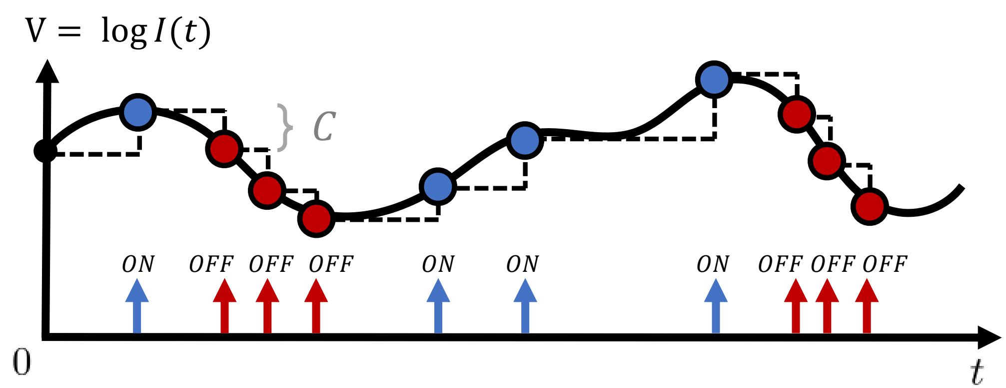
Image reconstruction from events in case of rotating event camera with known motion

Reference: Kim et al. BMVC 2014

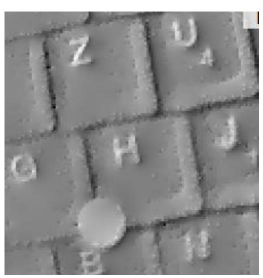
Image Reconstruction

- Recall: Events are generated any time a single pixel sees a change in brightness larger than C





The intensity signal at the event time can be reconstructed by **integration** of $\pm C$



Cook et al., IJCNN 2011

Kim et al., BMVC 2014

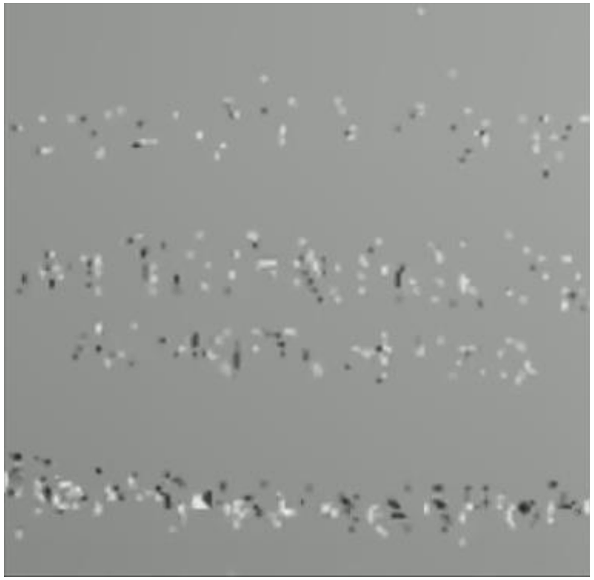
Image reconstruction

Given the events and the camera motion (rotation), recover the absolute brightness

Video by H. Rebecq, based on Kim et al., BMVC 2014

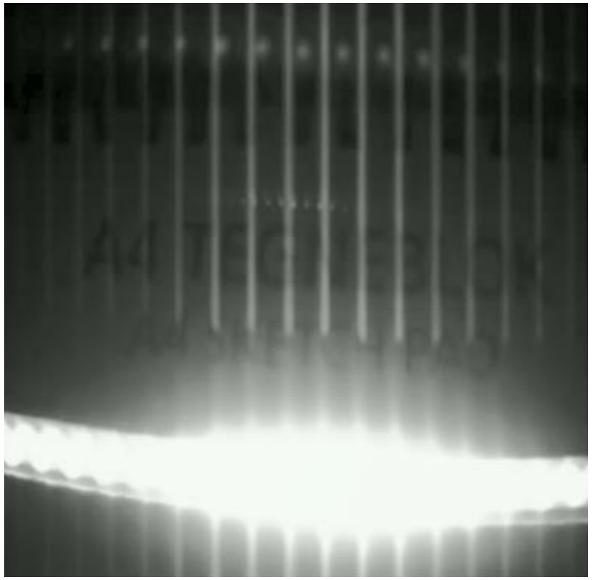
Image reconstruction

High Dynamic Range property comes “for free” with the sensor



Input events

HDR reconstructed intensity



Standard camera
(narrow dynamic range)

Kim et al., [*Simultaneous Mosaicing and Tracking with an Event Camera*](), BMVC 2014. [https://youtu.be/l6qxeM1DbXU]()

Image reconstruction

Given the events and the camera motion (rotation), recover the absolute brightness

- How is it possible?
- Intuitive explanation: an event camera naturally responds to edges, hence, if we know the motion, we can relate the events to "world coordinates" to get an edge/gradient map. Then, integrate the gradient map to get absolute intensity.

Steps:

1. Recover the gradient map of the scene

Events

Camera motion $R(t)$

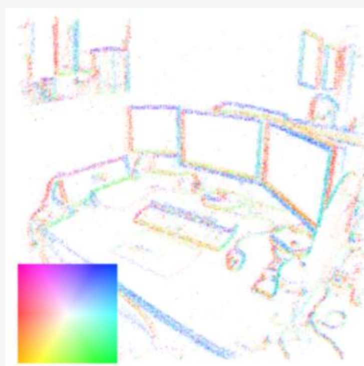


Image gradient
(magnitude and direction)

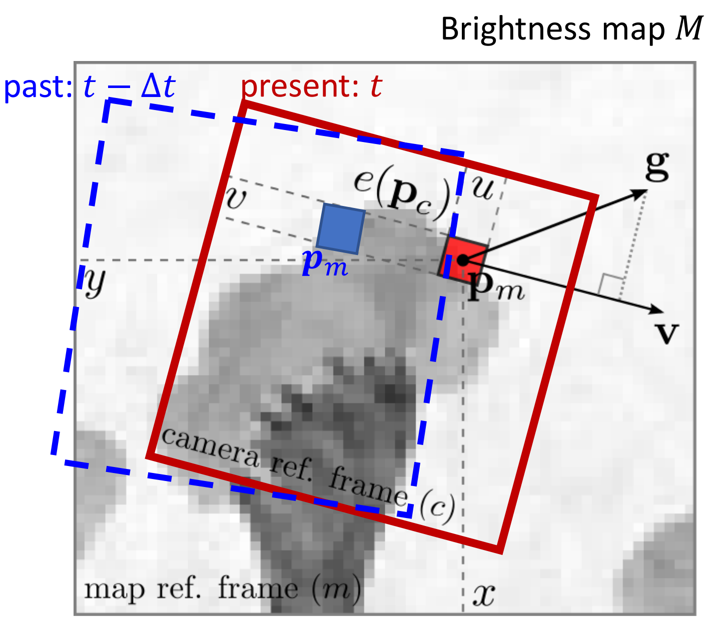
2. Integrate the gradient to obtain brightness

Solve Poisson Eq.

