**Department of Industrial Engineering and Management Sciences**

Northwestern University, Evanston, Illinois, 60208-3119, U.S.A.

***Working Paper No. 14-04***

# Parameter-free Sampled Fictitious Play for Solving Deterministic Dynamic Programming Problems

**Irina S. Dolinskaya**

Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL 60208, USA

dolira@northwestern.edu

**Marina A. Epelman**

Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, Michigan 48109, USA

mepelman@umich.edu

**Esra Sisikoglu**

Department of Industrial Engineering, Antalya International University, Dosemealti, Antalya, 07190, TURKEY

esras@antalya.edu.tr

**Robert L. Smith**

Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, Michigan 48109, USA

rlsmith@umich.edu

November 6, 2014

# Parameter-free Sampled Fictitious Play for Solving Deterministic Dynamic Programming Problems

Irina S. Dolinskaya
Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL 60208, USA, dolira@northwestern.edu

Marina A. Epelman
Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, Michigan 48109, USA, mepelman@umich.edu

Esra Şişikoğlu
Department of Industrial Engineering, Antalya International University, Dosemealti, Antalya, 07190, TURKEY, esras@antalya.edu.tr

Robert L. Smith
Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, Michigan 48109, USA, rlsmith@umich.edu

To facilitate fast solution of deterministic dynamic programming problems, we present a parameter-free variation of the Sampled Fictitious Play (SFP) algorithm. Its random tie-braking procedure imparts a natural randomness to the algorithm which prevents it from "getting stuck" at a local optimal solution and allows the discovery of an optimal path in a finite number of iterations. Furthermore, we illustrate through an application to maritime navigation that, in practice, parameter-free SFP finds a high quality solution after only a few iterations, in contrast with traditional methods.

*Key words*: Sampled Fictitious Play, Dynamic Programming, Maritime navigation

## 1. Introduction

Dynamic programming (DP) models are used to solve a large class of multi-stage decision problems that require finding an optimal sequence of decisions. Inventory control, production planning and path finding problems are just a few of the examples of applications of DP. With recent technolog-

ical advances in data collection and forecasting models, many problem settings have dynamically updating input parameters, which creates the need to re-solve the DP model in real time to fully take advantage of the new information. One such problem is our motivating application of maritime navigation in a dynamic environment. Specifically, we study the vessel path-finding problem in a non-stationary direction, location and time-dependent medium where information about the surrounding environment is updated in real time. We present a DP model that finds a path that minimizes a measure of vessel motion (roll) while precluding significant increase in travel time in search of a calmer path. One challenge of this problem is that the change in the environment and therefore the change in the input parameters cannot be analytically modeled and incorporated into a mathematical model but rather needs to be calculated via simulation. Therefore, as the information on the environment becomes available, the problem needs to be resolved in real time.

Our high fidelity model calls for a new solution method, since small computational time is critical to real time navigation in a dynamic environment. To this end, this paper presents a parameter-free Sampled Fictitious Play (SFP) algorithm that can be efficiently applied to finite-horizon deterministic DP problems with finite state and action spaces, and, more broadly, shortest path problems in acyclic networks. Parameter-free SFP eliminates the need for customizable parameters present in previous versions of the algorithm, and relies on a natural random tie-braking procedure that provides sufficient exploration within the algorithm to guarantee discovery of a globally optimal DP solution. Furthermore, we demonstrate that, in practice, parameter-free SFP finds a high quality solution after only a few iterations: numerical experiments on instances of the minimum-motion path finding problem for the S175 container ship in a time-varying nonlinear wave field demonstrate that SFP applied to the DP formulation of the problem finds solutions that reduce vessel RMS motion (roll) by up to 78% with an acceptable increase in total travel time. Finally, the SFP algorithm is easily amenable to parallelization, decreasing wall-clock run time by orders of magnitude, further facilitating real-time implementation.

