DART: A Particle-based Method for Generating Easy-to-Follow Directions

Robert Goeddel Edwin Olson

***Abstract*— Despite evidence that human wayfinders consider directions involving landmarks or topological descriptions easier to follow, the majority of commerical direction-planning services and GPS navigation units plan routes based on metrically or temporally shortest paths, ignoring this potentially valuable information. We propose a methodo for generating directions that maximizes the probability of a human arriving at the correct destination, taking into account a model of their ability to follow topological, metrical, and landmark-based directions. We discuss optimization techniques for employing these models and present a method, DART, for extracting model-improved sets of directions in a tractable amount of time. DART employs particle simulation techniques to maximize the probability that the modeled wayfinder will successfully reach their destination. Our synthetic evaluation shows that DART produces improvements in arrival rates over existing methods and illustrates how DART's directions reflect properties of the wayfinder model.**

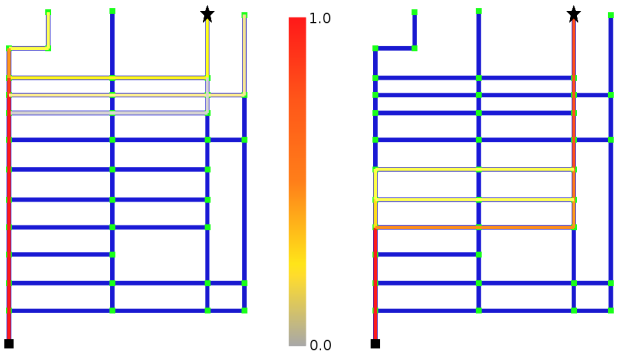
I. Introduction

Effectively directing a human to a particular destination is a difficult task requiring clear and concise directions. Predicting what directions will be most useful is challenging, but generative models offer a powerful tool towards accomplishing this goal. By using a generative model to describe the potential actions made by the *wayfinder* (the person following the directions), we can compute plans based on predictions about those actions.

Modern GPS navigation systems on phones and deployed in cars typically optimize over travel time or distance traveled instead of focusing on the ease of following the directions. Verbal turn reminders and visual depictions of the actions in question mitigate the difficulties in following these routes that may arise from sub-optimal routing through confusing areas. Easy-to-follow plans should be robust to lacking these cues and can be applied to GPS denied situations or to other un-instrumented environments.

Take, for example, a regional hospital. These hospitals are often large and full of difficult-to-navigate hallways. Hospitals typically have staffers handle queries and direct people to their destinations. However, this could be done efficiently and effectively by guide robots stationed at the entrances, as well. Such a system could return information about the locations of patients, particular wards, or the offices of certain doctors. Indeed, the only missing piece is the ability to generate a clear set of directions about how to get to the location of interest.

The authors are with the Department of Computer Science and Engineering, The University of Michigan, Ann Arbor, MI, {rgoeddel, ebolson}@umich.edu, http://april.eecs.umich.edu



(a) Bad shortest path directions (b) Key (c) Best shortest path directions

Fig. 1: Simulated wayfinder routes for two sets of shortest-path directions. Paths taken by simulated wayfinders for two sets of shortest-path directions are shown, with colors shown in (b) denoting the percentages of simulated wayfinders traversing a given path segment. The maximum likelihood set of directions may still result in many wayfinders deviating from the specified route. By exploiting knowledge of the wayfinder model to select error tolerant directions, (c) guides more wayfinders to the goal than (a).

In this paper, we will motivate and formulate a method for incorporating generative models of wayfinders into the direction-planning process. We will discuss scalability issues and propose optimization techniques for tackling these problems. Finally, we will present a method we call **D**irection **A**pproximation through **R**andom **T**rials, or **DART**, that uses particle-based estimation techniques to find effective sets of directions between points in a map. Action costs are determined by a generative model describing the wayfinder and the environment, rather than a fixed action cost DART approximates the search for the maximum likelihood solution to the problem, rapidly providing paths that take advantage of landmarks and other environmental structure to guide the modeled wayfinder to their goal. Our main contributions are:

- The formulation of the wayfinding problem incorporating a generative probabilistic model capable of approximating human wayfinding capabilities,
- Strategies for optimizing depth-first search in the context of wayfinding,
- DART, a tractable, particle-based method for computing directions optimized based on a supplied generative model of the wayfinder and the environment, and
- Evaluation of DART's performance and a framework for evaluating the effectiveness of a set of directions.

