15-780 Grad AI – Spring 2010

Data Driven Improvements to RRT

Filipe Militão, Karl Naden, Bernardo Toninho

Motivation

- Strengths of RRT
  - Simple
  - Fast
- RRT Weaknesses
  - Not robust – must be tuned to the world
  - Gathers no data about the world

- Question:

  Can we improve on RRT's weaknesses without sacrificing its strengths?

Extension Length

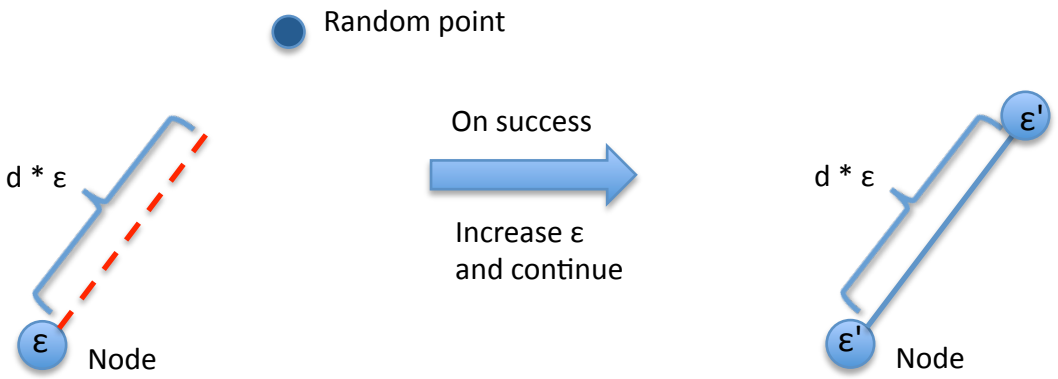
- Prevents robustness
  - World with large open space and small passages
- Possible data collection
  - Each extension success and failure provides data about good extension lengths

Specific Goals

- Gather data about extension length for use
  - For online learning to impact path planning
  - For learning about the world for use in re-planning
- Constraints
  - Data gathering and processing must be lightweight compared to testing the world.

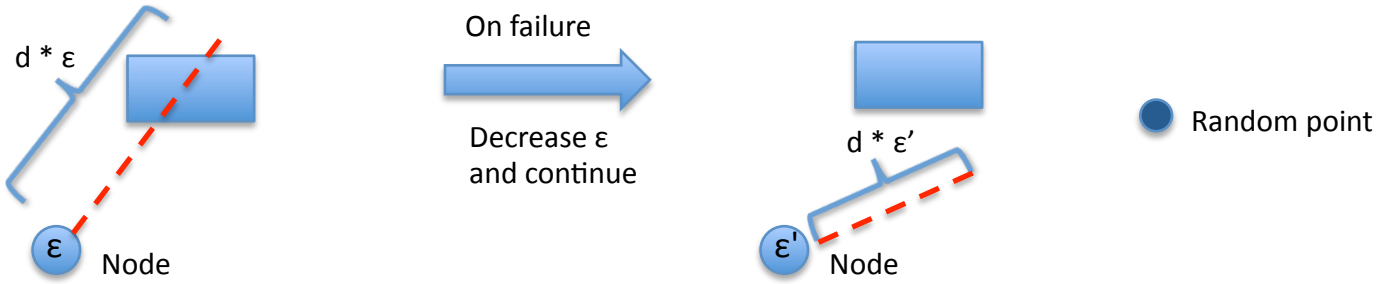
Variable Length RRT (VLRRT)

- Allows for extensions of varying lengths
- Collects *on the fly* information about the world to vary the extension length:
  - Each node has a weight ε that multiplies by the base ext. length
  - Nodes in *potentially* less cluttered regions will have a higher ε value
  - Intuition: Larger steps in uncluttered regions, smaller steps in cluttered ones.
- How can we learn the values for ε at each node?



