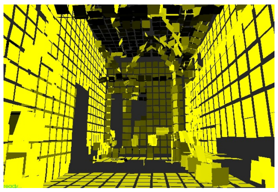
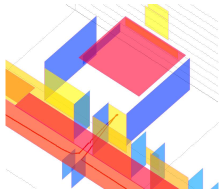
Sensing: using features

Lines and planes



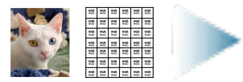


Many slides adapted from slides © R. Siegwart, Steve Seitz, J. Tim Oates, David Scaramuzza, Chris Clark

1

What's a Feature?

- Raw data is cumbersome, huge, and redundant



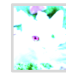
In practice, these are millions × millions

Video cameras produce 30/second

- Don't need every pixel or reading
  - But what data do you need?

Cat

Still a cat



Cat? Pig? Rose?

- Getting some subset is feature extraction
  - "Features" can be combinations of traits, the result of math operations, and many other things

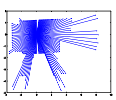
en.wikipedia.org/wiki/Heterochromia_iridum#/media/File:June_odd-eyed-cat_cropped.jpg

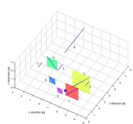
2

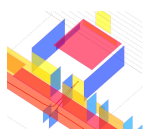
2

Motivation

- Raw perceptual data: huge to process, store*
  - Compact features require less storage (e.g. lines, planes)
- Provides rich and accurate information
  - For some tasks!
- Basis for high level features (e.g. more abstract features, objects)



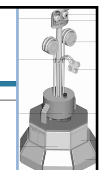




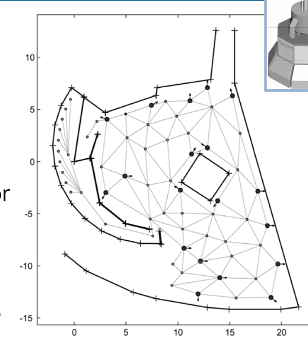
3

3

Environment Mapping



- Features for Localization
  - Compact map 26 bytes / m2
  - Multi-hypothesis tracking
- Topological map for global planning
- Raw data for local planning and obstacle avoidance



4

4

Line Extraction: Motivation

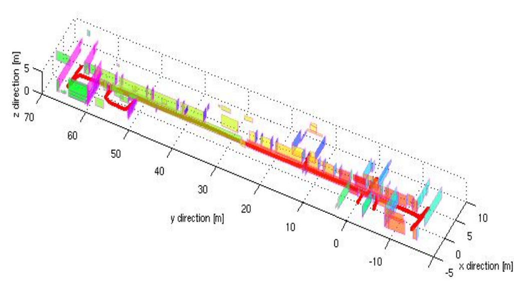
- Map of the ASL hallway built using line segments

7

7

Plane Extraction: Motivation

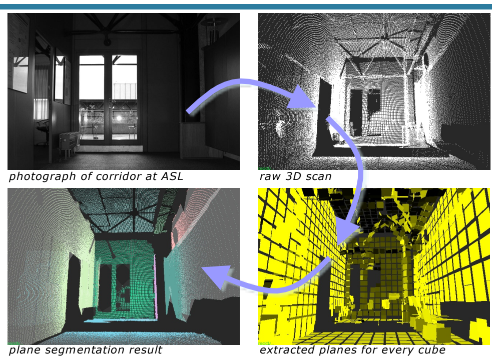
- Map of the ASL hallway built using orthogonal planes constructed from line segments



8

8

Example Result



9

Line Extraction

- Algorithms:
  - Split and merge
  - Linear regression
  - RANSAC
  - Hough-Transform