II. Related Work

A. Understanding Directions in the Context of Environment

Extensive effort has gone into studying the process of constructing "good" directions. Lynch contributed early thoughts to the literature in his book, *The Image of the City* [1]. Lynch observed that knowledge of environmental structure plays an important role in wayfinding.

Kuipers explored the cognitive maps of humans and how we navigate through the world [2]. With TOUR, he modeled human understanding of their surrounding environment as well as the process with which they navigate through that environment. Others have explored internal world representations and their impact on giving and following directions, finding that, even given human weaknesses in preserving spatial information, efficient and effective direction generation is closely linked with strong spatial abilities [3], [4].

MacMahon created MARCO, an agent designed to follow natural language directions [5]. MARCO models human reasoning about verbal directions, which are often unclear or incomplete. Particularly for directions considered "high quality" by human evaluators, MARCO was able to follow these directions with near-human consistency. Models such as MARCO enable accurate evaluation of direction quality.

B. Wayfinder Abilities

People have been shown to process different types of directions with varying levels of quality. Though humans have a tendency to give street directions in terms of "go N blocks and turn," these directions are the easiest for people to follow [6]. It has further been shown by Hund and Minarik that wayfinding abilities vary noticeably with gender, with men typically navigating more quickly than women [7].

Metrically based directions are popular in GPS navigation systems and in other commonly used routing tools, but humans have been shown to be inaccurate estimators of distance. Cohen et al. showed that geography plays a role in distance estimation [8]. Thorndyke showed that distance estimation error could be tied to the amount of clutter in the environment [9].

Landmarks have been shown to be useful as navigational aids. Directions often use andmarks to guide followers through tricky regions or to help localize them, and extensive work has been done to determine when and why such landmarks are useful [10], [11], [4]. Further consideration has been given to identifying landmarks and automatically extracting them from the environment [12], [13].

C. Route and Direction Generation

Elliot and Lesk studied how to replicate human direction giving abilities [14]. The authors observed that humans followed a guided depth-first search strategy, and they were able to generate similar paths through the use of a heuristically guided depth first search penalizing turning. Duckham and Kulik created a system for generating "simplest paths" based on a metric for evaluating the cost of different turn actions (turning at a T-intersection is typically better than turning at a 4-way intersection) and finding the path that minimizes this cost [15]. This technique was further modified to deal with complex intersections, avoiding such areas while attempting to keep path lengths short [16].

Richter and Klippel defined a strategy for generating what they call "context-specific route directions," directions which use knowledge of the structure of the environment to come up with easy-to-follow directions [17], later implementing this in the form of GUARD, a method for generating such directions [18]. GUARD considers turn distinctions (right vs. slight right) as well as landmarks in the environment to describe a given path in the best way possible. Later, Richter and Duckham created a method for generating the "simplest instructions" [19]. This method combines Richter's GUARD method with cost metrics used by Duckham and Kulik in computing simple paths to create more effective sets of directions. In doing so, the technique not only finds a simple path through the environment, but also considers how the description of that path will affects followability.

With the exception of the "simplest instructions" algorithm, the main weakness of the previous work is the separation of path selection from path description. These processes are *not* independent. For example, the "simplest path" may still not be as easy to describe as a path with a few more turns but many landmarks along the way.

Richter and Duckham address this concern, incorporating additional cost metrics to take into account what they refer to as the *cognitive cost* of actions. As a result, both the simplicity of the path as well as the ease of interpreting directions along that path are considered when choosing directions. However, the algorithm does not attempt to deal with ambiguous descriptions of the environment. Our proposed method replaces the concept of cognitive cost with weight based on expected success rates for following given directions. As a result, properties such as environmental ambiguity are automatically incorporated into the weights.

III. Incorporating Wayfinder Models Into Planning

Evaluating successful arrival at the goal involves not only understanding how well people are able to follow the specified path, but how well they are able to recover when they deviate from it. Using a generative model offers a principled approach to evaluating the cost of an action. Given a set of directions, we can sample from our model (representing the distribution $p(x|d)$, or wayfinder position given a direction) and observe the resulting trajectories. From these observations, we gain situation-specific knowledge about the effectiveness of particular directions in our environment.

