Some techniques for agile **visual** tracking and SLAM

Georg Klein
Active Vision Lab, Oxford

Workshop on Industrial Augmented Reality:
Needs & Solutions

ISMAR 2008
Cambridge, UK

Research ≠ Real world

Lab environment

- Small
- Static
- Controlled lighting
- Textured surfaces
- Controlled clutter
- **Expert user**

Real world

- Large
- Ever-changing
- Variable lighting
- Blank walls
- Dynamic clutter
- **Untrained user**

Researcher vs. untrained user

- SLAM systems and trackers often need constrained camera motions
- The researchers who wrote the software understand this
- Other users often break the system

We would like a system which any untrained user can pick up and use naturally

Two approaches:

1. Improve frame-to-frame robustness
   - Correctly track unconstrained motions
   - Reduce frequency of tracker failure
2. Fast recovery from tracking failure
   - All trackers fail eventually
   - Make this as painless as possible

Frame-to-frame visual tracking

Current approaches:

1. Bottom-up detection each frame

   (e.g. ARToolkit, EPFL)

2. Direct minimisation

   (Usually planar target)

3. Particle Filtering

4. **Active Search**

Active Search

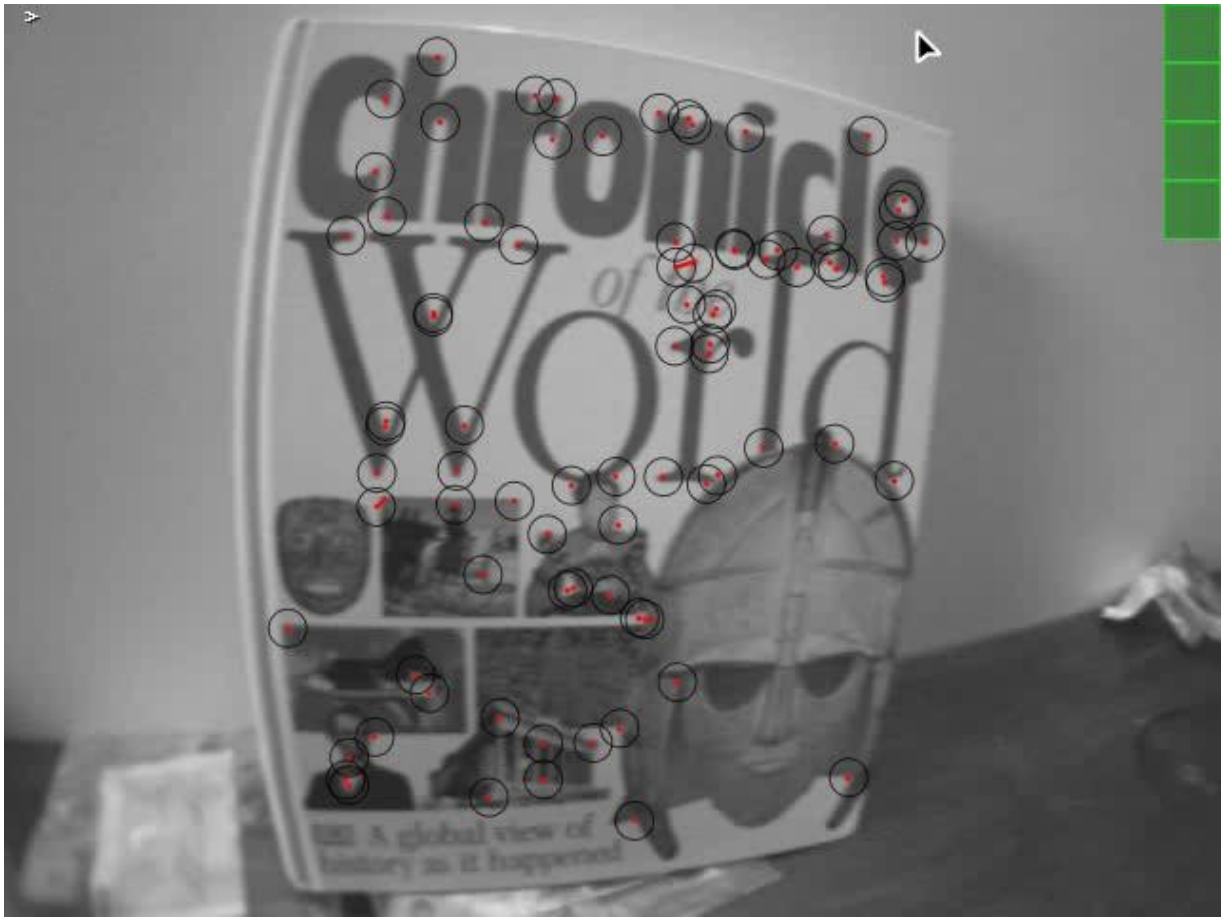
- Based on image features such as points or lines
- Start with a prior pose from previous frame + motion model
- Unknown acceleration produces a search region in the image
- E.g.: Andrew Davison's EKF-SLAM

Active search regions in EKF-SLAM

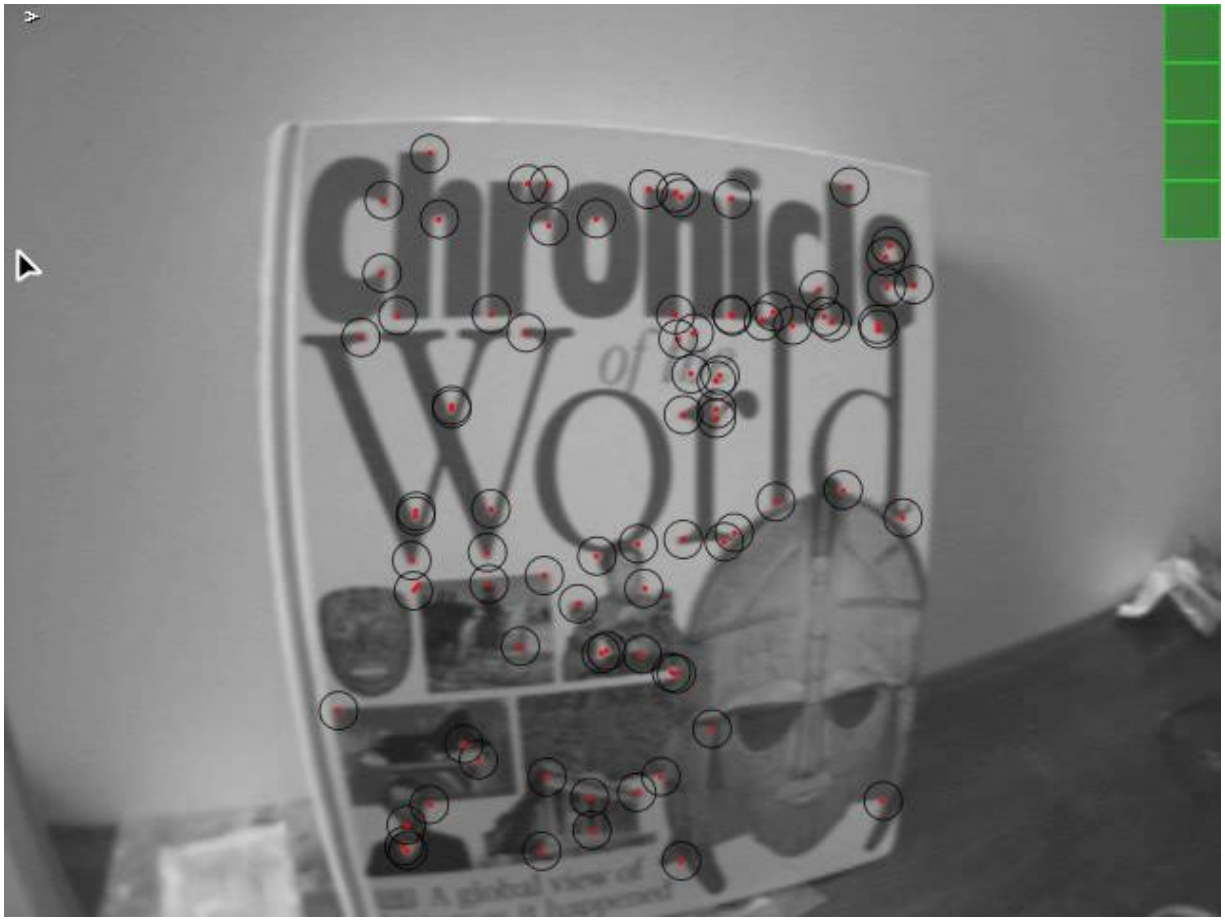
Active Search Advantages

- Fast:
  - Only look at interesting parts of image
  - Naturally culls outliers
- Probabilistically sound
  - Can include both acceleration and map uncertainty

Active Search Example: 10-pixel search region

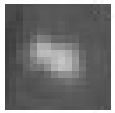


Active Search Example: 40-pixel search region

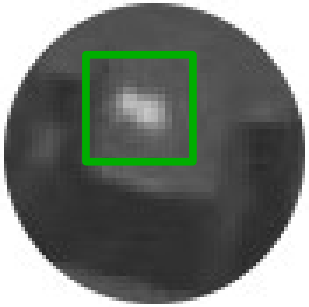


Search range compromise

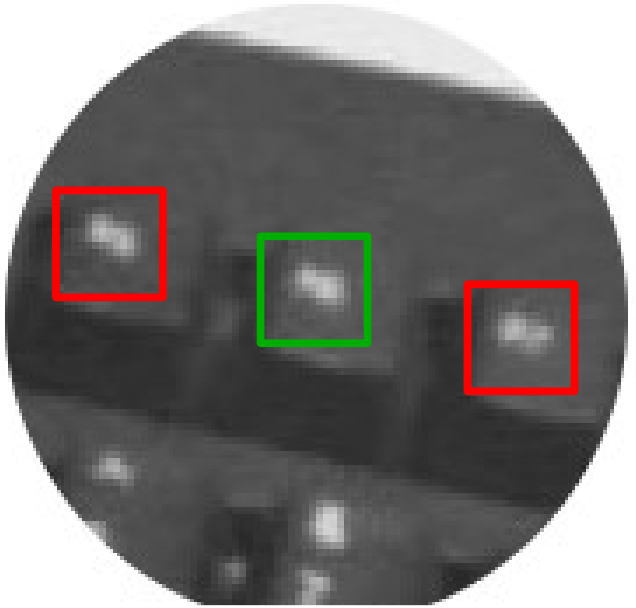
- Rapid accelerations require large search regions
  - Introduce outliers
  - Outliers produce jitter and tracking failure
  - Slow to process



