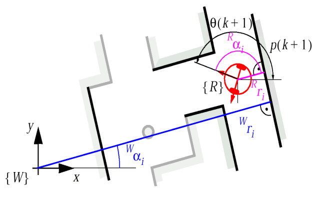
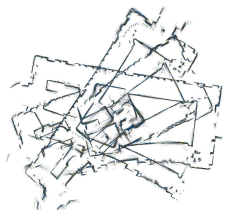
Mapping and SLAM: A (High-Level) Survey

Sebastian Thrun, 2002
Original presentation: David Black-Schaffer, Kristof Richmond
Used with many, many thanks to authors

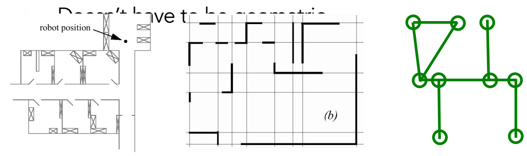




1

What is Mapping?

- Building a **model of the robot's environment**
  - Based on sensor data obtained in that environment
- Many different environment representations exist
- Mapping means constructing one
  - Based on sensor data
  - Doesn't have to be geometric

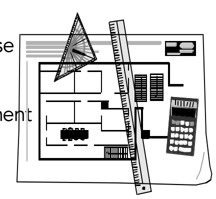


2

2

How to Get a Map?

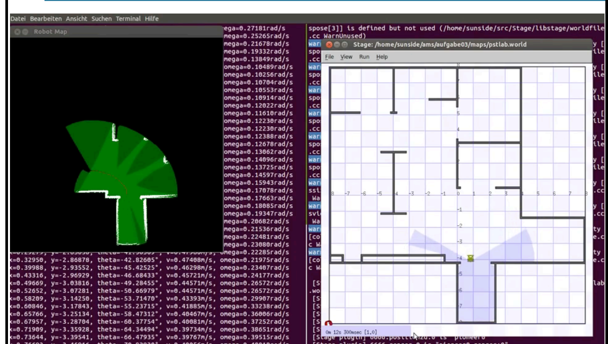
- By hand?
  - Slow, expensive and imprecise
- Automatically: Mapping
  - The robot learns its environment
- Can cope with dynamically changing environment
- Map is built from sensors that will be used to navigate



3

3

Basic Mapping Task



https://www.youtube.com/watch?v=eAbF3QBGwzA

4

4

Historical Overview

- Metric Mapping
  - Geometric representations
  - Occupancy Grids
  - Larger maps much more computationally intensive
- Topological Mapping
  - Milestones with connections
  - Require navigation information
  - Hard to scale

5

5

The Problems

- Measurement noise
  - Sensor and position noise are not independent
- Map size
  - High resolution maps can be very large
- Correspondence
  - Do multiple measurements at different times correspond to the same object?
  - Loop closure is a subset of this
- Dynamic environments
  - Many algorithms assume a static environment

6

6

Basic Mapping Requirements

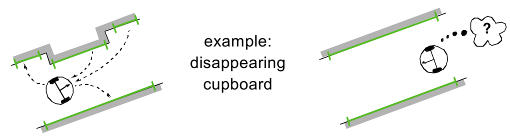
- A way to incorporate sensed information into world model
- Sufficient information for navigation
  - Estimating position, path planning, obstacle avoidance, …
- Correctness
- Predictability
  - Most environments are a mixture of predictable and unpredictable features

7

7

Challenges: Maintenance

- Map Maintenance: keeping track of changes in the environment



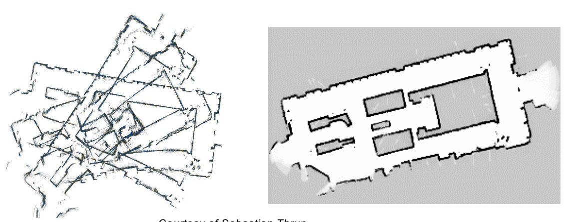
- Often uses a measure of belief of each feature
- If localizing: did the world change, or are you lost?

12

12

Challenges: Cyclic Environments

- Small local error accumulates
- This is usually irrelevant for navigation
- When closing loops, global error **does** matter



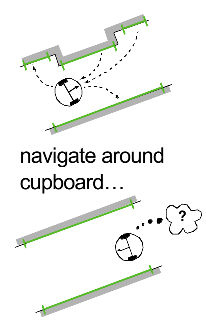
Courtesy of Sebastian Thrun

13

13

Dynamic Environments

- Dynamic changes require continuous (re-)mapping
- High-level features help if you can get them
  - Humans are more dynamic map features than walls
  - Can also learn that certain things are dynamic over time



14

14

Current State of Mapping

- Algorithms are:
  - Robust for static, structured, and small environments
  - Probabilistic
  - Only ok with correspondence problem
  - Incremental **or** multi-pass
    - Incremental algorithms only need one "pass" through data, and can possibly be run in real time
- Continuing areas
  - Dynamic environments
  - Semantic labeling of environments
  - Planning exploration paths of unknown environments

15

15

Why Probabilistic Mapping?

