SIFT SLAM Vision Details

MIT 16.412J Spring 2004

Vikash K. Mansinghka

Outline

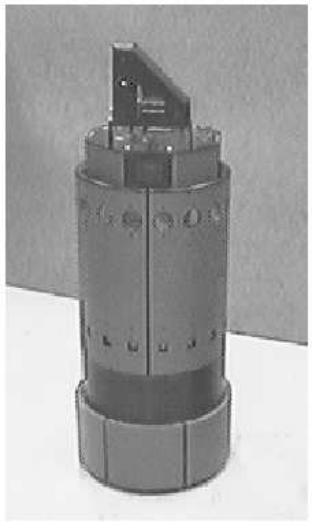
- Lightning Summary
- Black Box Model of SIFT SLAM Vision System
- Challenges in Computer Vision
- What these challenges mean for visual SLAM
- How SIFT extracts candidate landmarks
- How landmarks are tracked in SIFT SLAM
- Alternative vision-based SLAM systems
- Open questions

Lightning Summary

- Motivation: SLAM without modifying the environment
- Landmark candidates are extracted by the SIFT process
- Candidates matched between cameras to get 3D positions
- Candidates pruned according to consistency w/ robot's expectations
- Survivors sent off for statistical processing

Review of Robot Specifications

- Triclops 3-camera "stereo" vision system
- Odometry system which produces $[p, q, \delta]$
- Center camera is "reference"

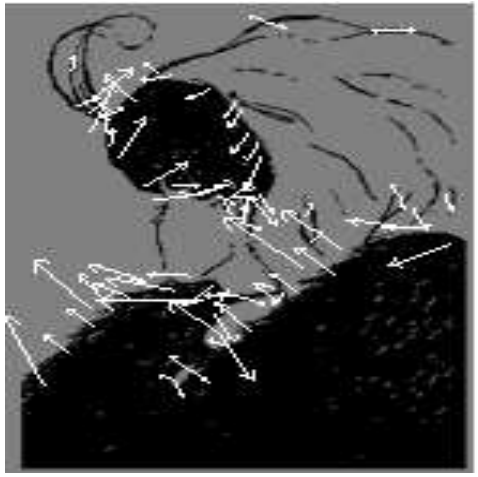


Black Box Model of Vision System

- For now, based on black-magic (SIFT). Produces landmarks.
- Assume landmarks globally indexed by i.
- Per frame inputs:
  - $[p, q, \delta]$ - odometry input (x, z, bearing deltas.)
  - List of (i, x_i) - new landmark pos (from SLAM)
- Per frame output is a list of (i, x'_i, x_i, r_i, c_i) for each visible landmark i where:
  - x'_i is its measured 3D pos (w.r.t. camera pos)
  - x_i is its map 3D pos (w.r.t. initial robot pos), if it isn't new
  - (r_i, c_i) is its pixel coordinates in center camera

Challenges in Computer Vision

- Intuitively appealing $\neq$ computationally realizable
- Stable feature extraction is hard; results rarely general
- Extracted features are sparse
- Matching requires exponential time
- Matches are often wrong