Image intensity

Image reconstruction

Step 1: compute gradient map



Event generated due to brightness change of size C

Let $L = \log I$,

$$\Delta L(t) \equiv L(t) - L(t - \Delta t) = C$$

In terms of the brightness map $M(x, y)$ (panorama):

$$M(\boldsymbol{p}_m(t)) - M\big(\boldsymbol{p}_m(t - \Delta t)\big) = C$$

Using Taylor 1st order approximation:

$$\boxed{M\big(\boldsymbol{p}_m(t)\big) - M\big(\boldsymbol{p}_m(t - \Delta t)\big) \approx \boldsymbol{g} \cdot \boldsymbol{v}\Delta t}$$

where brightness gradient $\boldsymbol{g} = \nabla M\big(\boldsymbol{p}_m(t)\big)$

displacement $\boldsymbol{v}\Delta t = \big(\boldsymbol{p}_m(t) - \boldsymbol{p}_m(t - \Delta t)\big)$

Image reconstruction (detailed)

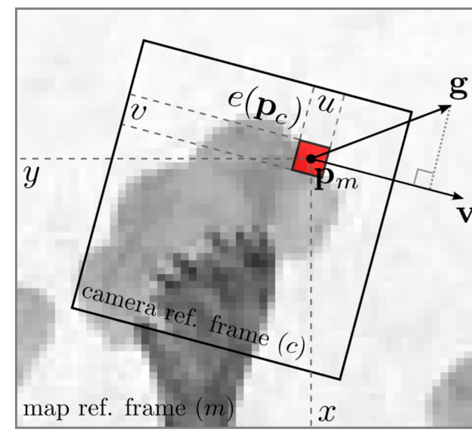
Step 1: compute gradient map

Idea: use the **event generation model** and set up an **EKF for every map pixel** (mosaic)

- **Event generation model**:
  An event is fired at a pixel $\boldsymbol{p}_c$ if there is a brightness change of size C.
  Letting $L = \log I$,
  $$\Delta L(t) \equiv L(t) - L(t - \Delta t) = C$$

- This is an eq. on the image plane.
  Let us **write it in map coordinates**:
  - Let $M(x, y)$ be the brightness of the world map (mosaic)
  - At t, the pixel $\boldsymbol{p}_c$ sees the map point $\boldsymbol{p}_m(t)$,
    where the brightness is $L(t) = M(\boldsymbol{p}_m(t))$
  - At $t - \Delta t$, the pixel $\boldsymbol{p}_c$ saw the point $\boldsymbol{p}_m(t - \Delta t)$,
    where the brightness was $L(t - \Delta t) = M(\boldsymbol{p}_m(t - \Delta t))$

  Hence, the brightness change in terms of the map is:



$$C = \Delta L = \Delta M = M(\boldsymbol{p}_m(t)) - M\big(\boldsymbol{p}_m(t - \Delta t)\big)$$

Image reconstruction (detailed)

Step 1: compute gradient map

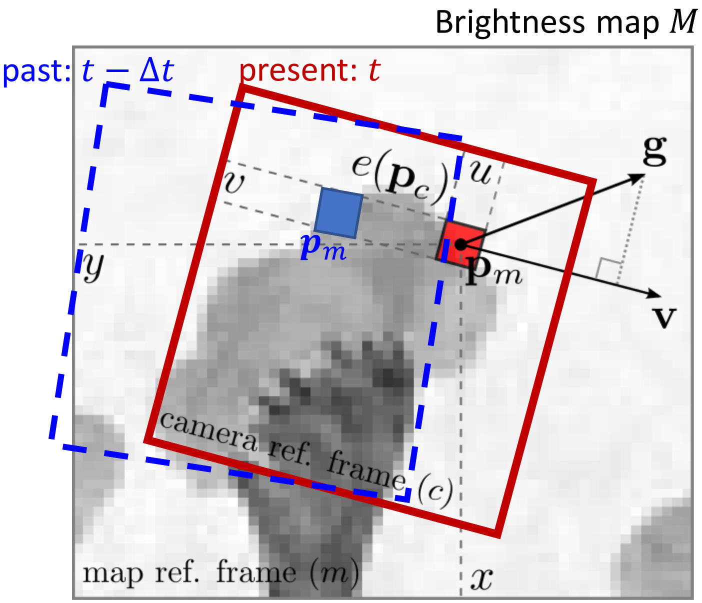
Event generation model

The brightness change in terms of the map is:

$$\Delta M = M(\boldsymbol{p}_m(t)) - M\big(\boldsymbol{p}_m(t-\Delta t)\big) = C$$

Linearization:

To first order, the brightness change can be computed using the map slope (i.e., gradient). By Taylor's approximation,



$$M(\boldsymbol{p} + \Delta\boldsymbol{p}) \approx M(\boldsymbol{p}) + \nabla M(\boldsymbol{p}) \cdot \Delta\boldsymbol{p}$$

$$M\big(\boldsymbol{p}_m(t)\big) - M\big(\boldsymbol{p}_m(t-\Delta t)\big) \approx \underbrace{\nabla M\big(\boldsymbol{p}_m(t)\big)}_{\boldsymbol{g}} \cdot \underbrace{\big(\boldsymbol{p}_m(t) - \boldsymbol{p}_m(t-\Delta t)\big)}_{\Delta\boldsymbol{p}}$$

Assuming a linear motion between the map points, $\Delta\boldsymbol{p} \approx \boldsymbol{v}\Delta t$

Hence, the event generation in terms of the map is

$$\Delta M = \boldsymbol{g} \cdot \boldsymbol{v}\Delta t \approx C$$

Image reconstruction

Step 1: compute gradient map

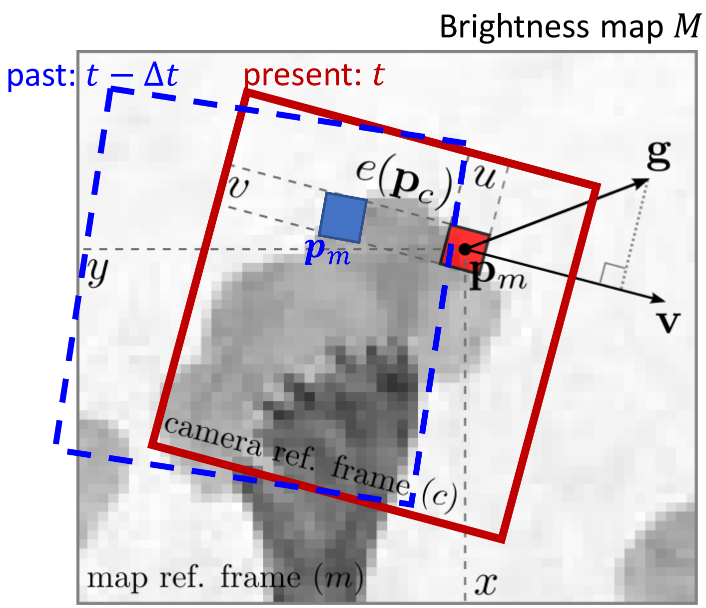
Event generation model

The event generation in terms of the map is

$$\Delta M = \boldsymbol{g} \cdot \boldsymbol{v} \Delta t \approx C$$

Similar interpretation as on the image plane: the contrast $\Delta M \propto \boldsymbol{g} \cdot \boldsymbol{v}$ is proportional to the dot product between:

- the brightness gradient $\boldsymbol{g}$
- the "motion flow" $\boldsymbol{v}$ of points on the map



Extreme cases:

- $\boldsymbol{g}$ and $\boldsymbol{v}$ are perpendicular → $\Delta M = 0$. No event is generated
- $\boldsymbol{g}$ and $\boldsymbol{v}$ are parallel → $\Delta M = C$, events are triggered the fastest (minimum Δt)

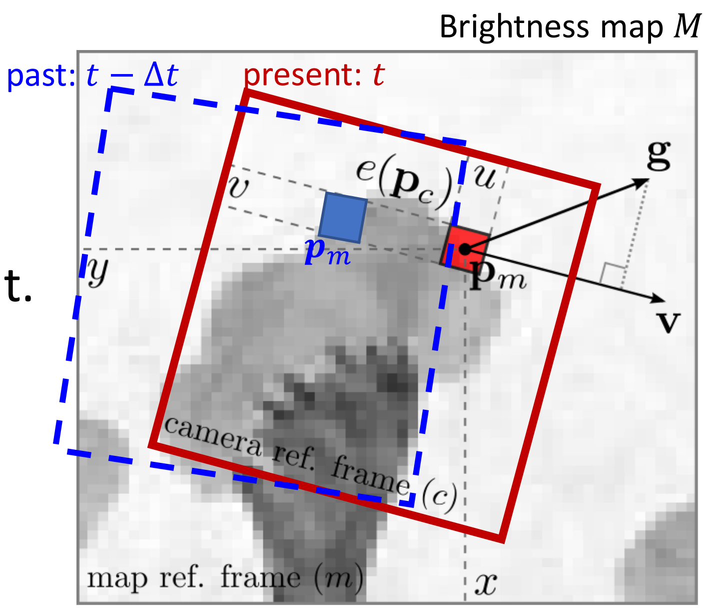
Image reconstruction

Step 1: compute gradient map

Extended Kalman Filter

The event generation in terms of the map is

$$\boldsymbol{g} \cdot \boldsymbol{v} \Delta t \approx C$$



In this equation, only $\boldsymbol{g}$ is unknown, and it is constant.
Let us use an EKF to estimate this constant vector.
(one EKF per map point)