- Noise in commands and sensors
  - Commands are not executed exactly (e.g., slippage leads to odometry errors)
  - Sensors noise due to real world (e.g., angle of incidence and scattering)

basically everything we've ever covered about error

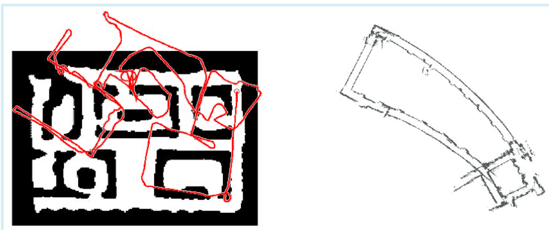
- Noise is not statistically independent
  - Control errors accumulate with time/distance
  - Can't "average out" noise

16

16

Command Noise

- Odometry Errors: heading and distance measurements accumulate errors with time



17

17

SLAM

- SLAM
  - **S**imultaneous **L**ocalization **A**nd **M**apping
  - Figure out where we are and what our world looks like at the same time
- Localization
  - Where are we?
  - Position error accumulates with movement
- Mapping
  - What does the environment look like?
  - Sensor error (not independent of position error)

19

19

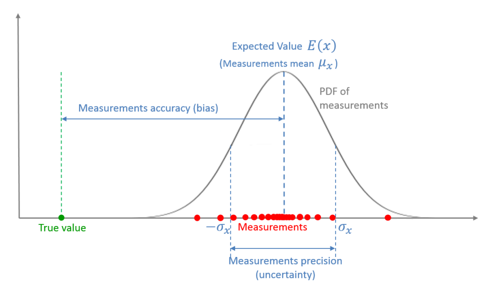
Mapping ≠ SLAM?

- Can you do mapping without localization?
  - Sort of.
- Gathering environment data doesn't depend on knowing where you are, *but…*
- If you don't track robot's location, the **relative position** of different sensor readings is harder to calculate
- Some problems (e.g., loop closure) may be impossible

20

20

What are we trying to do?

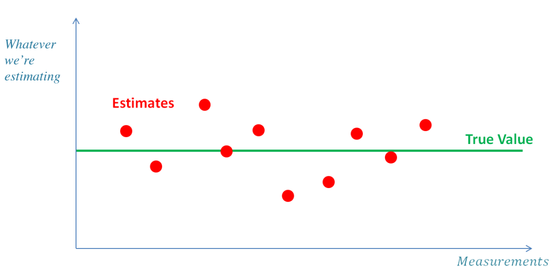


www.kalmanfilter.net

22

22

What are we trying to do?



www.kalmanfilter.net

23

23

Bayes Rule

- $p(x \mid d) = \eta\, p(d \mid x)\, p(x)$*
  - $p(x \mid d)$ is the probability of (the map) x being true given the (sensor) measurement d
  - $p(d \mid x)$ is the probabeingbility of the (sensor) measurement being d given (an object at) x
  - $p(x)$ is the prior probability (of the map)

* η = *normalization constant*

24

24

Bayes Rule over Time

- Notation
  - s = pose of robot (x, y, Θ)
  - u = command given to robot
  - z = sensor measurement
  - m = map
- All are functions of time
  - z_t = sensor measurements at time t
  - z^t = all sensor measurements up to time t
  - (same for s, u, and m)

25

25

Bayes Filter

s: robot configuration (pose)
u: command
z: sensor reading
m: map
$\square_t/\square^t$: value at/up to time t

$$p(s_t, m \mid z^t, u^t) = \eta \cdot p(z_t \mid s_t, m) \int p(s_t \mid u_t, s_{t-1}) p(s_{t-1}, m \mid z^{t-1}, u^{t-1}) ds_{t-1}$$

- Probability of a location (pose) and a map, given sensor readings and command.
- $p(z_t \mid s_t, m)$ is the **sensor model**
- $p(s_t \mid u_t, s_{t-1})$ is the **motion model**
- $p(s_{t-1}, m \mid z^{t-1}, u^{t-1})$ is probability we were **actually** where we thought we were last timestep

