Topological map merging

Kris Beevers

Algorithmic Robotics Laboratory

Department of Computer Science

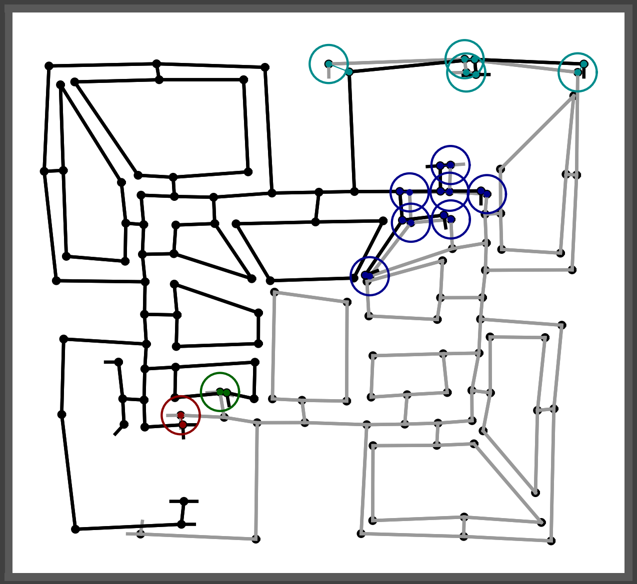
Rensselaer Polytechnic Institute

`beevek@cs.rpi.edu`

March 9, 2005

Motivation

- Multiple robots create topological maps individually
- From these, we want a single, consistent global map
- In order to do this, we must find correspondences between the individual maps
- Once we have correspondences, it is easy to merge the maps

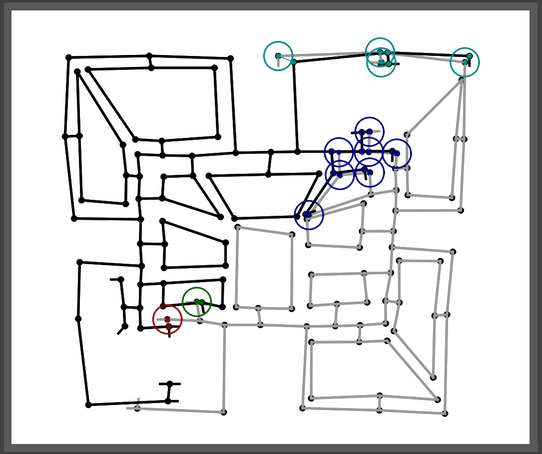


Overview

1. Problem setup
2. Previous work
3. Inspirations: graph matching, image registration
4. The algorithm
   - ↪ Structural phase
   - ↪ Geometrical phase
   - ↪ An example
5. Experimental results
6. Extensions

The problem

- Given two topological maps represented as graphs, embeddable in $\mathcal{R}^n$: $\mathcal{A} = (V_\mathcal{A}, E_\mathcal{A})$, $\mathcal{B} = (V_\mathcal{B}, E_\mathcal{B})$
- Goal: find correspondences that match a subset of $V_\mathcal{A}$ to a subset of $V_\mathcal{B}$, and a subset of $E_\mathcal{A}$ to a subset of $E_\mathcal{B}$
- Correspondences: places where the maps overlap
- Maps may overlap in multiple disjoint places — to get a consistent global map, we want all correspondences!



Previous work

- Most multi-robot mapping work assumes the robots share a common reference frame

- One exception: Ko et al. (2003) — robots exchange occupancy maps, localize with particle filters

- Closely related to our work: Dedeoglu and Sukhatme (2000):
  - ↪ Finding correspondences between landmark-based maps
  - ↪ No common reference frame
  - ↪ Estimate transformation based on a single-vertex match
  - ↪ Use simple heuristics to find correspondences

Idea #1: graph matching

- Our problem is a lot like the Maximal Common Subgraph problem: find largest set of compatible vertex/edge pairings
- General formulation of MCS: NP-hard
- But! We can do it in polynomial time:

  ↪ Edges at vertices have spatial interrelationships → linear number of edge pairings (instead of exponential)
- Fine, but what about disconnected subgraphs?

Idea #2: image registration

- Our problem is also a lot like image registration: find matching between feature points and compute a transformation

