Interaction with an autonomous agent

Jim Little
Laboratory for Computational Intelligence
Computer Science
University of British Columbia
Vancouver BC Canada

Overview

- Local Invariant Features for Object Recognition
  - David Lowe UBC
  - What features we use for landmarks
- Vision-based Mapping with Backward Correction
  - Stephen Se MDR, David Lowe, Jim Little UBC
  - How we use visual landmarks to create a map
- Global Localization using Distinctive Visual Features
  - Stephen Se MDR, David Lowe, Jim Little UBC
  - How a robot finds where it is in a map

Local Invariant Features for Object Recognition

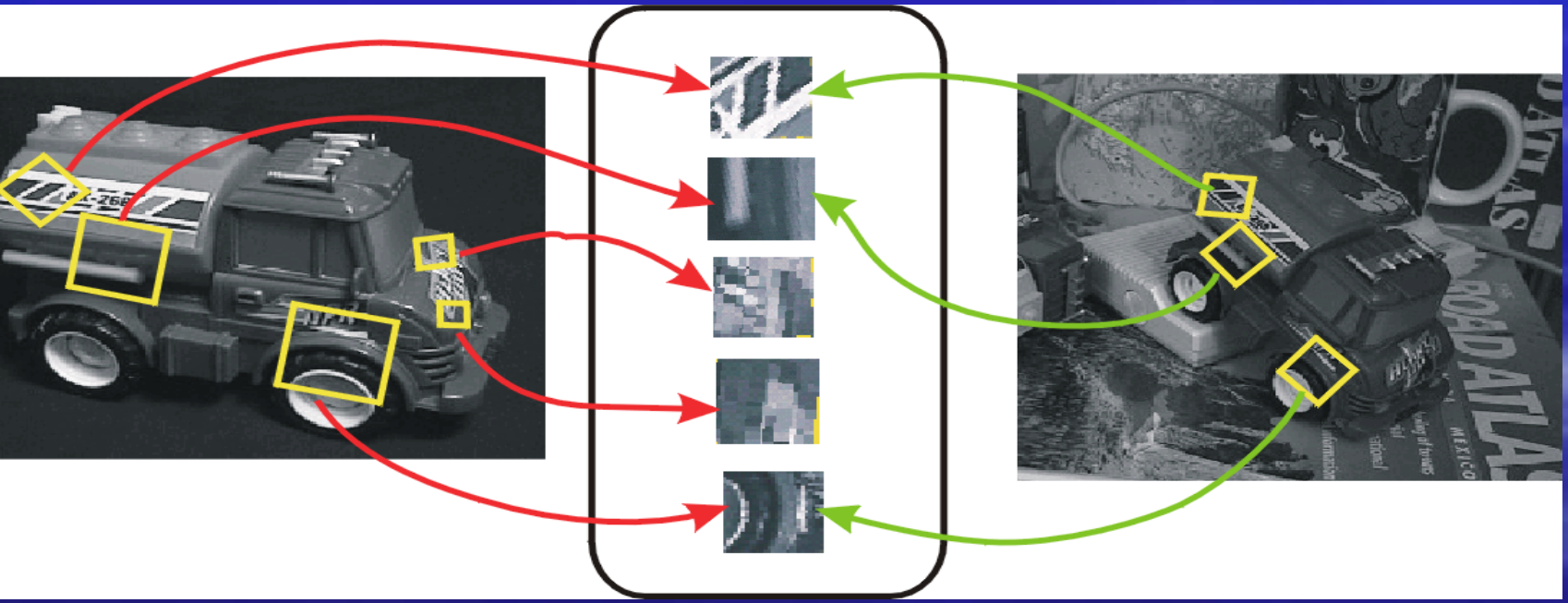
David Lowe UBC

Feature Detection

- Much faster than template matching
- **History:**
  - Edge segments: Roberts (65), Grimson (84)
  - Groupings: Lowe (87), Nelson (97)
  - Regions, Color: Jacobs (93), Swain (91)
- **Problem:** hard to find features that are frequent, stable, and distinctive

Invariant Local Features

- Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, and other imaging parameters



SIFT Features

Advantages of invariant local features

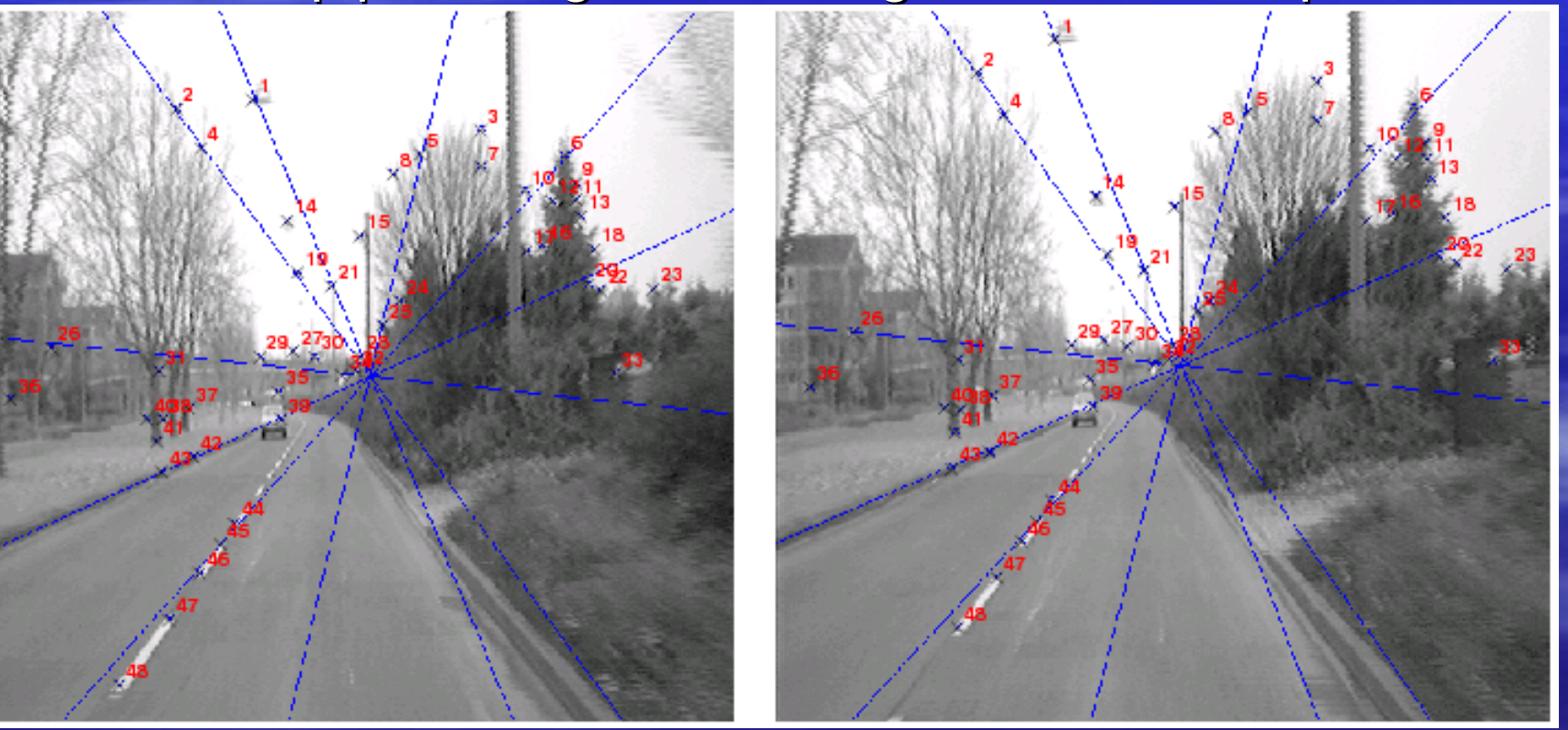
- **Locality:** features are local, so robust to occlusion and clutter (no prior segmentation)
- **Distinctiveness:** individual features can be matched to a large database of objects
- **Quantity:** many features can be generated for even small objects
- **Efficiency:** close to real-time performance
- **Extensibility:** can easily be extended to wide range of differing feature types, with each adding robustness

History

- **Torr & Murray (93); Zhang, Deriche, Faugeras, Luong (94)**
  - Apply Harris corner detector for feature locations
  - Match nearby points using proximity and correlation applied at corner locations
  - Only invariant to feature translation
- **Schmid & Mohr (96)**
  - Compute rotational invariants at Harris corners that are also distinctive
  - Demonstrated object recognition with high clutter and occlusion
  - Still needed: invariance to scale and 3D viewpoint, model fitting

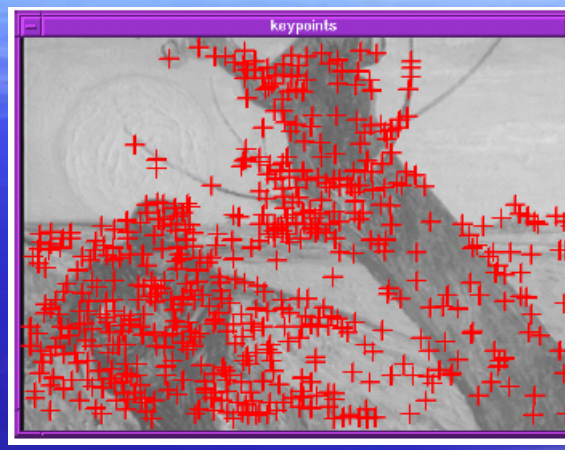
Zhang, Deriche, Faugeras, Luong (95)

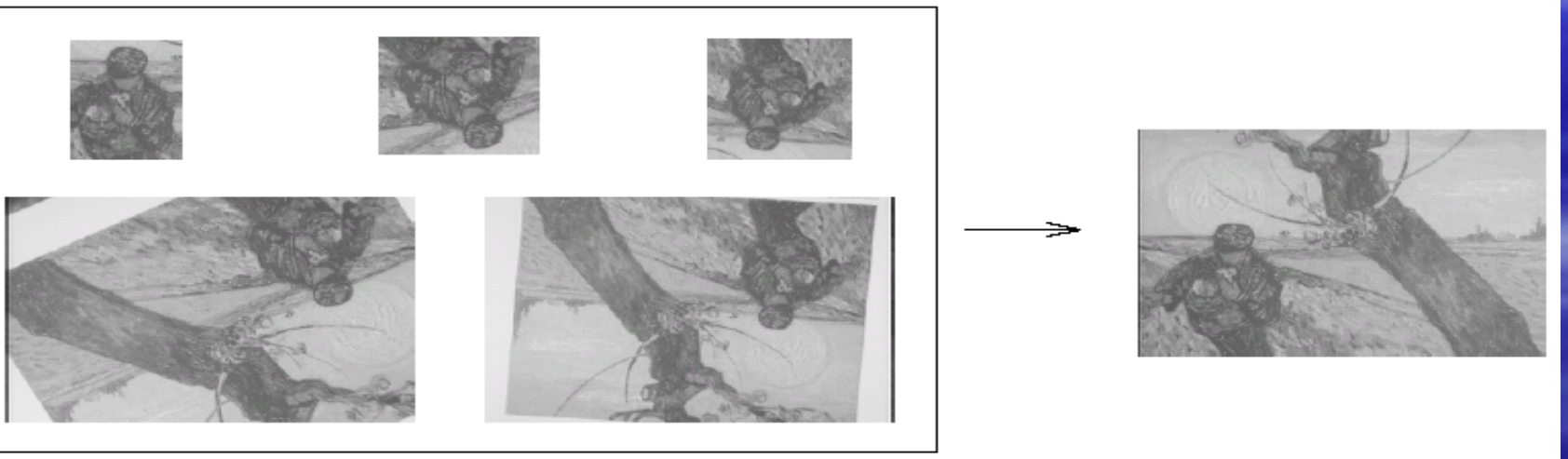
- Apply Harris corner detector
- Match points by correlating only at corner points
- Derive epipolar alignment using robust least-squares



Cordelia Schmid & Roger Mohr (97)

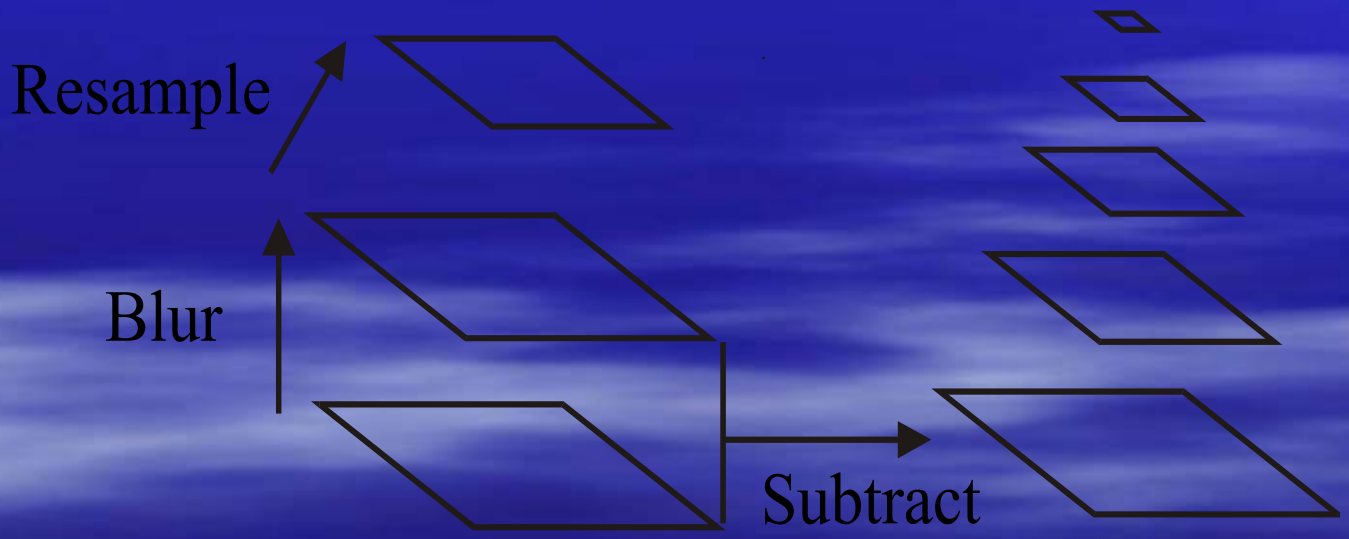
- Apply Harris corner detector
- Use rotational invariants at corner points
  - However, not scale invariant. Sensitive to viewpoint and illumination change.