### 1.1. Related Work and our Contributions

The parameter-free Sampled Fictitious Play algorithm for deterministic Dynamic Programming problems presented in this paper is rooted in the ideas of the well-known Fictitious Play (FP) algorithm (Brown 1951, Robinson 1951) for computing a Nash equilibrium of a finite non-cooperative game. In every iteration of FP, each player chooses a strategy that is a best response (with respect to that player's expected cost, which depends on decisions of all players) to the other players' strategies, assuming they will be chosen based on the empirical probability distribution induced by the historical frequency of the best response decisions in all previous iterations. For games of identical interest, i.e., ones in which all players have identical cost functions, it was shown in Monderer and Shapley (1996) that FP indeed converges to a Nash Equilibrium. *Sampled* Fictitious Play — in which, in each iteration, a *sample* of strategies from the above empirical probability distribution is used in calculation of best replies — was introduced by Lambert et al. (2005) and shown to converge to equilibrium for games of identical interests with probability 1 if the sample size increases at a polynomial rate with iterations. These results have motivated the use of SFP as a computationally efficient heuristic for solving optimization problems by representing variables (or groups of variables) as players in a game of identical interests, with each player's cost function equal to the shared objective function of the optimization problem. The strength of SFP as an optimization heuristic comes from the simplicity of best reply computations by each player; moreover, in each iteration, after the initial sampling step, each player's computations can be performed independently, and thus the algorithm is easily parallelizeable. Experiments with SFP in a variety of applications areas (see, for instance, Cheng et al. 2006, Garcia et al. 2000, 2007, Ghate et al. 2014, Lambert et al. 2005) also demonstrated that using a sample size of 1 in each iteration of SFP was sufficient for the algorithm to quickly discover high quality solutions in practice. More recent research has focused on applications of SFP to Dynamic Programming problems and Markov Decision Processes by viewing DP states as players in a game, actions available in each state as its available strategies, and total (expected) DP cost as the common cost function. Appropriately

modified versions of SFP with sample size of 1 have been shown to discover optimal policies of finite-horizon deterministic and stochastic DP problems (Sisikoglu 2009, Epelman et al. 2011) as well as discounted infinite horizon MDPs (Sisikoglu et al. 2011).

It should be noted that, to design SFP-based algorithms with provable convergence properties, the authors had to include various user-specified parameters in their versions of the algorithms. For example, the algorithm in Lambert et al. (2005) requires an increasing sequence of sample sizes to prove convergence to equilibrium (although implementations using samples of size 1 in each iteration perform well in computational tests). When applied to DP problems, appropriately modified versions of SFP can be shown to find optimal solutions using sample size of 1; however, to prove such results, Epelman et al. (2011) had to modify SFP by introducing a decreasing sequence of deliberate sampling errors to ensure discovery of a globally optimal solution, and both Epelman et al. (2011) and Sisikoglu (2009) used a version of SFP with a limit on the number of most recent past decisions of the players used to maintain their empirical distributions. Although the particular choices of values of these parameters in the algorithm's implementation could affect its computational performance, theoretical understanding of this dependence was lacking, calling for burdensome empirical tuning.

Unlike the preceding versions, parameter-free SFP, presented in this paper for finite-horizon deterministic DP problems with finite state and action spaces, is a "classic" SFP, with samples (of size 1) drawn directly from the full history of players' past best responses. We prove that the optimal DP solution will be discovered by the algorithm in a finite number of iterations with probability 1, relying on natural random tie-braking in best reply computations to establish optimality.

To test computational performance of the parameter-free SFP, we formulate a DP model of a novel path-finding problem arising in maritime navigation, discussed in details in Section 3. The application, in which we consider finding a path for a vessel traveling in a seaway with the goal of reducing roll motion due to the surrounding waves, provides a setting in which a high-quality solution to a large-scale DP needs to be delivered within a few seconds of computation,

despite computationally expensive action cost evaluations. As our computational experiments show, parameter-free SFP finds high quality solutions after only a few iterations, which could be executed in seconds by appropriately parallelizing the algorithm.

## 2. Parameter-Free Sampled Fictitious Play Algorithm

In this paper, we consider finite-horizon deterministic DP problems with finite state and action spaces. It is well known (see, e.g., Denardo 2003, Chapter 4) that any such DP can be cast as a shortest (or minimum cost) path finding problem on an acyclic *staged* directed network (defined rigorously in subsection 2.1) constructed by associating each stage-indexed state with a node and each action with a directed arc of length equal to the cost of applying this action in the state corresponding to the tail node of the arc. (Here, we allow for arc "lengths" to be negative.) Finally, a terminal node and corresponding arcs are introduced to capture the end of the DP horizon and terminal costs. Given an initial state, an optimal policy of the DP can then be found by applying any shortest path algorithm to the resulting network; variations of Dijkstra's algorithm (Dijkstra 1959) are a popular choice. (Conversely, shortest path problems on acyclic directed networks can be solved by DP methodologies, with Dijkstra's algorithm interpreted as a method of solving the DP functional equation via reaching.) Due to this equivalency between DPs and shortest path problems in acyclic networks, we present our algorithm and its analysis using the terminology of shortest path problems, to simplify the discussion.