- State eq: $\boldsymbol{g}_k = \boldsymbol{g}_{k-1}$
- Observation equation: $h_k = \frac{\boldsymbol{g}_k \cdot \boldsymbol{v}_k}{C}$

Thus, we use the "event rate" at the map point $\boldsymbol{p}_m$ as observation: $z_k = \frac{1}{\Delta t}$

Image reconstruction

Step 1: compute gradient map

Extended Kalman Filter

Iteration equations:

- Innovation: $\nu_k = z_k - h_k$
- Innovation covariance: $S_k = H_k P_{k-1} H_k^\top + R_k$
    - Measurement matrix is the Jacobian:
      $$H_k = \frac{\partial h_k}{\partial \boldsymbol{g}_k} = \frac{\boldsymbol{v}_k}{C}$$
    - R_k is the covariance of the measurement z_k
- Kalman gain: $K_k = P_{k-1} H_k^\top S_k^{-1}$

- <u>Gradient update</u>: $\boldsymbol{g}_k = \boldsymbol{g}_{k-1} + K_k \nu_k$
- <u>Covariance of gradient update</u>: $P_k = P_{k-1} - K_k S_k K_k^\top$

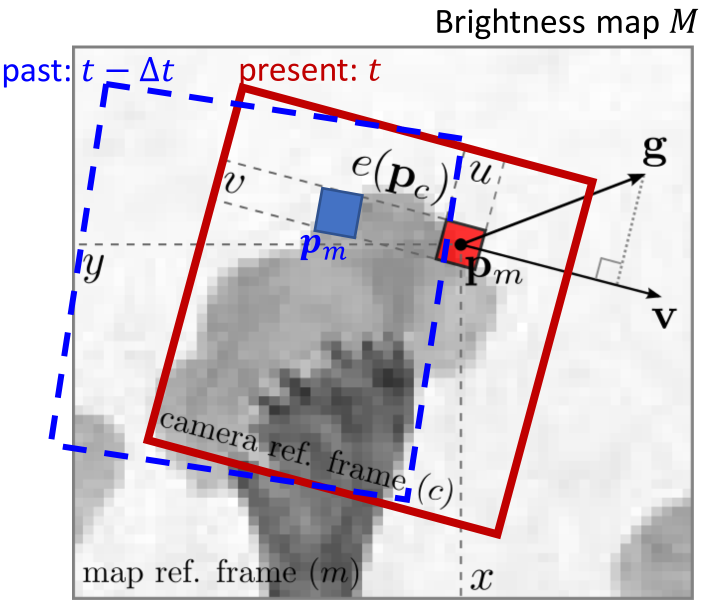
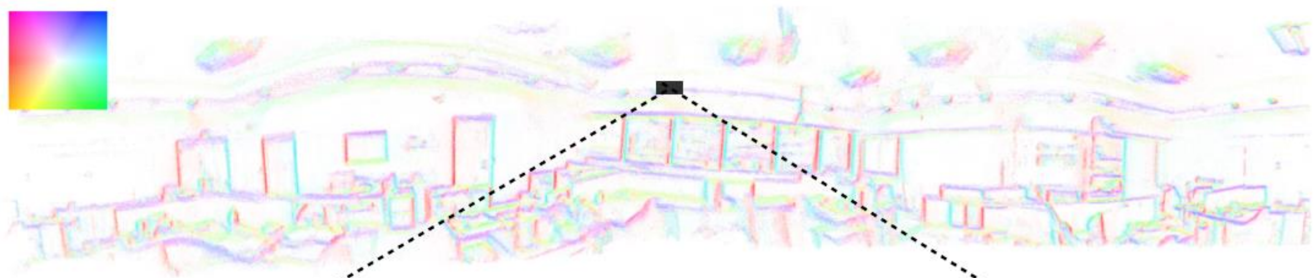


Image reconstruction

Step 1: compute gradient map

Extended Kalman Filter (one per map pixel)

Gradient map:
magnitude (radius)
and direction (color)



Arrow: gradient vector $\boldsymbol{g}$
Ellipse: covariance P

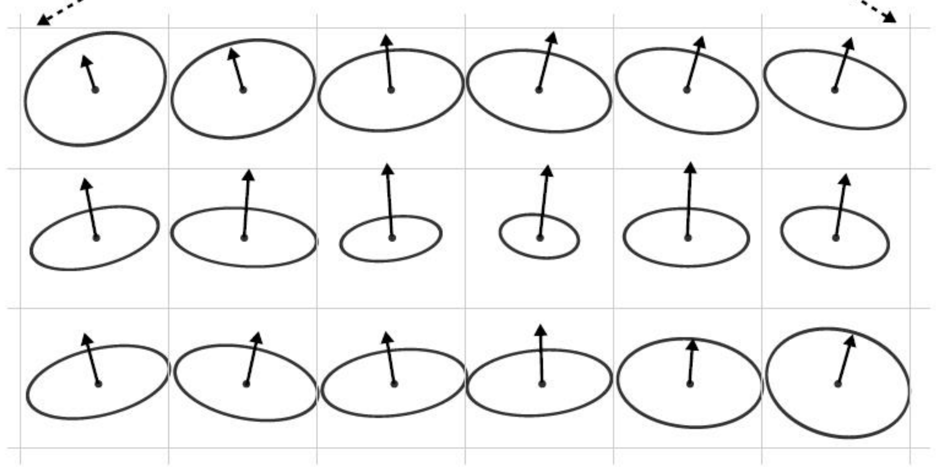


Image by Kim et al. BMVC 2014

Image reconstruction

Step 1: compute gradient map

Extended Kalman Filter (one per map pixel)

Iterations:

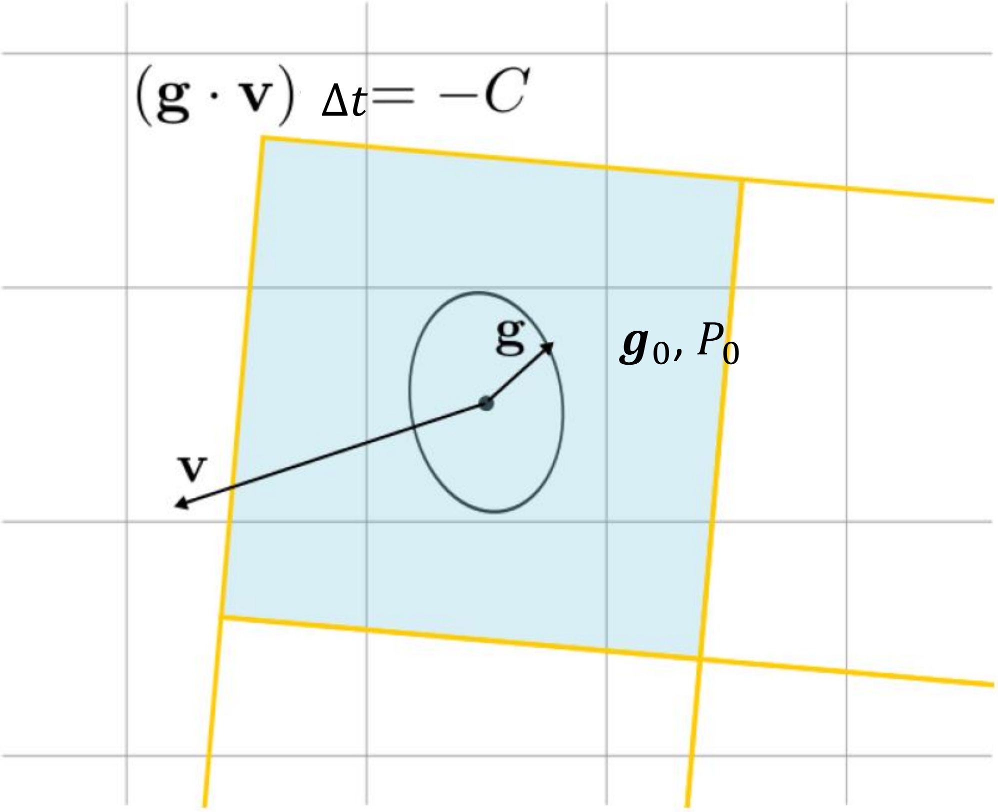


Image by Kim et al. BMVC 2014

Image reconstruction

Step 1: compute gradient map

Extended Kalman Filter (one per map pixel)