10

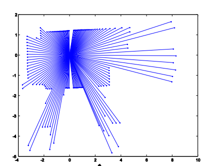
Line Extraction

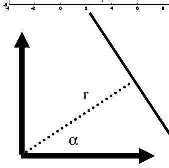
- Raw data: any depth sensor
- In practice, mostly laser range finders
  - Dense and accurate range measurements
  - High sampling rate, high angular resolution
  - Good range distance and resolution.
- Why line segments?
  - The simplest geometric primitive
  - Compact, requires almost no storage
  - Provides rich and accurate information
  - Matches indoor human environments, e.g., offices

11

Line Extraction: The Problem

- Scan point in polar form: (ϱ_i, θ_i)
- Assumptions:
  - Gaussian noise* in $[0, \sigma]$ for distance measurement ϱ
  - Negligible angular uncertainty
- Line model in polar form:
  - $x \cos \alpha + y \sin \alpha = r$
  - $-\pi < \alpha \leq \pi$
  - $r \geq 0$





* Values that noise can take on are distributed on a normal curve

12

Line Extraction: The Problem

- Three main problems:
- How many lines should we find?
- Which points belong to what line?
  - This problem is called SEGMENTATION
- Given points that belong to a line, how to estimate parameters?
  - This problem is called LINE FITTING

13

Split-and-Merge

- The most popular algorithm
- Originated from computer vision.
- A recursive procedure of fitting and splitting.
- A slightly different version, called Iterative-End-Point-Fit, simply connects the end points for line fitting.

14

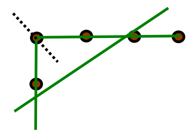
Algorithm 1: Split-and-Merge

Algorithm: Split-and-Merge

Setup
- Initialize set **S** to contain all points
- Set some distance threshold **t**

Split
- Fit a line to points in current set **S**
- Find the most distant point from the line
- If distance < **t**, repeat with left set S_1 and right set S_2

Merge
- If two consecutive segments are close/colinear enough, obtain the common line and find the most distant point
- If most distant point ≤ **t**, merge segments

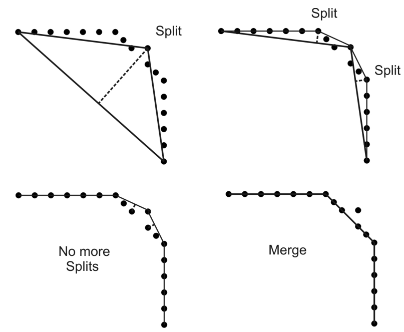


www.cs.princeton.edu/courses/archive/fall11/cos495/COS495-Lecture11-LineExtraction.pdf

15

15

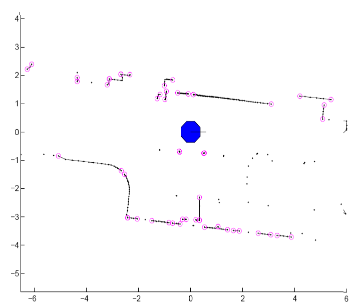
Split-and-Merge



16

16

Split-and-Merge Example



17

17

RANSAC Algorithm

- Random Sample Consensus ("Ran-sack")
- General, robust algorithm to fit models in the presence of outliers
- Good tool when goal is to identify points that satisfy a mathematical model or function (like a line or plane)
- Typical applications in robotics
  - Line extraction from 2D range data
  - Plane extraction from 3D range data
  - Structure from motion

18

18

RANSAC Algorithm

- RANSAC is an iterative method
- Drawback: A nondeterministic method, so results are different between runs
- Probability to find a line without outliers increases as more iterations are used

19

19

Algorithm 3: RANSAC

Algorithm 4: *RANSAC*

1. Initial: let A be a set of N points
2. **repeat**
3. Randomly select a sample of 2 points from A
4. Fit a line through the 2 points
5. Compute the distances of all other points to this line
6. Construct the inlier set (i.e. count the number of points with distance to the line $< d$)
7. Store these inliers
8. **until** Maximum number of iterations k reached
9. The set with the maximum number of inliers is chosen as a solution to the problem