In this section we introduce a parameter-free variation of a Sampled Fictitious Play (SFP) algorithm for solving shortest path problems on acyclic staged directed networks and prove its convergence to an optimal solution.

### 2.1. Notation and problem definition

Consider a directed graph $G = (\mathcal{V}, \mathcal{A})$, where $\mathcal{V} = \{1, 2, ..., n+1\}$ is the set of nodes and $\mathcal{A}$ is the set of arcs between the nodes. We assume that $G$ has no directed cycles and consists of $S+1$ stages, where the starting node 1 is the only node in stage 1, the target node $n+1$ is the only node in stage $S+1$, and all the arcs in $\mathcal{A}$ connect nodes from some stage $s \leq S$ to stage $s+1$.

(Note that we can make this assumption without loss of generality by introducing dummy nodes breaking up the arcs spanning multiple stages. We focus on staged networks since they are a natural representation of finite-horizon DPs.) We let $\mathcal{V}_s \subset \mathcal{V}$, for $s \leq S+1$, denote the set of all the nodes in stage $s$, and $\mathcal{A}_i \subset \mathcal{A}$, $i \in \mathcal{V}$, be the set of all outgoing arcs from node $i$, where by assumption $\mathcal{A}_i \neq \emptyset$ for $i = 1, \ldots, n$. Let $N$ be the maximum number of outgoing arcs of any node, that is, $N = \max\{|\mathcal{A}_i| : i \in \mathcal{V}\}$. For each arc $a \in \mathcal{A}$ we have a cost function $c(a)$ associated with traversing that arc. The objective of the Shortest Path Problem is to find a minimum-cost path from node 1 to $n+1$, where the cost of a path is equal to the sum of arc costs along that path.

### 2.2. Parameter-Free SFP Algorithm

We analyze the shortest path problem as a common utility non-cooperative game in which each node $i = 1, \ldots, n$ corresponds to a player and the outgoing arcs of node $i$, $\mathcal{A}_i$, correspond to the set of strategies, or actions, available to player $i$. Once each player chooses an action, the disutility, or cost, incurred by all the players in the network is the total cost associated with the path formed by those actions. That is, when player/node 1 (always the first node on the path) chooses an action $(1, j) \in \mathcal{A}_1$ for some $j \in \mathcal{V}_2$, $j$ becomes the next player/node on the path. Then, the action/arc chosen by player $j$ identifies the next player/node on the path, and so on until we reach node $n+1$ (always the last node on the path). We say that the nodes on this path from 1 to $n+1$ are *in play*, since the costs of their actions determine the costs incurred by all the players, irrespective of whether they are in play or not. For ease of notation, we assume there is a unique optimal path (i.e., path with minimum cost) from 1 to $n+1$. Let $p_s^\star \in \mathcal{V}_s$, for $s \leq S+1$, denote the player on that optimal path in stage $s$ and, with a slight abuse of notation, $a_s^\star = a_{p_s^\star} \in \mathcal{A}_{p_s^\star}$ denote the optimal action for player $p_s^\star$. Note that the following discussion can be easily extended to the case with multiple optimal paths, and the assumption is made solely for the purpose of clear and concise notation.

We now present our parameter-free variation of the Sampled Fictitious Play algorithm. In every iteration $k$ of the algorithm, we maintain, for each player $i = 1, \ldots, n$, $H_i^k$ — an ordered set of

actions in $\mathcal{A}_i$, representing the history of the player's *best replies* in the algorithm so far, up to iteration $k$.

**Sampled Fictitious Play (SFP) Algorithm**

*Step 0: Initialization.* For each player $i = 1, \ldots, n$, select an action from $\mathcal{A}_i$ uniformly at random, and set $H_i^1$ equal to that action. Set $k = 1$.