Search template



Small region: One match



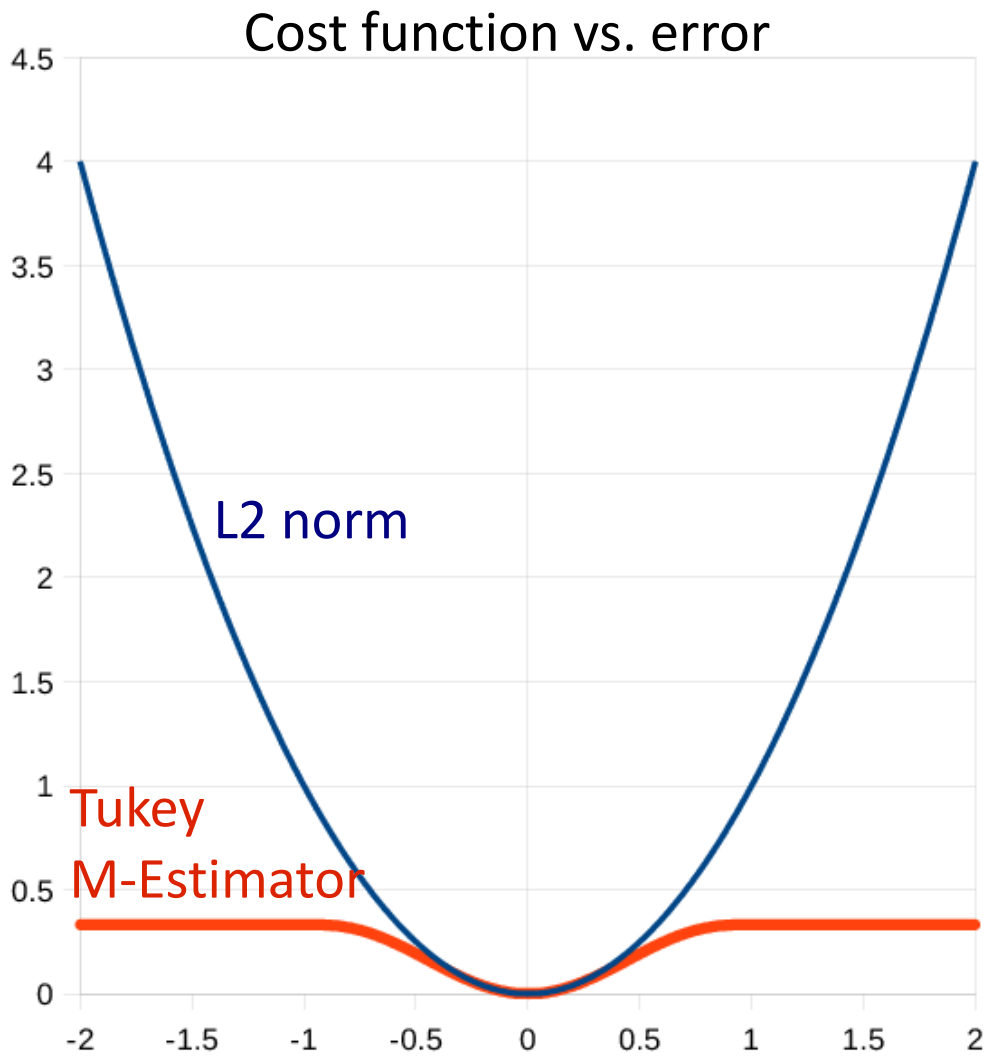
Large region: Three matches

Robust Estimation Techniques

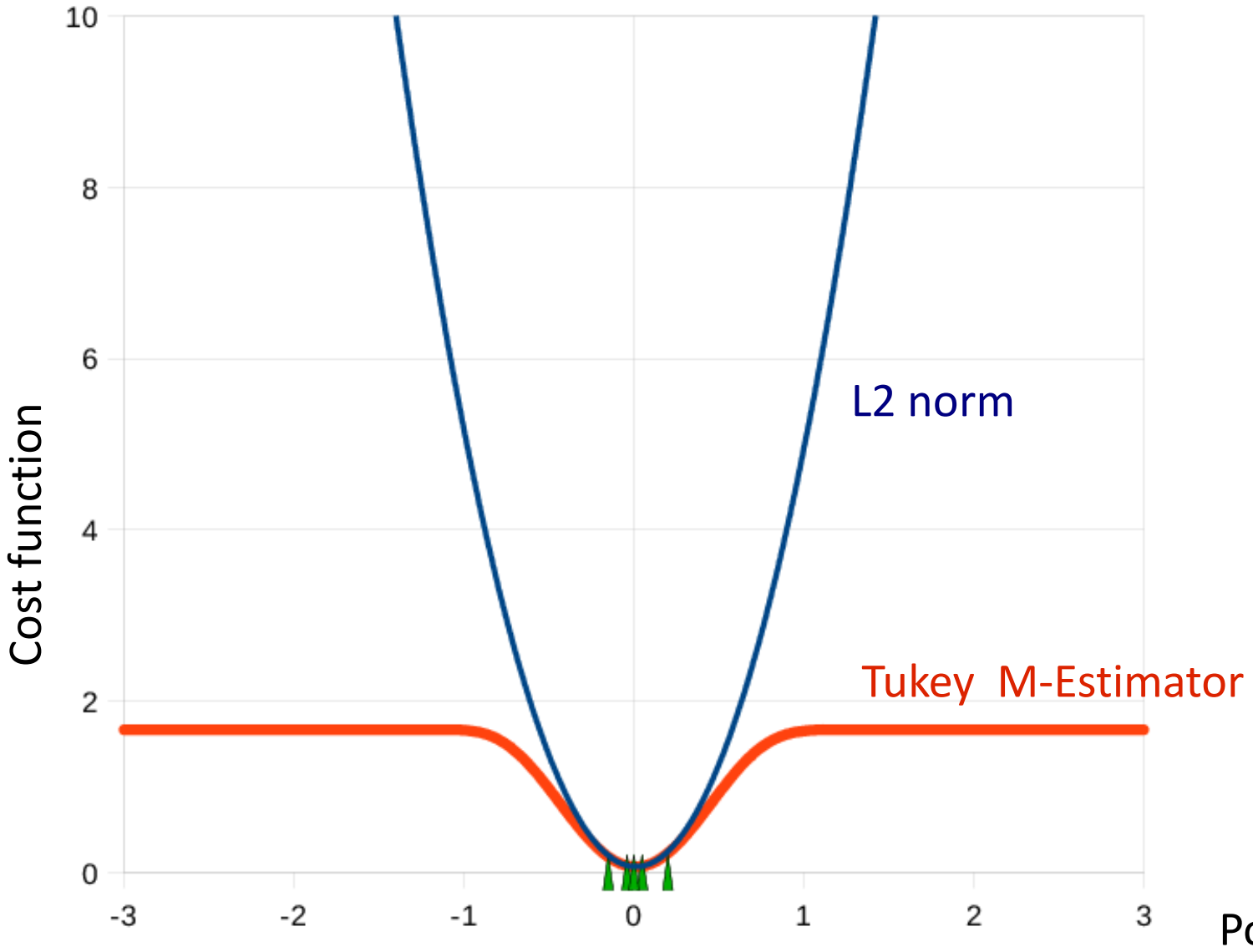
- Random Sample Consensus (RANSAC / MLESAC)
- Joint Compatibility Branch & Bound (JCBB)
- **M-Estimators**

M-Estimators

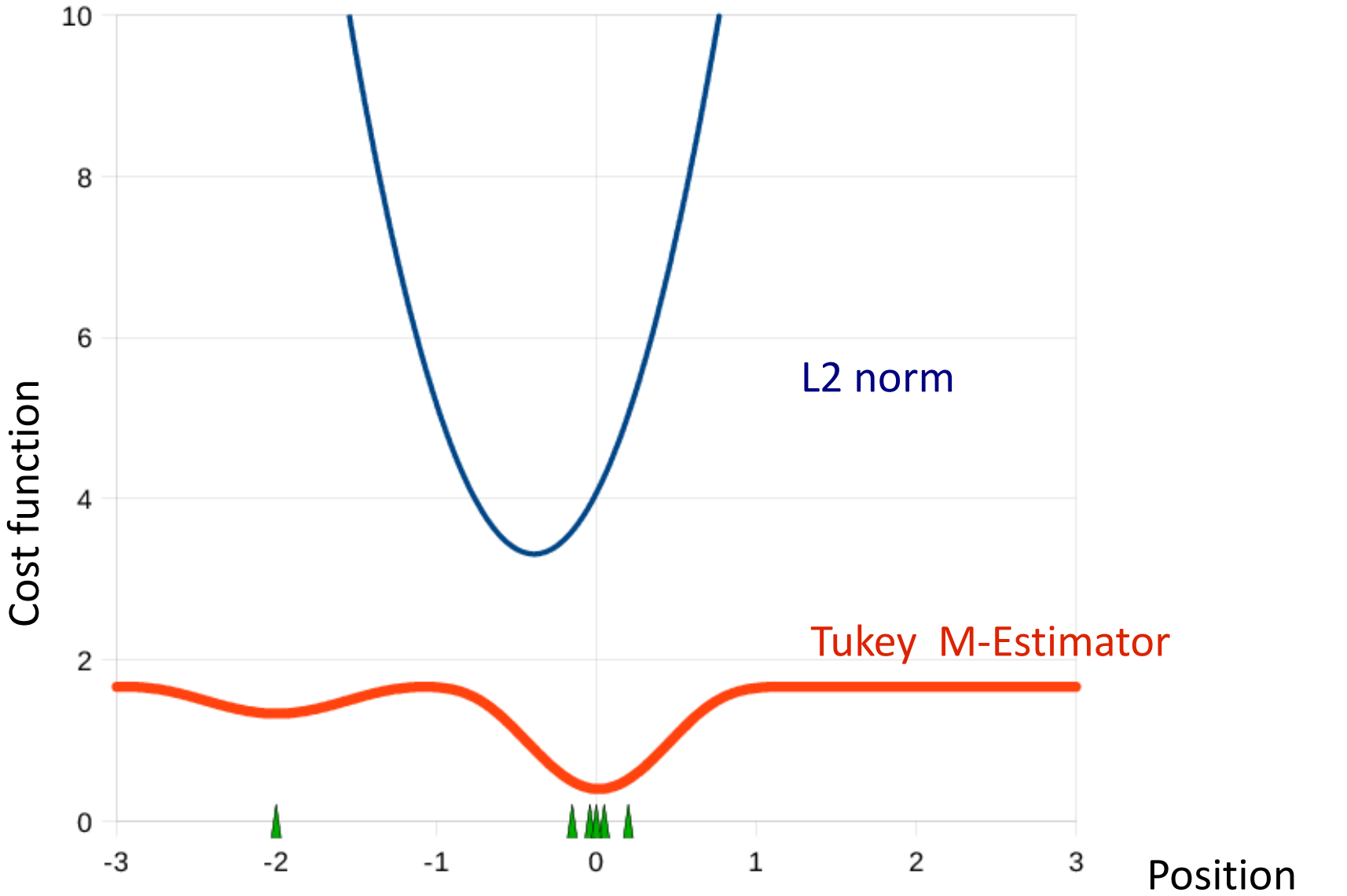
- Modify cost function from L2 norm
- Reduce influence of outliers
- Fit neatly into standard least-squares minimisation methods



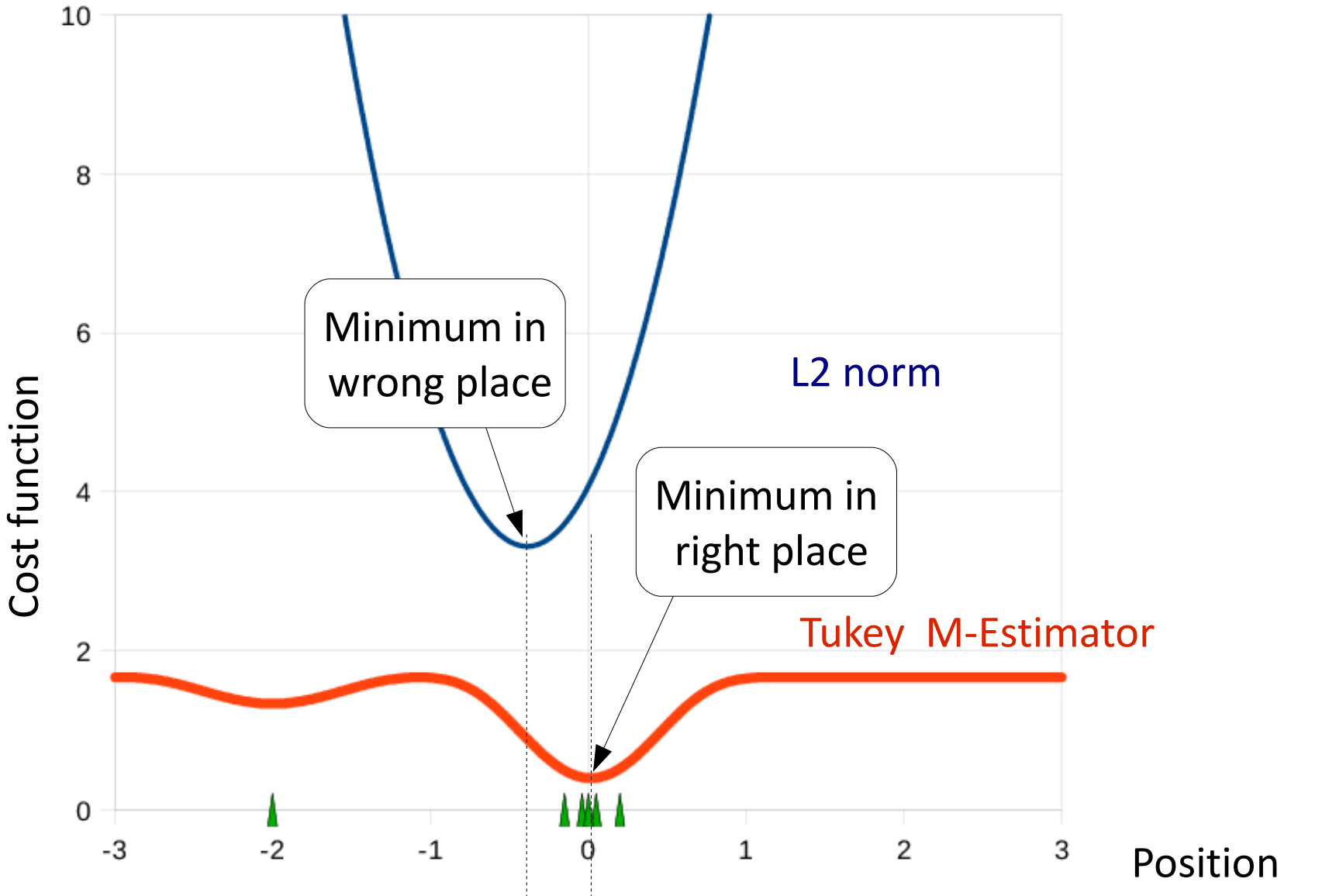
M-Estimators: 5 measurements, no outliers



