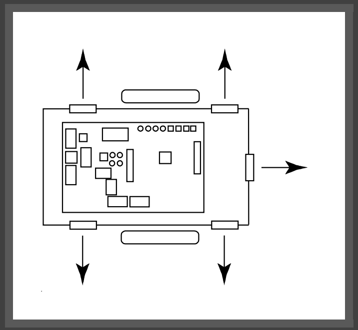
Loop closing in topological maps

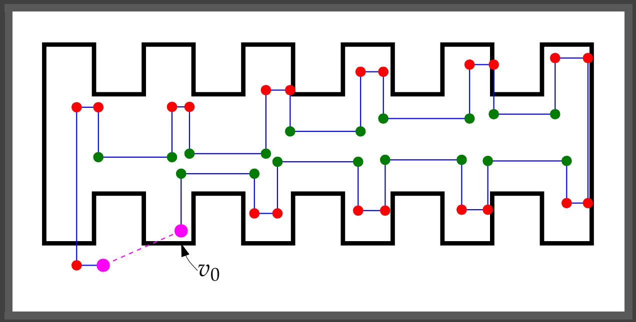
Kris Beevers and Wes Huang
Department of Computer Science
Rensselaer Polytechnic Institute
{beevek,whuang}@cs.rpi.edu

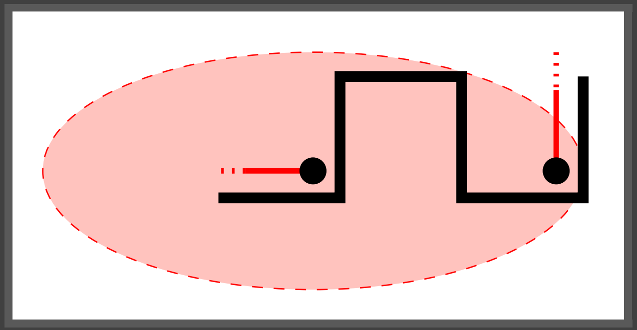
April 22, 2005

Motivation

- Sensing-limited robots: cheap, disposable
- Topological mapping
- Closing loops: recognizing when the robot has returned to a place it has been before
- Because of sensing limitations, we want to close loops based mainly on odometry measurements







Previous approaches

- Recognize unique places in maps:
  - ↪ Bender, *et al.*, 1998, and others: drop a marker
  - ↪ Kuipers and Beeson, 2002: recognize distinct "sensing signatures" of nodes
- Use map topology: structural characteristics of the map used to make loop-closing decisions
  - ↪ Kuipers and Byun, 1991 ("rehearsal procedure"), Choset and Nagatani, 2001, Tomatis *et al.*, 2002 (POMDP approach)
- SLAM approaches ("data association," "correspondence") — focused on landmark maps

Our approach

- Strategy:
  - ↪ Identify loop-closing hypotheses
  - ↪ Accumulate evidence about them based on measurements
  - ↪ Apply Dempster-Shafer theory to manage belief
- Similar to (Cox and Leonard, 1994): maintaining multiple hypotheses about dynamic world using Bayesian framework
- Problems to solve:
  1. Modifying a Dempster-Shafer frame of discernment
  2. Determining belief about hypotheses based on evidence provided by measurements
- Assumptions: known error models, can compute confidence bounds

Dempster-Shafer theory

- Alternative framework for representing uncertainty
- Allocate belief to sets of possibilities:

$$\begin{aligned} \Theta &= \{H_1, H_2, H_3\} \quad \text{(\textit{frame of discernment})} \\ 2^\Theta &= \{\emptyset, \{H_1\}, \{H_2\}, \{H_3\}, \{H_1, H_2\}, \{H_1, H_3\}, \{H_2, H_3\}, \{H_1, H_2, H_3\}\} \end{aligned}$$

- *Basic probability assignment* (BPA): $m : 2^\Theta \rightarrow [0, 1]$ such that:
  - $m(\emptyset) = 0$
  - $\sum_{A \subseteq \Theta} m(A) = 1$

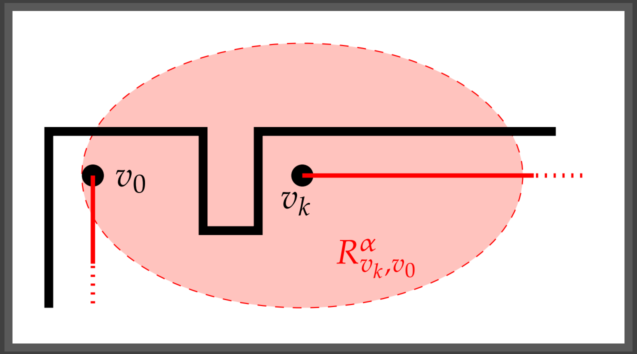
$\hookrightarrow$ Traditional probability assigns belief over Θ, not 2^Θ

$\hookrightarrow$ "Ignorance": belief assigned to a set of multiple possibilities

- Combine BPAs using *Dempster's rule of combination*

Making hypotheses

- Make loop-closing hypotheses based on odometry error model
- If compatible vertices are within a specified confidence bound:
  1. Add hypothesis $H_k \equiv v_k = v_0$ to Θ
  2. Continue traversing environment



$$\boxed{\Theta = \{N, H_1, H_2, \ldots, H_k\}}$$

Expanding the frame of discernment

- Need to be able to add hypotheses: $\Theta_k = \Theta_{k-1} \cup \{H_k\}$
- **Problem**: must recompute BPA given new frame

$\hookrightarrow$ Cannot copy old BPA m_{k-1}

$\hookrightarrow$ No belief would be assigned to elements of 2^{Θ_k} containing H_k

- Solution: $\forall \{A \in 2^{\Theta_{k-1}} | N \subseteq A\}$: $\boxed{m^k(A \cup \{H_k\}) = m^{k-1}(A)}$
- Any initial evidence about H_k is combined into the global BPA using Dempster's rule

Computing a BPA

- Measurements provide evidence about hypotheses in Θ:

$\hookrightarrow$ E.g., multiple odometry measurements of a path should match closely under the correct hypothesis

- Idea: compute a BPA based on new evidence and merge it with a global BPA to update belief about hypotheses

- Given measurements that should match under a hypothesis:

1. Use statistical significance test based on error model

   $\hookrightarrow$ How closely do the measurements really match?

2. Use this measure over all $H_i \in \Theta$ to compute a BPA over 2^Θ

Computing a BPA (cont.)

- For a new measurement, for each hypothesis H_i:

1. Compute squared z-score using expected measurement:

$$\boxed{z^2 = \sum_{\ell_j \in L_i} (\ell_j - \hat{\ell}\,)^2 / \hat{\sigma}^2}$$

2. Use z^2 in a χ^2 test (z^2 follows a χ^2 distribution with $n = |L_i|$ degrees of freedom):

$$\boxed{\Phi_i = \int_{z^2}^{\infty} \frac{y^{n/2-1} e^{-y/2}}{2^{n/2}\Gamma(n/2)}\, dy}$$

(Φ_i: probability that $\ell \in L_i$ came from same distribution)

3. Compute basic probability to assign to N:

$$\boxed{\Phi_0 = 1 - \max_{i>0} \Phi_i}$$

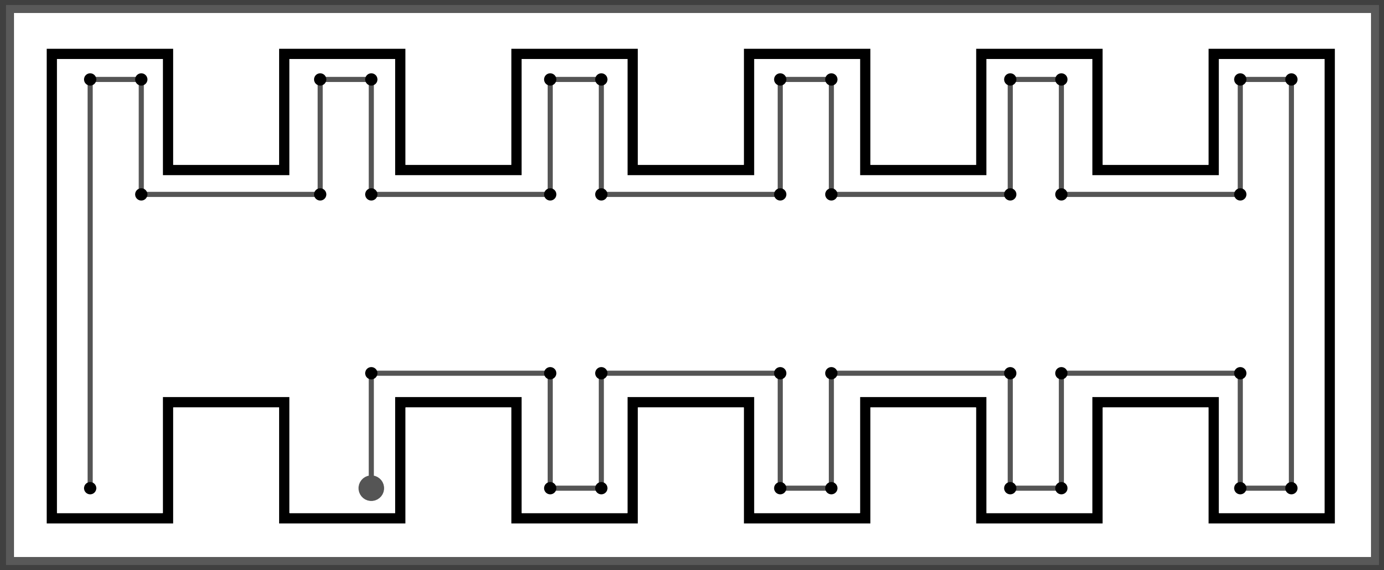
Algorithm: COMPUTE-BPA$(\Theta, \Phi_0, \Phi_1, \ldots, \Phi_k)$

