High performance tracking

Ed Rosten

Supervisor: Dr. Tom Drummond

UNIVERSITY OF CAMBRIDGE

Overview of tracking

- Trackers have complementary properties
  - Robust, drift free, accurate, efficient, etc. . .
- Failures are also complementary
  - Fragile, drift, inaccurate, etc. . .
- Combining trackers combines strengths

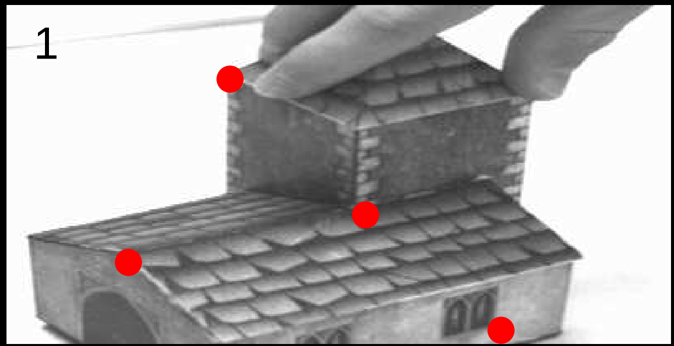
Overview of tracking

- Trackers have complementary properties
  - Robust, drift free, accurate, efficient, etc…
- Failures are also complementary
  - Fragile, drift, inaccurate, etc…
- Combining trackers combines strengths
- 6 DOF (degrees of freedom) tracking (position and orientation in 3D)
- Robust feature based tracker
  - New feature detector
- Edge based tracker
- Combining the trackers
  - Much better

Point based tracking

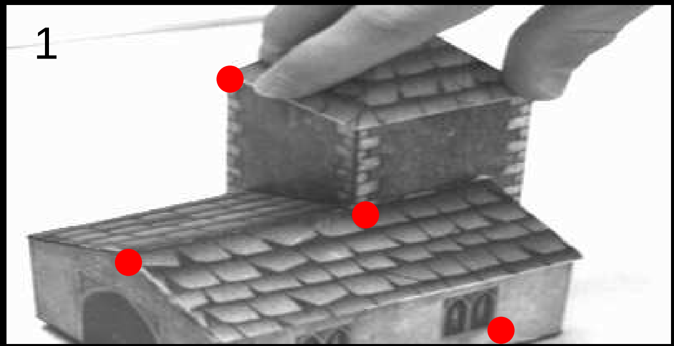
Point based tracking

Detect features

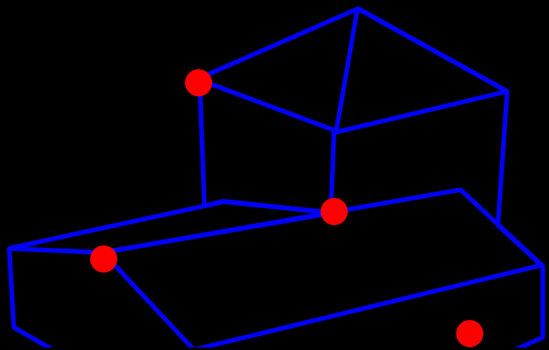


Point based tracking

Detect features



2



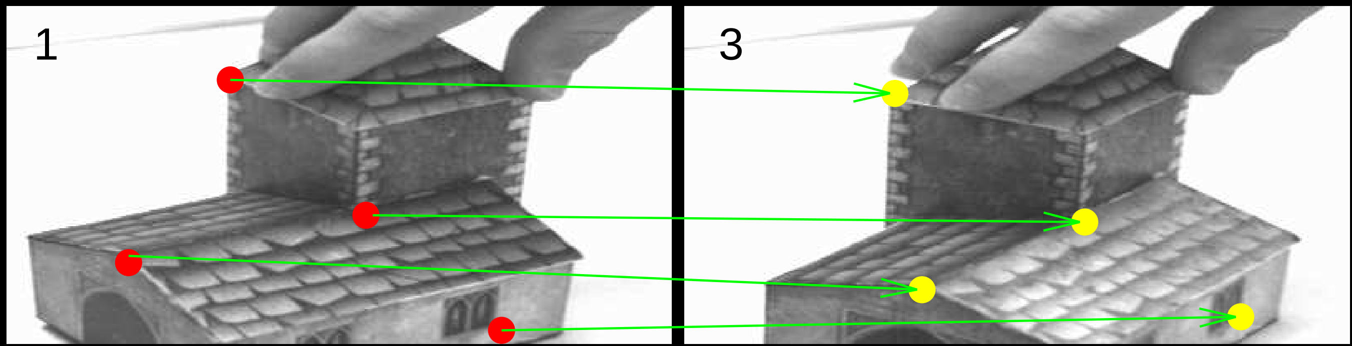
Project features on to model.

Drift occurs here

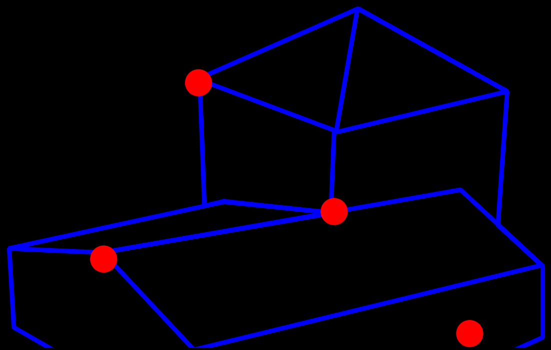
Point based tracking

Detect features

Detect and match features in next frame



2



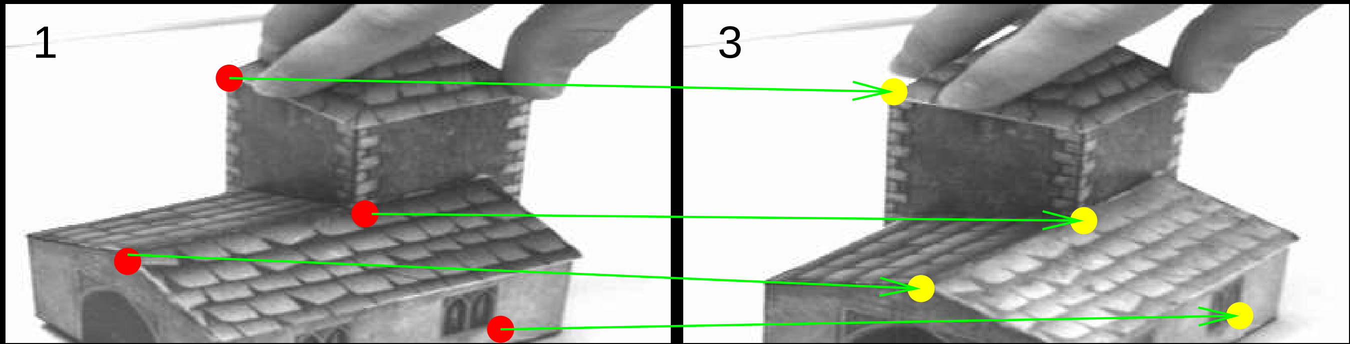
Project features on to model.

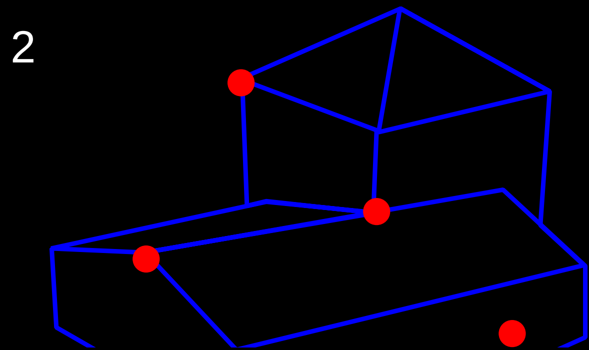
Drift occurs here

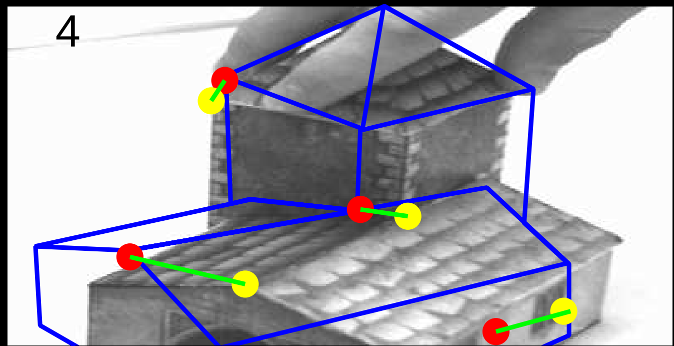
Point based tracking

Detect features

Detect and match features in next frame







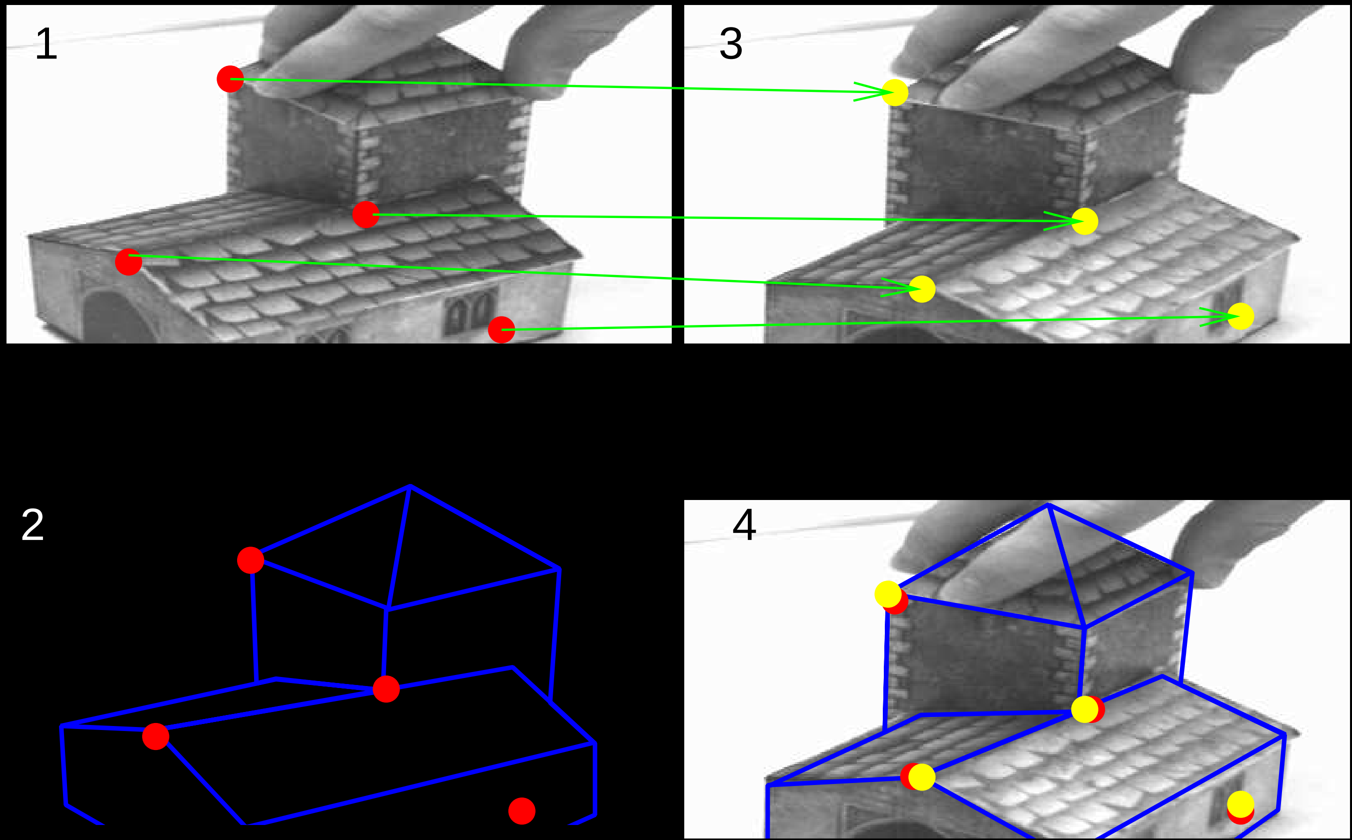
Project features on to model. Drift occurs here

Alter pose to minimize reprojection error

Point based tracking

Detect features

Detect and match features in next frame



Project features on to model. Drift occurs here

Alter pose to minimize reprojection error

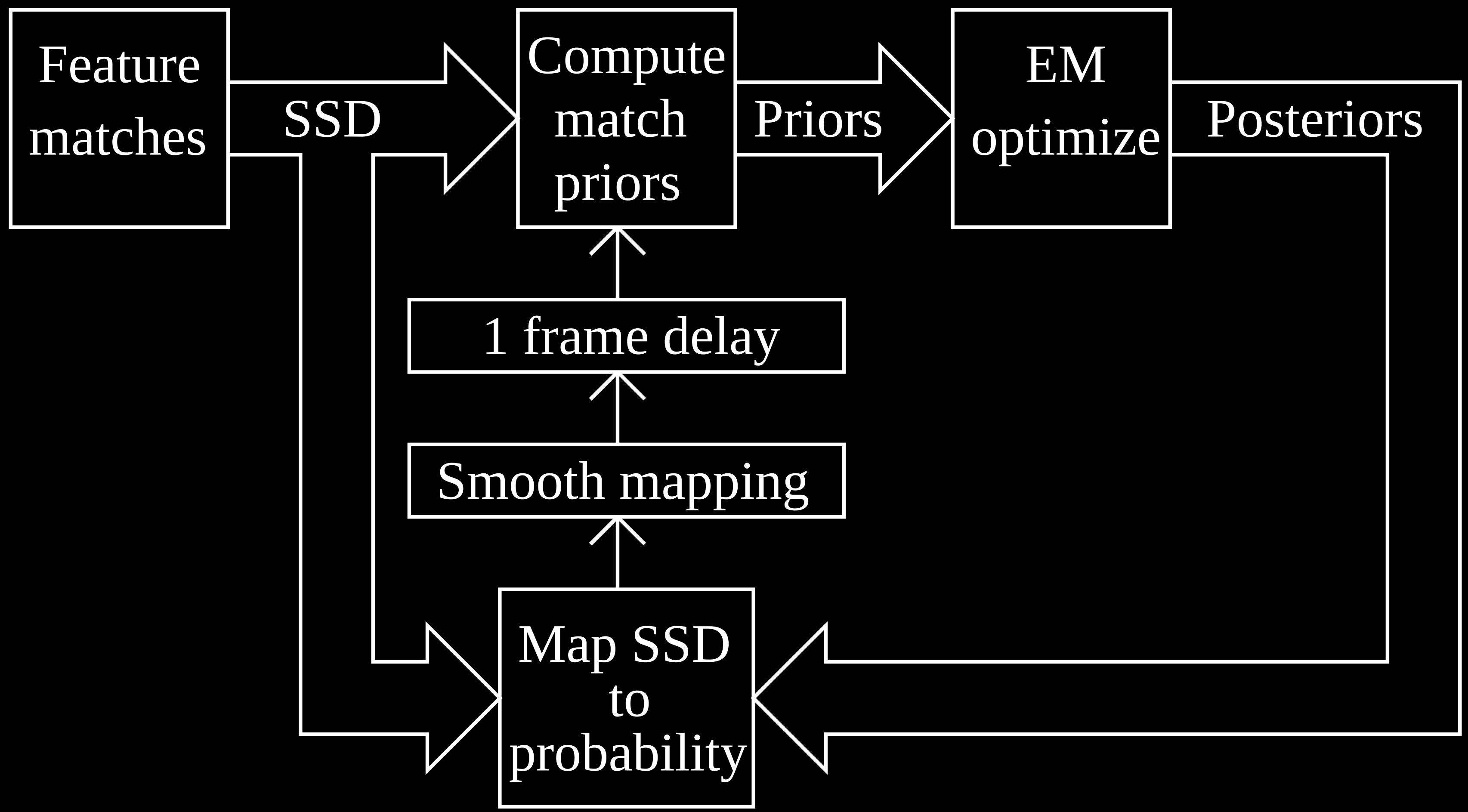
Position optimization

- Sometimes $> 90\%$ outliers (even with SIFT!)
  - Robust optimize required
- Use EM
  - Mixture model is
    Gaussian (inliers) + uniform (outliers)

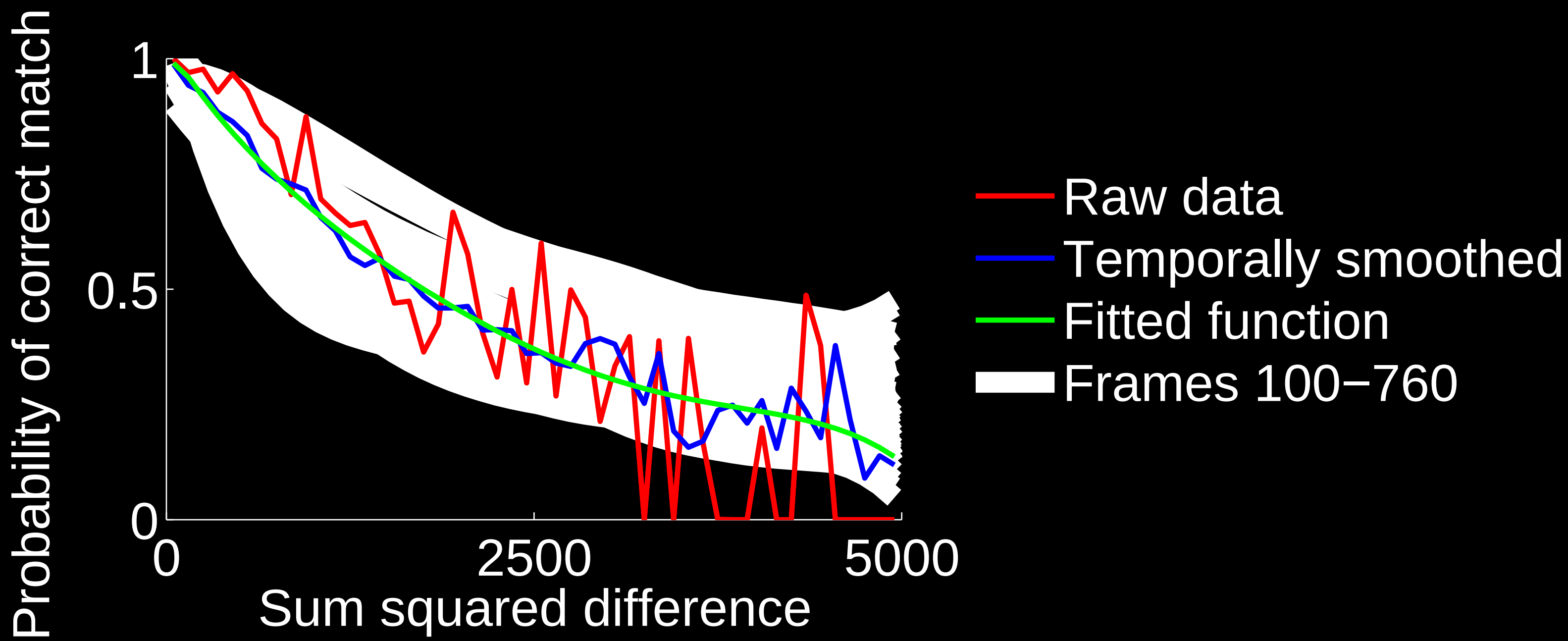
1. Compute $P(\text{match} \in \text{inliers} \mid \mu, \text{mixture model})$
2. Recompute μ (using Gauss-Newton)
3. Recompute mixture model

- SSD has some information about inlier probability
  - If only we knew the relationship...

Matching prior

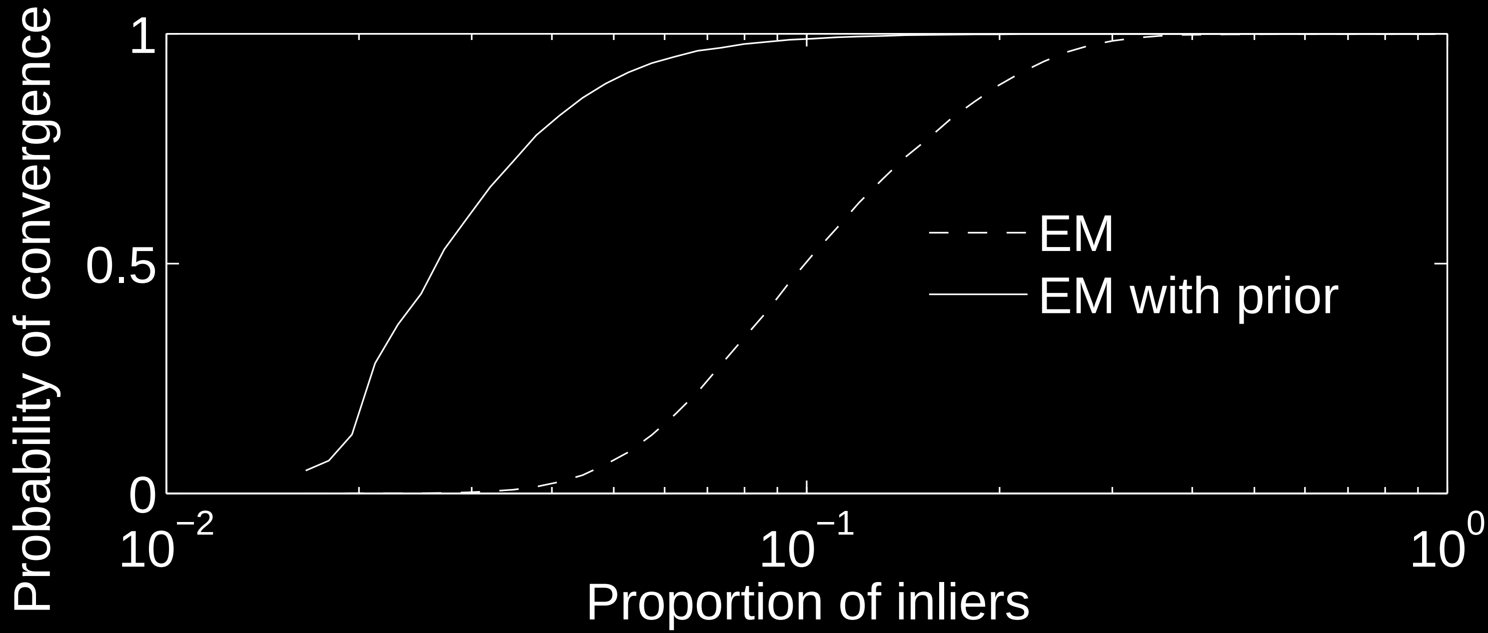


Matching prior



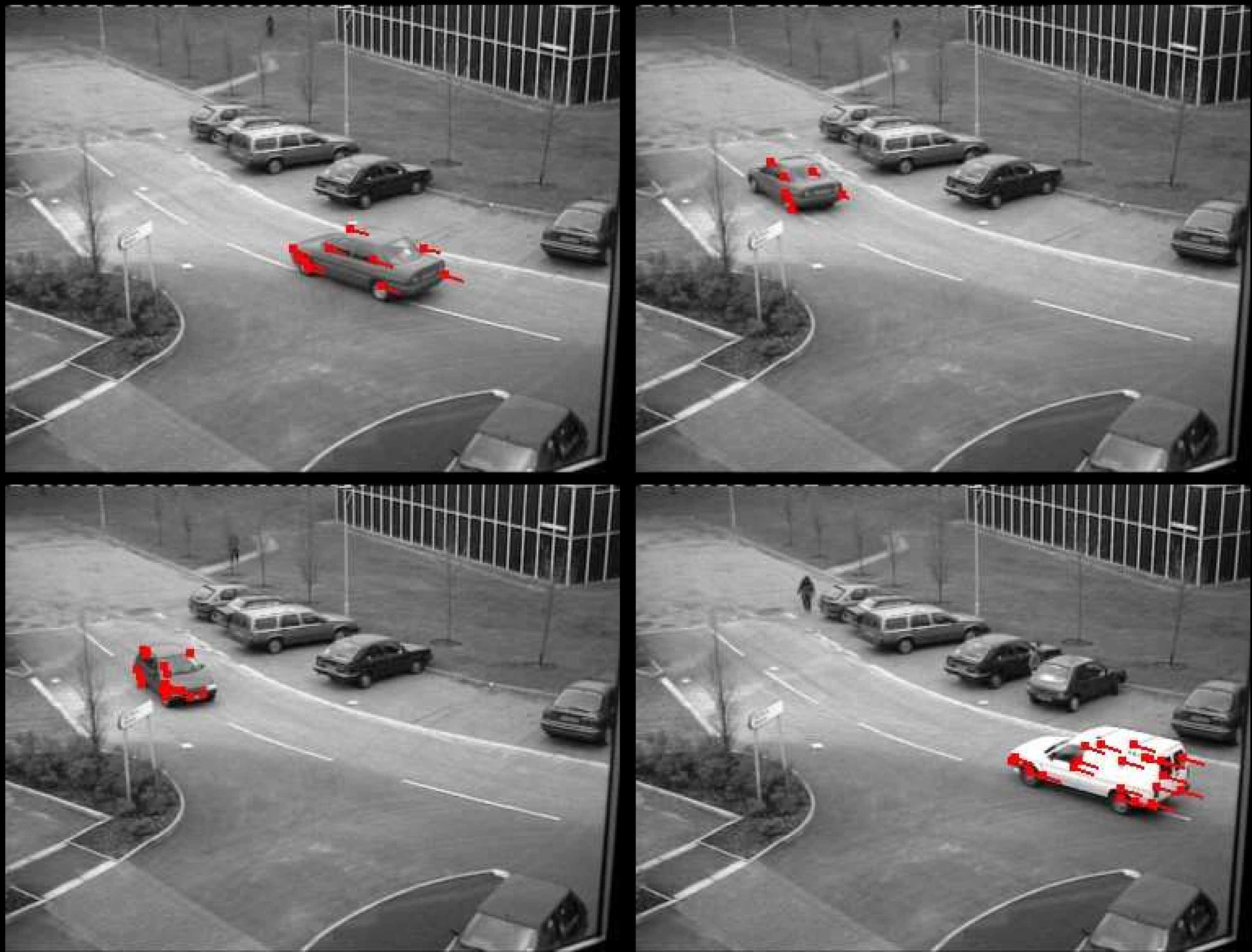
- EM provides probability that a match is correct
- SSD for each match is known
- Compute smooth function mapping SSD to probability
- Use function to compute priors for each match next frame