Sensor and motion models are usually static

26

26

Mapping Methods

	Kalman	EM	Occupancy Grids	Dogma
Representation	Landmark Locations	Point Obstacles	Occupancy Grids	Occupancy Grids
Incremental	YES	NO	YES	NO
Requires Poses	NO	NO	YES	YES
Handles Correspondence	NO	YES	YES	YES
Dynamic Environments	limited	NO	limited	YES

27

27

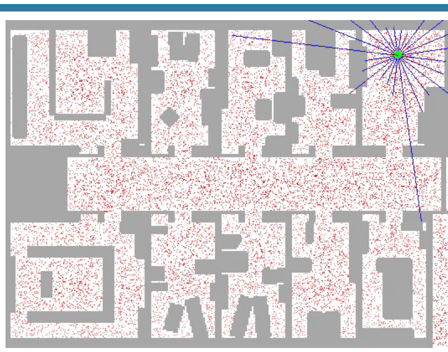
Monte Carlo Localization

- Probabilistic
  1. Start with a uniform distribution of possible poses (x, y, Θ)
  2. Compute the probability of each pose given current sensor data and a map
  3. Normalize probabilities
     - Throw out low probability points
     - Blur current points (we never know exactly where we are)
- Performance: excellent!
  - Need non-symmetric geometries

28

28

Monte Carlo Localization

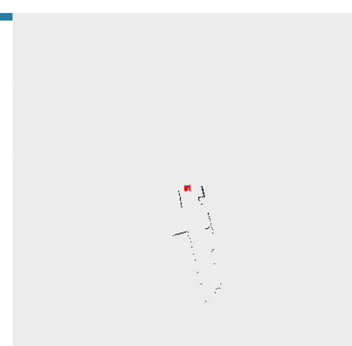


Thrun, Sebastian. "Animation of Monte Carlo Localization using laser range finders"

29

29

Monte Carlo SLAM



Thrun, Sebastian. "Animation: Online mapping with Monte Carlo Localization."

30

30

Expectation Maximization (EM)

- Find most likely map (and poses)

1. **E**xpectation step:
   - Calculate probabilities of robot poses for current guess of map
2. **M**aximization step:
   - Calculate single most likely map for distribution of robot poses

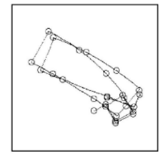
- Iterate

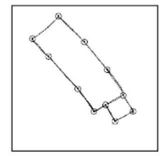
31

31

EM Performance

- Pros:
  - Resolves correspondences
- Cons:
  - Non-incremental
  - No posterior probabilities for map
  - Slow
  - Greedy
- Improvements: Hybrid approache
  - Incremental computation
  - Maintain a few possible robot poses



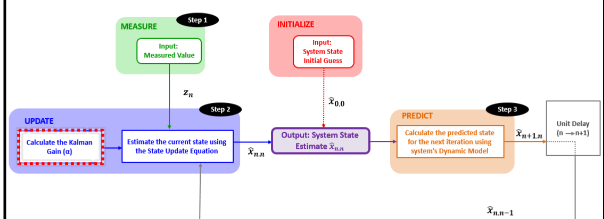


32

32

Kalman filters

- Prediction algorithm that handles uncertainty
  - Process (actuation) uncertainty
  - Measurement (sensor) uncertainty



33

33

Kalman filter (SLAM)

- Core idea: **predict** where we should be, take readings, then update our **estimate** of where we are
- We need to take prior states into consideration
  - ALL of them?!
- But, we can condition on the previous state
  - Which is conditioned on the state before that, …

The estimate of the current state = Predicted value of the current state + Factor x (Measurement − Predicted value of the current state)

34

34

KF core idea (2)

1. Position prediction:
   - Position at time step k+1 is predicted based on old location (k) and its movement
2. Observe: sense the new location at k+1
3. Measurement prediction:
   - Use the predicted robot position and the map to generate multiple **predicted observations**
   - Given where we might be, what do we expect to see?
4. Matching: Map from all predicted observations to all observations
5. Estimate location: keep best *n* and continue

35

35

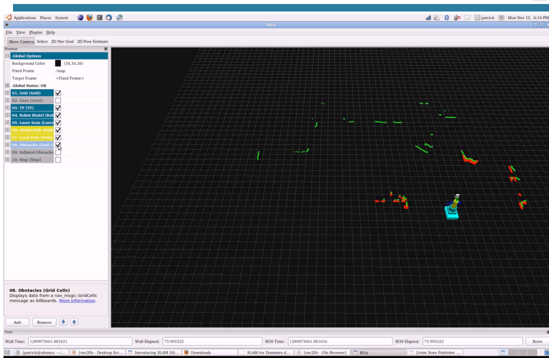
Kalman Filter Performance

- Pros:
  - Full (Gaussian) posterior probabilities
  - Incremental
  - Good convergence
- Cons:
  - Limited model
  - Correspondence problem
  - Limited map size
- Many improvements exist!