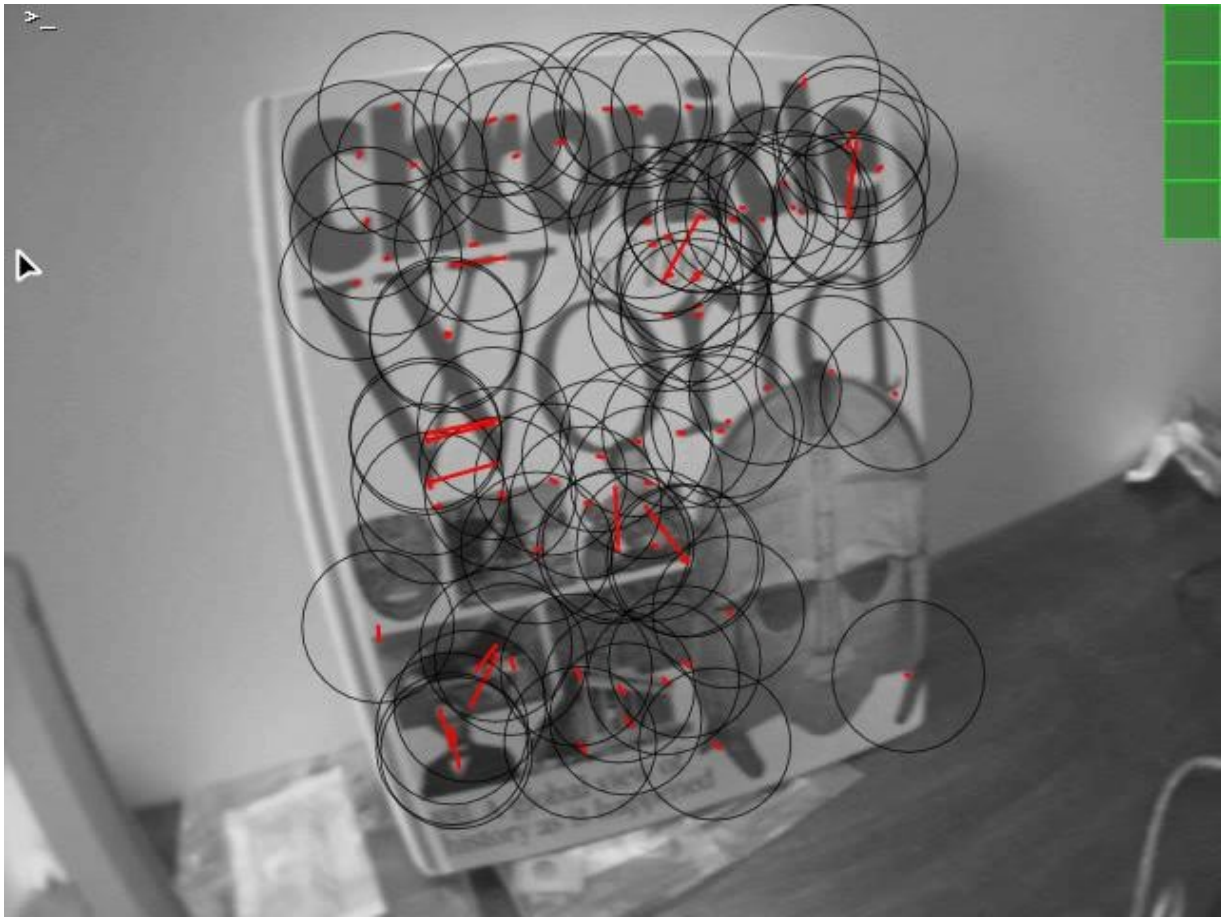
M-Estimators: 5 measurements, 1 outlier



M-Estimators: 5 measurements, 1 outlier



Large search range, M-Estimator



Large search range, M-Estimator

- M-Estimators work best with many measurements
- .. but large search range is expensive!
- Also motion blur destroys small patches

Large image patches

- Larger patches more resilient to repetition, blur
- ... but are slow to compute!
- Solution: Use an image pyramid

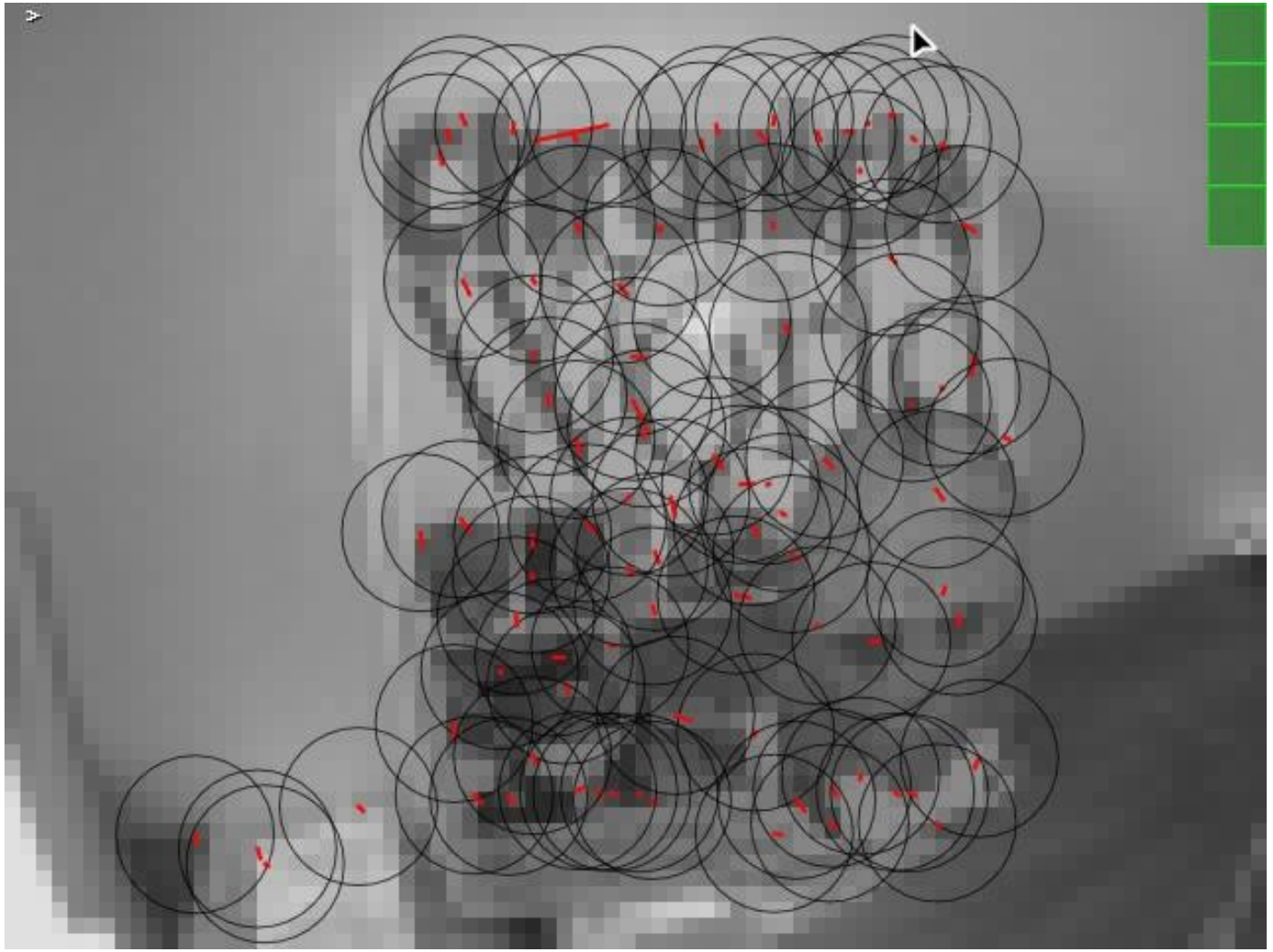
Four levels from
640x480 to 80x60

8x8 pixel patches at different levels

Coarse-to-fine tracking

- Find largest features with a large search radius
- Compute rough pose estimate from these
- Then search for small features with much tighter radius

Coarse-to-fine tracking



Coarse-to-fine tracking

- Fast to process and *reasonably* robust
- Implemented in a SLAM system for ISMAR 2007
- Source code available:

  `http://www.robots.ox.ac.uk/~gk/PTAM`

Image degradation: no motion

Image degradation: moderate motion

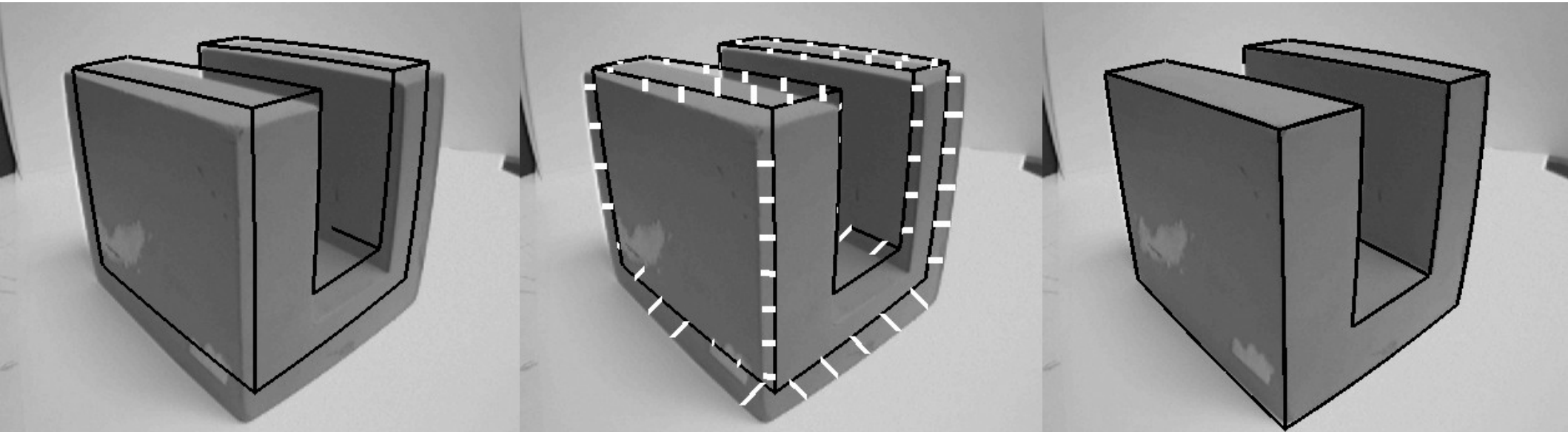
Image degradation: rapid motion

Tracking with Motion Blur

- Need features which are trackable despite blur
- Large image patches somewhat OK..
- Also use **Edges**

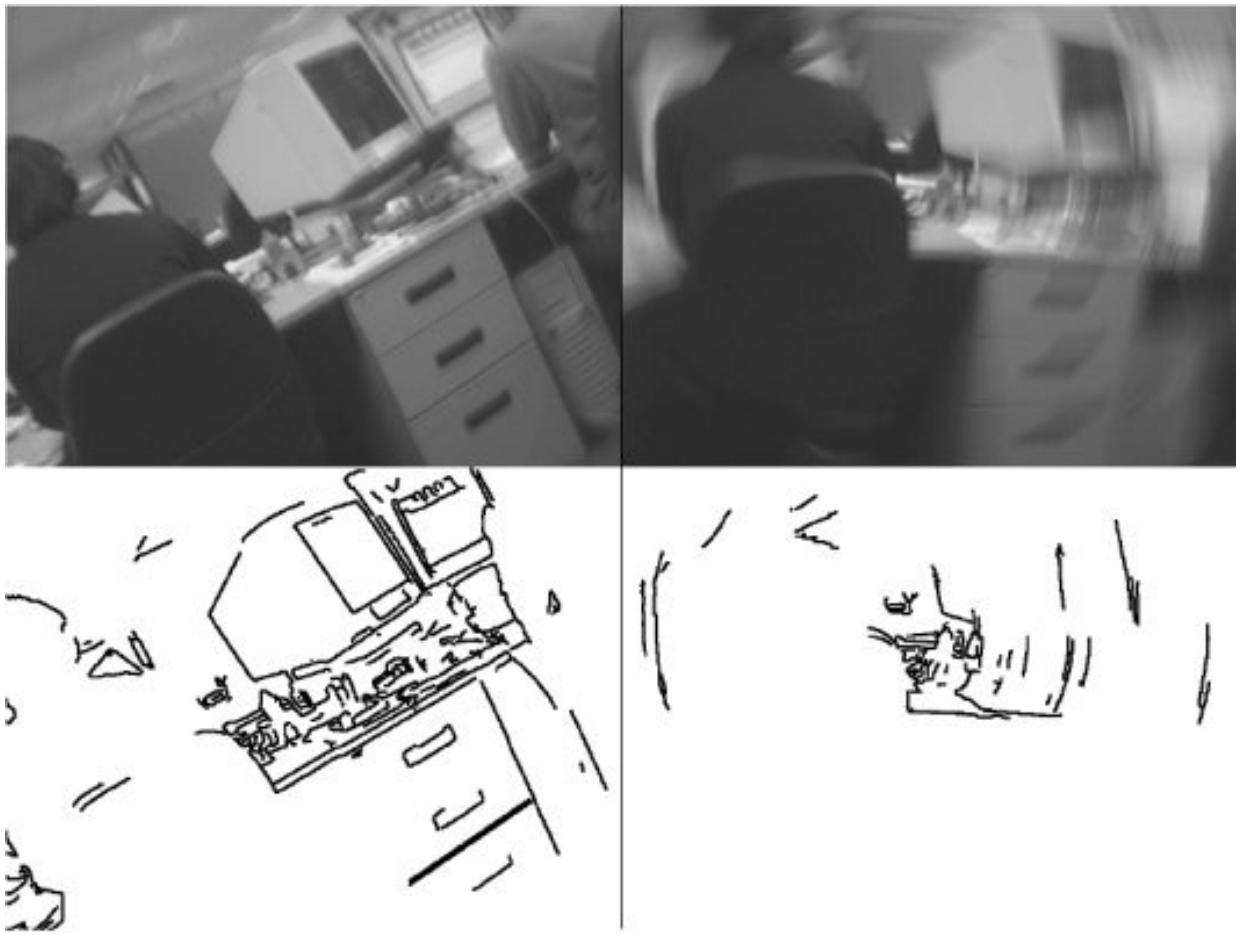
Edge Tracking

- Well-studied problem
- Very fast: real-time since 1989! (Harris' RAPID)
- Active search is **1-dimensional** along edge normal
- Downside: poor data association



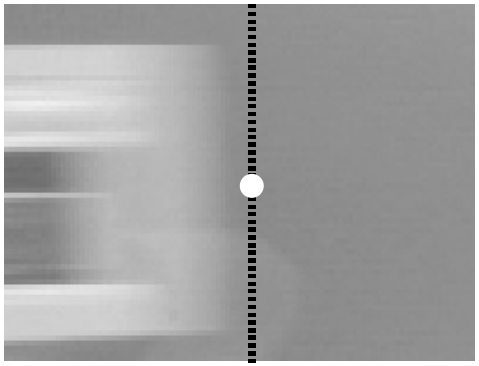
Edge Tracking with Blur

- Typically some edges will still be un-blurred!