20

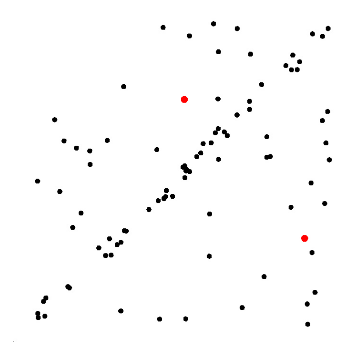
20

RANSAC

21

21

RANSAC

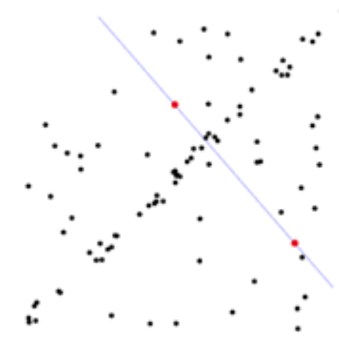


- **Select sample of 2 points at random**

22

22

RANSAC

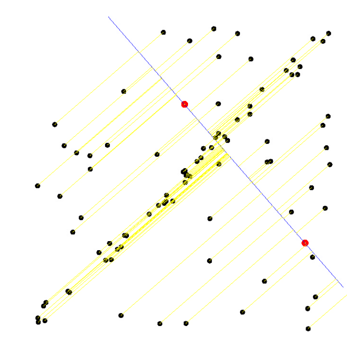


- Select sample of 2 points at random
- **Calculate model parameters that fit the data in the sample**

23

23

RANSAC

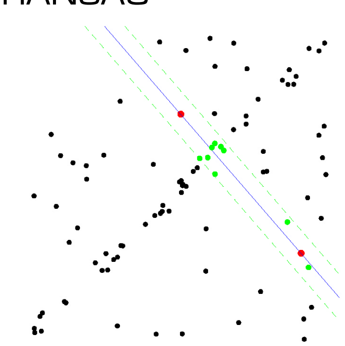


- Select sample of 2 points at random
- Calculate model parameters that fit the data in the sample
- **Calculate error function for each data point**

24

24

RANSAC

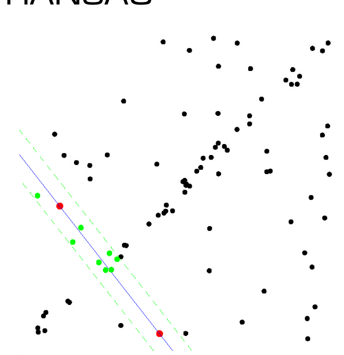


- Select sample of 2 points at random
- Calculate model parameters that fit the data in the sample
- Calculate error function for each data point
- **Select data that support current hypothesis**

25

25

RANSAC

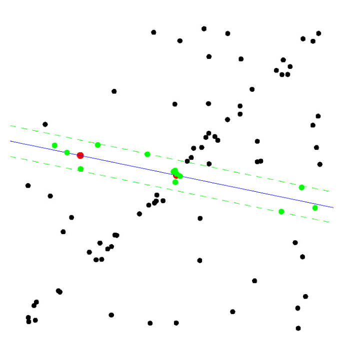


- Select sample of 2 points at random
- Calculate model parameters that fit the data in the sample
- Calculate error function for each data point
- Select data that support current hypothesis
- **Repeat sampling**

26

26

RANSAC

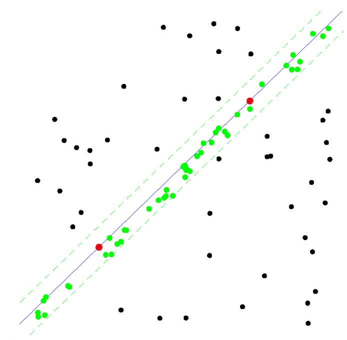


- Select sample of 2 points at random
- Calculate model parameters that fit the data in the sample
- Calculate error function for each data point
- Select data that support current hypothesis
- **Repeat sampling**

27

27

RANSAC



ALL-INLIER SAMPLE

28

28

How Many Iterations?