*Step 1: Draw.* For every player $i = 1, \ldots, n$, draw an action from $H_i^k$ uniformly at random.

*Step 2: Best Reply.* For each player $i = 1, \ldots, n$, compute a best reply to the actions drawn by other players, breaking ties uniformly at random.

*Step 3: Update.* Append the best replies computed in Step 2 to the history of each player, to get $H_i^{k+1}$ for $i = 1, \ldots, n$. Set $k = k + 1$ and go back to Step 1.

In Step 2, each player $i = 1, \ldots, n$ is given an opportunity to compute a *best reply*, i.e., change its action to any element of $\mathcal{A}_i$ in an effort to improve the cost of the path formed by the actions drawn in Step 1. The best reply calculation is performed independently for each player, and only takes into account the players' draws from Step 1. If a player is not in play, i.e., not on the path formed by actions drawn in Step 1, its decision does not affect the total cost, and the player chooses its reply from set $\mathcal{A}_i$ uniformly at random due to random tie breaking rule.

It is important to emphasize that, unlike the majority of metaheuristics available in the literature, including previous versions of Sampled Fictitious Play algorithms, our parameter-free SFP algorithm proposed here does not have any parameters that require customization or fine-tuning. The random tie-braking in Step 2 provides a natural and sufficient randomness to the algorithm to avoid "getting stuck" at a local optimal solution as well as facilitate the discovery of an optimal path in a finite number of iterations, as we show in the following subsection.

It should be noted that, in practice, the algorithm can be implemented more economically than suggested by the above description (see discussion in subsection 4.1); however, for analysis of theoretical properties of the algorithm in this section it is convenient to refer to the full statement of the algorithm above.

### 2.3. Algorithm Analysis: Convergence to the Optimal Solution

We now analyze the behavior of the parameter-free SFP algorithm. We begin by studying the probability of drawing the optimal path in Step 1 of some future iteration of the algorithm. To assist the discussion, we let $D_s^k$, for $s = 1, \ldots, S$, denote the event corresponding to player $p_s^\star$ drawing action $a_s^\star$ in Step 1 of iteration $k$, and $\mathbb{H}_s^k[l:m]$ — the event that, in Step 1 of iteration $k$, player $p_s^\star$ samples its action from among elements $l,\ l+1, \ldots,\ m-1,\ m$ of its history, where $l \geq 1$ and $m \leq k$. Finally, note that $H_i^k \in (\mathcal{A}_i)^k$ for $i = 1, \ldots, n$, where $(\mathcal{A}_i)^k$ is the set of all possible $k$-dimensional vectors of elements of $\mathcal{A}_i$ (i.e., all possible histories of player $i$ at the beginning of iteration $k$), and denote by $\mathbf{H}^k = [H_1^k, \ldots, H_n^k] \in \mathbf{A}^k := (\mathcal{A}_1)^k \times \cdots \times (\mathcal{A}_n)^k$ the collection of best reply histories for all the players at the beginning of iteration $k$.

PROPOSITION 1. *At the beginning of iteration $K$,*

$$\Pr(D_S^{k_0} | \mathbf{H}^K) = 1 \text{ for } k_0 \geq K, \tag{1}$$

$$\Pr(D_{S-1}^{k_1} | \mathbf{H}^K) \geq \frac{1}{2N} \text{ for } k_1 \geq 2K, \tag{2}$$

*and for $J \geq 2$,*

$$\Pr(D_{S-J}^{k_J} | \mathbf{H}^K) \geq f(J) \text{ for } k_J \geq 2^J K, \tag{3}$$

*where $f(J) = \frac{1}{2^{2^J-1}} \cdot \frac{1}{N^{2^J-2}}$. Moreover, for $J \geq 2$,*

$$\Pr\left(\bigcap_{j=0}^{J} D_{S-j}^{k_J} | \mathbf{H}^K\right) \geq h(J) \text{ in any iteration } k_J \geq 2^J K, \tag{4}$$

*where $h(J) = \frac{2}{2^{2^J}} \cdot \frac{1}{N^{2^J-1}}$.*

*In all of the above probability estimates, conditioning is done only on the history up to iteration $K$.*

*Proof.* First, observe that since stage $S+1$ of our network has only one node $(n+1)$, player $p_S^\star$ has only one action to choose from, $a_S^\star = (p_S^\star, n+1)$, and thus its history at any iteration consists of copies of $a_S^\star$, implying $\Pr(D_S^k | \mathbf{H}^K) = 1$ for all $k \geq K$, establishing (1).