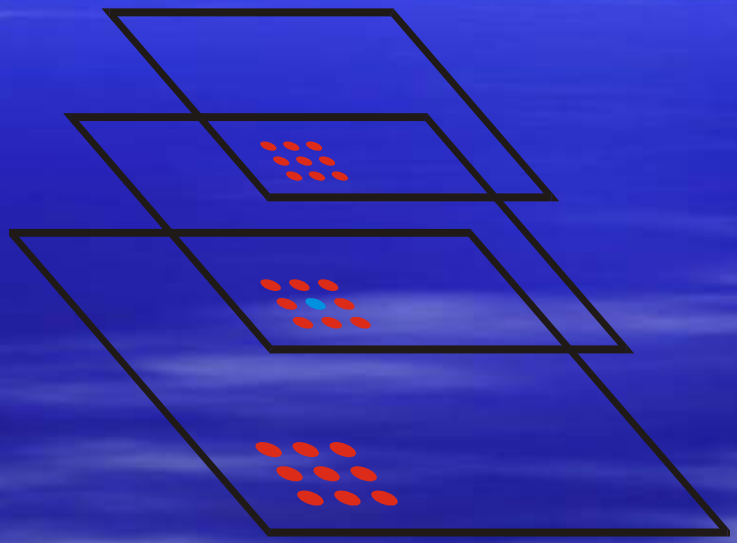
Build Scale-Space DOG Pyramid

- Most efficient function is to compute difference of Gaussian pyramid (Burt)
- Single blur by $\sqrt{2}$ used for DOG and resampling



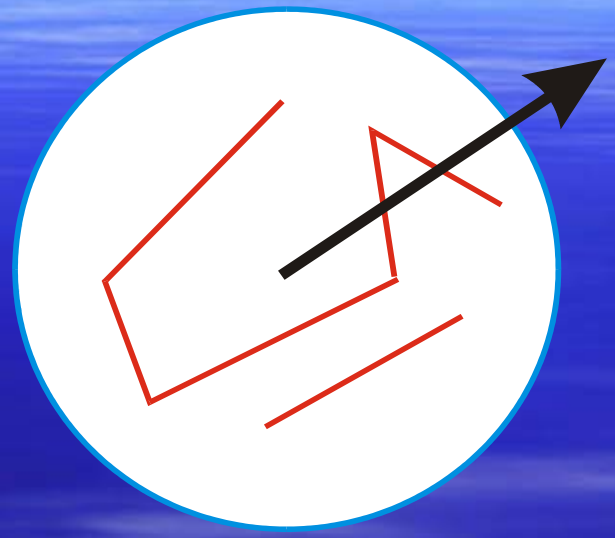
Key point localization

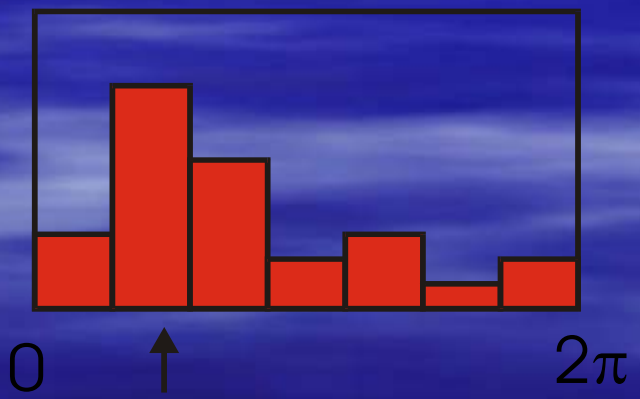
- Detect maxima and minima of difference of Gaussian in scale space
- Remove points with low stability (minimum contrast required in all directions)



Select canonical orientation

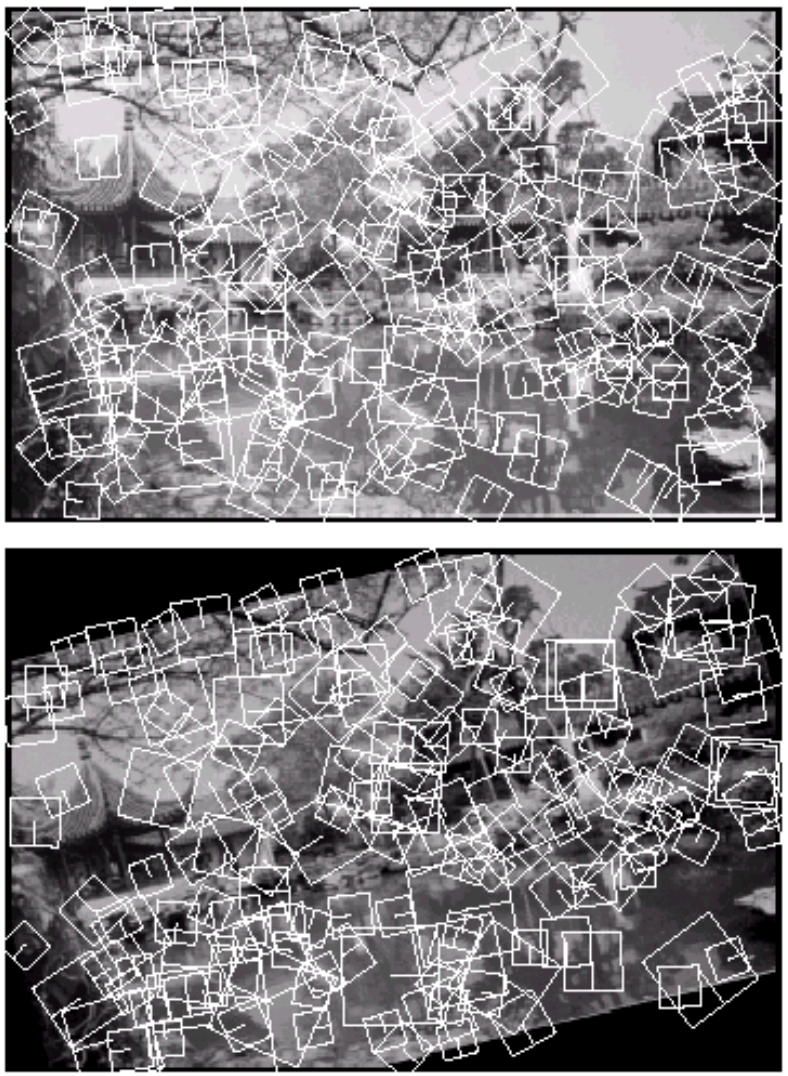
- Create histogram of local gradient directions computed at selected scale
- Assign canonical orientation at peak of smoothed histogram
- Each key specifies stable 2D coordinates (x, y, scale, orientation)





Testing for stability

- Check for stability following affine projection, change of brightness and contrast, and addition of noise
- This Figure shows only 2 octaves. Typical image produces 1000 keys.



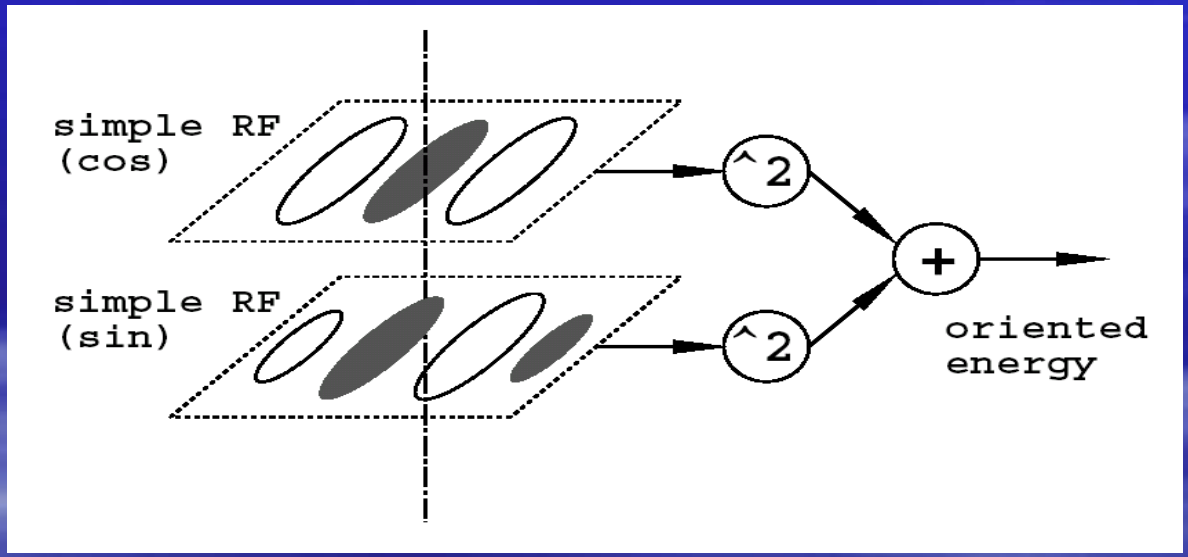
Statistics for stability testing

- Applied to 20 diverse images with 15,000 keys

Image transformation	Match %	Ori %
A. Increase contrast by 1.2	89.0	86.6
B. Decrease intensity by 0.2	88.5	85.9
C. Rotate by 20 degrees	85.4	81.0
D. Scale by 0.7	85.1	80.3
E. Stretch by 1.2	83.5	76.1
F. Stretch by 1.5	77.7	65.0
G. Add 10% pixel noise	90.3	88.4
H. All of A,B,C,D,E,G.	78.6	71.8

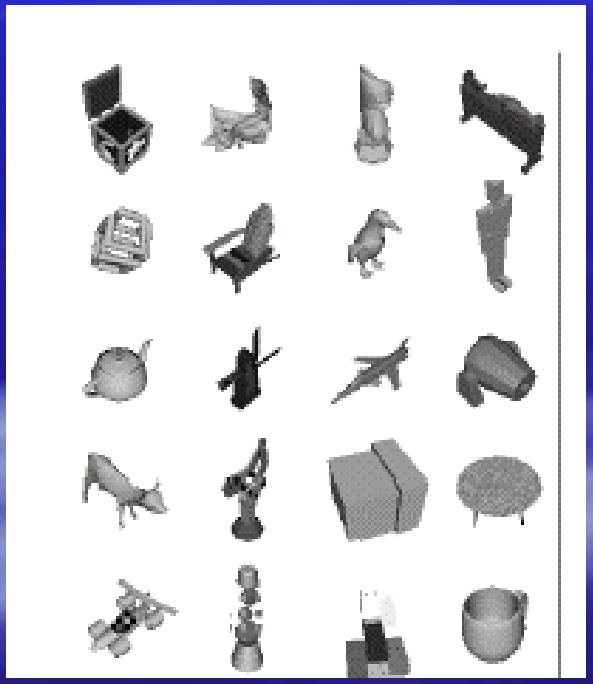
Creating features stable to viewpoint change

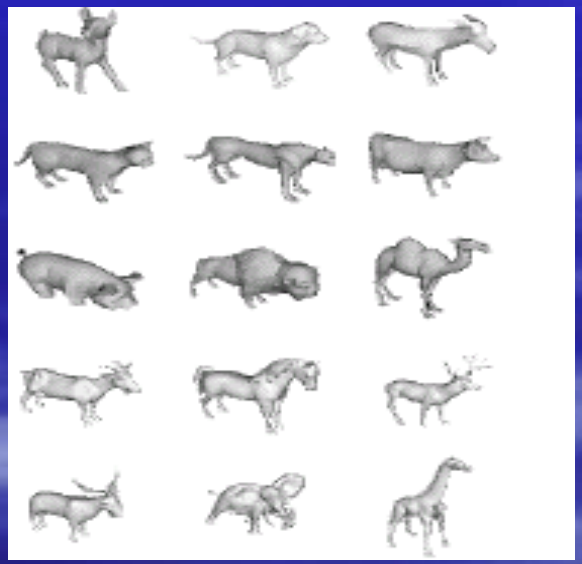
- Edelman, Intrator & Poggio (97) showed that complex cell outputs are better for 3D recognition than simple correlation

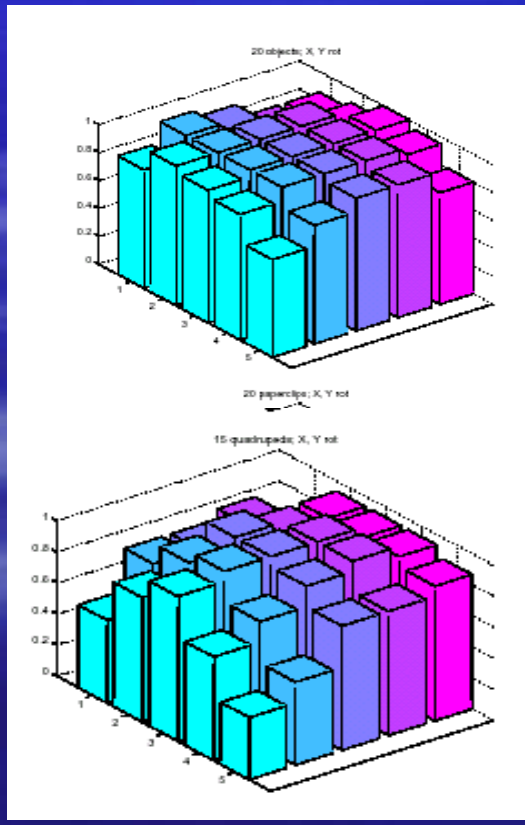


Stability to viewpoint change

- Classification of rotated 3D models (Edelman 97):
  - Complex cells: 94% vs simple cells: 35%