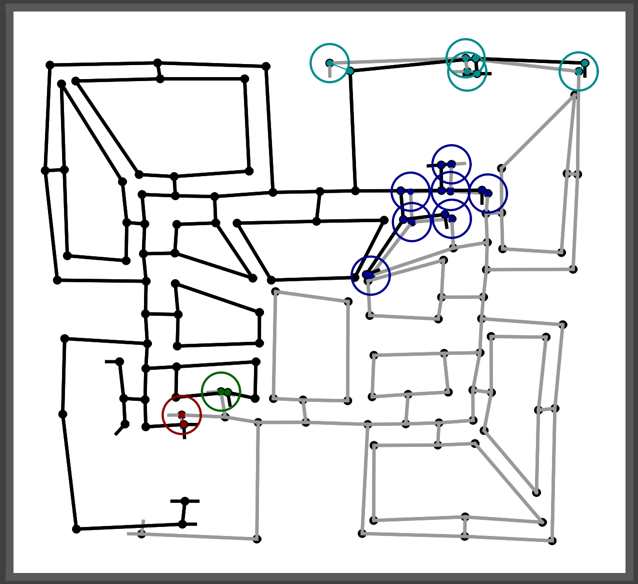
- Well-known algorithm: iterative closest point (ICP):
  - ↪ Compute transformation between two feature sets using an initial matching — minimize weighted squared error
  - ↪ Update matching by adding features that are close under this transformation
  - ↪ Repeat

- Fine, but this ignores the topology in our maps!

Our approach

- Combine graph matching and image registration ideas:

  1. "Grow" hypotheses based on map structure and attributes
  2. Compute geometric transforms of hypotheses
  3. Cluster in the transform space
  4. Pick the "best" cluster



Notation: exact attributes

- Vertices and edges may have *exact* attributes:
  - ↪ For $v \in V_{\mathcal{A}}, V_{\mathcal{B}}$: $\eta^{v} = \{\eta^{v}_{1}, \eta^{v}_{2}, \ldots\}$
  - ↪ For $e \in E_{\mathcal{A}}, E_{\mathcal{B}}$: $\eta^{e} = \{\eta^{e}_{1}, \eta^{e}_{2}, \ldots\}$
- Can be compared directly: $(\eta_i == \eta_j) \rightarrow \{\texttt{\#t,\#f}\}$
- Example: vertex degree

Notation: inexact attributes

- Vertices and edges may also have *inexact* attributes (noisy measurements):

$\hookrightarrow \iota^v = \{\iota_1^v, \iota_2^v, \ldots\}$ and $\iota^e = \{\iota_1^e, \iota_2^e, \ldots\}$

- We assume we have an error model $\Phi_i[\iota_i]$: a pdf parameterized by the value of the ith inexact attribute

- Compared by similarity test: $\textsc{sim}(\iota_i^{v_1}, \iota_i^{v_2}, \Phi_i) \rightarrow \{\texttt{\#t}, \texttt{\#f}\}$

- Example: path length

Algorithm: MERGE($\mathcal{A}, \mathcal{B}$)

$H \leftarrow$ GROW-HYPOTHESES($\mathcal{A}, \mathcal{B}$)
Embed $\mathcal{A}$ and $\mathcal{B}$ in $\mathcal{R}^n$
$C \leftarrow$ CLUSTER-HYPOTHESES(H)
$c \leftarrow$ PICK-BEST-CLUSTER(C)
if c is too small or has large error **then**
    return failure
Using vertex/edge correspondences in c, merge $\mathcal{A}$ and $\mathcal{B}$.
return the merged map

Growing hypotheses

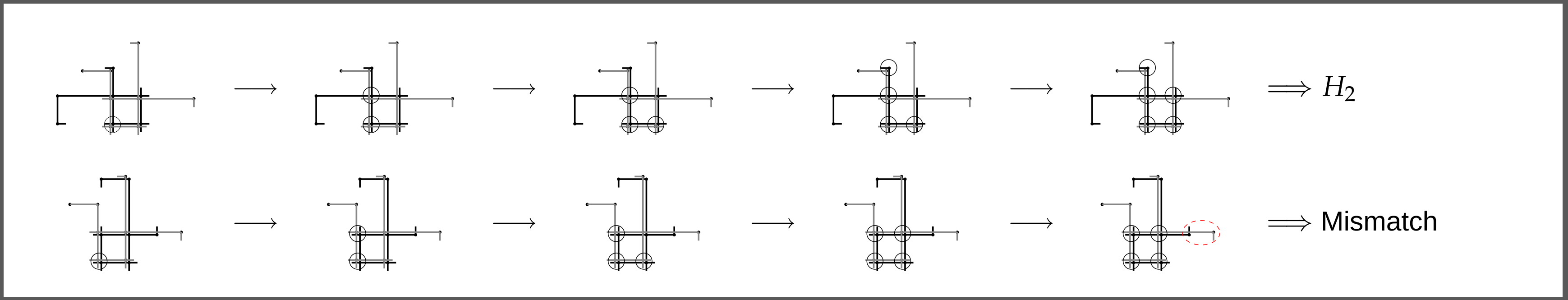
- Start by finding compatible *single-vertex* correspondences
- Compatible: $\forall_i, \eta_i^{v_1} == \eta_i^{v_2}$ and $\forall_j, \mathrm{SIM}(\iota_j^{v_1}, \iota^{v_2} j, \Phi_j) == \texttt{\#t}$
- Correspondence: $(a, b, E_{a,b}), a \in V_{\mathcal{A}}, b \in V_{\mathcal{B}}$, and $E_{a,b}$ is a set of pairings of incident edges; $O(|V_{\mathcal{A}}||V_{\mathcal{B}}|d)$ of these
- Growing a hypothesis:

1. Pick a correspondence
2. Try to add incident edges/vertices
3. If incompatible, discard hypothesis
4. Otherwise, grow as far as possible

Algorithm: GROW-HYPOTHESES($\mathcal{A}, \mathcal{B}$)

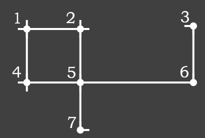
$H \leftarrow \{\}$ *// valid hypotheses*
Initialize M to the set of all single vertex pairs $(a, b, E_{a,b})$ where $a \in V_{\mathcal{A}}, b \in V_{\mathcal{B}},$ COMPAT$(a, b) == \texttt{\#t}$, and $E_{a,b}$ is a set of edge matchings for edges incident to a and b.
while M is not empty **do**
  remove an element m of M
  $P \leftarrow \{m\}$ *// pairs to expand*
  $Q \leftarrow \{\}$ *// pairs in hypothesis*
  while P is not empty **do**
    remove an element $p = (a, b, E_{a,b})$ from P, add p to Q
    for all $(e_a, e_b) \in E_{a,b}$ **do**
      Let $t_a =$ TARG$(a, e_a), t_b =$ TARG(b, e_b)
      if $(t_a, t_b, E_{t_a,t_b}) \in Q \cup P$ **then continue** *// already have these vertices*
      if $(t_a, t_b, E_{t_a,t_b}) \in M \mid (e_a, e_b) \in E_{t_a,t_b}$ and COMPAT(e_a, e_b) **then**
        remove (t_a, t_b, E_{t_a,t_b}) from M, add (t_a, t_b, E_{t_a,t_b}) to P
      else
        $P \leftarrow \{\}$, $Q \leftarrow \{\}$; **break** *// discard this hypothesis*
    end for
  end while
  if $Q \neq \{\}$ **then** add Q to H
end while
return H

Hypothesis growth example

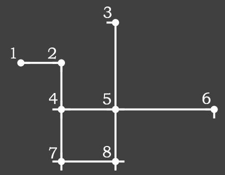


- Simple 2D rectilinear maps (just for the example!)
- Also assume static environment for the example (implies vertex degrees much match exactly)
- Degree-2 vertices (corners): single correspondence
- Degree-4 vertices: four correspondences

Hypothesis growth example (cont.)



Map $\mathcal{A}$



Map $\mathcal{B}$

Hyp.	**Rot.**	**Vertex correspondences**
H_1	0°	a3–b6
H_2	180°	a7–b2, a1–b8, a2–b7, a4–b5, a5–b4
H_3	180°	a7–b6
H_4	0°	a1–b4, a2–b5, a4–b7, a5–b8
H_5	0°	a1–b7, a2–b8
H_6	0°	a1–b8
H_7	0°	a2–b7
H_8	90°	a2–b8
H_9	180°	a1–b7, a4–b4
H_{10}	180°	a4–b7
H_{11}	270°	a1–b4, a2–b7
H_{12}	270°	a1–b5, a2–b8, a4–b4, a5–b5
H_{13}	270°	a1–b7
H_{14}	270°	a1–b8, a4–b7
H_{15}	270°	a4–b8

Algorithm: MERGE($\mathcal{A}, \mathcal{B}$)

1: $H \leftarrow$ GROW-HYPOTHESES($\mathcal{A}, \mathcal{B}$)
2: Embed $\mathcal{A}$ and $\mathcal{B}$ in $\mathcal{R}^n$
3: $C \leftarrow$ CLUSTER-HYPOTHESES(H)
4: $c \leftarrow$ PICK-BEST-CLUSTER(C)
5: **if** c is too small or has large error **then**
6:     **return** failure
7: Using vertex/edge correspondences in c, merge $\mathcal{A}$ and $\mathcal{B}$.
8: **return** the merged map

Transform estimation

- Once we finish growing hypothesized correspondences, we compute transformations using the matched vertices

- First need to embed the maps in $\mathcal{R}^n$ (measurements alone may be inconsistent)

$\hookrightarrow$ Plenty of approaches; we use Duckett and Saffiotti (2000) — spring-based method

- Then compute transform using methods from image registration

$\hookrightarrow$ We use 2D SVD-based closed form solution from Fitzpatrick, Hill, and Maurer (2000) — SVD of a 2x2 covariance matrix