Variable Length RRT (VLRRT)

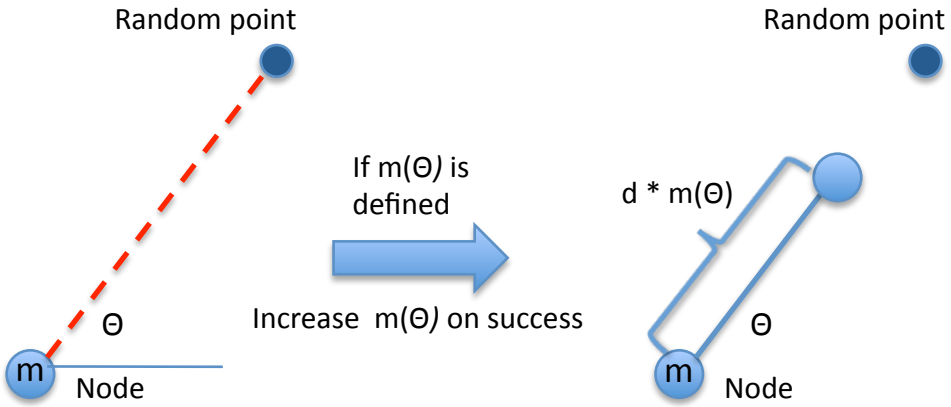
- How can we learn the values for ε at each node?



- If extensions succeed, its more likely that the region is uncluttered and so we scale up ε accordingly (and propagate to the new node).
- If extensions fail, its less likely that the region is uncluttered and we scale back ε.
- Abstracts away much of the topology of the world (can over- and under-estimate!)

Directional VLRRT (DVLRRT)

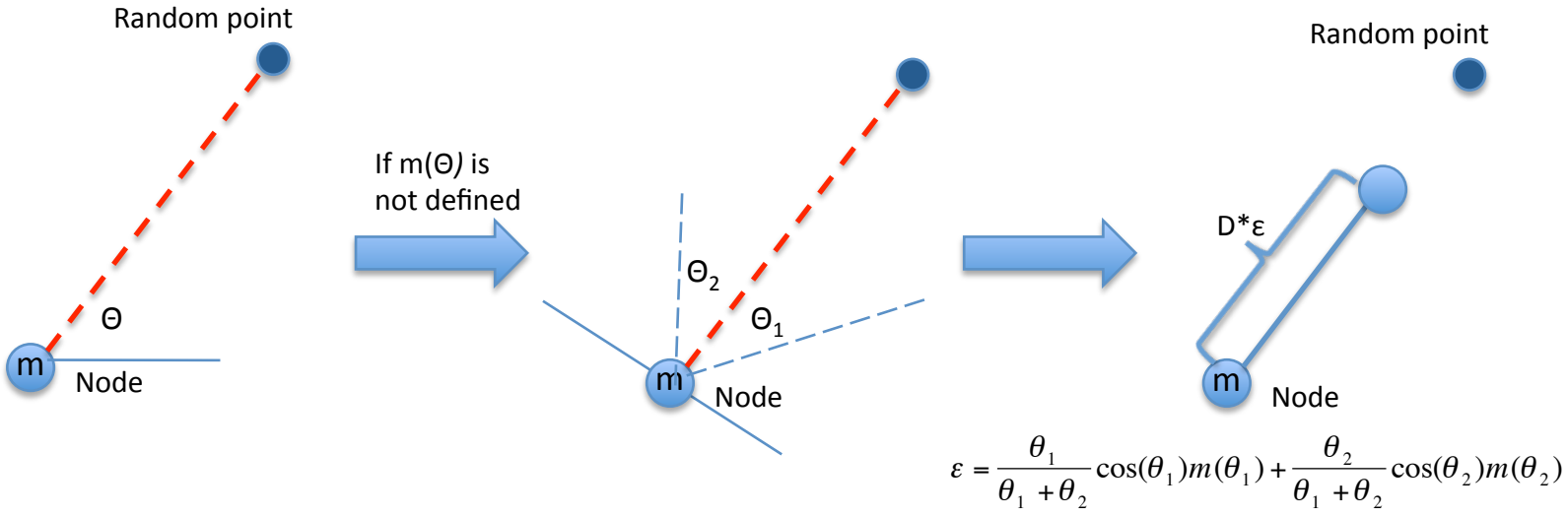
- Builds on the idea of VLRRT by taking into account direction of extensions.
- Nodes no longer have a single ε value, but a *directional map m(Θ)* of values .
- Abstracts away less of the topology to (hopefully) increase accuracy.
- How do we use and populate the directional map?