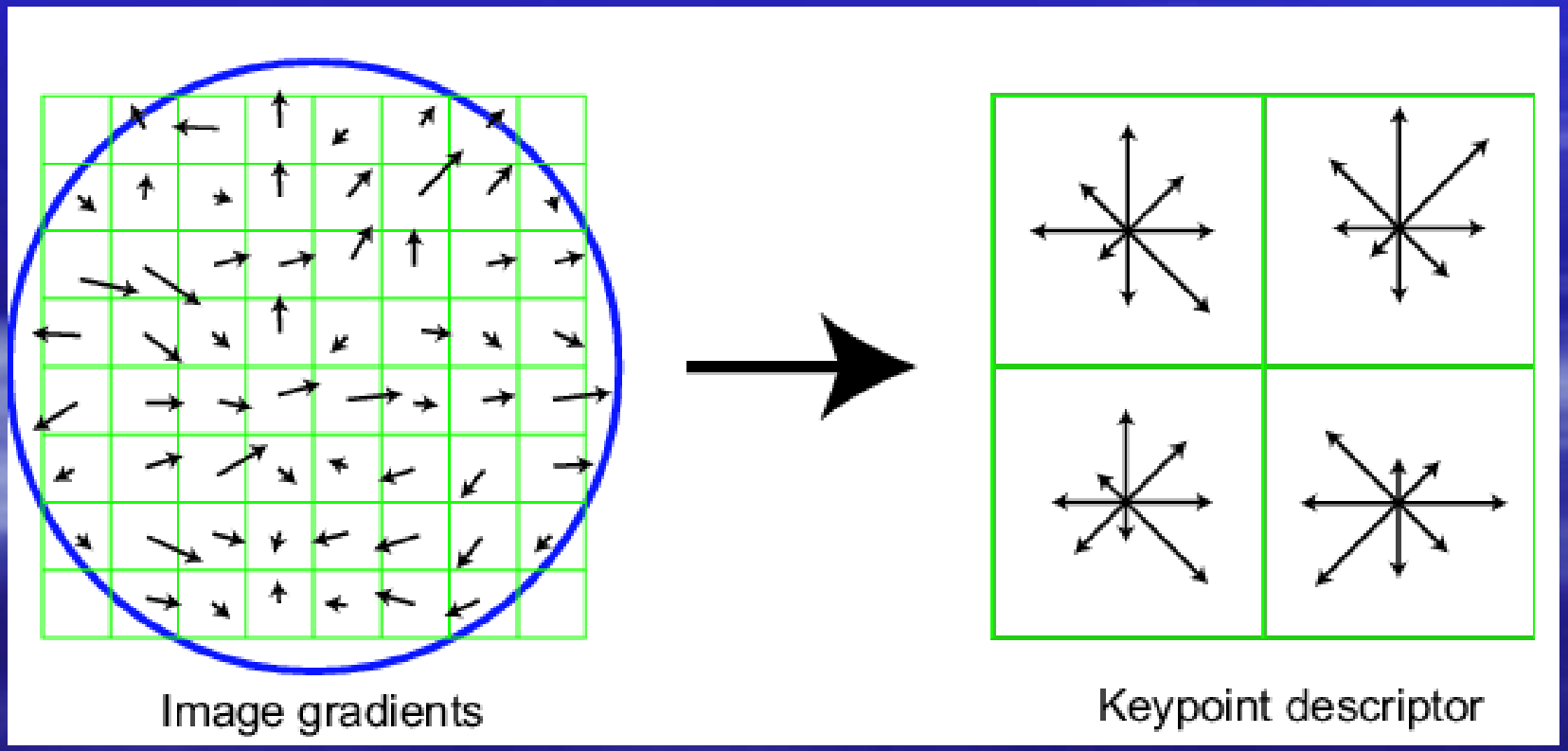






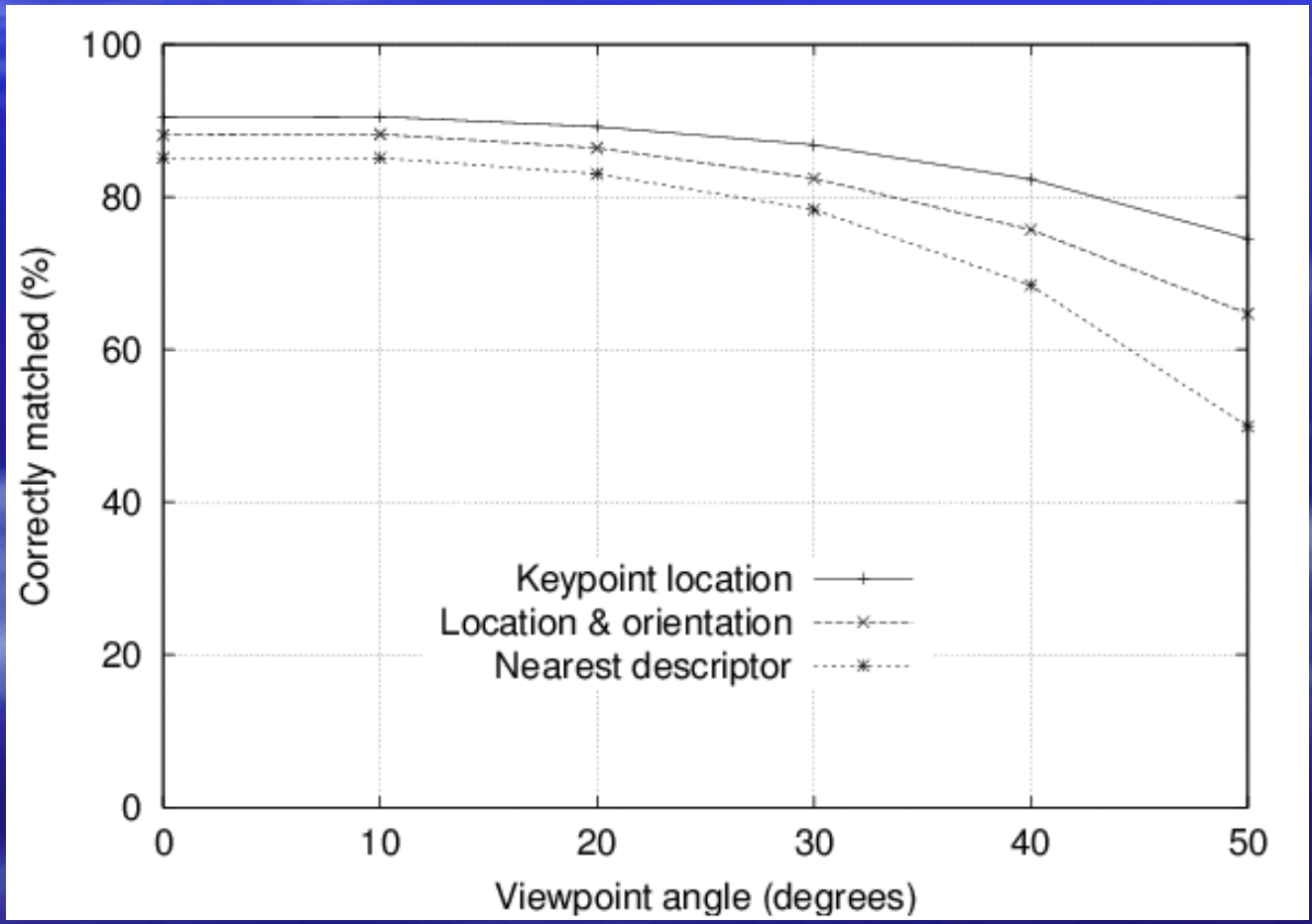
SIFT vector formation

- Thresholded image gradients are sampled over 16x16 array of locations in scale space
- Create array of orientation histograms
- 8 orientations x 4x4 histogram array = 128 dimensions



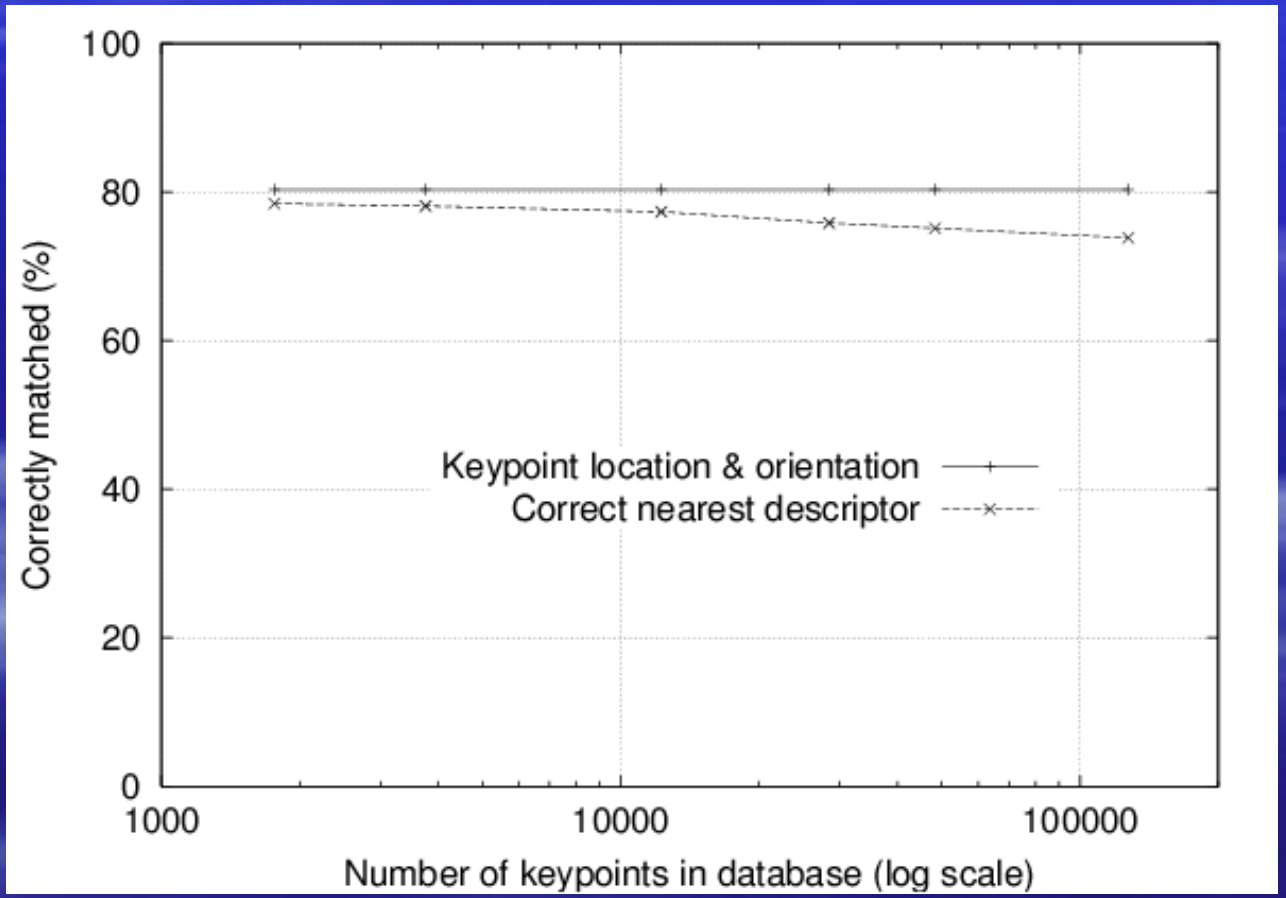
Feature stability

- Match features after random change in image scale & orientation, with 2% image noise, and affine distortion
- Find nearest neighbor in database of 30,000 features



Distinctiveness of features

- Vary size of database of features, with 30 degree affine change, 2% image noise
- Measure % correct for single nearest neighbor match

