# Formal-language-theoretic Optimal Path Planning For Accommodation of Amortized Uncertainties and Dynamic Effects[☆]

I. Chattopadhyay[a,*], A. Cascone[a], A. Ray[a]

[a]*Department of Mechanical Engineering, The Pennsylvnaia State University, University Park, 16802*



**Abstract**

We report a globally-optimal approach to robotic path planning under uncertainty, based on the theory of quantitative measures of formal languages. A significant generalization to the language-measure-theoretic path planning algorithm $\nu^\star$ is presented that explicitly accounts for average dynamic uncertainties and estimation errors in plan execution. The notion of the navigation automaton is generalized to include probabilistic uncontrollable transitions, which account for uncertainties by modeling and planning for probabilistic deviations from the computed policy in the course of execution. The planning problem is solved by casting it in the form of a performance maximization problem for probabilistic finite state automata. In essence we solve the following optimization problem: Compute the navigation policy which maximizes the probability of reaching the goal, while simultaneously minimizing the probability of hitting an obstacle. Key novelties of the proposed approach include the modeling of uncertainties using the concept of uncontrollable transitions, and the solution of the ensuing optimization problem using a highly efficient search-free combinatorial approach to maximize quantitative measures of probabilistic regular languages. Applicability of the algorithm in various models of robot navigation has been shown with experimental validation on a two-wheeled mobile robotic platform (SEGWAY RMP 200) in a laboratory environment.

*Keywords:*
Language Measure, Probabilistic Finite State Machines, Robotics, Path Planning, Supervisory Control

## 1. Introduction & Motivation

The objective of this paper is to report a globally-optimal approach to path planning under uncertainty, based on the theory of quantitative measures of formal languages. The field of trajectory and motion planning is enormous, with applications in such diverse areas as industrial robots, mobile robot navigation, spacecraft reentry, video games and even drug design. Many of the basic concepts are presented in [1] and in recent comprehensive surveys [2]. In the context of planning for mobile robots and manipulators much of the literature on path and motion planning is concerned with finding collision-free trajectories [3]. A great deal of the complexity in these problems arises from the topology of the robot's configuration space, called the $C$-Space. Various analytical techniques, such as wavefront expansion [4] and cellular decomposition, have been reported in recent literature [5], which partition the $C$-Space into a finite number of regions with the objective of reducing the motion planning problem as identification of a sequence of neighboring cells between the initial and final (i.e., goal) regions. Graph-theoretic search-based techniques have been used somewhat successfully in many wheeled ground robot path planning problems and have been used for some UAV planning problems, typically radar evasion [6]. These approaches typically suffer from complexity issues arising from expensive searches, particularly in complicated configuration spaces. To circumvent the complexity associated with graph-based planning, sampling based planning methods [7] such as probabilistic roadmaps have been proposed. However, sampling based approaches are only probabilistically complete (i.e. if a feasible solution exists it will be found, given enough time) but there is no guarantee of finding a solution within a specified time, and more often than not, global route optimality is not guaranteed. Distinct from these general approaches, there exist reported techniques that explicitly make use of physical aspects of specific problems for planning *e.g.* use of vertical wind component for generating optimal trajectories for UAVs [8], feasible collision-free trajectory generation for cable driven platforms [9], and the recently reported approach employing angular processing [10].

### *1.1. Potential Field-based Planning Methodology*

Among reported deterministic approaches, methods based on artificial potential fields have been extensively investigated, often referred to cumulatively as potential field methods (PFM). The idea of imaginary forces acting on a robot were suggested by several authors inclding [11] and [12]. In these approaches obstacles exert repulsive forces onto the robot, while the target applies an attractive force to the robot. The resultant of all forces determines the subsequent direction and speed of travel. One of the reasons for the popularity of this method is its simplicity and elegance. Simple PFMs can be implemented quickly

[☆]This work has been supported in part by the U.S. Army Research Laboratory and the U.S. Army Research Office under Grant No. W911NF-07-1-0376 and by the Office of Naval Research under Grant No. N00014-08-1-380

[*]Corresponding Author

and initially provide acceptable results without requiring many refinements. [13] has suggested a generalized potential field method that combines global and local path planning. Potential field based techniques have been also successfully employed in multi-robot co-operative planning scenarios [14, 15], where other techniques prove to be inefficient and impractical.

While the potential field principle is particularly attractive because of its elegance and simplicity, substantial shortcomings have been identified that are inherent to this principle. The interested reader is referred to [16] for a systematic criticism of PFM-based planning, where the authors cite the underlying differential equation based analysis as the source of the problems, and the fact that it combines the robot and the environment into one unified system. Key problems inherent to PFMs, independent of the particular implementation, are:

1. **Trap situations due to local minima:** Perhaps the best-known problem with PFMs are possible trap-situations [11, 17], which occur when when the robot runs into a dead end, due to the existence of a local extrema in the potential field. Trap-situations can be remedied with heuristic recovery rules, which are likely to result in non-optimal paths.
2. **No passage between closely spaced obstacles:** A severe problem with PFMs occurs when the robot attempts to travel through narrow corridors thereby experiencing repulsive forces simultaneously from opposite sides, leading to wavy trajectories, no passage etc.
3. **Oscillations in the presence of obstacles:** Presence of high obstacle clutter often leads to unstable motion, due to the complexity of the resultant potential.
4. **Effect of past obstacles:** Even after the robot has already passed an obstacle, the latter keeps affecting the robot motion for a significant period of time (until the repulsive potential dies down).

These disadvantages become more apparent when the PFM-based methods are implemented in high-speed real-time systems; simulations and slow speed experiments often conceal the issues; probably contributing to the widespread popularity of potential planners.

### *1.2. The $\nu^\star$ Planning Algorithm*

Recently, the authors reported a novel path planning algorithm $\nu^\star$ [18], that models the navigation problem in the framework of Probabilistic Finite State Automata (PFSA) and computes optimal plans via optimization of the PFSA from a strictly control-theoretic viewpoint. $\nu^\star$ uses cellular decomposition of the workspace, and assumes that the blocked grid locations can be easily estimated, upon which the planner computes an optimal navigation gradient that is used to obtain the routes. This navigation gradient is computed by optimizing the quantitative measure of the probabilistic formal language generated by the associated navigation automaton. The key advantages can be enumerated as:

1. **$\nu^\star$ is fundamentally distinct from a search:** The search problem is replaced by a sequential solution of sparse linear systems. On completion of cellular decomposition, $\nu^\star$ optimizes the resultant PFSA via a iterative sequence of combinatorial operations which elementwise maximizes the language measure vector [19][20]. Note that although $\nu^\star$ involves probabilistic reasoning, the final waypoint sequence obtained is deterministic.
2. **Computational efficiency:** The intensive step in $\nu^\star$ is a special sparse matrix inversion to compute the language measure. The time complexity of each iteration step can be shown to be linear in problem size implying significant numerical advantage over search-based methods for high-dimensional problems.
3. **Global monotonicity:** The solution iterations are globally monotonic, *i.e*, each iteration yields a better approximation to the final optimal solution. The final waypoint sequence is generated essentially by following the measure gradient which has a unique maxima at the goal.
4. **Global Optimality:** It can be shown that trap-situations are a mathematical impossibility for $\nu^\star$.

The optimal navigation gradient produced by $\nu^\star$ is reminiscent of potential field methods [7]. However, $\nu^\star$ automatically generates, *and optimizes* this gradient; no ad-hoc potential function is necessary.

### *1.3. Focus of Current Work & Key Contributions*

The key focus of this paper is extension of the $\nu^\star$ planning algorithm to optimally handle execution uncertainties. It is well recognized by domain experts that merely coming up with a navigation plan is not sufficient; the computed plan must be executed in the real world by the mobile robot, which often cannot be done exactly and precisely due to measurement noise in the exteroceptive sensors, imperfect actuations, and external disturbances. The idea of planning under uncertainties is not particularly new, and good surveys of reported methodologies exist [21]. In chronological order, the main family of reported approaches can be enumerated as follows:

- Pre-image Back-chaining [22, 23, 24] where the plan is synthesized by computing a set of configurations from which the robot can possibly reach the goal, and then propgating this *preimage* recursively backward or *back-chaining*, a problem solving approach originally proposed in [25].
- Approach based on sensory uncertainty fields (SUF) [26, 27, 28, 29] computed over the collision-free subset of the robot's configuration space, which reflects expected uncertainty (distribution of possible errors) in the sensed configuration that would be computed by matching the sensory data against a known environment model (*e.g*. landmark locations). A planner then makes use of the computed SUF to generate paths that minimize expected errors.
- Sensor-based planning approaches [30, 31, 32], which consider explicit uncertainty models of various motion primitives to compute a feasible robust plan composed of sensor-based motion commands in polygonal environments, with significant emphasis on wall-following schemes.

- Information space based approach using the Bellman principle of stochastic dynamic programming [33], which introduced key concepts such as setting up the problem in a probabilistic framework, and demanding that the optimal plan maximize the probability of reaching the goal. However, the main drawback was the exponential dependence on the dimension of the computed information space.
- The set-membership approach [34] which performs a local search, trying to deform a path into one that respects uncertainty constraints imposed by arbitrarily shaped uncertainty sets. Each hard constraint is turned into a soft penalty function, and the gradient descent algorithm is employed, hoping convergence to an admissible solution.
- Probabilistic approaches based on disjunctive linear programming [35, 36], with emphasis on UAV applications. The key limitation is the inability to take into account exteroceptive sensors, and also the assumption that dead-reckoning is independent of the path executed. Later extensions of this approach use particle representations of the distributions, implying wider applicability.
- Adaptation of search strategies in extended spaces [37, 38, 39, 40], which consider the classical search problem in configuration spaces augmented with uncertainty information.
- Approach based on Stochastic Motion Roadmaps (SRM) [41], which combines sampling-based roadmap representation of the configuration space, with the theory of Markov Decision Processes, to yield models that can be subsequently optimized via value-iteration based infinite horizon dynamic programming, leading to plans that maximize the probability of reaching the goal.

The current work adds a new member to the family of existing approaches to address globally optimal path planning under uncertainties. The key novelty of this paper is the association of *uncertainty with the notion of uncontrollability in a controlled system*. The navigation automaton introduced in [18] is augmented with uncontrollable transitions which essentially captures the possibility that the agent may execute actuation sequences (or motion primitives) that are not coincident with the planned moves. The planning objective is simple: *Maximize the probability of reaching the goal, while simultaneously minimizing the probability of hitting any obstacle.* Note that, in this respect, we are essentially solving the exact same problem investigated by [41]. However our solution approach is very different. Instead of using value iteration based dynamic programming, we use the theory of language-measure-theoretic optimization of probabilistic finite state automata [20]. Unlike the SRM approach, the proposed algorithm does not require the use of local auxiliary planners, and also needs to make no assumptions on the structure of the configuration space to guarantee iterative convergence. The use of arbitrary penalties for reducing the weight on longer paths is also unnecessary, which makes the proposed $\nu^\star$ under uncertainties completely free from heuristics. We show that all the key advantages that $\nu^\star$ has over the state-of-art carries over to this more general case; namely that of significantly better computational efficiency, simplicity of implementation, and achieving global optimality via monotonic sequence of *search-free combinatorial iterative improvements*, with guaranteed polynomial convergence. The proposed approach thus solves the inherently non-convex optimization [42] by mapping the physical specification to an optimal control problem for probabilistic finite state machines (the navigation automata), which admits efficient combinatorial solutions via the language-measure-theoretic approach. The source of many uncertainties, namely modeling uncertainty, disturbances, and uncertain localization, is averaged over (or amortized) for adequate representation in the automaton framework. This may be viewed as a source of approximation in the proposed approach; however we show in simulation and in actual experimentation that the amortization is indeed a good approach to reduce planning complexity and results in highly robust planning decisions. Thus the modified language-measure-theoretic approach presented in this paper, potentially lays the framework for seamless integration of data-driven and physics-based models with the high-level decision processes; this is a crucial advantage, and goes to address a key issue in autonomous robotics, *e.g.*, in a path-planning scenario with mobile robots, the optimal path may be very different for different speeds, platform capabilities and mission specifications. Previously reported approaches to handle these effects using exact differential models of platform dynamics results in overtly complex solutions that do not respond well to modeling uncertainties, and more importantly to possibly non-stationary environmental dynamics and evolving mission contexts. Thus the measure-theoretic approach enables the development of true **Cyber-Physical algorithms** for control of autonomous systems; algorithms that operate in the logical domain while optimally integrating, and responding to, physical information in the planning process.

The rest of the paper is organized in seven sections. Section 2 briefly explains the language-theoretic models considered in this paper, reviews the language-measure-theoretic optimal control of probabilistic finite state machines and presents the necessary details of the reported $\nu^\star$ algorithm. Section 3 presents the modifications to the navigation model to incorporate the effects of dynamic uncertainties within the framework of probabilistic automata. Section 4 presents the pertinent theoretical results and establishes the main planning algorithm. Section 5 develops a formulation to identify the key amortized uncertainty parameters of the PFSA-based navigation model from an observed dynamical response of a given platform. The proposed algorithm is summarized with pertinent comments in Section 6. The theoretical development is verified in high-fidelity simulations on different navigation models and validated in experimental runs on the SEGWAY RMP 200 in section 7. The paper is summarized and concluded in Section 8 with recommendations for future work.

## 2. Preliminaries: Language Measure-theoretic Optimization Of Probabilistic Automata

This section summarizes the signed real measure of regular languages; the details are reported in [43]. Let $G_i \equiv \langle Q, \Sigma, \delta, q_i, Q_m \rangle$ be a trim (i.e., accessible and co-accessible) finite-state automaton model that represents the discrete-event dynamics of a physical plant, where $Q = \{q_k : k \in \mathcal{I}_Q\}$ is the set of states and $\mathcal{I}_Q \equiv \{1, 2, \cdots, n\}$ is the index set of states; the automaton starts with the initial state $q_i$; the alphabet of events is $\Sigma = \{\sigma_k : k \in \mathcal{I}_\Sigma\}$, having $\Sigma \bigcap \mathcal{I}_Q = \emptyset$ and $\mathcal{I}_\Sigma \equiv \{1, 2, \cdots, \ell\}$ is the index set of events; $\delta : Q \times \Sigma \to Q$ is the (possibly partial) function of state transitions; and $Q_m \equiv \{q_{m_1}, q_{m_2}, \cdots, q_{m_l}\} \subseteq Q$ is the set of marked (i.e., accepted) states with $q_{m_k} = q_j$ for some $j \in \mathcal{I}_Q$. Let $\Sigma^*$ be the Kleene closure of $\Sigma$, i.e., the set of all finite-length strings made of the events belonging to $\Sigma$ as well as the empty string $\epsilon$ that is viewed as the identity of the monoid $\Sigma^*$ under the operation of string concatenation, i.e., $\epsilon s = s = s\epsilon$. The state transition map $\delta$ is recursively extended to its reflexive and transitive closure $\delta : Q \times \Sigma^* \to Q$ by defining $\forall q_j \in Q,\ \delta(q_j, \epsilon) = q_j$ and $\forall q_j \in Q, \sigma \in \Sigma, s \in \Sigma^\star,\ \delta(q_i, \sigma s) = \delta(\delta(q_i, \sigma), s)$

**Definition 1.** *The language $L(q_i)$ generated by a DFSA $G$ initialized at the state $q_i \in Q$ is defined as: $L(q_i) = \{s \in \Sigma^* \mid \delta^*(q_i, s) \in Q\}$ The language $L_m(q_i)$ marked by the DFSA $G$ initialized at the state $q_i \in Q$ is defined as: $L_m(q_i) = \{s \in \Sigma^* \mid \delta^*(q_i, s) \in Q_m\}$*

**Definition 2.** *For every $q_j \in Q$, let $L(q_i, q_j)$ denote the set of all strings that, starting from the state $q_i$, terminate at the state $q_j$, i.e., $L_{i,j} = \{s \in \Sigma^* \mid \delta^*(q_i, s) = q_j \in Q\}$*

The formal language measure is first defined for terminating plants [44, 45] with sub-stochastic event generation probabilities *i.e*. the event generation probabilities at each state summing to strictly less than unity.

**Definition 3.** *The event generation probabilities are specified by the function $\tilde{\pi} : \Sigma^\star \times Q \to [0, 1]$ such that $\forall q_j \in Q, \forall \sigma_k \in \Sigma, \forall s \in \Sigma^\star$,*

*(1) $\tilde{\pi}(\sigma_k, q_j) \triangleq \tilde{\pi}_{jk} \in [0, 1)$; $\sum_k \tilde{\pi}_{jk} = 1 - \theta$, with $\theta \in (0, 1)$;*
*(2) $\tilde{\pi}(\sigma, q_j) = 0$ if $\delta(q_j, \sigma)$ is undefined; $\tilde{\pi}(\epsilon, q_j) = 1$;*
*(3) $\tilde{\pi}(\sigma_k s, q_j) = \tilde{\pi}(\sigma_k, q_j)\, \tilde{\pi}(s, \delta(q_j, \sigma_k))$.*

*The $n \times \ell$ event cost matrix is defined as: $\widetilde{\mathbf{\Pi}}|_{ij} = \tilde{\pi}(q_i, \sigma_j)$*

**Definition 4.** *The state transition probability $\pi : Q \times Q \to [0, 1)$, of the DFSA $G_i$ is defined as follows: $\forall q_i, q_j \in Q, \pi_{ij} = \sum_{\sigma \in \Sigma \text{ s.t. } \delta(q_i, \sigma) = q_j} \tilde{\pi}(\sigma, q_i)$ The $n \times n$ state transition probability matrix is defined as $\mathbf{\Pi}|_{jk} = \pi(q_i, q_j)$*

The set $Q_m$ of marked states is partitioned into $Q_m^+$ and $Q_m^-$, i.e., $Q_m = Q_m^+ \cup Q_m^-$ and $Q_m^+ \cap Q_m^- = \emptyset$, where $Q_m^+$ contains all *good* marked states that we desire to reach, and $Q_m^-$ contains all *bad* marked states that we want to avoid, although it may not always be possible to completely avoid the *bad* states while attempting to reach the *good* states. To characterize this, each marked state is assigned a real value based on the designer's perception of its impact on the system performance.

**Definition 5.** *The characteristic function $\chi : Q \to [-1, 1]$ that assigns a signed real weight to state-based sublanguages $L(q_i, q)$ is defined as:*

$$\forall q \in Q, \quad \chi(q) \in \begin{cases} [-1, 0), & q \in Q_m^- \\ \{0\}, & q \notin Q_m \\ (0, 1], & q \in Q_m^+ \end{cases} \tag{1}$$

*The state weighting vector, denoted by $\boldsymbol{\chi} = [\chi_1\ \chi_2\ \cdots\ \chi_n]^T$, where $\chi_j \equiv \chi(q_j)\ \forall j \in \mathcal{I}_Q$, is called the $\boldsymbol{\chi}$-vector. The $j$-th element $\chi_j$ of $\boldsymbol{\chi}$-vector is the weight assigned to the corresponding terminal state $q_j$.*

In general, the marked language $L_m(q_i)$ consists of both good and bad event strings that, starting from the initial state $q_i$, lead to $Q_m^+$ and $Q_m^-$ respectively. Any event string belonging to the language $L^0 = L(q_i) - L_m(q_i)$ leads to one of the non-marked states belonging to $Q - Q_m$ and $L^0$ does not contain any one of the good or bad strings. Based on the equivalence classes defined in the Myhill-Nerode Theorem, the regular languages $L(q_i)$ and $L_m(q_i)$ can be expressed as: $L(q_i) = \bigcup_{q_k \in Q} L_{i,k}$ and $L_m(q_i) = \bigcup_{q_k \in Q_m} L_{i,k} = L_m^+ \cup L_m^-$ where the sublanguage $L_{i,k} \subseteq G_i$ having the initial state $q_i$ is uniquely labelled by the terminal state $q_k, k \in \mathcal{I}_Q$ and $L_{i,j} \cap L_{i,k} = \emptyset\ \forall j \neq k$; and $L_m^+ \equiv \bigcup_{q_k \in Q_m^+} L_{i,k}$ and $L_m^- \equiv \bigcup_{q_k \in Q_m^-} L_{i,k}$ are good and bad sublanguages of $L_m(q_i)$, respectively. Then, $L^0 = \bigcup_{q_k \notin Q_m} L_{i,k}$ and $L(q_i) = L^0 \cup L_m^+ \cup L_m^-$.

A signed real measure $\mu^i : 2^{L(q_i)} \to \mathbb{R} \equiv (-\infty, +\infty)$ is constructed on the $\sigma$-algebra $2^{L(q_i)}$ for any $i \in \mathcal{I}_Q$; interested readers are referred to [43] for the details of measure-theoretic definitions and results. With the choice of this $\sigma$-algebra, every singleton set made of an event string $s \in L(q_i)$ is a measurable set. By Hahn Decomposition Theorem [46], each of these measurable sets qualifies itself to have a numerical value based on the above state-based decomposition of $L(q_i)$ into $L^0$(null), $L^+$(positive), and $L^-$(negative) sublanguages.