Matching prior



- EM provides probability that a match is correct
- SSD for each match is known
- Compute smooth function mapping SSD to probability
- Use function to compute priors for each match next frame

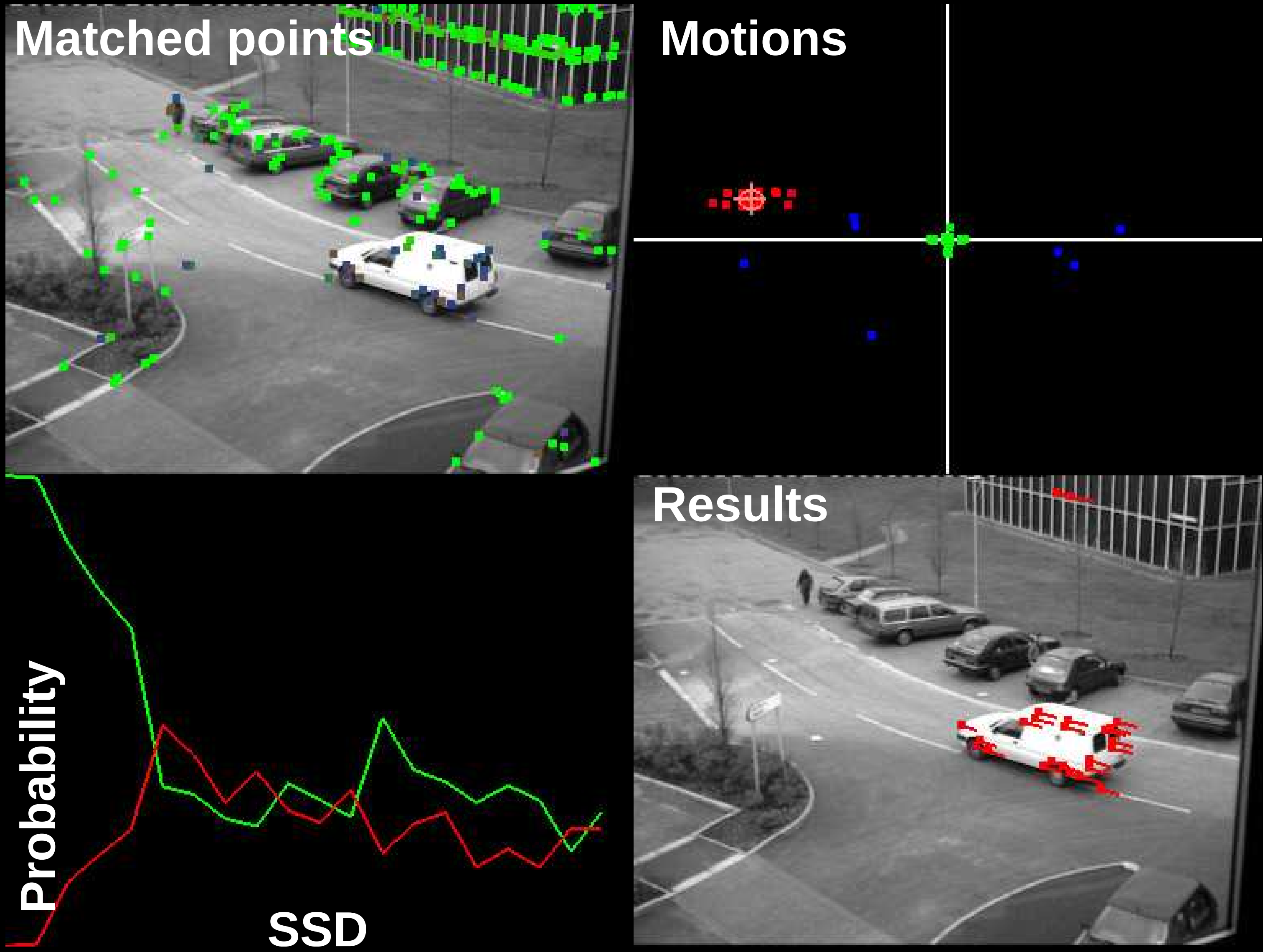
2D example - Motion tracking



Analysis

- Cel style animation - foreground and background can appear anywhere
- 3 kinds of match
  - Background (small offset)
  - Motion (many points with coherent offset)
  - Mismatch (ransom offset)
- Model offsets as GMM
- Compute $P(\text{match} \in \text{background} \mid \text{SSD})$, $P(\text{match} \in \text{motion} \mid \text{SSD})$, $P(\text{match} \in \text{mismatches} \mid \text{SSD})$

Video



2D example

- Not really tracking!
- Very simplistic:
  - No model of car
  - No motion model
  - No background model
  - Exactly one motion per frame modelled
- ...but it still works
- Possible improvements
  - Better modelling
  - Combining with other trackers

Back to 6 DOF tracking...

Measurement Properties

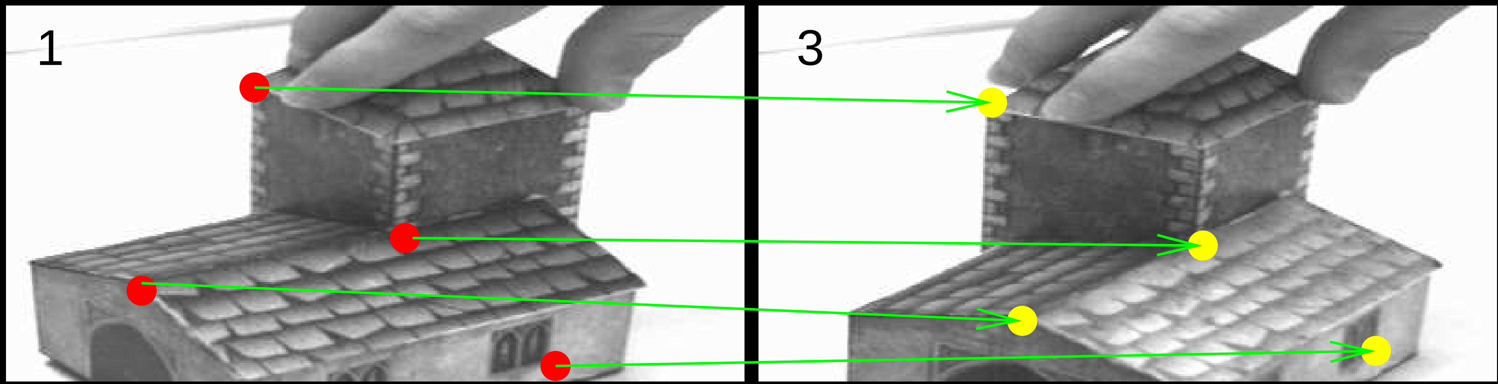
- Point based tracking
  - Requires
    - 3D point cloud
  - Provides
    - Robust differential measurements...
    - ...with approximately Gaussian posterior

Measurement Properties

- Point based tracking
  - Requires
    - 3D point cloud
  - Provides
    - Robust differential measurements...
    - ...with approximately Gaussian posterior
- Relies on full frame matching

Robust differential measurements

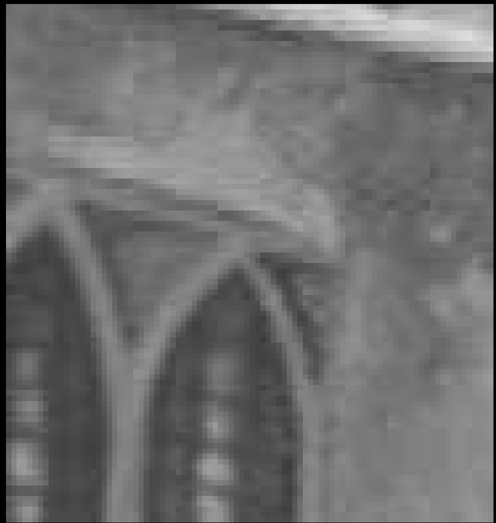
Detect and match features in next frame



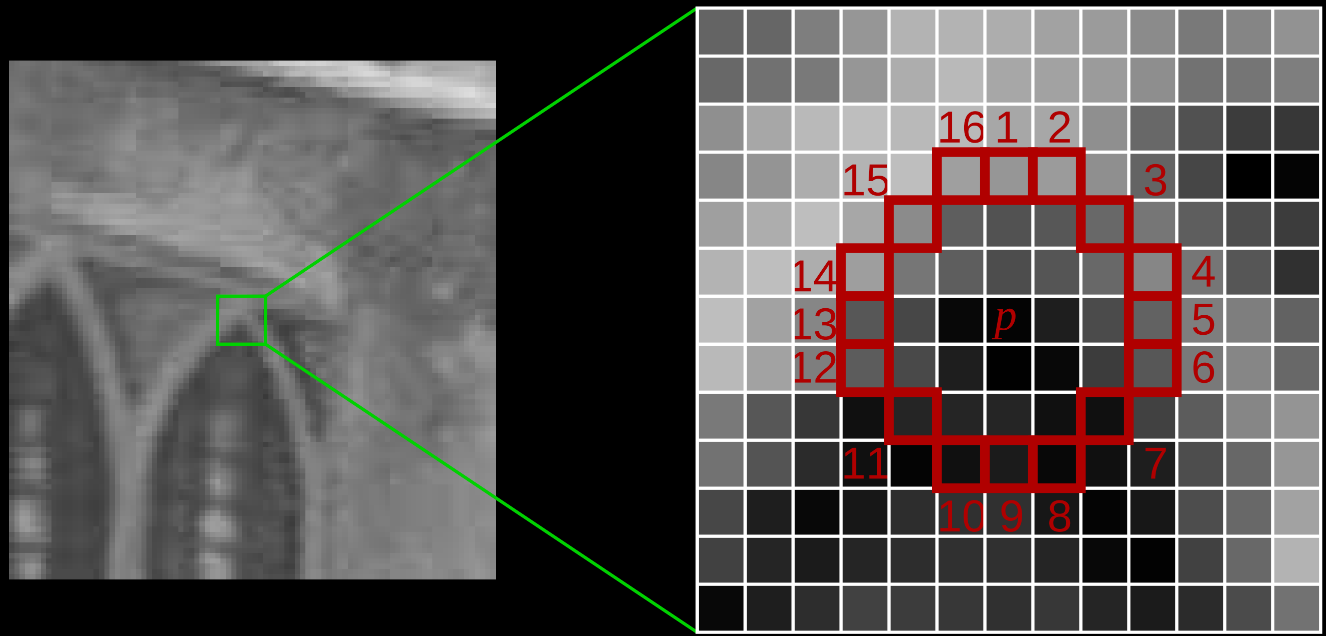
- Full frame matching makes it robust to large motions
- Detecting features in a whole frame is slow
- Matching can be $O(n^2)$
- Solution to detection is...

FAST feature detection

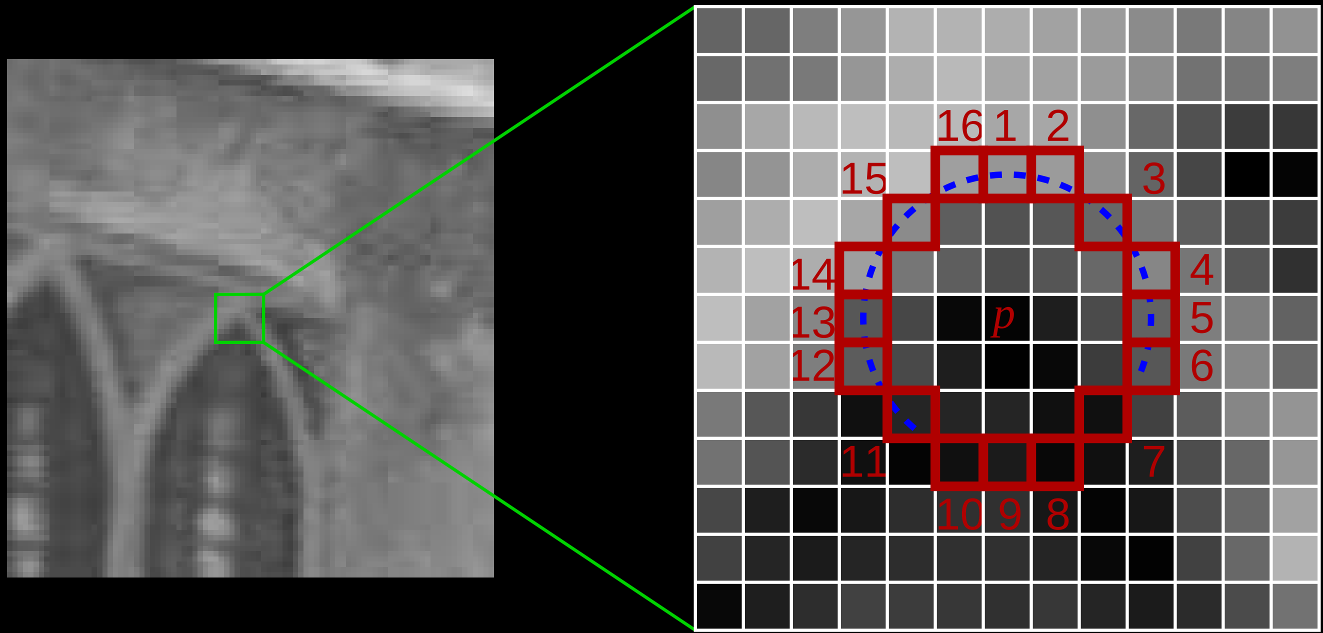
The FAST feature detector



The FAST feature detector

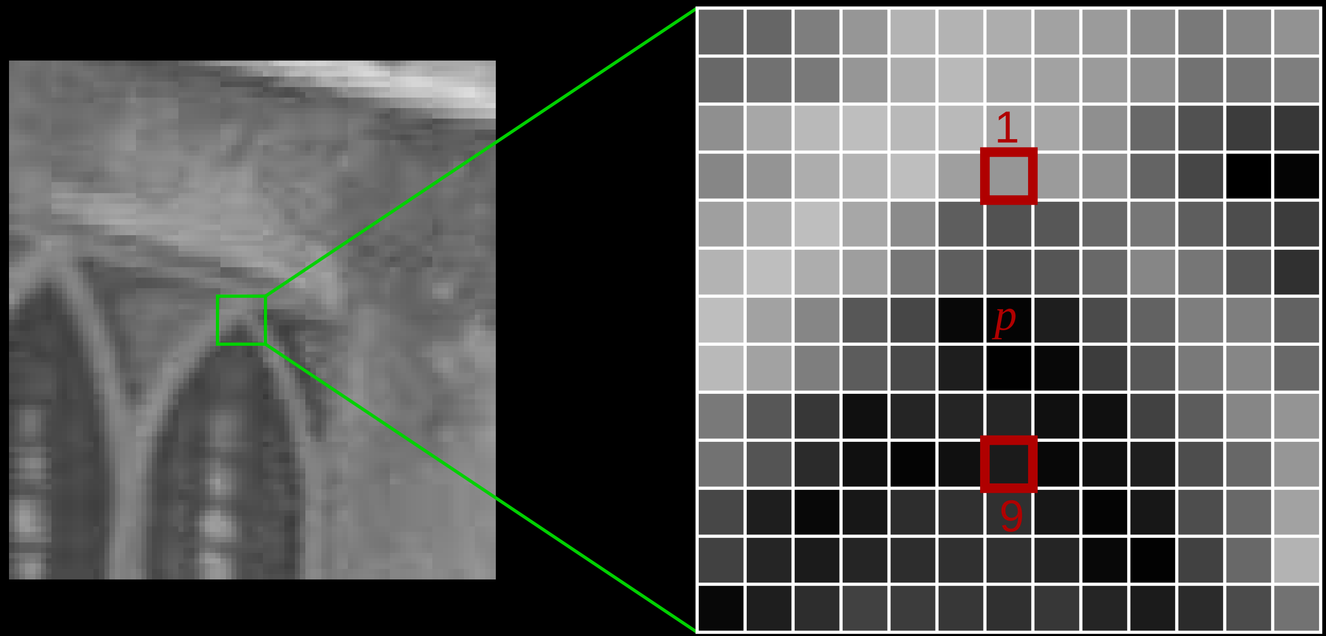


The FAST feature detector



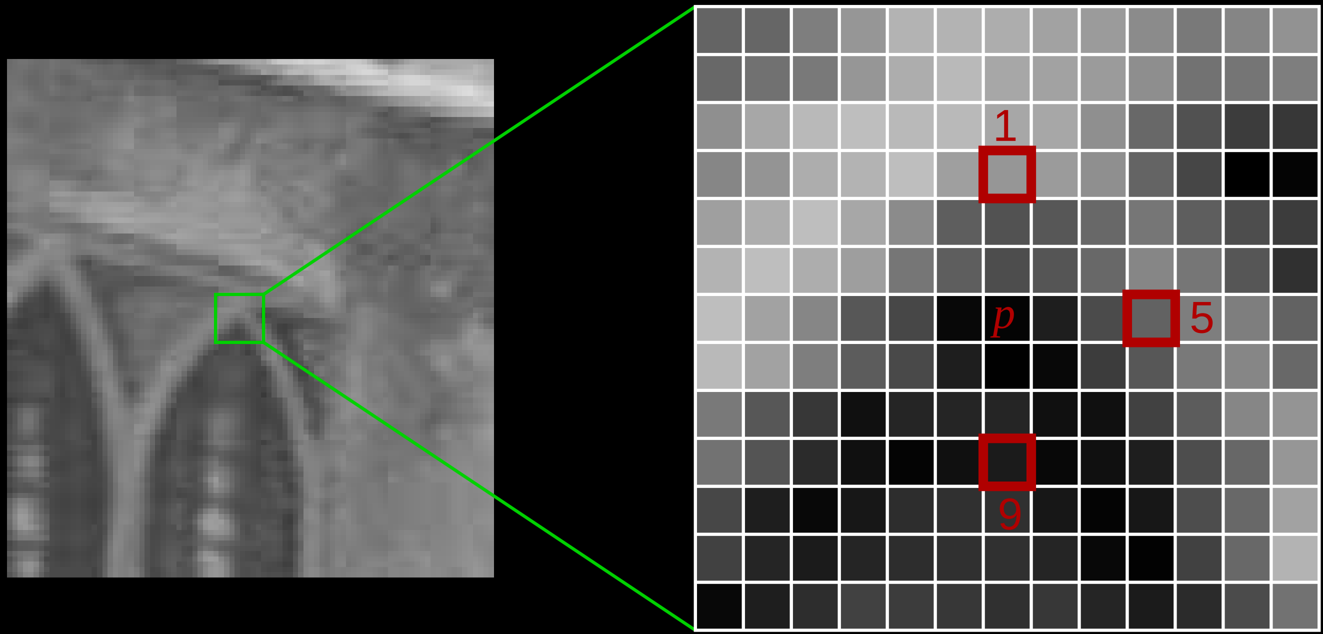
- $\geq$ 12 contiguous pixels brighter than p+*threshold*

The FAST feature detector



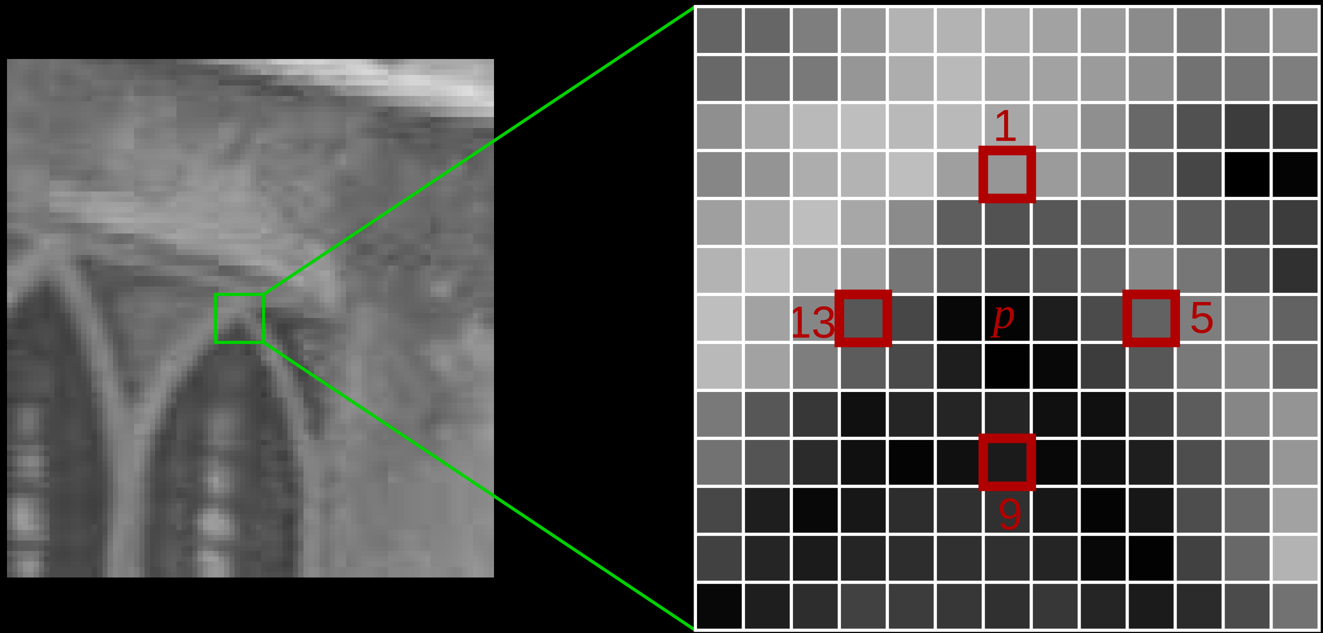
- $\geq$ 12 contiguous pixels brighter than *p+threshold*
- Rapid rejection by testing 1, 9