We formulate the direction finding task as an optimization problem. We assume that we are given a model m consisting of a map of the world and generative model that allows us to predict wayfinder actions for certain instructions. Then, assuming the wayfinder starts at position x_{start} and ends at x_{final}, our goal is to select a set of directions d maximizing the probability that the wayfinder reaches their goal:

$$\arg\max_{d} p(x_{\text{final}} = \text{destination}|d, m) \tag{1}$$

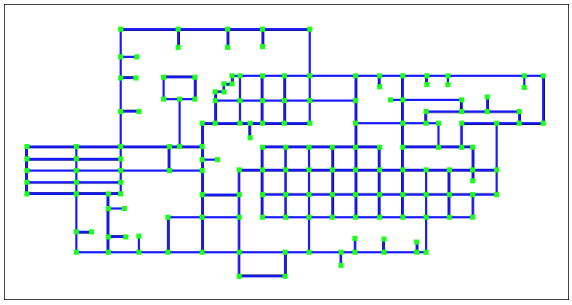


Fig. 2: A hand-constructed map for simulated testing. This map covers an area roughly 11 km $\times$ 5 km and contains 167 nodes and 135 landmarks. Landmarks were generated and placed randomly from a family of 15 unique landmarks.

Particle simulation has proven to be a useful tool for navigation problems [20]. We employ it here to estimate the distribution of wayfinder positions given a set of directions. A naïve algorithm to find the best directions could generate every possible set of directions and run particle simulation for each set. The best directions would be the set for which the most particles arrived at the goal.

Fig. 1 shows an example of how such a technique can take advantage of a model to pick the best directions. In this environment, we are attempting to navigate from the black square in the lower left to the black star in the upper right. The last two directions of a plan can be described as "Go until you stop at a T-intersection and turn left," and then "Go until you reach a dead end and you will have arrived at your destination," regardless of which of several horizontal crossroads you pick.

Fig. 1a shows the paths followed by simulated wayfinders for a set of directions following a shortest path. By chance, this happens to guide us to one of the aforementioned crossroads. However, due to the distance traveled to this crossroad and lack of good landmarks, many of our particles get lost.

A more complete search of the space based on our model turns up the directions highlighted by Fig. 1c. Many of the simulated particles make mistakes (in fact, the percentage of simulated wayfinders following every direction perfectly does not vary greatly between the two plans), but knowledge provided by our model allows us to position our plan among several acceptable crossroads so that erring wayfinders that turn late are more likely to turn onto a road that still allows them to correctly execute the later instructions. As a result, twice as many particles are able to navigate to the goal in Fig. 1c as in Fig. 1a.

A. Simulation and Model Formulation

We employ simulations, which allow us to rapidly evaluate the effects of employing different models as well as greatly accelerate and simplify the data collection process. To facilitate rapid implementation and testing, test environments were limited to planar graphs with only right angle turns (see Fig. 2 for a sample instance). For evaluation, three main environments were created by hand and randomly populated with landmarks, as well as a set of four environments used for collecting additional timing data. We assume that landmarks only affect one known decision point and that the direction from which we approach the intersection does not affect the usefulness of the landmark. As results were consistent across environments, we limit the presentation of results in this paper to the map seen in Fig. 2.

We employed four different categories of directions that our modeled wayfinder can understand. These were:

- METRIC: Go X m and turn
- INTERSECTION: Go to the X^{th} intersection and turn
- LANDMARK: Turn when you see *landmark* X
- GO UNTIL: Go in this direction until forced to make a choice and turn

We created a simple model based on previous literature in which humans become increasingly poor at counting intersections/measuring distances as the numbers in question become large. Conversely, LANDMARK and GO UNTIL directions were modeled as remaining robust regardless of distance, as previous research indicates that people are much better at following these types of instructions. We also represent the limitations of human memory with a fixed probability for which any given direction might be "forgotten," implicitly penalizing longer direction sets as being more difficult to follow.

To ensure that the directions generated were actually influenced by the model, we repeated tests with perturbed variations of the model. We expect, for example, that dramatically reducing the quality of LANDMARK directions should result in plans using them less often. Further discussion and results from these tests will be presented in Sec. IV-B. We also present results on the arrival rates of the wayfinder with varying direction-giving strategies. These numbers are not presented to show that our model is the state-of-the-art (it is not), but rather that the addition of a model to planning *improves* wayfinding performance.

B. Finding Directions Maximizing Arrival Rate

To find the most effective set of directions for getting the wayfinder to the goal, we search through the space of possible directions, estimating the probability of reaching our goal using particles. If a particle makes a mistake that leaves it unable to execute the next direction, we consider this particle "lost" and mark it so. Otherwise, we continue simulating the particle's decisions by sampling randomly from the distribution of possible actions defined by our model. We score a set of directions based on how many particles finish at the goal.