Implications for Visual SLAM

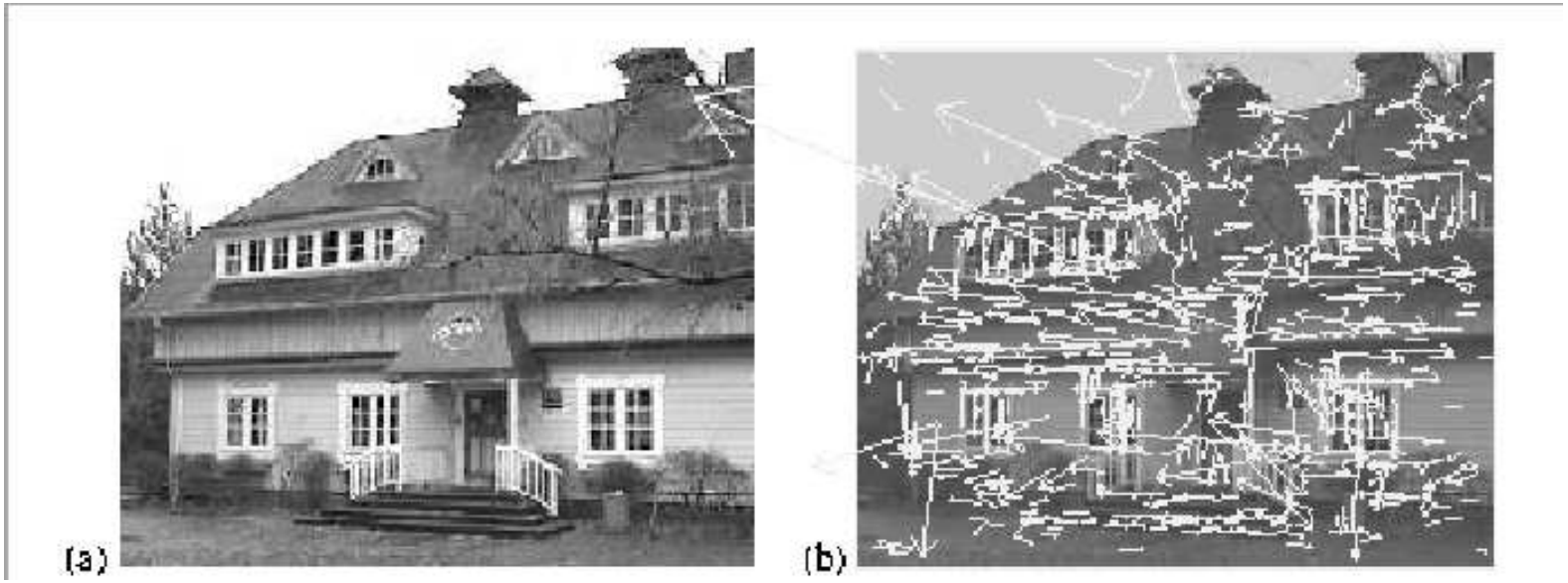
- Hard to reliably find landmarks
- **Really Hard to reliably find landmarks**
- **Really Really Hard to reliably find landmarks**
- Data association is slow and unreliable
- False matches introduce substantial errors
- Accurate probabilistic models unavailable

Remarks on SIFT approach

- For visual SLAM, landmarks must be identifiable across:
  - Large changes in distance
  - Small changes in view direction
  - (Bonus) Changes in illumination
- Solution:
  - Produce "scale-invariant" image representation
  - Extract points with associated scale information
  - Use matcher empirically capable of handling small displacements