- If m(Θ) is defined, try to extend by the corresponding value
- Increase m(Θ) on success and decrease on failure.

Directional VLRRT (DVLRRT)

- How do we use and populate the directional map?
- If m(Θ) is not defined, we compute it using the closest values in the map within the [Θ-π/2 , Θ+π/2] range.
- Assign more weight to closer directions (more reliable estimate):



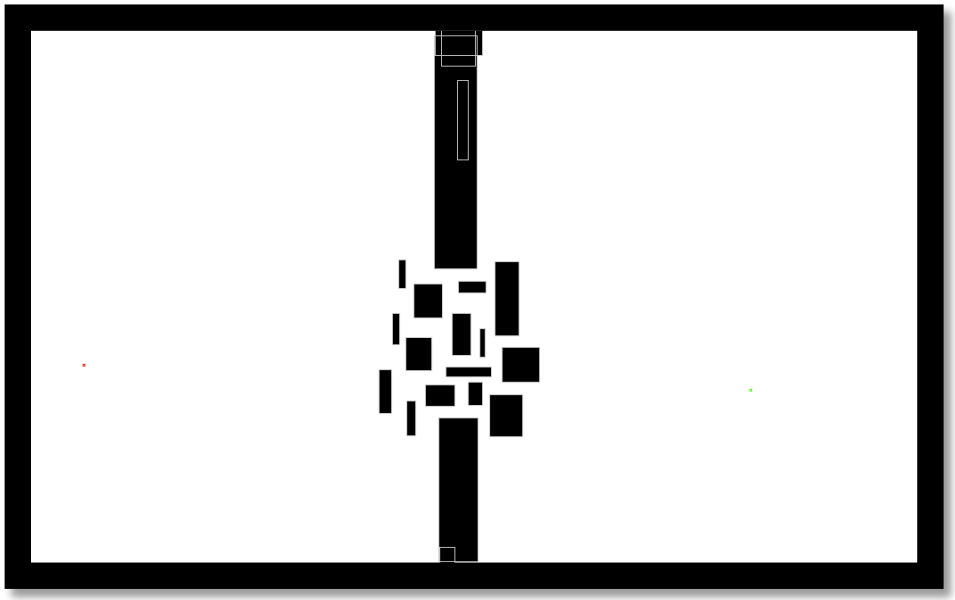
$$\varepsilon = \frac{\theta_1}{\theta_1 + \theta_2}\cos(\theta_1)m(\theta_1) + \frac{\theta_2}{\theta_1 + \theta_2}\cos(\theta_2)m(\theta_2)$$

- If the extension succeeds, scale up m(Θ). Else scale it down.
- New nodes inherit m(Θ) = ε and m(Θ+π) = ε into their directional maps.

Evaluating VL and DVLRRT

- We have yet to specify how to increase and decrease ε values:
  - Empirical results show best performance with a high increase rate and a higher decrease rate.
- We compared the performance of our algorithms with RRT in terms of running time and success in reaching the goal in several worlds:

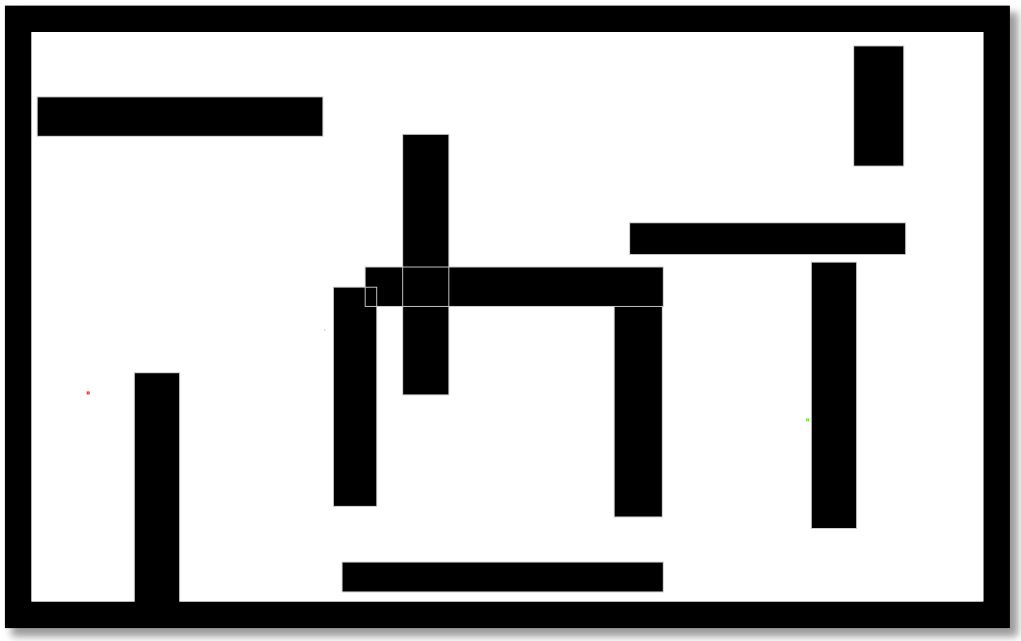
Alg.	World	Avg. Success	Avg Time (ms)
RRT	obstructed	95%	12.0
VLRRT	obstructed	96% (+1%)	13.3 (+1.3)
DVLRRT	obstructed	96% (+1%)	13.4 (+1.4)