**Definition 6.** *Let $\omega \in L(q_i, q_j) \subseteq 2^{L(q_i)}$. The signed real measure $\mu^i$ of every singleton string set $\{\omega\}$ is defined as: $\mu^i(\{\omega\}) := \tilde{\pi}(q_i, \omega)\chi(q_j)$. The signed real measure of a sublanguage $L_{i,j} \subseteq L(q_i)$ is defined as: $\mu_{i,j} := \mu^i(L(q_i, q_j)) = \left(\sum_{\omega \in L(q_i, q_j)} \tilde{\pi}(q_i, \omega)\right)\chi_j$*

Therefore, the signed real measure of the language of a DFSA $G_i$ initialized at $q_i \in Q$, is defined as $\mu_i := \mu^i(L(q_i)) = \sum_{j \in \mathcal{I}_Q} \mu^i(L_{i,j})$. It is shown in [43] that the language measure can be expressed as $\mu_i = \sum_{j \in \mathcal{I}_Q} \pi_{ij}\mu_j + \chi_i$. The language measure vector, denoted as $\boldsymbol{\mu} = [\mu_1\ \mu_2\ \cdots\ \mu_n]^T$, is called the $\boldsymbol{\mu}$-vector. In vector form, we have $\boldsymbol{\mu} = \mathbf{\Pi}\mu + \boldsymbol{\chi}$ whose solution is given by $\boldsymbol{\mu} = (\mathbf{I} - \mathbf{\Pi})^{-1}\boldsymbol{\chi}$ The inverse exists for terminating plant models [44] because $\mathbf{\Pi}$ is a contraction operator [43] due to the strict inequality $\sum_j \Pi_{ij} < 1$. The residual $\theta_i = 1 - \sum_j \Pi_{ij}$ is referred to as the termination probability for state $q_i \in Q$. We extend the analysis to non-terminating plants with stochastic transition probability matrices (*i.e*. with $\theta_i = 0,\ \forall q_i \in Q$) by renormalizing the language measure [19] with respect to the uniform

termination probability of a limiting terminating model as described next.

Let $\widetilde{\Pi}$ and $\Pi$ be the stochastic event generation and transition probability matrices for a non-terminating plant $G_i = \langle Q, \Sigma, \delta, q_i, Q_m \rangle$. We consider the terminating plant $G_i(\theta)$ with the same DFSA structure $\langle Q, \Sigma, \delta, q_i, Q_m \rangle$ such that the event generation probability matrix is given by $(1-\theta)\widetilde{\Pi}$ with $\theta \in (0,1)$ implying that the state transition probability matrix is $(1-\theta)\Pi$.

**Definition 7** (Renormalized Measure)**.** *The renormalized measure $\nu^i_\theta : 2^{L(q_i(\theta))} \to [-1,1]$ for the $\theta$-parametrized terminating plant $G_i(\theta)$ is defined as:*

$$\forall \omega \in L(q_i(\theta)),\ \nu^i_\theta(\{\omega\}) = \theta \mu^i(\{\omega\}) \tag{2}$$

*The corresponding matrix form is given by $\boldsymbol{\nu_\theta} = \theta \boldsymbol{\mu} = \theta [I-(1-\theta)\Pi]^{-1}\boldsymbol{\chi}$ with $\theta \in (0,1)$. We note that the vector representation allows for the following notational simplification $\nu^i_\theta(L(q_i(\theta))) = \boldsymbol{\nu_\theta}\big|_i$ The renormalized measure for the non-terminating plant $G_i$ is defined to be $\lim_{\theta \to o^+} \nu^i_\theta$.*

### 2.1. Event-driven Supervision of PFSA

Plant models considered in this paper are *deterministic* finite state automata (plant) with well-defined event occurrence *probabilities*. In other words, the occurrence of events is probabilistic, but the state at which the plant ends up, *given a particular event has occurred*, is deterministic. Since no emphasis is placed on the initial state and marked states are completely determined by $\chi$, the models can be completely specified by a sextuple as: $G = (Q, \Sigma, \delta, \widetilde{\boldsymbol{\Pi}}, \chi, \mathscr{C})$

**Definition 8** (Control Philosophy)**.** *If $q_i \underset{\sigma}{\to} q_k$, and the event $\sigma$ is disabled at state $q_i$, then the supervisory action is to prevent the plant from making a transition to the state $q_k$, by forcing it to stay at the original state $q_i$. Thus disabling any transition $\sigma$ at a given state $q$ results in deletion of the original transition and appearance of the self-loop $\delta(q,\sigma) = q$ with the occurrence probability of $\sigma$ from the state $q$ remaining unchanged in the supervised and unsupervised plants. For a given plant, transitions that can be disabled in the sense of Definition 8 are defined to be controllable transitions. The set of controllable transitions in a plant is denoted $\mathscr{C}$. Note controllability is state-based.*

### 2.2. Optimal Supervision Problem: Formulation & Solution

A supervisor disables a subset of the set $\mathscr{C}$ of controllable transitions and hence there is a bijection between the set of all possible supervision policies and the power set $2^{\mathscr{C}}$. That is, there exists $2^{|\mathscr{C}|}$ possible supervisors and each supervisor is uniquely identifiable with a subset of $\mathscr{C}$ and the language measure $\nu$ allows a quantitative comparison of different policies.

**Definition 9.** *For an unsupervised plant $G = (Q, \Sigma, \delta, \widetilde{\Pi}, \chi, \mathscr{C})$, let $G^\dagger$ and $G^\ddagger$ be the supervised plants with sets of disabled transitions, $\mathscr{D}^\dagger \subseteq \mathscr{C}$ and $\mathscr{D}^\ddagger \subseteq \mathscr{C}$, respectively, whose measures are $\boldsymbol{\nu}^\dagger$ and $\boldsymbol{\nu}^\ddagger$. Then, the supervisor that disables $\mathscr{D}^\dagger$ is defined to be superior to the supervisor that disables $\mathscr{D}^\ddagger$ if $\boldsymbol{\nu}^\dagger \geqq_{\texttt{(Elementwise)}} \boldsymbol{\nu}^\ddagger$ and strictly superior if $\boldsymbol{\nu}^\dagger >_{\texttt{(Elementwise)}} \boldsymbol{\nu}^\ddagger$.*

**Definition 10** (Optimal Supervision Problem)**.** *Given a (non-terminating) plant $G = (Q, \Sigma, \delta, \widetilde{\Pi}, \chi, \mathscr{C})$, the problem is to compute a supervisor that disables a subset $\mathscr{D}^\star \subseteq \mathscr{C}$, such that $\boldsymbol{\nu}^\star \geqq_{\texttt{(Elementwise)}} \boldsymbol{\nu}^\dagger \ \ \forall \mathscr{D}^\dagger \subseteq \mathscr{C}$ where $\boldsymbol{\nu}^\star$ and $\boldsymbol{\nu}^\dagger$ are the measure vectors of the supervised plants $G^\star$ and $G^\dagger$ under $\mathscr{D}^\star$ and $\mathscr{D}^\dagger$, respectively.*

**Remark 1.** *The solution to the optimal supervision problem is obtained in [20, 47] by designing an optimal policy for a terminating plant [45] with a sub-stochastic transition probability matrix $(1-\theta)\widetilde{\Pi}$ with $\theta \in (0,1)$. To ensure that the computed optimal policy coincides with the one for $\theta = 0$, the suggested algorithm chooses a small value for $\theta$ in each iteration step of the design algorithm. However, choosing $\theta$ too small may cause numerical problems in convergence. Algorithms reported in [20, 47] computes how small a $\theta$ is actually required, i.e., computes the critical lower bound $\theta_\star$, thus solving the optimal supervision problem for a generic PFSA. It is further shown that the solution obtained is optimal and unique and can be computed by an effective algorithm.*

**Definition 11.** *Following Remark 1, we note that algorithms reported in [20, 47] compute a lower bound for the critical termination probability for each iteration of such that the disabling/enabling decisions for the terminating plant coincide with the given non-terminating model. We define $\theta_{min} = \min_k \theta^{[k]}_\star$ where $\theta^{[k]}_\star$ is the termination probability computed in the $k^{th}$ iteration.*

**Definition 12.** *If $G$ and $G^\star$ are the unsupervised and supervised PFSA respectively then we denote the renormalized measure of the terminating plant $G^\star(\theta_{min})$ as $\nu^i_\# : 2^{L(q_i)} \to [-1,1]$ (See Definition 7). Hence, in vector notation we have: $\boldsymbol{\nu}_\# = \theta_{min}[I - (1-\theta_{min})\Pi^\#]^{-1}\boldsymbol{\chi}$ where $\Pi^\#$ is the transition probability matrix of the supervised plant $G^\star$, we note that $\boldsymbol{\nu}_\# = \nu^{[K]}$ where $K$ is the total number of iterations required for convergence.*

For the sake of completeness, the algorithmic approach is shown in Algorithms 1 and 2.

### 2.3. Problem Formulation: A PFSA Model of Autonomous Navigation

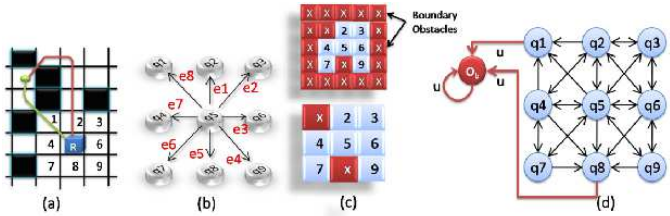



Figure 1: **(a)** shows the vehicle (marked "R") with the obstacle positions shown as black squares. The Green4 dot identifies the goal **(b)** shows the finite state representation of the possible one-step moves from the current position. **(d)** shows uncontrollable transitions "u" from states corresponding to blocked grid locations to "$q_\ominus$"

We consider a 2D workspace for the mobile agents. This restriction on workspace dimensionality serves to simplify the exposition and can be easily relaxed. To set up the problem, the

**Algorithm 1**: Computation of Optimal Supervisor

**input** : **P**, $\chi$, $\mathscr{C}$
**output**: Optimal set of disabled transitions $\mathscr{D}^\star$
**begin**
  Set $\mathscr{D}^{[0]} = \emptyset$ ; /* Initial disabling set */
  Set $\widetilde{\Pi}^{[0]} = \widetilde{\Pi}$ ; /* Initial event prob. matrix */
  Set $\theta_\star^{[0]} = 0.99$, Set $k = 1$ , Set `Terminate` = **false**;
  **while** (`Terminate` == **false**) **do**
    Compute $\theta_\star^{[k]}$; /* Algorithm 2 */
    Set $\widetilde{\Pi}^{[k]} = \frac{1-\theta_\star^{[k]}}{1-\theta_\star^{[k-1]}}\widetilde{\Pi}^{[k-1]}$;
    Compute $\boldsymbol{\nu}^{[k]}$ ;
    **for** $j = 1$ ***to*** $n$ **do**
      **for** $i = 1$ ***to*** $n$ **do**
        Disable all controllable $q_i \xrightarrow{\sigma} q_j$ s.t. $\boldsymbol{\nu}_j^{[k]} < \boldsymbol{\nu}_i^{[k]}$ ;
        Enable all controllable $q_i \xrightarrow{\sigma} q_j$ s.t. $\boldsymbol{\nu}_j^{[k]} \geqq \boldsymbol{\nu}_i^{[k]}$ ;
    Collect all disabled transitions in $\mathscr{D}^{[k]}$;
    **if** $\mathscr{D}^{[k]} == \mathscr{D}^{[k-1]}$ **then**
      `Terminate` = **true**;
    **else**
      $k = k + 1$ ;
  $\mathscr{D}^\star = \mathscr{D}^{[k]}$ ; /* Optimal disabling set */
**end**

workspace is first discretized into a finite grid and hence the approach developed in this paper falls under the generic category of discrete planning. The underlying theory does not require the grid to be regular; however for the sake of clarity we shall present the formulation under the assumption of a regular grid. The obstacles are represented as blocked-off grid locations in the discretized workspace. We specify a particular location as the fixed goal and consider the problem of finding optimal and feasible paths from arbitrary initial grid locations in the workspace. Figure 1(a) illustrates the basic problem setup. We further assume that at any given time instant the robot occupies one particular location (*i.e.* a particular square in Figure 1(a)). As shown in Figure 1, the robot has eight possible moves from any interior location. The boundaries are handled by removing the moves that take the robot out of the workspace. The possible moves are modeled as controllable transitions between grid locations since the robot can "choose" to execute a particular move from the available set. We note that the number of possible moves (8 in this case) depends on the chosen fidelity of discretization of the robot motion and also on the intrinsic vehicle dynamics. The complexity results presented in this paper only assumes that the number of available moves is significantly smaller compared to the number of grid squares, *i.e.*, the discretized position states. Specification of inter-grid transitions in this manner allows us to generate a finite state automaton (FSA) description of the navigation problem. Each square in the discretized workspace is modeled as a FSA state with the controllable transitions defining the corresponding state transition map. The formal description of the model is as follows:

**Algorithm 2**: Computation of the Critical Lower Bound $\theta_\star$

**input** : **P**, $\chi$
**output**: $\theta_\star$
**begin**
  Set $\theta_\star = 1$, Set $\theta_{curr} = 0$;
  Compute $\mathscr{P}$ , $M_0$ , $M_1$, $M_2$;
  **for** $j = 1$ ***to*** $n$ **do**
    **for** $i = 1$ ***to*** $n$ **do**
      **if** $(\mathscr{P}\chi)_i - (\mathscr{P}\chi)_j \neq 0$ **then**
        $\theta_{curr} = \frac{1}{8M_2}\left|(\mathscr{P}\chi)_i - (\mathscr{P}\chi)_j\right|$
      **else**
        **for** $r = 0$ ***to*** $n$ **do**
          **if** $(M_0\chi)_i \neq (M_0\chi)_j$ **then**
            **Break**;
          **else**
            **if** $\left(M_0 M_1^r \chi\right)_i \neq \left(M_0 M_1^r \chi\right)_j$ **then**
              **Break**;
        **if** $r == 0$ **then**
          $\theta_{curr} = \frac{|\{(M_0 - \mathscr{P})\chi\}_i - \{(M_0 - \mathscr{P})\chi\}_j|}{8M_2}$;
        **else**
          **if** $r > 0$ ***AND*** $r \leq n$ **then**
            $\theta_{curr} = \frac{|(M_0 M_1 \chi)_i - (M_0 M_1 \chi)_j|}{2^{r+3} M_2}$ ;
          **else**
            $\theta_{curr} = 1$ ;
      $\theta_\star = \min(\theta_\star, \theta_{curr})$ ;
**end**

Let $\mathbb{G}_{\mathsf{NAV}} = (Q, \Sigma, \delta, \tilde{\Pi}, \chi)$ be a Probabilistic Finite State Automaton (PFSA). The state set $Q$ consists of states that correspond to grid locations and one extra state denoted by $q_\ominus$. The necessity of this special state $q_\ominus$ is explained in the sequel. The grid squares are numbered in a pre-determined scheme such that each $q_i \in Q \setminus \{q_\ominus\}$ denotes a specific square in the discretized workspace. The particular numbering scheme chosen is irrelevant. In the absence of dynamic uncertainties and state estimation errors, the alphabet contains one uncontrollable event *i.e.* $\Sigma = \Sigma_C \bigcup \{u\}$ such that $\Sigma_C$ is the set of controllable events corresponding to the possible moves of the robot. The uncontrollable event $u$ is defined from each of the blocked states and leads to $q_\ominus$ which is a deadlock state. All other transitions (*i.e.* moves) are removed from the blocked states. Thus, if a robot moves into a blocked state, it uncontrollably transitions to the deadlock state $q_\ominus$ which is physically interpreted to be a collision. We further assume that the robot fails to recover from collisions which is reflected by making $q_\ominus$ a deadlock state. We note that $q_\ominus$ does not correspond to any physical grid location. The set of blocked grid locations along with the obstacle state $q_\ominus$ is denoted as $Q_{\mathsf{OBSTACLE}} \subseteqq Q$. Figure 1 illustrates the navigation automaton for a nine state discretized workspace with two blocked squares. Note that the only outgoing transition from the blocked states $q_1$ and $q_8$ is $u$. Next we augment the navigation FSA by specifying event generation probabilities defined by the map $\tilde{\pi} : Q \times \Sigma \to [0, 1]$ and the characteristic state-weight vector specified as $\chi : Q \to [-1, 1]$. The characteristic state-weight vector [20] assigns scalar weights to the PFSA states to capture the desirability of ending up in each state.

**Definition 13.** *The characteristic weights are specified for the*

*navigation automaton as follows:*

$$\chi(q_i) = \begin{cases} -1 & \text{if } q_i \equiv q_\ominus \\ 1 & \text{if } q_i \text{ is the goal} \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

In the absence of dynamic constraints and state estimation uncertainties, the robot can "choose" the particular controllable transition to execute at any grid location. Hence we assume that the probability of generation of controllable events is uniform over the set of moves defined at any particular state.

**Definition 14.** *Since there is no uncontrollable events defined at any of the unblocked states and no controllable events defined at any of the blocked states, we have the following consistent specification of event generation probabilities:* $\forall q_i \in Q, \sigma_j \in \Sigma$,

$$\tilde{\pi}(q_i, \sigma_j) = \begin{cases} \frac{1}{\text{No. of controllable events at } q_i}, & \text{if } \sigma_j \in \Sigma_C \\ 1, & \text{otherwise} \end{cases}$$

The boundaries are handled by "surrounding" the workspace with blocked position states shown as "boundary obstacles" in the upper part of Figure 1(c).

**Definition 15.** *The navigation model id defined to have identical connectivity as far as controllable transitions are concerned implying that every controllable transition or move (i.e. every element of* $\Sigma_C$*) is defined from each of the unblocked states.*

### *2.4. Decision-theoretic Optimization of PFSA*

The above-described probabilistic finite state automaton (PFSA) based navigation model allows us to compute optimally feasible path plans via the language-measure-theoretic optimization algorithm [20] described in Section 2. Keeping in line with nomenclature in the path-planning literature, we refer to the language-measure-theoretic algorithm as $\boldsymbol{\nu}^\star$ in the sequel. For the unsupervised model, the robot is free to execute any one of the defined controllable events from any given grid location (See Figure 1(b)). The optimization algorithm selectively disables controllable transitions to ensure that the formal measure vector of the navigation automaton is elementwise maximized. Physically, this implies that the supervised robot is constrained to choose among only the enabled moves at each state such that the probability of collision is minimized with the probability of reaching the goal simultaneously maximized. *Although* $\boldsymbol{\nu}^\star$ *is based on optimization of probabilistic finite state machines, it is shown that an optimal and feasible path plan can be obtained that is executable in a purely deterministic sense.*

Let $\mathbb{G}_{\mathsf{NAV}}$ be the unsupervised navigation automaton and $\mathbb{G}^\star_{\mathsf{NAV}}$ be the optimally supervised PFSA obtained by $\boldsymbol{\nu}^\star$. We note that $\nu^i_\#$ is the renormalized measure of the terminating plant $\mathbb{G}^\star_{\mathsf{NAV}}(\theta_{min})$ with substochastic event generation probability matrix $\widetilde{\Pi}^{\theta_{min}} = (1-\theta_{min})\widetilde{\Pi}$. Denoting the event generating function (See Definition 3) for $\mathbb{G}^\star_{\mathsf{NAV}}$ and $\mathbb{G}^\star_{\mathsf{NAV}}(\theta_{min})$ as $\tilde{\pi} : Q \times \Sigma \to Q$ and $\tilde{\pi}^{\theta_{min}} : Q \times \Sigma \to Q$ respectively, we have

$$\tilde{\pi}^{\theta_{min}}(q_i, \epsilon) = 1 \quad (4a)$$

$$\forall q_i \in Q, \sigma_j \in \Sigma,\ \tilde{\pi}^{\theta_{min}}(q_i, \sigma_j) = (1-\theta_{min})\tilde{\pi}(q_i, \sigma_j) \quad (4b)$$

**Notation 2.1.** *For notational simplicity, we use*

$$\nu^i_\#(L(q_i)) = \nu_\#(q_i) = \boldsymbol{\nu}_\#|_i$$
$$\text{where } \boldsymbol{\nu}_\# = \theta_{min}[I - (1-\theta_{min})\Pi^\#]^{-1}\boldsymbol{\chi}$$

**Definition 16** ($\boldsymbol{\nu}^\star$-path)**.** *A* $\boldsymbol{\nu}^\star$*-path* $\rho(q_i, q_j)$ *from state* $q_i \in Q$ *to state* $q_j \in Q$ *is defined to be an ordered set of PFSA states* $\rho = \{q_{r_1}, \cdots, q_{r_M}\}$ *with* $q_{r_s} \in Q$, $\forall s \in \{1, \cdots, M\}, M \leq \mathsf{CARD}(Q)$ *such that*

$$q_{r_1} = q_i \quad (5a)$$

$$q_{r_M} = q_j \quad (5b)$$

$$\forall i, j \in \{1, \cdots, M\},\ q_{r_i} \neq q_{r_j} \quad (5c)$$

$$\forall s \in \{1, \cdots, M\}, \forall t \leqq s,\ \nu_\#(q_{r_t}) \leqq \nu_\#(q_{r_s}) \quad (5d)$$

We reproduce without proof the following key results pertaining to $\boldsymbol{\nu}^\star$- planning as reported in [18].