The FAST feature detector



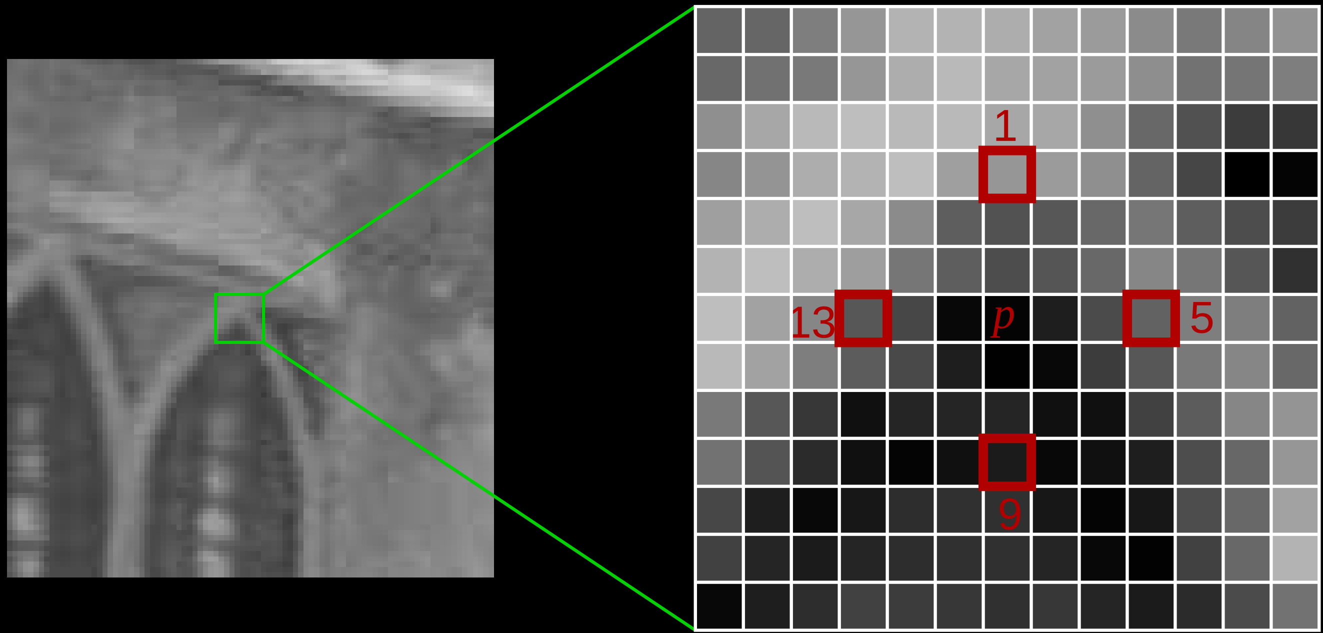
- $\geq$ 12 contiguous pixels brighter than *p+threshold*
- Rapid rejection by testing 1, 9, 5

The FAST feature detector



- $\geq$ 12 contiguous pixels brighter than *p+threshold*
- Rapid rejection by testing 1, 9, 5 then 13

The FAST feature detector



- $\geq$ 12 contiguous pixels brighter than *p+threshold*
- Rapid rejection by testing 1, 9, 5 then 13
- 1.59ms (Opteron 2.6GHz) - 8% of available CPU time
- Source code available
- `http://savannah.nongnu.org/projects/libcvd`

Problems

- Corners are clustered together
  - Use non maximal suppression

$$V = \max \begin{cases} \sum(\text{pixel values} - p) \text{ if } (\text{value} - p) > t \\ \sum(p - \text{pixel values}) \text{ if } (p - \text{value}) > t \end{cases}$$

  - Bias bigger differences over more points

Problems

- Corners are clustered together
  - Use non maximal suppression

$$V = \max \begin{cases} \sum(\text{pixel values} - p) \text{ if } (\text{value} - p) > t \\ \sum(p - \text{pixel values}) \text{ if } (p - \text{value}) > t \end{cases}$$

  - Bias bigger differences over more points
- High speed test does not generalize well to $n < 12$
- Choice of high speed test is not optimal
- Results of test are thrown away

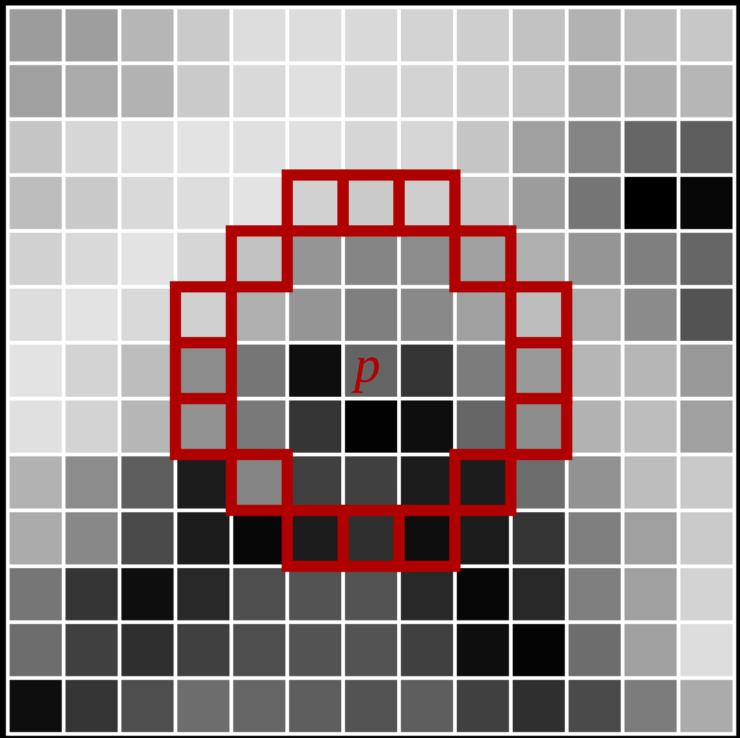
Problems

- Corners are clustered together
  - Use non maximal suppression

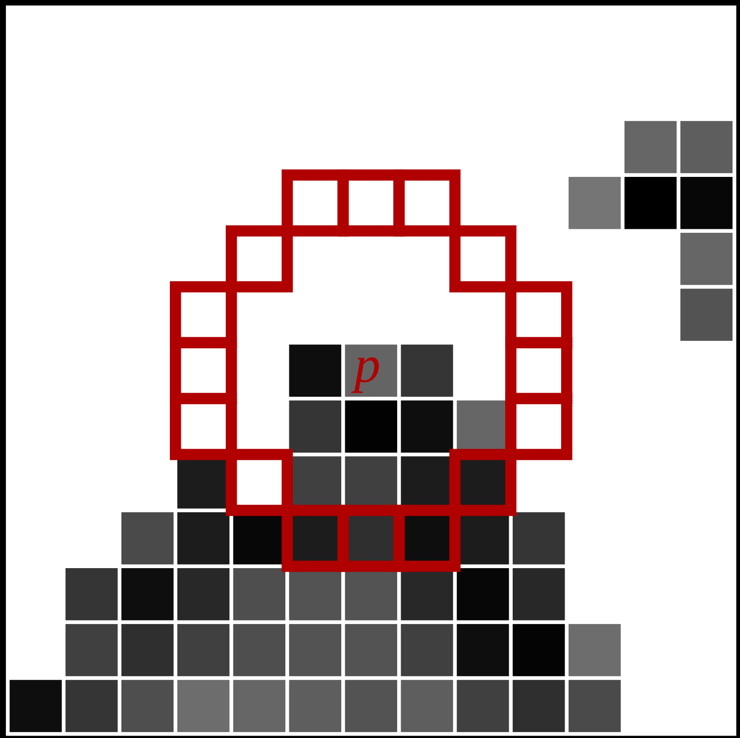
$$V = \max \begin{cases} \sum(\text{pixel values} - p) \text{ if } (\text{value} - p) > t \\ \sum(p - \text{pixel values}) \text{ if } (p - \text{value}) > t \end{cases}$$

  - Bias bigger differences over more points
- High speed test does not generalize well to $n < 12$
- Choice of high speed test is not optimal
- Results of test are thrown away
- Learn question ordering

Analysis of pixels

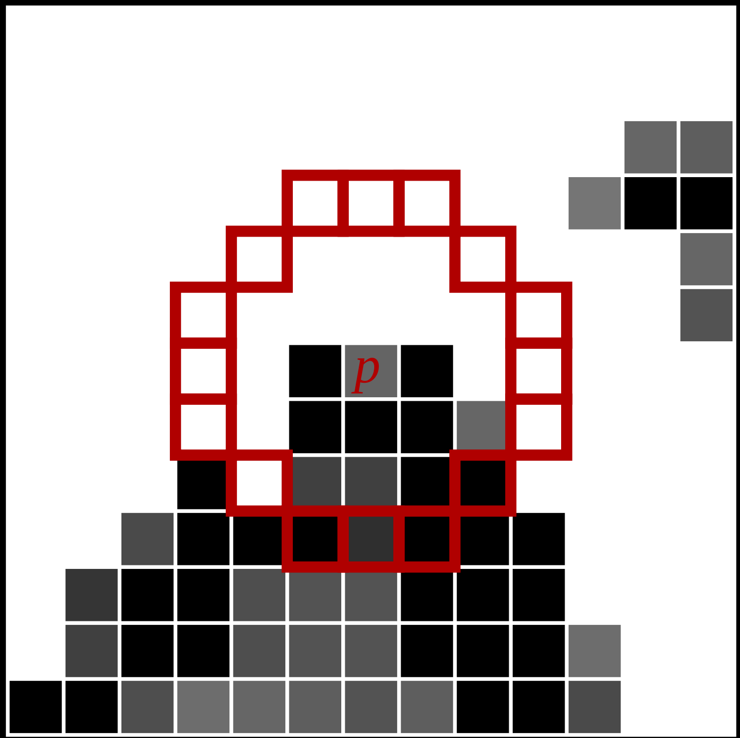


Analysis of pixels



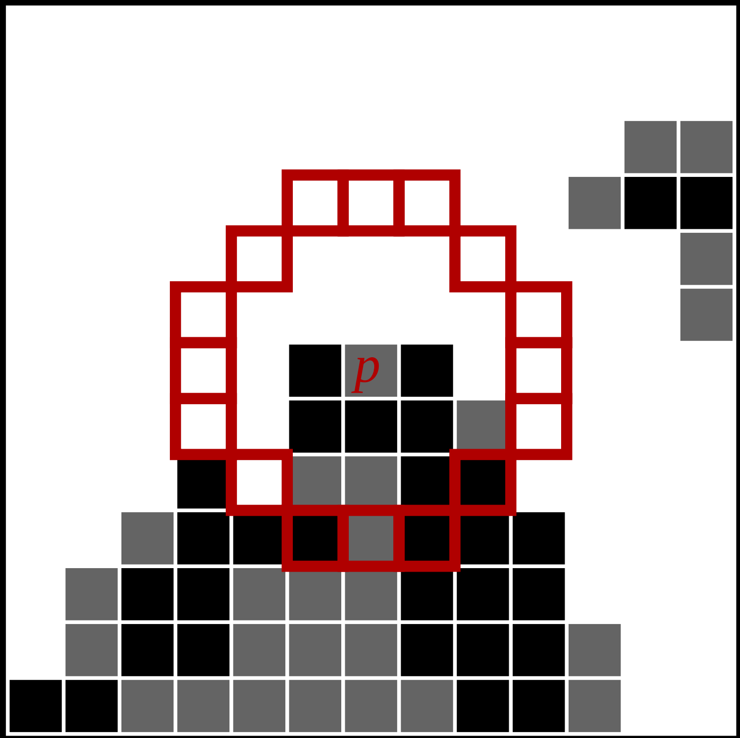
- Pixels are either:
    - Much brighter

Analysis of pixels



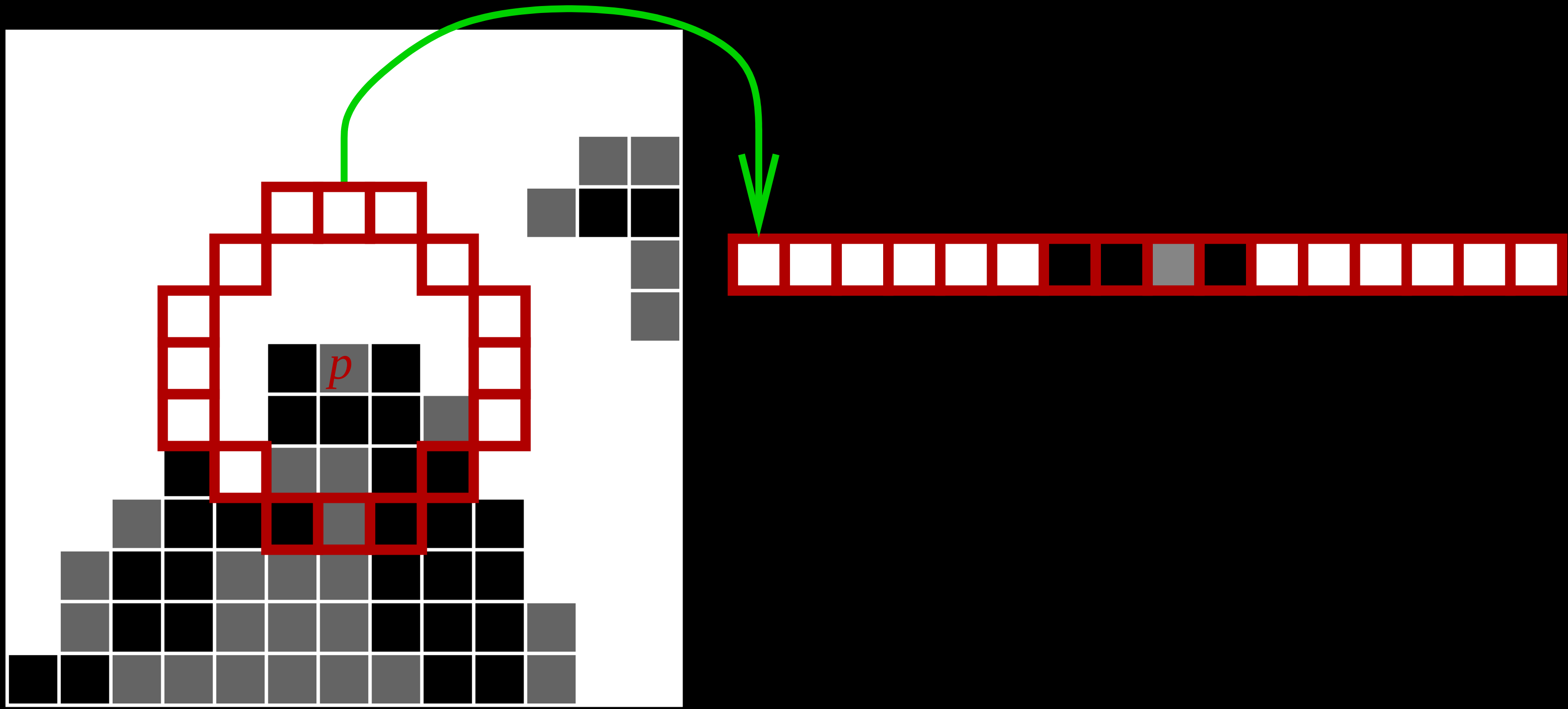
- Pixels are either:
  - Much brighter
  - Much darker

Analysis of pixels



- Pixels are either:
  - Much brighter
  - Much darker
  - Similar

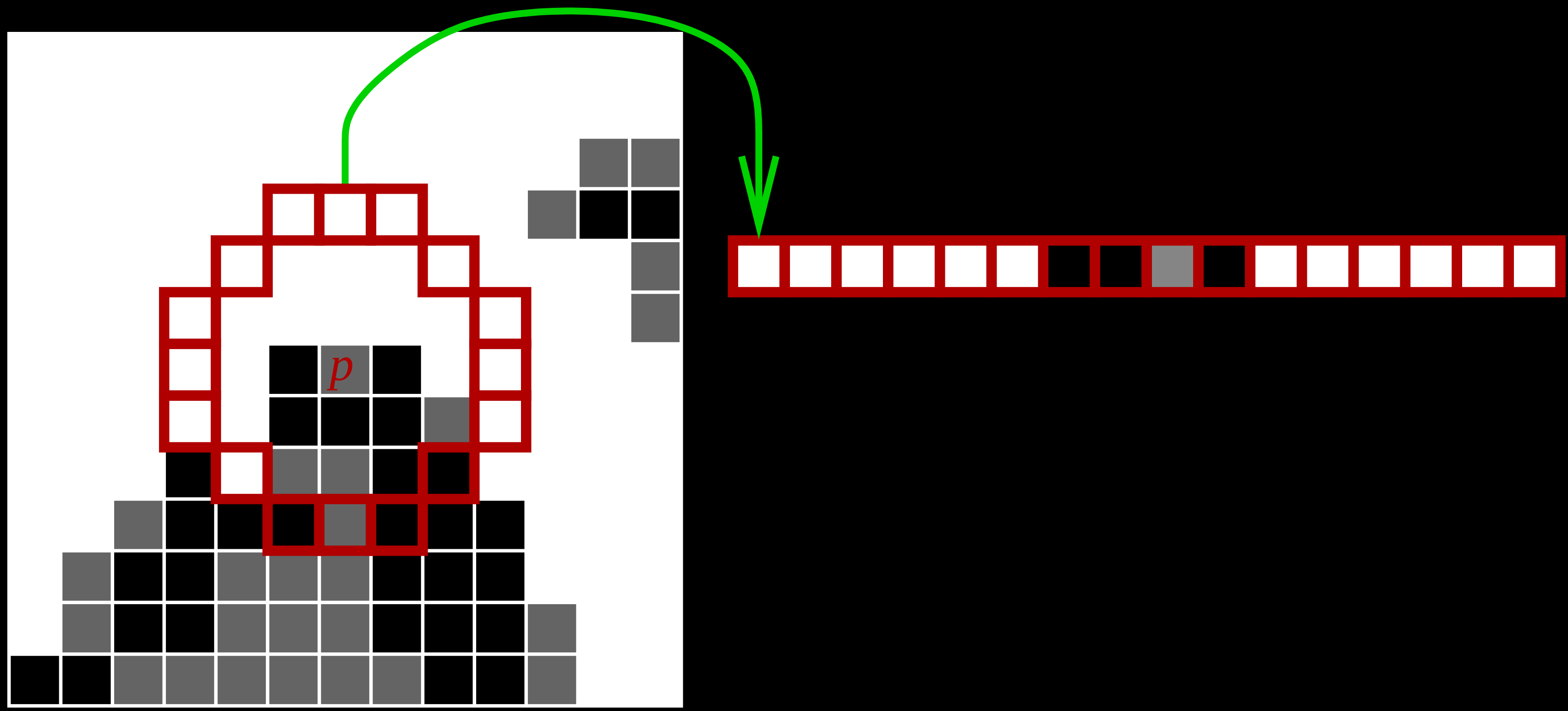
Analysis of pixels



- Pixels are either:
    - Much brighter
    - Much darker
    - Similar

- Ring represented as ternary vector

Analysis of pixels



- Pixels are either:
  - Much brighter
  - Much darker
  - Similar
- Ring represented as ternary vector
- Extract vectors for **ALL** pixels

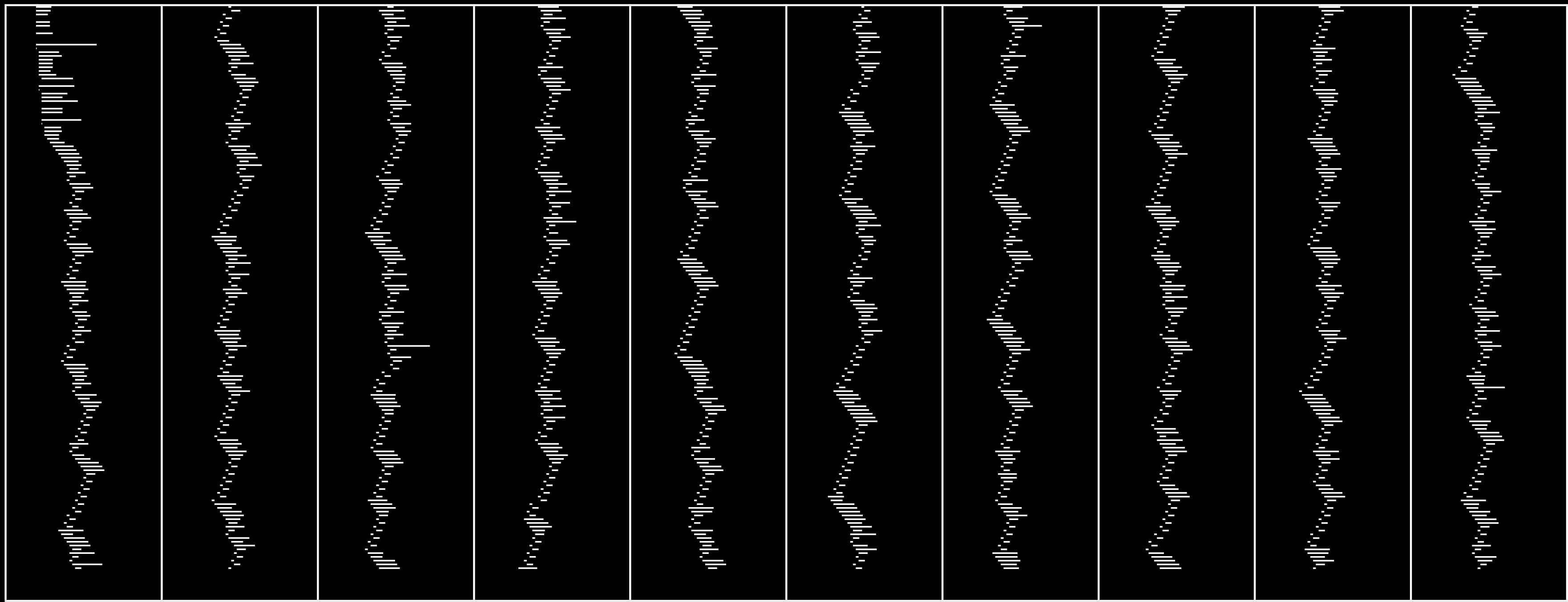
Ask ternary questions

- List of all potential features:
    - Ternary vector
    - Is it a feature?
- Question splits list in to 3 sublists
- Query each sublist
- Recurse until list contains all features or all non features

- Use questions on new feature

Output C++ code

A long string of nested if-else statements:



. . . which continues for 2 more pages.