Evaluating VL and DVLRRT

- We have yet to specify how to increase and decrease ε values:
  - Empirical results show best performance with a high increase rate and a higher decrease rate.
- We compared the performance of our algorithms with RRT in terms of running time and success in reaching the goal in several worlds:

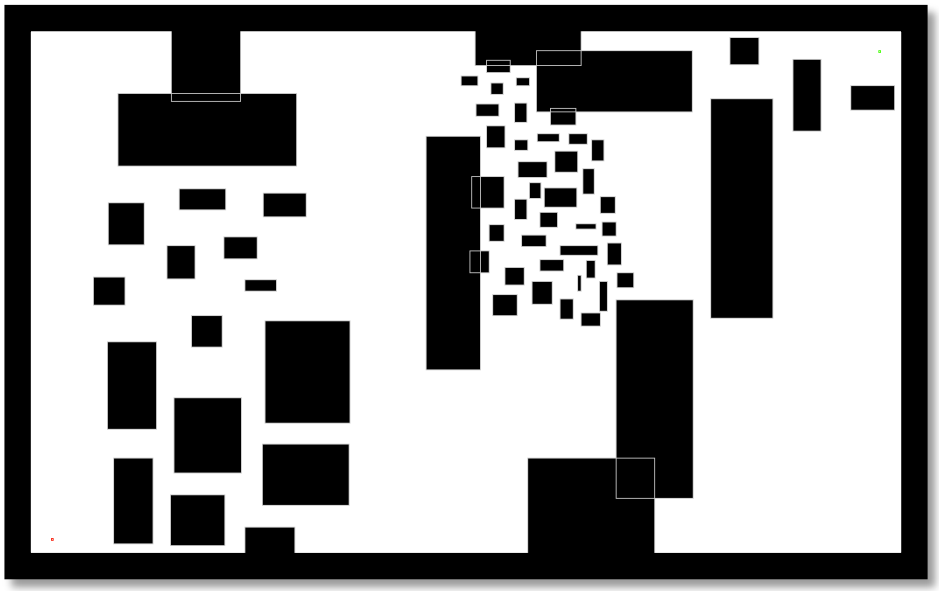
Alg.	World	Avg. Success	Avg. Time (ms)
RRT	maze	91%	17.2
VLRRT	maze	99% (+9%)	10.6 (-6.6)
DVLRRT	maze	99% (+9%)	10.6 (-6.6)



Evaluating VL and DVLRRT

- We have yet to specify how to increase and decrease ε values:
  - Empirical results show best performance with a high increase rate and a higher decrease rate.
- We compared the performance of our algorithms with RRT in terms of running time and success in reaching the goal in several worlds:

Algorithm	World	Avg. Success	Avg Time (ms)
RRT	cluttered	33%	39.2
VLRRT	cluttered	60% (+27%)	33.6 (-5.6)
DVLRRT	cluttered	60% (+27%)	33.7 (-5.5)



Re-Planning

- WORLD CHANGED!
- GOAL: use information gathered on a previous search to try to **improve** the next.

(SEARCH + OLD-INFO) **>** SEARCH **?**

- PROBLEM: competing against just plain ("uninformed") search.
  - Extracted information must do better than using that time for plain search.