Iterations:

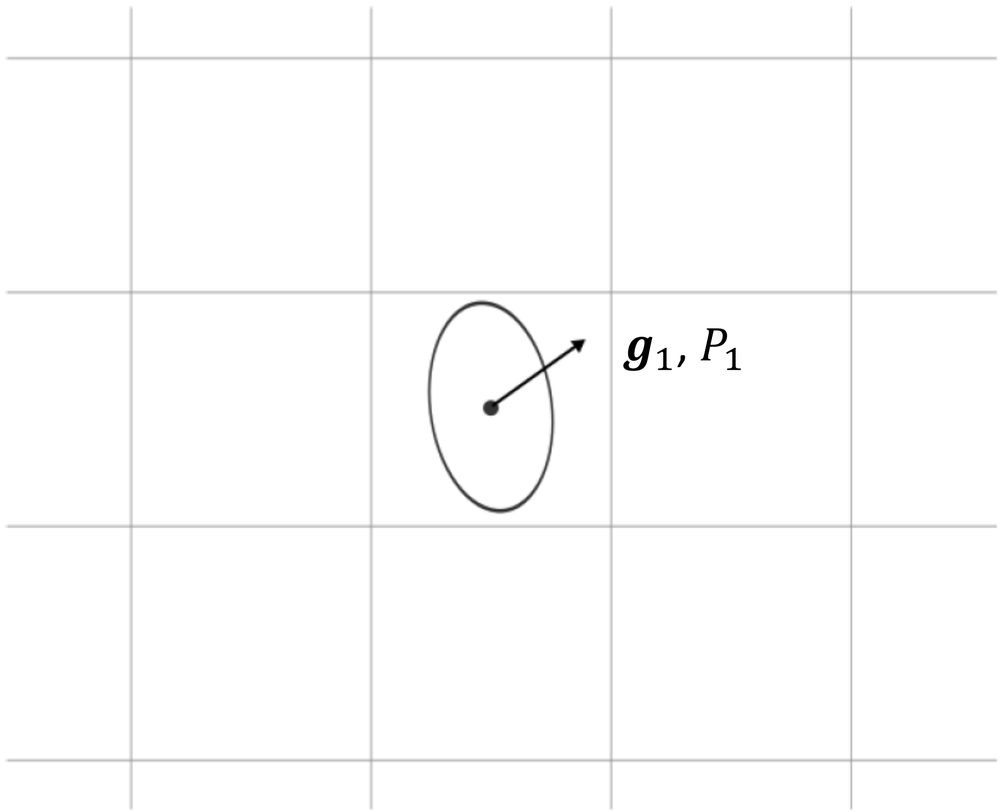


Image by Kim et al. BMVC 2014

Image reconstruction

Step 1: compute gradient map

Extended Kalman Filter (one per map pixel)

Iterations:

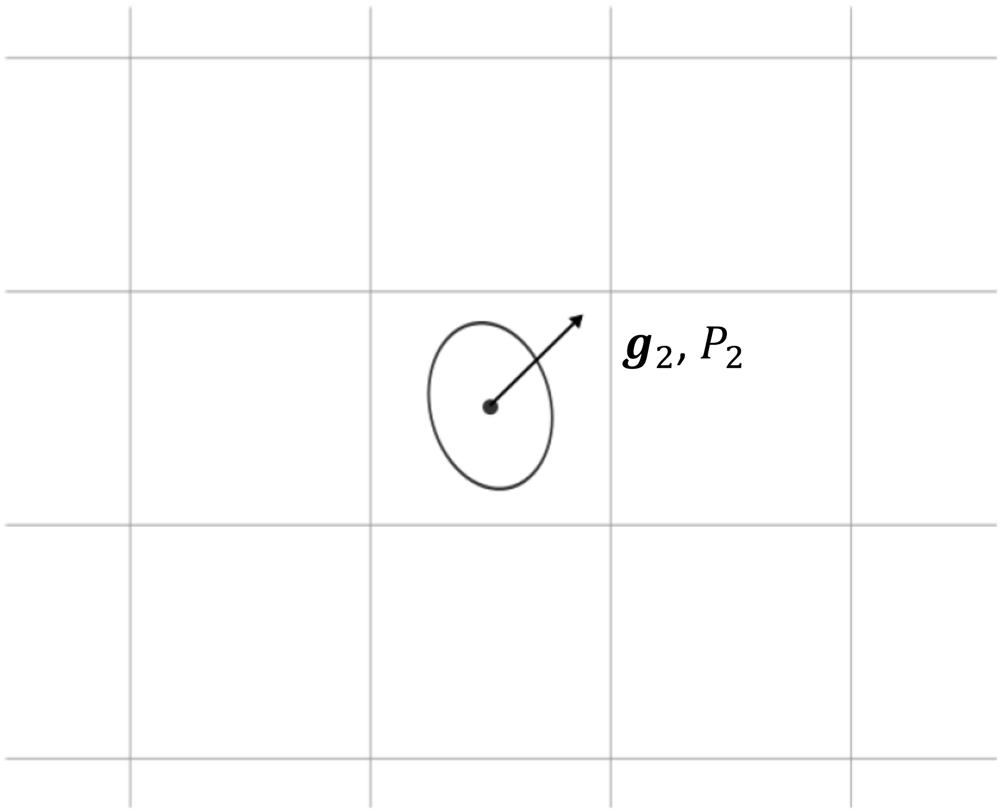


Image by Kim et al. BMVC 2014

Image reconstruction

Step 1: compute gradient map

Extended Kalman Filter (one per map pixel)

Iterations:

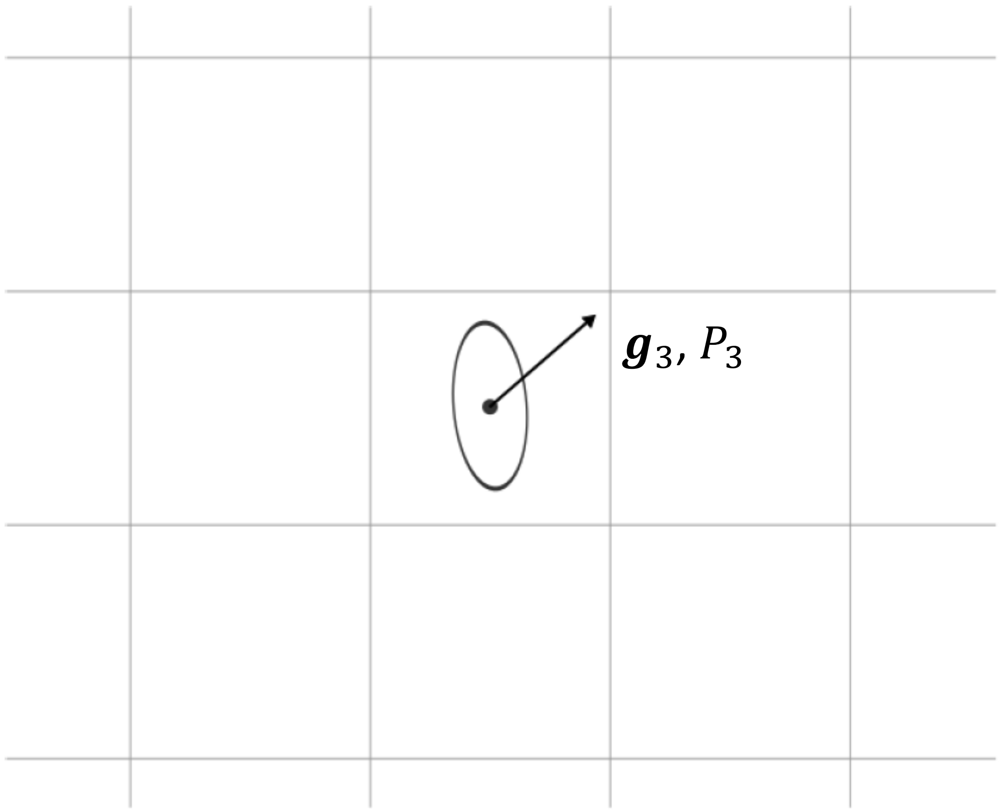


Image by Kim et al. BMVC 2014

Image reconstruction

Step 1: compute gradient map

Extended Kalman Filter (one per map pixel)