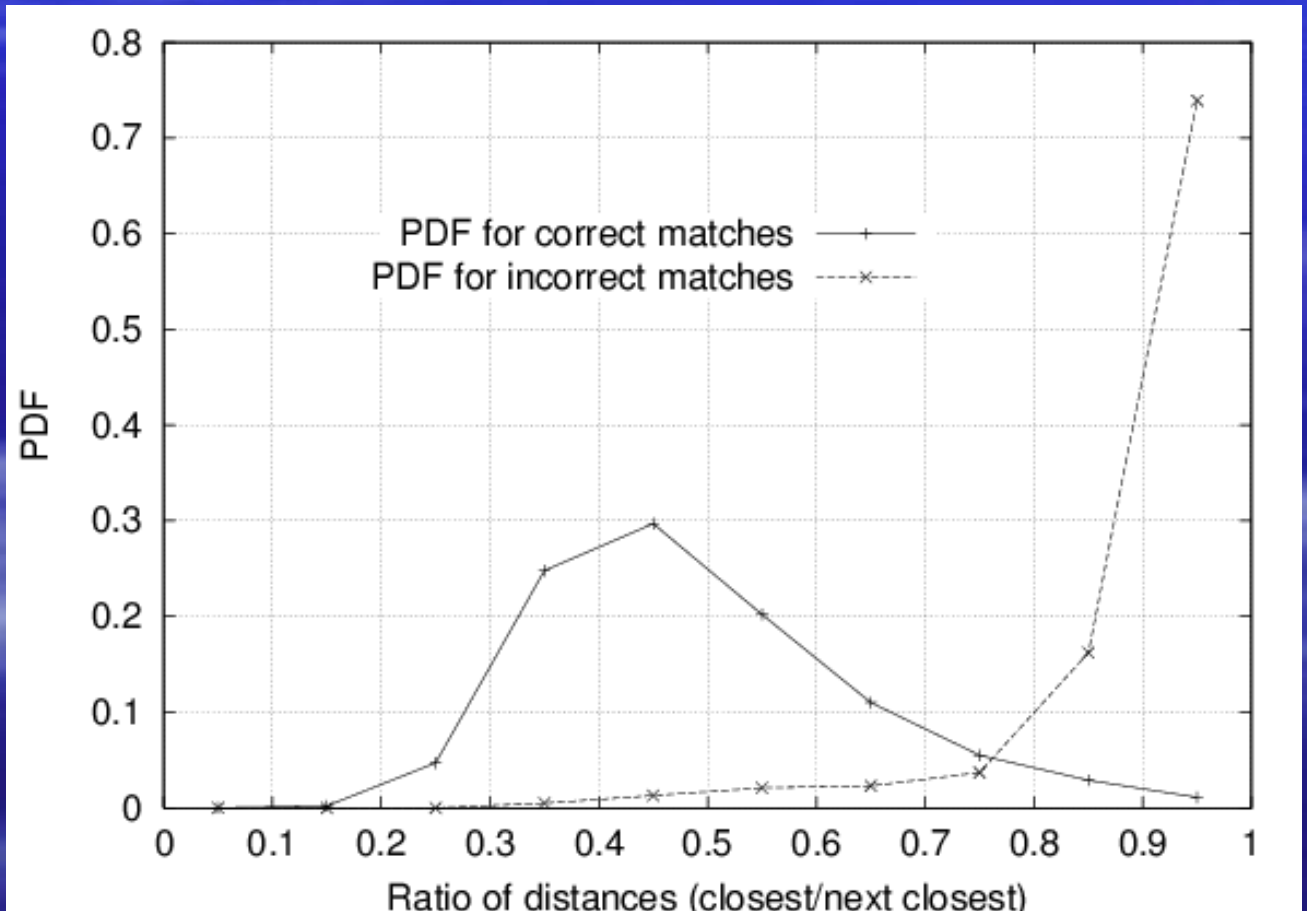


Detecting 0.1% inliers among 99.9% outliers

- Hypotheses are generated by matching each feature to nearest neighbor vectors in database
  - Use best-bin-first (Beis & Lowe, 97) modification to k-d tree algorithm
- Need to recognize clusters of just 4 consistent features among 4000 feature match hypotheses
- LMS or RANSAC would be hopeless!
- Generalized Hough transform
  - Hash each key according to model ID and pose, allowing for error in similarity approximation
  - Second-level hashing avoids need to form empty bins

Probability of correct match

- Compare distance of nearest neighbor to second nearest neighbor (from different object)
- Threshold of 0.8 provides excellent separation



Model verification

- Examine all clusters in Hough transform with at least 3 features
- Perform least-squares fit to model (similarity, affine, or 3D). Discard outliers and perform top-down check for additional features.
- Evaluate probability that match is correct
  - Use Bayesian model, with probability that features would arise by chance if object was *not* present
  - Takes account of object size in image, textured regions, model feature count in database, accuracy of fit (Lowe, CVPR 01)

Solution for affine parameters

- Affine transform of [x,y] to [u,v]:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

- Rewrite to solve for transform parameters:

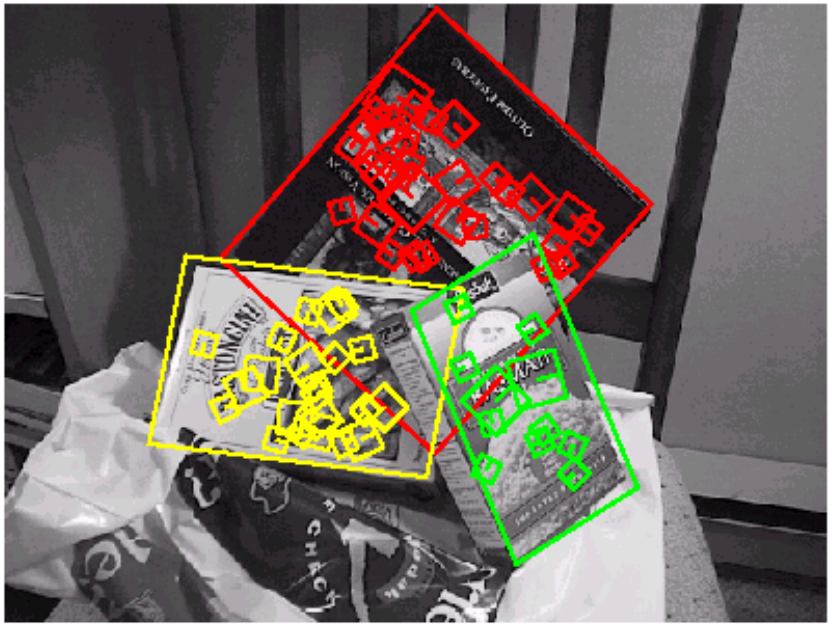
$$\begin{bmatrix} x & y & 0 & 0 & 1 & 0 \\ 0 & 0 & x & y & 0 & 1 \\ & & \cdots & & & \\ & & \cdots & & & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_x \\ t_y \end{bmatrix} = \begin{bmatrix} u \\ v \\ \vdots \end{bmatrix}$$

Planar texture models

- Models for planar surfaces with SIFT keys

Planar recognition

- Planar surfaces can be reliably recognized at a rotation of 60° away from the camera
- Affine fit approximates perspective projection
- Only 3 points are needed for recognition



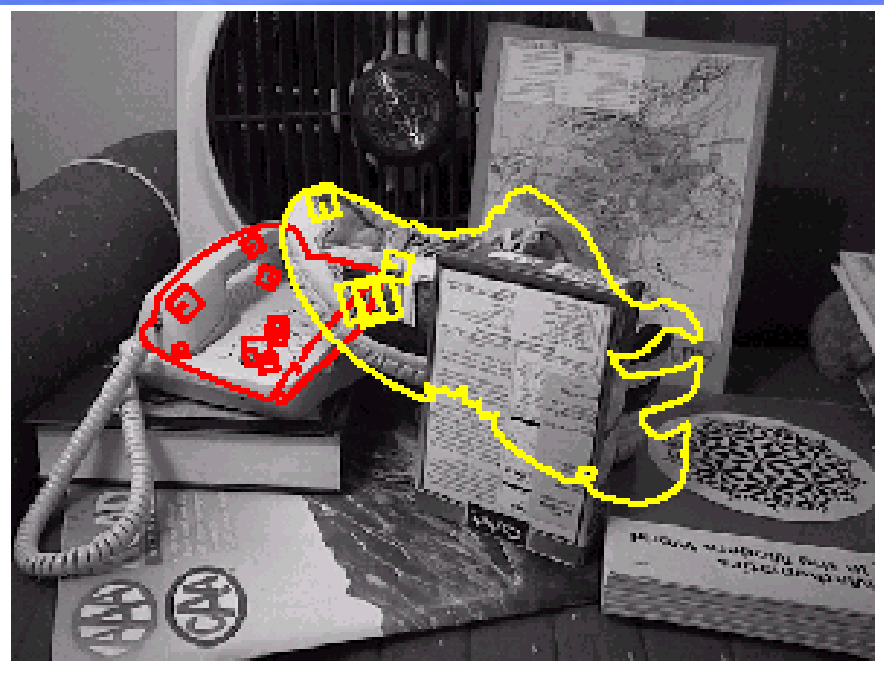
3D Object Recognition

- Extract outlines with background subtraction

3D Object Recognition

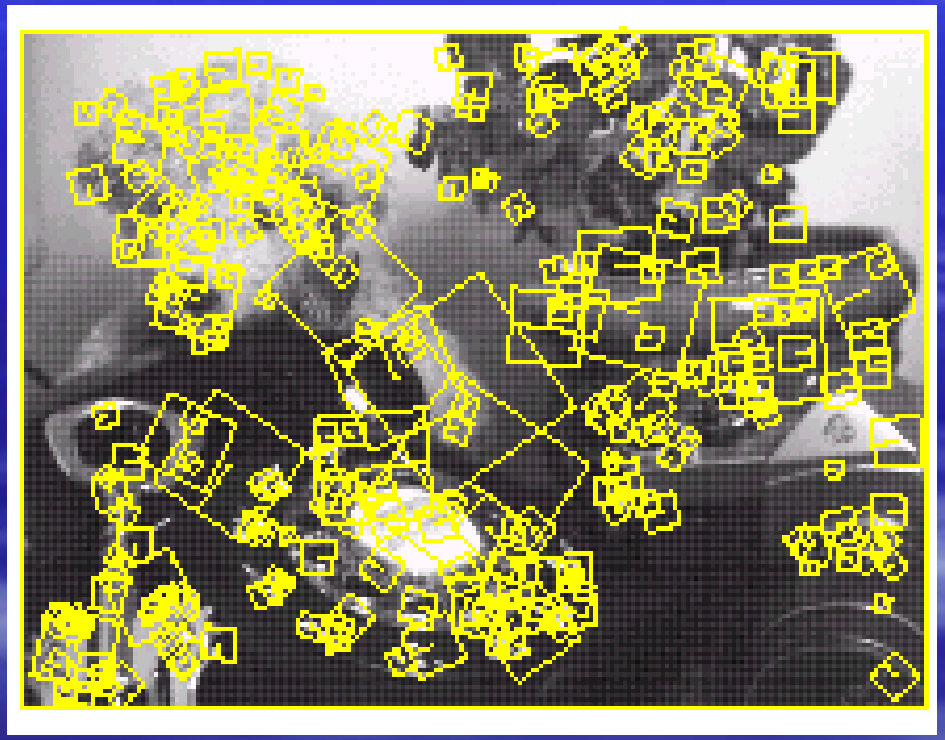
- Only 3 keys are needed for recognition, so extra keys provide robustness
- Affine model is no longer as accurate