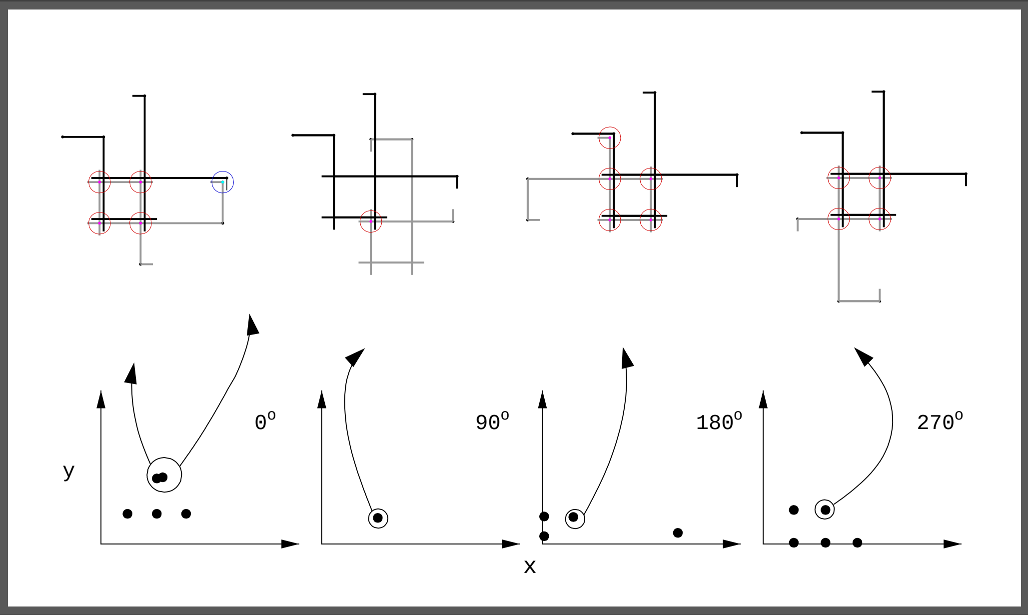
Clustering

- Hypotheses that are close together in transformation space are "geometrically consistent"
- If they do not conflict structurally, they are likely to be disconnected but consistent matches
- Since we want all consistent matches, cluster the hypotheses in transformation space
- Use simple agglomerative clustering: $O(n^2 \log n)$, implemented well ($n \equiv$ number of hypotheses)

Algorithm: CLUSTER-HYPOTHESES(H)

```
for all h ∈ H do
    Compute t[h] (hypothesis transform) using SVD-based registration
end for
Let C = H
repeat
    Find c_i, c_j such that d = min_{c_i,c_j ∈ C} ||t[c_i] − t[c_j]||
    if d < ε then
        C ← C − {c_i, c_j}
        C ← C ∪ {c_i ∪ c_j}
        Compute t[{c_i ∪ c_j}]
    end if
until d ≥ ε
return C
```

Clustering example



Algorithm: MERGE($\mathcal{A}, \mathcal{B}$)

1: $H \leftarrow$ GROW-HYPOTHESES($\mathcal{A}, \mathcal{B}$)
2: Embed $\mathcal{A}$ and $\mathcal{B}$ in $\mathcal{R}^n$
3: $C \leftarrow$ CLUSTER-HYPOTHESES(H)
4: $c \leftarrow$ PICK-BEST-CLUSTER(C)
5: **if** c is too small or has large error **then**
6:     **return** failure
7: Using vertex/edge correspondences in c, merge $\mathcal{A}$ and $\mathcal{B}$.
8: **return** the merged map

Choosing a cluster

- “Quality” of a cluster depends on application, *a priori* knowledge
- Some heuristics (by priority):
  1. Total number of vertices
  2. Squared error between matched vertices under cluster transform
  3. Number of hypotheses in a cluster
- If the “best” cluster is very small or has large error, return failure

Algorithm: PICK-BEST-CLUSTER(C)

$v_{\max} \leftarrow \max_{x \in C} \sum_{h \in x} |h|$ *// most vertices*

$B \leftarrow \{c \in C \mid \sum_{h \in c} |h| = v_{\max}\}$

$\epsilon_{\max} \leftarrow \min_{x \in B} \sum_{a,b \in h \in x} ||a - b||$ *// smallest error*

$B \leftarrow \left\{c \in B \mid \sum_{a,b \in h \in c} ||a - b|| = \epsilon_{\max}\right\}$

$h_{\max} = \min_{x \in B} |x|$ *// fewest hypotheses*

$B \leftarrow \{c \in B \mid |c| = h_{\max}\}$

if $|B| == 1$ **then**

    return $c \in B$

else

    return an arbitrarily chosen $c \in B$

Algorithm: MERGE($\mathcal{A}, \mathcal{B}$)