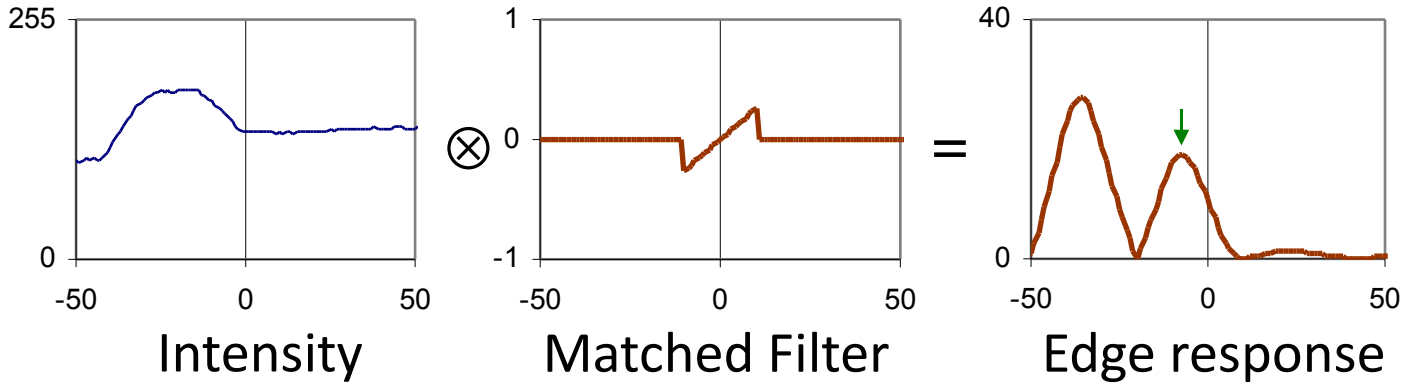


Edge Tracking with Blur

- Typically some edges will still be un-blurred!
- We can even track the blurred ones:
  - Predict amount of blur
  - Use matched filter to detect blurred edge



Blurred edge



Edges in SLAM

- Use short, locally straight segments: "**Edgelets**"

  (Introduced by Eade & Drummond 2006)

- Add these to Klein & Murray 2007
  - Used along-side point features
  - Triangulate from key-frames with Canny detector
  - Adapt for motion blur

Edges in SLAM: Limitations

- High **velocities** are now OK...
- ... but sudden **accelerations** are still a problem!
  - These cause **unpredicted** blur
  - Edge detection fails!
- Mostly caused by fast saccades (snappy camera rotations)

Inter-frame rotation estimation

- Solution: rapidly guess rotation between every frame
- Assume **only rotation**
- Direct 3-DOF image-to-image minimisation