Normalize Φ_i's
for $j = |\Theta| \ldots 2$ **do**
  Let $\Lambda = \{A \in 2^\Theta \mid |A| = j\}$
  for all $A \in \Lambda$ **do** *// compute probability information content*
    $p_A \leftarrow \sum_{i \mid H_i \in A} \Phi_i$
    $\eta_A \leftarrow 1 + \frac{\sum_{i \mid H_i \in A} \frac{\Phi_i}{p_A} \log \frac{\Phi_i}{p_A}}{\log |A|}$
  for all $H_i \in \Theta$ **do** *// compute normalization constant*
    $t_i \leftarrow \max\left(1, \sum_{A \in \Lambda \mid H_i \in A} \eta_A\right)$
  for all $A \in \Lambda$ **do** *// compute b.p.a.*
    $m(A) \leftarrow \sum_{i \mid H_i \in A} \Phi_i \frac{1-\eta_A}{t_i}$
  for all $H_i \in \Theta$ **do** *// "bleed off" probability mass*
    $\Phi_i \leftarrow t_i \Phi_i$
for all $H_i \in A \in 2^\Theta \mid |A| = 1$ **do**
  $m(A) = \Phi_i$ *// assign remaining mass to singletons*

Algorithm: COMPUTE-BPA$(\Theta, \Phi_0, \Phi_1, \ldots, \Phi_k)$

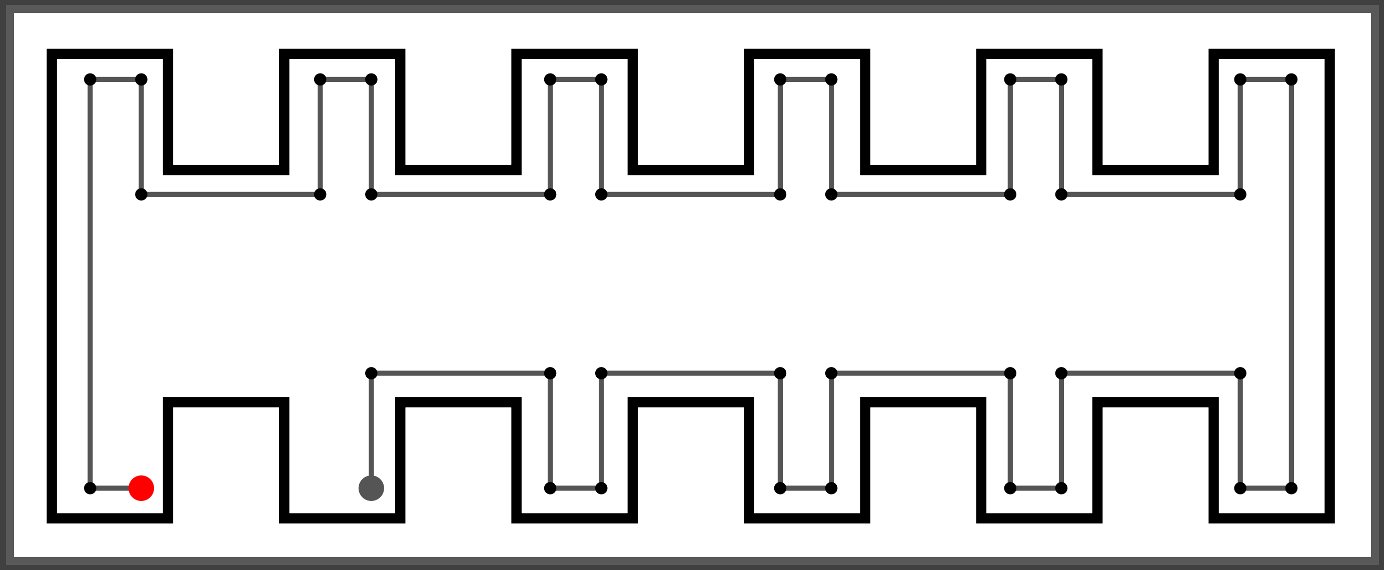
Normalize Φ_i's
for $j = |\Theta| \ldots 2$ **do**
  Let $\Lambda = \{A \in 2^\Theta \mid |A| = j\}$
  for all $A \in \Lambda$ **do** *// compute probability information content*
    $p_A \leftarrow \sum_{i \mid H_i \in A} \Phi_i$
    $\eta_A \leftarrow 1 + \frac{\sum_{i \mid H_i \in A} \frac{\Phi_i}{p_A} \log \frac{\Phi_i}{p_A}}{\log |A|}$
  for all $H_i \in \Theta$ **do** *// compute normalization constant*
    $t_i \leftarrow \max\left(1, \sum_{A \in \Lambda \mid H_i \in A} \eta_A\right)$
  for all $A \in \Lambda$ **do** *// compute b.p.a.*
    $m(A) \leftarrow \sum_{i \mid H_i \in A} \Phi_i \frac{1-\eta_A}{t_i}$
  for all $H_i \in \Theta$ **do** *// "bleed off" probability mass*
    $\Phi_i \leftarrow t_i \Phi_i$
for all $H_i \in A \in 2^\Theta \mid |A| = 1$ **do**
  $m(A) = \Phi_i$ *// assign remaining mass to singletons*