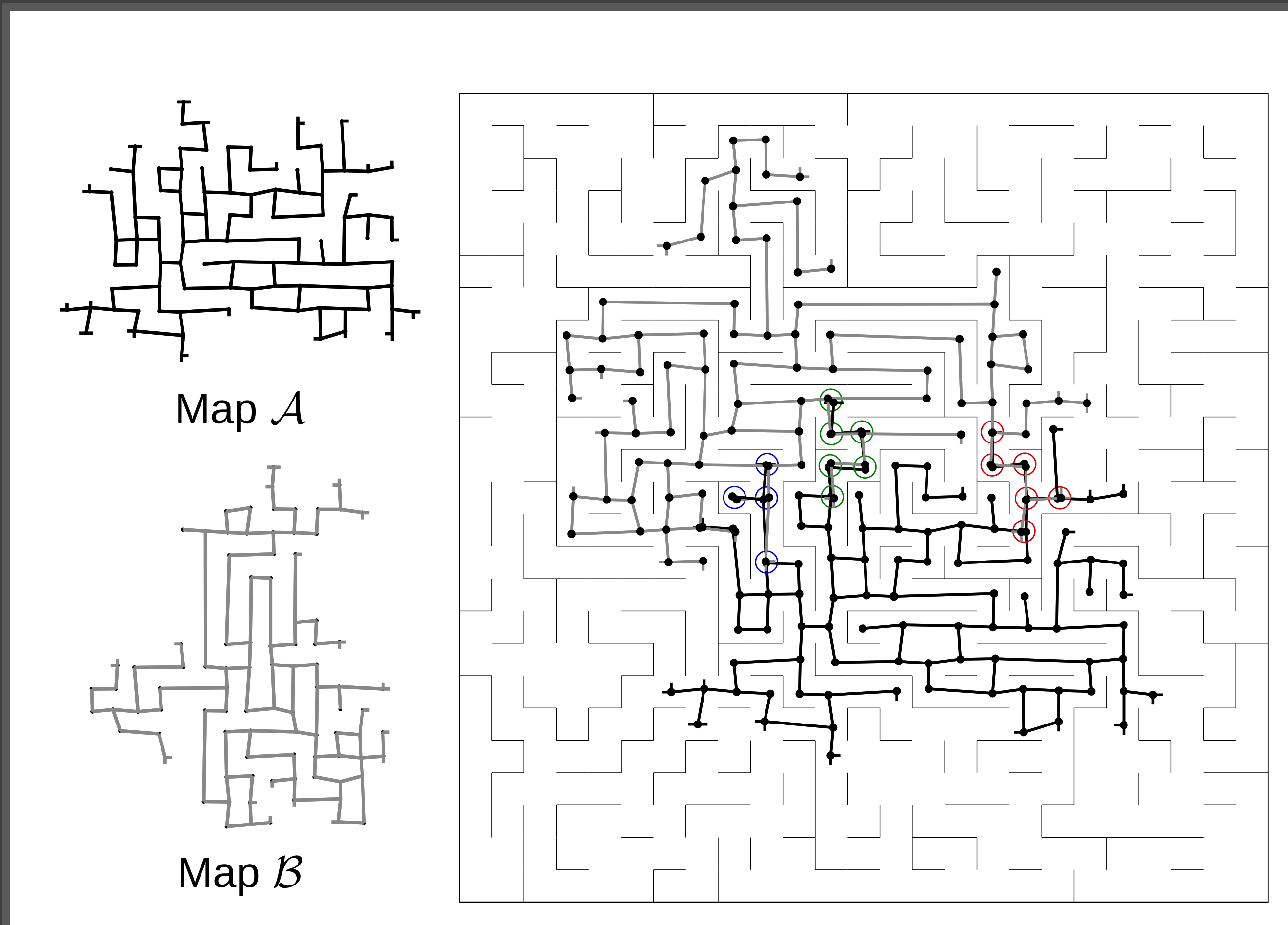
$H \leftarrow$ GROW-HYPOTHESES($\mathcal{A}, \mathcal{B}$)
Embed $\mathcal{A}$ and $\mathcal{B}$ in $\mathcal{R}^n$
$C \leftarrow$ CLUSTER-HYPOTHESES(H)
$c \leftarrow$ PICK-BEST-CLUSTER(C)
if c is too small or has large error **then**
    return failure
Using vertex/edge correspondences in c, merge $\mathcal{A}$ and $\mathcal{B}$.
return the merged map