Inter-frame rotation estimation

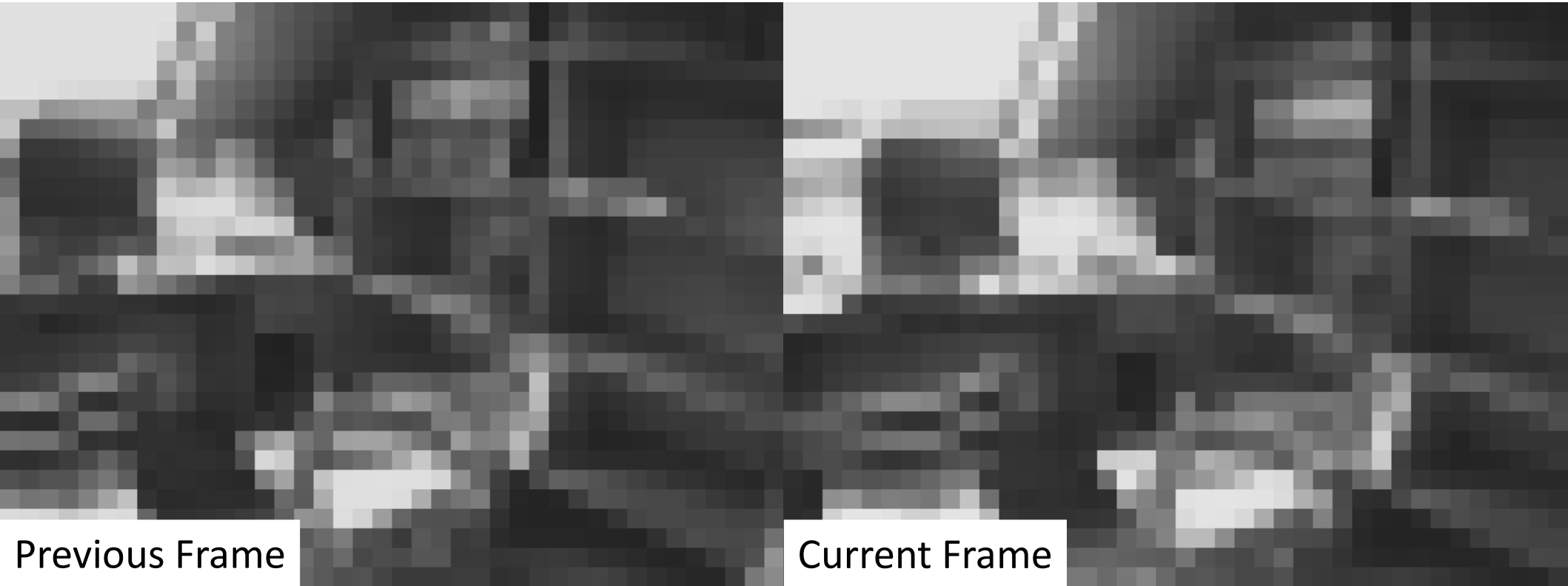
- Sub-sample two frames to 40x30 pixels

Previous Frame

Current Frame

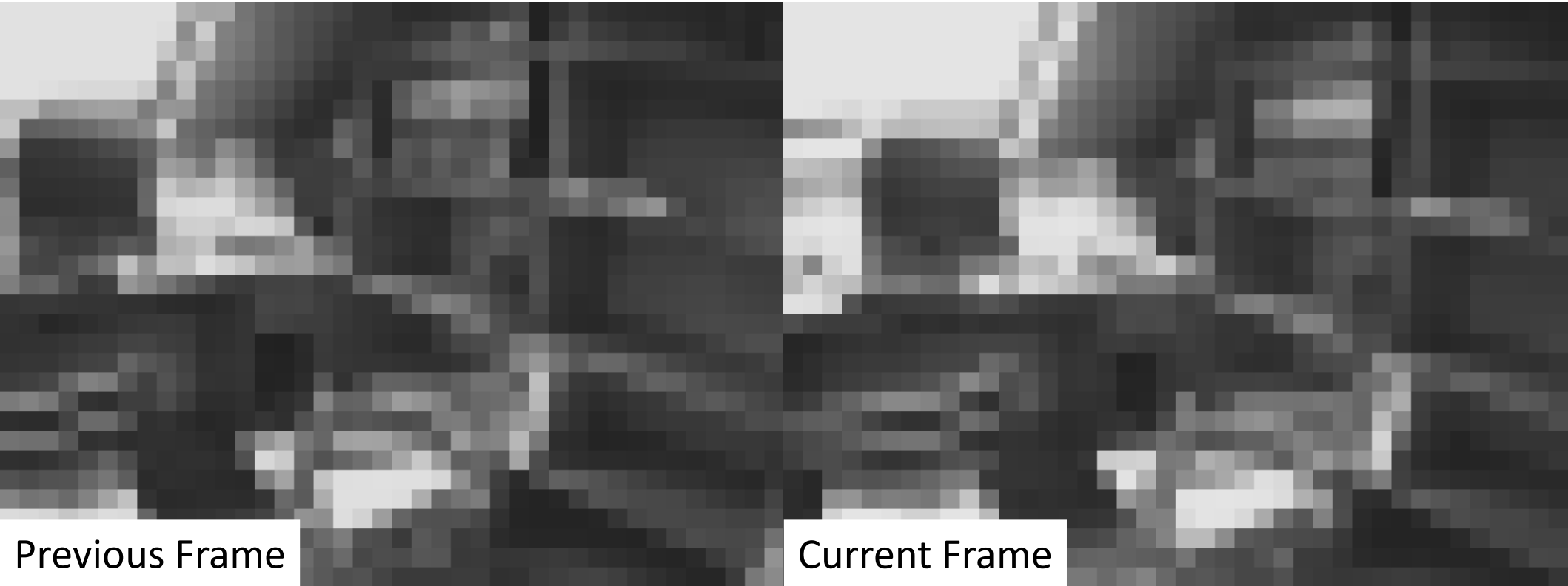
Inter-frame rotation estimation

- Sub-sample two frames to 40x30 pixels



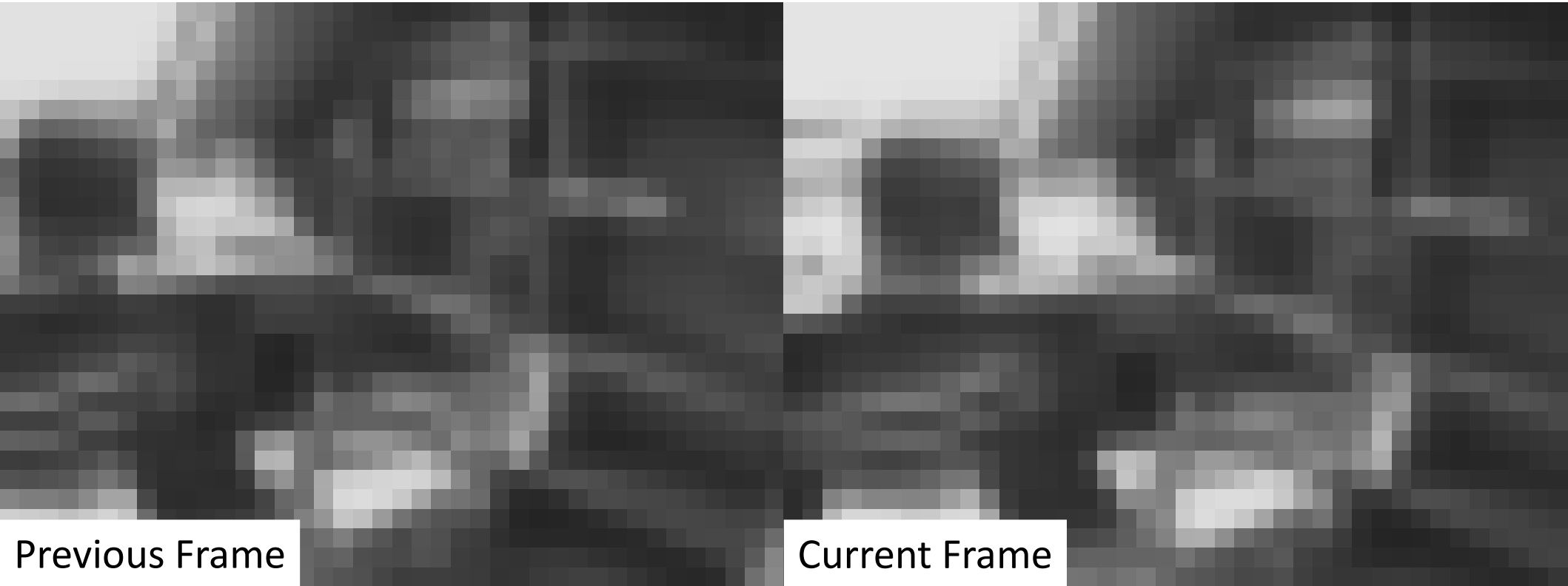
Inter-frame rotation estimation

- Apply Gaussian blur with 0.75 pixel sigma



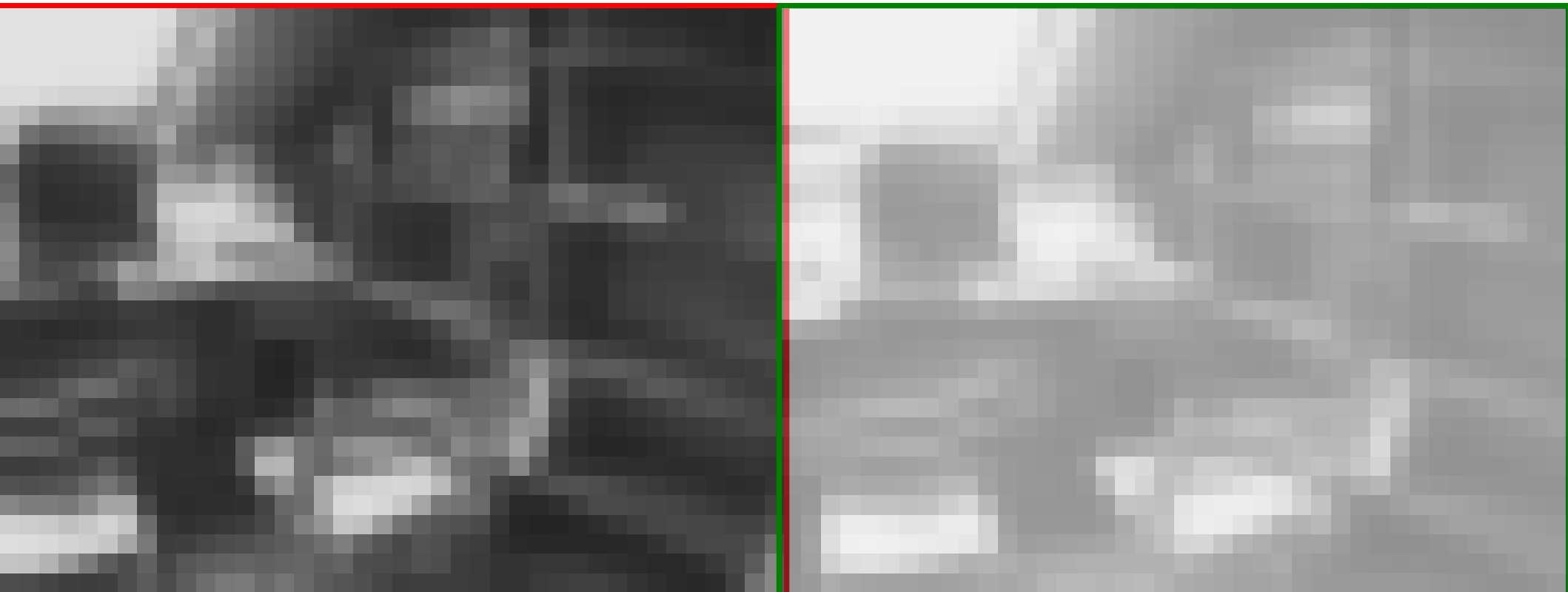
Inter-frame rotation estimation

- Apply Gaussian blur with 0.75 pixel sigma



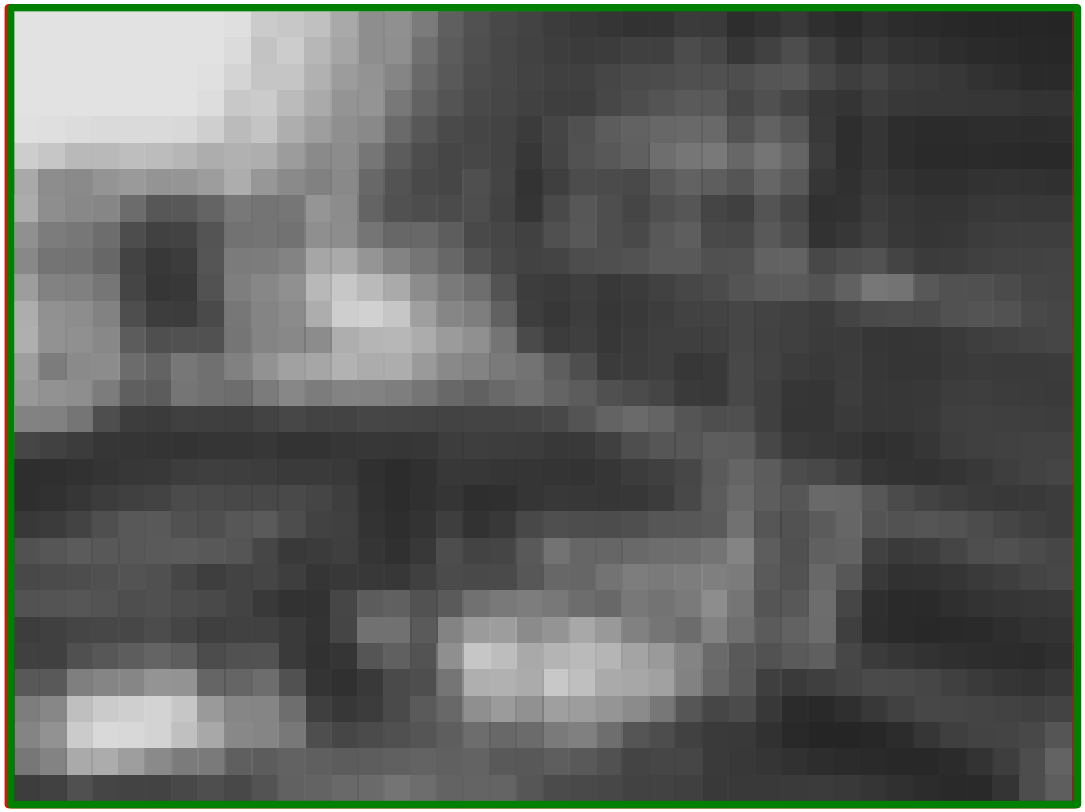
Inter-frame rotation estimation

- Align the images by direct 3-DOF minimisation



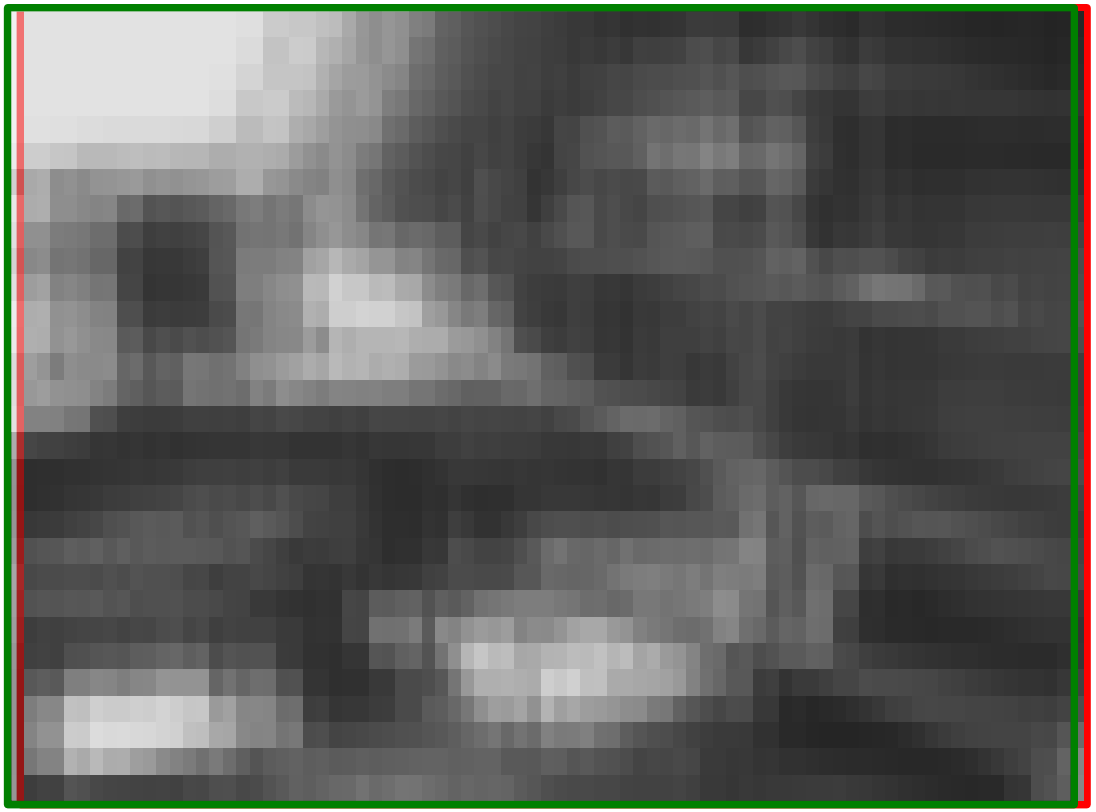
Inter-frame rotation estimation

- Align the images by direct 3-DOF minimisation



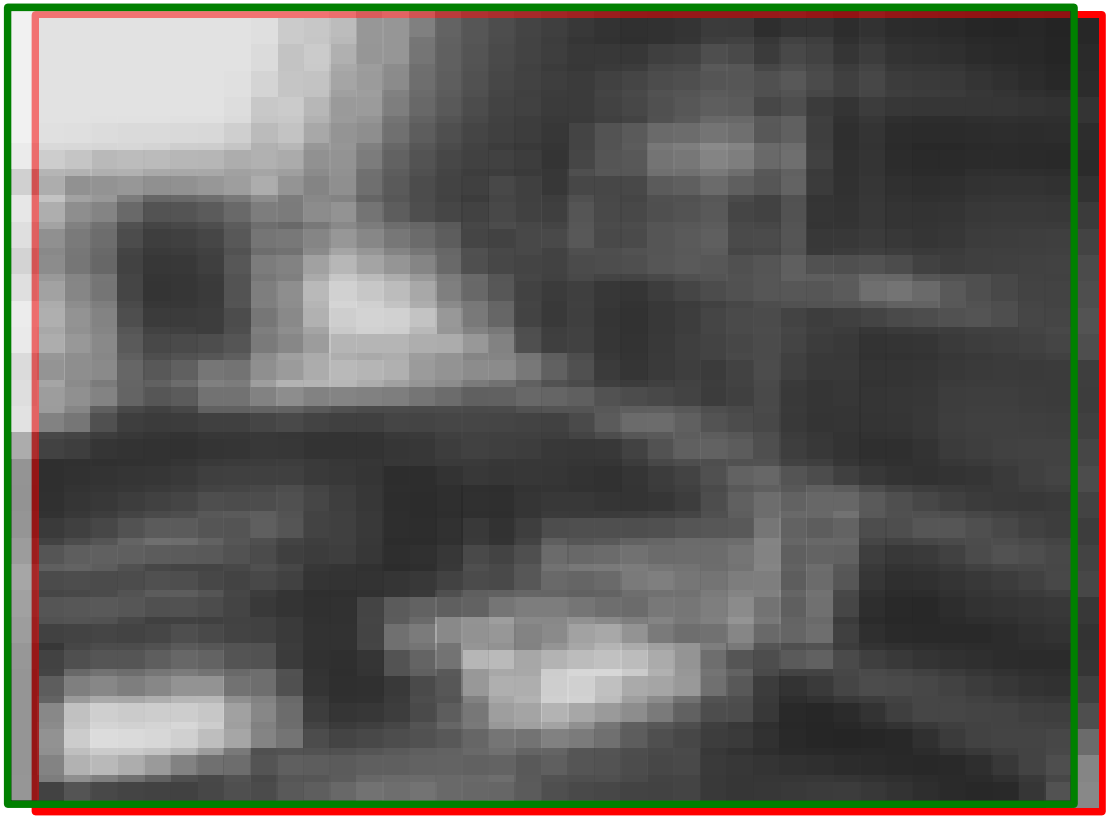
Inter-frame rotation estimation

- Align the images by direct 3-DOF minimisation



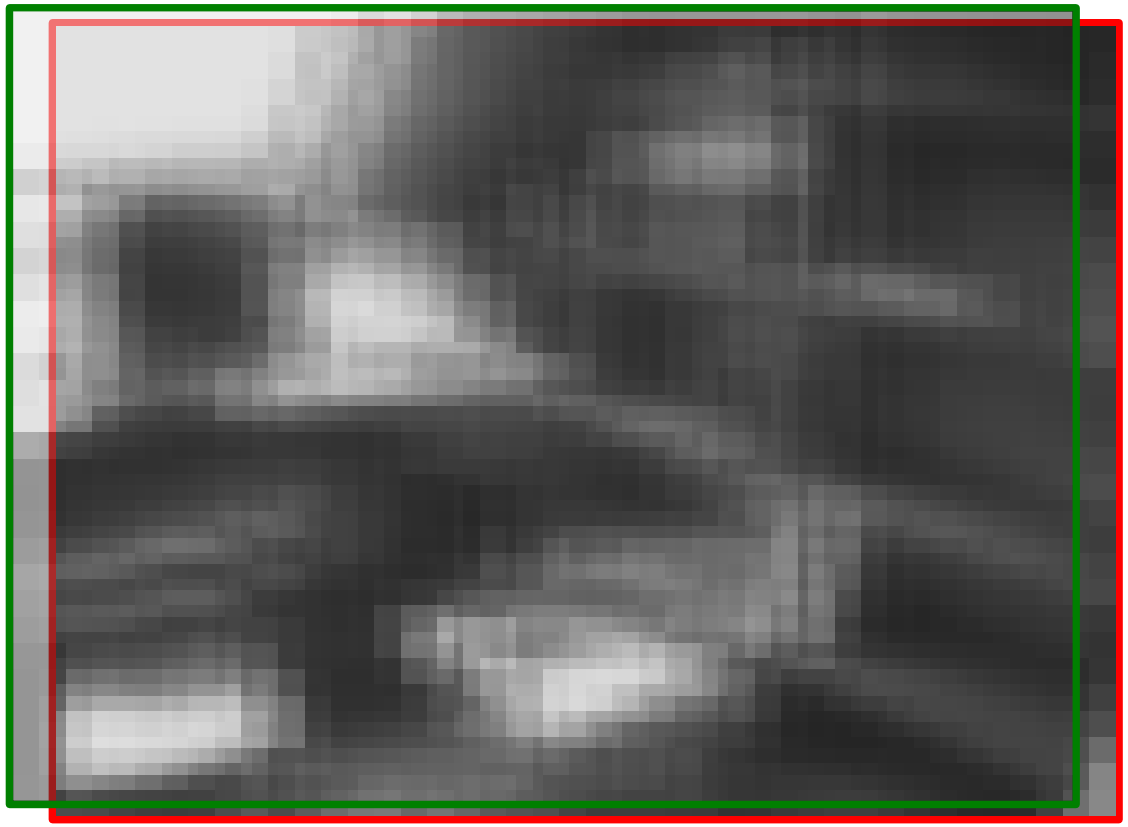
Inter-frame rotation estimation

- Align the images by direct 3-DOF minimisation



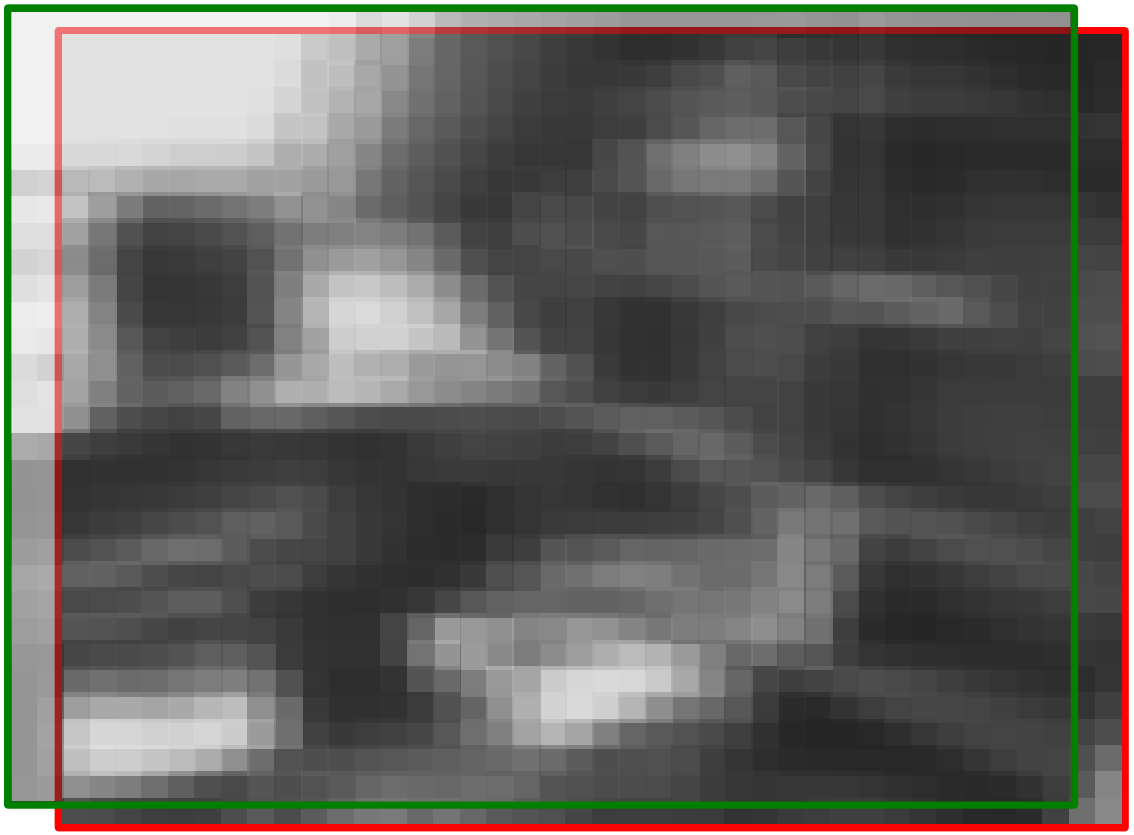
Inter-frame rotation estimation

- Align the images by direct 3-DOF minimisation