Iterations:

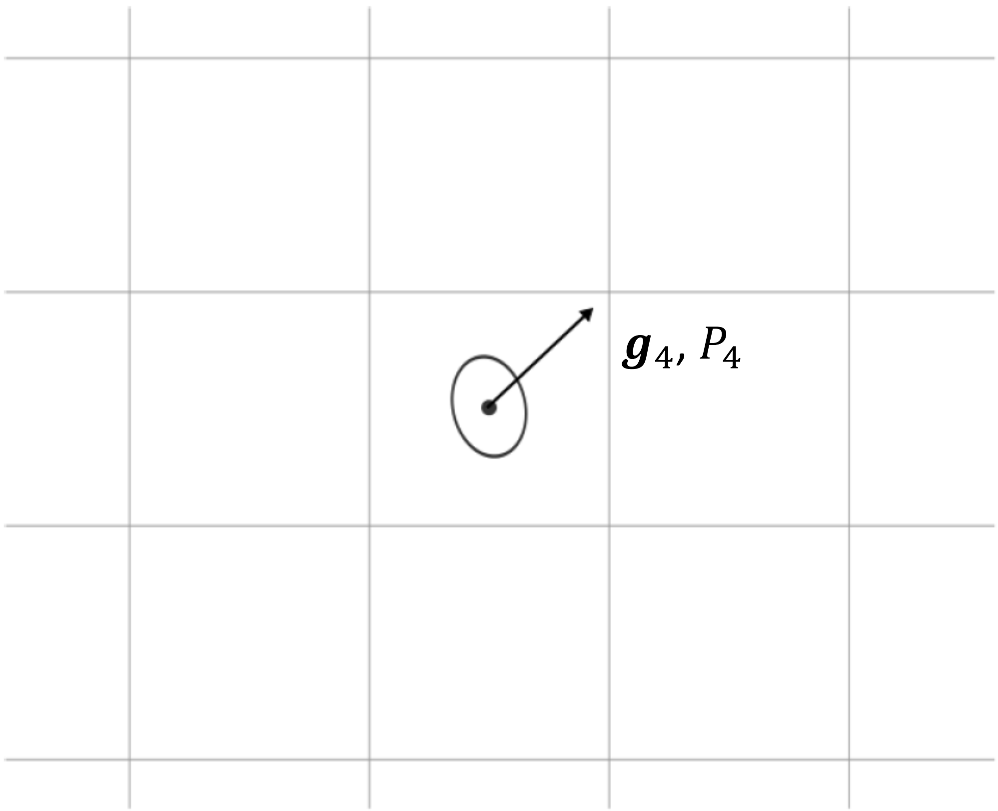


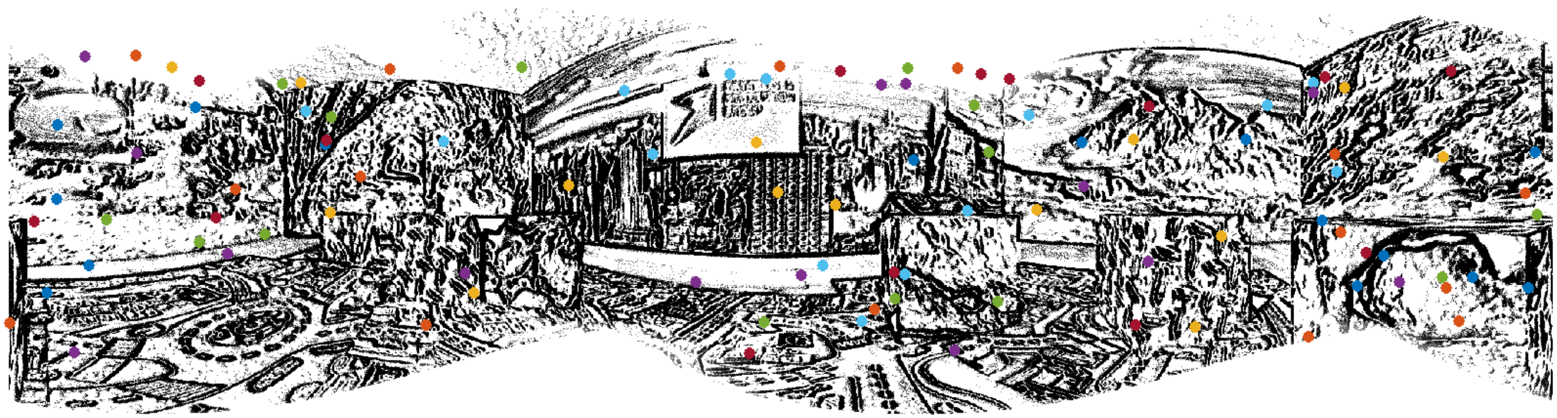
Image by Kim et al. BMVC 2014

Image reconstruction

Step 1: compute gradient map

Extended Kalman Filter

Convergence: evolution of the trace of the covariance of some points



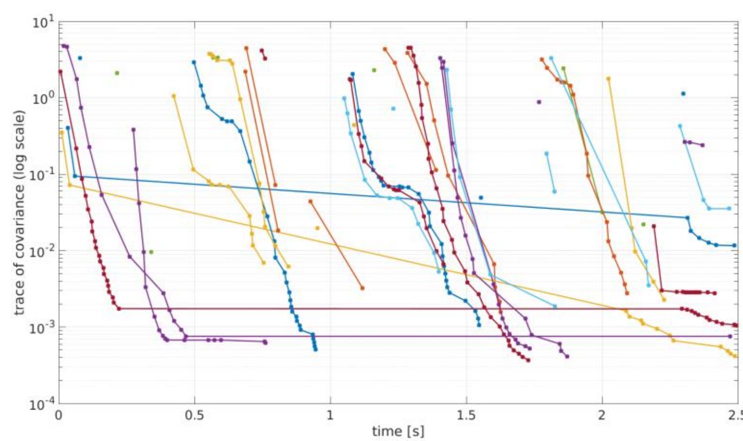
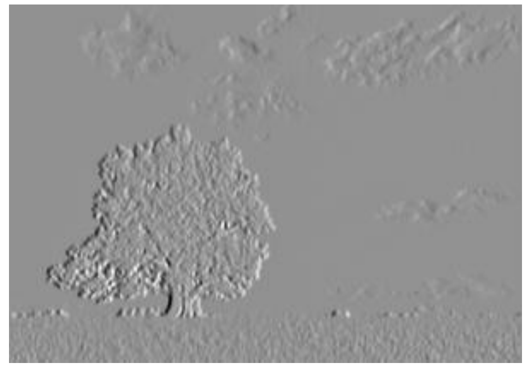