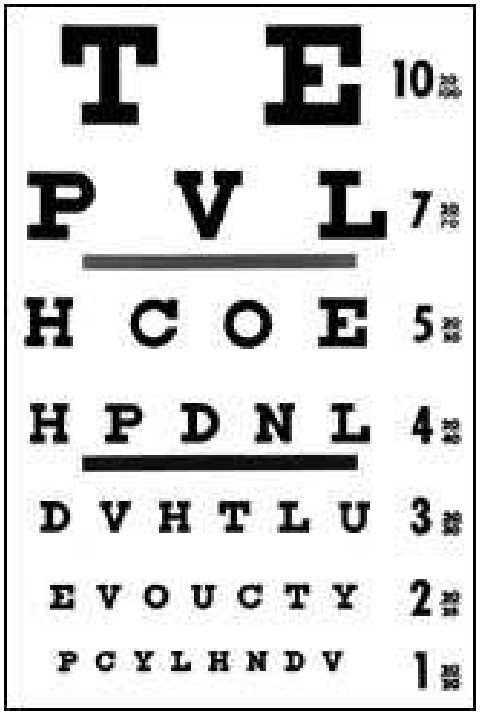
The Scale-Invariant Feature Transform

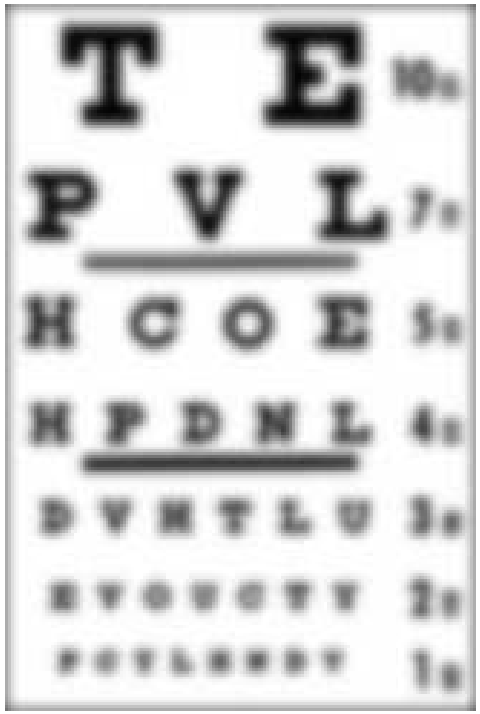
- Described in Lowe, IJCV 2004 (preprint; use Google)
- Four stages:
  - Scale-space extrema extraction
  - Keypoint pruning and localization (not used in SLAM)
  - Orientation assignment
  - Keypoint descriptor (not used in SLAM)

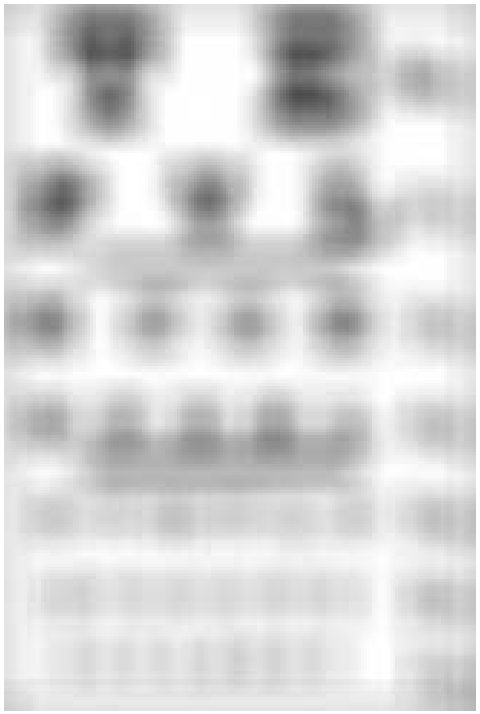


Lightning Introduction to Scale Space

- Motivation:
  - Objects can be recognized at many levels of detail
  - Large distances correspond to low l.o.d.
  - Different kinds of information are available at each level