Searching for the maximum likelihood plan is challenging, though, given the variety of space and computation complexity issues that arise. The number of decisions to make from any given intersection is generally larger than just the node degree $\times$ the number of possible direction types (already a distressingly large branching factor). Directions do not merely describe how to get to immediately adjacent intersection, but possibly to a location many intersections away. This large branching factor makes breadth-first search

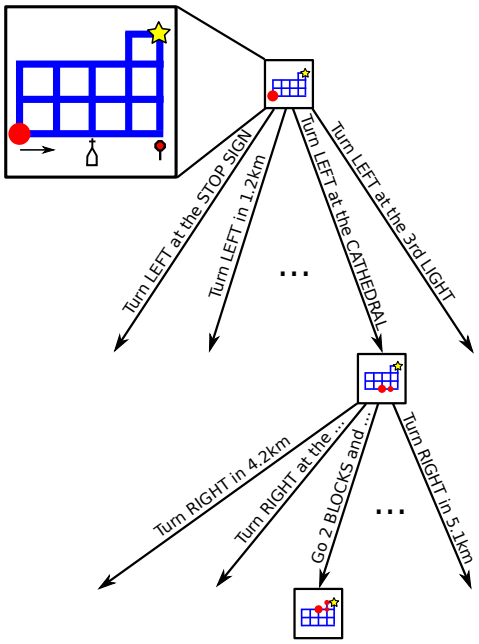


Fig. 3: A small portion of a depth-first search through direction space. The distribution of particles through the environment is shown at several of the decision steps as red circles, with larger radii corresponding to higher particle densities.

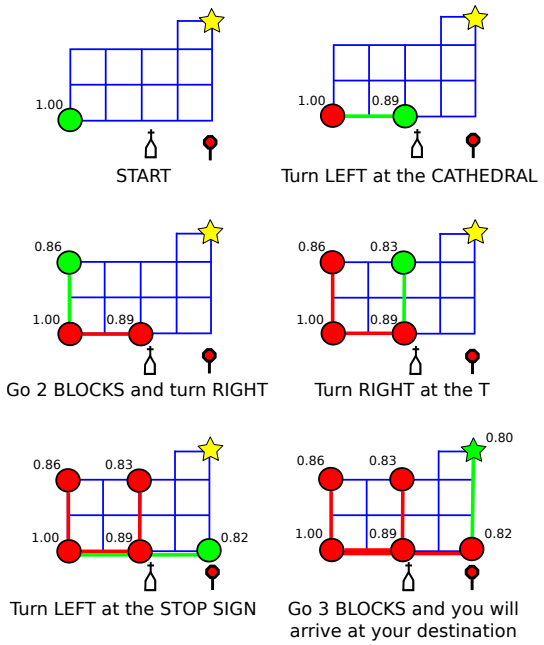


Fig. 4: An small example of a DART search. At each step, DART expands the node with the largest number of on-track particles (depicted in green) and adds it to a closed-list (depicted in red). When the goal node is expanded, the search ends.

(BFS) intractable, particularly in regards to memory usage, so we employ an iteratively deepening depth-first search (DFS) to determine the best set of directions, a small portion of which is depicted in Fig. 3.

Our algorithm can be summarized:

1) Initialize a "plan" with many particles and an empty set of directions and add it to a stack.
2) While the stack still has entries, pop the top entry and consider all possible, non-backtracking directions that can be taken from the current position as defined by the model.
3) Make copies of the plan for each possible direction and simulate particles following that direction. Now the particles form some new distribution of positions and states in the world.
4) Sort the plans by distance of theoretical position to the goal and, if they do not violate our "depth" constraint, add them to the stack.
5) If iteratively deepening, repeat until a plan is found, increasing the search depth as necessary.

Some pruning may be performed by tracking how many of the particles are "lost." If the number of lost particles drives our total potential success rate below the minimum search depth, we prune out that plan. Likewise, if we find a solution reaching the goal with a success rate exceeding our minimum search depth, we may shift the minimum depth upwards to reflect this knowledge.

Runtime increases proportionately with the number of particles simulated, but the depth of the search is dependent on the structure of the map and the wayfinder's abilities. However, if we assume that the maximum search depth is d and our model can be applied to a direction in constant time, then for number of particles P and branching factor b, the worst-case asymptotic runtime of our DFS is $\mathcal{O}(Pb^d)$. As will be discussed in Sec. IV, we found that, even with our optimizations, the search times for non-trivial sets of directions were too long to be of use, indicating that our pruning is unable to substantially reduce the search space.