- How many iterations does RANSAC need?
- Can't know in advance if observed set contains maximum number of inliers
- Ideal: check all possible combinations of 2 points
- N(N-1)/2 (for a line) – infeasible if N is too large
- Do not need to check all combinations – just a subset if we have a rough estimate of the percentage of inliers in our dataset
- This can be done in a probabilistic way

29

29

RANSAC Iterations

- Let w be fraction of inliners: w = number of inliers / N
  - N is the total number of points.
  - w is also the probability of selecting an inlier
- p = probability of finding a set of points without outliers
- w^2: probability that both points are inliers
- $1-w^2$: probability that at least one of these two points is an outlier

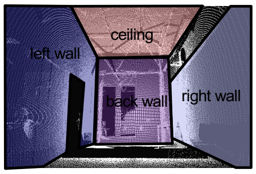
$$\boxed{k = \frac{\log(1-p)}{\log(1-w^2)}}$$

30

30

Extraction of Planar Features

- Goal: extract planar features from a dense point cloud



31

31

RANSAC for Planes

- For every cell, use Ransac to segment the best

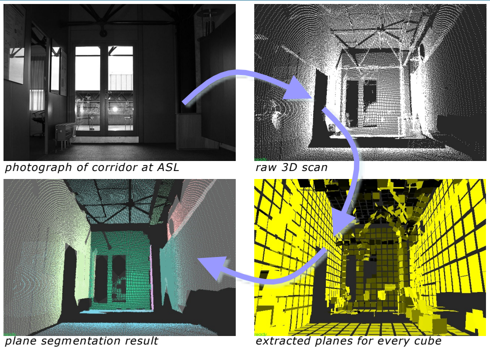
pseudo code

```
threshold = 0.05; // predefined threshold in [m]
m = 100; // predefined the number of ransac iterations

for every cell ci do
        npmax = 0; // initialize maximum number of found closest points

        for m ransac-iterations do
                randomly choose 3 different points p1,p2,p3 from points
                contained in cell ci;

                calculate plane p, i.e. normal n and orth. distance to
                origin d defined by p1,p2,p3;

                np = 0;

                for all points pj in ci do
                        calculate orthogonal distance s from pj to p;
                        if (s < threshold)
                                np++;
                        endif
                endfor

                // maximize the number of points close to the plane
                if (np > npmax)
                        npmax = np; // the number of points close to the plane
                        pbest = p; // this is the found best plane
                endif

        endfor
endfor
```

33

33

Example Result



photograph of corridor at ASL

raw 3D scan

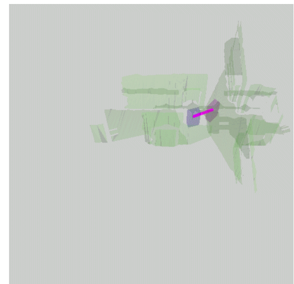
plane segmentation result

extracted planes for every cube

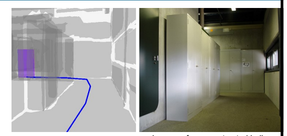
34

34

Probabilistic Feature-Based 3D SLAM



The same experiment as before but this time planar segments are visualized and integrated into the estimation process

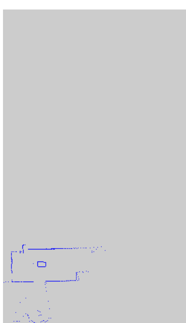


close-up of a reconstructed hallway

close-up of reconstructed bookshelves

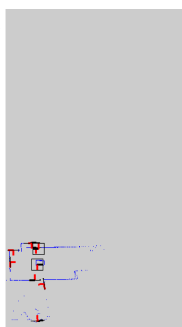
35

Experimental Results



estimation not using planar features

estimation using planar features depicted by red coordinate frames



36

36