Recognition under occlusion



Test of illumination invariance

- Same image under differing illumination

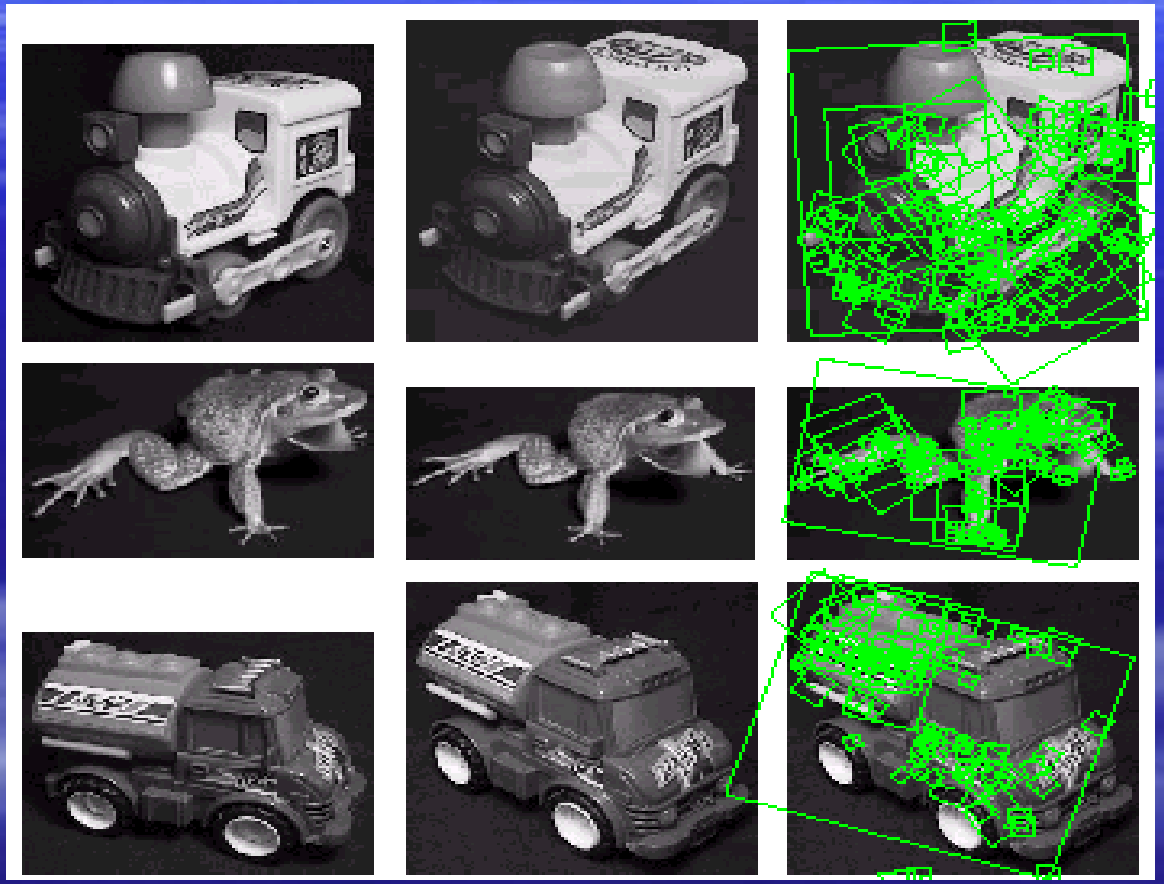


273 keys verified in final match

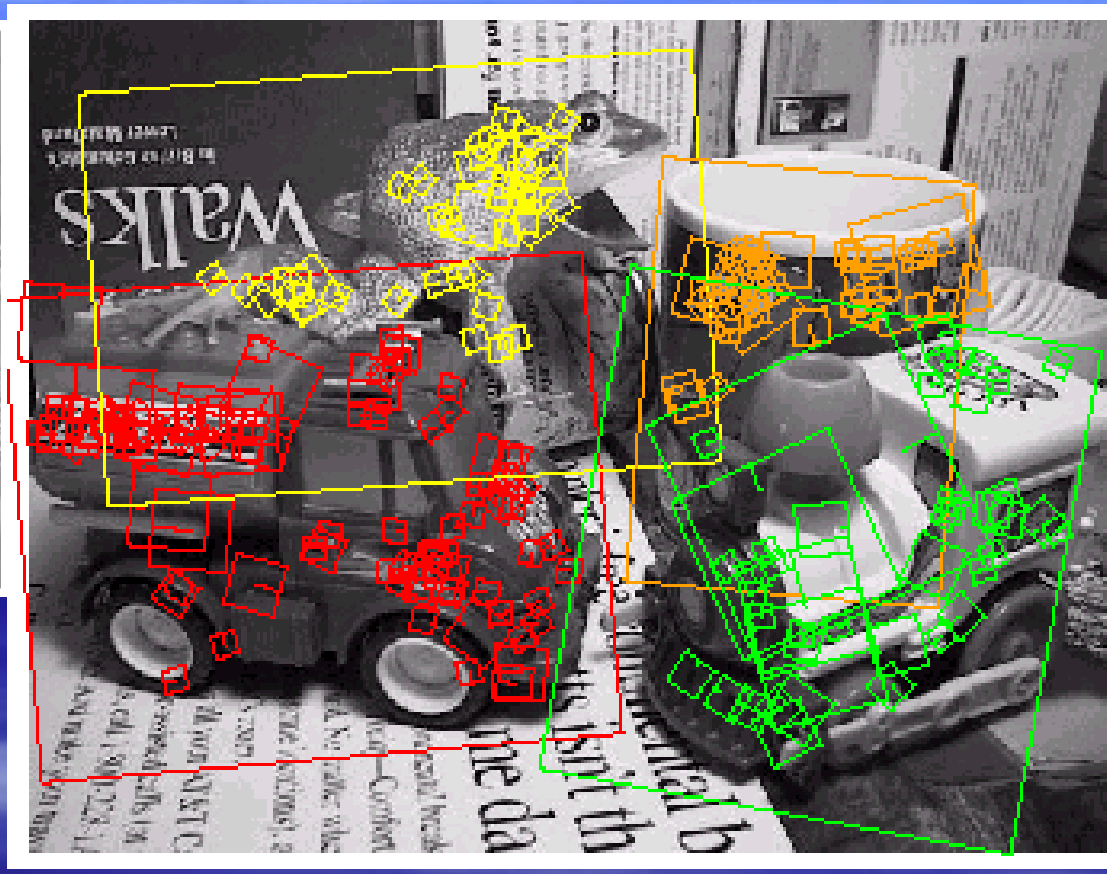
View interpolation for 3D viewpoint change

- Change in 3D viewpoint under orthography can be correctly modeled by linear view interpolation (Seitz & Dyer [1995])
  - Requirement is that interpolation be along epipolar lines
- When two training images agree with low residual, then features are combined. Otherwise, perform linear interpolation.
- Linear interpolation can handle some non-rigid and generic objects (such as change in facial expression)