Re-Planning

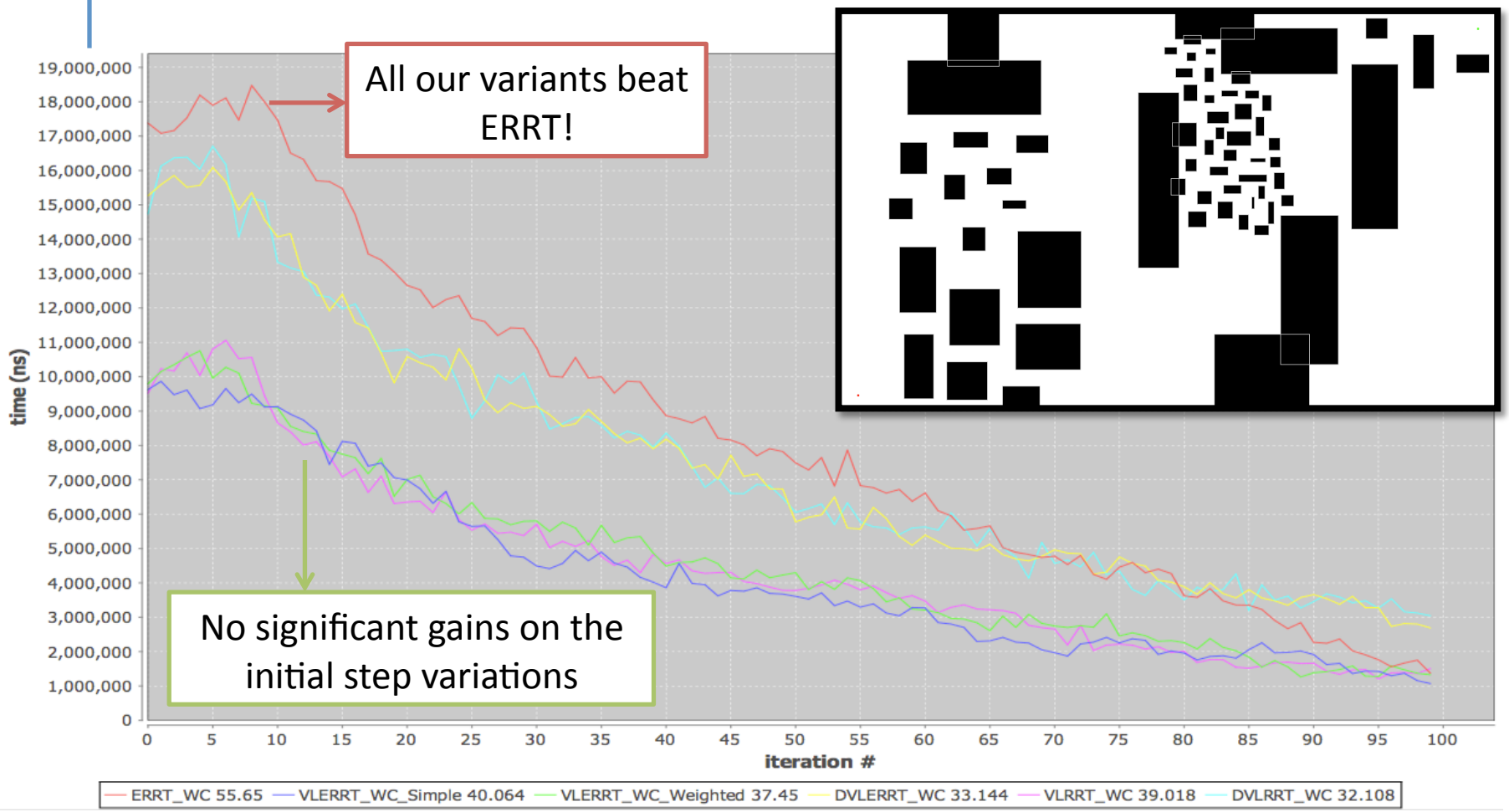
- EXTRA INFORMATION USED:
  - Previous search tree: **initial step size**
    (memory not an issue to us)
    Use the *N* closest neighbors in the old tree:
    - (VLERRT) Simple average: $\varepsilon = \frac{\sum n_\varepsilon}{n}$
    - (VLERRT) Weighted average: $\varepsilon = \frac{\sum n_\varepsilon * d_n}{n * \sum d_n}$
    - (DVLERRT) Pull from neighbors.
  - **Waypoints** (fixed number, picked from previous search solution):
    - Random from previous path (ERRT)
    - Bias towards less dense nodes