**Lemma 1.** *There exists an enabled sequence of transitions from state* $q_i \in Q \setminus Q_{\mathsf{OBSTACLE}}$ *to* $q_j \in Q \setminus \{q_\ominus\}$ *in* $\mathbb{G}^\star_{\mathsf{NAV}}$ *if and only if there exists a* $\boldsymbol{\nu}^\star$*-path* $\rho(q_i, q_j)$ *in* $\mathbb{G}^\star_{\mathsf{NAV}}$.

**Proposition 1.** *For the optimally supervised navigation automaton* $\mathbb{G}^\star_{\mathsf{NAV}}$*, we have*

$$\forall q_i \in Q \setminus Q_{\mathsf{OBSTACLE}},\ L(q_i) \subseteqq \Sigma^\star_C$$

**Corollary 1.** ***(Obstacle Avoidance:)*** *There exists no* $\boldsymbol{\nu}^\star$*-path from any unblocked state to any blocked state in the optimally supervised navigation automaton* $\mathbb{G}^\star_{\mathsf{NAV}}$.

**Proposition 2** (Existence of $\boldsymbol{\nu}^\star$-paths)**.** *There exists a* $\boldsymbol{\nu}^\star$*-path* $\rho(q_i, q_{\mathsf{GOAL}})$ *from any state* $q_i \in Q$ *to the goal* $q_{\mathsf{GOAL}} \in Q$ *if and only if* $\nu_\#(q_i) > 0$.

**Corollary 2.** ***(Absence of Local Maxima:)*** *If there exists a* $\boldsymbol{\nu}^\star$*-path from* $q_i \in Q$ *to* $q_j \in Q$ *and a* $\boldsymbol{\nu}^\star$*-path from* $q_i$ *to* $q_{\mathsf{GOAL}}$ *then there exists a* $\boldsymbol{\nu}^\star$*-path from* $q_j$ *to* $q_{\mathsf{GOAL}}$*, i.e.,*

$$\forall q_i, q_j \in Q\Big(\exists \rho_1(q_i, q_{\mathsf{GOAL}}) \bigwedge \exists \rho_2(q_i, q_j) \Rightarrow \exists \rho(q_j, q_{\mathsf{GOAL}})\Big)$$

### *2.5. Optimal Tradeoff between Computed Path Length & Availability Of Alternate Routes*

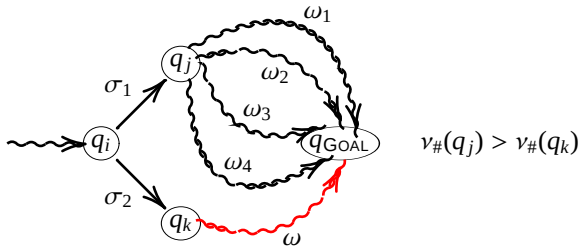



Figure 2: Tradeoff between path-length and robustness under dynamic uncertainty: $\sigma_2\omega$ is the shortest path to $q_{\mathsf{GOAL}}$ from $q_i$; but the $\boldsymbol{\nu}^\star$ plan may be $\sigma_1\omega_1$ due to the availability of larger number of feasible paths through $q_j$.

Majority of reported path planning algorithms consider minimization of the computed feasible path length as the sole optimization objective. However, the $\nu^\star$ algorithm can be shown to achieve an optimal trade-off between path lengths and availability of feasible alternate routes. If $\omega$ is the shortest path to goal from state $q_k$, then the shortest path from state $q_i$ (with $q_i \xrightarrow{\sigma_2} q_k$) is given by $\sigma_2\omega$. However, a larger number of feasible paths may be available from state $q_j$ (with $q_i \xrightarrow{\sigma_1} q_j$) which may result in the optimal $\nu^\star$ plan to be $\sigma_1\omega_1$. Mathematically, each feasible path from state $q_j$ has a positive measure which may sum to be greater than the measure of the single path $\omega$ from state $q_k$. The condition $\nu_\#(q_j) > \nu_\#(q_k)$ would then imply that the next state from $q_i$ would be computed to be $q_j$ and not $q_k$. Physically it can be interpreted that the mobile gent is better off going to $q_j$ since the goal remains reachable even if one or more paths become unavailable. The key results [18] are as follows:

**Lemma 2.** *For the optimally supervised navigation automaton* $\mathbb{G}^\star_{\text{NAV}}$*, we have* $\forall q_i \in Q \setminus Q_{\text{OBSTACLE}}$,

$$\forall \omega \in L(q_i),\ \nu^i_\#(\{\omega\}) = \theta_{min}\left(\frac{1-\theta_{min}}{\text{CARD}(\Sigma_C)}\right)^{|\omega|}\chi(\delta^\#(q_i,\omega))$$

**Proposition 3.** *For* $q_i \in Q \setminus Q_{\text{OBSTACLE}}$*, let* $q_i \xrightarrow{\sigma_1} q_j \to \cdots \to q_{\text{GOAL}}$ *be the shortest path to the goal. If there exists* $q_k \in Q \setminus Q_{\text{OBSTACLE}}$ *with* $q_i \xrightarrow{\sigma_2} q_k$ *for some* $\sigma_2 \in \Sigma_C$ *such that* $\nu_\#(q_k) > \nu_\#(q_j)$*, then the number of distinct paths to goal from state* $q_k$ *is at least* $\text{CARD}(\Sigma_C) + 1$.

The lower bound computed in Proposition 3 is not tight and if the alternate paths are longer or if there are multiple 'shortest' paths then the number of alternate routes required is significantly higher. Detailed examples can be easily presented to illustrate situation where $\nu^\star$ opts for a longer but more robust plan.

## 3. Generalizing The Navigation Automaton To Accommodate Uncertain Execution

In this paper, we modify the PFSA-based navigation model to explicitly reflect uncertainties arising from imperfect localization and the dynamic response of the platform to navigation commands. These effects manifest as uncontrollable transitions in the navigation automaton as illustrated in Figure 4. Note, while in absence of uncertainties and dynamic effects, one can disable transitions perfectly, in the modified model, such disabling is only partial. Choosing the probabilities of the uncontrollable transitions correctly allows the model to incorporate physical movement errors and sensing noise in an amortized fashion.

A sample run with a SEGWAY RMP at NRSL is shown in Figure 3. Note that the robot is unable to follow the plan exactly due to cellular discretization and dynamic effects. Such effects can be conceptually modeled by decomposing trajectory fragments into sequential combinations of controllable and uncontrollable inter-cellular moves as illustrated in Figure 4(c). We do not need to actually decompose trajectories, it is merely a conceptual construct that gives us a theoretical basis for computing the probabilities of uncontrollable transitions from observed robot dynamics (as described later in Section 5, and therefore incorporate the amortized effect of uncertainties in the navigation automaton.

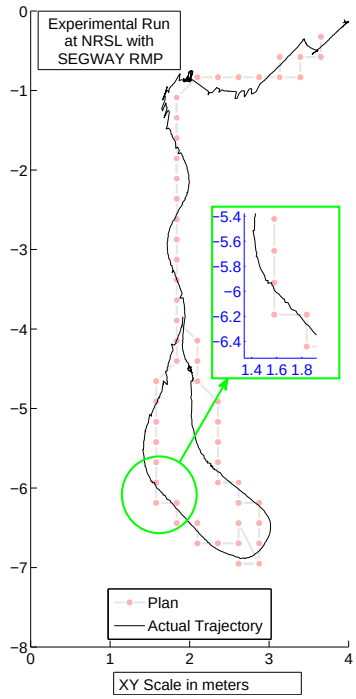



Figure 3: Plan execution with SEGWAY RMP at NRSL, Pennstate

### 3.1. The Modified Navigation Automaton

The modified navigation automaton $\mathbb{G}^{MOD}_{\text{NAV}} = (Q, \Sigma, \delta, \widetilde{\Pi}, \chi)$ is defined similar to the formulation in Section 2.3, with the exception that the alphabet $\Sigma$ is defined as follows:

$$\Sigma = \Sigma_C \cup \Sigma_{UC} \cup \{u\} \tag{6}$$

where $\Sigma_C$ is the set of controllable moves from any unblocked navigation state (as before), while $\Sigma_{UC}$ is the set of uncontrollable transitions that can occur as an effect of the platform dynamics and oather uncertainty effects. We assume that for each $\sigma \in \Sigma_C$, we have a corresponding event $\sigma_u$ in $\Sigma_{UC}$, such that both $\sigma$ and $\sigma_u$ represent the same physical move from a given navigation state; but while $\sigma$ is controllable and may be disabled, $\sigma_u$ is uncontrollable. Although for 2D circular robots we have: $\text{CARD}(\Sigma_C) = \text{CARD}(\Sigma_{UC})$, in general, there can exist uncontrollable moves reflecting estimation errors that cannot be realized via a single controllable move. For example, for planar rectangular robots with a non-zero minimum turn radius, there can be an uncontrollable shift in the heading without any change in the *xy*-positional coordinates, which may reflect errors in heading estimation, but such a move cannot be executed via controllable transitions due to the restriction on the minimum turn radius. We will discuss these issues in more details in the sequel.

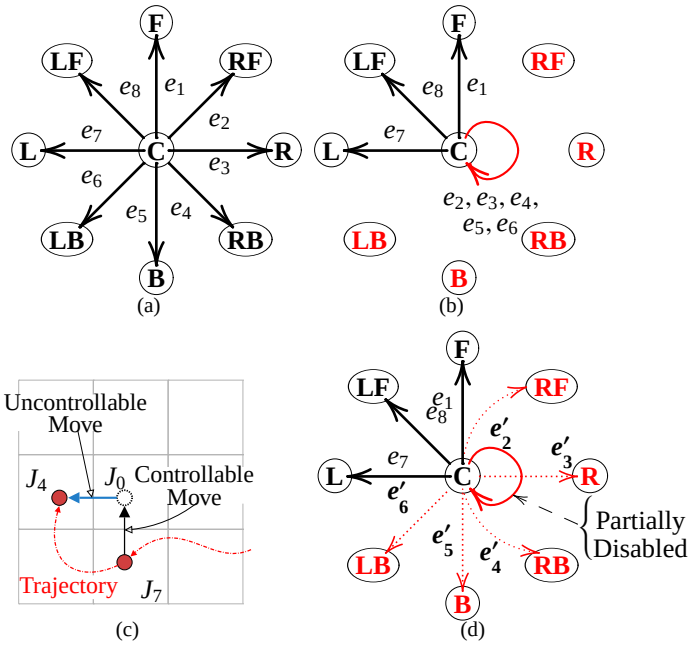



Figure 4: (a) shows available moves from the current state (C) in unsupervised navigation automaton. (b) shows the enabled moves in the optimally supervised PFSA with no dynamic uncertainty, (c) illustrates the case with dynamic uncertainty, so that the robot can still uncontrollably (and hence unwillingly) make the disabled transitions, albeit with a small probability, *i.e.*, probability of transitions $e'_2, e'_3, e'_4$ etc. is small. (d) illustrates the concept of using uncontrollable transitions to model dynamical response for a 2D circular robot: $J_0$ is the target cell from $J_7$, while the actual trajectory of the robot (shown in dotted line) ends up in $J_4$. We can model this trajectory fragment as first executing a controllable move to $J_0$ and then uncontrollably moving to $J_4$.

**Definition 17.** *The coefficient of dynamic deviation* $\gamma(\mathbb{G}^{MOD}_{\mathsf{NAV}})$ *is defined as follows:*

$$\gamma(\mathbb{G}^{MOD}_{\mathsf{NAV}}) = 1 - \max_{q_i \in Q} \sum_{\sigma_u \in \Sigma_{UC}} \tilde{\pi}(q_i, \sigma_u) \tag{7}$$

**Definition 18.** *The event generation probabilities for* $\mathbb{G}^{MOD}_{\mathsf{NAV}}$ *is defined as follows:* $\forall q_i \in Q \setminus \{q_{\mathsf{GOAL}}\}, \sigma_j \in \Sigma$,

$$\tilde{\pi}(q_i, \sigma_j) = \begin{cases} \frac{1 - \sum_{\sigma_u \in \Sigma_{UC}} \tilde{\pi}(q_i, \sigma_u)}{\text{No. of controllable events at } q_i}, & \text{if } \sigma_j \in \Sigma_C \\ \tilde{\pi}(q_i, \sigma_j), & \text{if } \sigma_j \in \Sigma_{UC} \\ 1, & \text{otherwise} \end{cases}$$

*and for the goal, we define as before:*

$$\tilde{\pi}(q_{\mathsf{GOAL}}, \sigma_j) = \begin{cases} \frac{1}{\text{No. of controllable events at } q_i}, & \text{if } \sigma_j \in \Sigma_C \\ 1, & \text{otherwise} \end{cases}$$

*Note that we assume there is no uncontrollability at the goal. This assumption is made for technical reasons clarified in the sequel and also to reflect the fact that once we reach the goal, we terminate the mission and hence such effects can be neglected.*

We note the following:

- In the idealized case where we assume platform dynamics is completely absent, we have $\tilde{\pi}(q_i, \sigma_u) = 0, \forall q_i \in Q, \forall \sigma_u \in \Sigma_{UC}$ implying that $\gamma(\mathbb{G}^{MOD}_{\mathsf{NAV}}) = 1$, while in practice, we expect $\gamma(\mathbb{G}^{MOD}_{\mathsf{NAV}}) < 1$.
- In Definition 17, we allowed for the possibility of $\tilde{\pi}(q_i, \sigma_u)$ being dependent on the particular navigation states $q_i \in Q$. A significantly simpler approach would be to redefine the probability of the uncontrollable events $\tilde{\pi}(q_i, \sigma_u)$ as follows:

$$\tilde{\pi}_{AV}(\sigma_u) = \frac{1}{\mathsf{CARD}(Q)} \sum_{q_i \in Q} \tilde{\pi}(q_i, \sigma_u) \tag{8}$$

  where $\tilde{\pi}_{AV}(\sigma_u)$ is the average probability of the uncontrollable event $\sigma_u$ being generated.

The averaging of the probabilities of uncontrollable transitions is justified in situations where we can assume that the dynamic response of the platform is not dependent on the location of the platform in the workspace. In this simplified case, the event generation probabilities for $\mathbb{G}^{MOD}_{\mathsf{NAV}}$ can be stated as: $\forall q_i \in Q \setminus \{q_{\mathsf{GOAL}}\}, \sigma_j \in \Sigma$,

$$\tilde{\pi}(q_i, \sigma_j) = \begin{cases} \frac{\gamma(\mathbb{G}^{MOD}_{\mathsf{NAV}})}{\text{No. of controllable events at } q_i}, & \text{if } \sigma_j \in \Sigma_C \\ \tilde{\pi}_{AV}(\sigma_j), & \text{if } \sigma_j \in \Sigma_{UC} \\ 1, & \text{otherwise} \end{cases}$$

The key difficulty is allowing the aforementioned dependence on states is not the decision optimization that would follow, but the complexity of identifying the probabilities; averaging results in significant simplification as shown in the sequel. Thus, even if we cannot realistically average out the uncontrollable transition probabilities over the entire state space, we could decompose the workspace to identify subregions where such an assumption is locally valid. In this paper, we do not address formal approaches to such decomposition, and will generally assume that the afore-mentioned averaging is valid throughout the workspace; the explicit identification of the sub-regions is more a matter of implementation specifics, and has little to do with the details of the planning algorithm presented here, and hence will be discussed elsewhere. In Section 5, we will address the computation of the probabilities of uncontrollable transitions from observed dynamics. First, we will establish the main planning algorithm as a solution to the performance optimization of the navigation automaton in the next section.

## 4. Optimal Planning Via Decision Optimization Under Dynamic Effects

The modified model $\mathbb{G}^{MOD}_{\mathsf{NAV}}$ can be optimized via the measure-theoretic technique in a straightforward manner, using the $\nu^\star$-algorithm reported in [18]. The presence of uncontrollable transitions in $\mathbb{G}^{MOD}_{\mathsf{NAV}}$ poses no problem (as far as the automaton optimization is concerned), since the underlying measure-theoretic optimization is already capable of handling such effects [20]. However the presence of uncontrollable transitions weakens some of the theoretical results obtained in [18] pertaining to navigation, specifically the absence of local maxima. We show

that this causes the $\nu^\star$ planner to lose some of its crucial advantages, and therefore must be explicitly addressed via a recursive decomposition of the planning problem.

**Proposition 4** (Weaker Version of Proposition 2)**.** *There exists a $\nu^\star$-path $\rho(q_i, q_{\text{GOAL}})$ from any state $q_i \in Q$ to the goal $q_{\text{GOAL}} \in Q$ if $\nu_\#(q_i) > 0$.*

*Proof.* We note that $\nu_\#(q_i) > 0$ implies that there necessarily exists at least one string $\omega$ of positive measure initiating from $q_i$ and hence there exists at least one string that terminates on $q_{\text{GOAL}}$. The proof then follows from the definition of $\nu^\star$-paths (See Definition 16). □

**Remark 2.** *Comparing with Proposition 2, we note that the only if part of the result is lost in the modified case.*

**Remark 3.** *We note that under the modified model, $\nu_\#(q_i) < 0$ needs to be interpreted somewhat differently. In absence of any dynamic uncertainty, $\nu_\#(q_i) < 0$ implies that no path to goal exists. However, due to weakening of Proposition 1 (See Proposition 4), $\nu_\#(q_i) < 0$ implies that the measure of the set of strings reaching the goal is smaller to that of the set of strings hitting an obstacle from the state $q_i$.*

The $\nu^\star$-planning algorithm is based on several abstract concepts such as the navigation automaton and the formal measure of symbolic strings. It is important to realize that in spite of the somewhat elaborate framework presented here, $\nu^\star$-optimization is free from heuristics, which is often not the case with competing approaches. In this light, the next proposition is critically important as it elucidates this concrete physical connection.

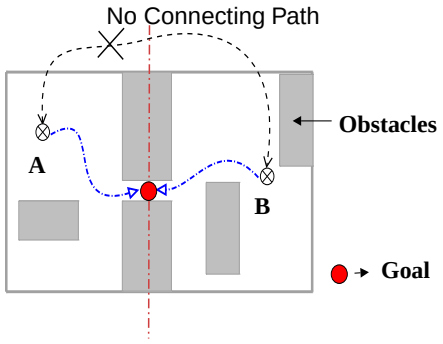



Figure 5: Absence of uncontrollable transitions at the goal imply that there is no path in the optimally disabled navigation automaton from point A to point B (or vice versa), since all controllable transitions will necessarily be disabled at the goal. It follows that the stationary probability vector may be different depending on whether one starts left or right to the goal. However, note that that any two points on the same side have a path (possibly made of uncontrollable transitions) between them; implying that the stationary probability vector will be identical if either of them is chosen as the start locations.