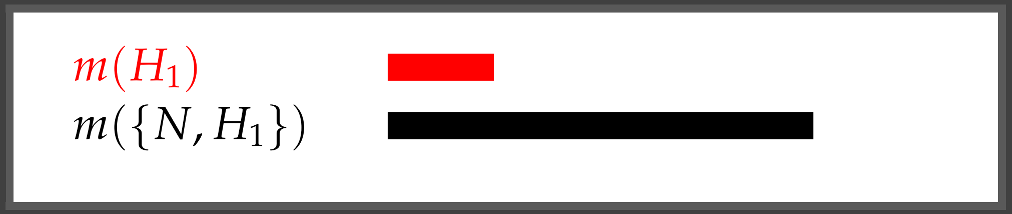
Example: self-similarity



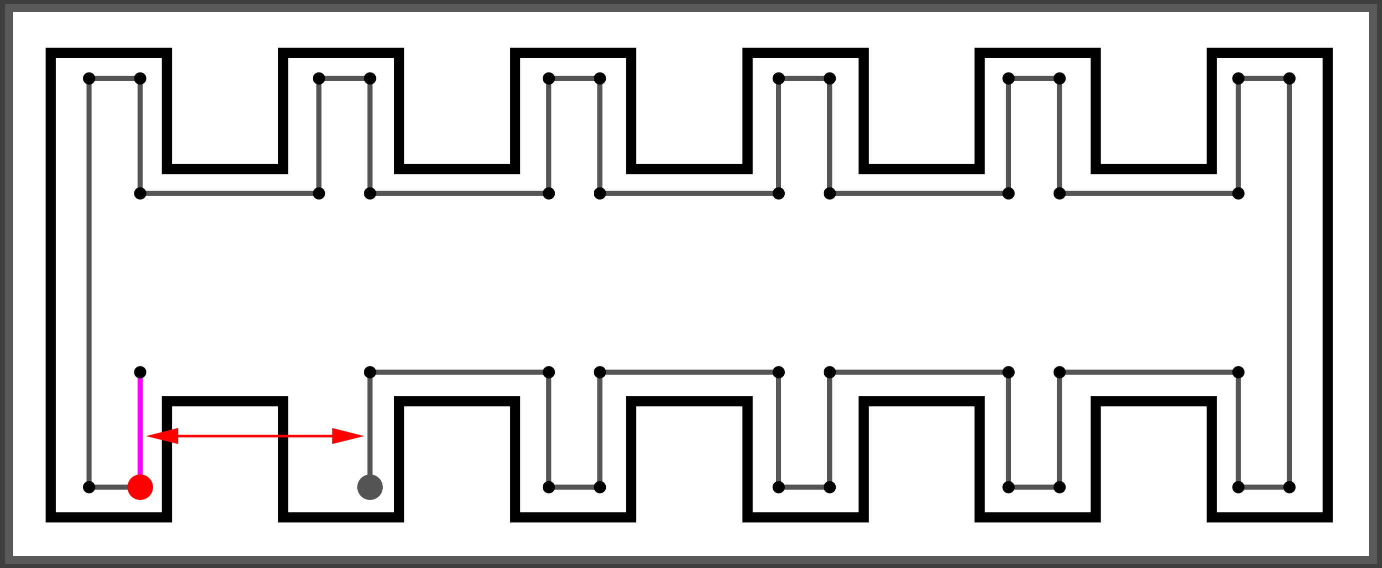
No hypotheses so far

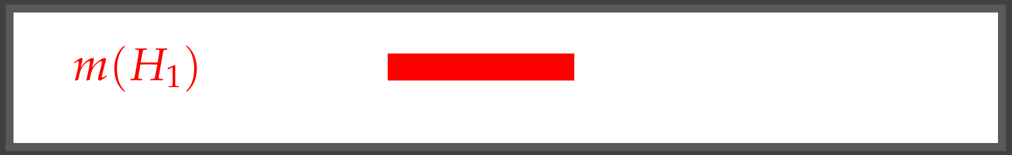
Example: self-similarity



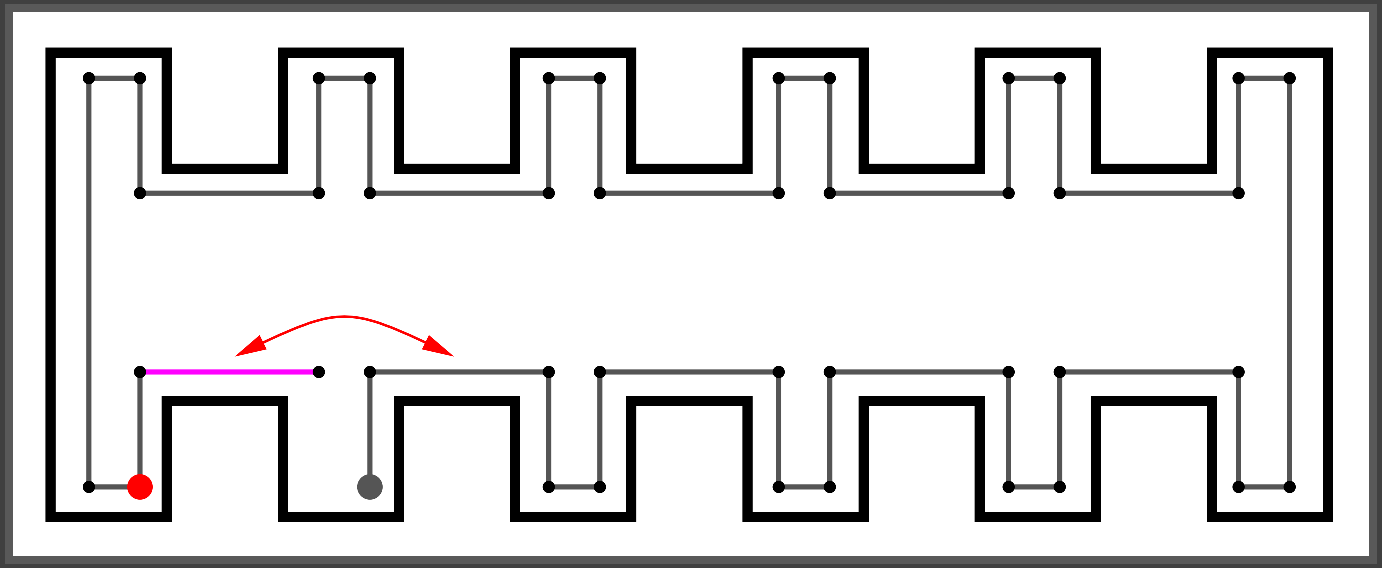


Example: self-similarity

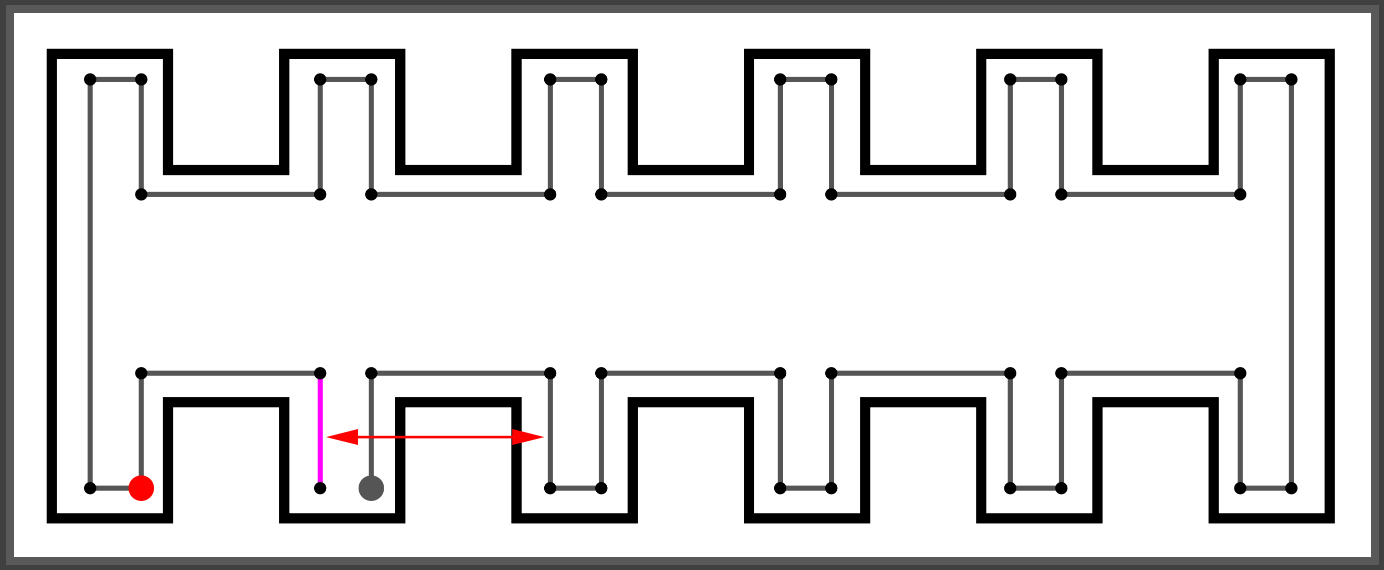