38

38

Particle Filter SLAM



39 https://www.youtube.com/watch?v=khSrWtB0Xik

39

Occupancy Grids

- Impose grid on space to be mapped
- Find *inverse sensor model*

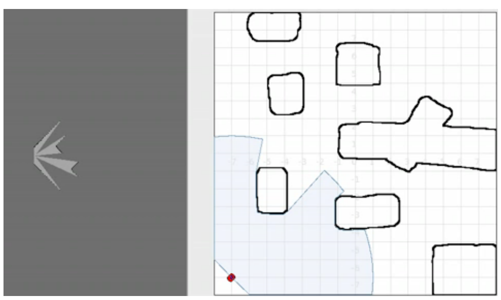
$$p(m_{x,y}|z_t,s_t)$$

- Update odds that grid cells are occupied

41

41

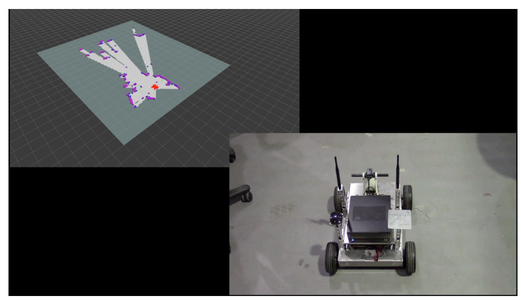
Occupancy Grid (simple)



42 https://www.youtube.com/watch?v=zOj4s9TEmg8

42

Occupancy Grid (real)



43 https://www.youtube.com/watch?v=iD47JWVqTCk

43

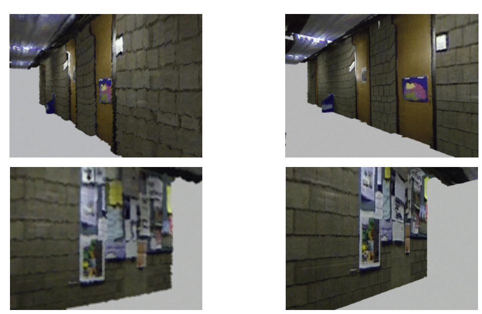
Occupancy Grids

- Pros
  - Simple
  - Accurate
  - Incremental
- Cons
  - Require known poses
  - Independence assumptions
- Extensions: Object Maps
  - Reduced memory requirements
  - Better for dynamic environments
  - Limited by available object models

46

46

Object Maps

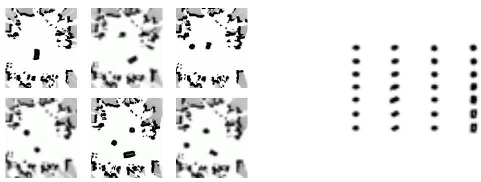


47

47

Dynamic Environments

- Kalman filters
- Decaying occupancy grids
- Dogma
  - **D**ynamic **o**ccupancy **g**rid **m**apping **a**lgorithm

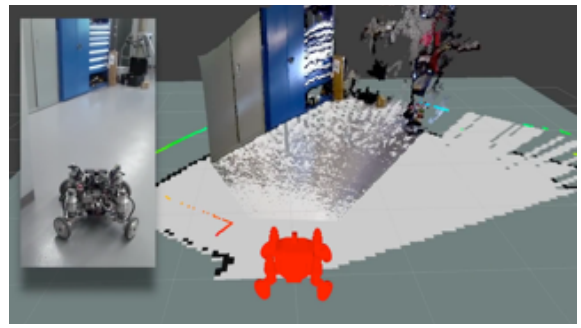


48

48

A Real Example

Using a Laser and an RGB-D Sensor



https://www.youtube.com/watch?v=o1GSQanY-Do

49

49

Mapping

- Turning percepts (sensor readings) into a model of an environment (a map)
  - Maps come in many forms
  - Must have sufficient information for navigation tasks
  - Estimating position, path planning, obstacle avoidance, …
- Many challenges
  - Difficult environments: cycles, dynamism, …
  - Sensor noise and precision
  - Actuator noise
  - Labeling environment

50

50

Approaches

- Many algorithms exist
- All modern approaches are probabilistic
- Many are particle filter based
- Usually SLAM is involved
  - Doesn't strictly have to be
- Different approaches address different challenges

51

51