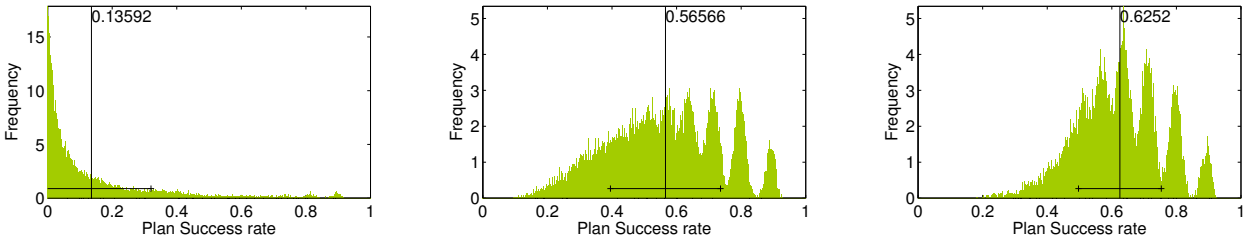
C. Direction Approximation through Random Trials

Since we cannot always afford to compute the best possible set of directions for an environment with our DFS method, we instead seek to quickly approximate the best directions while still making use of our model. One of the main difficulties with optimizing the depth-first search is that plan quality does not decrease monotonically as the search progresses. Though a particle may not be on the directed route at a given step, this does not preclude it getting back on course and successfully reaching the goal. As a result, we are conservative when eliminating potentials plans from consideration. For DART, we take an aggressive approach, treating particles that are off course as irrecoverable when evaluating current plan quality. We make this decision based on the belief that situations in which the environmental structure allows particles to recover are rare. Thus, the losses in plan quality are negligible. Based on these assumptions, we may now build a more rapid search.

Given this assumption, the problem is greatly simplified.

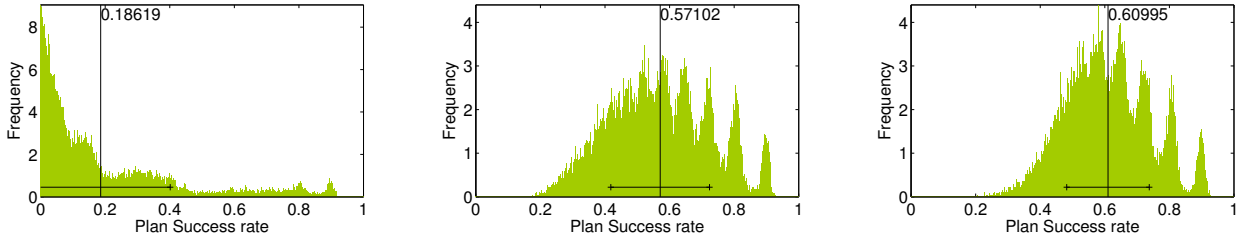
$$p(x_{\text{final}} = \text{destination}|d, m) \approx \prod_{i=0}^{|d|} p(d_i|m, d_{i-1}) \qquad (2)$$

In (2), $p(d_i|m, d_{i-1})$ is the probability of executing the i^{th}



(a) METRIC-only shortest-path (b) Model-optimized shortest-path (c) DART

Fig. 5: Histograms of successful arrival rates for METRIC-only shortest-path, model-optimized shortest path, and DART best path directions for every pair of nodes in the map depicted in Fig. 2. The mean is denoted by a vertical black line, with horizontal bars extending out one standard deviation. Area under the curve sums to 1.0.



(a) Metric-only shortest-path (b) Model-optimized shortest-path (c) DART

Fig. 6: Histograms of success rates for METRIC-only shortest-path, model-optimized shortest path, and DART best path directions for every pair of nodes in the map depicted in Fig. 2 when employing our perturbed model.

instruction successfully. The success of a plan, then, is based on the success of executing each individual direction in the plan. We may make this claim because we expect each additional instruction to, at best, leave the probability of the wayfinder reaching unchanged and at worst decrease it. Therefore, actions may be considered in sequence. Looking back to (1), this means our optimization problem is now:

$$\arg\max_{d} \prod_{i=0}^{|d|} p(d_i|m, d_{i-1}) \tag{3}$$

Now the problem resembles a traditional shortest-path graph algorithm, in this case with edge weights equal to the probability of following the next direction. To find the optimal path to all destinations from a given start point, we apply a version of Dijkstra's single-source shortest path algorithm [21] to find the best set of directions to any given point. A short example of the resulting search process can be seen in Fig. 4.