Choosing questions

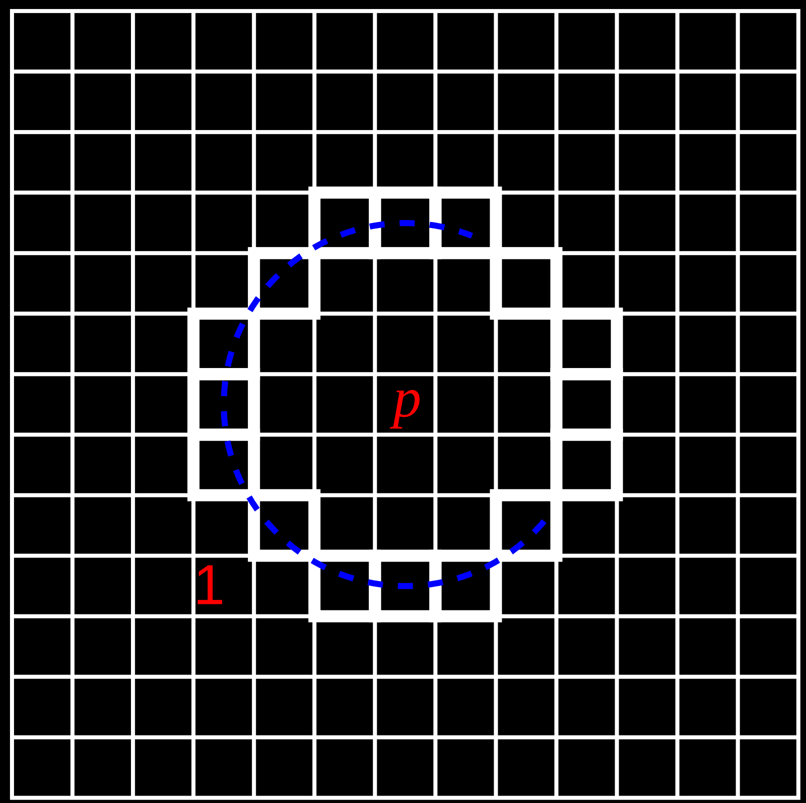
- Minimize average number of questions per feature
- Choose question to eliminate largest number of features

Or

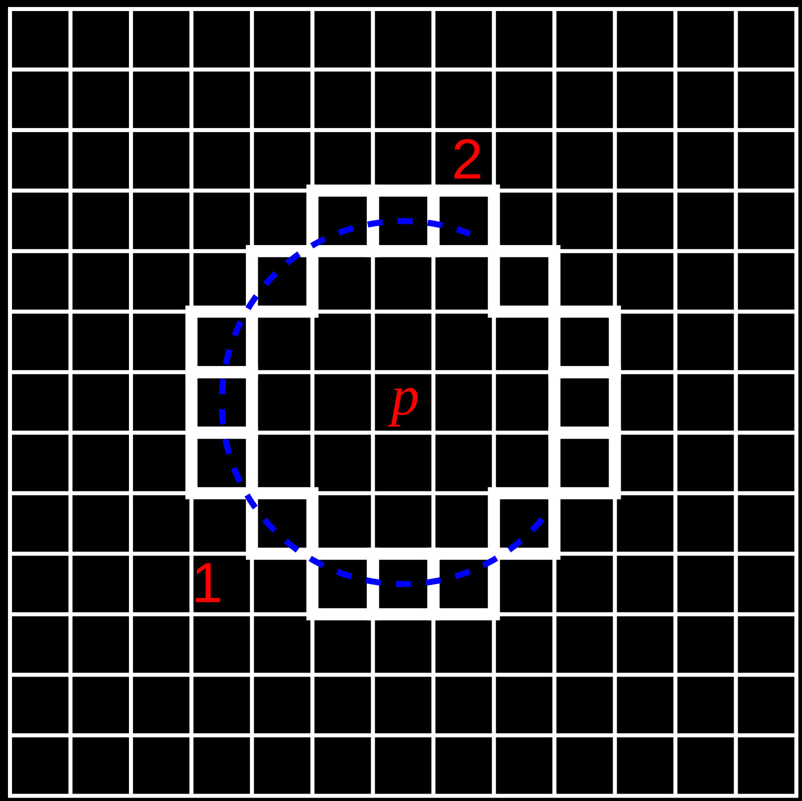
- Use entropy
    - Entropy of a list depends on distribution of features
    - Questions yield information
    - Total entropy of sublists is less
- Choose questions to maximize entropy gain
  (This is the ID3 algorithm)