**Proposition 5.** *Given that a feasible path exists from the starting state to the goal, the $\nu^\star$ planning algorithm under non-trivial dynamic uncertainty (i.e. with $\gamma(\mathbb{G}_{\text{NAV}}^{MOD}) < 1$) maximizes the probability $\wp_{\text{GOAL}}$ of reaching the goal while simultaneously minimizing the probability $\wp_\ominus$ of hitting an obstacle.*

*Proof.* Let $\wp$ be the stationary probability vector for the stochastic transition probability matrix corresponding to the navigation automaton $\mathbb{G}_{\text{NAV}}^{MOD}$, for a starting state from which a feasible path to goal exists. (Note that $\wp$ may depend on the starting state; Figure 5 illustrates one such example. However, once we fix a particular starting state, the stationary vector $\wp$ is uniquely determined). The selective disabling of controllable events modifies the transition matrix and in effect alters $\wp$, such that $\wp^T\chi$ is maximized [20], where $\boldsymbol{\chi}$ is the characteristic weight vector, *i.e.* , $\boldsymbol{\chi}_i = \chi(q_i)$. Recalling that $\chi(q_{\text{GOAL}}) = 1, \chi(q_\ominus) = -1$ and $\chi(q_i) = 0$ if $q_i$ is neither the goal nor the abstract obstacle state $q_\ominus$, we conclude that the optimization, in effect, maximizes the quantity:

$$\psi = \wp_{\text{GOAL}} - \wp_\ominus \tag{9}$$

Also, note that the optimized navigation automaton has only two dump states, namely the goal $q_{\text{GOAL}}$ and the abstract obstacle state $q_\ominus$. That the goal $q_{\text{GOAL}}$ is in fact a dump state is ensured by not having uncontrollable transitions at the goal (See Definition 18). Hence we must have

$$\forall q_i \in Q \setminus \{q_{\text{GOAL}}, q_\ominus\},\ \wp_i = 0 \tag{10}$$

implying that

$$\wp_{\text{GOAL}} + \wp_\ominus = 1 \tag{11a}$$

$$\implies \psi = 2\wp_{\text{GOAL}} - 1 = 1 - 2\wp_\ominus \tag{11b}$$

Hence it follows that the optimization maximizes $\wp_{\text{GOAL}}$ and simultaneously minimizes $\wp_\ominus$. □

**Remark 4.** *It is easy to see that Proposition 5 remains valid if $\chi(q_{\text{GOAL}}) = \chi_{\text{GOAL}} > 1$. In fact, the result remains valid as long as the characteristic weight of the goal is positive and the characteristic weight of the abstract obstacle state $q_\ominus$ is negative.*

### *4.1. Recursive Problem Decomposition For Maxima Elimination*

Weakening of Proposition 1 (See Proposition 4) has the crucial consequence that Corollary 2 is no longer valid. Local maxima can occur under the modified model. This is a serious problem for autonomous planning and must be remedied. The problem becomes critically important when applied to solution of mazes; larger the number of obstables, higher is the chance of ending up in a local maxima. While elimination of local maxima is notoriously difficult for potential based planning approaches, $\nu^\star$ can be modified with ease into a recursive scheme that yields maxima-free plans in models with non-zero dynamic effects (*i.e.* with $\gamma(\mathbb{G}_{\text{NAV}}^{MOD}) < 1$).

It will be shown in the sequel that for successful execution of the algorithm, we may need to assign a larger than unity characteristic weight $\chi_{\text{GOAL}}$ to the goal $q_{\text{GOAL}}$. A sufficient lower bound for $\chi_{\text{GOAL}}$, with possible dependence on the recursion step, is given in Proposition 6. The basic recursion scheme can be described as follows (Also see the flowchart illustration in Algorithm 3):

1. In the first step (*i.e.*, at recursion step $k = 1$) we execute $\boldsymbol{\nu}^\star$-optimization on the given navigation automaton $\mathbb{G}_{\text{NAV}}^{MOD}$ and obtain the measure vector ${\boldsymbol{\nu}_\#}^{[k]}$.
2. We denote the set of states with strictly positive measure as $Q_k$ ($k$ denotes the recursion step), *i.e.*,

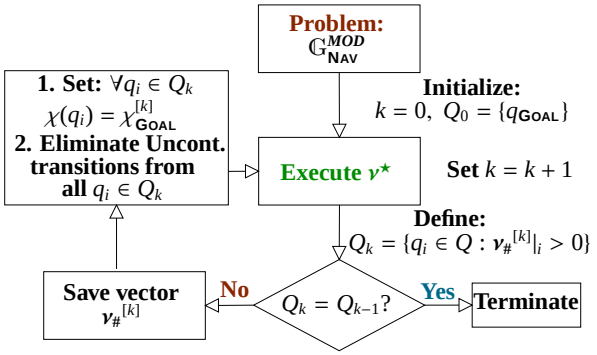
$$Q_k = \{q_i \in Q : {\boldsymbol{\nu}_\#}^{[k]}|_i > 0\} \tag{12}$$

3. If $Q_k = Q_{k-1}$, the recursion terminates; else we update the characteristic weights as follows:

$$\forall q_i \in Q_k,\ \chi(q_i) = \chi_{\text{GOAL}}^{[k]} \tag{13}$$

and continue the recursion by going back to the first step and incrementing the step number $k$.

**Algorithm 3**: Flowchart for recursive $\boldsymbol{\nu}^\star$-planning




**Proposition 6.** *If $\theta_{min}^{[k]}$ is the critical termination probability (See Definition 11) for the $\boldsymbol{\nu}^\star$-optimization in the $k^{th}$ recursion step of Algorithm 3, then the following condition*

$$\chi_{\text{GOAL}}^{[k]} > \frac{\text{CARD}(\Sigma_C)}{1 - \theta_{min}^{[k]}} \left(\frac{1}{\gamma} - 1\right) \tag{14}$$

*is sufficient to guarantee that the following statements are true:*

1. *If there exists a state $q_i \in Q \setminus Q_k$ from which at least one state $q_\ell \in Q_k$ is reachable in one hop, then ${\boldsymbol{\nu}_\#}^{[k]}|_i > 0$.*
2. *The recursion terminates in at most $\text{CARD}(Q)$ steps.*
3. *For the $k^{th}$ recursion step, either $Q_k \supsetneqq Q_{k-1}$ or no feasible path exists to $q_{\text{GOAL}}$ from any state $q_i \in Q \setminus Q_{k-1}$.*

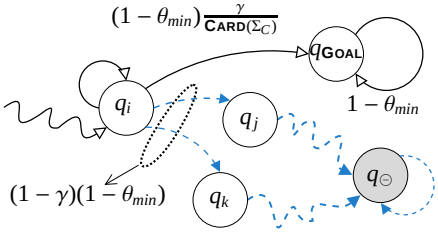



Figure 6: Illustration for Proposition 6. Uncontrollable events and strings are shown in dashed.

*Proof.* **Statement 1:**
We first consider the first recursion step, *i.e.*, the case where $k = 0$ and $Q_k = \{q_{\text{GOAL}}\}$ (See Algorithm 3). We note that the goal $q_{\text{GOAL}}$ achieves the maximum measure on account of the fact that only $q_{\text{GOAL}}$ has a positive characteristic weight, *i.e.*, we have

$$\forall q_i \in Q,\ {\boldsymbol{\nu}_\#}^{[1]}|_{\text{GOAL}} \geqq {\boldsymbol{\nu}_\#}^{[1]}|_i \tag{15}$$

It follows that all controllable transitions from the goal will be disabled in the optimized navigation automaton obtained at the end of the first recursion step (See Definition 8 and Algorithms 1 & 2), which in turn implies that the non-renormalized measure of the goal (at the end of the first recursion step) is given by $\chi_{\text{GOAL}} \frac{1}{\theta_{min}}$.

The Hahn Decomposition Theorem [46], allows us to write:

$$\boldsymbol{\nu}_\#|_i = \nu_\#(L^+(q_i)) + \nu_\#(L^-(q_i)) \tag{16}$$

where $L^+(q_i), L^-(q_i)$ are the sets of strings initiating from state $q_i$ that have positive and negative measures respectively.

Let $q_i \in Q \setminus \{q_{\text{GOAL}}\}$ such that $q_{\text{GOAL}}$ is reachable from $q_i$ in one hop. We note that since it is possible to reach the goal in one hop from $q_i$, we have:

$$\nu_\#(L^+(q_i)) \geqq \theta_{min} \times \frac{\gamma(1-\theta_{min})}{\text{CARD}(\Sigma_C)} \times \frac{\chi_{\text{GOAL}}}{\theta_{min}} \tag{17}$$

where the first term arises due to renormalization (See Definition 7), the second term denotes the probability of the transition leading to the goal and the third term is the non-renormalized measure of the goal itself (as argued above). Since it is obvious that the goal achieves the maximum measure, the transition to the goal will obviously be enabled in the optimized automaton, which justifies the second term. It is clear that there are many more strings of positive measure (*e.g.* arising due to the self loops at the state $q_i$ that correspond to the disabled controllable events that do not transition to the goal from $q_i$) which are not considered in the above inequality (which contributes to making the left hand side even larger); therefore guaranteeing the correctness of the lower bound stated in Eq.(17).

Next, we compute a lower bound for $\nu_\#(L^-(q_i))$. To that effect, we consider an automaton $G'$ identical to the navigation automaton at hand in ever respect, but the fact that the $q_{\text{GOAL}}$ has zero characteristic. We denote the state corresponding to $q_i$ in this hypothesized automaton as $q_i'$, and the set of al states in $G'$ as $Q'$. We claim that, after a measure-theoretic optimization (*i.e.* after applying Algorithms 1 and 2), the measure of $q_i'$, denoted as $\nu^\star(q_i')$, satisfies:

$$\nu^\star(q_i') \geqq -(1-\gamma) \tag{18}$$

To prove the claim in Eq. (18), we first note that denoting the renormalized measure vector for $G'$ before any optimization as $\boldsymbol{\nu}'$, the characteristic vector as $\chi'$ and for any termination probability $\theta \in (0,1)$, we have:

$$\begin{aligned} \|\boldsymbol{\nu}'\|_\infty &= \|\theta[\mathbb{I} - (1-\theta)\Pi]^{-1}\chi'\|_\infty \\ &\leqq \|\theta[\mathbb{I} - (1-\theta)\Pi]^{-1}\|_\infty \times 1 = 1 \end{aligned} \tag{19}$$

which follows from the following facts:

1. For all $\theta \in (0,1]$, $\theta[\mathbb{I} - (1-\theta)\Pi]^{-1}$ is a row-stochastic matrix and therefore has unity infinity norm [19]
2. $\|\chi'\|_\infty = 1$, since all entries of $\chi'$ are 0 except for the state corresponding to the obstacle state in the navigation automaton, which has a characteristic of $-1$.

Since the only non-zero characteristic is $-1$, it follows that no state in $G'$ can have a positive measure and we conclude from Eq. (19) that:

$$\forall q'_j \in Q',\ \nu(q'_j) \in [-1, 0] \tag{20}$$

Note that $q'_i$ is not blocked itself (since we chose $q_i$ such that a feasible 1-hop path to the goal exists from $q_i$). Next, we subject $G'$ to the measure-theoretic optimization (See Algorithms 1 & 2), which disables all controllable transitions to the blocked states. In order to compute a lower bound on the optimized measure for the state $q'_i$, (denoted by $\nu^\star(q'_i)$ ), we consider the worst case scenario where all neighboring states that can be reached from $q'_i$ in single hops are blocked. Denoting the set of all such neighboring states of $q_i$ by $\mathcal{N}(q'_i)$, we have:

$$\nu^\star(q'_i) = \sum_{q'_j \in \mathcal{N}(q'_i)} \Pi^u_{ij} \nu(q'_j) \geqq -1 \times \sum_{q'_j \in \mathcal{N}(q'_i)} \Pi^u_{ij} = -1 \times (1-\gamma) \tag{21}$$

where $\Pi^u_{ij}$ is the probability of the uncontrollable transition from $q'_i$ to the neighboring state $q'_j$. Note that we can write Eq. (21) in the worst case scenario where each state in $\mathcal{N}(q'_i)$ is blocked, since all controllable transitions from $q'_i$ will be disabled in the optimized plant under such a scenario, and only the uncontrollable transitions will remain enabled; and the probabilities of all uncontrollable transitions defined at state $q_i$ sums to $1-\gamma$. It is obvious that the lower bound computed in Eq. (21) also reflects a lower bound for $\nu_\#(L^-(q_i))$, since addition of state(s) with positive characteristic or eliminating obstacles cannot possibly make strings more negative. Furthermore, recalling that the goal $q_{\text{GOAL}}$ is actually reachable from state $q_i$ by a single hop, it follows that not all neighbors of $q_i$ in the navigation automaton are blocked, and hence we have the strict inequality:

$$\nu_\#(L^-(q_i)) > -(1-\gamma) \tag{22}$$

Combining Eqns. (17) and (22), we note that the following condition is sufficient for guaranteeing $\boldsymbol{\nu}_\#|_i > 0$.

$$\frac{\gamma(1-\theta_{min})}{\text{CARD}(\Sigma_C)} \times \chi_{\text{GOAL}} > 1-\gamma \tag{23}$$

which after a straightforward calculation yields the bound stated in Eq. 14, and the Statement 1 is proved for the first recursion step, *i.e.* for $Q_k = \{q_{\text{GOAL}}\}$.

To extend the argument to later recursion steps of Algorithm 3, *i.e.*, for $k > 0$, we argue as follows. Let $Q_k \supsetneqq \{q_{\text{GOAL}}\}$ and we have eliminated all uncontrollable transitions from all $q_j \in Q_k$ (as required in Algorithm 3). Further, let $q_i \in Q \setminus Q_k$ such that it is possible to reach some $q_j \in Q$ in a single controllable hop, *i.e.*

$$q_i \xrightarrow[controllable]{\sigma} q_j,\ q_j \in Q_k \tag{24}$$

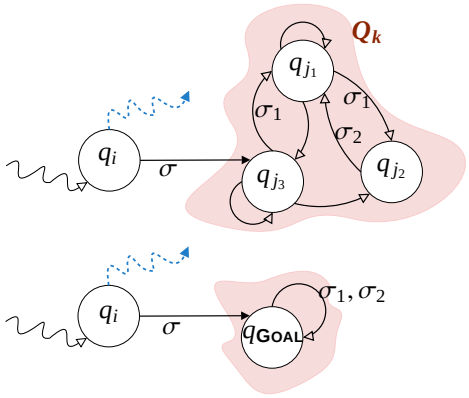



Figure 7: Illustration for Statement 1 of Proposition 6. Note that even if $Q_k$ has multiple states, $q_{j_1}, q_{j_2}, q_{j_3}$, the measure of any string (say $\sigma\sigma_1\sigma_1\sigma_2$) from $q_i$ is the same as if $q_i$ was directly connected to the goal $q_{\text{GOAL}}$ with all controllable events disabled at $q_{\text{GOAL}}$. The bottom plate illustrates this by showing the hypothetical scenario where $q_i$ is connected to $q_{\text{GOAL}}$ by $\sigma$ and $\sigma_1, \sigma_2$ are controllable events disabled at $q_{\text{GOAL}}$. Note that for this argument to work, we must eliminate uncontrollable transitions from all states in $Q_k$.

We first claim that

$$\forall q_j \in Q_k, q_r \notin Q_k,\ \boldsymbol{\nu}_\#^{[k]}|_j > \boldsymbol{\nu}_\#^{[k]}|_r \tag{25}$$

which immediately follows from the fact that the optimal configuration (of transitions from states in $Q_k$) at the end of the $\boldsymbol{\nu}^\star$-optimization at the $k^{th}$ step would be to have all controllable transitions from states $q_j \in Q_k$ enabled if and only if the transition goes to some state in $Q_k$, since in that case every string initiating from $q_j$ terminates on a state having characteristic $\chi_{\text{GOAL}}$ (since there is no uncontrollability from states within $Q_k$ by construction), whereas if a transition $q_j \xrightarrow[controllable]{\sigma} q_r$, where $q_j \in Q_k, q_r \notin Q_k$ allows strings which end up in zero-characteristic states and also (via uncontrollable transitions) on negative-characteristic states.

Eq. (25) implies that no enabled string exits $Q_k$. It therefore follows that every string $\sigma\omega$ starting from the state $q_i$, with $\omega \in \Sigma_C^\star$ and $\delta(q_i, \sigma) \in Q_k$ (*i.e.*, $\sigma$ leads to some state within $Q_k$) has exactly the same measure as if $q_i$ is directly connected to $q_{\text{GOAL}}$ and all controllable transitions are disabled at $q_{\text{GOAL}}$ (See Figure 7 for an illustration). This conceptual reduction implies that Eq. (17) is valid when $Q_k \supsetneqq \{q_{\text{GOAL}}\}$ since the lower bound for $\nu_\#(L^+(q_i))$ can be computed exactly as already done for the case with $Q_k = \{q_{\text{GOAL}}\}$. The argument for obtaining the lower bound for $\nu_\#(L^-(q_i))$ is the same as before, thus completing the proof for Statement 1 for all recursion steps of Algorithm 3 .

**Statement 2:**

Let $Q_R \subsetneqq Q$ be the set of states from which a feasible path to the goal exists. If $\text{CARD}(Q_R) = 1$, then we must have $Q_R = \{q_{\text{GOAL}}\}$ and the recursion terminates in one step. In general, for the $k^{th}$ recursion step, let $\text{CARD}(Q_k) < \text{CARD}(Q_R)$. Since there exists at least one state, not in $Q_k$, from which a feasible path to the goal exists, it follows that there exists at least one state $q_j$ from which it is possible to reach a state in $Q_k$ in one hop. Using Statement 1, we can then conclude:

$$Q_{k+1} \neq Q_k \Rightarrow \text{CARD}(Q_{k+1}) \geqq \text{CARD}(Q_k) + 1$$

$$\Rightarrow \textsc{Card}(Q_{k+1}) \geqq k+1 \tag{26}$$

which immediately implies that the recursion must terminate in at most $\textsc{Card}(Q)$ steps.

**Statement 3:**

Follows immediately from the argument used for proving Statement 2. □

**Remark 5.** *The generality of Eq.* (14) *is remarkable. Note that the lower bound is not directly dependent on the exact structure of the navigation automaton; what only matters is the number of controllable moves available at each state, the coefficient of dynamic deviation* $\gamma(\mathbb{G}_{\textsc{Nav}}^{MOD})$ *and the critical termination probability* $\theta_{min}$*. Although the exact automaton structure and the probability distribution of the uncontrollable transitions are not directly important, their effect enters, in a somewhat non-trivial fashion, through the value of the critical termination probability. The reader might want to review Algorithm 2 (See also [19, 20]) which computes the critical termination probability in each step of the* $\nu^\star$*-optimization for a better elucidation of the aforementioned connection between the structure of the navigation automaton and* $\chi_{\textsc{Goal}}$*.*

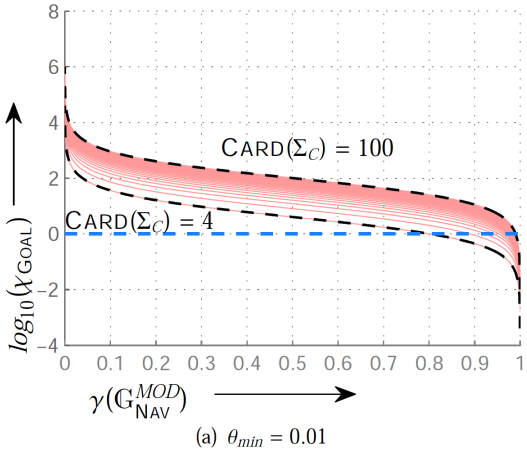
The dependencies of the acceptable lower bound for $\chi_{\textsc{Goal}}$ with the coefficient of dynamic deviation $\gamma(\mathbb{G}_{\textsc{Nav}}^{MOD})$, as computed in Proposition 6, is illustrated in Figures 8(a) and (b). The key points to be noted are:

1. As $\gamma(\mathbb{G}_{\textsc{Nav}}^{MOD}) \to 0^+, \chi_{\textsc{Goal}} \to +\infty$; which reflects the physical fact that if no events are controllable, then we cannot optimize the mission plan no matter how large $\chi_{\textsc{Goal}}$ is chosen.
2. As $\gamma(\mathbb{G}_{\textsc{Nav}}^{MOD}) \to 1, \chi_{\textsc{Goal}} \to 0$; which implies that in the absence of dynamic effects any positive value of $\chi_{\textsc{Goal}}$ suffices. This reiterates the result obtained with $\chi_{\textsc{Goal}} = 1$ in [18].
3. As the number of available controllable moves increases (See Figure 8(a)), we need a larger value of $\chi_{\textsc{Goal}}$; similarly if the critical termination probability $\theta_{min}$ is large, then the value of $\chi_{\textsc{Goal}}$ required is also large (See Figure 8(b)).
4. The functional relationships in Figures 8(a) and (b) establish the fact that for relatively smaller number of controllable moves, a large value of $\gamma(\mathbb{G}_{\textsc{Nav}}^{MOD})$ and a small termination probability, a constant value of $\chi_{\textsc{Goal}} = 1$ may be sufficient.