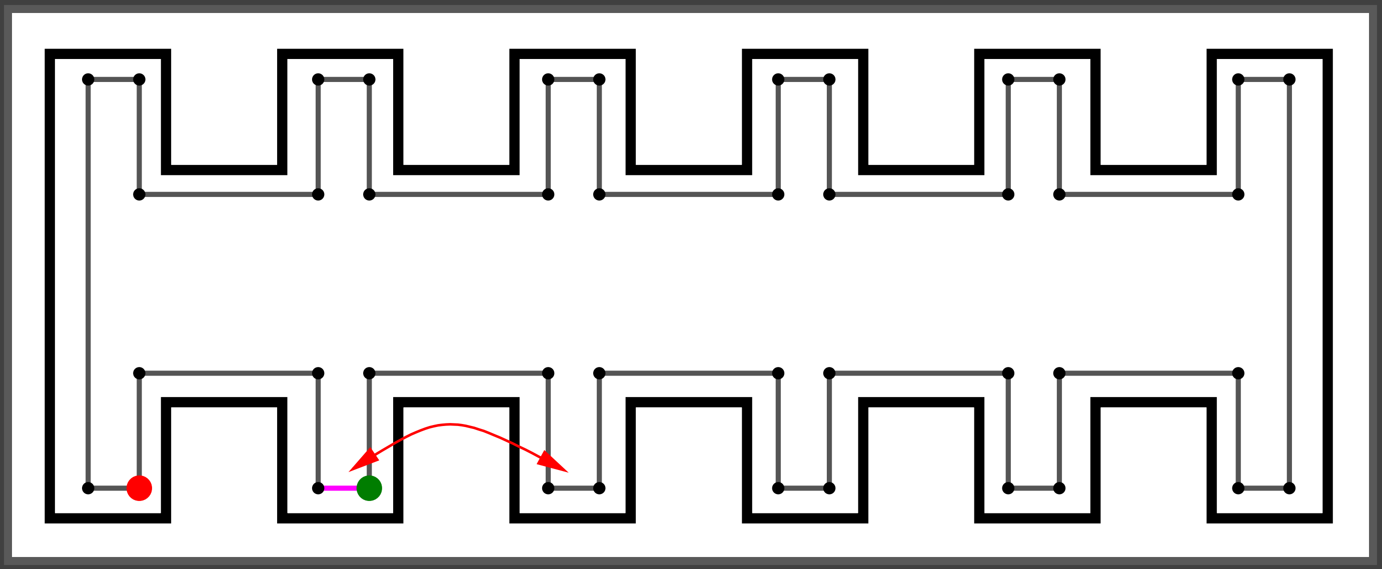
Example: self-similarity

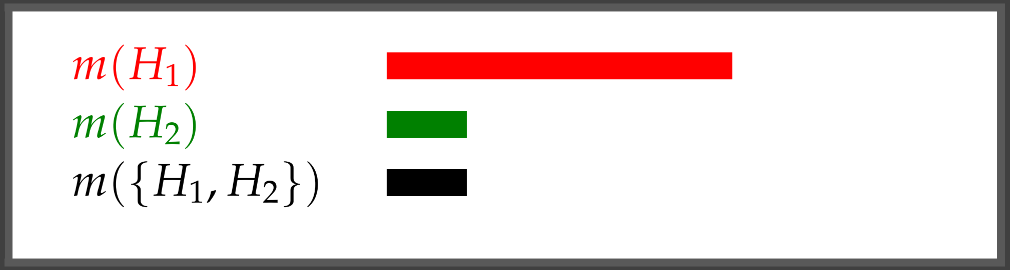


Example: self-similarity

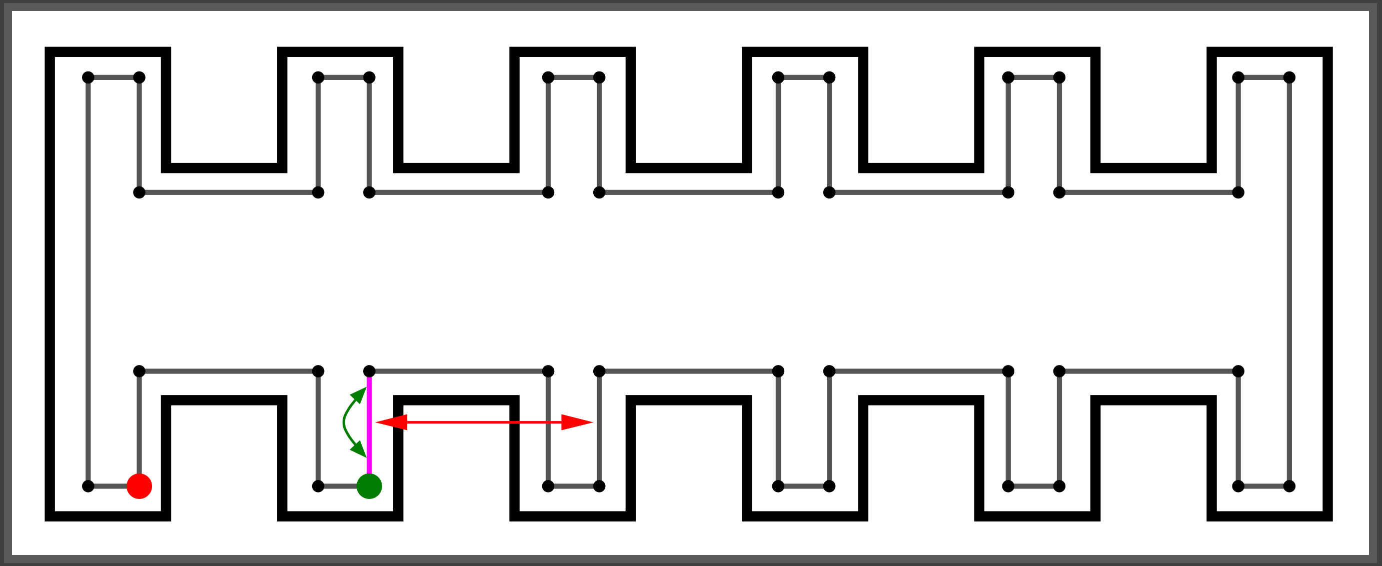


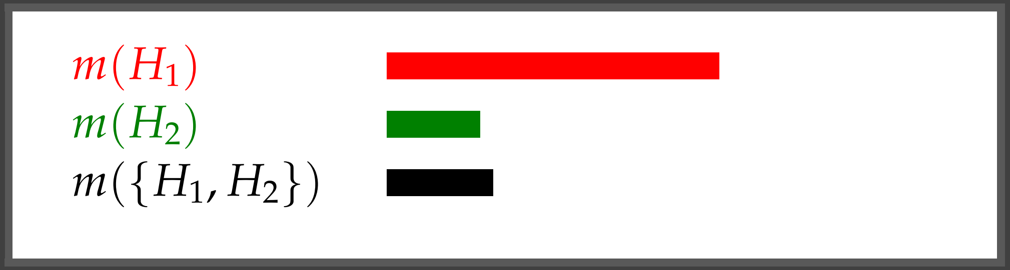
Example: self-similarity



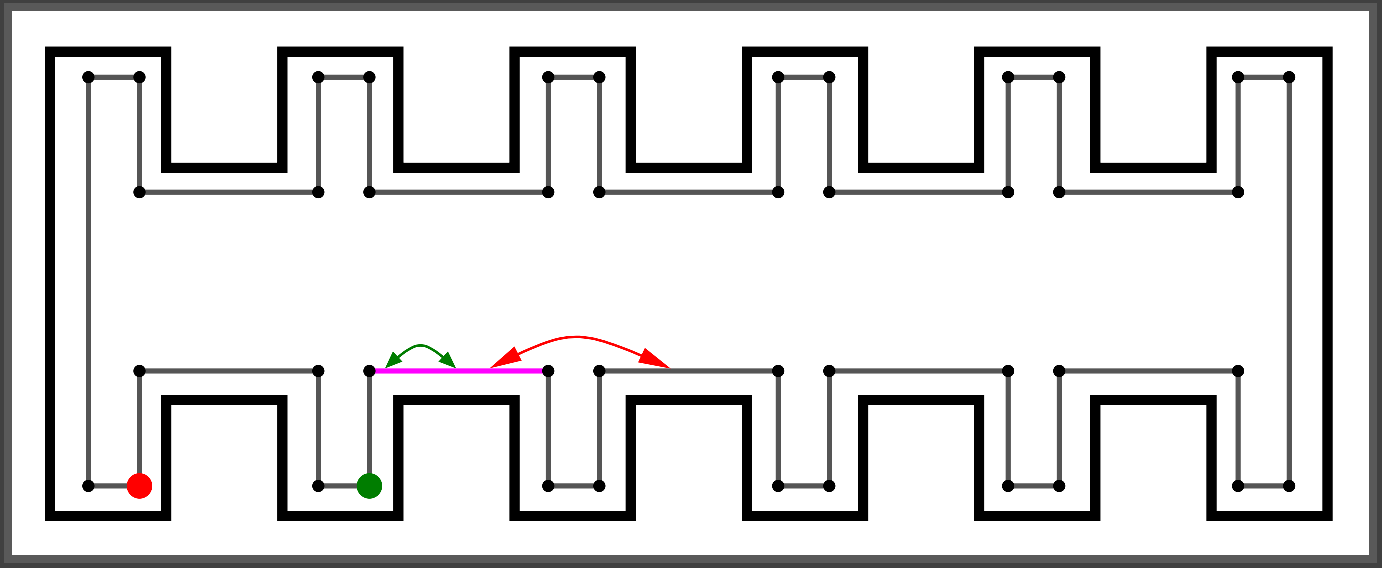


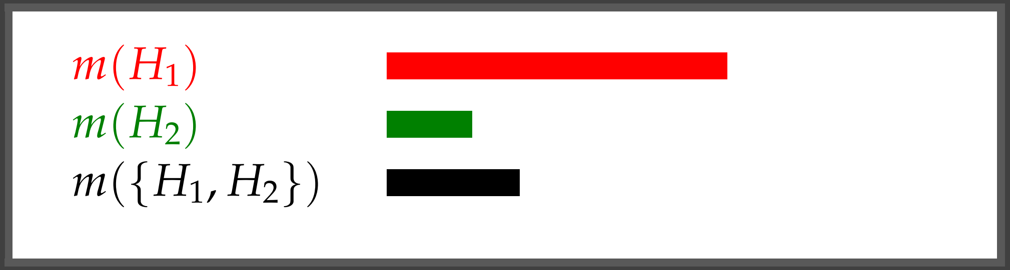
Example: self-similarity