Using entropy is better

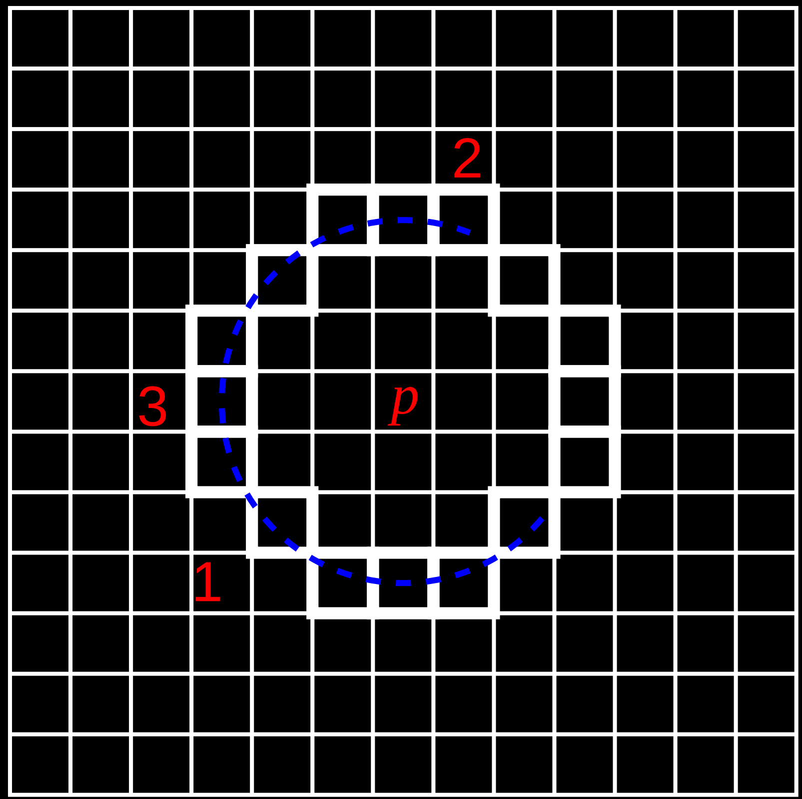
Example tree



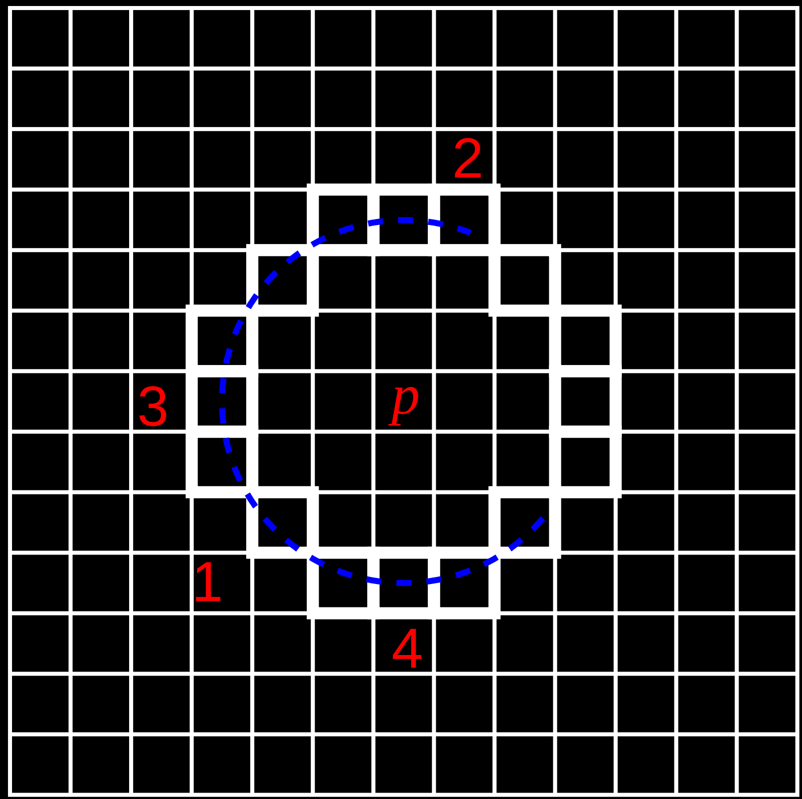
Example tree



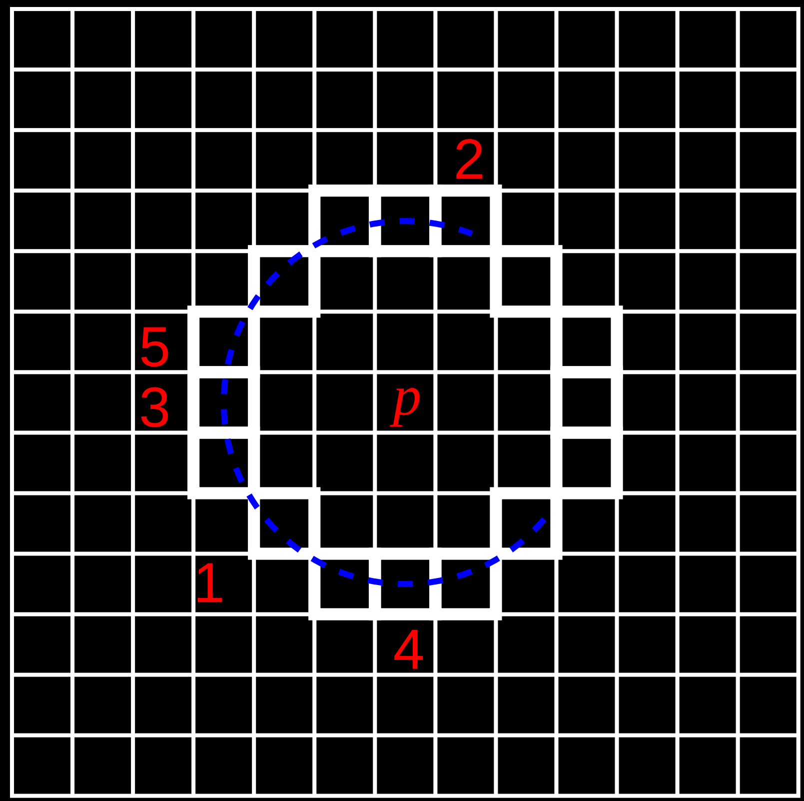
Example tree



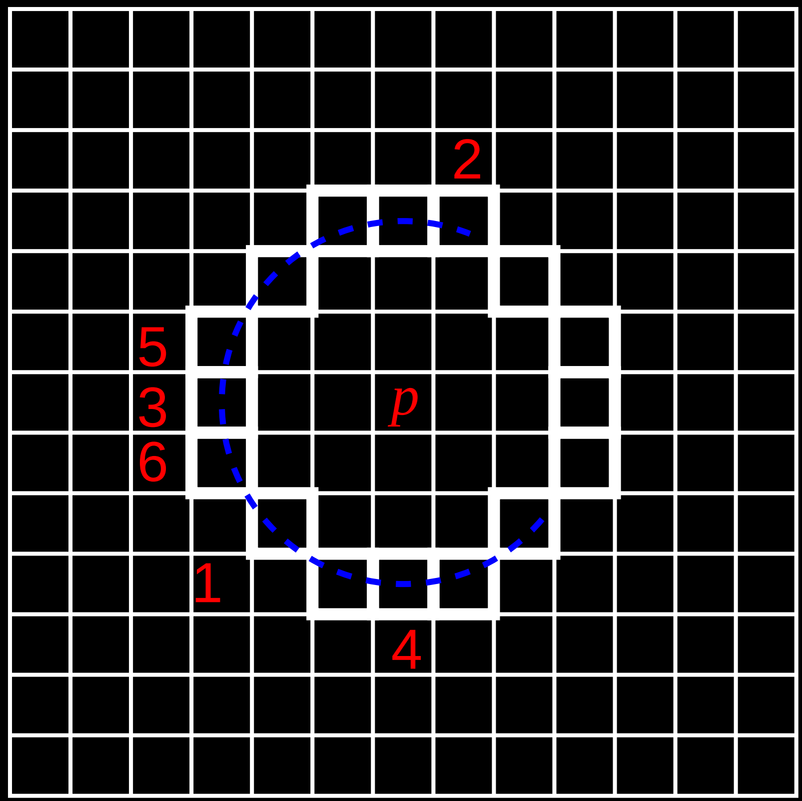
Example tree



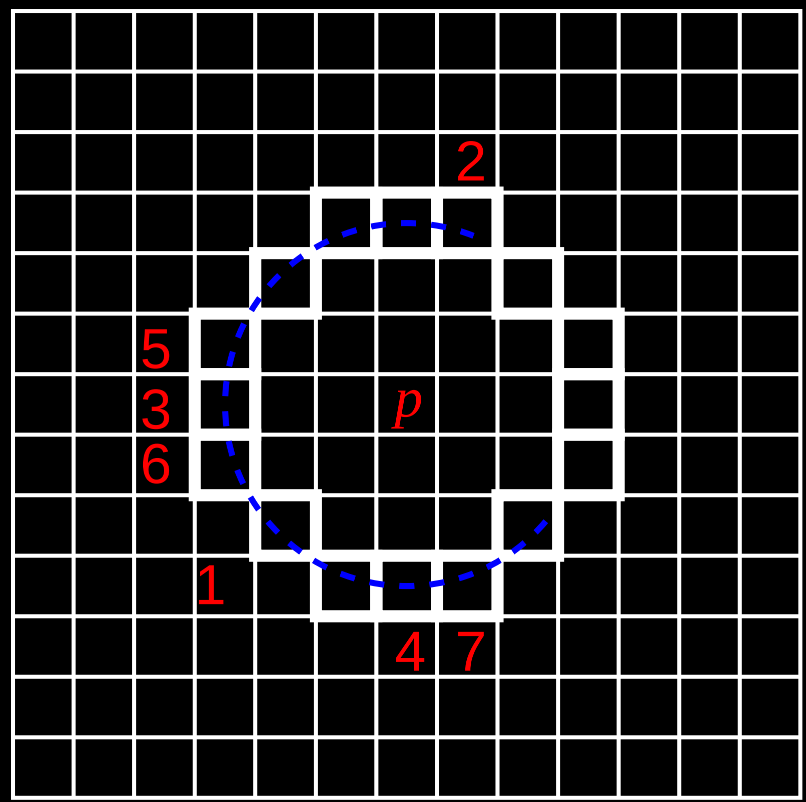
Example tree



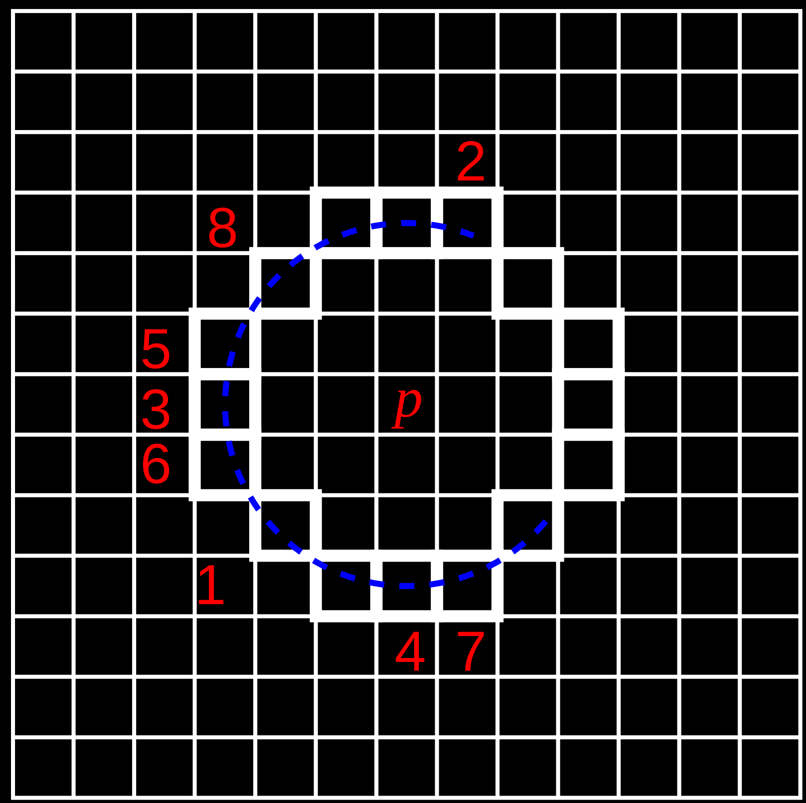
Example tree



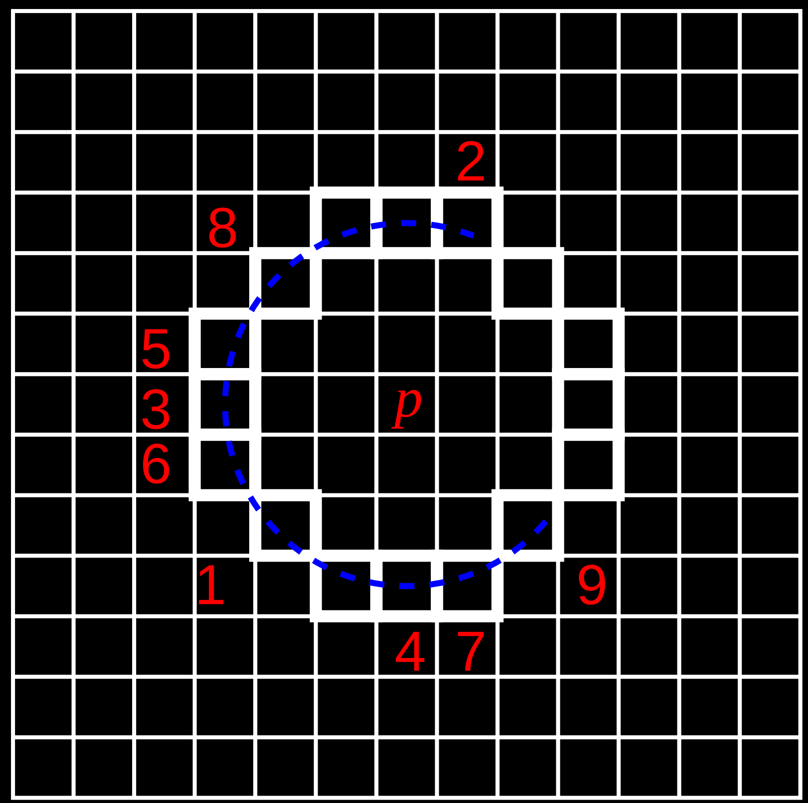
Example tree



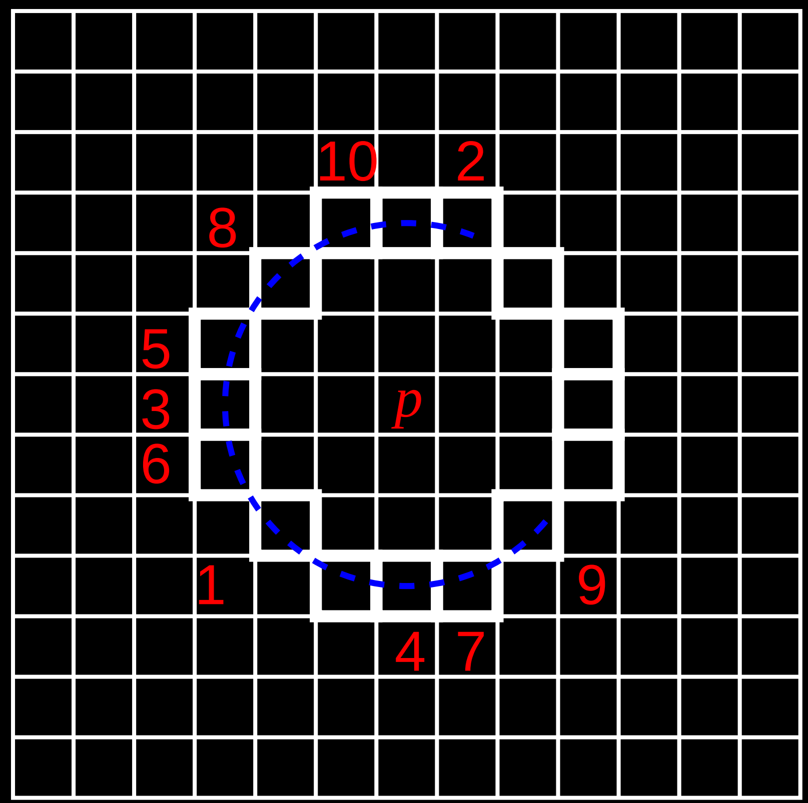
Example tree



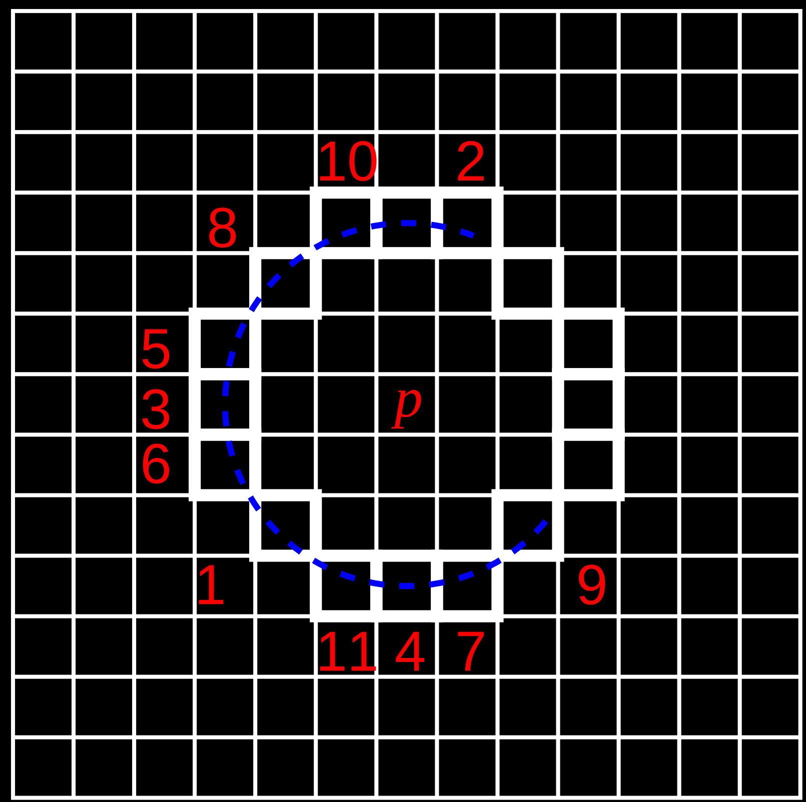
Example tree



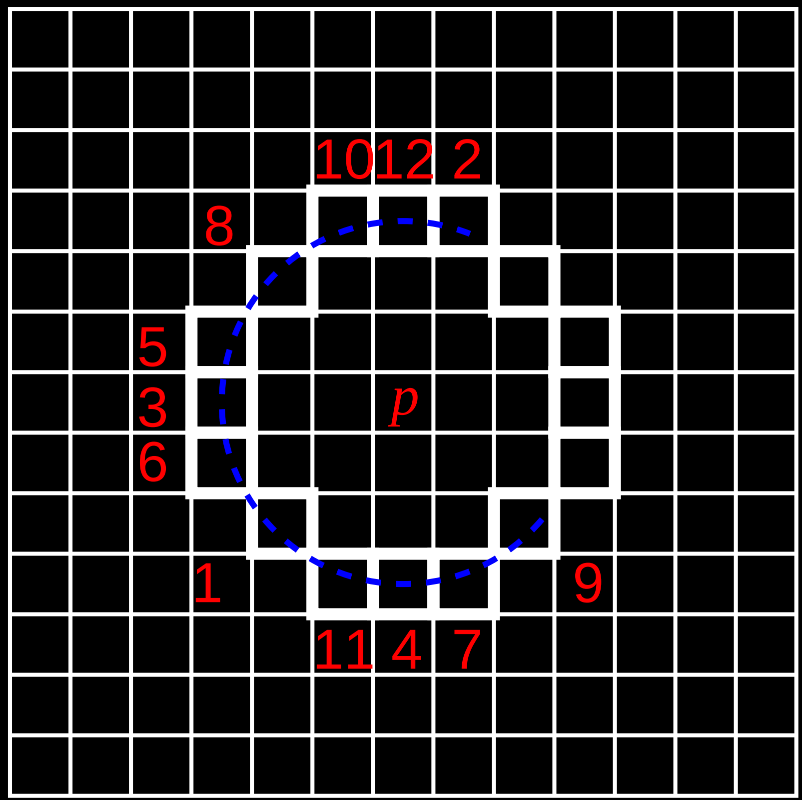
Example tree



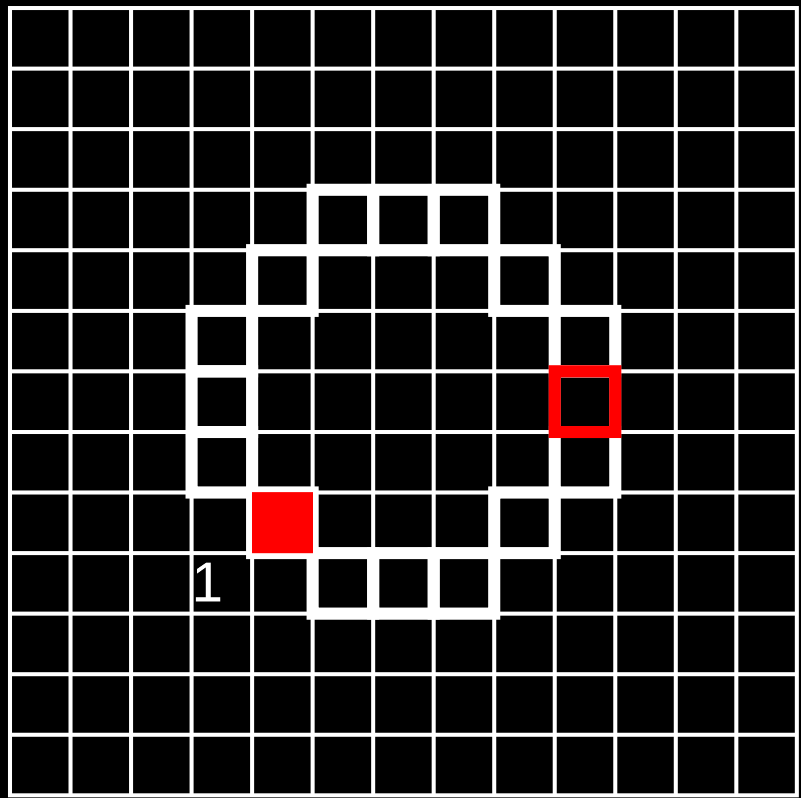
Example tree



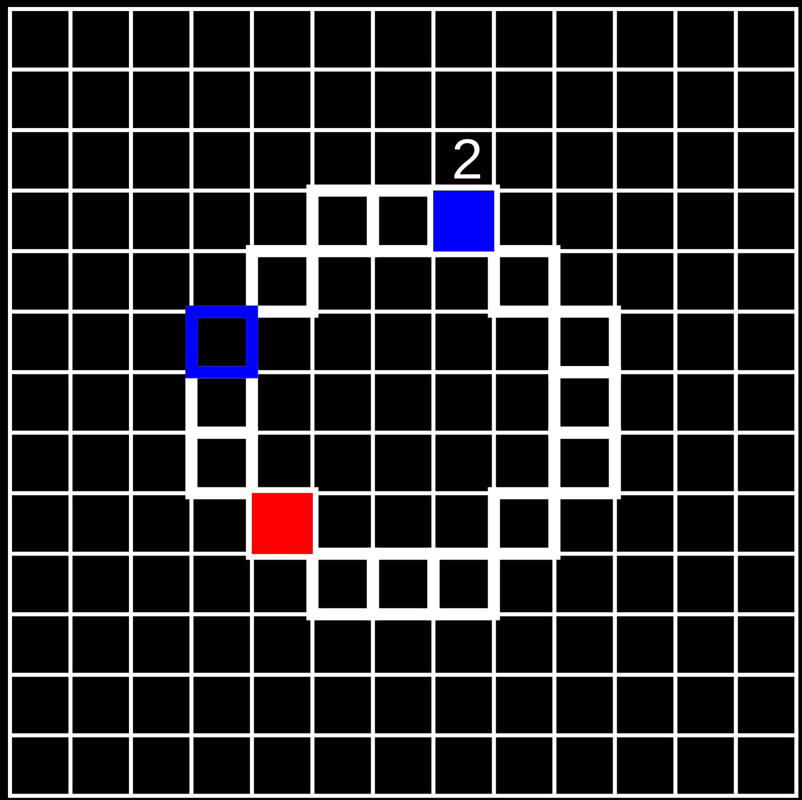
Example tree



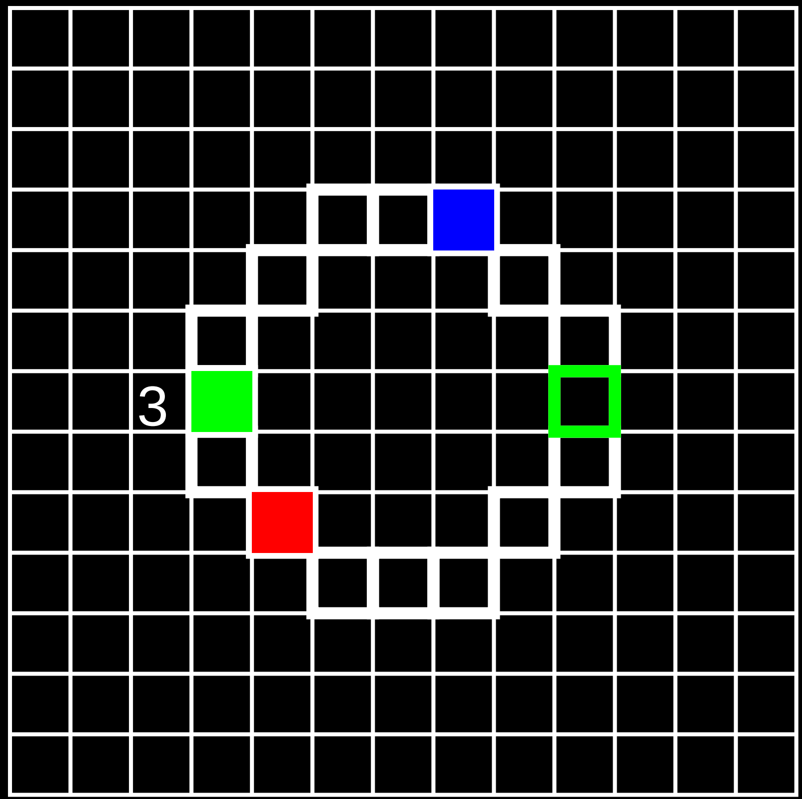
Example tree



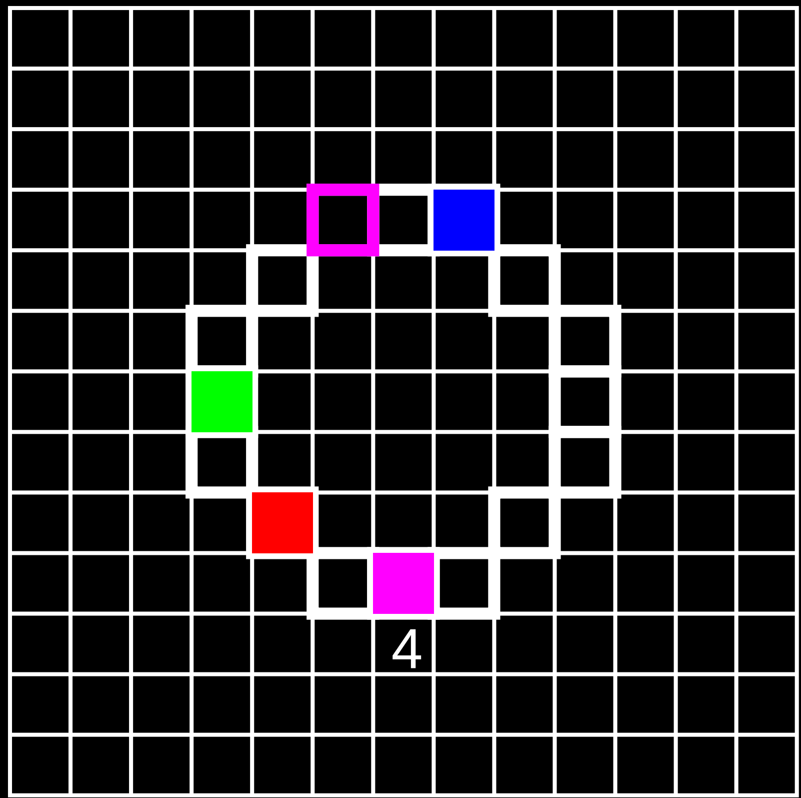
Example tree



Example tree



Example tree



How FAST?

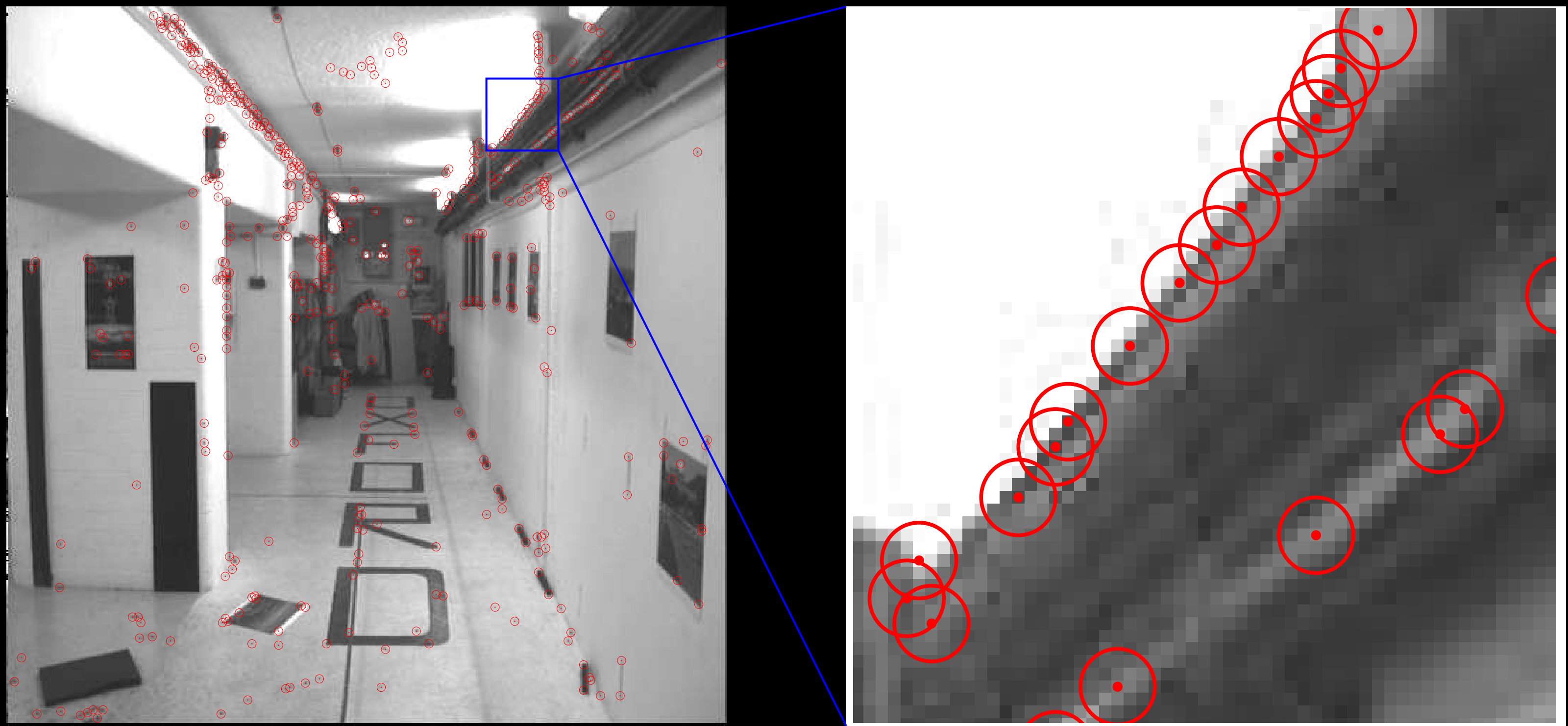
Percentage of available CPU time (typical video)

Detector	2.6 GHz (%)	850 MHz (%)
New FAST	5.4	21.7
FAST	7.45	48.5
DoG	301	1280
SUSAN	37.9	137.5
Harris	120	830

- New FAST: 2.2 questions per feature

Is it any good...?

An example failure mode:



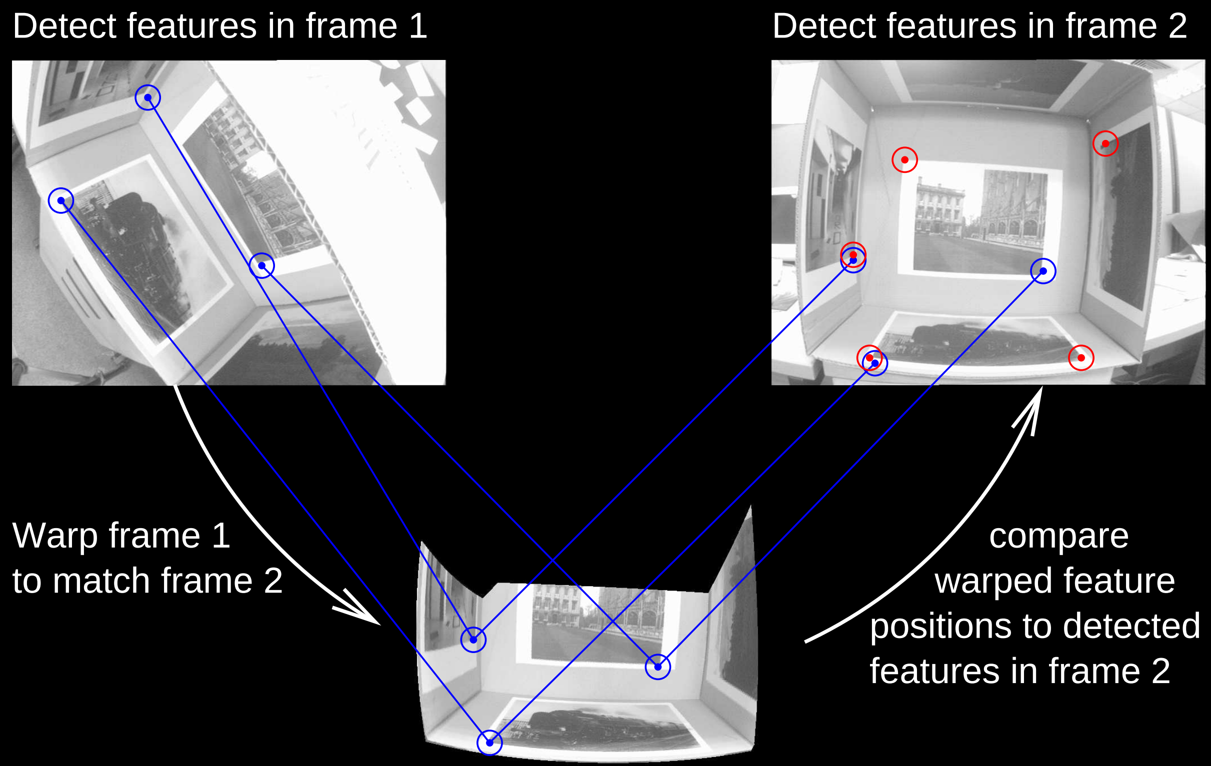
- Ring misses thin quantized lines
- ‘Obvious’ corners missed

Compare against others

- Harris
- Shi/Kanade and Tomasi
- SUSAN
- Multiscale DoG (used by SIFT)
- Harris-Laplace

Comparison methodology

Is the same real-world 3D point detected from multiple views?

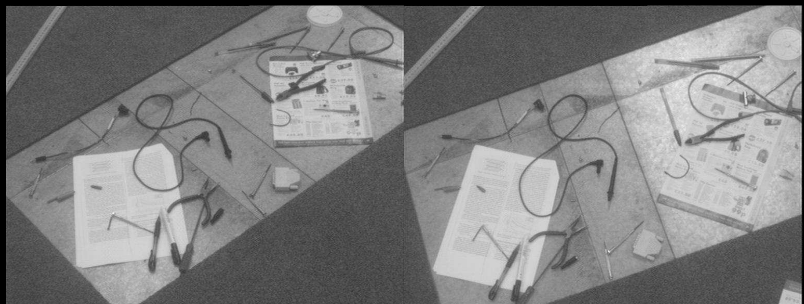


Repeat for all pairs in a sequence

Data sets

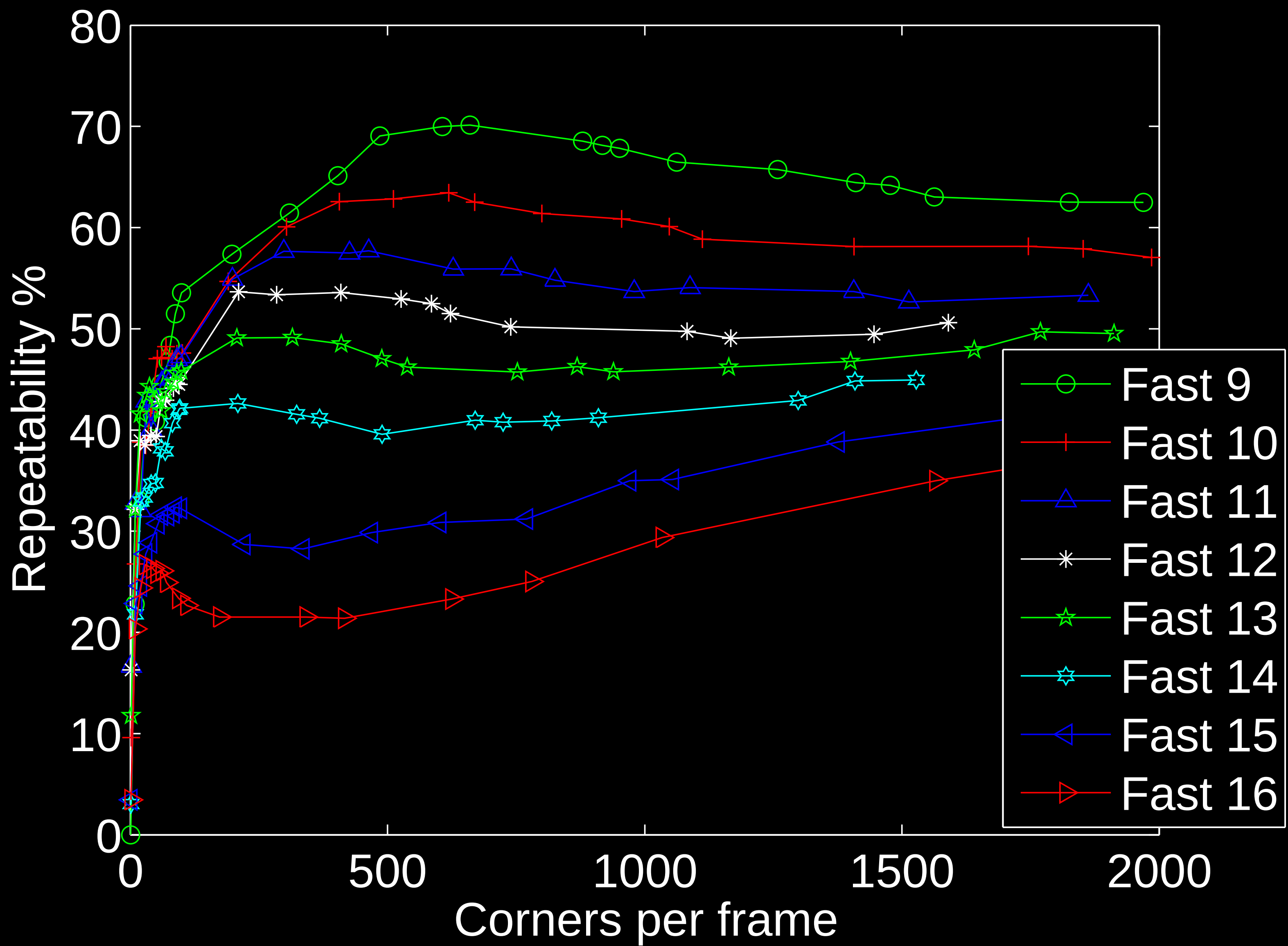
Affine (14 images)

Geometric (15 images)

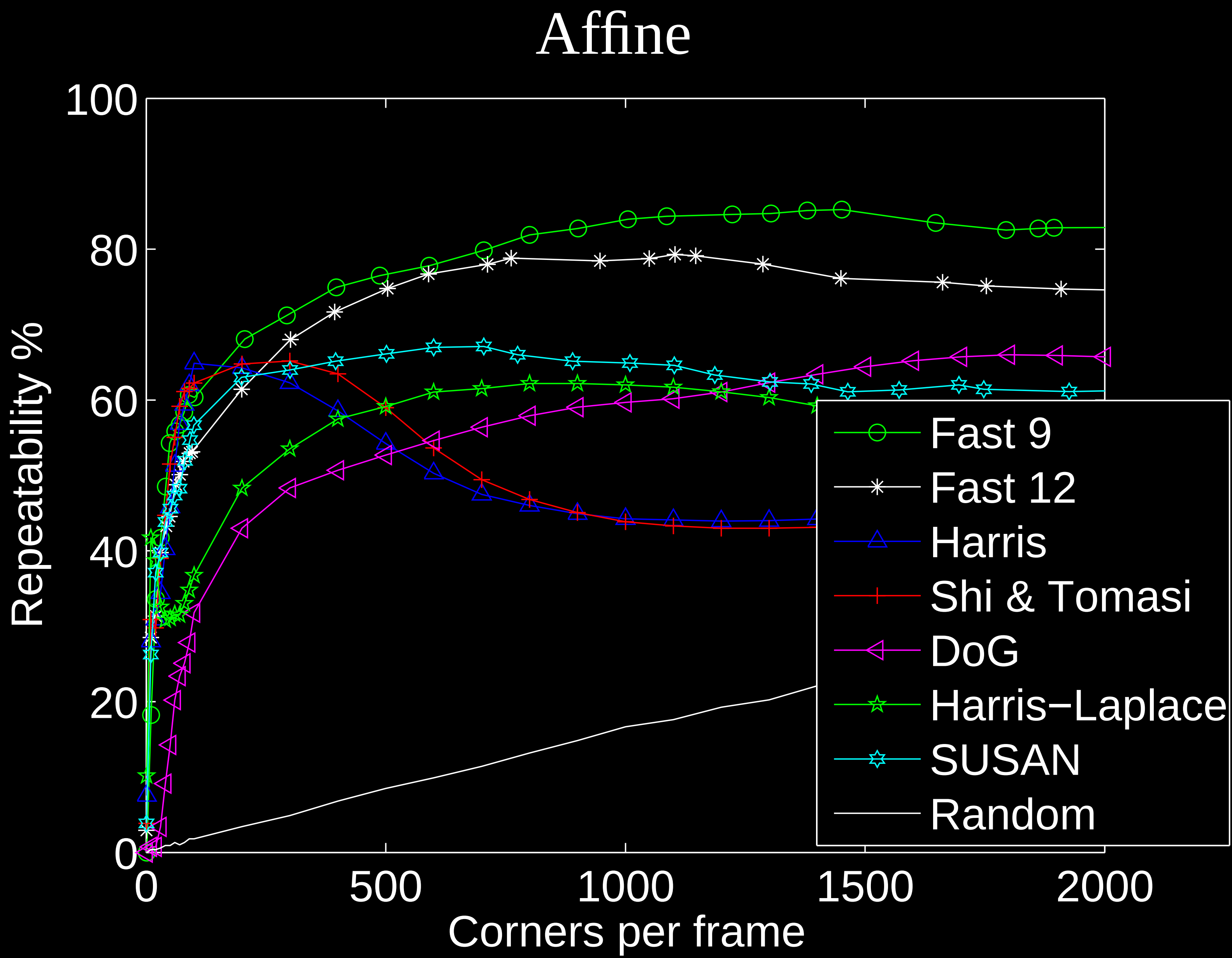


Bas-relief (8 images)