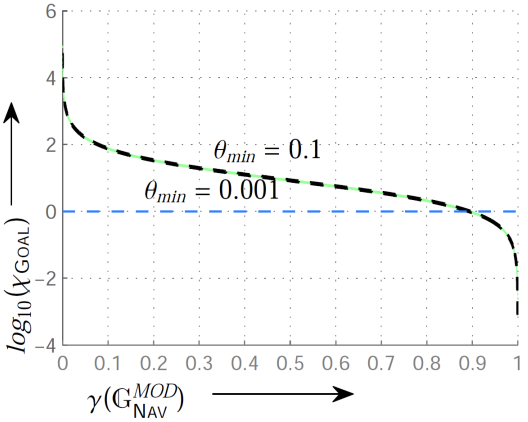
### 4.2. Plan Assembly & Execution Approach

The plan vectors $\boldsymbol{\nu}_\#^{[k]}$ (Say, there are $K$ of them, *i.e.*, $k \in \{1, \cdots, K\}$) obtained via the recursive planning algorithm described above, can be used for subsequent mission execution in two rather distinct ways:

1. *(The Direct Approach:)*
   - At any point during execution, if the current state $q_i \in Q_k$ for some $k \in \{1, \cdots, K\}$, then use the gradient defined by the plan vector $\boldsymbol{\nu}_\#^{[k]}$ to decide on the next move, *i.e.*, $q_j$ is an acceptable next state if $\boldsymbol{\nu}_\#^{[k]}|_j > \boldsymbol{\nu}_\#^{[k]}|_i$ and for states $q_\ell$ that can be reached from the current state $q_i$ via controllable events, we have $\boldsymbol{\nu}_\#^{[k]}|_j \geqq \boldsymbol{\nu}_\#^{[k]}|_\ell$.
   - if $\forall k \in \{1, \cdots, K\}$, $q_i \notin Q_k$, then terminate operation because there is no feasible path to the goal.
   - Note that this entails keeping $K$ vectors in memory.
2. *(The Assembled Plan Approach:)*
   - Use $\boldsymbol{\nu}_\#^{[k]}, k \in \{1, \cdots, K\}$ to obtain the assembled plan vector $\boldsymbol{\nu}_\#^{\mathbf{A}}$ following Algorithm 4, which assigns a real value $\boldsymbol{\nu}_\#^{\mathbf{A}}|_i$ to each state $q_i$ in the workspace. We refer to this map as the assembled plan.
   - Make use of the gradient defined by $\boldsymbol{\nu}_\#^{\mathbf{A}}$ to reach the goal, by sequentially moving to states with increasing values specified by the assembled plan, *i.e.*,. if the current state is $q_i \in Q$, then $q_j$ is an acceptable next state if $\boldsymbol{\nu}_\#^{\mathbf{A}}|_j > \boldsymbol{\nu}_\#^{\mathbf{A}}|_i$ and for states $q_\ell$ that can be reached from the current state $q_i$ via controllable events, we have $\boldsymbol{\nu}_\#^{\mathbf{A}}|_j \geqq \boldsymbol{\nu}_\#^{\mathbf{A}}|_\ell$.




(a) $\theta_{min} = 0.01$




(b) $\textsc{Card}(\Sigma_C) = 8$

Figure 8: Variation of the acceptable lower bound for $\chi_{\textsc{Goal}}$ with $\gamma(\mathbb{G}_{\textsc{Nav}}^{MOD})$. (a) The set of controllable moves is expanded from $\textsc{Card}(\Sigma_C) = 4$ to $\textsc{Card}(\Sigma_C) = 100$ while holding $\theta_{min} = 0.01$ (b) The critical termination probability $\theta_{min}$ is varied from 0.001 to 0.1 while holding $\textsc{Card}(\Sigma_C) = 8$. Note the lines are almost coincident in this case.

**Algorithm 4**: Assembly of Plan Vectors

**input** : $\boldsymbol{\nu}_{\#}^{[k]}, k = 1, \cdots, K$
(Plan Vectors) **output**: $\boldsymbol{\nu}_{\#}^{\mathbf{A}}$ (Assembled Plan)
**begin**
  Set $\boldsymbol{\nu}_{\#}^{\mathbf{A}} = \mathbf{0}$; /* Zero vector */
  **for** $k = 1 : K$ **do**
    **for** $i \in Q$ **do**
      $\boldsymbol{\nu}_{\#}^{tmp}|_i = 0$;
      **if** $\boldsymbol{\nu}_{\#}^{[k-1]}|_i > 0$ **then**
        $\boldsymbol{\nu}_{\#}^{tmp}|_i = 1$;
      **else**
        **if** $\boldsymbol{\nu}_{\#}^{[k]}|_i > 0$ **then**
          $\boldsymbol{\nu}_{\#}^{tmp}|_i = \boldsymbol{\nu}_{\#}^{[k]}|_i$;
        **endif**
      **endif**
    **endfor**
    $\boldsymbol{\nu}_{\#}^{\mathbf{A}} = \boldsymbol{\nu}_{\#}^{\mathbf{A}} + \boldsymbol{\nu}_{\#}^{tmp}$;
  **endfor**
**end**

- We show in the sequel that if $\boldsymbol{\nu}_{\#}^{\mathbf{A}}|_i < 0$, then no feasible path exists to the goal.

Before we can proceed further, we need to formally establish some key properties of the assembled plan approach. In particular, we have the following proposition:

**Proposition 7.** 1. *For a state $q_i \in Q$, a feasible path to the goal exists from the state $q_i$, if and only if $\boldsymbol{\nu}_{\#}^{\mathbf{A}}|_i > 0$.*

2. *The assembled plan $\boldsymbol{\nu}_{\#}^{\mathbf{A}}$ is free from local maxima, i.e., if there exists a $\boldsymbol{\nu}^{\star}$-path (w.r.t. to $\boldsymbol{\nu}_{\#}^{\mathbf{A}}$) from $q_i \in Q$ to $q_j \in Q$ and a $\boldsymbol{\nu}^{\star}$-path from $q_i$ to $q_{\mathsf{GOAL}}$ then there exists a $\boldsymbol{\nu}^{\star}$-path from $q_j$ to $q_{\mathsf{GOAL}}$, i.e.,*

$$\forall q_i, q_j \in Q\Big(\exists \rho_1(q_i, q_{\mathsf{GOAL}}) \bigwedge \exists \rho_2(q_i, q_j) \Rightarrow \exists \rho(q_j, q_{\mathsf{GOAL}})\Big)$$

3. *If a feasible path to the goal exists from the state $q_i$, then the agent can reach the goal optimally by following the gradient of $\boldsymbol{\nu}_{\#}^{\mathbf{A}}$, where the optimality is to be understood as maximizing the probability of reaching the goal while simultaneously minimizing the probability of hitting an obstacle (i.e. in the sense stated in Proposition 5).*

*Proof.* **Statement 1:**
Let the plan vectors obtained by the recursive procedure stated in the previous section be $\boldsymbol{\nu}_{\#}^{[k]}$ (Say, there are $K$ of them, *i.e.*, $k \in \{1, \cdots, K\}$) and further let the current state $q_i \in Q_k$ for some $k \in \{1, \cdots, K\}$. We observe that on account of Proposition 4, if $k = 1$, then $\boldsymbol{\nu}_{\#}^{[k]}|_i > 0$ is sufficient to guarantee that there exists a $\boldsymbol{\nu}^{\star}$-path $\rho(q_i, q_{\mathsf{GOAL}})$ w.r.t the plan vector $\boldsymbol{\nu}_{\#}^{[1]}$. We further note that $\boldsymbol{\nu}_{\#}^{[1]}|_i <= 0 \Rightarrow q_i \notin Q_1$ (See Eq. (12)), implying that $\boldsymbol{\nu}_{\#}^{[1]}|_i > 0$ is also necessary for the existence of $\rho(q_i, q_{\mathsf{GOAL}})$. Extending this argument, we note that, for $k > 1$, a $\boldsymbol{\nu}^{\star}$-path $\rho(q_i, q_j)$ with $q_j \in Q_{k-1}$ exists (with respect to the plan vector $\boldsymbol{\nu}_{\#}^{[k]}$) if and only if $\boldsymbol{\nu}_{\#}^{[k]}|_i > 0$. Noting that $\boldsymbol{\nu}_{\#}^{[k]}|_i > 0 \Leftrightarrow \boldsymbol{\nu}_{\#}^{\mathbf{A}}|_i > 0$, (See Algorithm 4) we conclude that a $\boldsymbol{\nu}^{\star}$-path $\rho(q_i, q_j)$ with $q_j \in Q_{k-1}$ exists (with respect to the plan vector $\boldsymbol{\nu}_{\#}^{[k]}$) if and only if $\boldsymbol{\nu}_{\#}^{\mathbf{A}}|_i > 0$. Also, since $q_j \in Q_{k-1} \wedge q_i \in Q_k$, it follows from Algorithm 4, that $\boldsymbol{\nu}_{\#}^{\mathbf{A}}|_j \geqq 1 + \boldsymbol{\nu}_{\#}^{\mathbf{A}}|_i > 0$. It follows that the same argument can be used recursively to find $\boldsymbol{\nu}^{\star}$-paths $\rho(q_j, q_{\ell_1}), \cdots, \rho(q_j, q_{\mathsf{GOAL}})$ if and only if $\boldsymbol{\nu}_{\#}^{\mathbf{A}}|_i > 0$.

To complete the proof, we still need to show that if there exists a feasible path from a state $q_i$ to the goal $q_{\mathsf{GOAL}}$, then there exists a $\boldsymbol{\nu}^{\star}$-path $\rho(q_i, q_{\mathsf{GOAL}})$. We argue as follows: Let $q_i = q_{r1} \rightarrow q_{r2} \rightarrow \cdots \rightarrow q_{r_{m-1}} \rightarrow q_{r_m} = q_{\mathsf{GOAL}}$ be a feasible path from the state $q_i$ to $q_{\mathsf{GOAL}}$. Furthermore, assume if possible that

$$\forall k \; \boldsymbol{\nu}_{\#}^{[k]}|_i \leqq 0 \tag{27}$$

*i.e.*, there exists no $\boldsymbol{\nu}^{\star}$-path from $q_i$ to $q_{\mathsf{GOAL}}$ w.r.t $\boldsymbol{\nu}_{\#}^{\mathbf{A}}$. We observe that since it is possible to reach $q_{\mathsf{GOAL}}$ from $q_{r_{m-1}}$ in one hop, using Proposition 6 we have:

$$\boldsymbol{\nu}_{\#}^{[1]}|_{r_{m-1}} > 0 \Rightarrow q_{r_{m-1}} \in Q_1 \tag{28}$$

We further note:

$$\boldsymbol{\nu}_{\#}^{[1]}|_{r_{m-2}} > 0 \Rightarrow q_{r_{m-2}} \in Q_1 \tag{29}$$

$$\boldsymbol{\nu}_{\#}^{[1]}|_{r_{m-2}} \leqq 0 \Rightarrow \boldsymbol{\nu}_{\#}^{[2]}|_{r_{m-2}} > 0 \Rightarrow q_{r_{m-2}} \in Q_2 \tag{30}$$

Hence, we conclude either $q_{r_{m-2}} \in Q_2$ or $q_{r_{m-2}} \in Q_1$. It follows by straightforward induction that either $q_1 \in Q_{m-1}$ or $q_1 \in Q_{m-2}$, which contradicts the statement in Eq. (27). Therefore, we conclude that if a feasible path to the goal exists from any state $q_i$, then a $\boldsymbol{\nu}^{\star}$-path $\rho(q_i, q_{\mathsf{GOAL}})$ (w.r.t $\boldsymbol{\nu}_{\#}^{\mathbf{A}}$) exists as well. This completes the proof of Statement 1.

**Statement 2:**
Given states $q_i, q_j \in Q$, assume that we have the $\boldsymbol{\nu}^{\star}$-paths $\rho_1(q_i, q_{\mathsf{GOAL}})$ and $\rho_2(q_i, q_j)$. We observe that:

$$\exists \rho_1(q_i, q_{\mathsf{GOAL}}) \Rightarrow \boldsymbol{\nu}_{\#}^{\mathbf{A}}|_i > 0 \text{ (See Statement 1)} \tag{31a}$$

$$\exists \rho_2(q_i, q_j) \Rightarrow \boldsymbol{\nu}_{\#}^{\mathbf{A}}|_j \geqq \boldsymbol{\nu}_{\#}^{\mathbf{A}}|_i \text{ (See Definition 16)} \tag{31b}$$

$$\Rightarrow \boldsymbol{\nu}_{\#}^{\mathbf{A}}|_j > 0 \Rightarrow \exists \rho(q_j, q_{\mathsf{GOAL}}) \text{ (See Statement 1)} \tag{31c}$$

which proves Statement 2.

**Statement 3:**
Statements 1 and 2 guarantee that if a feasible path to the goal exists from a state $q_i \in Q$, then an agent can reach the goal by following a $\boldsymbol{\nu}^{\star}$-path (w.r.t $\boldsymbol{\nu}_{\#}^{\mathbf{A}}$) from $q_i$, *i.e.*, by sequentially moving to states which have a better measure as compared to the current state.

We further note that a $\boldsymbol{\nu}^{\star}$-path $\omega$ w.r.t $\boldsymbol{\nu}_{\#}^{\mathbf{A}}$ from any state $q_i$ to $q_{\mathsf{GOAL}}$ can be represented as a concatenated sequence $\omega_1\omega_2\cdots\omega_r\cdots\omega_m$ where $\omega_r$ is a $\boldsymbol{\nu}^{\star}$-path from some intermediate state $q_j \in Q_s$, for some $s \in \{1, \cdots, K\}$, to some state $q_\ell \in Q_{s-1}$. Since the recursive procedure optimizes all such intermediate plans, and since the outcome "reached goal from $q_i$" can be visualized as the intersection of the mutually independent outcomes "reached $Q_s$ from $q_i \in Q_{s-1}$", "reached $Q_{s+1}$ from $q_j \in Q_s$" , $\cdots$ , "reached $q_{\mathsf{GOAL}}$ from $q_\ell \in Q_1$", the overall path must be optimal as well. This completes the proof. □

We compute the set of acceptable next states from the following definition.

**Definition 19.** *Given the current state $q_i \in Q$, $Q_{next}$ is the set of states satisfying the strict inequality:*

$$Q_{next} = \{q_j \in Q : \nu_{\#}{}^{\mathbf{A}}|_j > \nu_{\#}{}^{\mathbf{A}}|_i\} \tag{32}$$

We note that Proposition 7 implies that $Q_{next}$ is empty if and only if the current state is the goal or if no feasible path to the goal exists from the current state.

## 5. Computation of Amortized Uncertainty Parameters

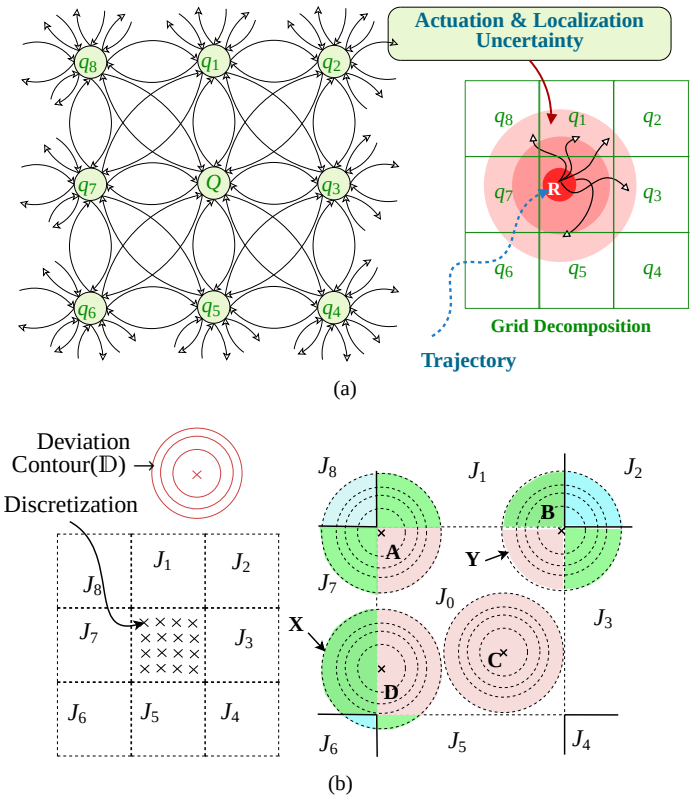



Figure 9: (a) Model for 2D circular robot(b) Numerical integration technique for computing the dynamic parameters for the case of a circular robot model*e.g.* a SEGWAY RMP 200

Specific numerical values of the uncertainty parameters, *i.e.* the probability of uncontrollable transitions in the navigation automaton can be computed from a knowledge of the average uncertainty in the robot localization and actuation in the configuration space. For simplicity of exposition, we assume a 2D circular robot; however the proposed techniques are applicable to more general scenarios. The complexity of this identification is related to the dynamic model assumed for the platform (*e.g.* circular robot in a 2D space, rectangular robot with explicit heading in the configuration etc.), the simplifying assumptions made for the possible errors, and the *degree of averaging* that we are willing to make. Uncertainties arise from two key sources:

1. Actuation errors: Inability of the robot to execute planned motion primitives exactly, primarily due to the dynamic response of the physical platform.
2. Localization errors: Estimation errors arising from sensor noise, and the limited time available for post-processing exteroceptive data for a moving platform. Even if we assume that the platform is capable of processing sensor data to eventually localize perfectly for a static robot, the fact that we have to get the estimates while the robot pose is changing in real time, implies that the estimates lag the actual robot configuration. Thus, this effect cannot be neglected even for the best case scenario of a 2D robot with an accurate global positioning system (unless the platform speed is sufficiently small).

In our approach, we do not distinguish between the different sources of uncertainty, and attempt to represent the overall amortized effect as uncontrollability in the navigation automaton. The rationale for this approach is straightforward: we visualize actuation errors as the uncontrollable execution of transitions before the controllable planned move can be executed, and for localization errors, we assume that any controllable planned move is followed by an uncontrollable transition to the actual configuration. Smaller is the probability of the uncontrollable transitions in the navigation automaton, *i.e.*, larger is the coefficient of dynamic deviation for each state, smaller is the uncertainty in navigation. From a history of observed dynamics or from prior knowledge, one can compute the distribution of the robot pose around the estimated configuration (in an amortized sense). Then the probability of uncontrollable transitions can be estimated by computing the probabilities with which the robot must move to the neighboring cells to approximate this distribution. The situation for a 2D circular robot is illustrated in Figure 9(a), where we assume that averaging over the observations lead to a distribution with zero mean-error; *i.e.*, the distribution is centered around the estimated location in the configuration space. For more complex scenarios (as we show in the simulated examples), this assumption can be easily relaxed. We call this distribution the *deviation contour* ($\mathbb{D}$) in the sequel. The amortization or averaging is involved purely in estimating the deviation contour from observed dynamics (or from prior knowledge); a simple methodology for which will be presented in the sequel. However, we first formalize the computation of the uncertainty parameters from a knowledge of the deviation contour.

For that purpose, we consider the current state in the navigation automaton to be $q_i$. Recall that $q_i$ maps to a set of possible configurations in the workspace. For a 2D circular robot, $q_i$ corresponds to a set of $x-y$ coordinates that the robot can occupy, while for a rectangular robot, $q_i$ maps to a set of $(x, y, \theta)$ coordinates. The footprint of the navigation automaton states in the configuration space can be specified via the map $\xi : Q \to 2^C$, where $C$ is the configuration space of the robot. In general, for a given current state $q_i$, we can identify the set $\mathcal{N}(q_i) \subset Q$ of neighboring states that the robot can transition to in one move. The current state $q_i$ is also included in $\mathcal{N}(q_i)$ for notational simplicity. In case of the 2D circular robot model considered in this paper, the cardinality of $\mathcal{N}(q_i)$ is 8 (provided of course that $q_i$ is not blocked and is not a boundary state). For a position $s \in \xi(q_i)$ of the robot, we denote a neighborhood of radius $r$ of

the position $s$ in the configuration space as $\mathcal{B}_{s,r}$. The normalized "volume" intersections of $\mathcal{B}_{s,r}$ with the footprints of the states included in $\mathcal{N}(q_i)$ in the configuration space can be expressed as :

$$F_j(s,r) = \frac{\int_{A_j} \mathrm{d}x}{\int_{\cup_{q_j} A_j} \mathrm{d}x}, \forall q_j \in \mathcal{N}(q_i) \tag{33}$$

where $A_j = \mathcal{B}_{s,r} \bigcap \xi(q_j)$ and $\mathrm{d}x$ is the appropriate Lebesgue measure for the continuous configuration space.

We observe that the expected or the average probability of the robot deviating to a neighboring state $q_j \in \mathcal{N}(q_i)$ from a location $s \in \xi(q_i)$ is given by:

$$\int_0^\infty F_j(s,r)\mathbb{D}\mathrm{d}r \tag{34}$$

Hence, the probability $\Pi_{ij}^{uc}$ of uncontrollably transitioning to a neighboring state $q_j$ from the current state $q_i$ is obtained by considering the integrated effect of all possible positions of the robot within $\xi(q_i)$, *i.e.* we have:

$$\Pi_{ij}^{uc} = \frac{\int_{\xi(q_i)} \int_0^\infty F_j(s,r)\mathbb{D}\mathrm{d}r\mathrm{d}s}{\sum_{q_j \in \mathcal{N}(q_i)} \int_{\xi(q_i)} \int_0^\infty F_j(s,r)\mathbb{D}\mathrm{d}r\mathrm{d}s} \tag{35}$$

where $\mathrm{d}r, \mathrm{d}s$ are appropriate Lebesgue measures on the continuous configuration space of the robot. It is important to note that the above formulation is completely general and makes no assumption on the structure of the configuration space, *e.g.*, the calculations can be carried out for 2D circular robots, rectangular robots or platforms with more complex kinematic constraints equally well. Figures 13(a)-(c) illustrate the computation for a circular robot with eight controllable moves, *.e.g.*, the situation for a SEGWAY RMP. The 2D circular case is however the simplest, where any state that can be reached by an uncontrollable transition, can also be reached by a controllable move. For more complex scenarios, this may not be the case. For example, in the rectangular model, with constraints on minimum turn radius, the robot may not be able to move via a controllable transition from $(x, y, h_1)$ to $(x, y, h_2)$, where $h_i, i = 1, 2$ is the heading in the initial and final configurations. However, there most likely will be an uncontrollable transition that causes this change, reflecting uncertainty in the heading estimation (See Figure 10). Also, one can reduce the averaging effect by considering more complex navigation automata. For example, for a 2D circular robot, the configuration state can be defined to be $(x_{previous}, y_{previous}, x_{current}, y_{current})$, *i.e.* essentially considering a 4D problem. The identification of the uncertainty parameters on such a model will capture the differences in the uncontrollable transition probabilities arising from arriving at a given state from different directions. While the 2D model averages out the differences, the 4D model will make it explicit in the specification of the navigation automaton (See Figure 15). In the sequel, we will present comparative simulation results for these models. Note, in absence of uncertainty, the 4D implementation is superfluous; $(x_{previous}, y_{previous}, x_{current}, y_{current})$ has no more information than $(x_{current}, y_{current})$ in that case.