$\|$ Merge measurements for vertices/edges in c, combine the rest

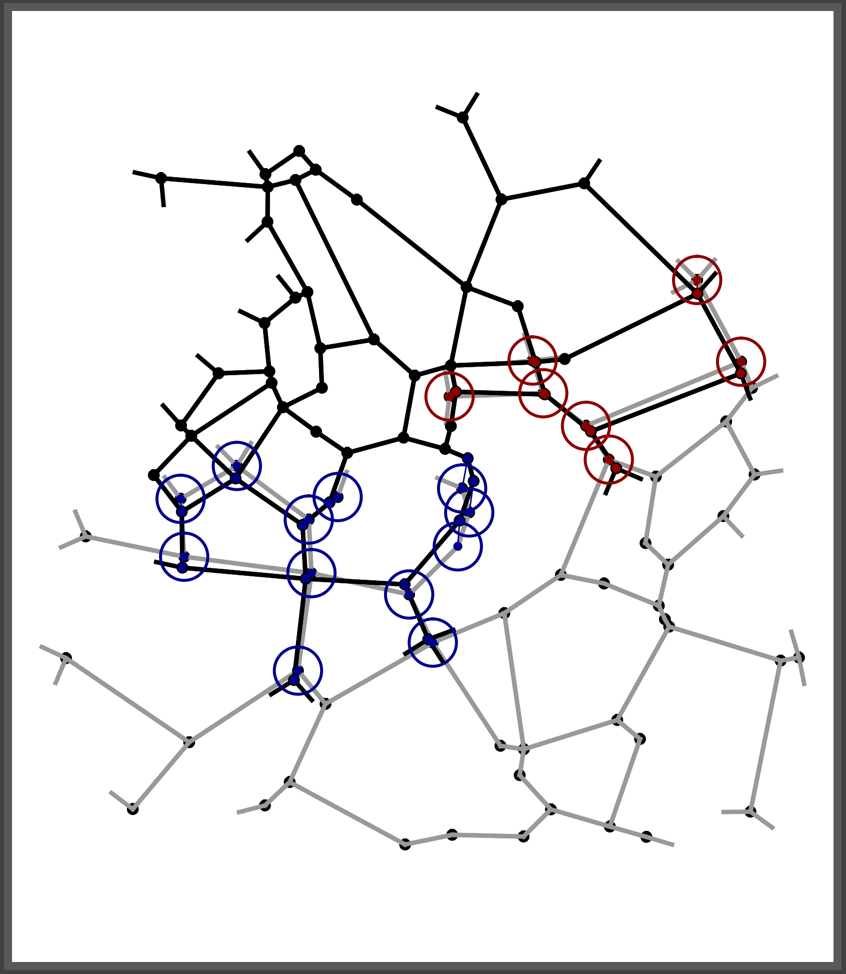
Results

- Implemented for arbitrary 2D topological maps
- Preliminaries:
  - Assume degree of vertices is known once they are discovered
  - Edge angles (inexact) and an ordering can be obtained
  - Similarity of edge lengths computed using a χ^2 test based on a Gaussian odometry error model
  - Intra-cluster translational distance threshold: 0.5 m
  - Intra-cluster rotational distance threshold: $\pi/8$