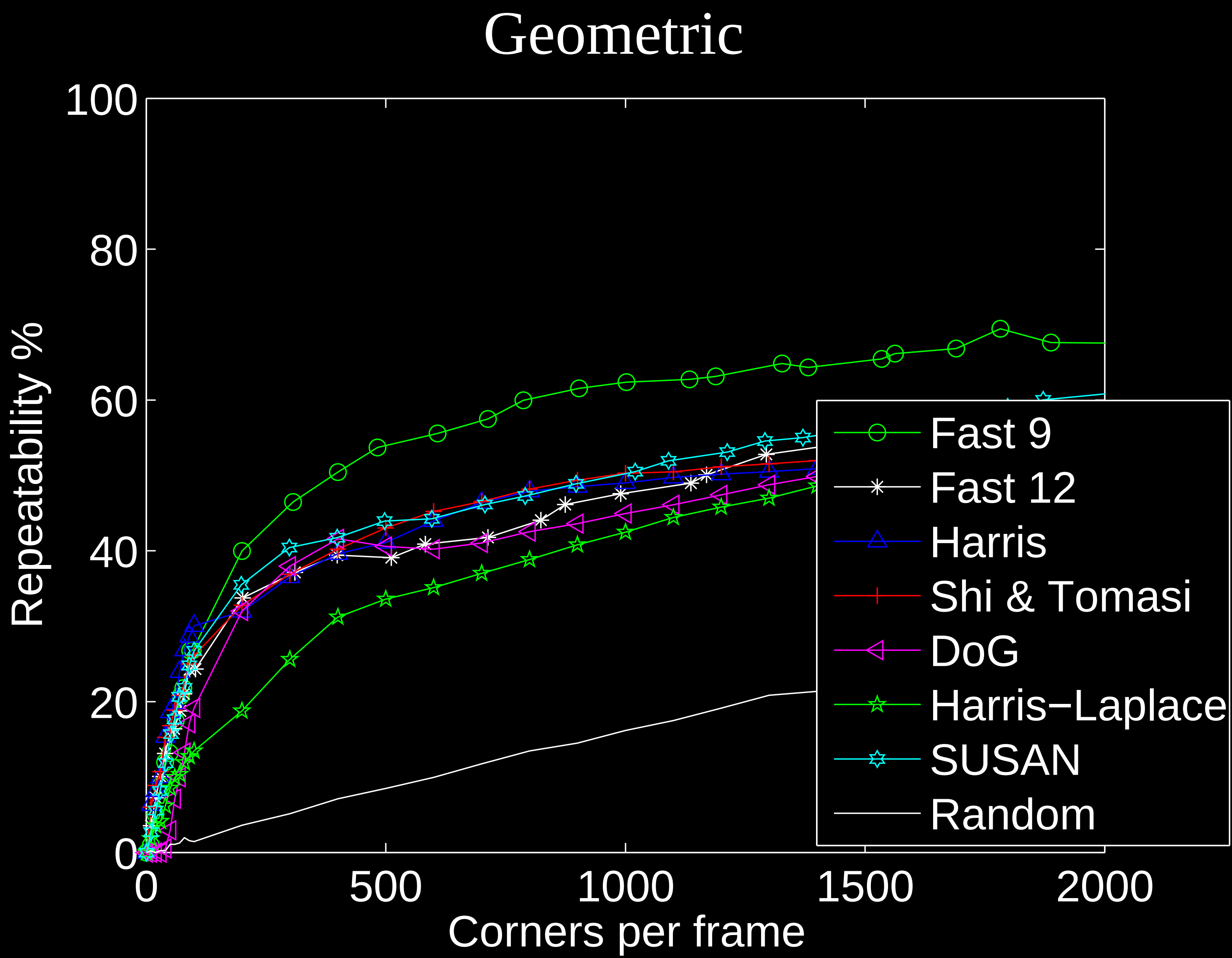
Which FAST is the best?



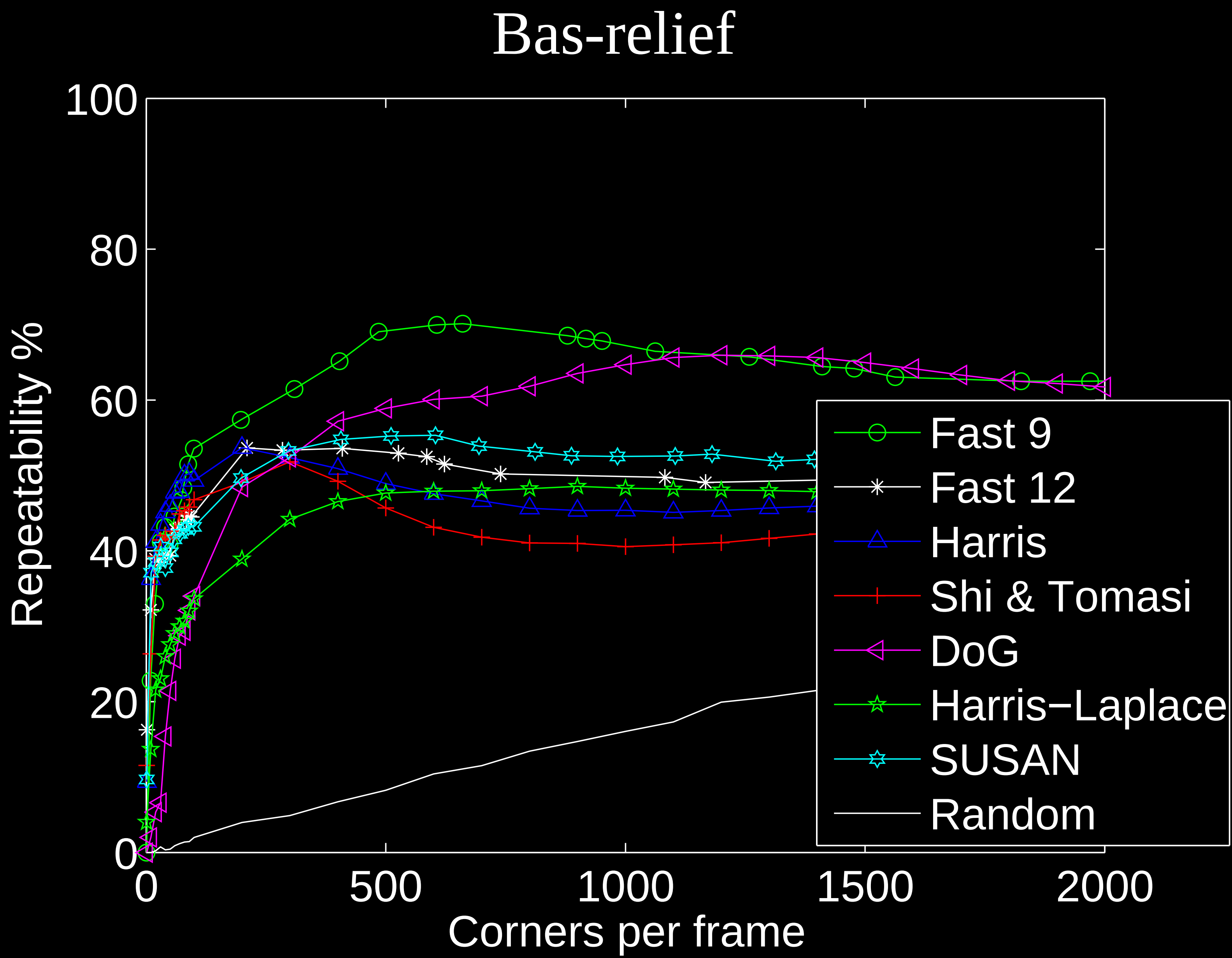
How good is FAST?



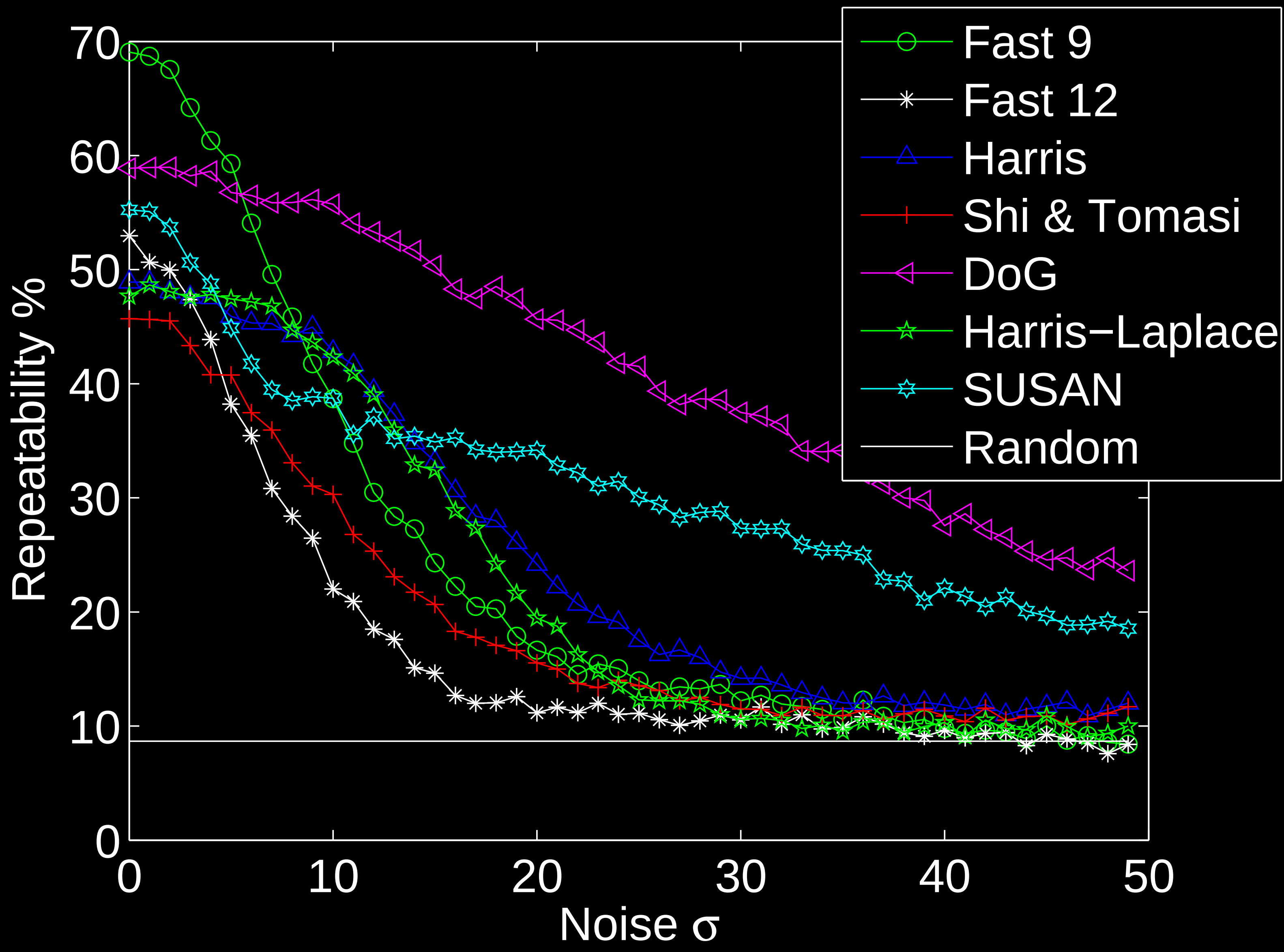
How good is FAST?



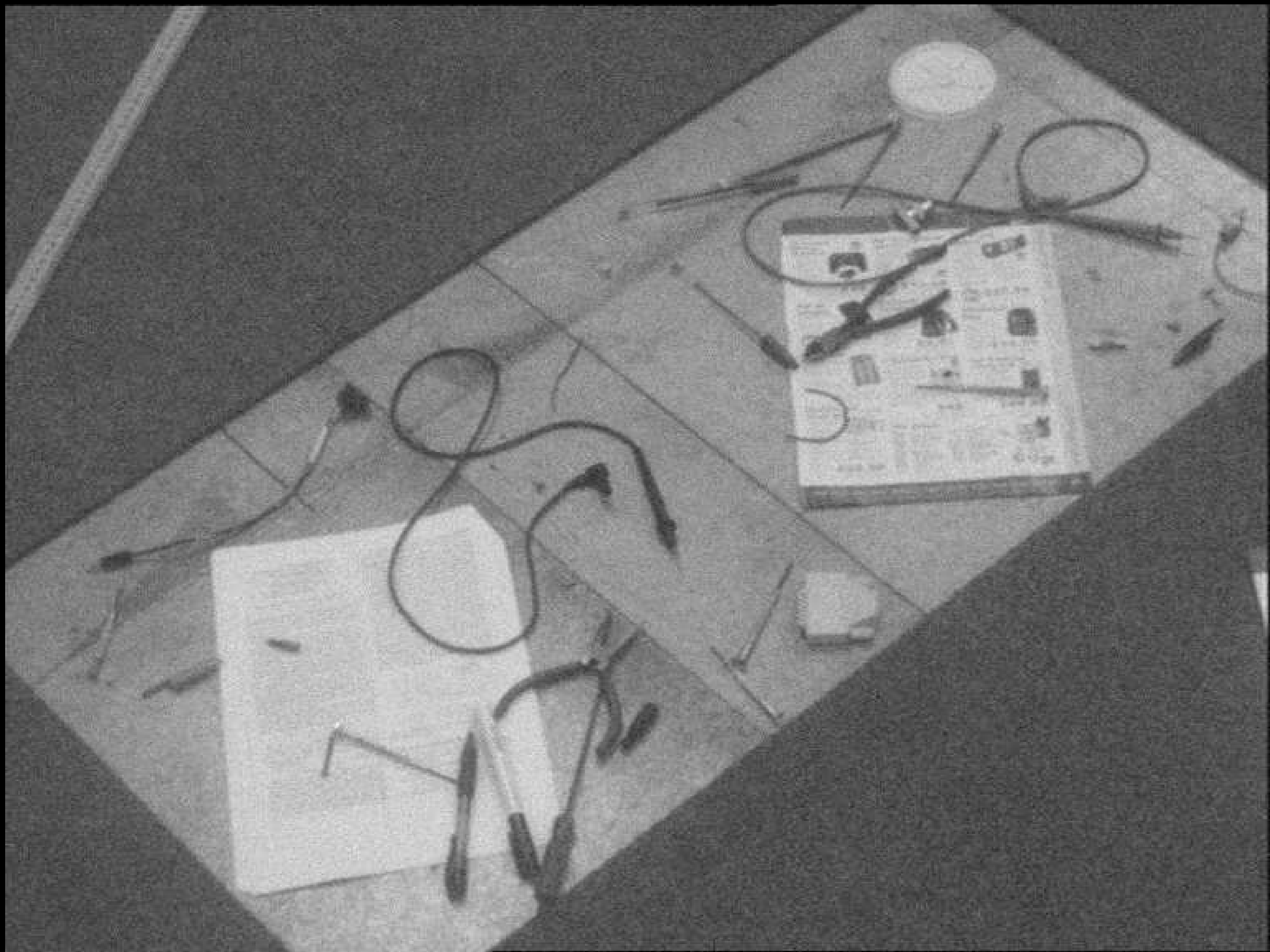
How good is FAST?



Noise performance



Noise performance



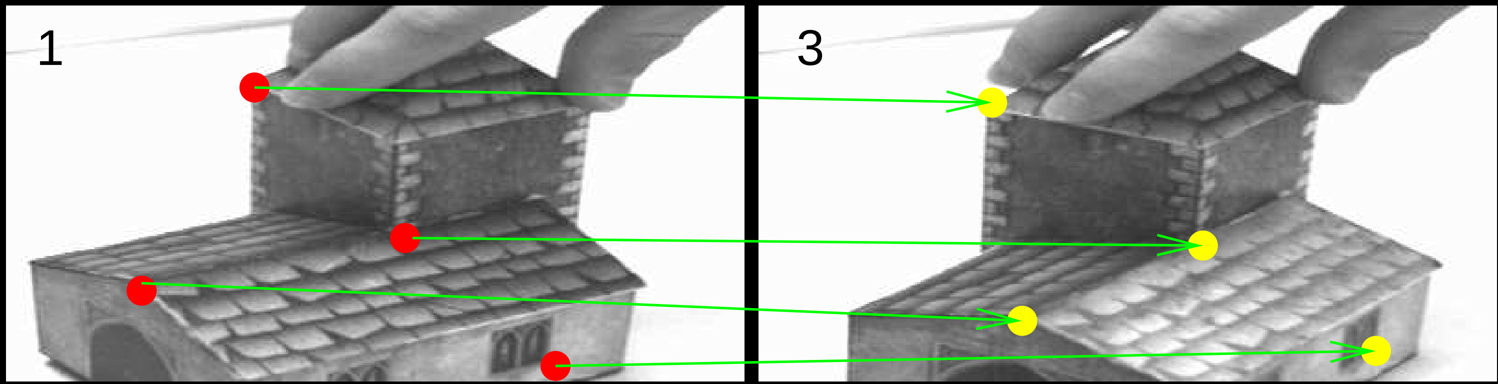
Conclusions on FAST

- Very fast
  - 190 MPixels/s (1.48 Gi b/s)!
  - Used machine learning to learn for speed
- Produces high quality features
  - Results from real features from representative images

Back to tracking

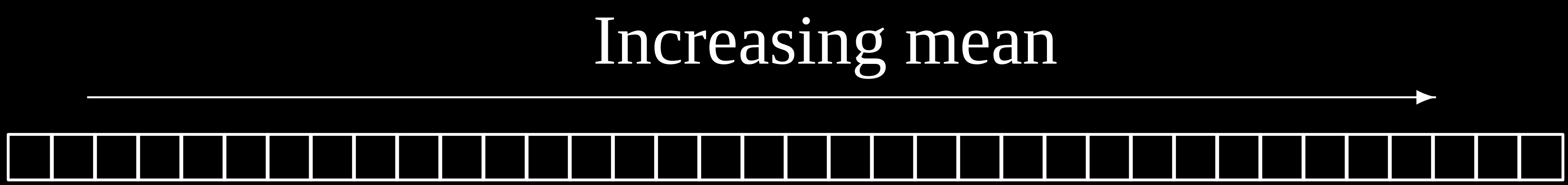
Robust differential measurements

Detect and match features in next frame



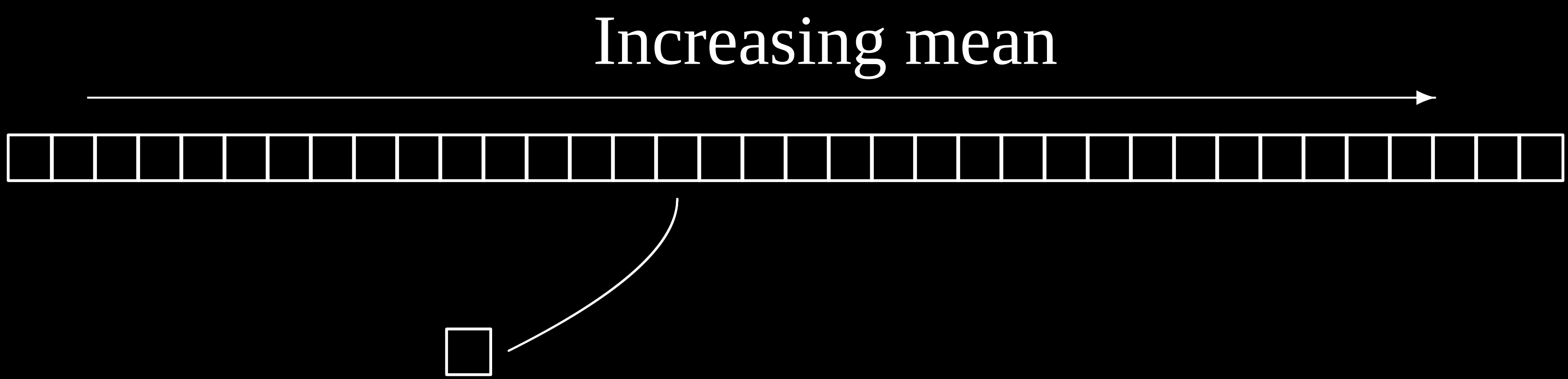
- Full frame matching makes it robust to large motions
- Detecting features in a whole frame is slow
- Matching can be $O(n^2)$

Efficient feature matching



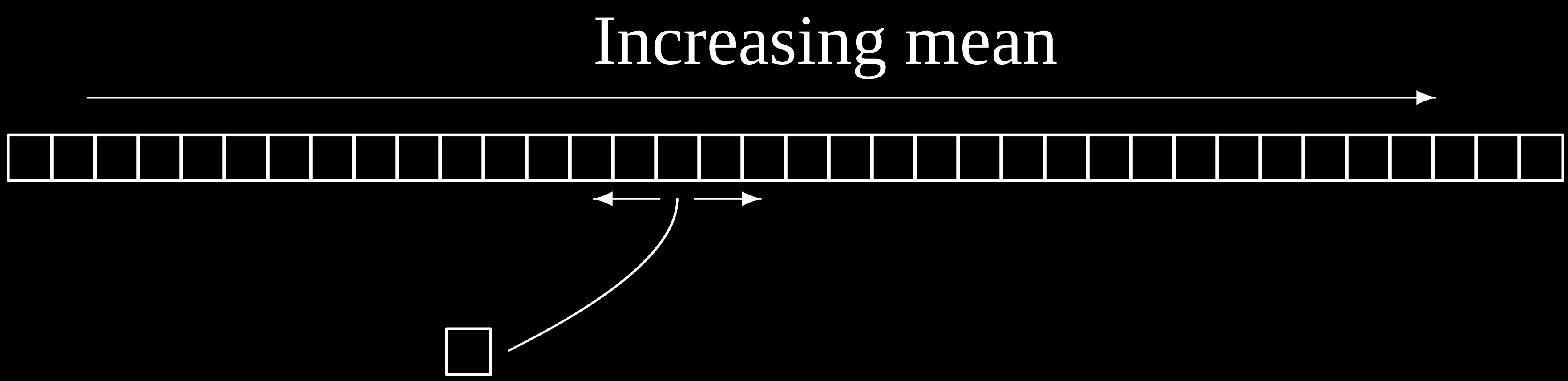
- Sort features by mean value of feature vectors