Examples of view interpolation

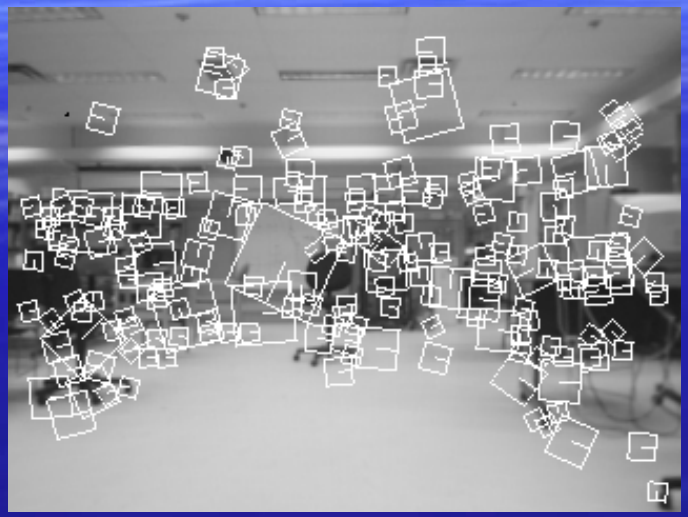


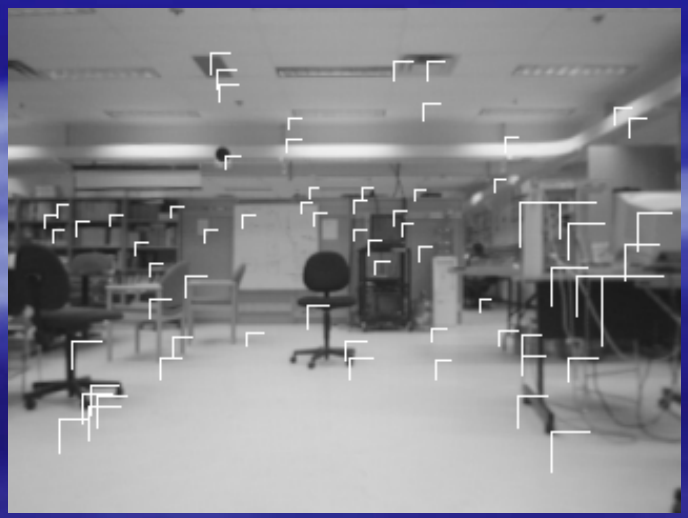
Recognition using View Interpolation



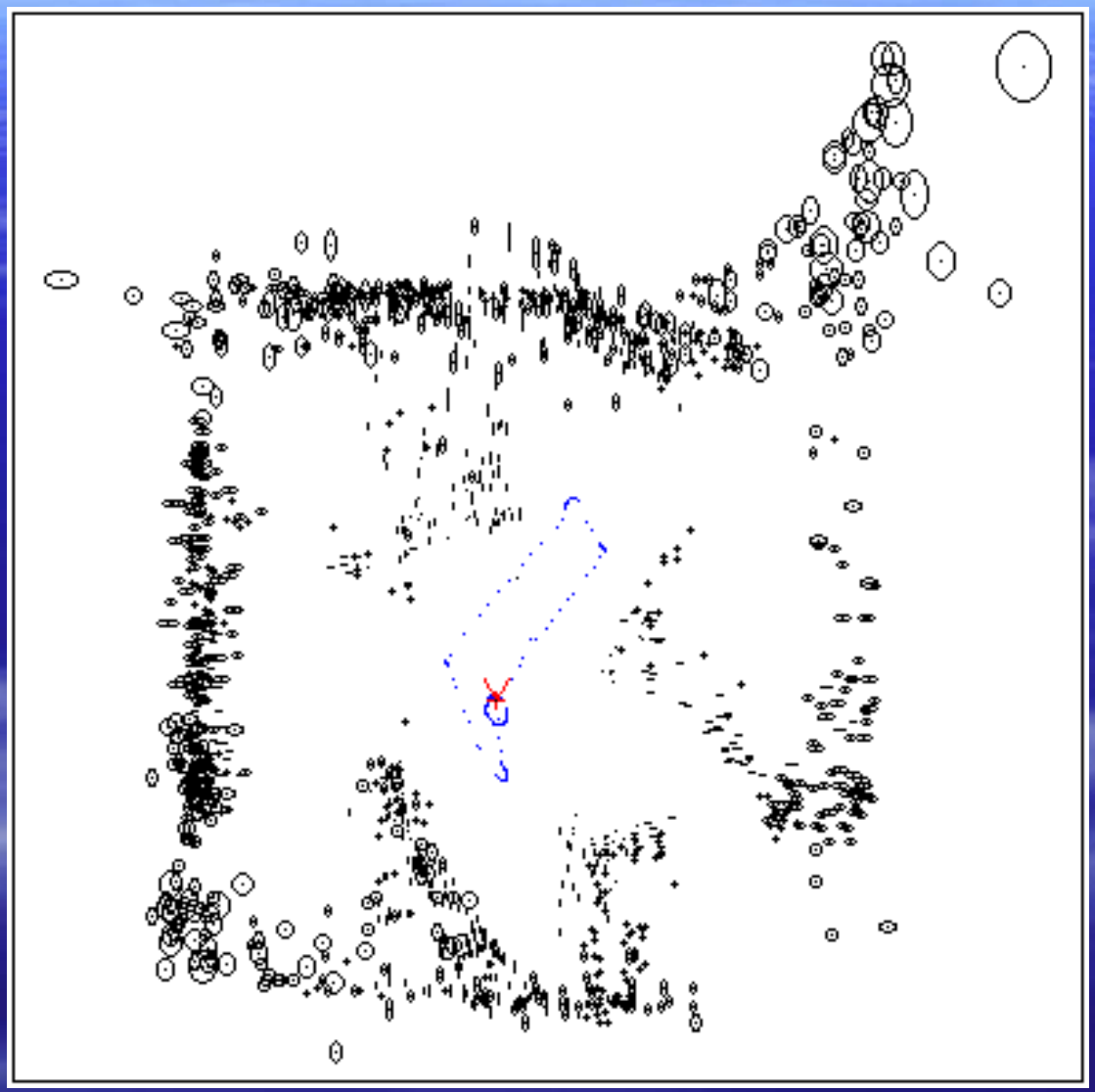
Robot Localization

- Joint work with Stephen Se, Jim Little

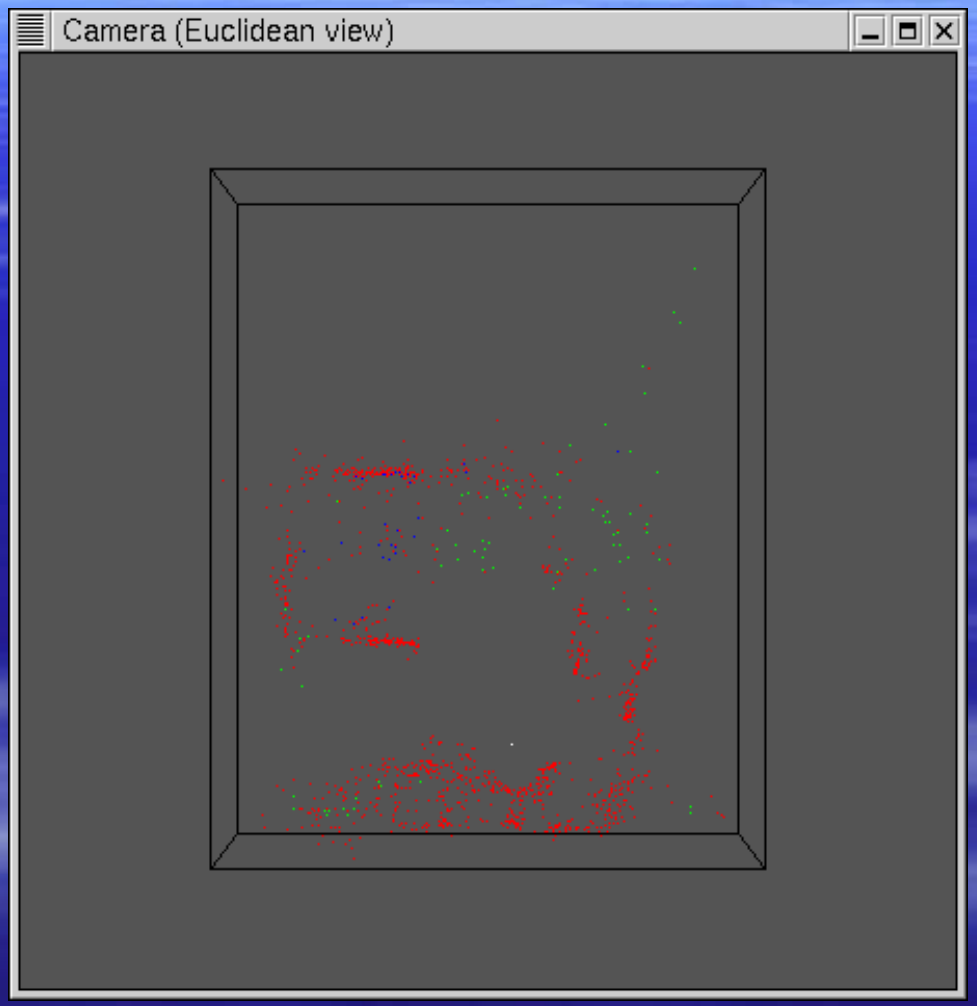




Map continuously built over time

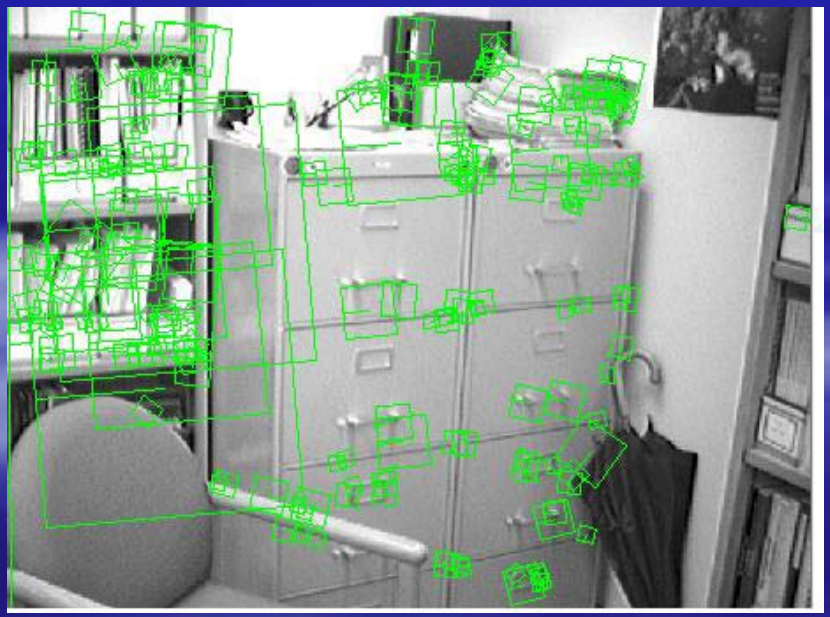


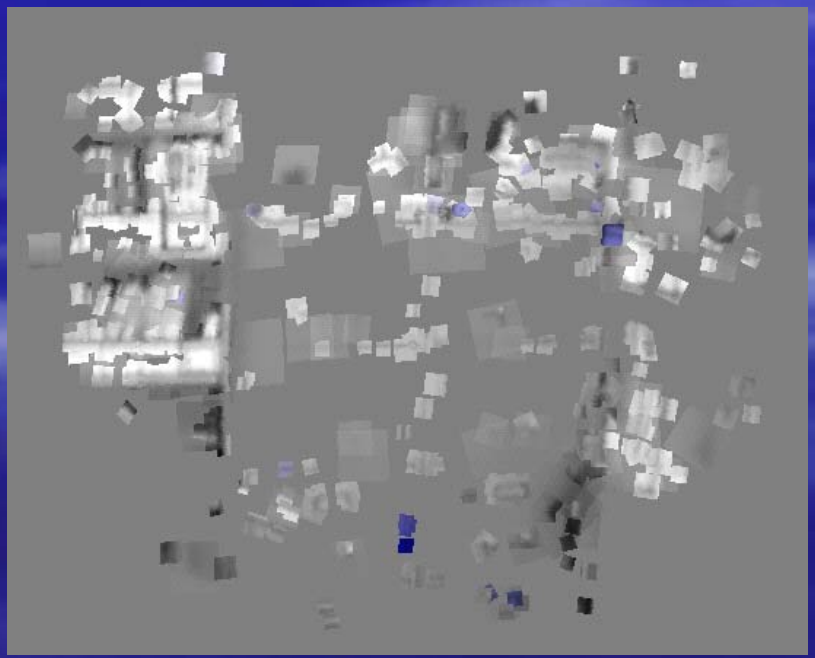
Locations of map features in 3D



Multi-view solution for 3D structure

- Match features between 3 images from nearby viewpoints
- Use robust least-squares minimization to solve for camera locations and 3D structure
  - Approach of (Szeliski & Kang, 94)
- Match 3D model to additional views, integrating new features.
  - Provides robust model integrating features under wide range of imaging conditions





Computation times

- Recognition time: 0.5 seconds on 600MHz Pentium
  - 0.3 seconds to build scale-space pyramid
  - 0.2 seconds for indexing and verification
  - Image sizes: 512x384 pixels, greyscale
- Additional 0.5 second preprocessing for each model
  - Times should scale sub-linearly for additional models
  - About 100-500K memory for each model

Comparison to template matching

- Costs of template matching
  - 250,000 locations x 30 orientations x 4 scales = 30,000,000 evaluations
  - Does not easily handle partial occlusion and other variation without large increase in template numbers
- Costs of SIFT approach
  - 1000 evaluations (reduction by factor of 30,000)
  - Features are more invariant to illumination, 3D rotation, and object variation
  - Use of many small subtemplates increases robustness to partial occlusion and other variations

Future directions

- Build true 3D models
  - Integrate features from large number of training views and perform continuous learning
- Feature classes can be greatly expanded
  - Affine-invariant features (Tuytelaars & Van Gool, Mikolajczyk & Schmid, Schaffalitzky & Zisserman, Brown & Lowe)
  - Incorporate color, texture, varying feature sizes
  - Include edge features that separate figure from ground
- Address instance recognition of generic models
  - Map feature probabilities to measurements of interest (e.g., specific person, expression, age)

Conclusions

- Object recognition can be achieved with a dense set of local features of intermediate complexity
- A staged approach to feature detection leads to efficient matching
- Final model-based verification process is important for selecting features that form a consistent object interpretation.
- The approach can be easily extended with new feature types

Vision-based Mapping with Backward Correction

David Lowe (UBC)
Jim Little (UBC)
Stephen Se (MD Robotics, Canada)

Outline

- Introduction
- SIFT Stereo and SLAM
- Map Alignment
- Building Submaps
  - pair-wise & incremental
- Closing the Loop
  - global constraint
  - landmark uncertainty
- Conclusion

Erik

Introduction

- Vision-based Simultaneous Localization And Mapping (SLAM) algorithm
  - builds 3D map continuously
  - no backward correction when closing the loop
  - problems with large slippages and long-term drifts

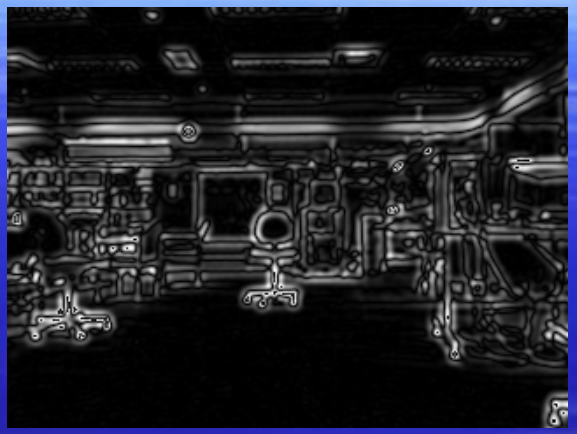
- Map building
  - incrementally integrate new data to map
  - full bundle adjustment
  - related work
    - Leonard and Feder 99, Lu and Milios 97, Gutmann and Konolige 99, Thrun et al 98

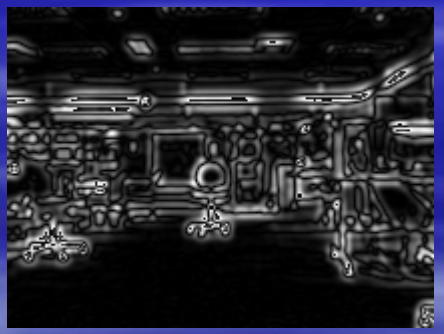
SIFT Features

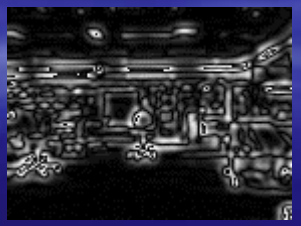
- SIFT (Scale Invariant Feature Transform)
  - object recognition (Lowe 1999)
  - invariant to image translation, scaling, rotation, illumination changes, affine projection
  - previous feature detectors sensitive to scale
- Algorithm
  - subtract image from its Gaussian smoothed image to get DOG for each pyramid level
  - key locations at maxima & minima relative to surrounding pixels and adjacent scales
  - subpixel image location, scale and orientation

SIFT Detection





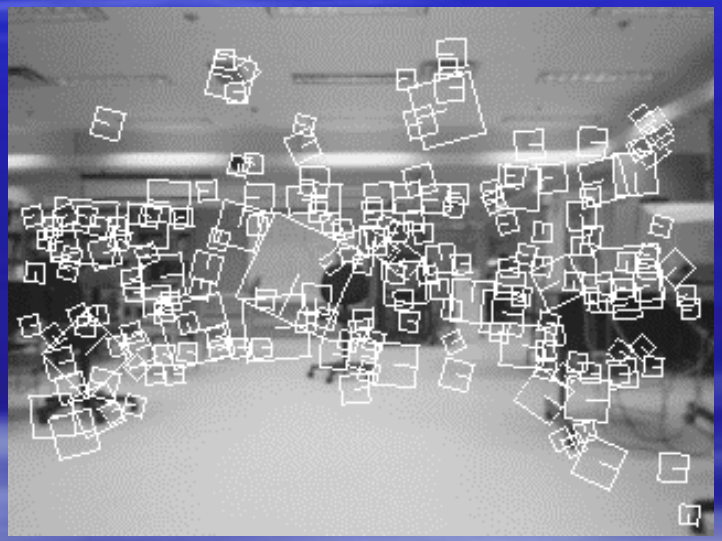
Resampled image

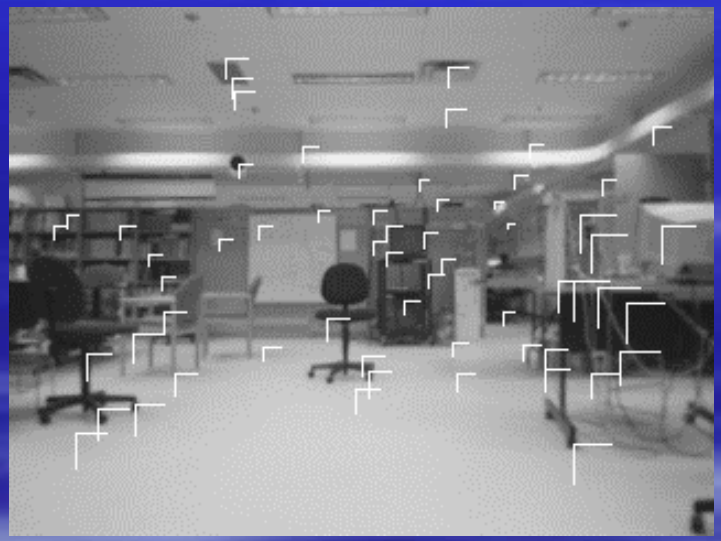
Gaussian smoothed

DOG

SLAM

- SIFT stereo
  - epipolar and disparity constraints
  - SIFT scale and orientation constraints





- 3D SIFT landmarks kept in database map
- Camera ego-motion estimation
  - least-squares minimization

Map Alignment

- Build submaps of environment and align them afterwards to obtain consistent global 3D map
- Distinctive features required to match scenes
- SIFT local image characteristics
  - measure local image gradient at a number of orientation relative to location, scale & orientation
  - blur gradient locations to reduce sensitivity
  - 2 x 2 grid with 4 orientation in each cell
  - 16 element vector describing SIFT feature
  - scalable easily for more specificity

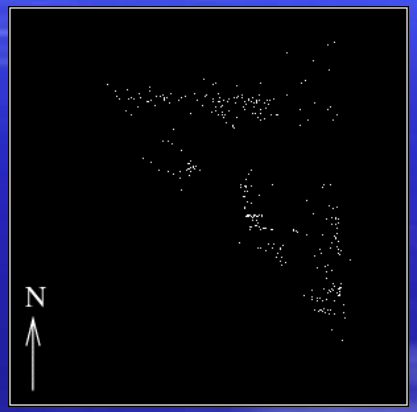
RANSAC Approach

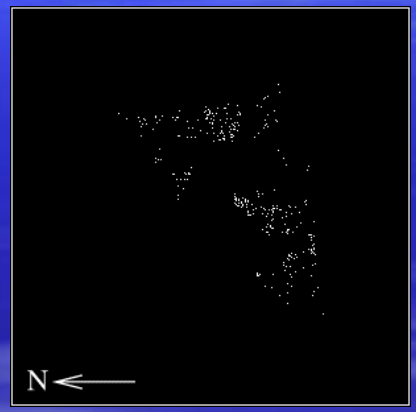
- Given 2 sets of SIFT 3D database map
- RANdom SAmpling Consensus
- Find best database landmark for each feature, based on local image characteristic & height
- Repeat 50 times
  - randomly select 2 tentative matches
  - compute pose parameter (X,Z,θ)
  - check all tentative matches for support
- Select alignment with most support
- LS minimization on all supporting matches

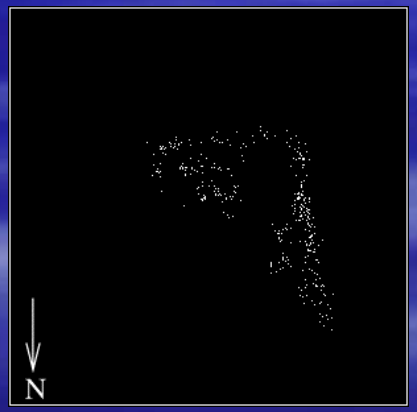
Building Submaps

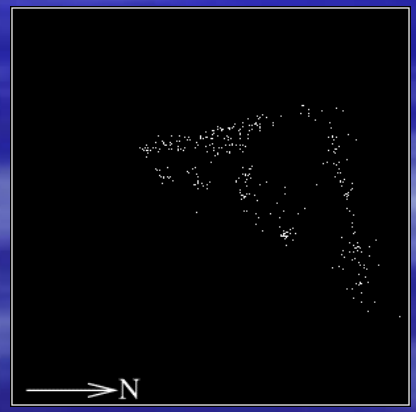
- Odometry is corrected locally so far in SLAM
- Not sufficient for long-term drifts
- Start building a new submap when a drift is detected
- When a drift occurs, few feature matches at current position
- Build new submaps on a regular basis, e.g. every M frames, to avoid drift accumulation
- Combine the submaps at the end