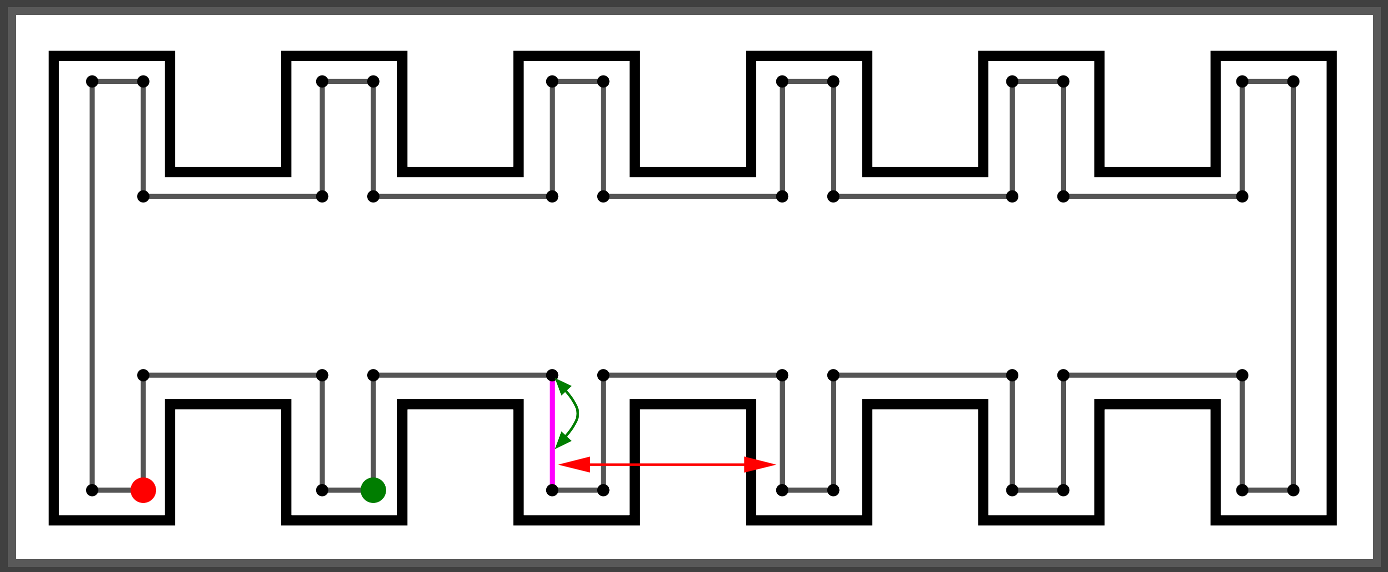


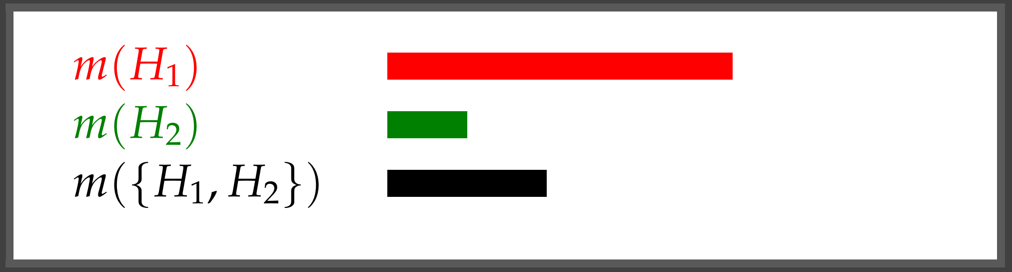
Example: self-similarity





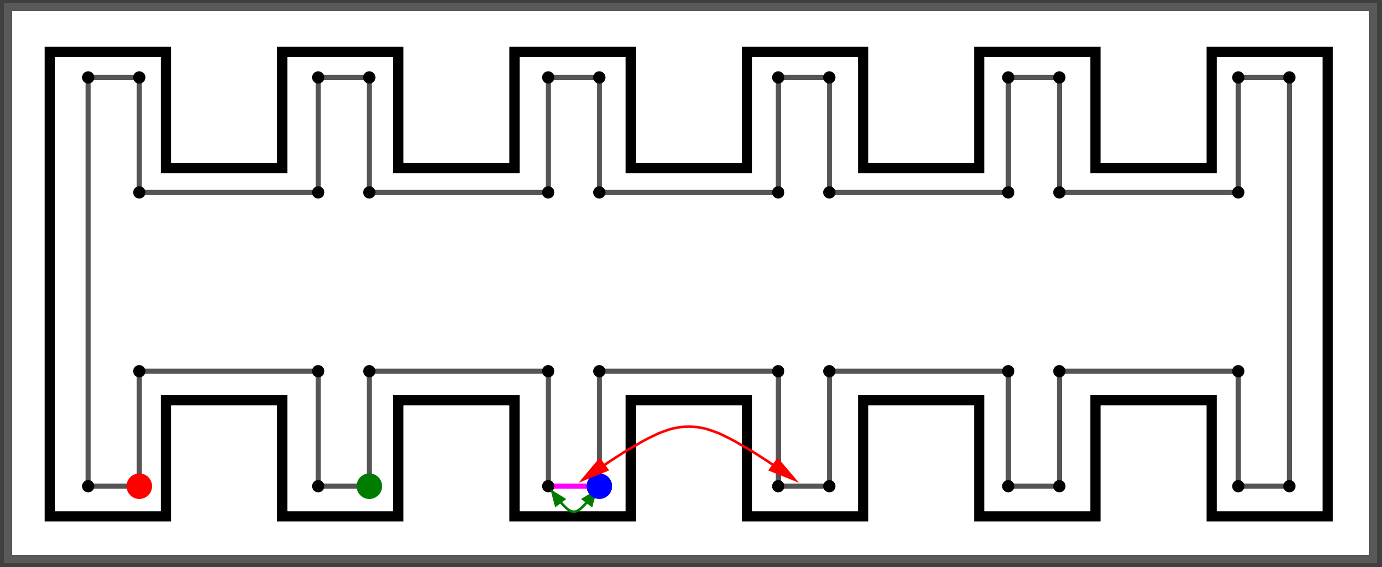
Example: self-similarity

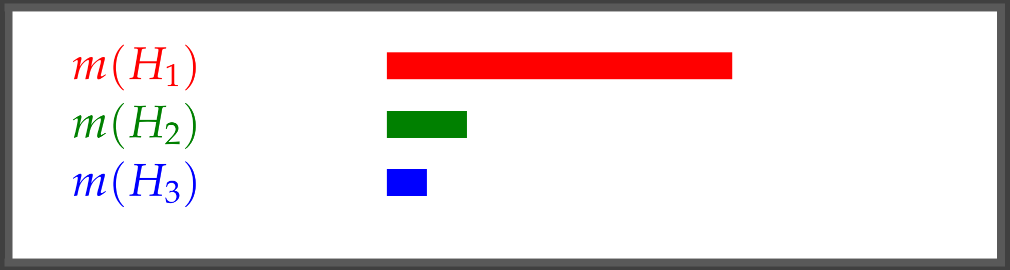




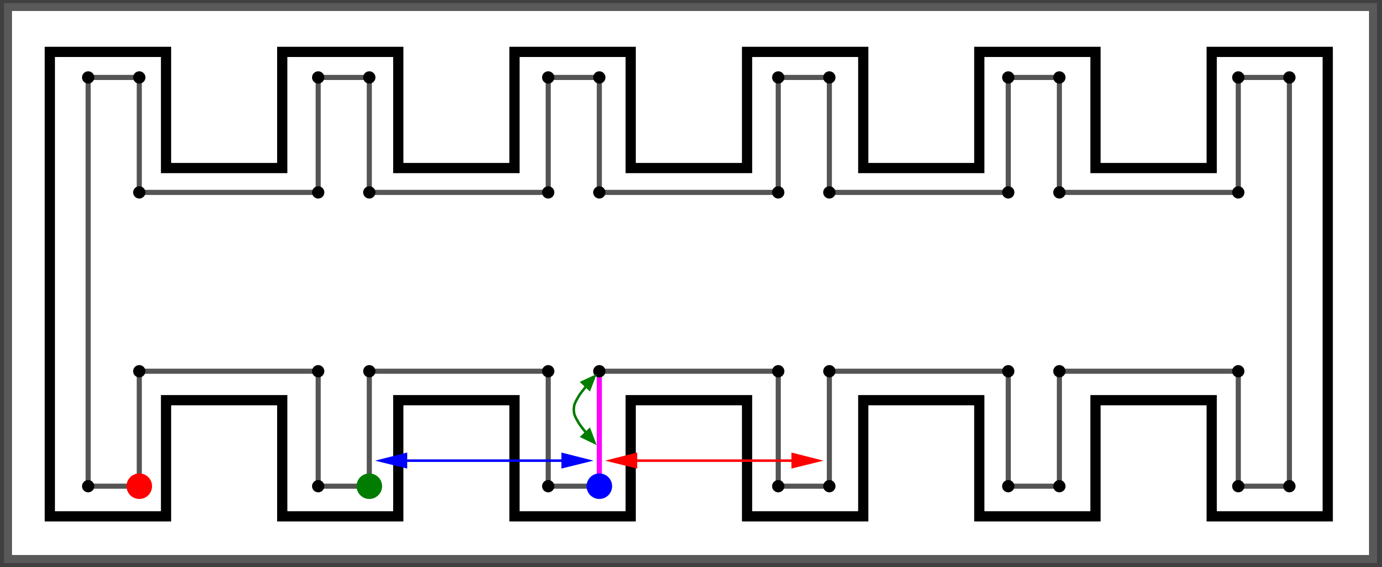
Kris Beevers and Wes Huang
Rensselaer Polytechnic Institute