Inter-frame rotation estimation

- Align the images by direct 3-DOF minimisation



Inter-frame rotation estimation

- Align the images by direct 3-DOF minimisation

Inter-frame rotation estimation

- Transform this 2D warp to a 3D rotation

Inter-frame rotation estimation

- Solution: rapidly guess rotation between every frame
- Assume **only rotation**
- Direct 3-DOF image-to-image minimisation
- **Very fast**: 0.5ms for subsample, blur, and 10 iterations of ESM tracking

Recovery from Tracking Failure

- Even the most robust visual tracker can fail

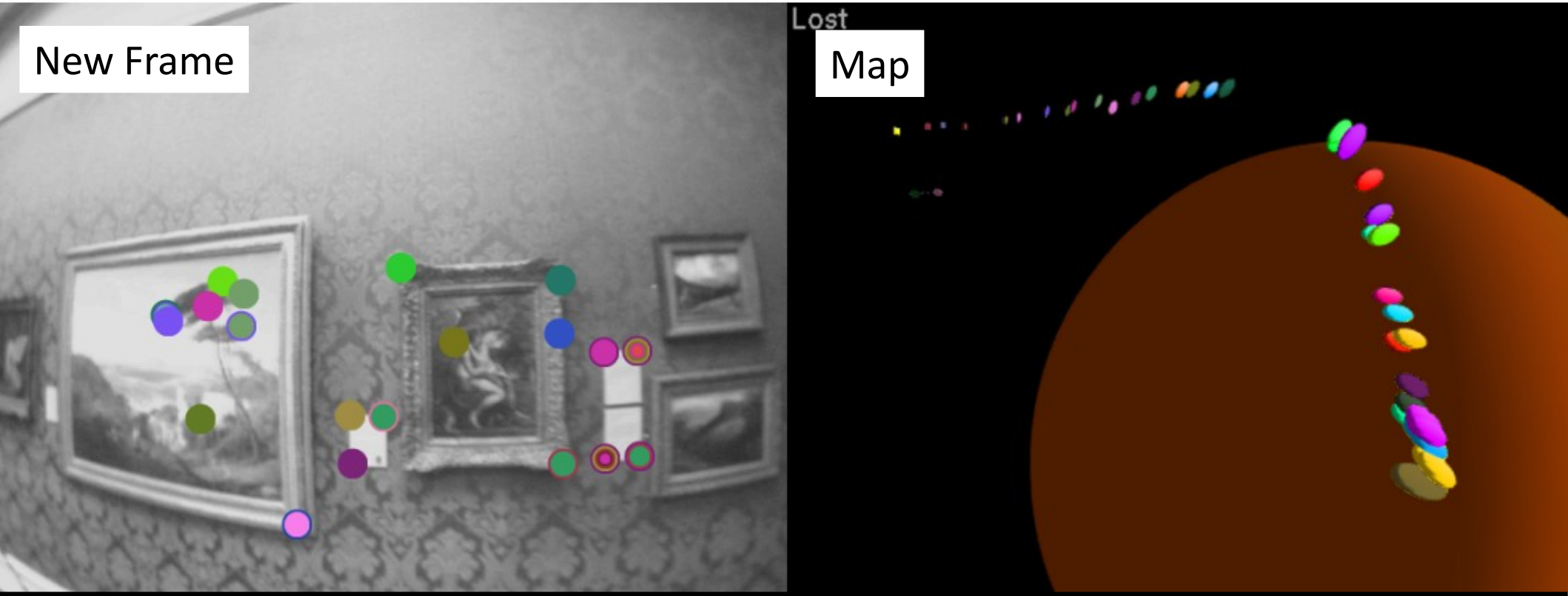
  (Just cover the lens with your hand!)
- A rapid failure recovery method is **essential**
- Some trackers do this the whole time:
  - ARToolkit
  - EPFL's tracking-by-detection
- Frame-to-frame trackers need separate procedure

Map-based Relocalisation for SLAM

- One approach is to relocalise from map features
- E.g. Williams et al ICCV'07

Map-based Relocalisation for SLAM

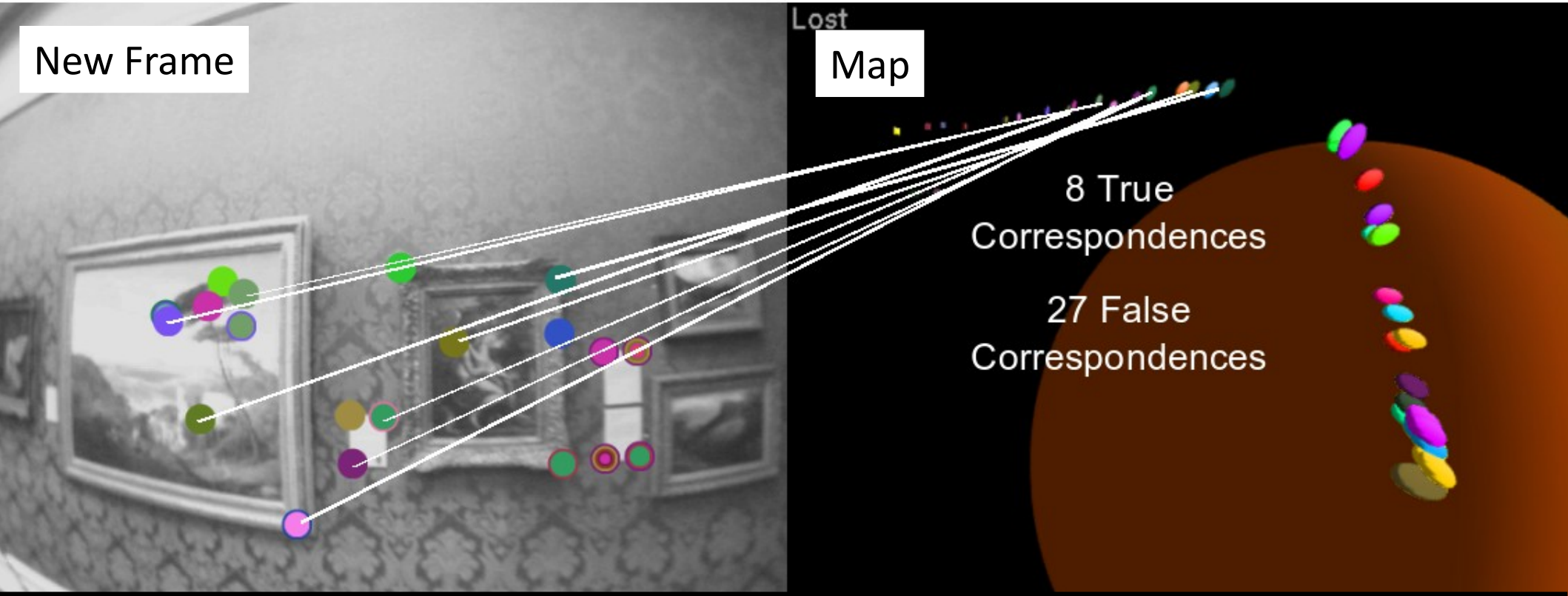
- One approach is to relocalise from map features
- E.g. Williams et al ICCV'07