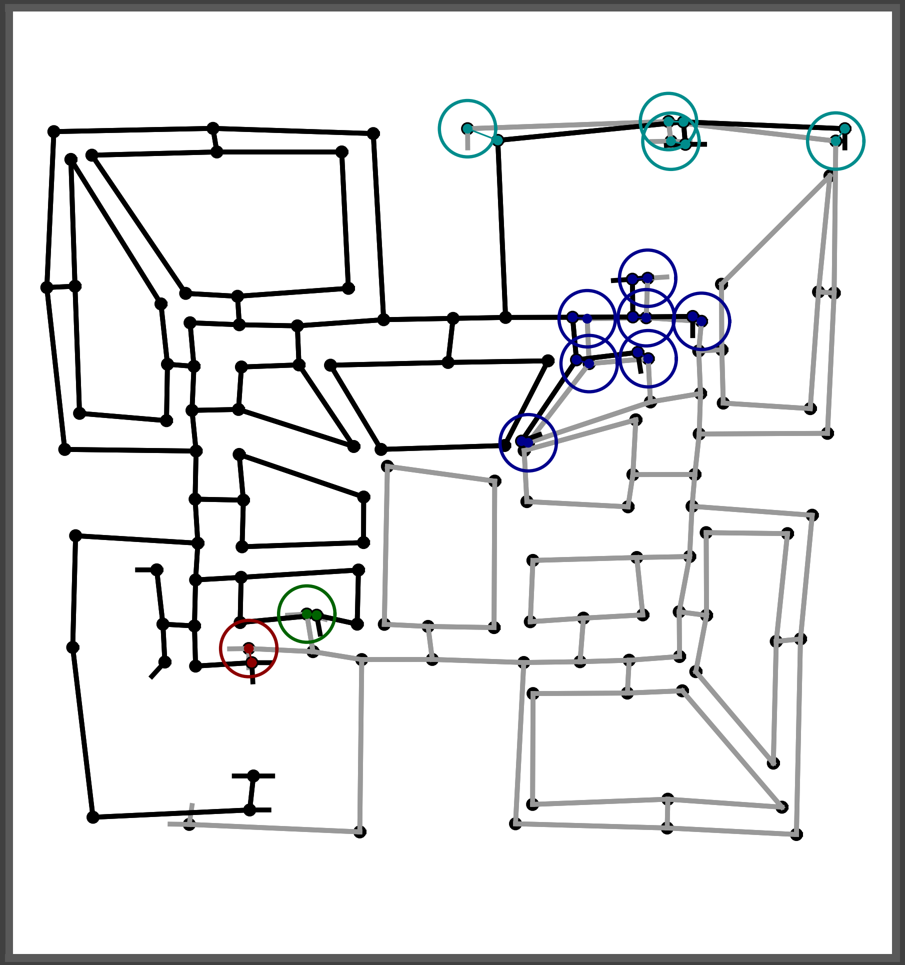
Maze maps



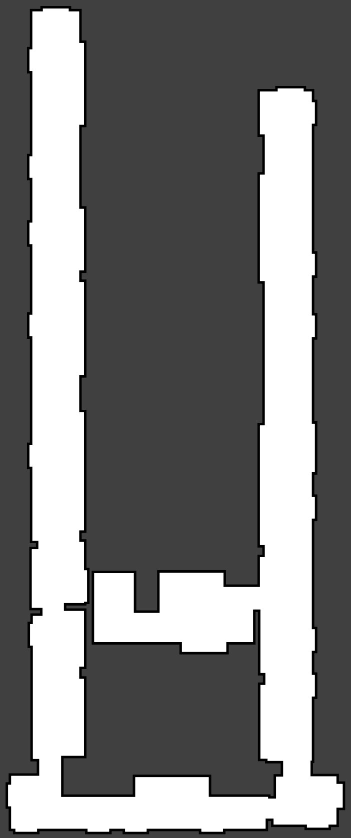
Random planar maps

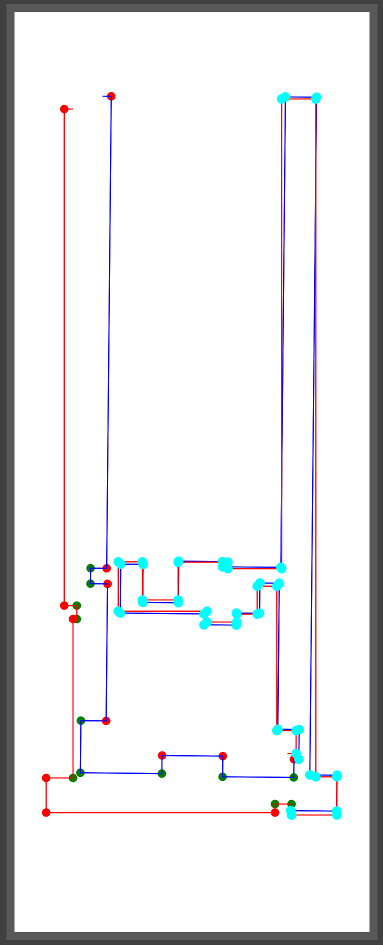


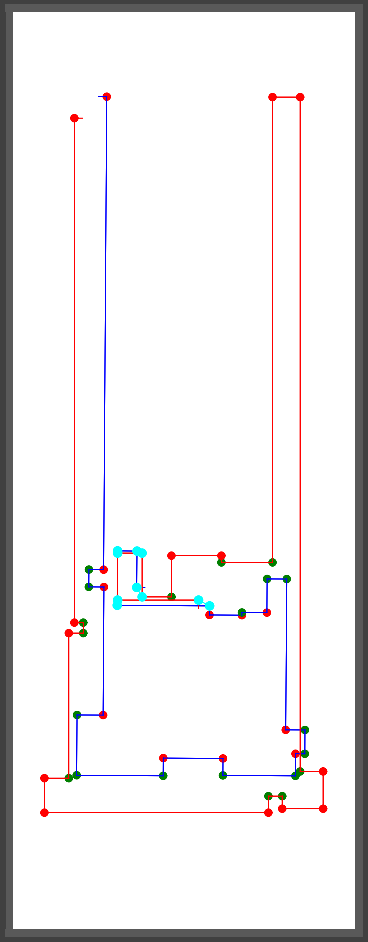
Hand generated maps



Real-world maps (Amos Eaton)







Performance: correctness

Map merging statistics (1000 runs for each trial) — varying map overlap and measurement error; error model assumed $\sigma\% = 7\%$:

Overlap	**% correct**
0%	99.2%
2%	99.1%
4%	99.6%
6%	99.4%
8%	99.2%
10%	99.3%
12%	99.3%

map σ%	**% correct**
1%	99.0%
3%	99.5%
5%	99.6%
7%	91.9%
9%	36.5%
11%	12.6%

Performance: correctness (cont.)

Structure-only growth map merging statistics (200 runs for each trial) — no inexact attribute similarity tests performed:

Overlap	% correct
0%	99.0%
2%	98.5%
4%	99.0%
6%	99.5%
8%	98.0%
10%	99.0%
12%	99.0%