Testing - Initial Step

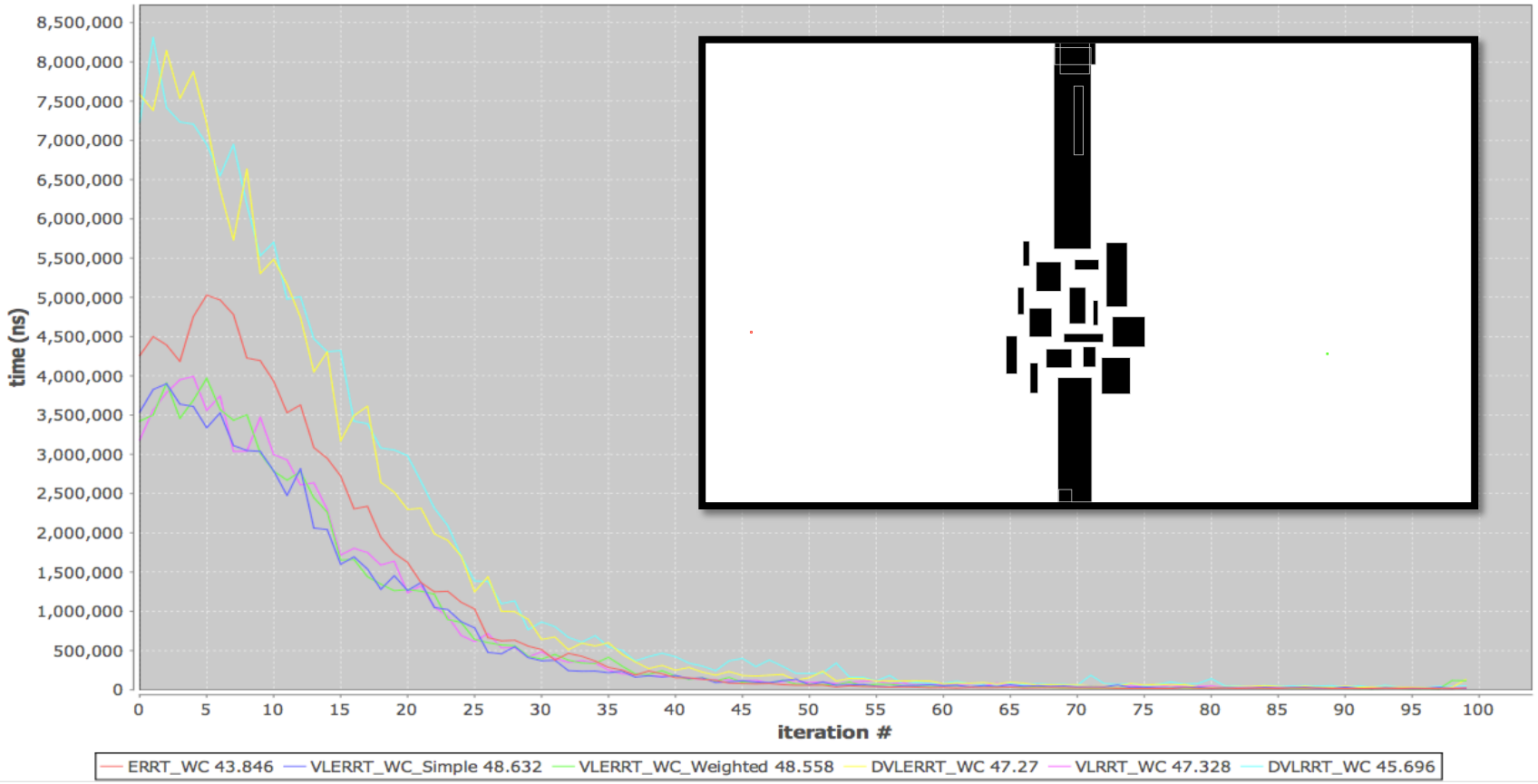
- CHANGES TO TESTING:
  - World changes (small random movements in the obstacles)
  - Start moves a bit towards the goal (using the path that got closest to it)
- TESTING METHODOLOGY:
  - Average 1000 runs each with 100 iterations (moving towards goal) using 50 ms time slice.
  - Compare results for all combinations of information.