1. Classify corner points in new view

Map-based Relocalisation for SLAM

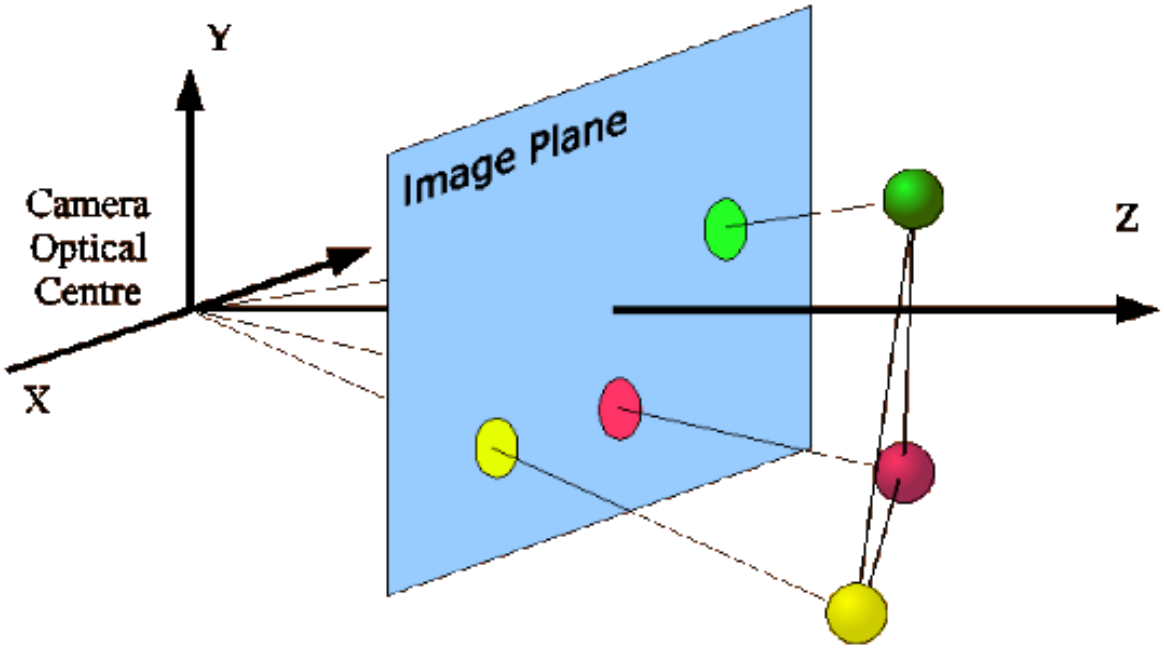
- One approach is to relocalise from map features
- E.g. Williams et al ICCV'07



1. Classify corner points in new view

Map-based Relocalisation for SLAM

- One approach is to relocalise from map features
- E.g. Williams et al ICCV'07



Map-based Relocalisation for SLAM

- Fairly wide-spread approach
- Many classifiers/matchers used:
  - Williams et al ICCV'07: Randomised Trees
  - Eade & Drummond BMVC'08: Light-weight SIFT + BOW
  - Checklov et al BMVC'08: Haar wavelets
  - Calonder et al ECCV'08: Pre-learned Randomised Trees

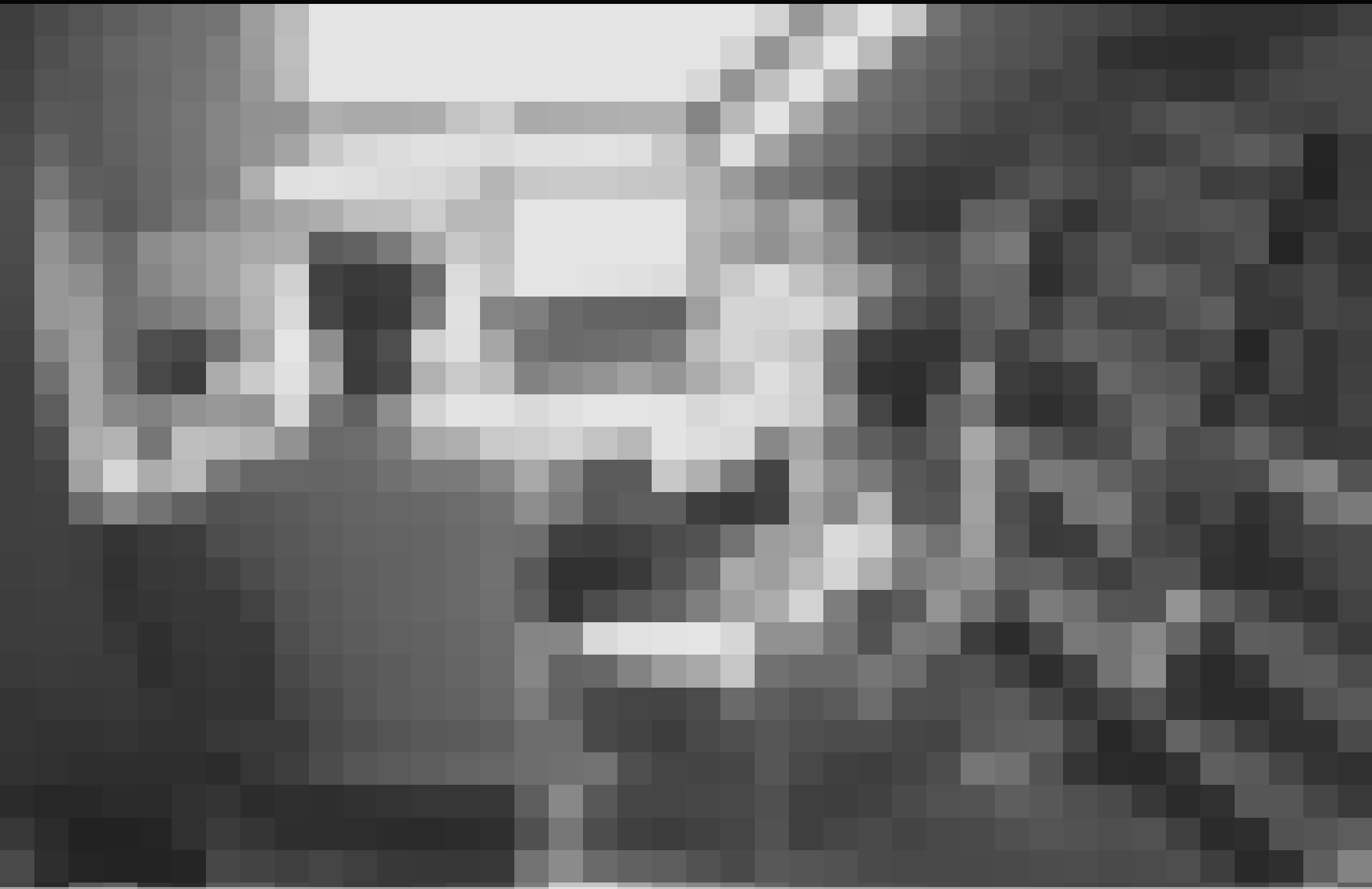
Map-based Relocalisation for SLAM

- Fairly wide-spread approach
- Many classifiers/matchers used:
  - Williams et al ICCV'07: Randomised Trees
  - Eade & Drummond BMVC'08: Light-weight SIFT + BOW
  - Checklov et al BMVC'08: Haar wavelets
  - Calonder et al ECCV'08: Pre-learned Randomised Trees
- **All fairly complex**

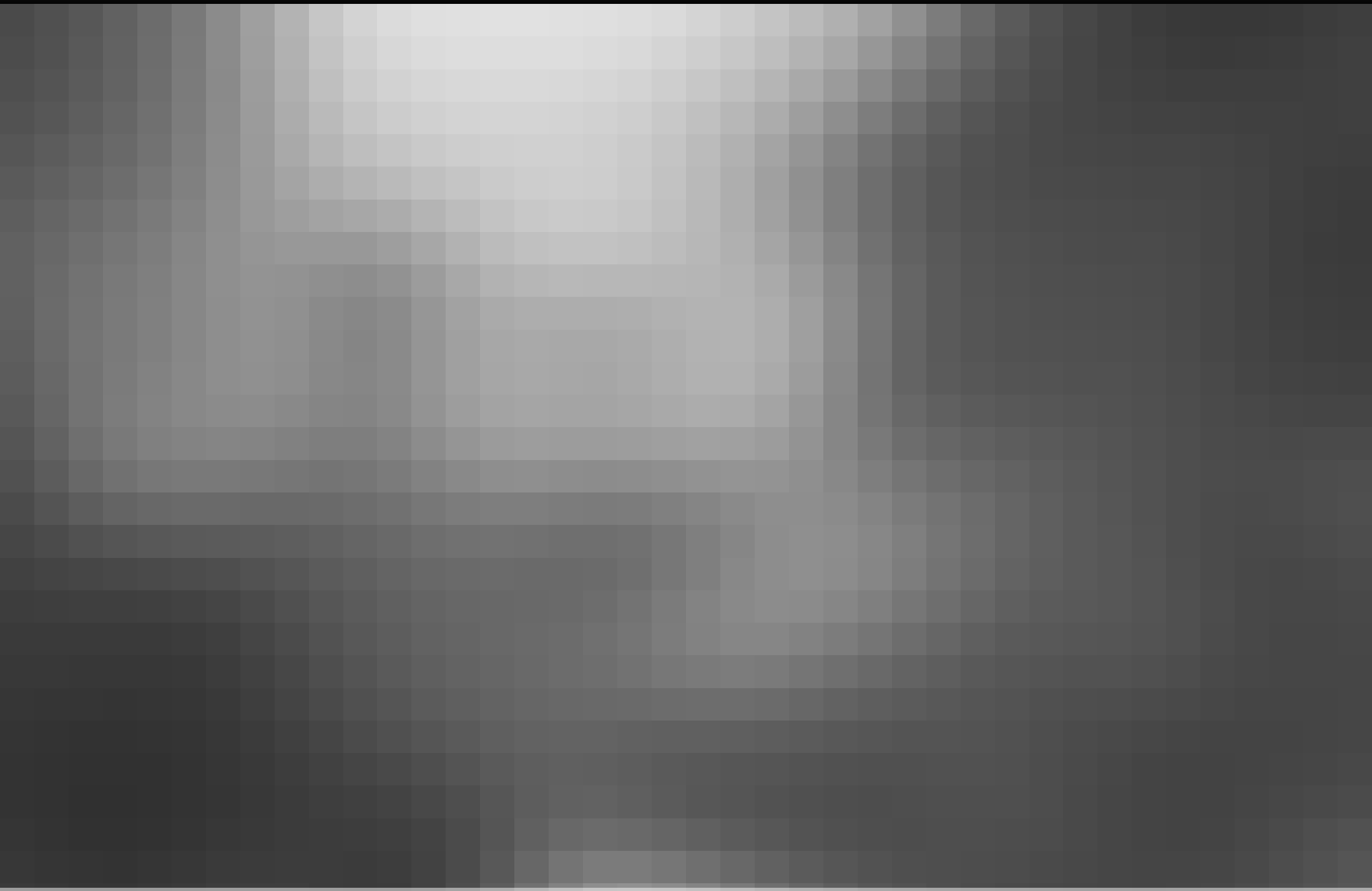
Trivial Relocalisation for SLAM

- Exploit the presence of **keyframes**
  - Know pose of each keyframe
  - Have image of each keyframe
- Compare current frame to all keyframe
- Set pose to the best match
- Optimise rotation