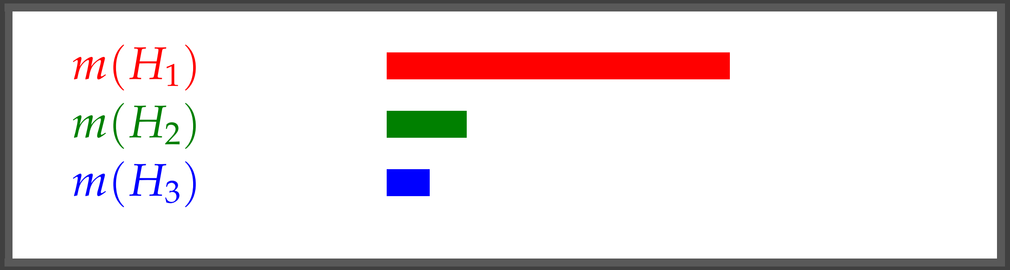
Example: self-similarity



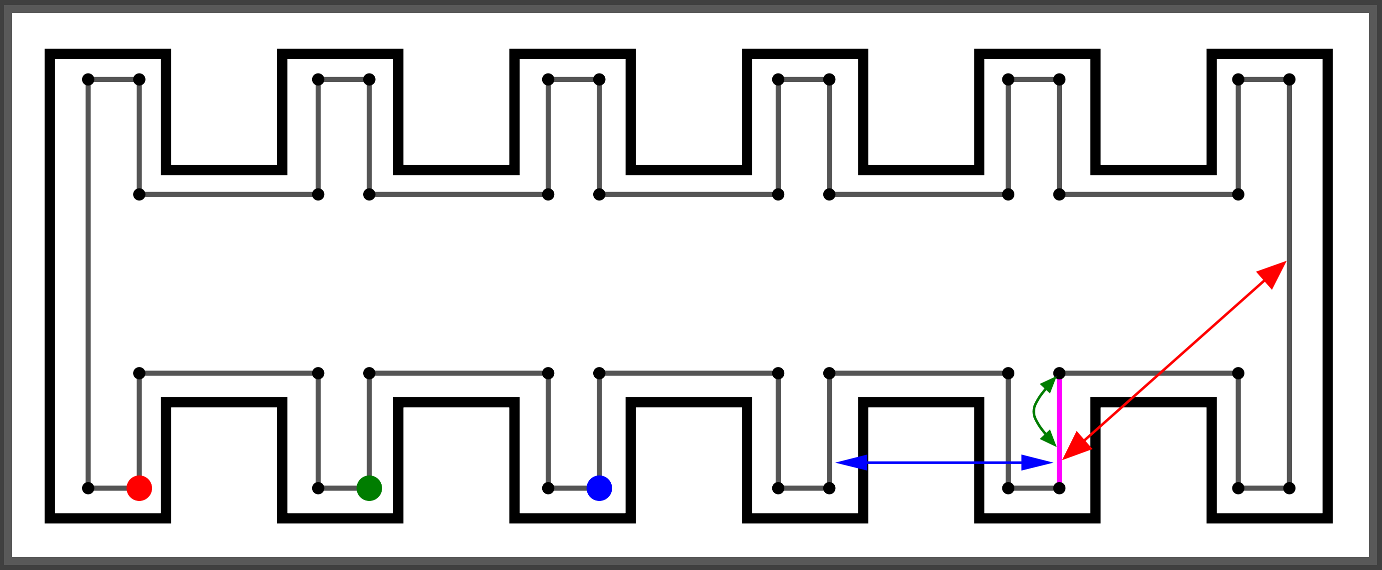


Example: self-similarity



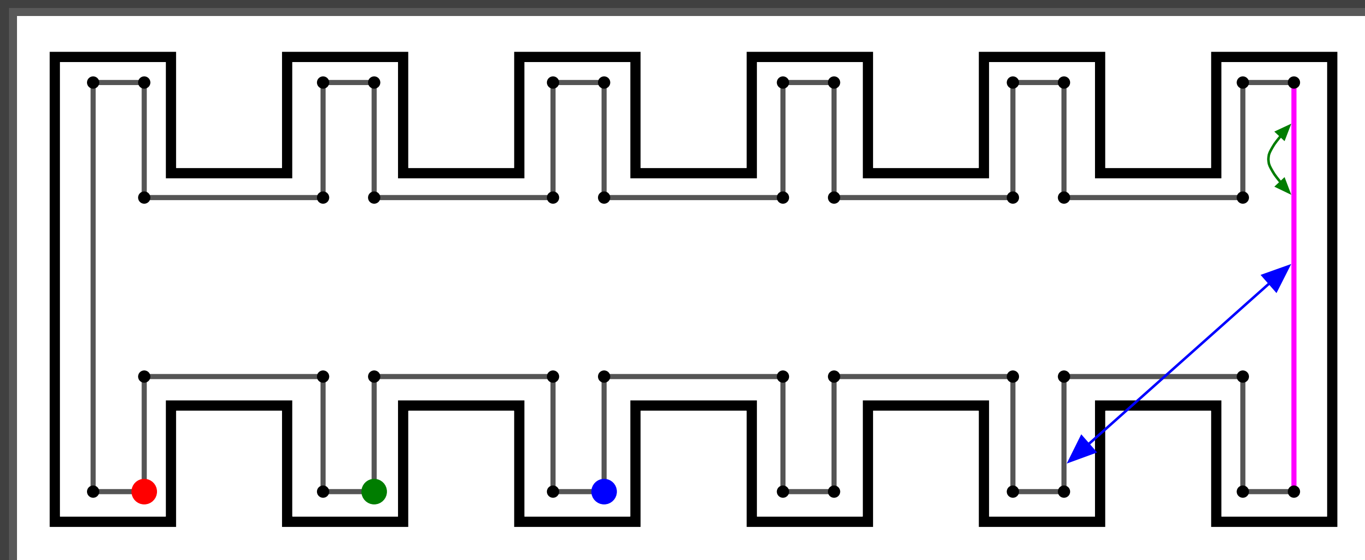


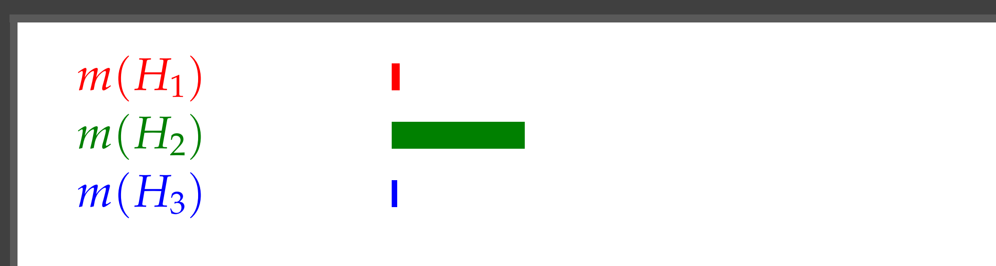
Example: self-similarity



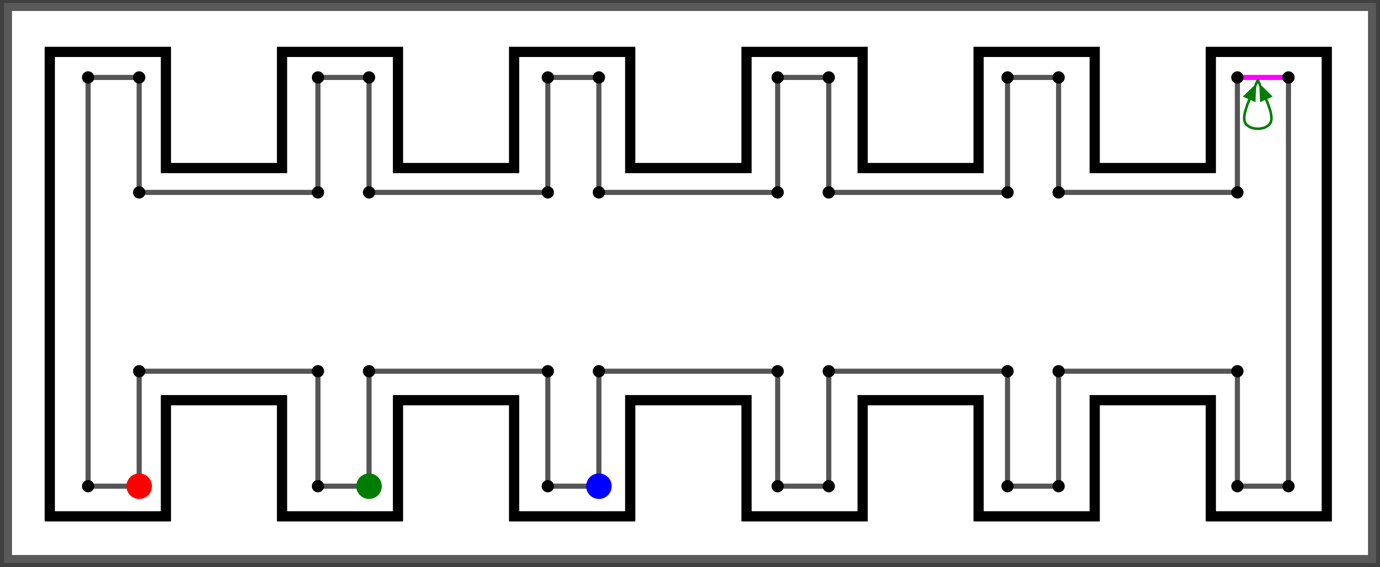
$m(H_1)$		mostly $\longrightarrow m(\{H_2, H_3\})$
$m(H_2)$		
$m(H_3)$		