map σ%	% correct
1%	99.5%
3%	98.5%
5%	99.5%
7%	99.0%
9%	98.5%
11%	99.5%

Performance: running time

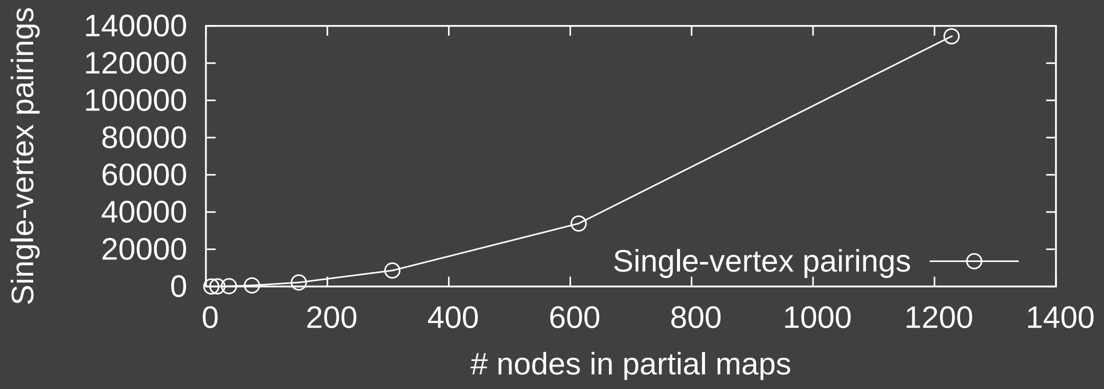
- Growth phase: $s = O(|V_{\mathcal{A}}||V_{\mathcal{B}}|d)$ initial matches; assuming d is a constant and each map has $O(n)$ vertices, $s = O(n^2)$
- Worst case: $m = O(s)$ hypotheses remaining after growth
- Clustering (implemented efficiently): $O(m^2 \log m) =$

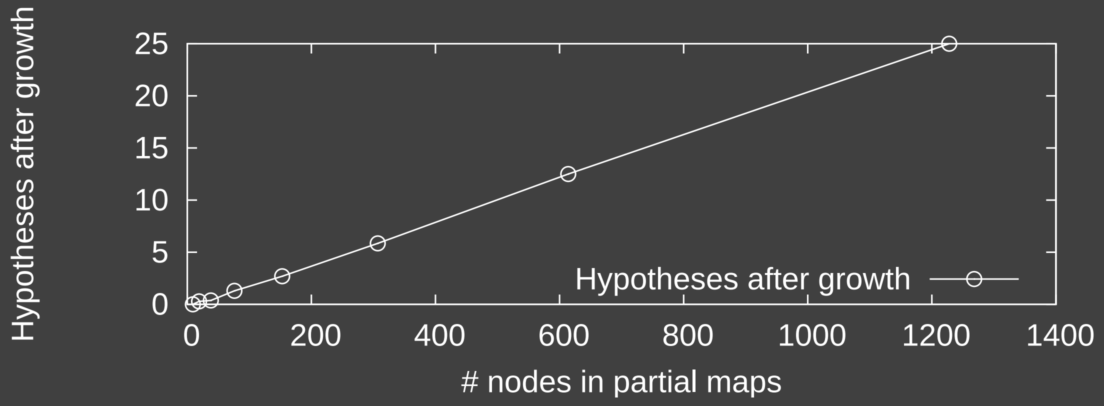
  Theoretical worst case: $O(n^4 \log n)$

- In reality: $m \ll n^2$, and growth phase dominates:

  Practical worst case: $O(n^2)$

Performance: running time (cont.)





$\|$ Merging two ~ 100-vertex maps: < 0.05 s (P3-650 MHz)

Extension ideas

‖ Rollback-capable map storage

↪ Store maps from each robot separately

↪ Compute separate merged map, or compute merged measurements on-the-fly

↪ Extends to > 2 robots with a dependency tree structure

‖ Iterative reclustering

↪ Problem: sometimes small hypotheses have significantly skewed transformations, and they are not clustered correctly

↪ Idea: after initial clustering, add new hypotheses to cluster based on metric error under hypothesis transform

↪ Repeat until no changes occur

Extension ideas (cont.)

- Incremental updates — for “re-”merging already-merged maps
  1. Discard old hypotheses invalidated by updates
  2. Extend existing hypotheses from new and modified edges
  3. Create/grow hypotheses from new and modified vertices
  4. Recluster hypotheses
- Can trade off computational savings vs. information retention:

  ↪ Only retain best (few) cluster(s)

  ↪ Only expand best (few) cluster(s)

  ↪ Retain all clusters, expand all clusters

  ↪ Etc.

References

Dedeoglu, G. and Sukhatme, G. 2000. Landmark-based matching algorithm for cooperative mapping by autonomous robots. In *Proceedings of the 2000 International Symposium on Distributed Autonomous Robotic Systems (DARS)*, 251–260.

Duckett, T. and Saffiotti, A. 2000. Building globally consistent gridmaps from topologies. In *Proceedings of the 6th International IFAC Symposium on Robot Control (SYROCO)*, 357–361.

Fitzpatrick, J., Hill, D., and Maurer, Jr., C. Image registration. In Sonka, M. and Fitzpatrick, J., editors, *Handbook of Medical Imaging, volume 2: Medical Image Processing and Analysis*, chapter 8. SPIE, 2000.

Ko, J., Stewart, B., Fox, D., Konolige, K., and Limketkai, B. 2003. A practical, decision-theoretic approach to multi-robot mapping and exploration. In *Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots & Systems*, 212–217.