Efficient feature matching



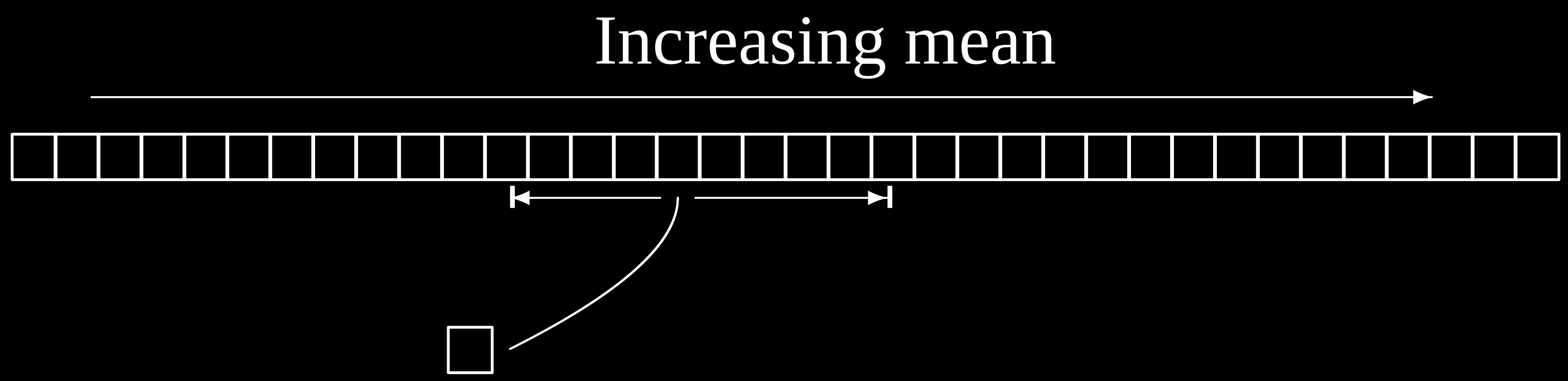
- Sort features by mean value of feature vectors
- Find closest mean by binary search

Efficient feature matching



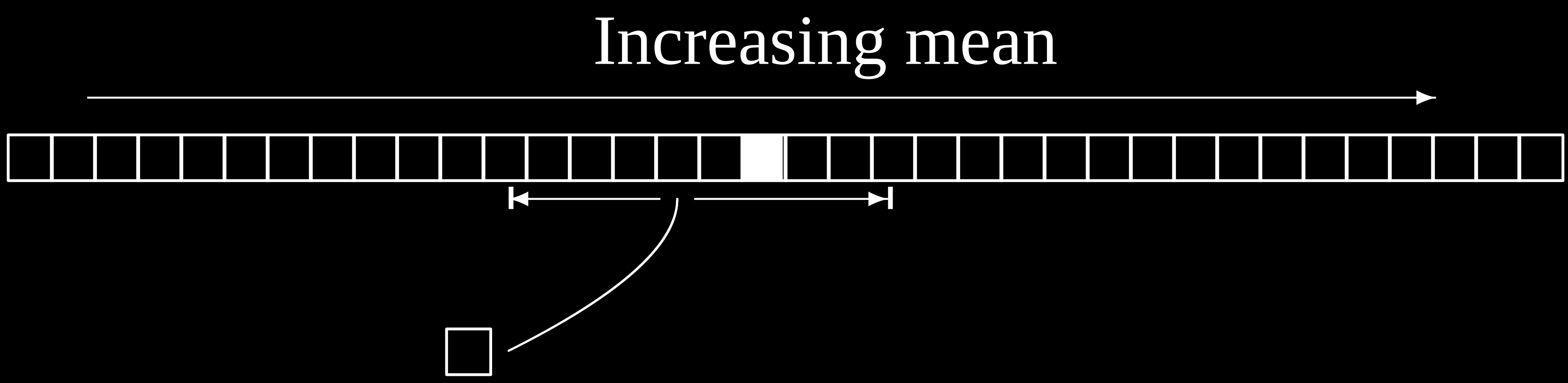
- Sort features by mean value of feature vectors
- Find closest mean by binary search
- Search outwards

Efficient feature matching



- Sort features by mean value of feature vectors
- Find closest mean by binary search
- Search outwards
- SSD between means bounds search

Efficient feature matching



- Sort features by mean value of feature vectors
- Find closest mean by binary search
- Search outwards
- SSD between means bounds search
- Best match has lowest SSD

Conclusions on point tracking

Statistical properties:

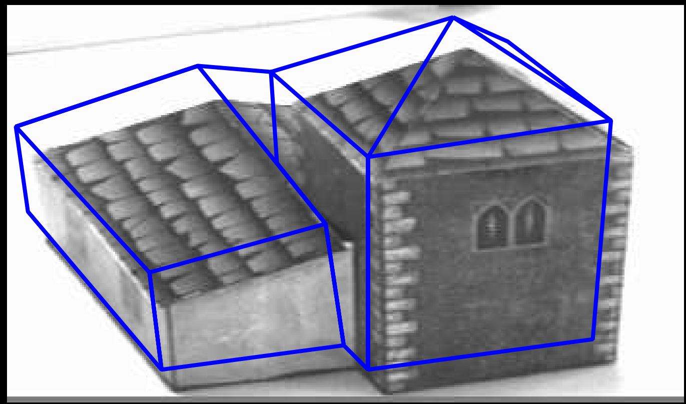
- Point based tracking
  - Requires
    - 3D point cloud
  - Provides
    - Robust *differential* measurements...
    - ...with approximately Gaussian posterior

Conclusion:

- Useful, but incomplete.

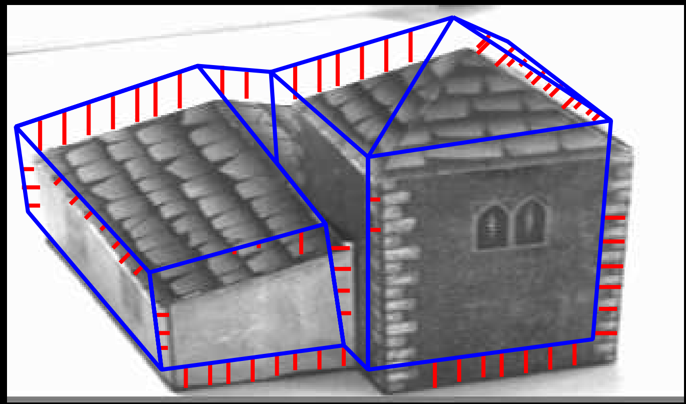
Edge based tracking

Edge based tracking



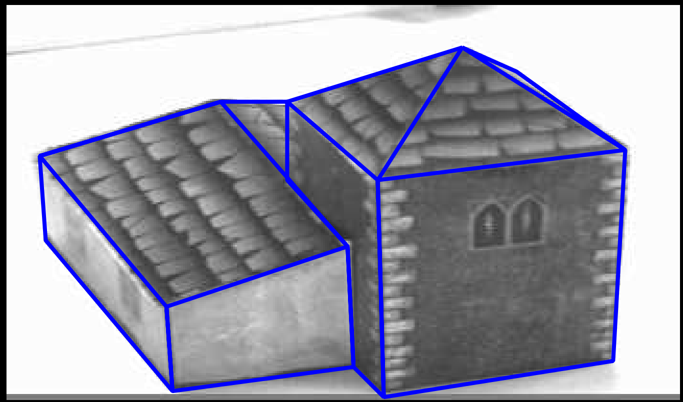
- Start from position prior

Edge based tracking



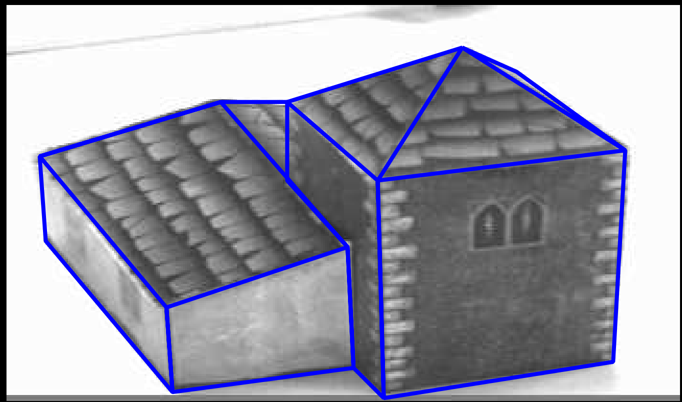
- Start from position prior
- Search along edge-normal lines

Edge based tracking



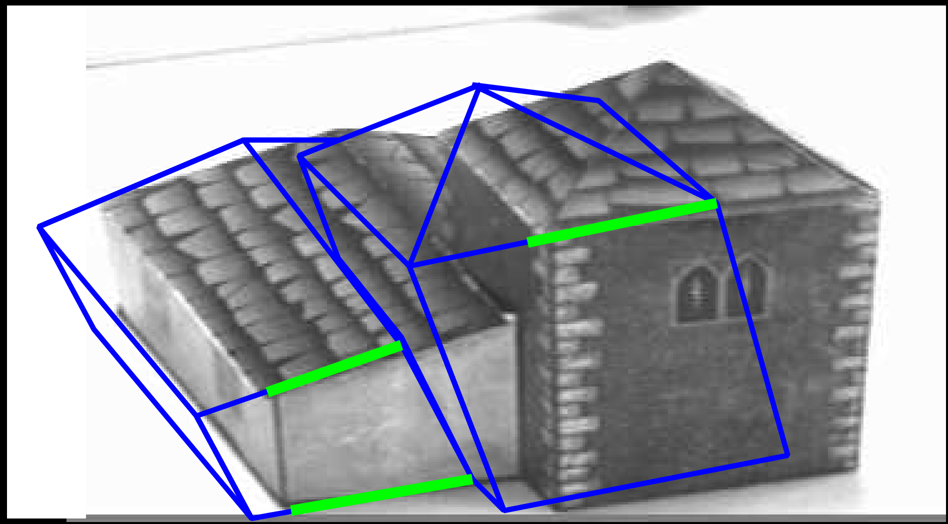
- Start from position prior
- Search along edge-normal lines
- Adjust position to minimize errors

Edge based tracking



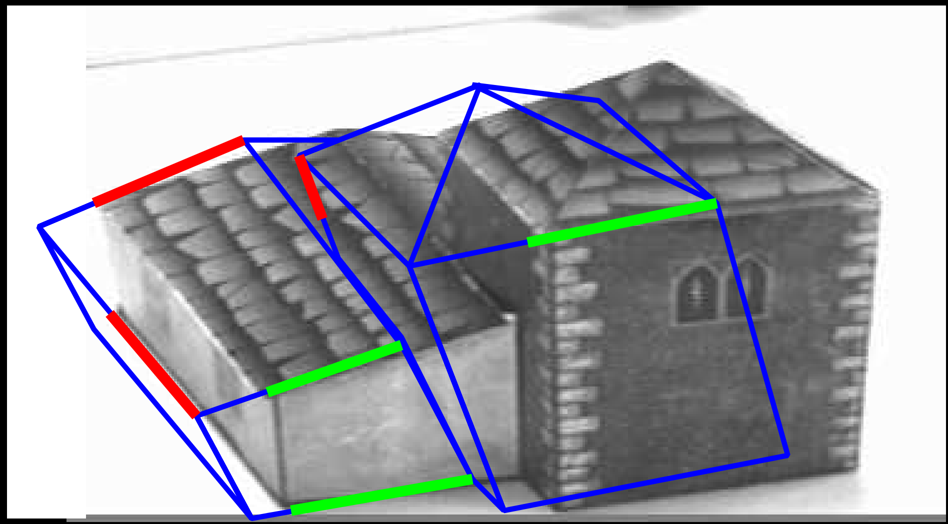
- Start from position prior
- Search along edge-normal lines
- Adjust position to minimize errors
- Gives drift free measurements
    - Model is static

Good prior needed



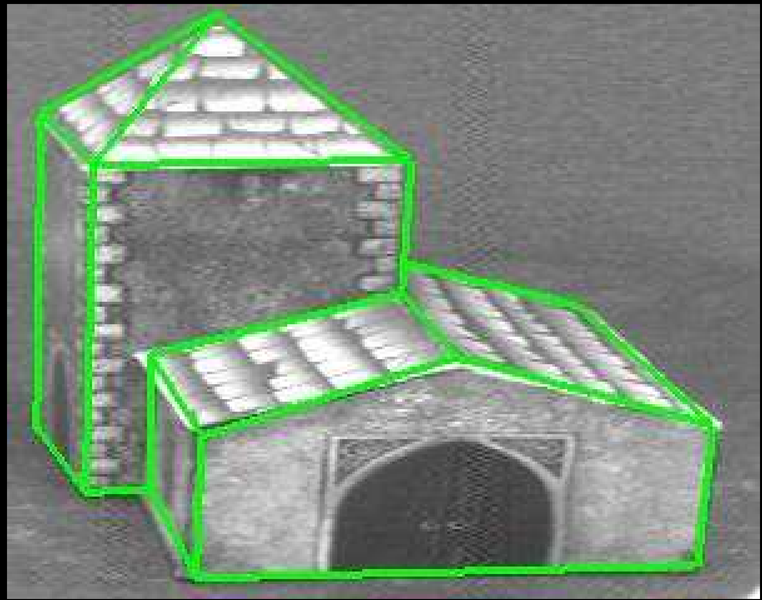
- Edges are a step change in intensity
- Correspondence is hard—pick closest edge

Good prior needed



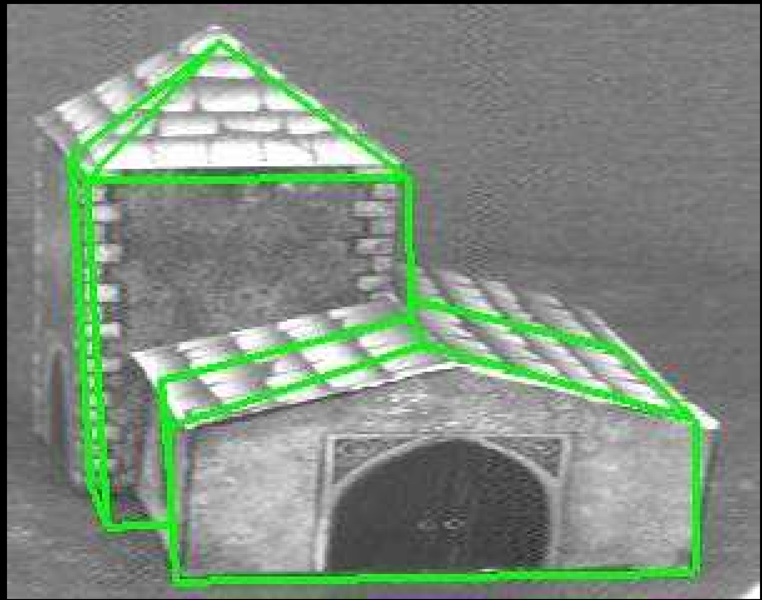
- Edges are a step change in intensity
- Correspondence is hard—pick closest edge
- Prior must be good, or the wrong edge will be found
  - Correct edge might be nowhere near

Non Gaussian posterior



- Correct correspondences
  - Tracking is accurate
- Incorrect correspondences
  - Tracking is inaccurate—even if prior is good

Non Gaussian posterior



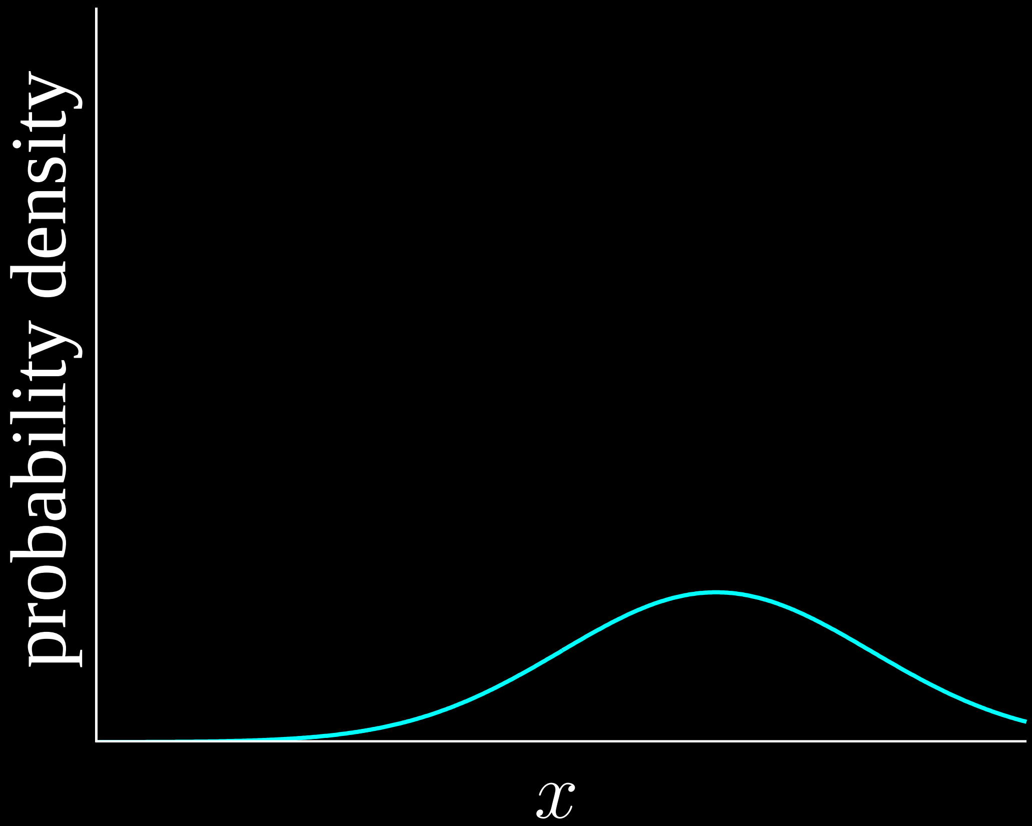
- Correct correspondences
  - Tracking is accurate
- Incorrect correspondences
  - Tracking is inaccurate—even if prior is good

Summary

- Edge based tracking
  - Requires
    - ⋆ 3D geometric model
    - ⋆ Good pose prior
  - Provides
    - ⋆ Drift free measurements
    - ⋆ Non Gaussian posterior
- Point based tracking
  - Requires
    - ⋆ 3D point cloud
  - Provides
    - ⋆ Robust differential measurements...
    - ⋆ ...with approximately Gaussian posterior