Image reconstruction

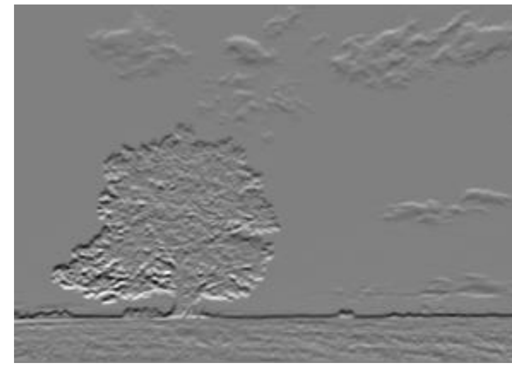
Step 2: Poisson integration

Integrate gradient map $\boldsymbol{g}$ to get absolute brightness M

Original Image



Gradient in *x* direction
$(g_{\mathrm{x}} = \partial_{\mathrm{x}} I)$

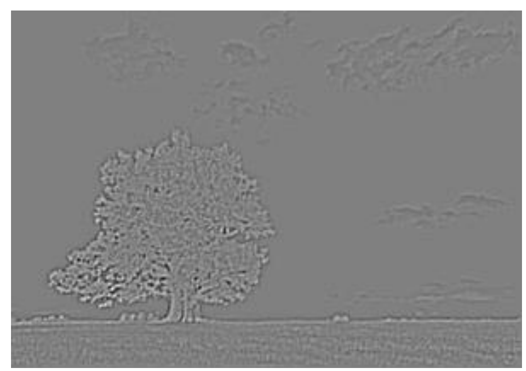


Gradient in *y* direction
$(g_{\mathrm{y}} = \partial_{\mathrm{y}} I)$

Reconstructed Image

2D Integration

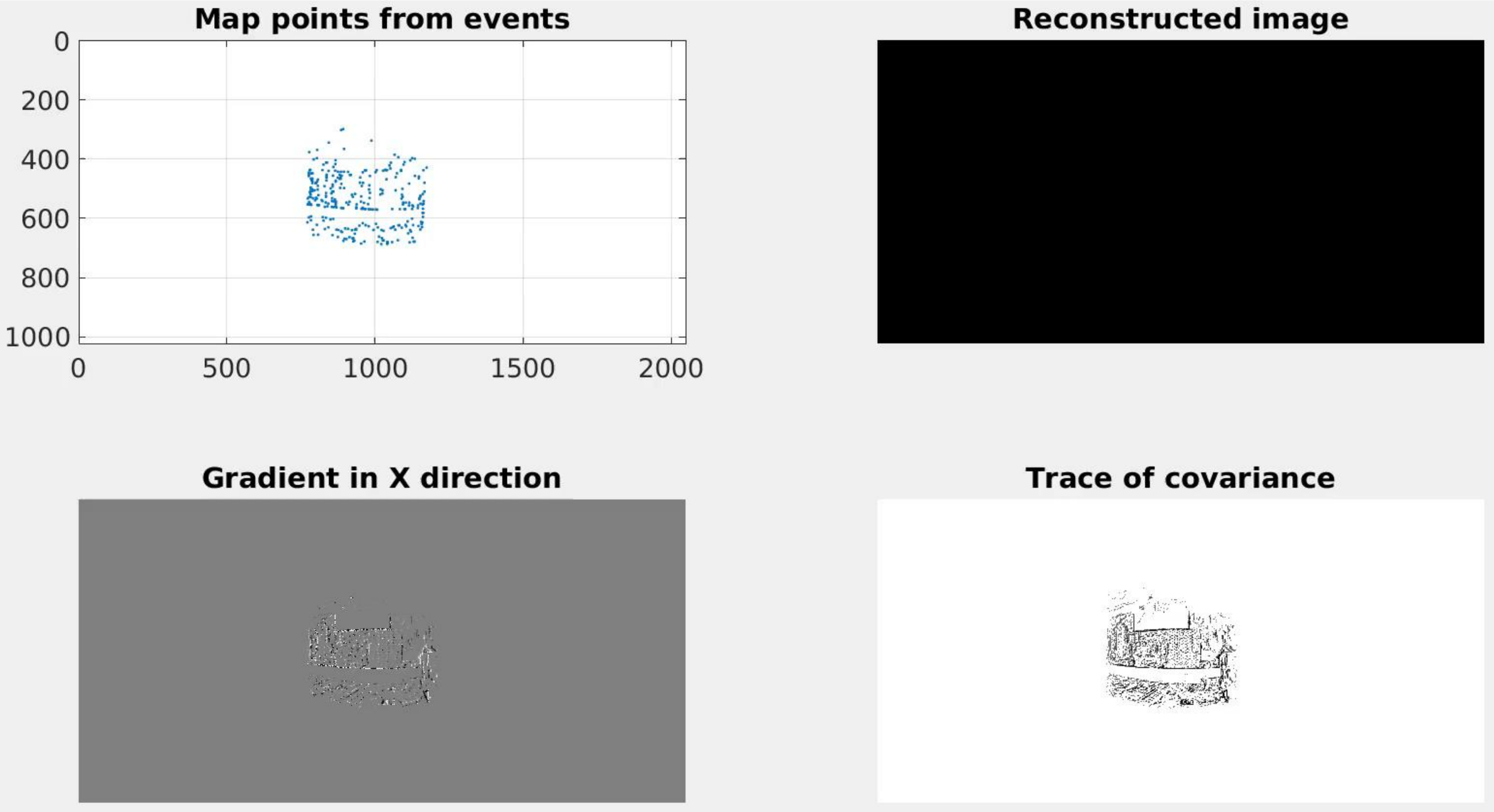
Solve Poisson eq:
$(\Delta \tilde{I} = \operatorname{div} \boldsymbol{g})$
fast using the FFT



Divergence
$(\operatorname{div} \boldsymbol{g} = \partial_{\mathrm{x}} g_{\mathrm{x}} + \partial_{\mathrm{y}} g_{\mathrm{y}})$