Raw Keyframe

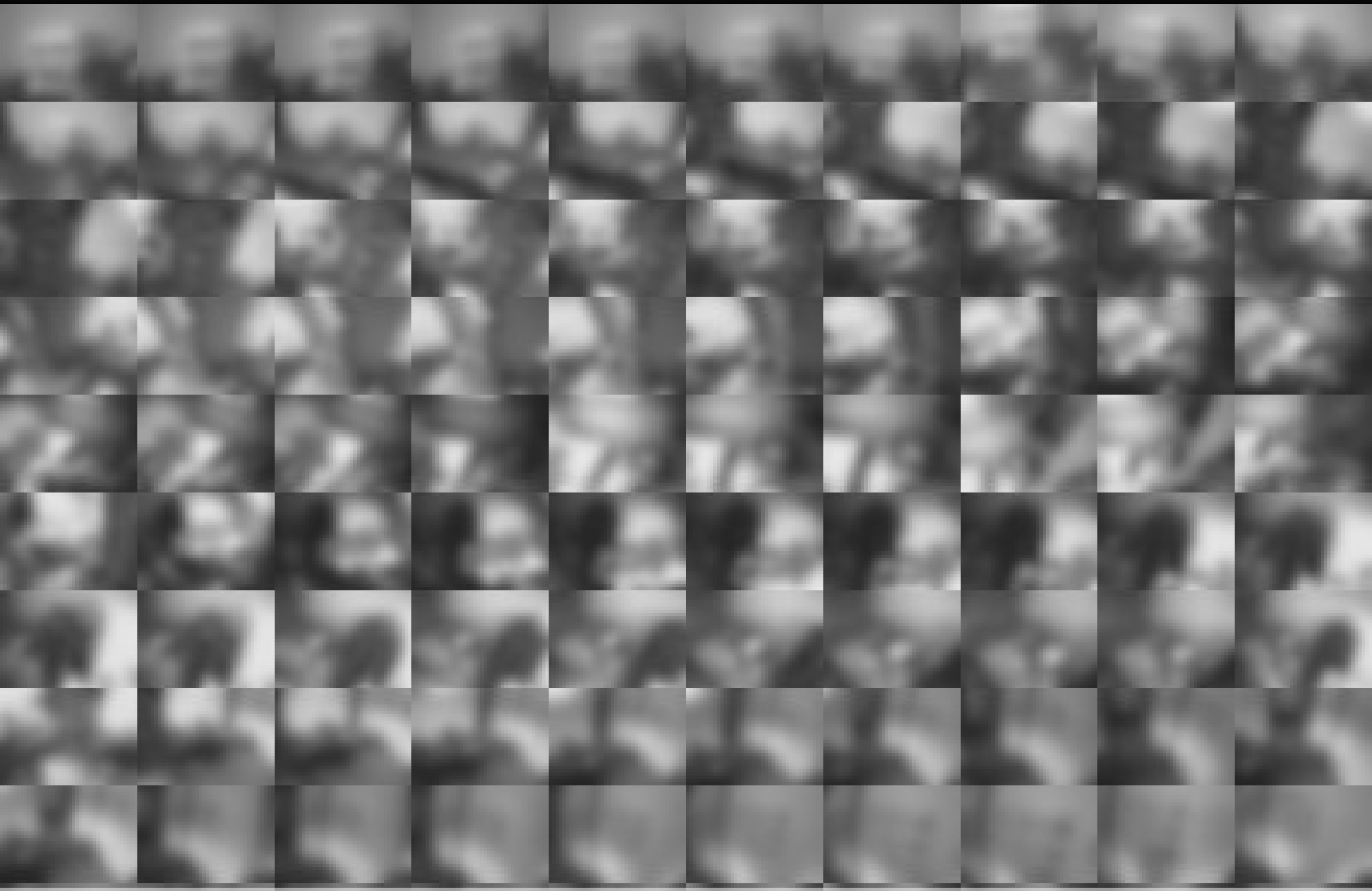


40x30 sub-sampled keyframe

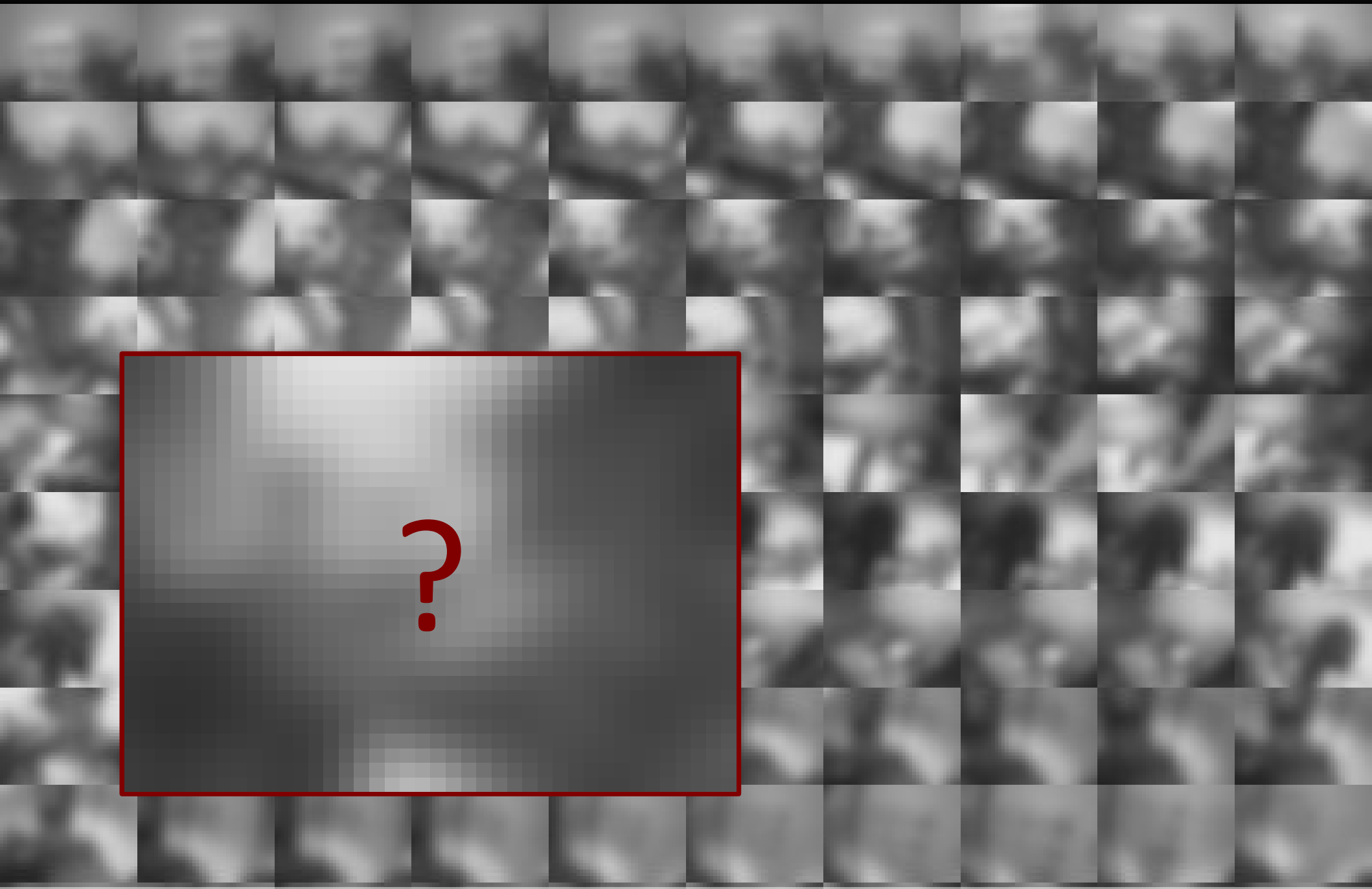


Blurred subsampled keyframe = Descriptor

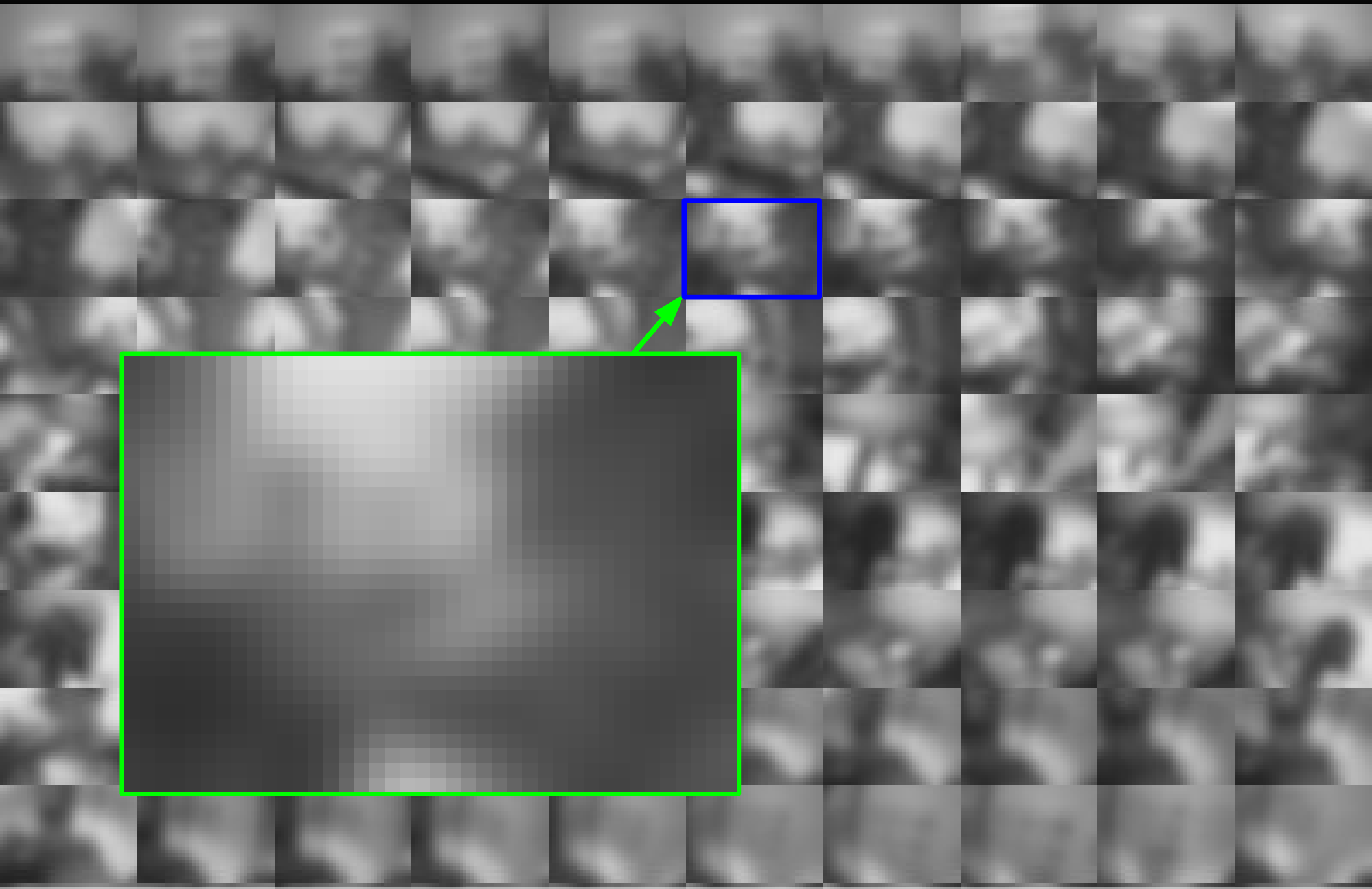
Agile tracking and SLAM



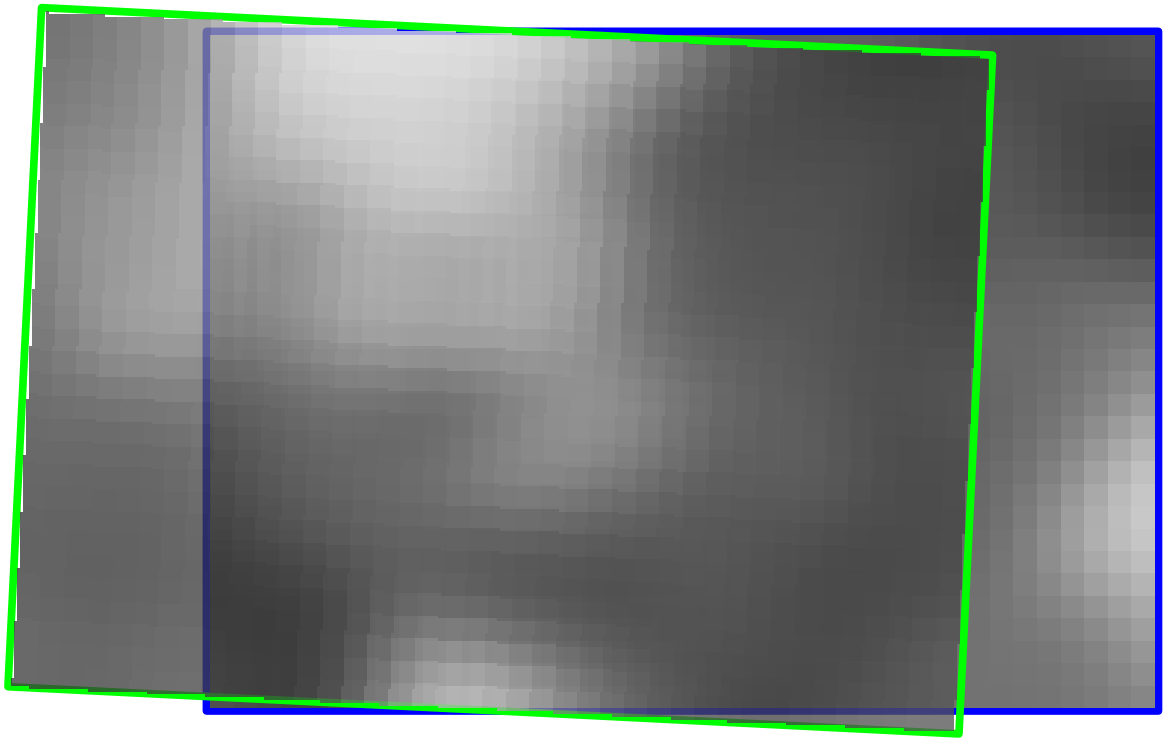
100 keyframes



Query view



Query view: best match



Align best match and re-start tracker

Trivial Relocalisation for SLAM

- Fast: **1.5ms** with 250 keyframes
- Predictable: works when camera near old pose
- Sensitive to scene changes
- Cannot relocalise upside-down

Agile Tracking and SLAM

- Can get reasonable performance from a single camera, using active search + initialisation
- Could go further with **extra sensors**..

  .. or even just **fiducials**
- Have not addressed other challenges:
  - Ligthing, clutter, occlusions, dynamic objects..

Some techniques for agile visual tracking and SLAM

Georg Klein
Active Vision Lab, Oxford

Workshop on Industrial Augmented Reality:
Needs & Solutions

ISMAR 2008
Cambridge, UK