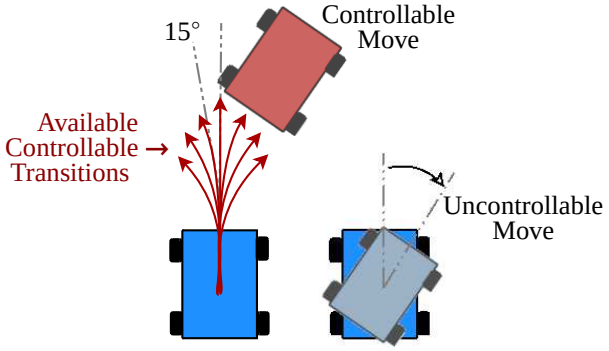



Figure 10: A rectangular robot unable to execute zero-radius turns. There exists uncontrollable transitions that alter heading in place, which reflects uncertainties in heading estimation, although there are no controllable moves that can achieve this transition

Next, we present a methodology for computing the relevant uncertainty parameters as a function of the robot dynamics. We assume a modular plan execution framework, in which the low-level continuous controller on-board the robotic platform is sequentially given a target cell (neighboring to the current cell) to go to, as it executes the plan. The robot may be able to reach the cell and subsequently receives the next target, or may end up in a different cell due to dynamic constraints, when it receives the next target from this deviated cell as dictated by the computed plan. The inherent dynamical response of the particular robot determines how well the patform is able to stick to the plan. We formulate a framework to compute the probabilities of uncontrollable transitions that best describe these deviations.

**Definition 20.** *The raw deviation $\Delta_R(t)$ as a function of the operation time $t$ is defined as follows:*

$$\Delta_R(t) = \Theta(p(t), \zeta(t)) \tag{36}$$

*where $p(t)$ is the current location of the robot in the workspace coordinates, $\zeta(t)$ is the location of the point within the current target cell which is nearest to the robot position $p(t)$ (See Figure 11), and $\Theta(\cdot,\cdot)$ is an appropriate distance metric in the configuration space.*

The robot will obviously take some time to reach the target cell, assuming it is actually able to do so. We wish to eliminate the effect of this delay from our calculations, since a platform that is able to sequentially reach each target cell, albeit with some delay, does not need the plan to be modified. Furthermore, unless velocity states are incorporated in the navigation automata, the plan cannot be improved for reducing this delay. We note that the raw deviation $\Delta_R(t)$ incorporates the effect of this possibly variable delay and needs to be corrected for. We do so by introducing the delay corrected deviation $\Delta(t)$ as follows:

**Definition 21.** *The delayed deviation $\Delta_d(t, \eta)$ is defined as:*

$$\Delta_d(t, \eta) = \Theta(p(t + \eta(t)), \zeta(t)) \tag{37}$$

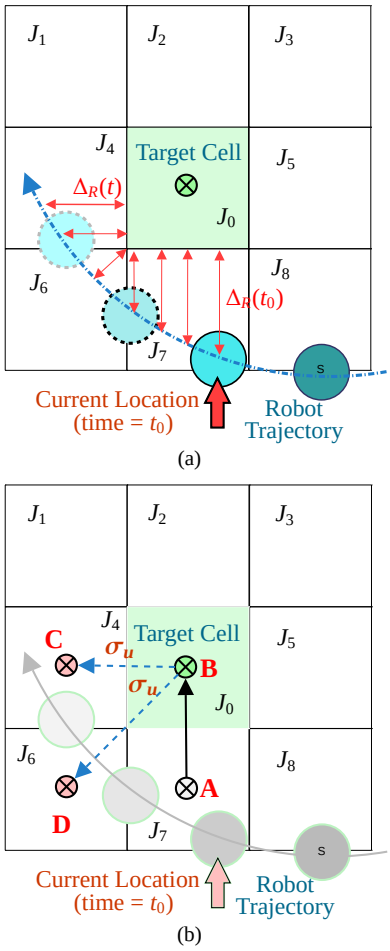



Figure 11: Illustration for computation of amortized dynamic uncertainty parameters

*where $\eta(t)$ is some delay function satisfying $\forall \tau \in \mathbb{R}, \eta(\tau) \geq 0$.*

**Definition 22.** *The delay corrected deviation $\Delta(t)$ as a function of the operation time $t$ is defined as:*

$$\forall t \in \mathbb{R},\ \Delta(t) = \underset{\Delta_d(t,\eta)\ :\ \forall \tau \in \mathbb{R}, \eta(\tau) \geq 0}{argin f} \left\| \Delta_d(t,\eta) \right\| \tag{38}$$

Note that Definition 22 incorporates the possibility that the delay may vary in the course of mission execution. *We will make the assumption that although the delay may vary, it does so slowly enough to be approximated as a constant function over relatively short intervals.*

If we further assume that we can make observations only at discrete intervals, we can approximately compute $\Delta(t)$ over a short interval $I = [t_{init}, t_{final}]$ as follows:

$$\forall t \in I, \Delta(t) = \underset{\Theta(p(t+\eta),\zeta(t)):\eta \in \mathbb{N} \cup \{0\}}{argmin} \left\| \Theta(p(t+\eta), \zeta(t)) \right\| \tag{39}$$

Furthermore, the approximately constant average delay $\eta^\star$ over the interval $I$ can be expressed as:

$$\eta^\star = \underset{\eta \in \mathbb{N} \cup \{0\}}{argmin} \left\| \Theta(p(t+\eta), \zeta(t)) \right\| \tag{40}$$

Since the delay may vary slowly, the computed value of $\eta^\star$ may vary from one observation interval to another. For each interval $I_k \in \{I_1, \cdots, I_M\}$, one can obtain the approximate probability distribution of the delay corrected deviation $\Delta(t)$, which is denoted as $\mathbb{D}^{[k]}$. Therefore, from a computational point of view, $\mathbb{D}^{[k]}$ is just a histogram constructed from the $\Delta(t)$ values for the interval $I_k$ (for a set of appropriately chosen histogram bins or intervals). For a sufficiently large number of observation intervals $\{I_1, \cdots, I_M\}$, one can capture the deviation dynamics of the robotic platform by computing the expected distribution of $\Delta(t)$, *i.e.* computing $\mathbb{D}$, which can be estimated simply by:

$$\mathbb{D} = \frac{1}{M} \sum_{k=1}^{M} \mathbb{D}^{[k]} \tag{41}$$

Once the distribution for the delay corrected deviation has been computed, we can proceed to estimate the probabilities of the uncontrollable transitions, as described before. Determination of the uncertainty parameters in the navigation model then allows us to use the proposed optimization to compute optimal plans which the robot can execute. We summarize the sequential steps in the next section.

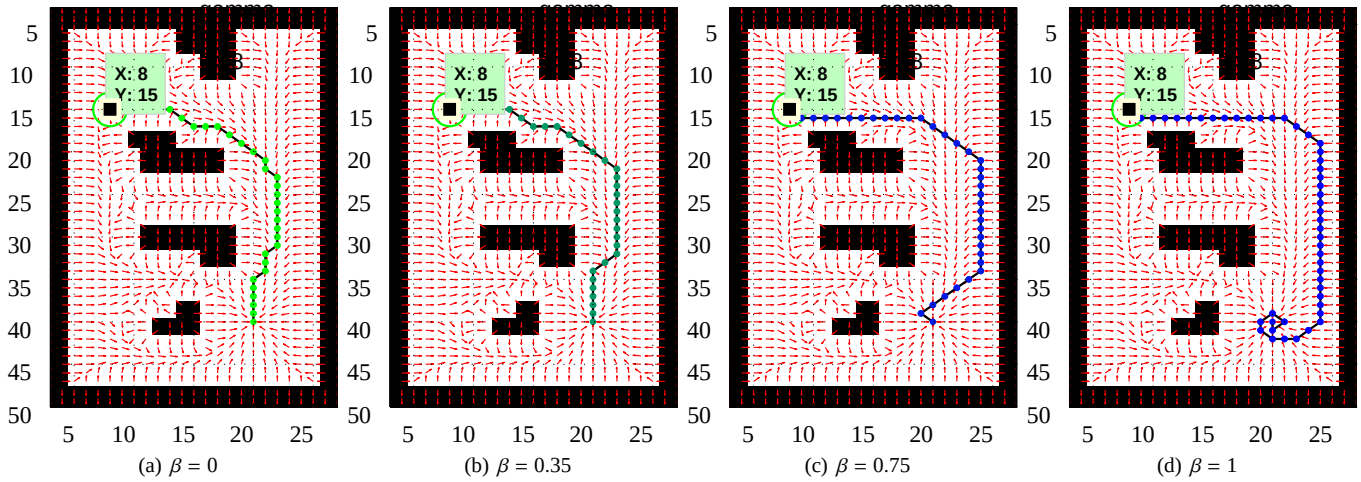
## 6. Summarizing $\nu^\star$ Planning & Subsequent Execution

The complete approach is summarized in Algorithm 5. The planning and plan assembly steps (Lines 2 & 3) are to be done either offline or from time to time when deemed necessary in the course of mission execution. Replanning may be necessary if the dynamic model changes either due to change in the environment or due to variation in the operational parameters of the robot itself, *e.g.*, unforeseen faults that alter or restrict the kinematic or dynamic degrees of freedom. Onwards from Line 4 in Algorithm 5 is the set of instructions needed for mission execution. Line 5 computes the set of states to which the robot can possibly move from the current state. We select one state from the set of possible next states which have a strictly higher measure compared to the current state in the computed plan. It is possible that the set of such states $Q_{next}$ (See Line 6) has more than one entry. Choice of any element in $Q_{next}$ as the next desired state is optimal in the sense of Proposition 5 . However, one may use some additional criteria to aid this choice to yield plans suited to the application at hand, *e.g.*, to minimize change of current direction of travel. For example one may choose the state from $Q_{next}$ that requires least deviation from the current direction of movement, to minimize control effort. In general, we can penalize turning using a specified penalty $\beta \in [0, 1]$ as follows:

**Definition 23.** *Given a turning penalty $\beta \in [0, 1]$, the turn penalized measure values on the set of possible next states $Q_{next}$ is computed as follows:*

$$\forall q \in Q_{next}, \nu^\beta(q) = (1-\beta)\nu_\#^{\mathbf{A}}(q) + \beta \cos(h(q)) \tag{42}$$

*where $h(q)$ is the heading correction required for transitioning to $q \in Q_{next}$, which for 2D circular robots is calculated as the*




(a) $\beta = 0$ (b) $\beta = 0.35$ (c) $\beta = 0.75$ (d) $\beta = 1$

Figure 12: Effect of increasing turning penalty $\beta$ on making turns in choice of local transitions post $\boldsymbol{\nu}^\star$-optimization. Note that the *potential gradient* is identical in all four cases. The start state is $(8, 15)$ as shown. The gradient shown (by the arrows) is for the computed measure vector ${\boldsymbol{\nu}_\#}^{\mathbf{A}}$, and hence is identical for all four cases

*angular correction between the line joining the center of the current state to that of q, and the line joining the center of the last state with that of the current one. the direction of the last state. The robot then chooses $q_{next}$ as the state $q \in Q_{next}$ which has the maximum value for $\nu^\beta(q)$.*

The effect of penalizing turns is shown in the Figure 12. Note that for maximum turn penalty, the computed plan is almost completely free from kinks. Also, note that the $\boldsymbol{\nu}^\star$ optimization ensures that all these plans have the same probability of success and collision.

As stated in Line 8, the robot may not be successful to actually transition to the next chosen state due to dynamic effects. In particular, if the state that the robot actually ends up in has a non-positive measure (due to uncertainties and execution errors), then execution is terminated and the mission is declared unfeasible from that point (See Line 14).

It is important to note that if a particular configuration maps to a navigation state with non-positive measure, then no feasible path to goal exists from that configuration, *irrespective of uncertainty effects*. This underscores the property of the proposed algorithm that it finds optimal feasible paths; even if the only feasible path is very unsafe, it still is the *only* feasible path; and is therefore the optimal course of action (See Proposition 7 Statement 3).

**Algorithm 5**: Summarized Planning & Mission Execution

```
   input : Model G_NAV^MOD
1  begin
       /* ⟵ Planning & Plan Assembly ⟶                          */;
2      Compute decomposed plans ν_#^[k] ; /* Algorithm 3        */
3      Compute assembled plan ν_#^A ; /* Algorithm 4            */
       /* ⟵ Mission Execution ⟶                                 */;
4      while true do
5          Find set of neighbors: N(q_i) = {q ∈ Q : ∃σ ∈ Σ_C s.t. q_i --σ--> q};
6          Compute Q_next = {q_j ∈ N(q_i) : ∀q_k ∈ N(q_i), ν_#^A|_j > ν_#^A|_k};
7          Choose one state q_next from set Q_next;
8          Attempt to move to q_next; /* May be unsuccessful
           */
9          Read current state q_i; /* Possibly q_i ≠ q_next      */
10         if q_i == q_GOAL then
11             Mission Successful Terminate Loop;
12         else
13             if ν_#^A|_i ≦ 0 then
14                 Mission Failed Terminate Loop;
15             endif
16         endif
17     endw
18 end
```

## 7. Verification & Validation

In this section we validate the proposed planning algorithm via detailed high fidelity simulation studies and in experimental runs on a heavily instrumented SEGWAY RMP 200 at the robotic testbed at the Networked Robotics & Systems Laboratory (NRSL), Pennstate. The results of these experiments adequately verify the theoretical formulations and the key claims made in the preceding sections.

**Remark 6.** *In depicting simulation results in the sequel, we refer to "computed paths/plans". It is important to clarify, what we mean by such a computed or simulated path. The computed path is the sequence of configuration states that the robot would enter, if the uncertainties do not force it to deviate, i.e., the path depicts the best case scenario under the uncertainty model. Thus, the depictions merely give us a feel for the kind of paths the robot would take; in actual implementation, the trajectories would differ between runs. Also, when we refer to lengths of the computed paths, we are referring to the lengths of the paths in the best case scenario, i.e., the tight lower bound on the path length that will actually be encountered.*

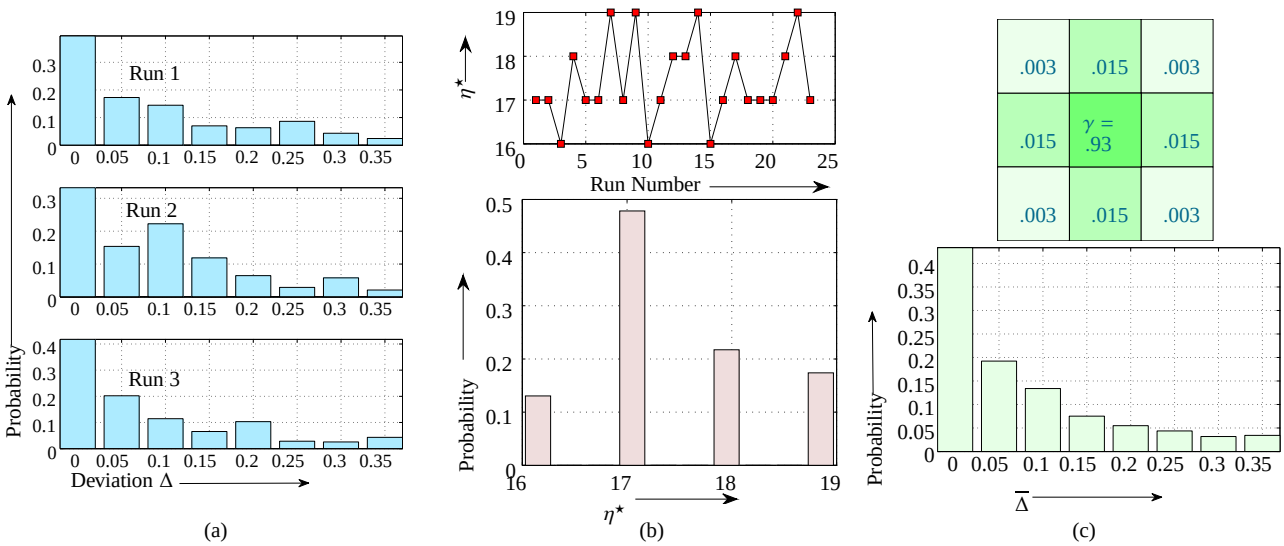



Figure 13: Computation of dynamic uncertainty parameters from observed dynamics for SEGWAY RMP 200 at NRSL, Pennstate: (a) shows the observed distribution $\mathbb{D}$ of the deviation $\Delta$ for different runs, (b) shows the distribution of the delay $\eta^\star$ for the various runs, Lower plate in (c) illustrates the expected distribution $\mathbf{E}(\mathbb{D})$ for deviation $\Delta$ while the upper plate in (c) enumerates the probabilities of the uncontrollable transitions to the neighboring cells, and the coefficient of dynamic deviation $\gamma(\mathbb{G}^{MOD}_{\mathsf{NAV}})$

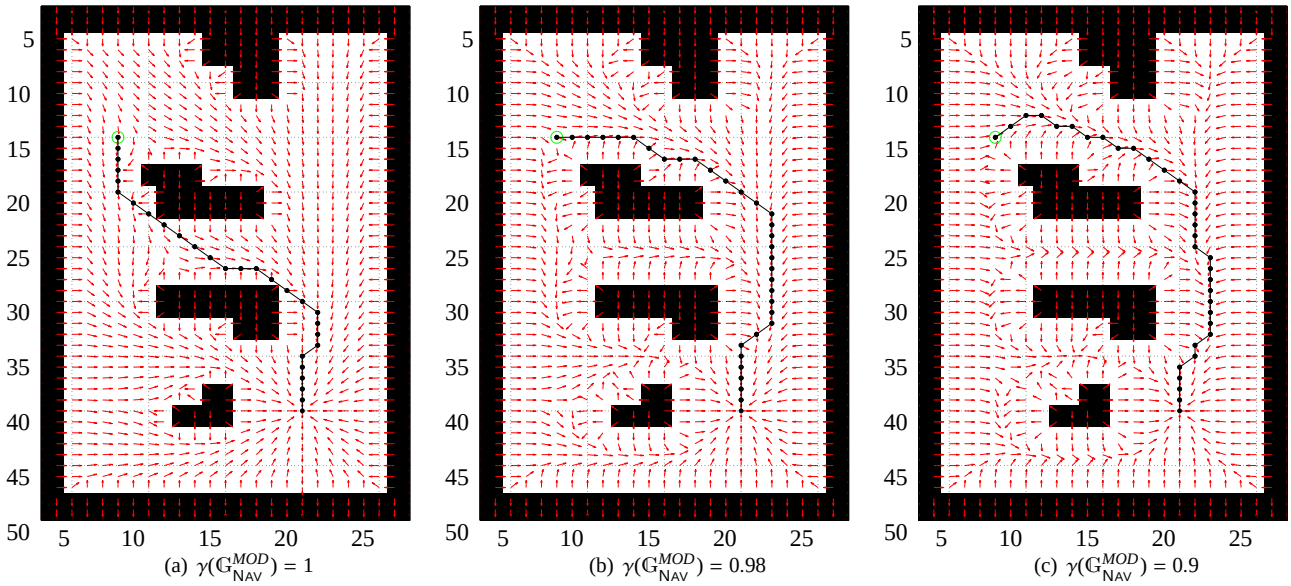



Figure 14: Simulation results with a circular robot model and different values of the coefficient of dynamic deviation $\gamma(\mathbb{G}^{MOD}_{\mathsf{NAV}})$. Note the response of the navigation gradient as $\gamma(\mathbb{G}^{MOD}_{\mathsf{NAV}})$ is decreased.