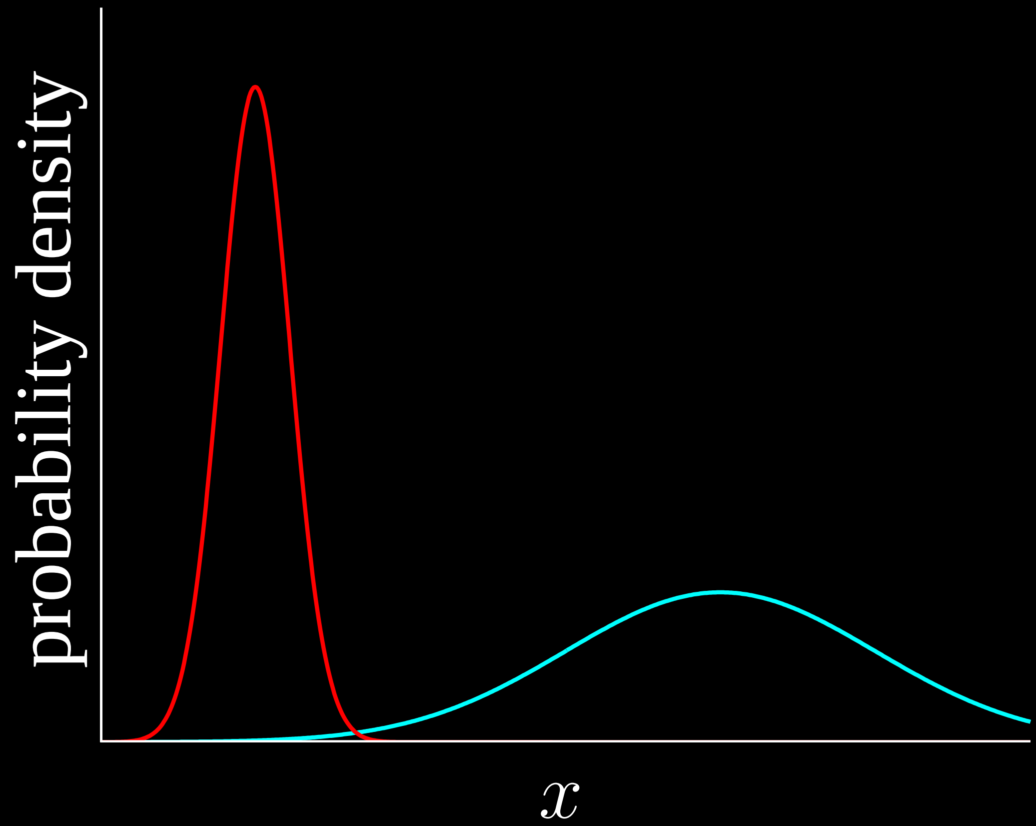
Sensor fusion

Combining measurements



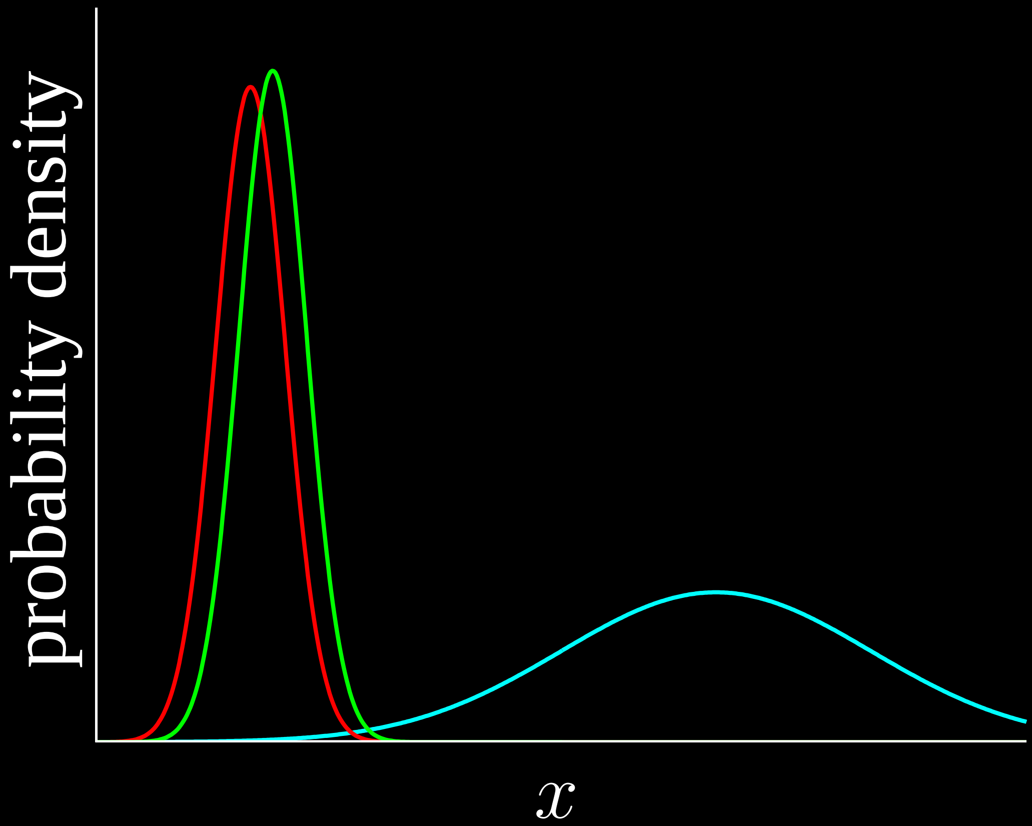
prior

Combining measurements



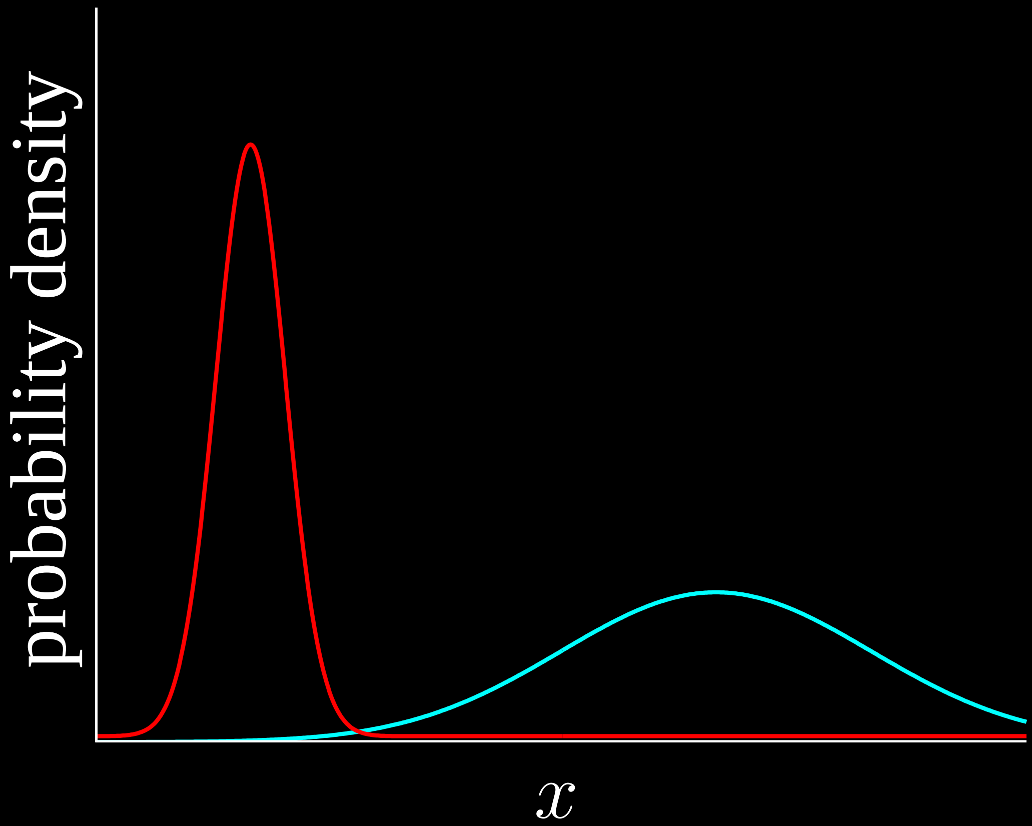
prior $\times$ liklihood

Combining measurements



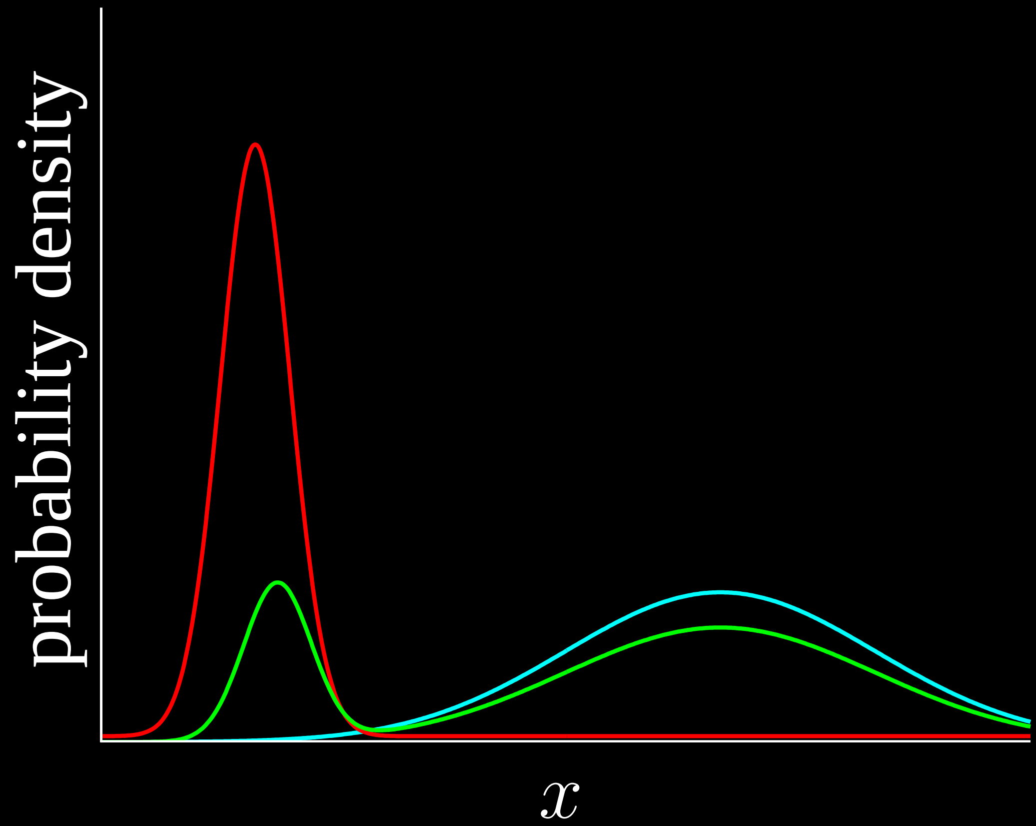
prior $\times$ liklihood $=$ posterior

Combining measurements



prior $\times$ liklihood (with outliers)

Combining measurements

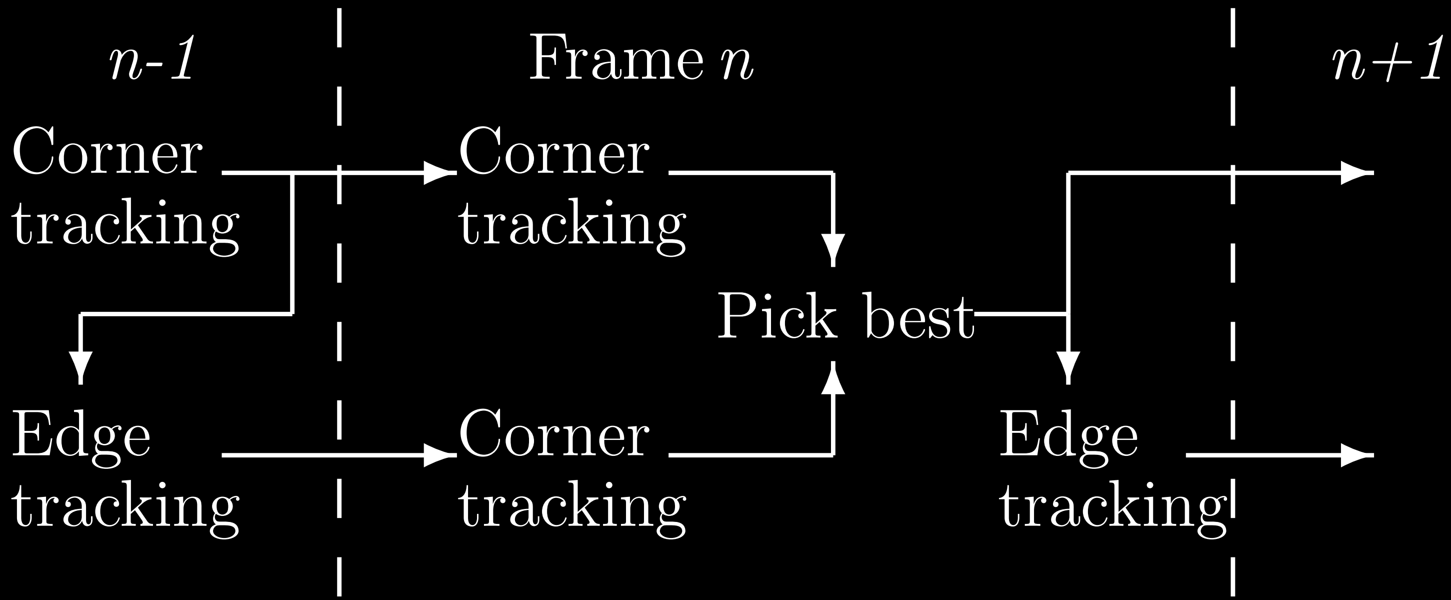


prior $\times$ liklihood (with outliers) $=$ multimodal posterior

Multimodal posterior propagation

- Either tracker can be wrong
  - Edge tracker can get correspondence wrong
  - Point based tracker can drift
- Posterior can be multimodal
  - Simple solutions do not work

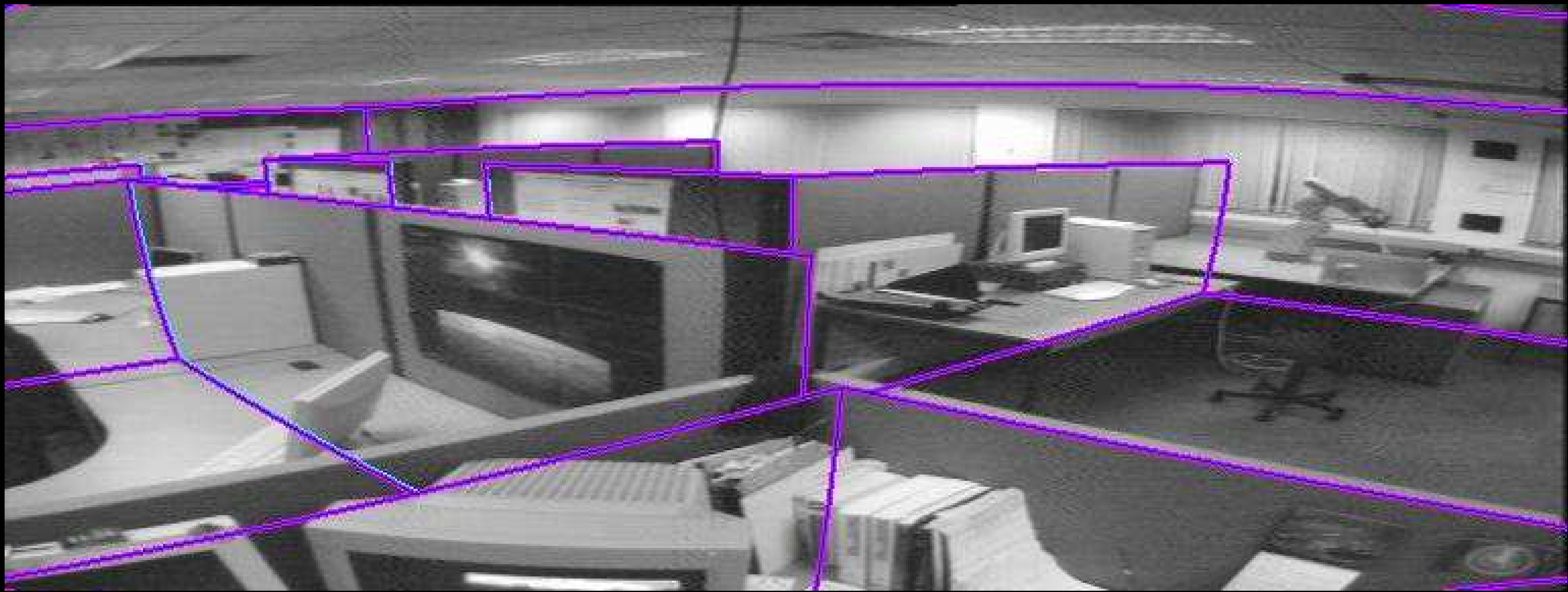
Multimodal posterior propagation



- Either tracker can be wrong
  - Edge tracker can get correspondence wrong
  - Point based tracker can drift
- Posterior can be multimodal
- Evaluate modes *next* frame when more data arrives

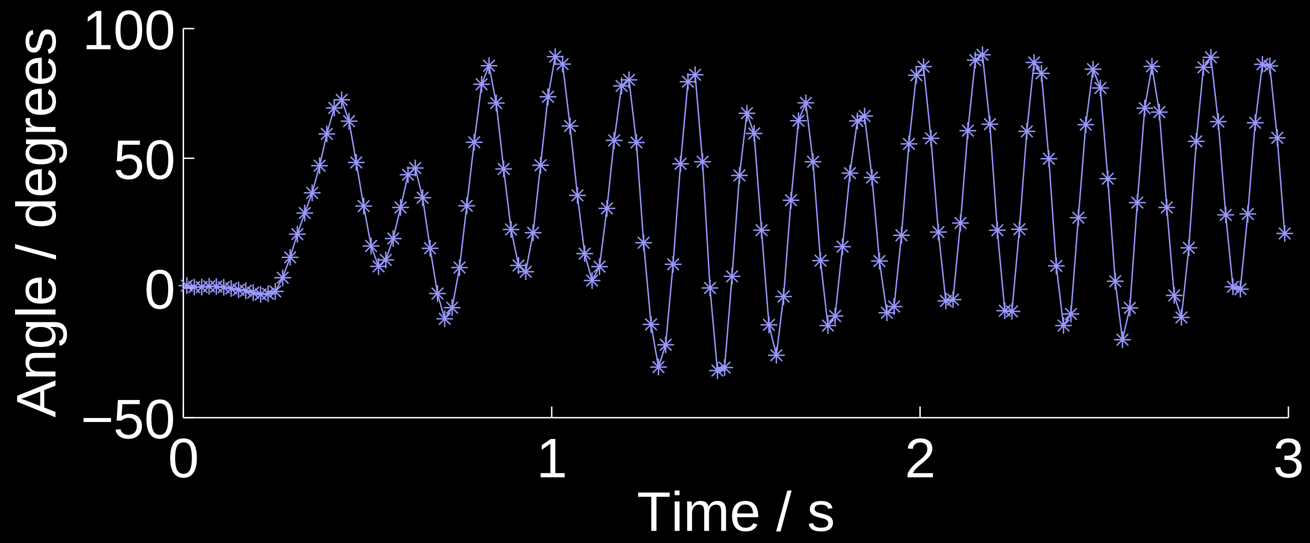
Results

Results - Camera shake



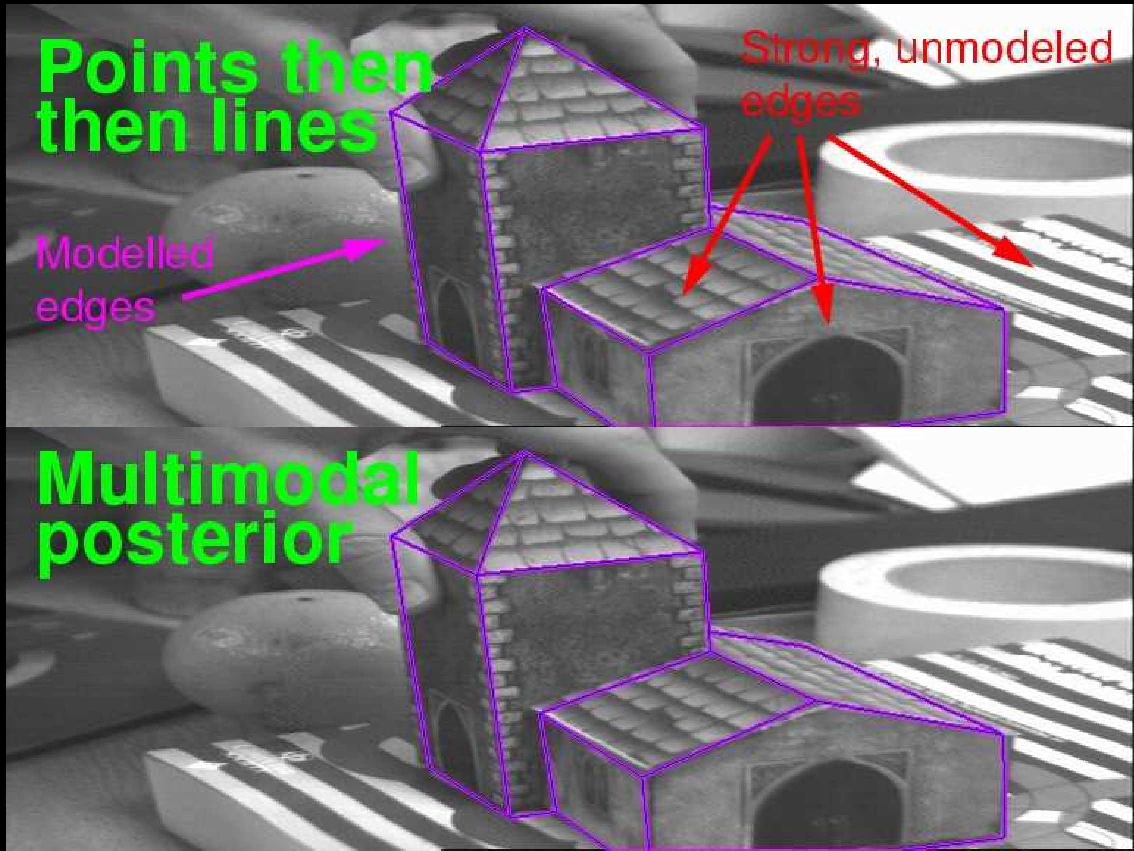
- Pick up camera and shake *really* hard
- Mainly tests point tracker
- Can you follow the video? I can’t (but my tracker can)

Results - Camera shake



- 6Hz Camera shake
- Up to 204 pixels prediction error (89 average)

Results - Strong unmodelled edges



- Strong unmodelled edges frequently break the edge tracker
- Breaks without proper sensor fusion

Results - Handheld camera

Pick up the camera and run around the lab

Summary

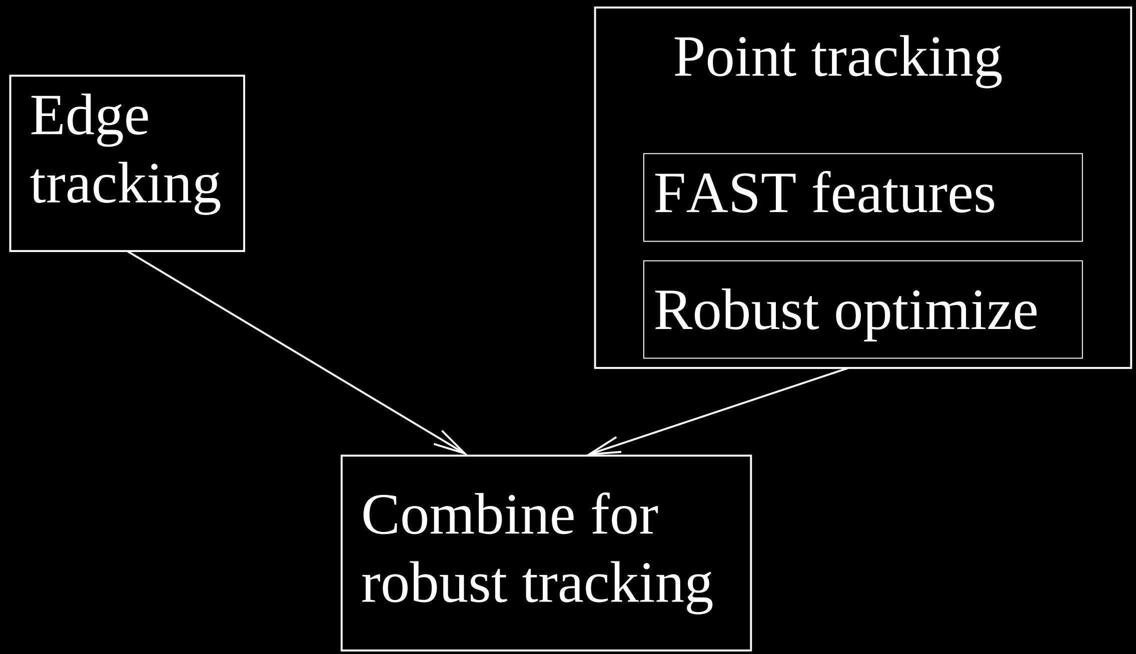
- An efficient, robust point based tracker

  Built using:

  - A *very* fast, repeatable feature detector
    - ⋆ Now used in crowd tracking, SLAM, localisation...
  - Online learning of match quality
- Careful modelling allows combination of trackers for extra robustness
- Technologies described apply more widely than to 6 DOF tracking

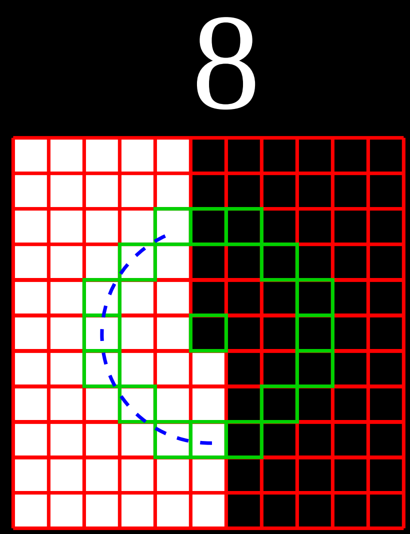
Any questions?

Model based tracking

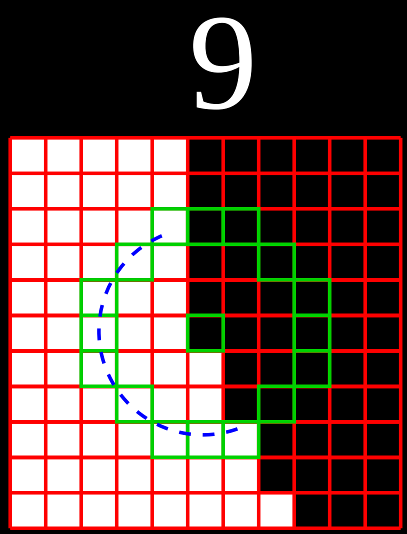


- Different failure modes
  - Combine for extra robusteness
  - Combination is difficult
    - ⋆ Statistics are non Gaussian

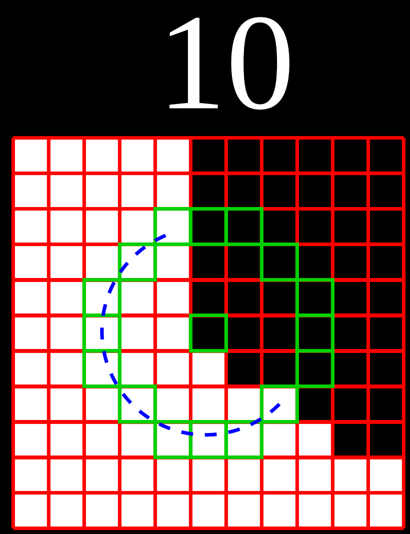
Example detectors



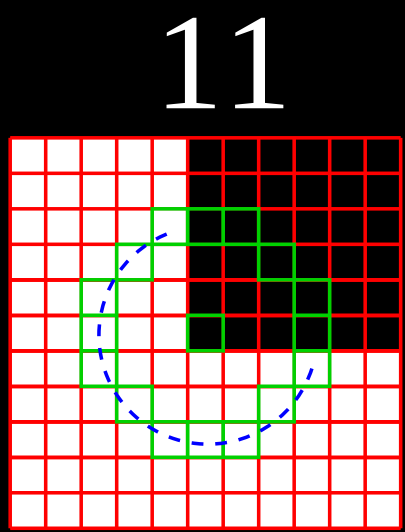
Example detectors



Example detectors



Example detectors



Example detectors

12