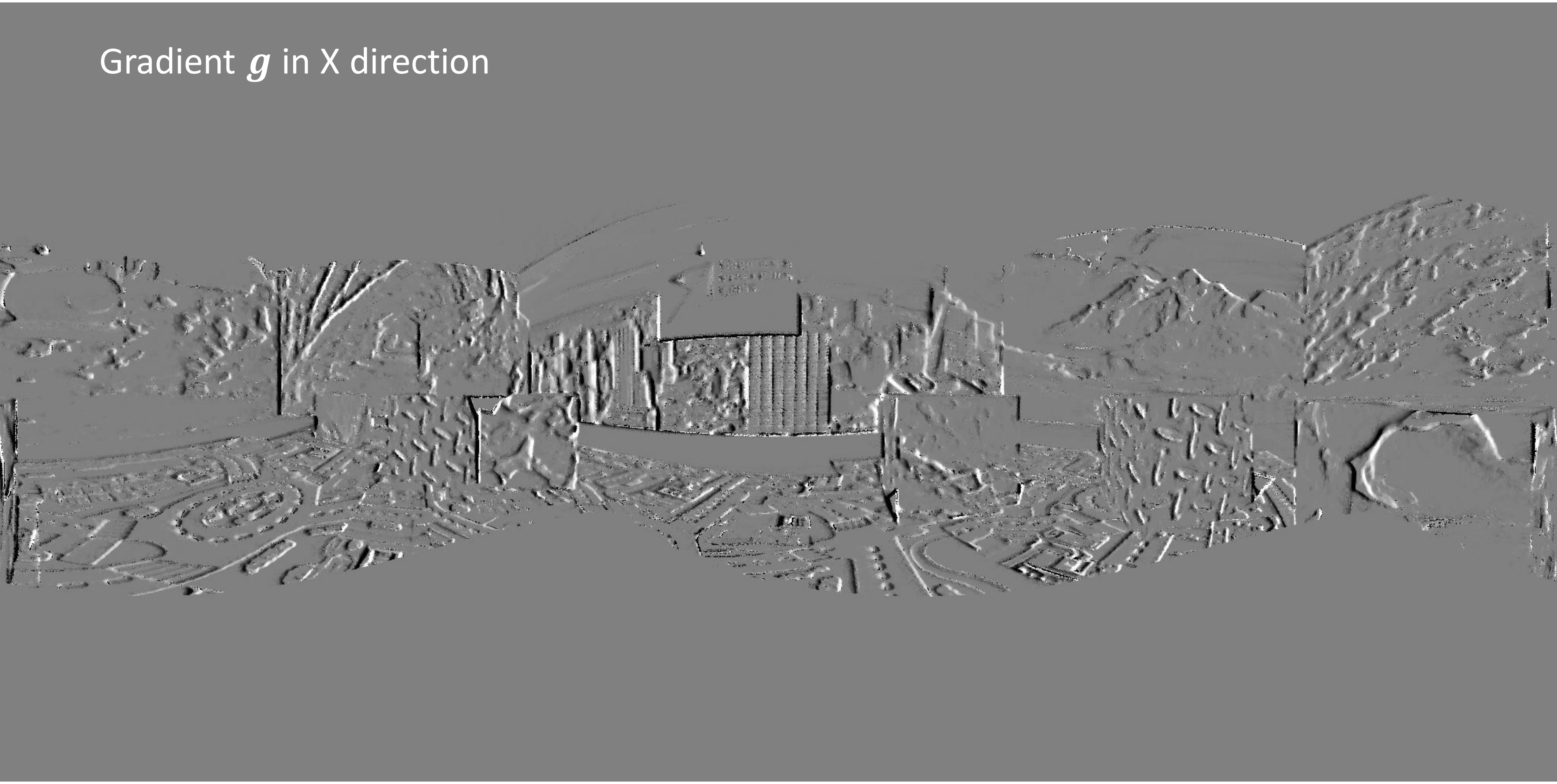
Slide by T. Horstschäfer

Image reconstruction demo



https://github.com/uzh-rpg/rpg_image_reconstruction_from_events

Image reconstruction demo



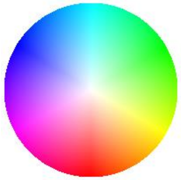
https://github.com/uzh-rpg/rpg_image_reconstruction_from_events

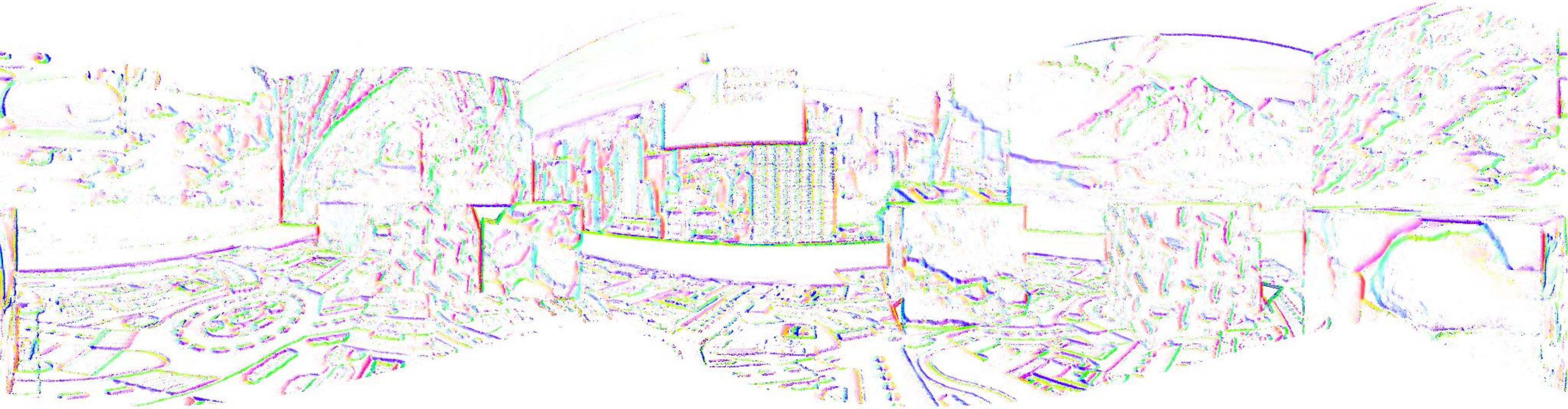
Image reconstruction demo

Gradient g in Y direction

https://github.com/uzh-rpg/rpg_image_reconstruction_from_events

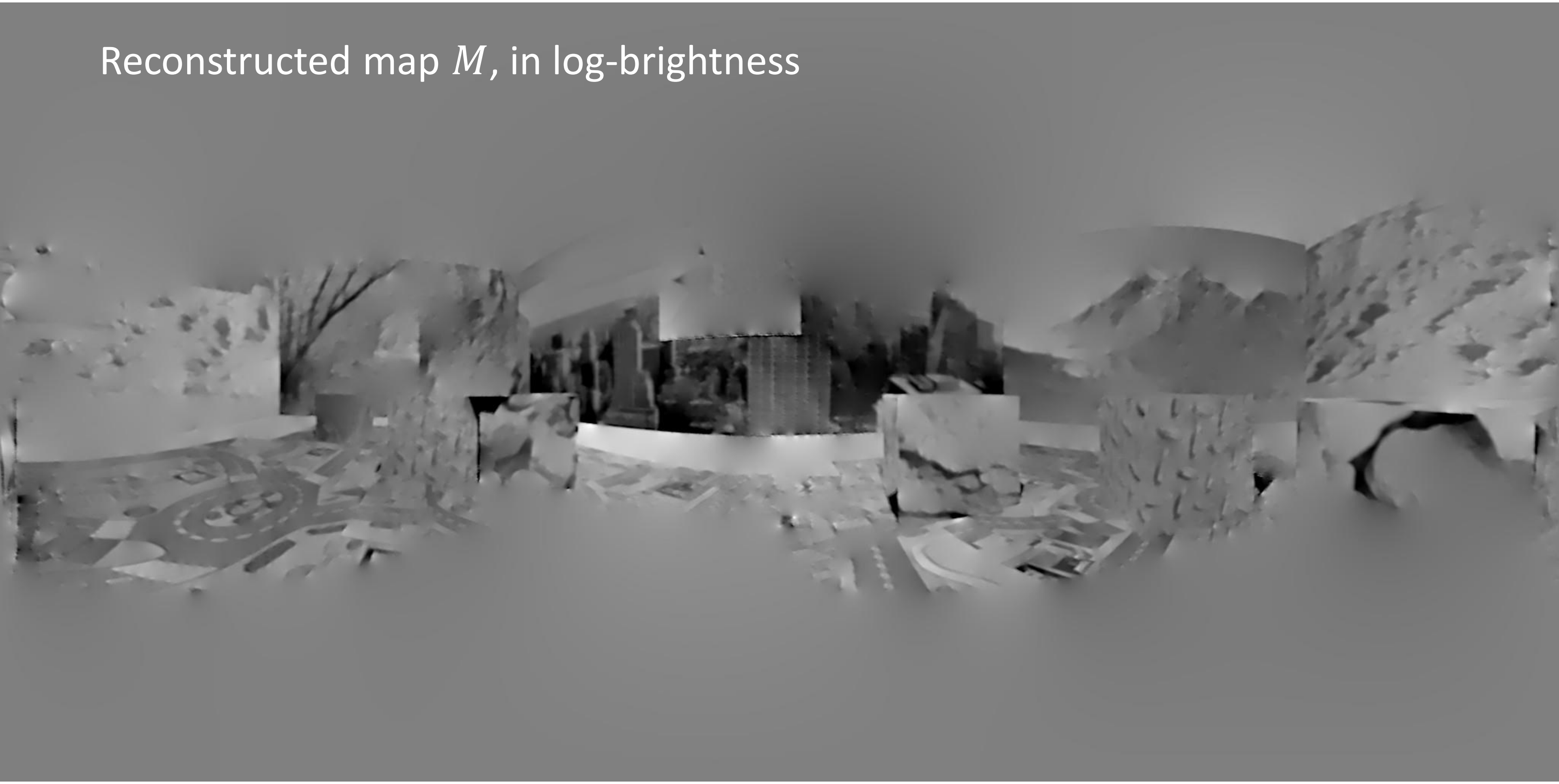
Image reconstruction demo

Gradient $\boldsymbol{g}$: magnitude (radius) and orientation (color) 