We track potential plans in a priority queue, keeping plans with more particles that have correctly followed the directions so far at the top of the queue. During the expansion calls, we generate the copies of the plan, including the state of all of the current particles, for each possible next direction and simulate taking that step before adding the new plans to the priority queue. A straightforward implementation of this algorithm performs in $\mathcal{O}(|P||D||E|+|V|\log|V|)$, where $|P|$ is the number of particles to simulate, $|D|$ is the number of possible direction types, and $|E|$ and $|V|$ are the number of edges and vertices, respectively.

IV. Results

To evaluate our technique, we constructed environments in simulation as discussed in Section III-A. Simulations were performed with 1000 particles per plan, unless otherwise noted.

A. DART vs. Shortest-Path plans

Our metric for evaluation is the successful arrival rate of our simulated particles. Higher percentages imply better sets of directions according to the model, since more particles were able to successfully execute the plan. The "forgetfulness factor" acts to penalize longer plans, and for our parameter settings results in periodic peaks in our distribution of direction quality corresponding with the number of directions given (and thus more opportunities for forget directions and become lost). As a baseline, we compared directions generated based on our wayfinder model to the standard of the day: shortest-path based METRIC directions. Since it is commonly accepted in the literature that fewer turns are preferable to many, we choose the shortest path with the fewest turns when evaluating.

We first use our model to compute the best directions to follow the same shortest-path described by the metric-only plan. Given knowledge of the model, the quality of the plan should only increase, at worst selecting only metric directions. Second, we compute directions using DART, searching beyond the bounds of the shortest-path to see if plan quality can be improved.

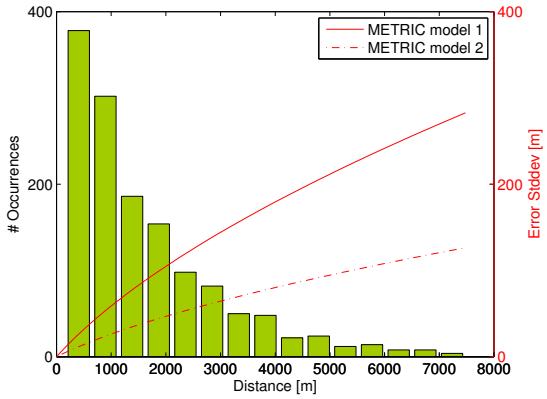


Fig. 7: The distribution of possible METRIC direction distances compared to the stddev in error for the two test models employed.

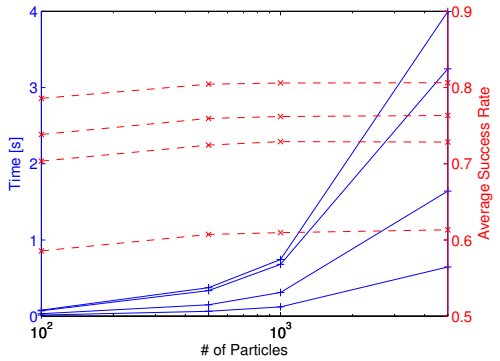


Fig. 8: Plan computation time (blue solid) vs. success rate (red dashed) for four maps of varying complexity and varying quantities of particles. Please note the log scale.

For every start/goal pair in the environment depicted in Fig 2, we computed a plan using the METRIC-only shortest-path method, model-based shortest-path method, and DART. The resulting distributions of success rates can be seen in Fig 5. As expected, METRIC-only directions proved hard to follow for our modeled wayfinder. By employing the model to improve direction selection along the shortest path, we were able to greatly improve arrival rates, but limiting the directions to describing the shortest path still leaves room for improvement. DART returns the directions yielding the highest mean success rate, taking advantage of longer, more easy-to-follow routes deviating from the shortest path. However, DART only offers a small improvement on the optimized shortest-path directions. In part, this is due to the data collection strategy. By computing directions for all possible start/goal pairs, we ensure many very simple sets of directions will be generated where it is unlikely that anything other than the straight-line shortest path will offer significant gains in performance. In these situations, DART and optimized shortest-path directions should look identical. DART shines when computing plans for longer paths where errors become more likely or the shortest path becomes more complex.

The downside to DART is that, be definition, the routes it selects are of equal length or longer than those selected by the shortest-path strategies. To gain a better picture of how much DART deviates from the shortest path, we also collected travel distances for our three evaluation methods. As expected, the improvements in arrival rate come at the price of distance traveled. The worst-case increase between plan distances is roughly 20%.