Results: CLUTTERED

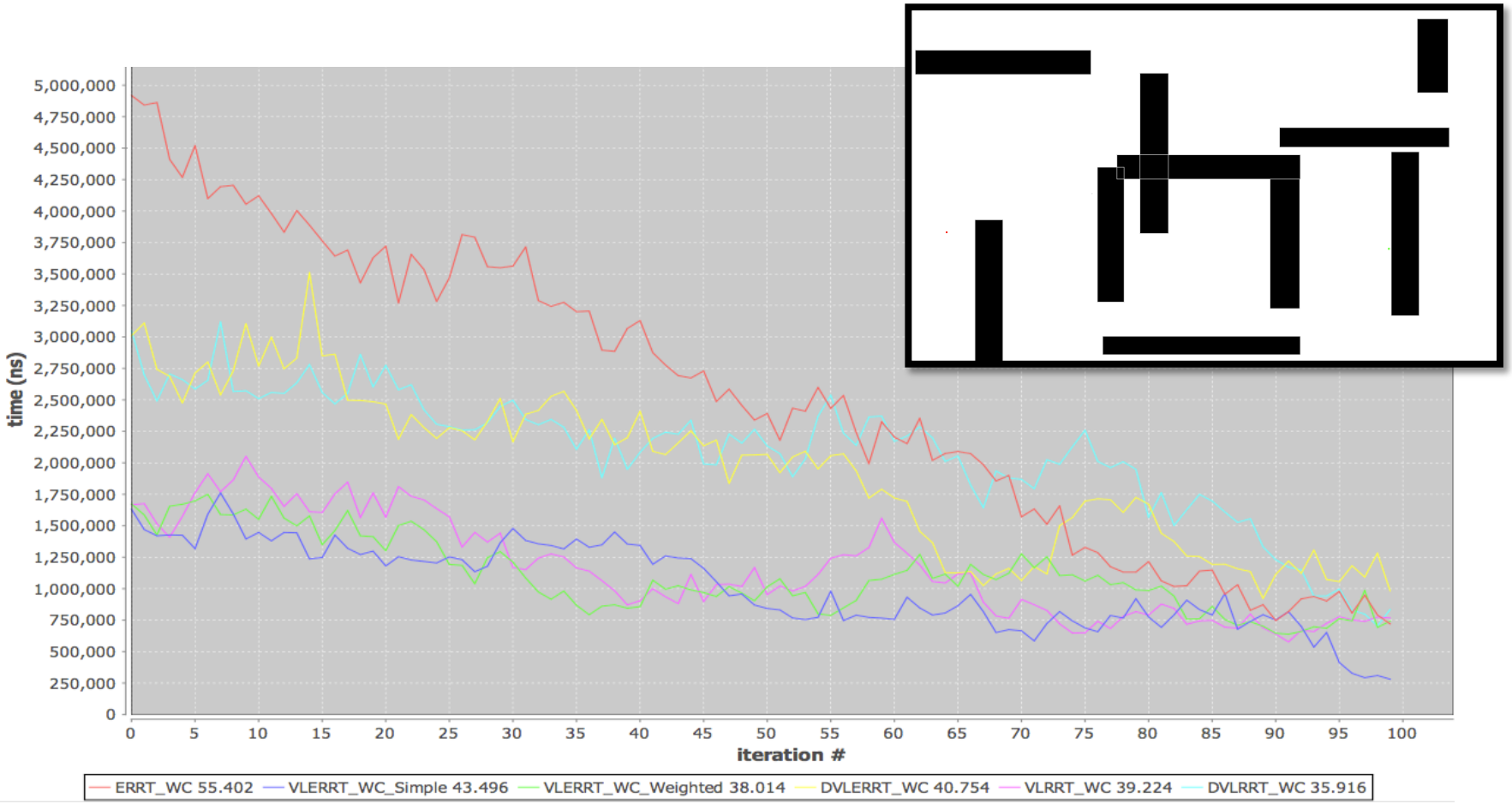
Run Time
(< 50 ms → goal)