Example









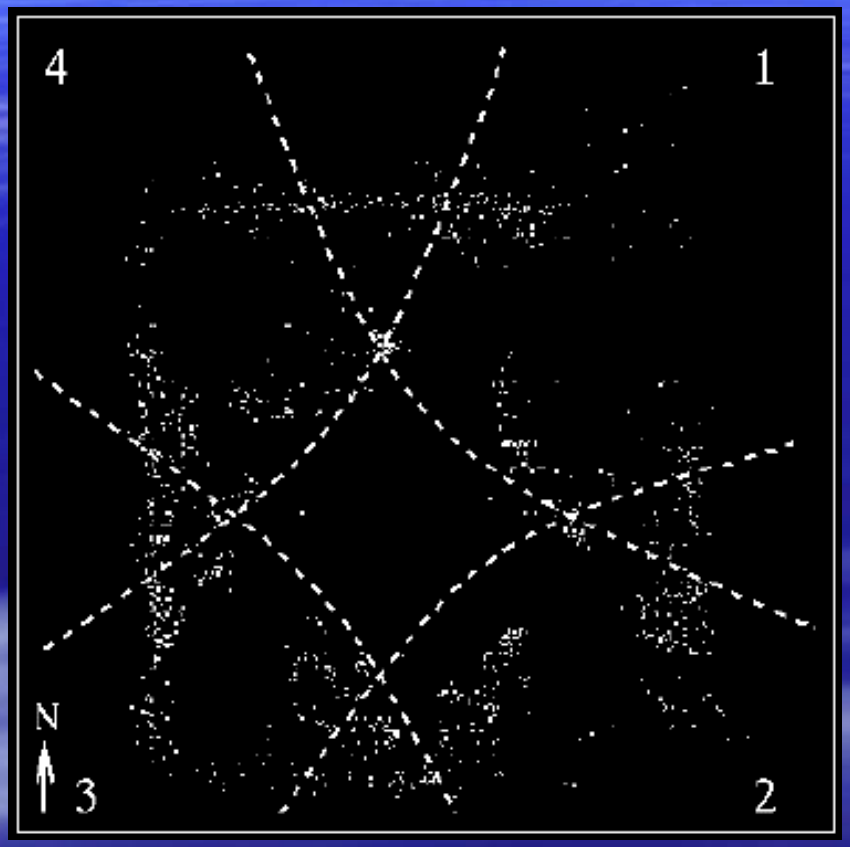
Original map with slips

Submaps built due to slips

Submap Alignment

- Start building the next submap when the previous submap is terminated
  - overlapping SIFT landmarks exist
- Pair-wise alignment
  - align consecutive pair of submaps
  - obtain transformation from submap1 to submap2, submap2 to submap3, submap3 to submap4, etc
- Incremental alignment
  - use the new combined map to align with the next submap

Alignment Results



Pair-wise

Incremental

Closing the Loop

- Pair-wise and incremental alignments are the same if each submap overlaps only with previous submap
- Close-the-loop can be detected by checking significant overlap of landmarks between current submap and initial submap

  - When closing the loop,
    - incremental alignment attributes all correction to last alignment
    - should spread out backward correction throughout all alignments

Global Minimization

- For submaps 1, 2, …, n which closes the loop, obtain n pair-wise alignments between each pair, including submap n to submap 1
- Set up matrix system
  - for all local pair-wise landmark matches
  - add global constraint for perfect alignment

$$T_1 T_2 \ldots T_{n-1} T_n = I$$

- Carry out least-squares minimization
  - use pair-wise alignments as initial estimates
  - minimizes local pair-wise errors
  - minimizes global constraint errors

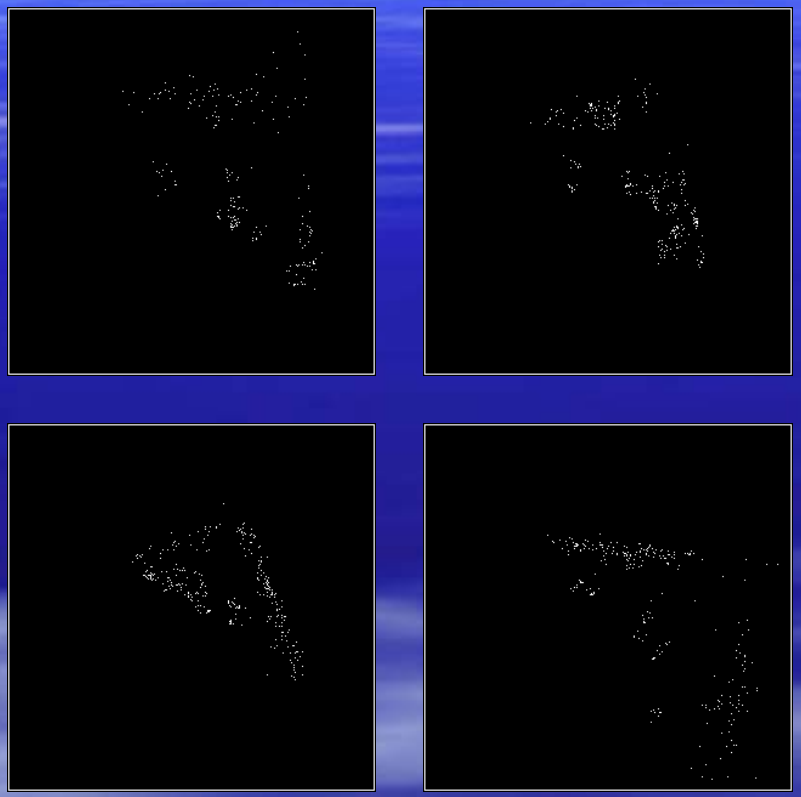
Landmark Uncertainty

- Covariance information for each SIFT landmark available from SLAM
- Use weighted least-squares minimization for both pair-wise alignment and the global constraint
  - trust the more reliable landmarks more for better local alignment
  - trust the more reliable pair-wise alignment more for better backward correction

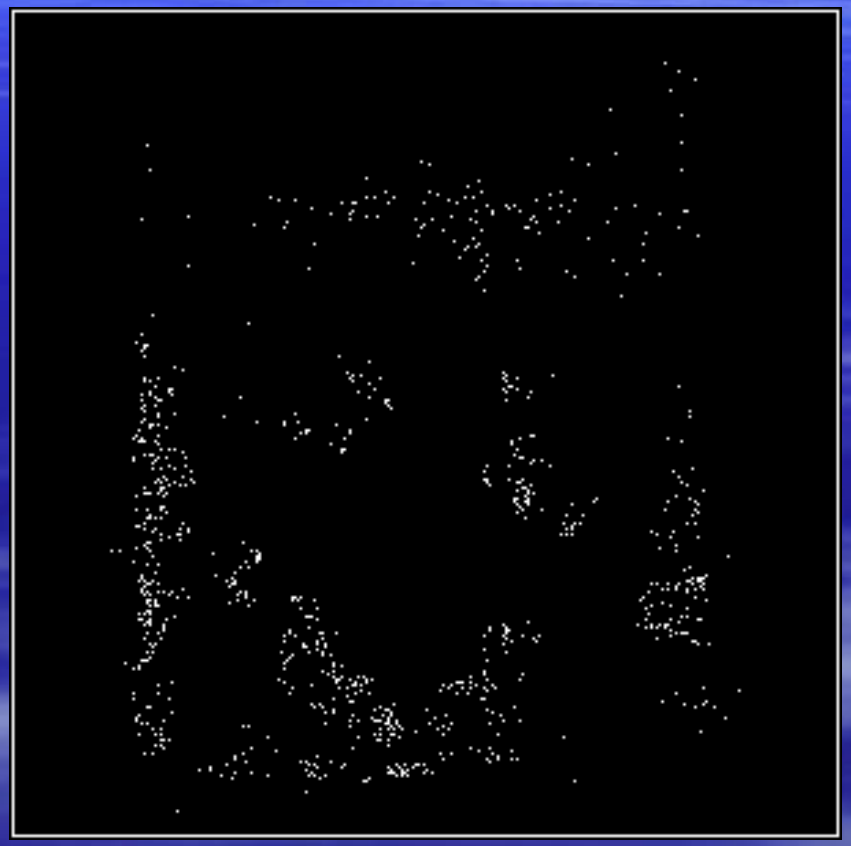
Results

- Compare misalignment for each approach, from the product of all transformations
  - pair-wise : (5.45cm,8.85cm,2.8deg)
  - weighted pair-wise : (3.00cm,5.92cm,0.43deg)
  - weighted pair-wise with backward correction : (0.15cm,0.37cm,0.03deg)
    - 0.03 sec to align a pair of submaps, 0.01 sec for global minimization on PIII 700MHz
    - Without initial pair-wise estimates, still converges in several iterations

More Results



Submaps built every 30 frames



Weighted least-squares
with backward correction

Conclusion and Future Work

- Vision-based SLAM using 3D SIFT features
  - highly distinctive for map alignment
- Map building with backward correction
  - building submaps which are pair-wisely aligned
  - constrained optimization on all alignments when closing the loop, to produce consistent global 3D map
  - allows backward correction between submaps
  - odometry corrected locally within submap
- Multi-robot collaboration
- Further experiments in larger environments

Global Localization using Distinctive Visual Features

David Lowe, Jim Little
University of British Columbia
Stephen Se (MD Robotics, Canada)

Outline

- Introduction
- SIFT Stereo and SLAM
- Global Localization
  - Hough Transform
  - RANSAC
  - Comparison
- Map Alignment
- Conclusion

Erik

Introduction

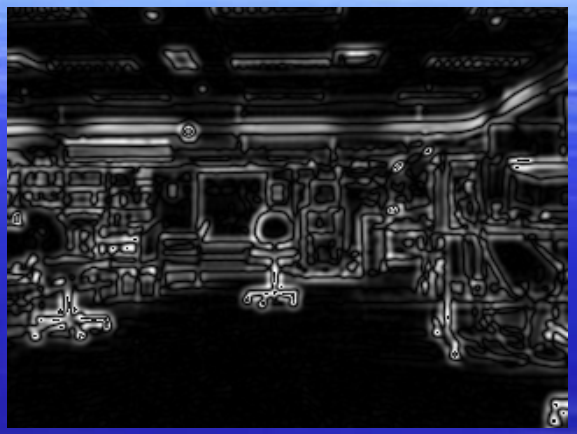
- Simultaneous Localization And Mapping (SLAM)
  - trinocular stereo vision-based
  - natural landmarks in unmodified environments
  - compensate for odometry errors
- Global localization
  - handle serious positioning errors
  - kidnapped robot problem
- Using distinctive visual landmarks
  - global localization from just one frame
  - different from stochastic localization methods

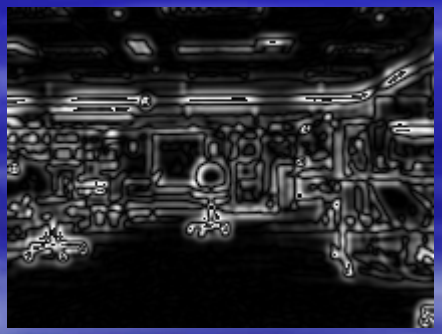
SIFT Features

- SIFT (Scale Invariant Feature Transform)
  - object recognition (Lowe 1999)
  - invariant to image translation, scaling, rotation, illumination changes, affine projection
  - previous feature detectors sensitive to scale
- Algorithm
  - subtract image from its Gaussian smoothed image to get DOG for each pyramid level
  - key locations at maxima & minima relative to surrounding pixels and adjacent scales
  - subpixel image location, scale and orientation