B. Effects of Model Variation

We wish to verify two properties of DART: first, that our improvement over METRIC-only directions is not solely due to our model excessively penalizing these directions and second, to show that DART is able to adapt to different models. We, performed a second test with a model greatly penalizing the wayfinder's ability to recognize landmarks while improving intersection counting and distance tracking abilities. As a result, we expect the improved performance across METRIC and INTERSECTION directions to be reflected in the types of directions chosen. Further, the performance of METRIC-only shortest-path directions should improve.

We model METRIC direction following capabilities with a Gaussian centered at the requested direction distance. The variance of this Gaussian grows as the mean grows larger, following the form:

$$\sigma^2 = \frac{\mu}{k}\log(\mu + 1) \tag{4}$$

where μ is measured in kilometers and k is a parameter.

Our METRIC model is loosely based on the literature stating that humans have trouble with large numbers [6]. The leading μ term is multiplied by a $\log$ term representing a slow-percentage of the distance requested. The k term scales this percentage. We chose to model the proportional increase as slowly growing instead of linearly proportional based off of the intuition that, when traveling much longer distances, it becomes increasingly difficult to remember exactly how far one has traveled The Gaussian was selected for simplicity.

For the results already presented, we used the parameter $k = 200$, meaning the large majority of wayfinders traveling 1 km will travel a distance within roughly 0.1 km of this amount. To improve the quality of METRIC directions in our second model, we set $k = 1000$, as a result roughly halving the expected error in metric directions. As seen in Fig. 7, around half of the possible directions in our testing space fall at or under the 2 km mark, so the majority of METRIC directions considered will be followed fairly accurately.

With our modified model, the penalized LANDMARK type directions completely disappear in favor of the now more reliable INTERSECTION and METRIC directions. METRIC directions, in particular, benefit from a vast increase in modeled wayfinder following ability. This highlights our method's ability to successfully adapt to varying models as well as improve greatly on the standard practice of only issuing METRIC directions along the shortest path.

Fig. 6 shows the success rate distributions based on our perturbed model. Enhancing the reliability of METRIC

directions helps the metric-only shortest-path plans, but these solutions still lag behind the model-optimized shortest-path and DART best directions. DART still computes the best plans of the three, and with tight distribution of success rates.

C. DFS vs. DART

Though the full DFS described in Sec. III-B can be used to create directions that more-easily recover from error, we were interested in seeing how much this actually matters. Attempts to collect data comparing DART to DFS at the same scale as our DART vs. shortest-path data sets were impractically computationally expensive for 1000 particles. Instead, we tested the two on the small environment seen in Fig. 1. The environment was designed to offer many routes between points with similar structure to see how much DFS could benefit from its more complete search.

Even at this small scale, DFS of $\sim$1600 examples was time consuming, taking 17.7 hrs to compute, or an average of 40 s/plan compared to 0.5 s/plan for DART. As expected, DFS computes some routes that dramatically improve on the results of DART. Typically, however, DART was able to find equivalent or near-equivalent routes to DFS. This suggests that, generally, one set of directions is so clearly the best that there is little to gain baking error recovery into the plan.

D. The Effect of Particle Population on Performance

It is desireable to determine how many particles to use in simulation. We would like to balance the speed of algorithmic performance against the accuracy of our representations of the distributions of directions. To determine this value, we consider two metrics: the time is takes to return a query and the quality of the solution. We expect quality to increase with particle quantity while performance speed will suffer.

We performed trials for 100, 500, 1000, and 5000 particles on 4 maps of gradually increasing complexity. The maps in question had 29, 55, 98, and 167 decision points, respectively, and branching factors ranging from 8.4 to 11.9 before considering direction type.

For each map/particle number combination, we ran 50 planning trials on each of 50 randomly chosen start/goal pairs and computed the mean planning time as well as the mean success rate across trials. The results can be seen in Fig. 8. Returns in plan quality for additional particles diminish rapidly, showing almost no change when increasing the number of particles from 1000 to 5000 particles. Based on this data, 1000 particles seem to be the happy medium between guaranteed performance and quick (sub-second) computation at our testing scales.