Results: **OBSTRUCTED**

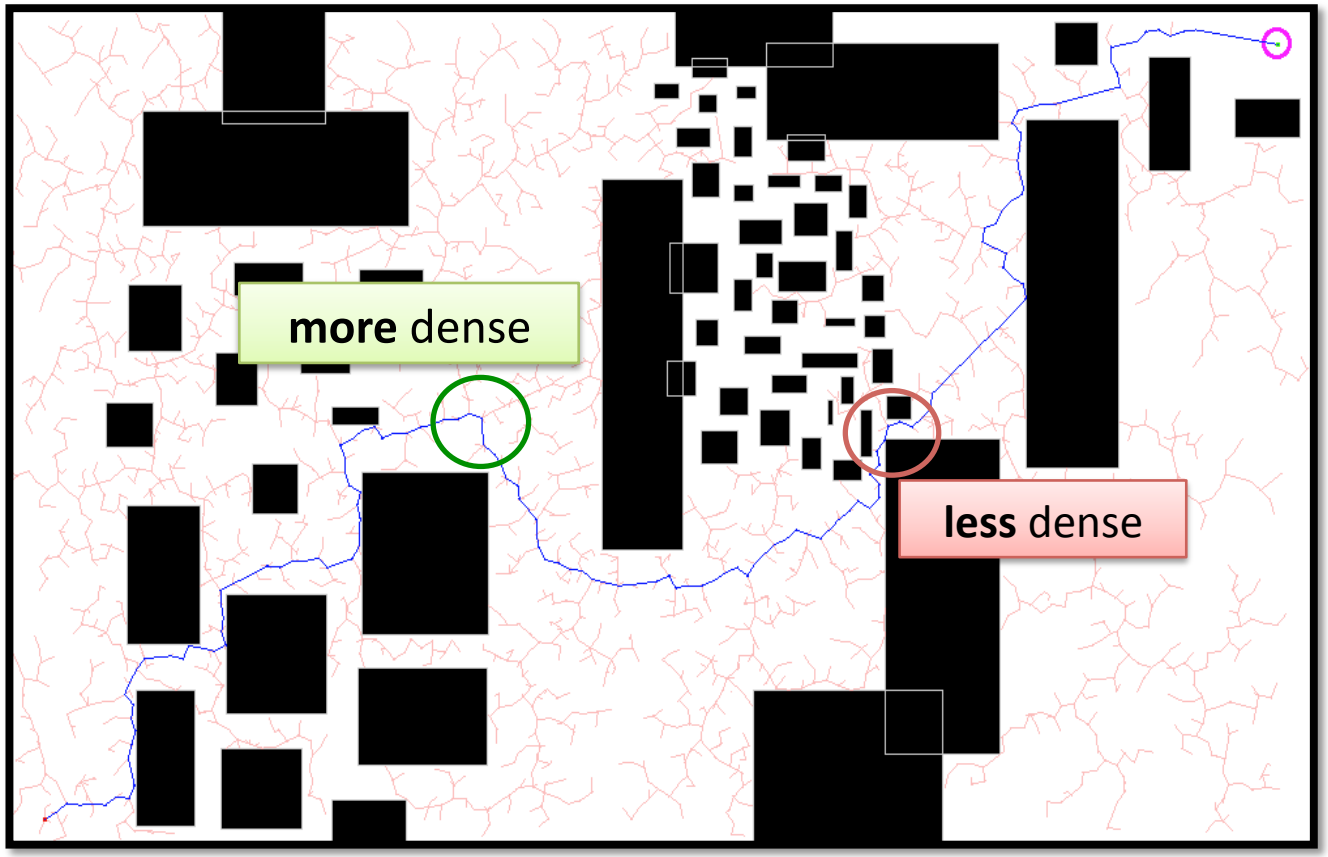


Results: MAZE



Density based Waypoints

- Bias waypoint generation towards *useful* points:
  - a less dense node (with fewer neighbors) is likely to be more useful.



Conclusions

- Online learning helpful
  - + Increased robustness
  - + Decreased planning iteration time
  - – At the cost of Increased path length
- Reusing the data for re-planning more difficult
  - – Online learning provides the same information
  - – More precise calculations no different from consulting the world