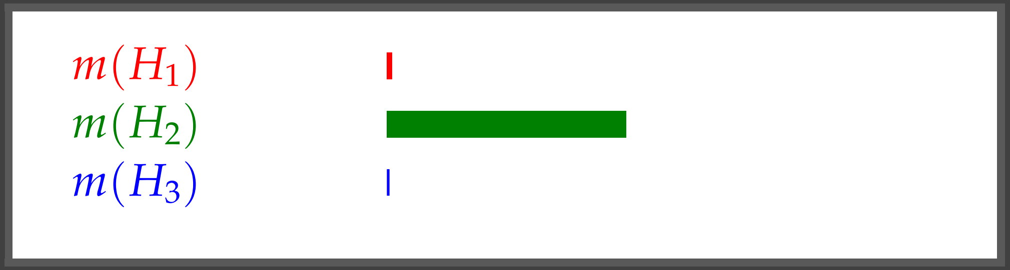
Example: self-similarity



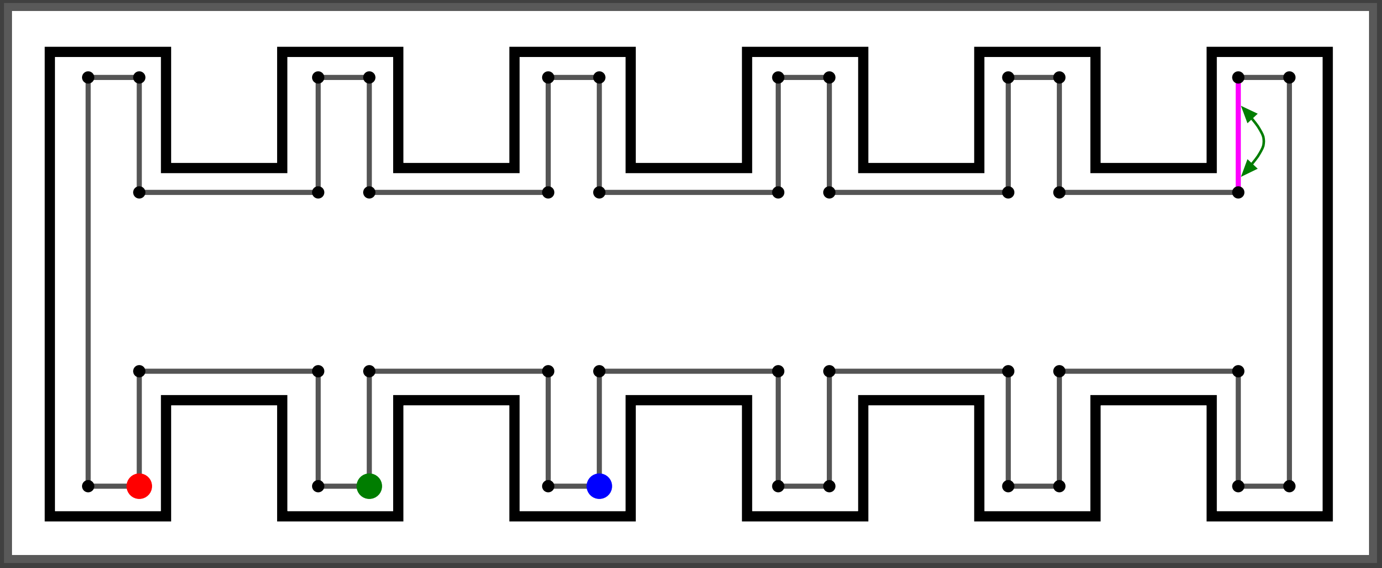


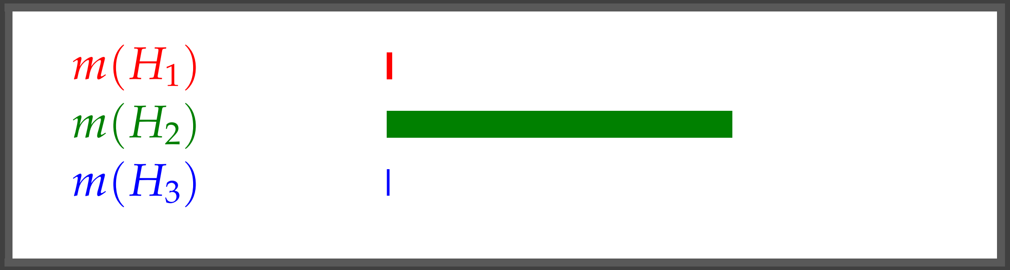
Example: self-similarity



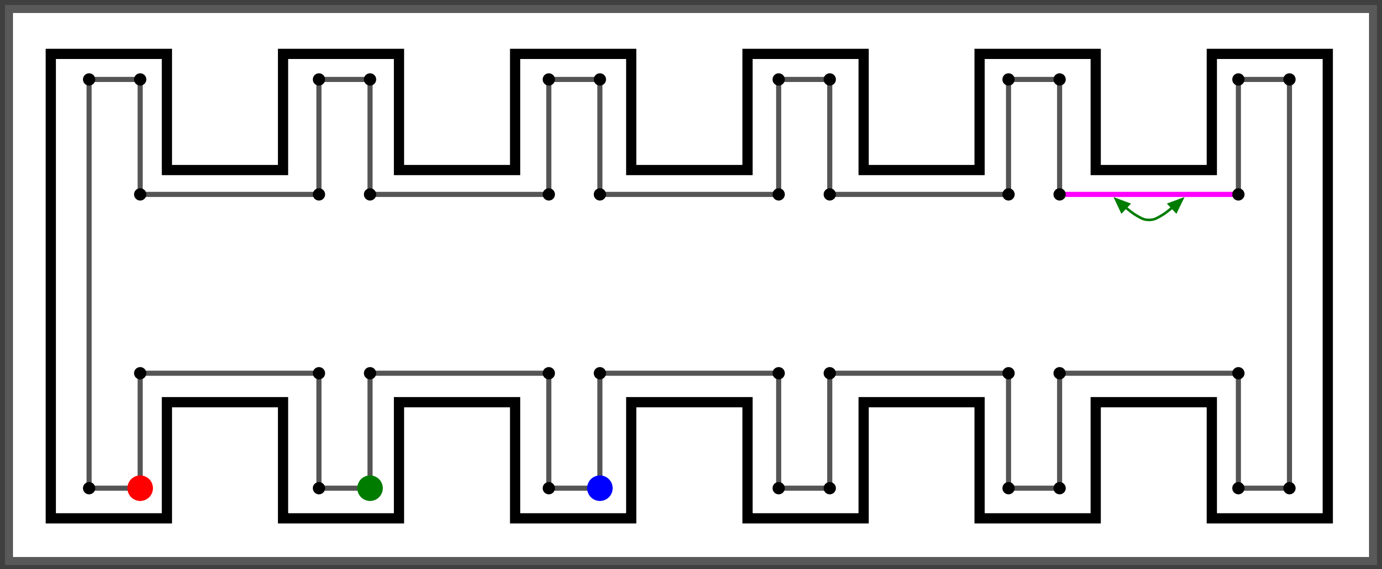


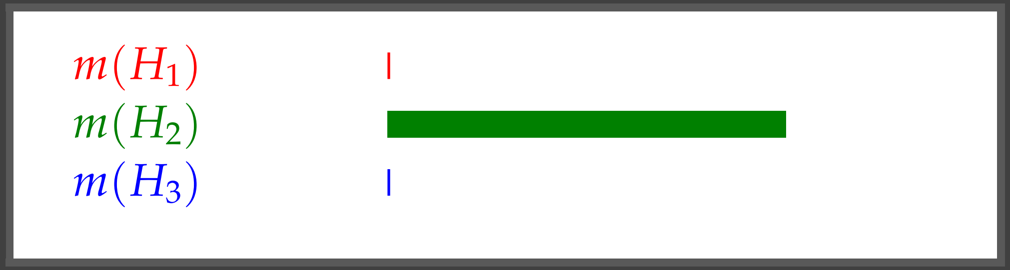
Example: self-similarity



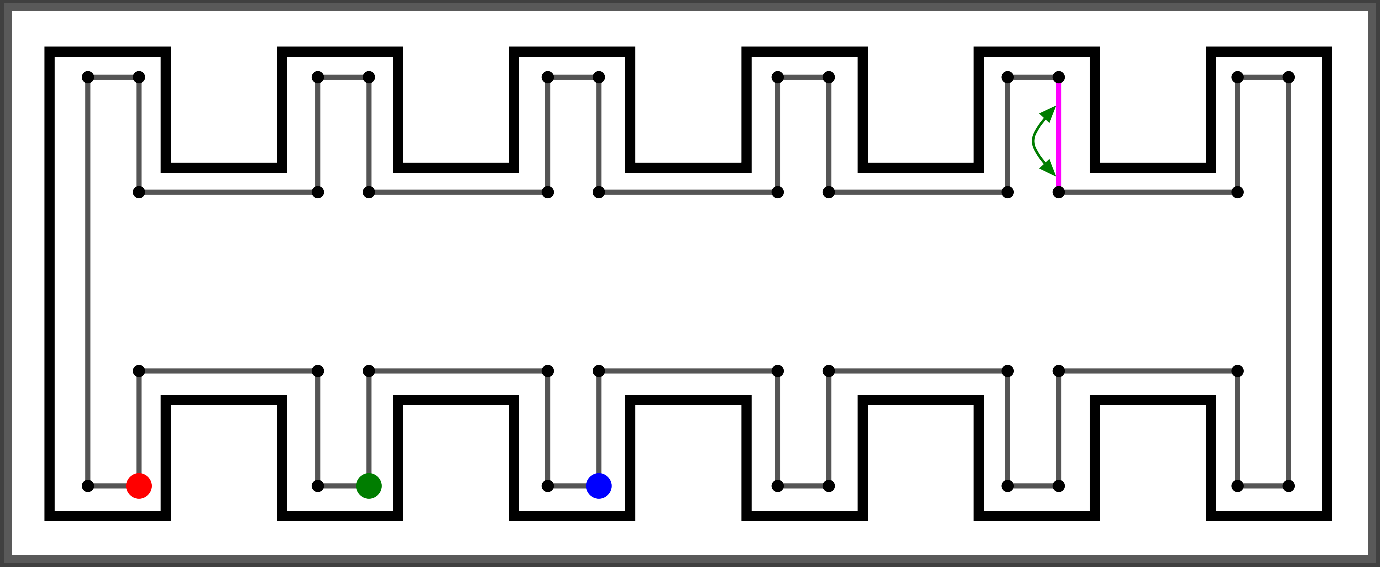


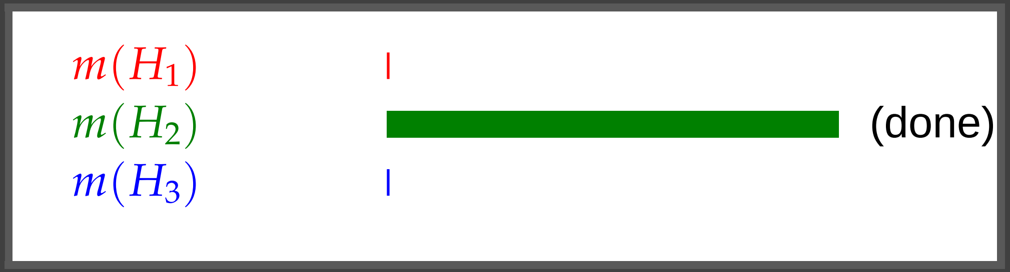
Example: self-similarity





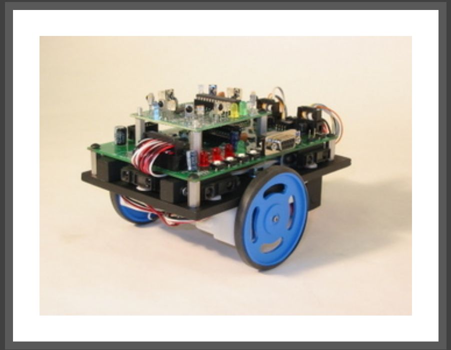
Example: self-similarity

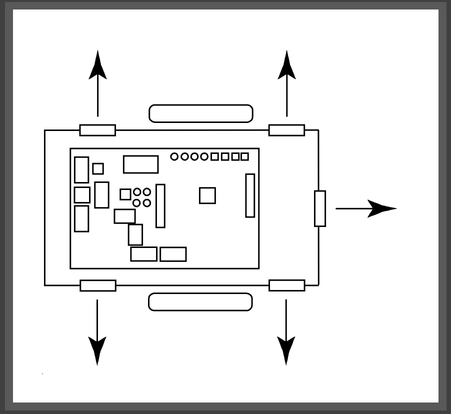




Results

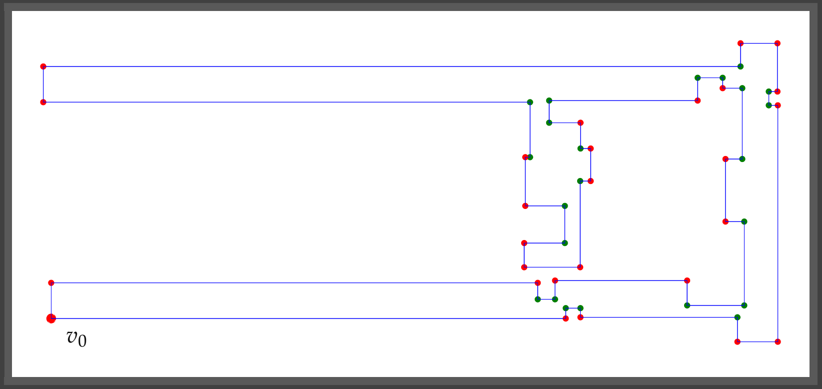
- Simulated experiments: used mainly to test difficult scenarios such as highly self-similar environments
- Real-world experiments:
  - $\hookrightarrow$ Hand-made and unmodified building environments
  - $\hookrightarrow$ Mapping: wall-following strategy, nodes $\equiv$ corners