V. Conclusions

In this work, we formulated a strategy for constructing reliable textual directions given a generative model of a wayfinder. We developed two particle simulation techniques for determining the best set of directions between two points, given this model. Using such techniques, we are able to find directions through the environment that utilize knowledge of their surroundings such as landmarks in addition to harnessing knowledge about the ways in which the wayfinder will err to pick directions facilitating error recovery, when appropriate. Our preliminary results show that, even given very simple models and minimal knowledge of the environment, particle methods offer an improvement over currently-employed shortest-path methods and can adapt well to varying models.

References

[1] K. Lynch, *The Image of the City*, ser. Publications of the Joint Center for Urban Studies. Technology Press, 1960.

[2] B. Kuipers, "Modeling spatial knowledge," *Cognitive Science*, vol. 2, no. 2, pp. 129–153, 1978.

[3] E. J. Vanetti and G. L. Allen, "Communication Environmental Knowledge: The Impact of Verbal and Spatial Abilities on the Production and Comprehension of Route Directions," *Environment and Behavior*, vol. 20, no. 6, pp. 667–682, November 1988.

[4] K. Lovelace, M. Hegarty, and D. Montello, "Elements of Good Route Directions in Familiar and Unfamiliar Environments," in *Spatial Information Theory. Cognitive and Computational Foundations of Geographic Information Science*. Springer Berlin / Heidelberg, 1999, vol. 1661, pp. 751–751.

[5] M. T. Macmahon, "Following natural language route instructions," Ph.D. dissertation, Austin, TX, USA, 2007.

[6] L. A. Streeter, D. Vitello, and S. A. Wonsiewicz, "How to tell people where to go: comparing navigational aids," *International Journal of Man-Machine Studies*, vol. 22, no. 5, pp. 549 – 562, 1985.

[7] A. M. Hund and J. L. Minarik, "Getting From Here to There: Spatial Anxiety, Wayfinding Strategies, Direction Type, and Wayfinding Efficiency," *Spatial Cognition & Computation*, vol. 6, no. 3, pp. 179–201, 2006.

[8] R. Cohen, L. M. Baldwin, and R. C. Sherman, "Cognitive Maps of a Naturalistic Setting," *Child Development*, vol. 49, no. 4, pp. pp. 1216–1218, 1978.

[9] P. W. Thorndyke, "Distance estimation from cognitive maps," *Cognitive Psychology*, vol. 13, no. 4, pp. 526 – 550, 1981.

[10] M. Denis, F. Pazzaglia, C. Cornoldi, and L. Bertolo, "Spatial discourse and navigation: an analysis of route directions in the city of Venice," *Applied Cognitive Psychology*, vol. 13, no. 2, pp. 145–174, 1999.

[11] P.-E. Michon and M. Denis, "When and Why Are Visual Landmarks Used in Giving Directions?" in *Spatial Information Theory*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2001, vol. 2205, pp. 292–305.

[12] C. C. Presson and D. R. Montello, "Points of reference in spatial cognition: Stalking the elusive landmark*," *British Journal of Developmental Psychology*, vol. 6, no. 4, pp. 378–381, 1988.

[13] G. Burnett, D. Smith, and A. May, "Supporting the navigation task: characteristics of 'good' landmarks," in *Proceedings of the Annual Conference of the Ergonomics Society*, November 2001, pp. 441–446.

[14] R. J. Elliott and M. Lesk, "Route Finding in Street Maps by Computers and People," in *AAAI*, 1982, pp. 258–261.

[15] M. Duckham and L. Kulik, ""Simplest" Paths: Automated Route Selection for Navigation," in *COSIT*, 2003, pp. 169–185.

[16] S. Haque, L. Kulik, and A. Klippel, "Algorithms for Reliable Navigation and Wayfinding," in *Spatial Cognition V Reasoning, Action, Interaction*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2007, pp. 308–326.

[17] K.-F. Richter and A. Klippel, "A Model for Context-Specific Route Directions," in *Spatial Cognition IV. Reasoning, Action, Interaction*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2005, vol. 3343, pp. 58–78.

[18] K.-F. Richter, "A uniform handling of different landmark types in route directions," in *Spatial Information Theory*. Springer Berlin / Heidelberg, 2007, vol. 4736, pp. 373–389.

[19] K.-F. Richter and M. Duckham, "Simplest Instructions: Finding Easy-to-Describe Routes for Navigation," in *GIScience*, 2008, pp. 274–289.

[20] S. Thrun, D. Fox, W. Burgard, and F. Dellaert, "Robust monte carlo localization for mobile robots," *Artificial Intelligence*, vol. 128, no. 1-2, pp. 99–141, 2001.

[21] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, pp. 269–271, 1959.