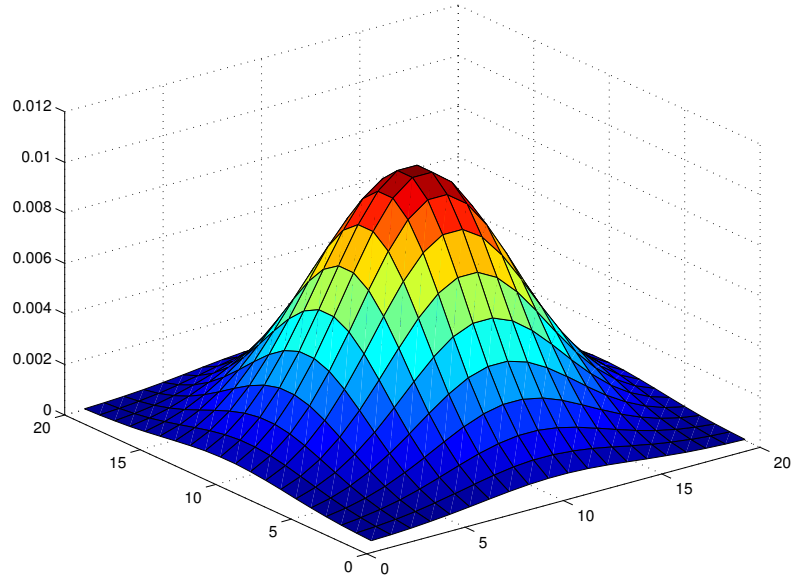






Lightning Introduction to Scale Space

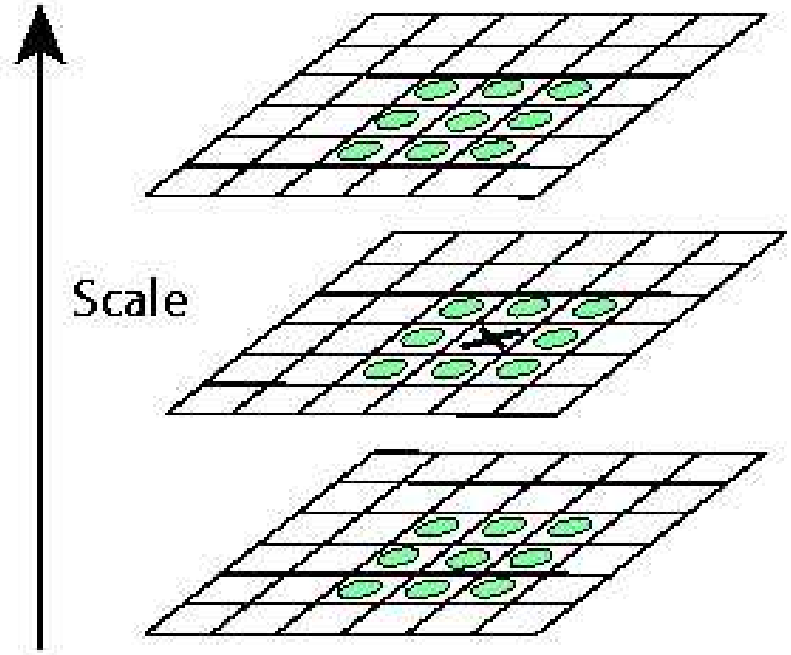
- Idea: Extract information content from an image at each l.o.d.
- Detail reduction typically done by Gaussian blurring
- Long history in both machine and human vision
  - Marr in late 1970s
  - Henkel in 2000
- Analogous concepts used in speech processing



Scale Space in SIFT

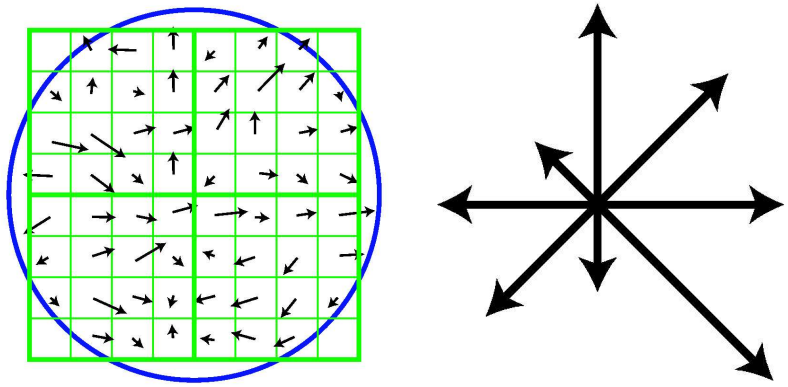
- $I(x, y)$ is input image. $L(x, y, \sigma)$ is rep. at scale σ.
- $G(x, y, \sigma)$ is 2D Gaussian with variance σ^2
- $L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$ ("only" choice; see Koenderink 1984)
- $D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma)$
- D approximates $\sigma^2 \nabla^2 G * I$ (see Mikolajczyk 2002 for significance)
- D also edge-detector-like; newest SIFT "corrects" for this
- Details of discretization (e.g. resampling, k choice) unimportant

Scale Space in SIFT



- Compute local extrema of D as above
- Each such (x, y, σ) is a feature
- (x, y) part "should" be scale and planar rotation invariant

SIFT Orientation Assignment



- For each feature (x, y, σ):
  - Find fixed-pixel-area patch in $L(x, y, \sigma)$ around (x, y)
  - Compute gradient histogram; call this b_i
  - For b_i within 80% of max, make feature (x, y, σ, b_i)
- Enables matching by including illumination-invariant feature content (Sinha 2000)