- Making hypotheses: compute confidence bounds using 0.99 confidence limit
- Loop-closing decisions: choose H_k when $m(H_k) > 0.99$

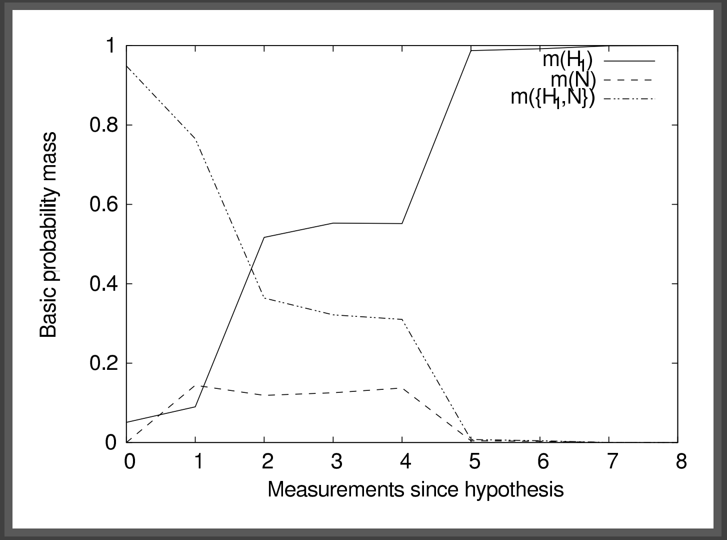




Amos Eaton Building

- Map of the first floor of an academic building (Amos Eaton) at RPI
- Correct hypothesis was the first hypothesis, confirmed within seven measurements
- **Overall results** (simulated and real-world): correct loop-closing decision made 98+% of the time, 99+% in non-pathological environments





Conclusions

- Contributions:
  1. Decision-theoretic approach to closing loops in topological maps using only odometry
  2. Method for modifying Dempster-Shafer frame of discernment whenever a new hypothesis is discovered
  3. Method for computing a BPA reflecting belief in each hypothesis given evidence
- Results: works well (98+% accuracy) in both simulated and real-world tests

Muchas gracias.

Questions?