SIFT Detection





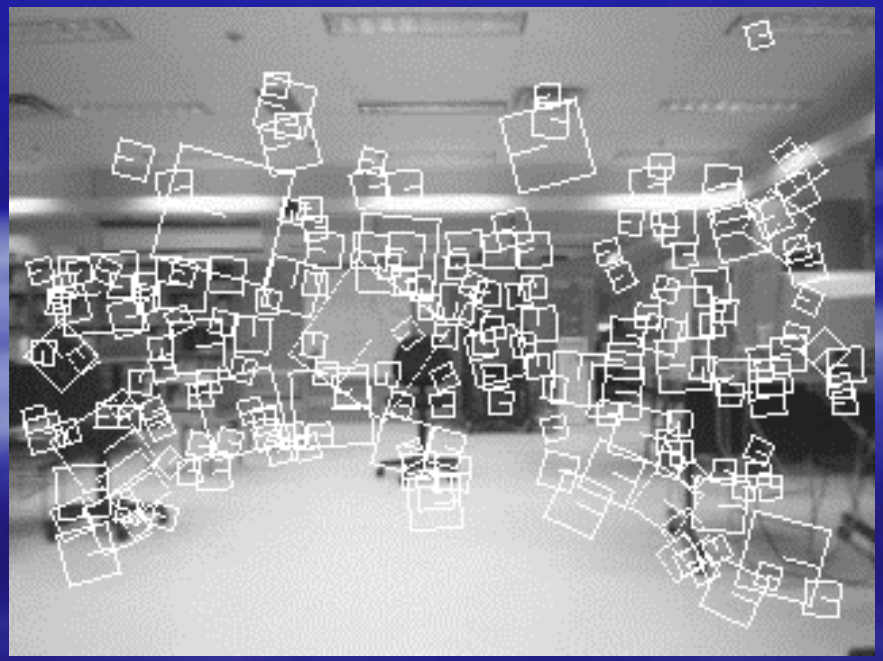

Resampled image

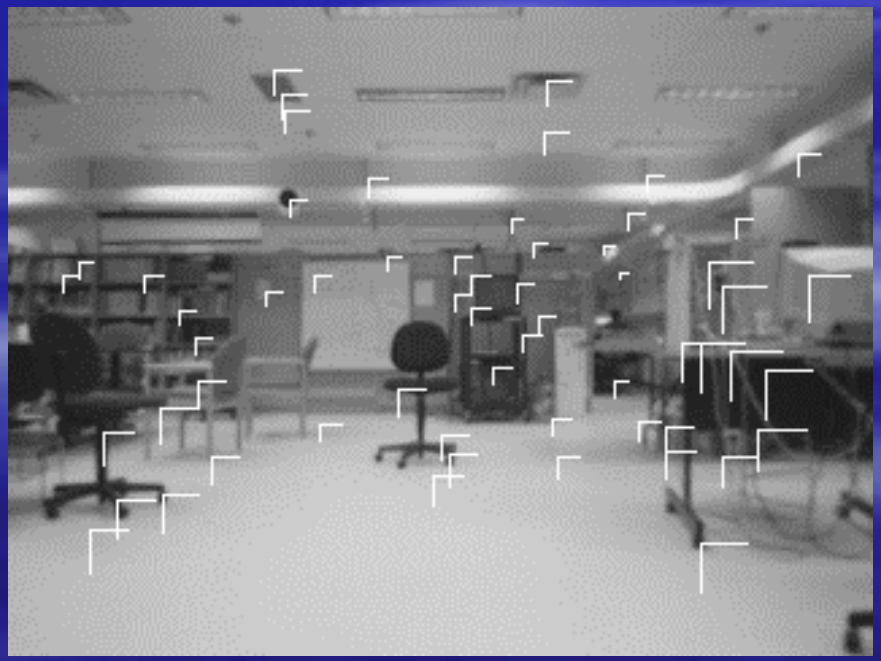
Gaussian smoothed

DOG

SIFT Stereo

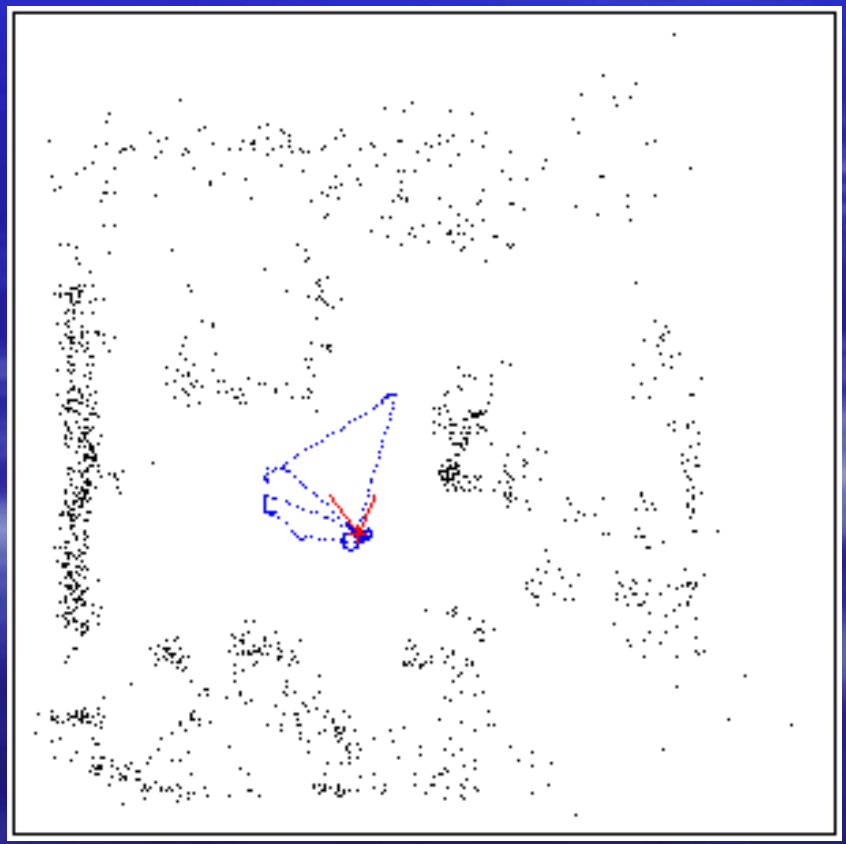
- Epipolar and disparity constraints
- SIFT scale and orientation constraints
- Robust : features must appear in all 3 images





3D Database Map

- 3D SIFT landmarks kept in database map
- Camera ego-motion estimation by least-squares minimization
- After 435 frames
- 2783 SIFT landmarks



Global Localization

- Kidnapped robot problem
- Recognize where the robot is relative to a previously built map
- Distinctive features required to identify scenes in the map
- SIFT features
  - use scale and orientation so far
  - local image characteristics provides feature specificity for object recognition
- Hough Transform and RANSAC approaches

Local Image Characteristics

- Measure local image gradient at a number of orientation relative to location, scale & orientation of feature
- Blur gradient locations to reduce sensitivity
- 2 x 2 grid with 4 orientation in each cell
- 16 element vector describing SIFT feature
- Scalable easily for more specificity

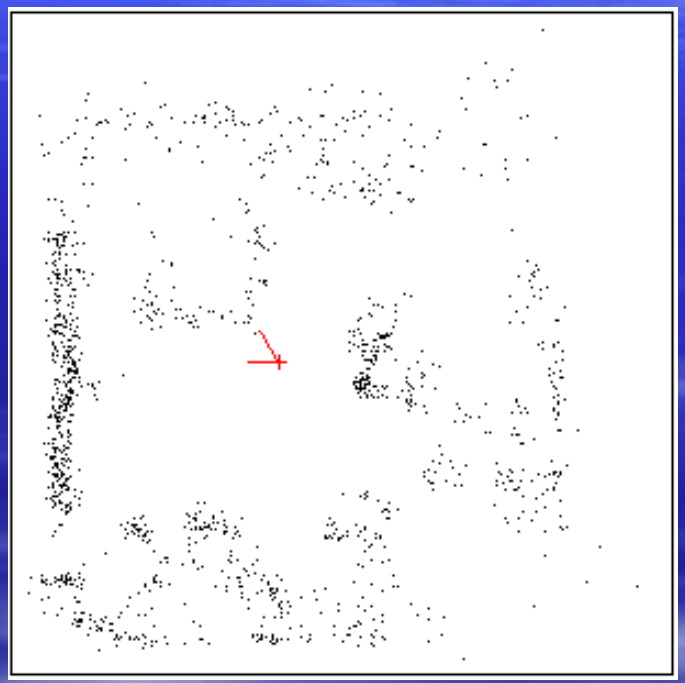
Hough Transform Approach

- Landmarks database & current SIFT features
- 3-D discretized search space (X,Z,θ)
- Local image characteristic & height to find a set of potential database landmarks for each feature
- For each potential match
  - vote all multiple poses covering an arc
  - also vote bins in ellipse region due to uncertainty
- Peaks : pose configuration with many matches
  - select top K peaks & carry out LS minimization
  - best pose : most matches with lowest LS error

RANSAC Approach

- RANdom SAmpling Consensus
- Find best database landmark for each feature, based on local image characteristic & height
- Repeat 50 times
  - randomly select 2 tentative matches
  - compute pose parameter (X,Z,θ)
  - check all tentative matches for support
- Select pose with most support
- LS minimization on all supporting matches
- Similar pose results as in Hough Transform

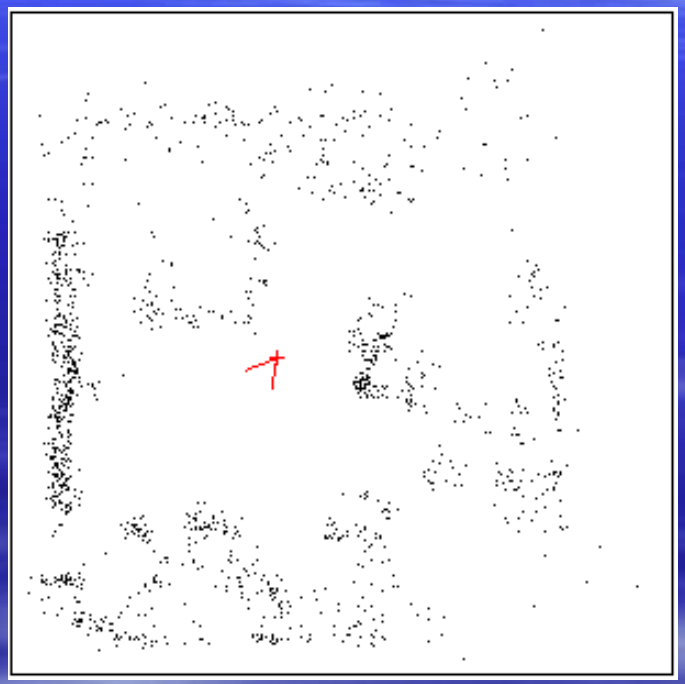
Experimental Results



Measured: (-10cm,120cm,-60deg)

Estimated: (-13.3cm,127.6cm,-60.5deg)

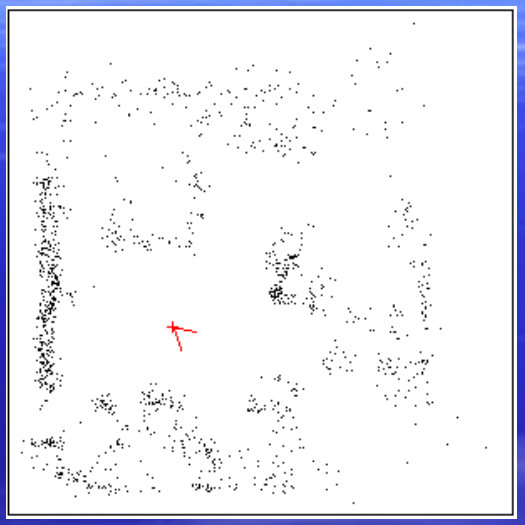
More Results

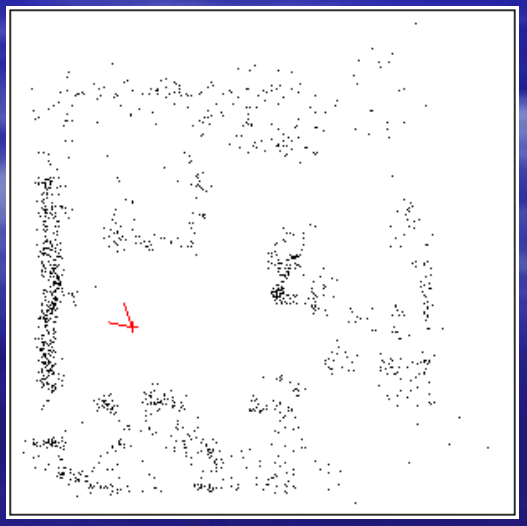


Measured: (-15cm,130cm,-140deg)

Estimated: (-16.0cm,134.9cm,-140.5deg)

More Results





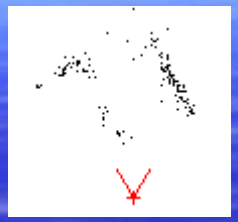
Discussion

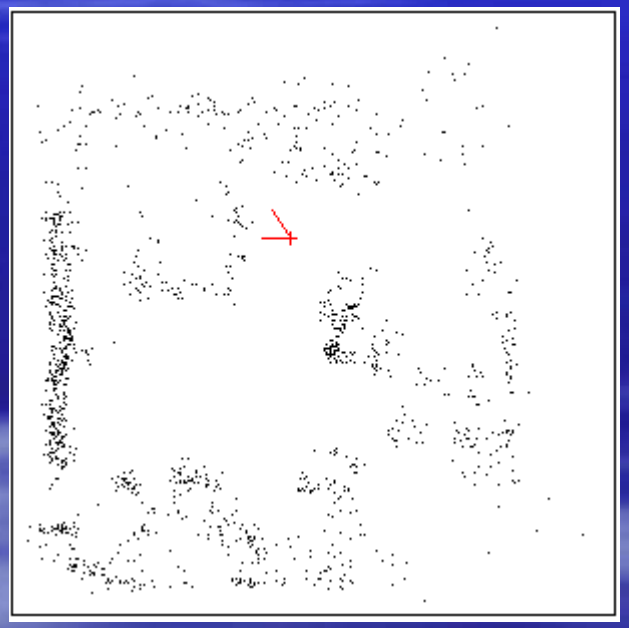
- Using SIFT features, both Hough Transform and RANSAC give good estimate
  - with RANSAC more efficient
- Computational cost increases linearly with database size
  - Non-specific features
    - consider all matches between features in current frame and all database landmarks
    - difficult to achieve from one frame as multiple robot poses possible
    - stochastic localization techniques

Map Alignment

- Global localization currently based on one frame
- Build a small submap of a local region from multiple frames
- Use RANSAC approach to align the small submap to the original map
- More robustness, particular in scenes with few SIFT landmarks
- Rotates from -15 degrees to 15 degrees

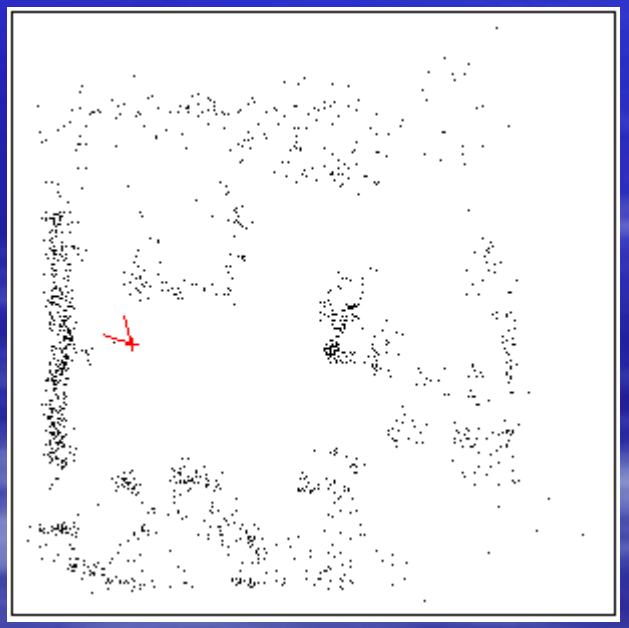
Results





Measured: (60,310,-65)

Estimated: (56.9,312.8,-63.5)



Measured: (-270,100,-45)

Estimated: (-259.7,101.8,-43.5)

Conclusion and Future Work

- Vision-based SLAM using 3D SIFT features
  - highly distinctive for global localization
- Global Localization
  - Hough Transform and RANSAC approaches
  - building submaps provides more robustness
  - maps re-used when robot starts at different positions
- Multi-robot collaboration
- Further experiments in larger environments
- Robot exploration strategies to build a good SIFT database map

Localization and Maps

- Visual landmarks using SIFT features form the basis of maps that can be constructed incrementally during robot exploration
- The maps support correction of robot odometry so the robot can maintain pose
- Localization can be accomplished both incrementally and globally, without prior information from odometry, so that the robot can initialize its pose and can survive large impulsive odometry errors
- Challenges: accommodating dynamic models of the environment

  substructures, changes in movement