https://github.com/uzh-rpg/rpg_image_reconstruction_from_events

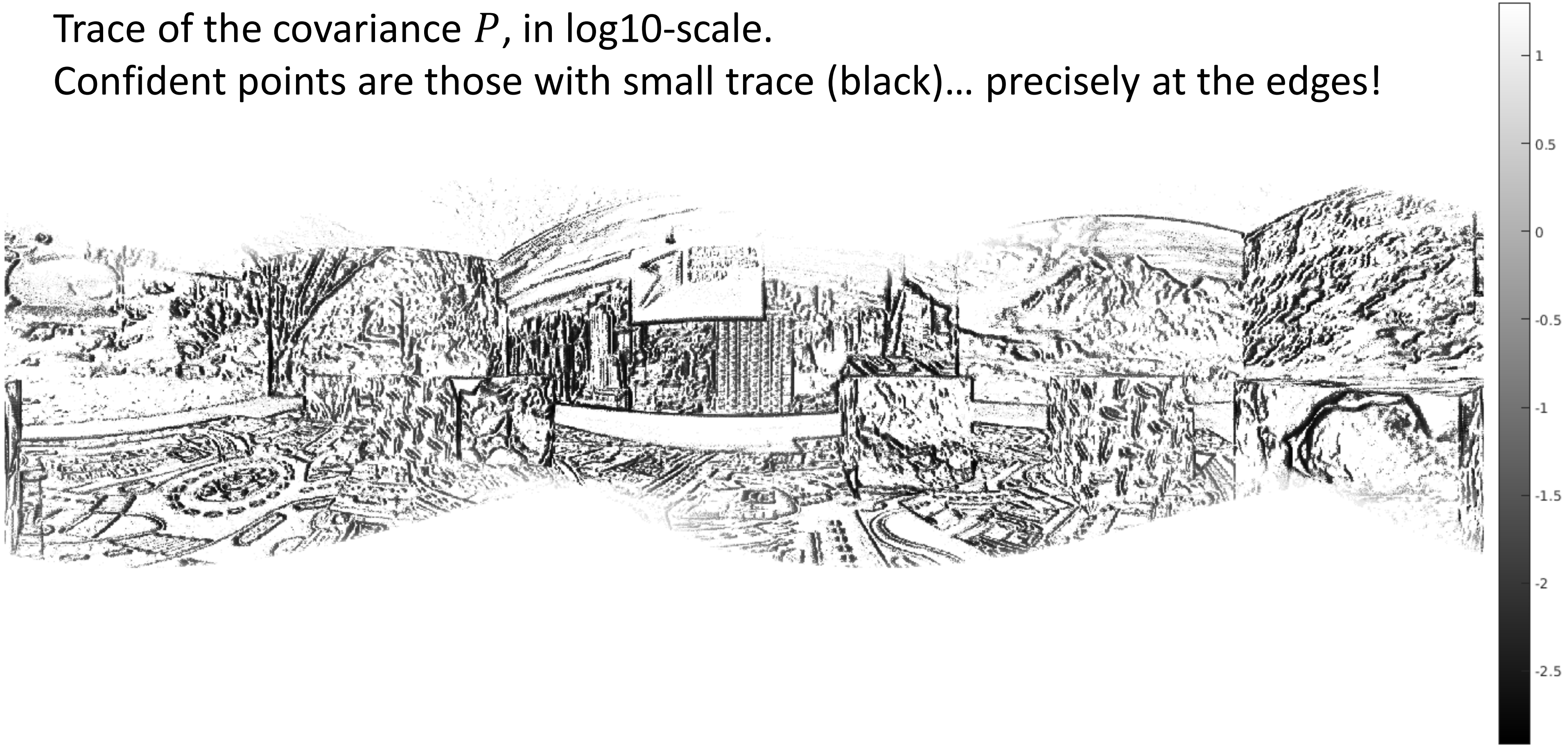
Image reconstruction demo



https://github.com/uzh-rpg/rpg_image_reconstruction_from_events

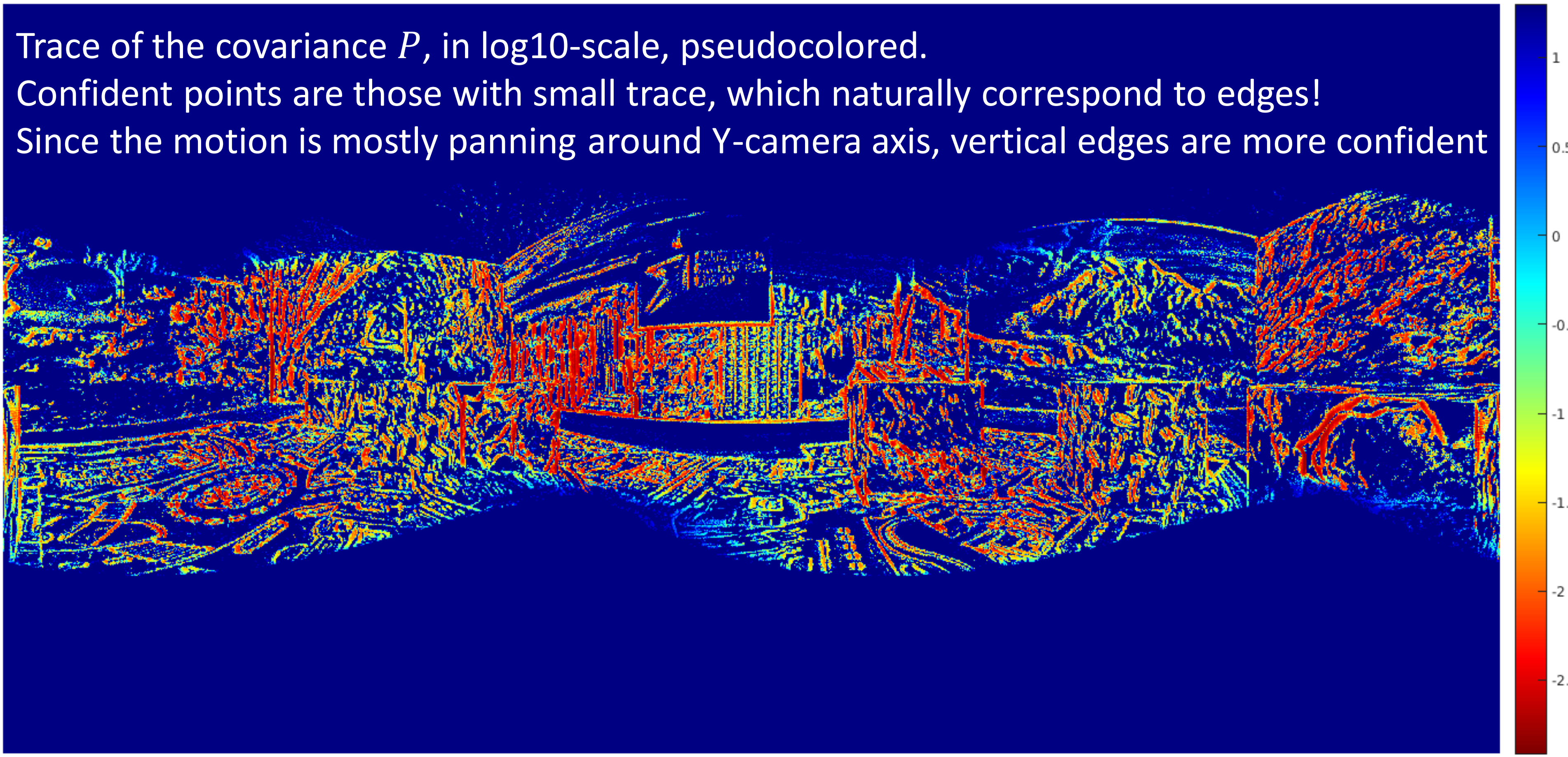
Image reconstruction demo

Trace of the covariance P, in log10-scale.
Confident points are those with small trace (black)… precisely at the edges!



https://github.com/uzh-rpg/rpg_image_reconstruction_from_events

Image reconstruction demo



https://github.com/uzh-rpg/rpg_image_reconstruction_from_events