### *7.1. Simulation Results for Circular Robots*

The recursive version of the modified $\nu^\star$ planning algorithm presented in this paper (See Algorithm 5) is first validated in a detailed simulation example as illustrated in Figures 13(a-c) and 14(a-c). The workspace is chosen to be a $50 \times 30$ grid, with obstacles placed at shaded locations, as illustrated. The size of the workspace is chosen to correspond with the size of the actual test-bed, where experimental runs on the robotic platform would be performed subsequently. Plates (a)-(c) in Figure 14 illustrates the gradient of the optimized measure vector (by short arrows) and a sample optimal path from location $(15, 10)$ (upper,left) to the goal $(40, 20)$ (down,right). We note that the "potential field" defined by the measure gradient converges (*i.e.* has an unique sink) at the goal. Also, note that the coefficient of dynamic deviation $\gamma(\mathbb{G}^{MOD}_{\mathsf{NAV}})$ is decreased, the algorithm responds by altering the optimal route to the goal. In particular, the optimal path for smaller values of $\gamma(\mathbb{G}^{MOD}_{\mathsf{NAV}})$ stay further away from the obstacles. The key point to note here is that the the proposed algorithm guarantees that this lengthening of the route to account for dynamic uncertainty is *optimal*, *i.e.*, further lengthening by staying even further from the obstacles yields no advantage in a probabilistic sense. This point has a direct practical

implication; one that can be verified experimentally as follows. Let us assume that we have a real-world robot equipped with on-board reactive collision avoidance, by which we can ensure that the platform does not *actually collide* with obstacles under dynamic uncertainty, but executes corrections dictated by local reactive avoidance. Then, the preceding result would imply that a using the correct value of dynamic deviation (for the specific platform) in the planning algorithm would result in routes that require the least number of local corrections; which in turn ensures minimum time route traversals on the average.

It is important to note that the assumption of a circular robot poses no critical restrictions. Similar results can be obtained for more complex models as well, *i.e.* rectangular platform with constrained turn radius. However, extension to multi-body motion planning would require addressing the algorithmic complexity issues that become important even for the recursive $\nu^\star$ for very large configuration spaces, and is a topic of future work.

### *7.2. Simulation Results for Non-symmetric Uncertainty*

As stated in the course of the theoretical development, it is possible to choose the degree of amortization or averaging that one is willing to allow in the specification of the navigation automaton. As a specific example, one may choose to compute the probabilities of uncontrollable transitions with respect to some length of trajectory history; the simplest case is using the previous state information to yield non-symmetric deviation contours (See Figure 15). The particular type of deviation contours illustrated in Figure 15 is obtained if the platform has a large stopping distance and inertia, and the heading and positional estimates are more or less accurate, *i.e.*, the uncontrollability in the model is a stronger function of the dynamic response, rather than the estimation errors. A typical scenario is the SEGWAY RMP 200 with good global positioning capability, in which factoring in the dynamic response is important due to the inverted-pendulum two-wheel kinematics. For this simulation, we use a navigation automaton obtained from discretizing an essentially 4D underlying continuous configuration space. Each state (except the obstacle state) in the navigation automaton maps to a discretized pair of locations, reflecting the current robot location and the one from which it transitioned to the current location. We call this the *4D Model* to distinguish it from the the significantly smaller and simpler 2D model. Note that the 2D model can be obtained from the 4D model by merging states with the same current location via averaging the probabilities of uncontrollable transitions over all possible previous states. Also note that (as stated earlier), in the absence of uncertainty, the 4D formulation adds nothing new; explicitly encoding the previous location in the automaton state gives us no new information. Table 1 enumerates the comparative model sizes.

Table 1: Comparison of 4D and 2D Models

| | Map Size | No. of States | Alphabet Size |
|---|---|---|---|
| 2D Model | $40 \times 40$ | 1600 | 8 |
| 4D Model | $40 \times 40$ | $256 \times 10^4$ | 8 |

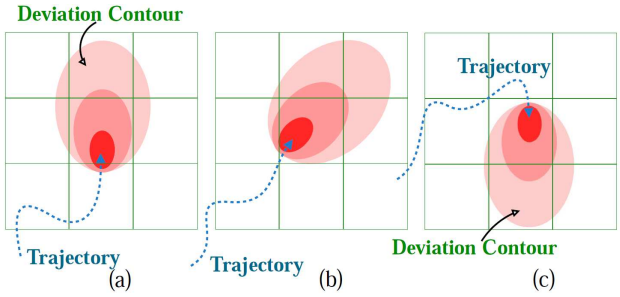



Figure 15: Non-symmetric deviation contours arising from explicit dependence on the last discrete position for the robot

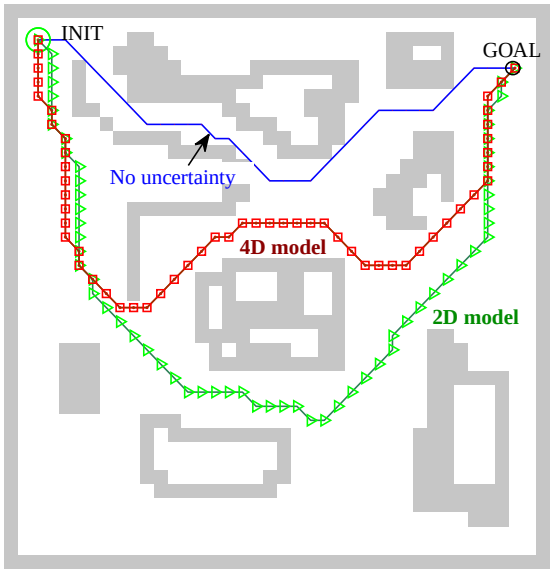



Figure 16: Comparative pathlengths for the chosen initial and goal configurations: 65 steps for 2D model, 51 steps for 4D model, 36 steps for the case with no uncertainty

Comparisons of computed plans for a particular set of initial and goal configurations is shown in Figure 16. To accentuate the differences in the computed plans, the deviation contours were chosen so that the probability of uncontrollable transition in the current direction of travel is significantly more compared to that of deviating to left or right, *i.e.*, the contours are really narrow ellipses. Under such a scenario, the platform is more capable of navigating narrow corridors as compared to the amortized 2D counterpart. This is reflected in the paths shown in Figure 16, where the path for the 4D model is shorter, and goes through some of the narrow bottlenecks, while the path for the 2D model takes a safer path. Note, the path for the no-uncertainty case is even more aggressive, and shorter. In practical implementation, when the uncontrollable probabilities are identified from observed dynamics or pre-existing continuous models, the differences in the two cases are often significantly less.

### *7.3. Simulation Results for Rectangular Robots*

The proposed planning algorithm is next applied to the case of rectangular robots, specifically ones that have a minimum non-zero turn radius. We further impose the constraint that the

platform cannot travel backwards, which is a good assumption for robots that have no ranging devices in the rear, and also for aerial vehicles (UAVs). Even assuming planar operation, this problem is essentially 3D, with the navigation automaton reflecting the underlying configuration states of the form $(x, y, h)$ where $h$ is the current heading, which can no longer be neglected due to the inability of the patform to turn in place. A visual comparison of the models for the circular and rectangular cases is shown in Figure 17(e). The heading is discretized at 15° intervals, implying we have 24 discrete headings. This also means that for the same planar workspace, the number of states in the rectangular model is about 24 times larger the number of states for the circular model. Also, while in the circular case, we had 8 neighbors, the number of neighboring configurations increases to $8 \times 24 + 24 = 216$ However, not all neighbors can be reached via controllable transitions due to the restriction on the turn radius; we assume a maximum turn of ±45° in the model considered for the simulation. As explained earlier, all the neighbors may be reachable via uncontrollable transitions, which reflects uncertainties in estimation (See Figure 10).

We test the algorithm with different values of $\gamma(\mathbb{G}_{\mathsf{NAV}}^{MOD})$ as illustrated in Figure 17(a-d). Note the trajectories become more rounded and less aggressive (as expected) as the uncertainty is increased. Also note that the heading at the goal is different for the different cases. This is because, in the model , we specified as goals any state that maps to the goal location in the planar grid irrespective of the heading, *i.e.*, the problem was solved with essentially 24 goals. Although for simplicity, the theoretical development was presented assuming a single goal, the results can be trivially shown to extend to such scenarios. The trajectories however, will be significantly different if we insist on having a particular heading at the goal (See Figure 18).

In the simulations for the rectangular model, we deliberately assume that any neighbor that cannot be reached via a controlled move is not reachable by an uncontrollable transition as well. Although this is not what we expect to encounter in field, the purpose of this assumption is to bring out an interesting consequence that we illustrate in Figures 19(a-b). As explained above, this assumption implies that we have little or no uncertainty in local heading estimations (since the robot cannot turn in place, so there is no uncontrollable transition that alters heading in place). It therefore follows, that under this scenario, the platform would find it relatively safe to navigate narrow passages. This is exactly what we see in Figure 19(b), where the circular robot with same coefficient of dynamic uncertainty, really goes out of way to avoid the narrow passage, while the rectangular robot goes through.

### *7.4. Simulation Results for Mazes*

We simulate planning in a maze of randomly placed static obstacles. A sample case with optimal paths computed for different coefficients of dynamic deviation is illustrated in Figure 20(a). A key point to note is that the optimal path is lengthens first as $\gamma(\mathbb{G}_{\mathsf{NAV}}^{MOD})$ is decreased, and then starts shortening again, which may seem paradoxical at first sight. However, this is exactly what we expect. Recall that the proposed algorithm minimizes the probability of collision. Also note that there are two opposing effects in play here; while a longer path that stays away from the obstacles influences to decrease the collision probability, the very fact that the path is longer has an increasing influence arising from the increased probability that an uncontrollable sequence would execute from some point in the path that leads to a collision. At relatively high values of $\gamma(\mathbb{G}_{\mathsf{NAV}}^{MOD})$, the first effect dominates and we can effectively decrease the collision probability by staying away from the obstacles thereby increasing the path length. However, at low values of $\gamma(\mathbb{G}_{\mathsf{NAV}}^{MOD})$, the latter effect dominates, implying that increased path lengths are no longer advantageous. This interesting phenomenon is illustrated in Figure 20(b), where we clearly see that the path lengths peak in the $\gamma(\mathbb{G}_{\mathsf{NAV}}^{MOD}) = 0.72$ to $\gamma(\mathbb{G}_{\mathsf{NAV}}^{MOD}) = 0.85$ range (for the maze considered in Figure 20(a)). Also note that the configuration space has to be sufficiently complex to actually see this effect; which is why we do not see this phenomenon in the simulation results presented in Figures 14(a-c).

### *7.5. Experimental Runs on SEGWAY RMP 200*

The proposed algorithm is validated on a SEGWAY RMP 200 which is a two-wheeled robot with significant dynamic uncertainty. In particular, the inverted-pendulum dynamics prevents the platform from halting instantaneously, and making sharp turns at higher velocities. At low velocities, however, the platform can make zero radius turns. The global positional fix is provided via an (in-house developed) over-head multi-camera vision system, which identifies the position and orientation of the robot in the testbed. The vision system yields a positional accuracy of ±7.5 cm, and a heading accuracy or ±0.1 rad for a stationary robot. The accuracy deteriorates significantly for a mobile target, but noise correction is intentionally not applied to simulate a high noise uncertain work environment. Furthermore, the cameras communicate over a shared wireless network and randomly suffers from communication delays from time to time, leading to delayed positional updates to the platform. In the experimental runs conducted at NRSL the workspace discretized into a $53 \times 29$ grid. Each grid location is about 4 sq. ft. allowing the SEGWAY to fit completely inside each such discretized positional state which justifies the simplified *circular robot* modeling. The runs are illustrated in Figure 22. The robot was run at various allowed top speeds ($v_{max}$) ranging from 0.5 mph to over 2.25 mph. Only the extreme cases are illustrated in the figure. For each speed, the uncertainty parameters were estimated using the formulation presented in Section 5. The sequence of computational steps for the low velocity case ($v_{max} = 0.5$ mph) are shown in Figure 13. Note the coefficient of dynamic deviation for the low velocity case turns out to be $\gamma^{low} = 0.973$. For the high velocity case, ($v_{max} = 2.25$ mph), the coefficient is computed to have a value of $\gamma^{high} = 0.93$ (calculation not shown). Also, the robot is equipped with an on-board low-level reactive collision avoidance algorithm, which ensures that the platform does not *actually* collide due path deviations; but executes local reactive corrections when faced with such situations. The platform is equipped with multiple high frequency sonars, infra-red range finders and a high-precision SICK LMS

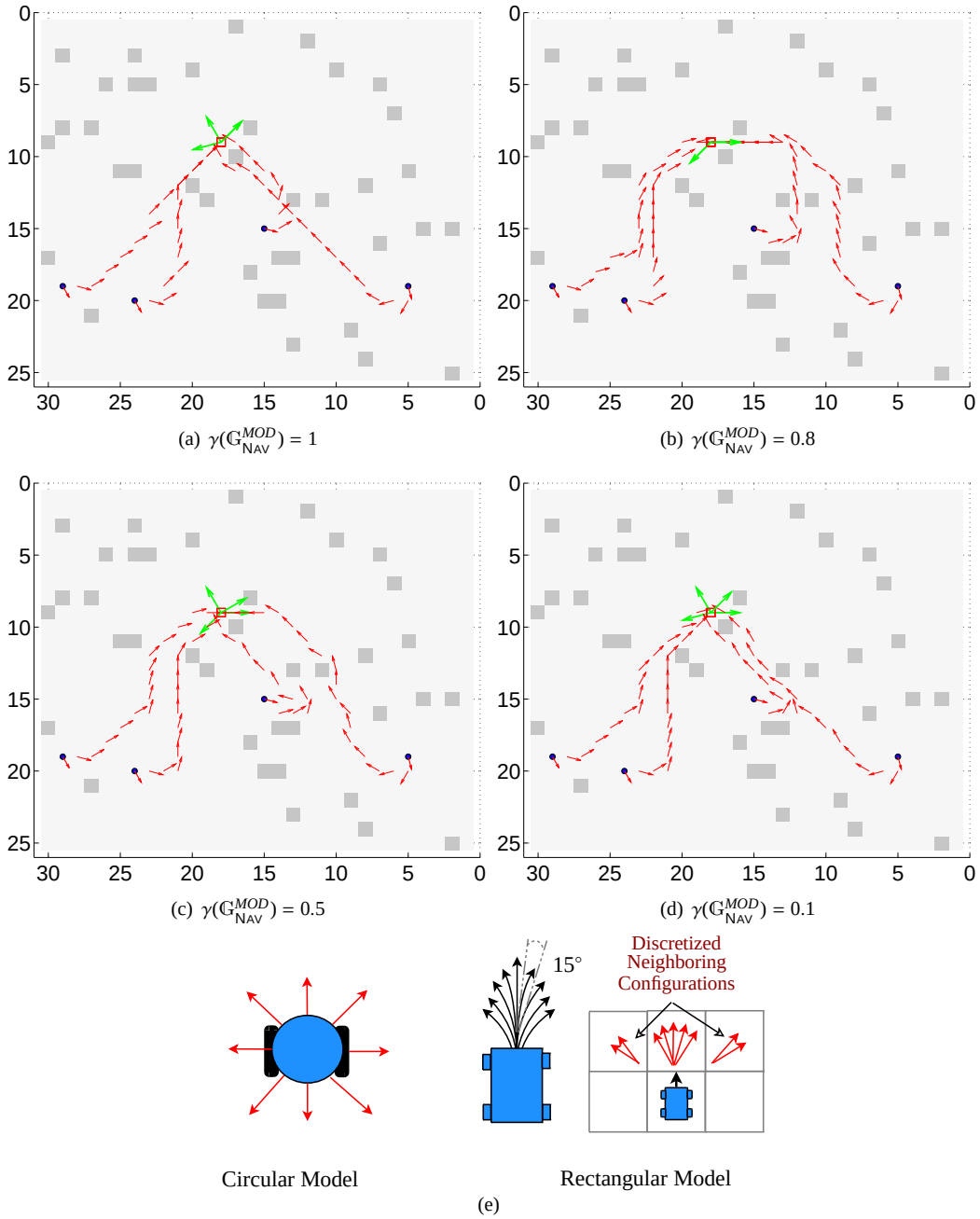
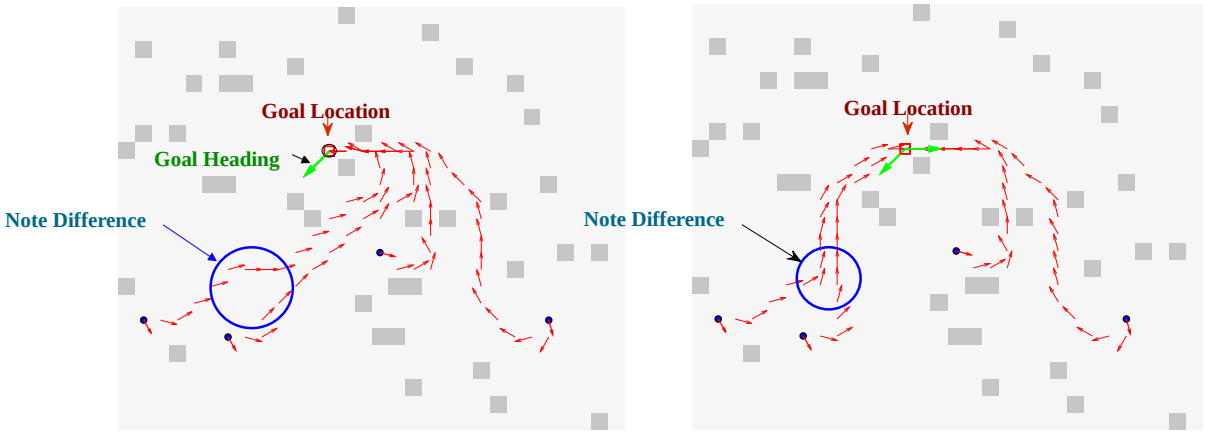



Figure 17: Comparative trajectories for rectangular model

200 laser range finder. The data from these multiple ranging devices, mounted at various key locations on the platform, must be fused to obtain correct situational awareness. In this paper, we skip the details of this on-board information processing for the sake of brevity. The overall scheme is illustrated in Figure 21

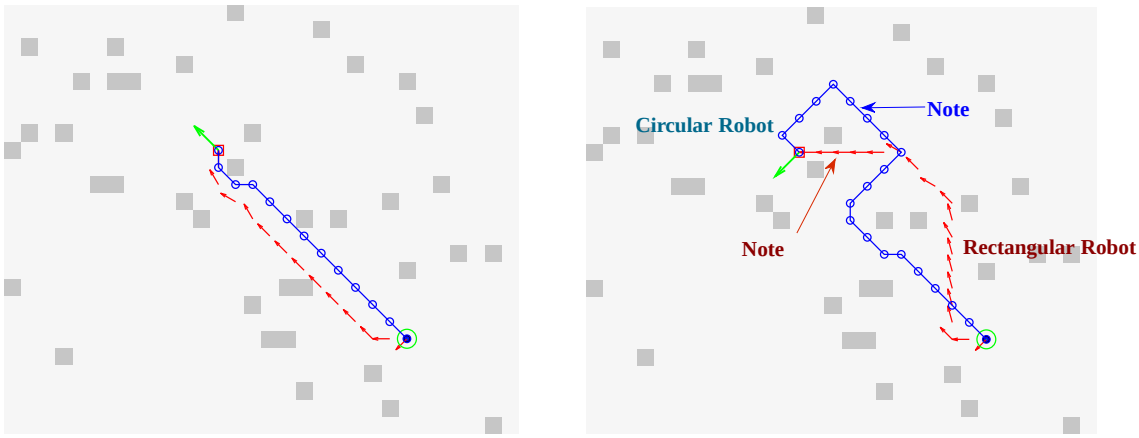
In the experimental runs, we choose two waypoints (marked *A* and *B*) in Figure 22 (plates a,b,d,e), and the mission is to plan and execute the optimal routes in sequence from *A* to *B*, back to *A* and repeat the sequence a specified number of times (thirty). This particular mission is executed for each top speed for a range of $\gamma$ values, namely with $\gamma \in [0.75, 1]$ with increments of 0.1. The expectation is that using the correct coefficient of dynamic deviation (as computed above for the two chosen speeds) for the given speed, would result in the minimum number of local corrections, leading to minimum average traversal times over thirty laps.