SIFT Stereopsis

- Apply SIFT to image from each camera.
- Match center feature (x, y, σ, θ) and right feature $(x', y', \sigma', \theta')$ if:
  1. $|y - y'| \leq 1$
  2. $0 < |x' - x| \leq 20$
  3. $|\theta - \theta'| \leq 20$ degrees
  4. $\frac{2}{3} \leq \frac{\sigma}{\sigma'} \leq \frac{3}{2}$
  5. No other matches consistent with above exist
- Match similarly for left and top; discard all not matched twice
- Compute 3D positions (trig) as average from horiz. and vert.

Sample SIFT Stereo Output

Recapitulation (in G Minor)

- Procedure so far:
  1. For each image:
     (a) Produce scale-space representation
     (b) Find extrema
     (c) Compute gradient orientation histograms
  2. Match features from center to right and center to top
  3. Compute relative 3D positions for survivors
- This gives us potential features from a given frame
- How do we use them?

Where's Waldo?

Sadly, SIFT failed. :(

Landmark Tracking

- Predict where landmarks should appear (reliability, speed)
- Note: Robot moves in xz plane
- Given $[p, q, \delta]$ and old relative position $[X, Y, Z]$, find expected position $[X', Y', Z']$ by:

  $X' = (X - p)cos(\delta) - (Z - q)sin(\delta)$

  $Y' = Y$

  $Z' = (X - p)sin(\delta) - (Z - q)cos(\delta)$
- By pinhole camera model ((u_0, v_0) image center coords, I interocular distance, f focal length):

  $r' = v_0 - f\frac{Y'}{Z'}$

  $c' = u_0 + f\frac{X'}{Z'}$

  $d' = f\frac{I}{Z'}$

  $\sigma' = \sigma\frac{Z}{Z'}$

Landmark Tracking

- V is camera field of view angle (60 degrees)
- A landmark is expected to be in view if:

  $Z' > 0$

  $tan^{-1}(\frac{|X'|}{Z'}) < \frac{V}{2}$

  $tan^{-1}(\frac{|Y'|}{Z'}) < \frac{V}{2}$
- An expected landmark matches an observed landmark if:
  - Obs. center within a 10x10 region around expected
  - Obs. scale within 20% of expected
  - Obs. orientation within 20 degrees of expected
  - Obs. disparity within 20% of expected

Sample Landmark Matching Results

Landmark Tracking

- A SIFT view is: (SIFT feature, relative 3D pos, absolute view dir)
- Each landmark is: (3D position, list of views, misses)
- Algorithm:

```
For each frame, find expected landmarks w/ odometry
For each observed view v:
  If v matches an expected landmark l:
    Set l.misses = 0
    Add v to view list for l
  Else add l to DB
For each expected, unobserved landmark l:
  If one view direction within 20 degrees of current:
    l.misses++
    If l.misses >= 20, delete l from DB
```

Other Examples of Vision-based SLAM

- Ceiling lamp based (Panzieri et al. 2003)
- Bearing-only visual SLAM (lots; unclear to me)
- Monocular visual SLAM, e.g. by optical flow (lots; unclear to me)
- Shi and Tomasi feature based (Davison and Murray 1998)

Open Questions

- How to speed vision processing? (Move to hardware?)
- Should full SLAM (not decorrelated SLAM) be used?
- If full SLAM, how can numerics be sped up? (FMM?)
- Can thresholds be automatically tuned? (Maximum information?)
- Will movement confuse SIFT SLAM? (Fix by optical flow?)
- What environments foil SIFT SLAM?
- How to get dense features? (Geometric model fitting?)
- Can this handle large environments/long trajectories? (Fix by qualitative navigation?)
- Why does David Lowe rock so hard?

Acknowledgements

- Technical content primarily from Lowe 2004 and Se, Lowe, Little 2002.
- Various images liberated from their papers and from the internet.
- Others produced and/or processed by MATLAB, Lowe's SIFT demo and ImageMagick.
- My sincerest apologies to Terry Pratchett.