The results are summarized in plates (c) (for the low velocity case) and (f) (for the high velocity case) in Figure 22. Note the fitted curve in both cases attain the minimal point very close to the corresponding computed $\gamma$ values, namely, 0.97 for $v_{max} = 0.5$ mph and 0.92 for $v_{max} = 2.25$ mph. A visual comparison of the trajectories in the plates (a) and (d) clearly reveal that the path execution has significantly more uncertainties




(a) Strict Goal Heading Requirement (b) No Heading Requirement

Figure 18: Comparative trajectories for rectangular model for $\gamma(\mathbb{G}_{\text{NAV}}^{MOD}) = 0.8$: Case (a) we demand that the robot reach the goal with a specified heading ($-150°$). Case (b): Any heading at the goal is acceptable




(a) $\gamma(\mathbb{G}_{\text{NAV}}^{MOD}) = 1$ (b) $\gamma(\mathbb{G}_{\text{NAV}}^{MOD}) = 0.8$

Figure 19: Comparative trajectories for rectangular model and circular model to illustrate the effect of different uncertainty assumptions: (a) Trajectories in the absence of uncertainty (b) Trajectories with $\gamma(\mathbb{G}_{\text{NAV}}^{MOD}) = 0.8$.

in the high velocity case. Also note, that the higher average speed leads to repeated loss of position fix information in locations around ($row = 20, column = 35$). Plates (b) and (e) illustrate the sequence of waypoints invoked by the robot in the two cases, being the centers of the states in the navigation automaton that the robot visits during mission execution. Note that in the high velocity case, the variance of the trajectories is higher leading to a larger set of waypoints been invoked. Note that three distinct zones (denoted as Zone A, Zone B and Zone C) can be identified in the plates (e) and (f) of Figure 22. Zone A reflects the operation when $\gamma$ is (incorrectly assumed to be) too large, leading to too many corrections, and hence execution time can be reduced by reducing $\gamma$. In Zone B, reducing $\gamma$ increases execution time, since now the trajectories becomes unnecessarily safe, *i.e.* stays away from obstacles way more than necessary leading to longer than required paths and hence increased execution time. Zone C represents a sort of saturation zone where reducing $\gamma$ has no significant effect, arising from the fact that the paths cannot be made arbitrarily safe by increasing path lengths. Although the experimental runs were not done for smaller values of $\gamma$, we can say from the experience with maze simulations (See Section 7.4), that the execution times will start reducing again as $\gamma$ is further reduced.

These results clearly show that the approach presented in this paper successfully integrates amortized dynamical uncertainty with autonomous planning, and establishes a computationally efficient framework to cyber-physical motion planning.

## 8. Summary & Future Research

The recently proposed PFSA-based path planning algorithm $\nu^\star$ is generalized to handle amortized dynamic uncertainties in plan execution, arising from the physical limitations of sensing and actuation, and the inherent dynamic response of the physical platforms. The key to this generalization is the introduction of uncontrollable transitions in the modified navigation automaton, and showing that $\nu^\star$ can be implemented in a recursive fashion to guarantee plan optimality under such circumstances.

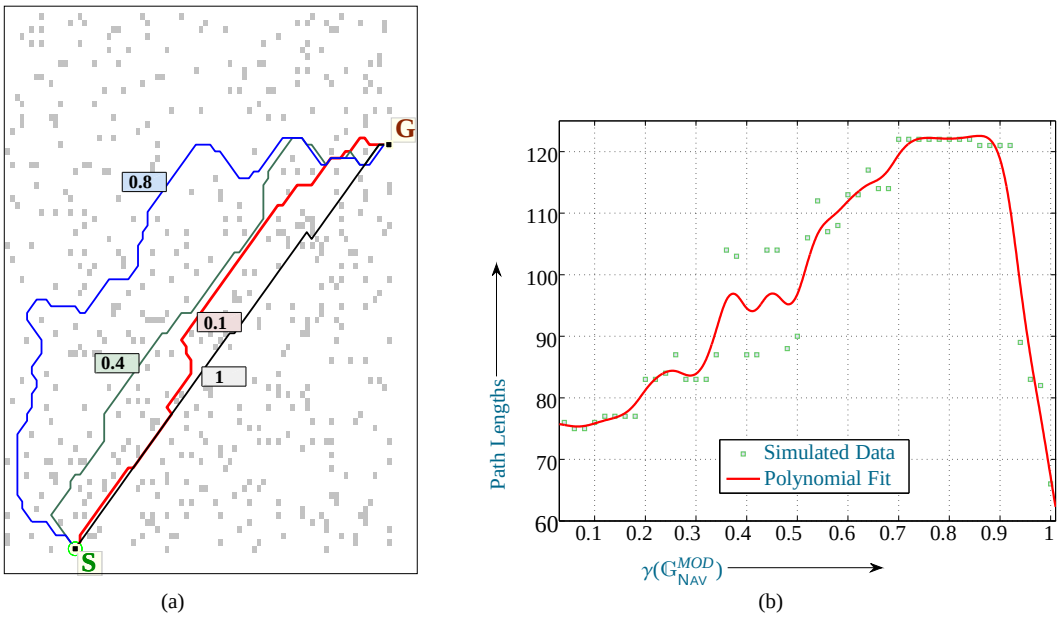



Figure 20: Effect of dynamic uncertainty on the optimal path lengths computed by $\nu^\star$. Plate (a) illustrates the optimal paths for $\gamma(\mathbb{G}^{MOD}_{\text{NAV}}) = 1.0, 0.8, 0.4, 0.1$ from the start location marked by $S$ and the goal marked by $G$. Plate (b) illustrates the variation of the length of the optimal paths as a function of $\gamma(\mathbb{G}^{MOD}_{\text{NAV}})$ for the maze illustrated in (a).

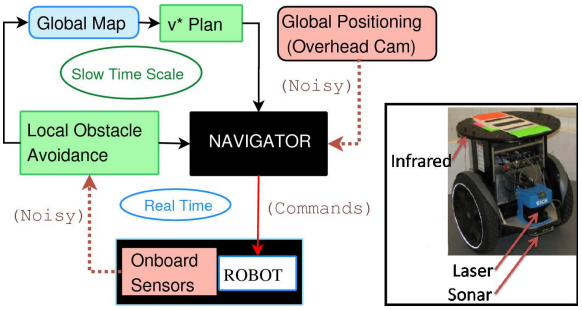



Figure 21: Autonomous navigation scheme with $\nu^\star$ implementation on heavily instrumented Segway RMP (shown in inset).

The theoretical algorithmic results is verified in detailed high-fidelity simulations and subsequently validated in experimental runs on the SEGWAY RMP 200 at NRSL, Pennstate.

### *8.1. Future Work*

Future work will extend the language-measure theoretic planning algorithm to address the following problems:

1. **Multi-robot coordinated planning:** Run-time complexity grows exponentially with the number of agents if one attempts to solve the full Cartesian product problem. However $\nu^\star$ can be potentially used to plan individually followed by an intelligent assembly of the plans to take interaction into account.
2. **Hierarchical implementation to handle very large workspaces:** Large workspaces can be solved more efficiently if planning is done when needed rather than solving the whole problem at once; however care must be taken to ensure that the computed solution is not too far from the optimal one. One the areas of current research is an algorithmic decomposition of the configuration space such that individual blocks are solved in parallel on communicating processors, with the interprocessor communication ensuring close-to-global optimality. We envision such an approach to be ideaally suited to scenarios involving multiple agents distributed over a large workspace which cooperatively solve the global planning problem in an efficient resource-constrained manner.
3. **Handling partially observable dynamic events:** In this paper all uncontrollable transitions are assumed to be perfectly observable. Physical errors and onboard sensor failures may need to be modeled as unobservable transitions and will be addressed in future publications. A generalization of the measure-theoretic optimization technique under partial observation has been already reported [48]. The future goal in this direction is to incorporate the modifications to allow $\nu^\star$ handle loss of observation and feedback information.

## References


[1] J.-C. Latombe, Robot Motion Planning, International Series in Engineering and Computer Science; Robotics: Vision, Manipulation and Sensors, Kluwer Academic Publishers, Boston, MA, U.S.A., 1991, 651 pages.

[2] S. M. LaValle, Planning Algorithms, Cambridge University Press, Cambridge, U.K., 2006, available at http://planning.cs.uiuc.edu/.

[3] K. Kondo, Motion planning with six degrees of freedom by multistrategicbidirectional heuristic free-space enumeration, IEEE Transactions on Robotics and Automation 7 (3) (1991) 267–277.

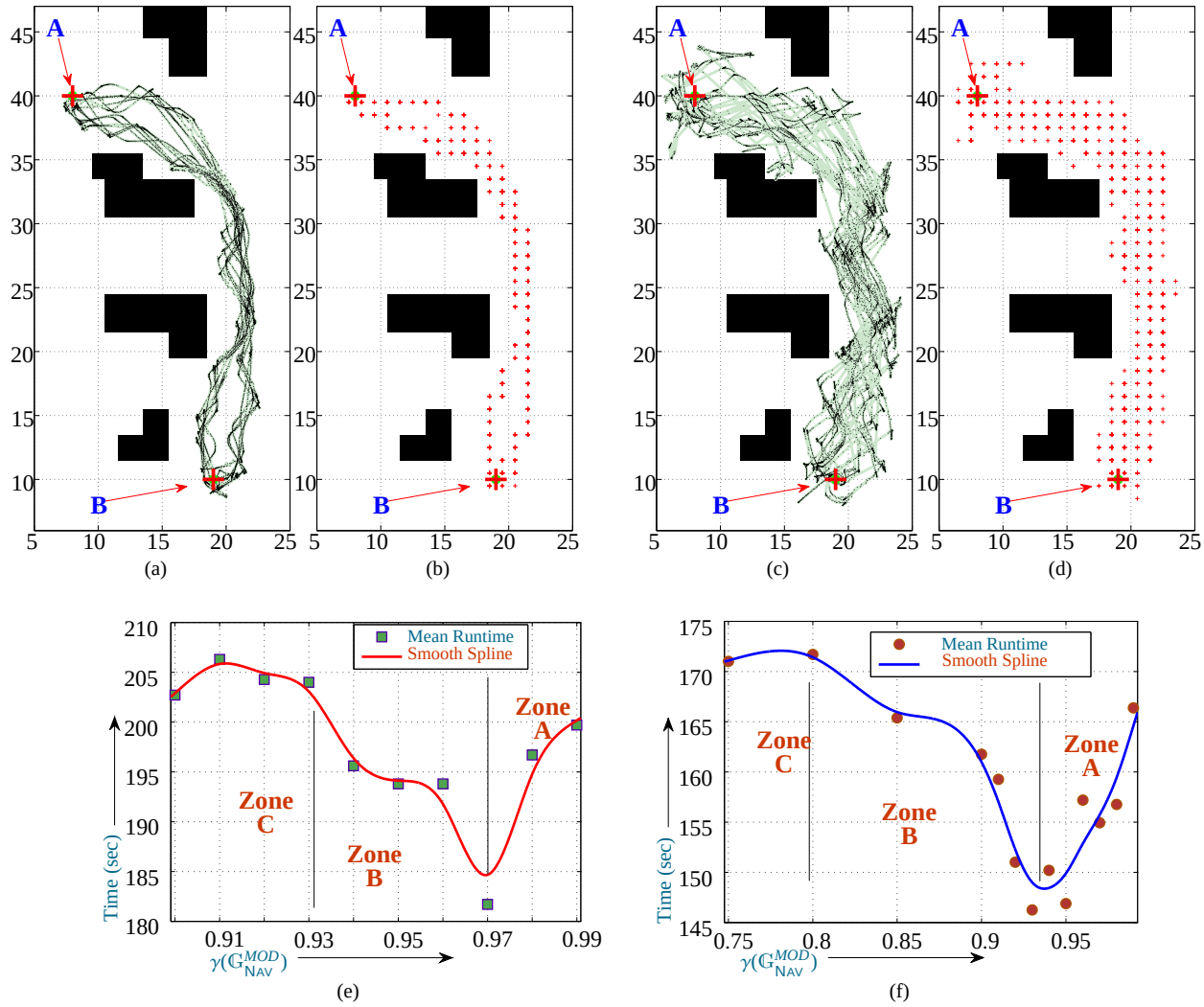



Figure 22: Experimental runs on SEGWAY RMP 200: (a)-(c) Low speed runs and (d)-(f) High speed runs: Plates (a) and (d) shows the trace of the robot positions as read by the overhead vision system at NRSL for the low and high speed runs respectively. (b) and (e) shows the waypoints invoked by the robot in course of executing the specified mission in the low and high speed cases respectively. Plates (c) and (f) illustrate the variation of the mean mission execution times with the coefficient of dynamic deviation used fro planning in the low and high speed cases respectively.


[4] J. Borenstein, Y. Koren, The vector field histogram-fast obstacle avoidance for mobile robots, Robotics and Automation, IEEE Transactions on 7 (3) (2002) 278–288. doi:10.1109/70.88137.
URL http://dx.doi.org/10.1109/70.88137

[5] T. Lozano-Perez, A simple motion-planning algorithm for general robot manipulators, IEEE Transactions on Robotics and Automation 3 (3) (1987) 224–238.

[6] D. A. Anisi, J. Hamberg, X. Hu, Nearly time-optimal paths for a ground vehicle, Journal of Control Theory and Applications.

[7] J. Barraquand, B. Langlois, J.-C. Latombe, Robot motion planning with many degrees of freedom and dynamic constraints, MIT Press, Cambridge, MA, USA, 1990.

[8] J. Langelaan, Tree-based trajectory planning to exploit atmospheric energy, in: American Control Conference, 2008, 2008, pp. 2328–2333. doi:10.1109/ACC.2008.4586839.

[9] S. Lahouar, E. Ottaviano, S. Zeghoul, L. Romdhane, M. Ceccarelli, Collision free path-planning for cable-driven parallel robots, Robotics and Autonomous Systems 57 (11) (2009) 1083 – 1093. doi:DOI:10.1016/j.robot.2009.07.006.

[10] L. M. Ortega, A. J. Rueda, F. R. Feito, A solution to the path planning problem using angle preprocessing, Robotics and Autonomous Systems In Press, Corrected Proof (2009) –. doi:DOI:10.1016/j.robot.2009.07.028.

[11] J. R. Andrews, N. Hogan, Impedance Control as a Framework for Implementing Obstacle Avoidance in a Manipulator, ASME, Boston, MA, 1983, pp. 243–251.

[12] O. Khatib, Real-time obstacle avoidance for manipulators and mobile robots, in: IEEE International Conference on Robotics and Automation, Vol. 2, St. Louis, MI, 1985, pp. 500–505.

[13] B. H. Krogh, A generalized potential field approach to obstacle avoidance control, in: International Robotics Research Conference, Bethlehem, 1984.

[14] M. Kumar, D. Garg, R. Zachery, Multiple mobile agents control via artificial potential functions and random motion, in: Proceedings of the ASME International Mechanical Engineering Congress and Exposition, ASME, Seattle, WA, 2007. doi:PaperNo.IMECE2007-41521.

[15] S. Sarkar, E. Halland, M. Kumar, Mobile robot path planning using support vector machines, in: ASME Dynamic Systems and Control Conference, ASME, Ann Arbor, Michigan, 2008. doi:PaperNo.DSCC2008-2200.

[16] J. Borenstein, Y. Koren, Potential field methods and their inherent limitations for mobile robot navigation, in: Proceedings of the 1991 IEEE International Conference on Robotics and Automation, 1991, pp. 1398–1404.

[17] R. Tilove, Local obstacle avoidance for mobile robots based on the method of artificial potentials, Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on (1990) 566–571 vol.1doi:10.1109/ROBOT.1990.126041.

[18] I. Chattopadhyay, G. Mallapragada, A. Ray, $\nu^\star$ : a robot path planning algorithm based on renormalized measure of probabilistic regular languages, International Journal of Control 82 (5) (2008) 849–867.

[19] I. Chattopadhyay, A. Ray, Renormalized measure of regular languages, Int. J. Control 79 (9) (2006) 1107–1117.

[20] I. Chattopadhyay, A. Ray, Language-measure-theoretic optimal control of probabilistic finite-state systems, Int. J. Control.

[21] J. M. O'kane, B. Tovar, P. Cheng, S. M. Lavalle, Algorithms for planning under uncertainty in prediction and sensing, in: Chapter 18 in Autonomous Mobile Robots: Sensing, Control, Decision-Making, and Applications, Marcel Dekker, 2005, pp. 501–547.

[22] T. Lozano-Perez, M. T. Mason, R. H. Taylor, Automatic Synthesis of Fine-Motion Strategies for Robots, The International Journal of Robotics Research 3 (1) (1984) 3–24. doi:10.1177/027836498400300101.

[23] A. Lazanas, J. Latombe, Landmark-based robot navigation, Vol. 92, AAAI Press, San Jose, California, 1992, pp. 816–822.

[24] T. Fraichard, R. Mermond, Path planning with uncertainty for car- like robots, in Proc. of the IEEE Intl. conf. on Robotics & Automation (1998) 27–32.

[25] N. J. Nilsson, Principles of Artificial Intelligence, Tioga, 1980.

[26] H. Takeda, J.-C. Latombe, Sensory uncertainty field for mobile robot navigation, in Proc. of the IEEE Intl. conf. on Robotics & Automation (1992) 2465–2472.

[27] P. E. Trahanias, Y. Komninos, Robot motion planning: Multi- sensory uncertainty fields enhanced with obstacle avoidance, in: Proc. of the IEEE/RSJ Intl. conf. on Intelligent Robots and Systems, 1996.

[28] N. A. Vlassis, P. Tsanakas, A sensory uncertainty field model for unknown and non-stationary mobile robot environments, in: Proceedings of the IEEE Intl. conf. on Robotics & Automation, 1998.

[29] N. Roy, S. Thrun, Coastal navigation with mobile robots, in: Advances in Neural Information Processing, Systems (NIPS, 1999.

[30] R. Alami, T. Simeon, Planning robust motion strategies for a mobile robot, in: Proc. of the IEEE Intl. conf. on Robotics & Automation, 1994.

[31] B. Bouilly, T. Simeon, R. Alami, A numerical technique for plan- ning motion strategies of a mobile robot in presence of uncertainty, in: Proc. of the IEEE Intl. conf. on Robotics & Automation, 1995.

[32] M. Khatib, B. Bouilly, T. Simeon, R. Chatila, Indoor navigation with uncertainty using sensor-based motions, in Proc. of the IEEE Intl. conf. on Robotics & Automation 4 (1997) 3379–3384.

[33] J. Barraquand, P. Ferbach, Motion planning with uncertainty: The information space approach, in: Proc. of the IEEE Intl. conf. on Robotics & Automation, 1995.

[34] L. A. Page, A. C. Sanderson, Robot motion planning for sensor-based control with uncertainties, Vol. 2, Nagoya, Japan, 1995, pp. 1333–1340.

[35] L. Blackmore, H. Li, B. Williams, A probabilistic approach to optimal robust path planning with obstacles, in: Proceedings of the AIAA Guidance, Navigation and Control ConferenceNavigation and Control Conference, 2006.

[36] L. Blackmore, A probabilistic particle control approach to optimal, robust predictive control, in: Proceedings of the AIAA Guidance, Navigation and Control ConferenceNavigation and Control Conference, 2006.

[37] A. Lambert, N. L. Fort-Piat, Safe task planning integrating uncertainties and local maps federations, International Journal of Robotics Research, volume 19 (2000) 597–611.

[38] A. Lambert, D. Gruyer, Safe path planning in an uncertain-configuration space, in: Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on, Vol. 3, 2003, pp. 4185–4190. doi:10.1109/ROBOT.2003.1242246.

[39] J. P. Gonzalez, A. T. Stentz, Planning with uncertainty in position: An optimal and efficient planner, in Proc. of the IEEE/RSJ Intl. conf. on Intelligent Robots and Systems (2005) 2435–2442.

[40] J. P. Gonzalez, A. Stentz, Planning with uncertainty in position using high-resolution maps, in: Proc, of the IEEE Intl. conf. on Robotics & Automation, Rome, Italy, 2007.

[41] R. Alterovitz, T. Siméon, K. Y. Goldberg, The stochastic motion roadmap: A sampling framework for planning with markov motion uncertainty, in: Robotics: Science and Systems, 2007.

[42] P. Singla, T. Singh, A novel coordinate transformation for obstacle avoidance and optimal trajectory planning, in: 2008 AAS/AIAA Astrodynamics Specialist Conference and Exhibit, 2008.

[43] A. Ray, Signed real measure of regular languages for discrete-event supervisory control, Int. J. Control 78 (12) (2005) 949–967.

[44] V. Garg, An algebraic approach to modeling probabilistic discrete event systems, Proceedings of 1992 IEEE Conference on Decision and Control (Tucson, AZ, December 1992) 2348–2353.

[45] V. Garg, Probabilistic lnaguages for modeling of DEDs, Proceedings of 1992 IEEE Conference on Information and Sciences (Princeton, NJ, March 1992) 198–203.

[46] W. Rudin, Real and Complex Analysis, 3rd ed., McGraw Hill, New York, 1988.

[47] I. Chattopadhyay, Quantitative control of probabilistic discrete event systems, PhD Dissertation, Dept. of Mech. Engg. Pennsylvania State University, http:// etda.libraries.psu.edu / theses / approved / WorldWideIndex / ETD-1443.

[48] I. Chattopadhyay, A. Ray, Optimal control of infinite horizon partially observable decision processes modeled as generatorsof probabilistic regular languages, International Journal of Control In Press.