Suppose the algorithm just completed iteration $K-1$ (i.e., each player's history, $H_i^K$, consists of $K$ entries, for $i=1,\ldots,n$). Consider stage $S-1$ and player $p^\star_{S-1}$. At worst, all $K$ entries in its history, $H^K_{p^\star_{S-1}}$, are non-optimal. However, in any future iteration, due to the above discussion, this player's best reply is optimal if it is in play in this iteration, or, if it is not in play, its randomly selected best reply is optimal with probability at least $\frac{1}{N}$. Therefore, at any future iteration of the algorithm, $p^\star_{S-1}$ selects $a^\star_{S-1}$ as its best reply with probability at least $\frac{1}{N}$. In other words, each entry of $H_{p^\star_{S-1}}$ beginning with $K+1$ is equal to $a^\star_{S-1}$ with probability at least $\frac{1}{N}$, as estimated at the beginning of iteration $K$. This allows us to estimate the probability of $p^\star_{S-1}$ sampling $a^\star_{S-1}$ after another $K$ iterations, i.e., in any iteration $k_1 \geq 2K$, as follows:

$$\begin{aligned}
\Pr(D^{k_1}_{S-1}|\,\mathbf{H}^K) &= \Pr(D^{k_1}_{S-1}\,|\,\mathbf{H}^K, \mathbb{H}^{k_1}_{S-1}[1:K]) \cdot \Pr(\mathbb{H}^{k_1}_{S-1}[1:K]|\,\mathbf{H}^K) \\
&\quad + \Pr(D^{k_1}_{S-1}\,|\,\mathbf{H}^K, \mathbb{H}^{k_1}_{S-1}[K+1:k_1]) \cdot \Pr(\mathbb{H}^{k_1}_{S-1}[K+1:k_1]|\,\mathbf{H}^K) \\
&\geq \Pr(D^{k_1}_{S-1}\,|\,\mathbf{H}^K, \mathbb{H}^{k_1}_{S-1}[K+1:k_1]) \cdot \Pr(\mathbb{H}^{k_1}_{S-1}[K+1:k_1]|\,\mathbf{H}^K) \\
&\geq \frac{1}{N} \cdot \frac{k_1 - K}{k_1} \geq \frac{1}{2N},
\end{aligned}$$

establishing (2). It should again be noted that the above probability is estimated at the beginning of iteration $K$, i.e., conditioning is done only on the history up to iteration $K$.

Let us proceed to stage $S-2$ and player $p^\star_{S-2}$. At any iteration of the algorithm, if this player is not in play, it will choose $a^\star_{S-2}$ as the best reply with probability at least $\frac{1}{N}$. Moreover, if $p^\star_{S-2}$ is in play, it will identify $a^\star_{S-2}$ as the best reply if $p^\star_{S-1}$ has drawn $a^\star_{S-1}$ in Step 1 of the iteration — but by (2) the probability of this happening in iteration $k \geq 2K$ (conditional on the history at the beginning of iteration $K$) is at least $\frac{1}{2N}$. In other words, each entry of $H_{p^\star_{S-2}}$ beginning with $2K+1$ is equal to $a^\star_{S-2}$ with probability at least $\frac{1}{2N}$. So, if we let another $2K$ iterations elapse, for $k_2 \geq 4K$,

$$\begin{aligned}
\Pr(D^{k_2}_{S-2}|\,\mathbf{H}^K) &\geq \Pr(D^{k_2}_{S-2}\,|\,\mathbf{H}^K, \mathbb{H}^{k_2}_{S-2}[2K+1:k_2]) \cdot \Pr(\mathbb{H}^{k_2}_{S-2}[2K+1:k_2]|\,\mathbf{H}^K) \\
&\geq \frac{1}{2N} \cdot \frac{k_2 - 2K}{k_2} \geq \frac{1}{4N}.
\end{aligned}$$

Once again, the above probability estimate is calculated at the beginning of iteration $K$, i.e., conditioning is done only on the history up to iteration $K$, and not on the trajectory of the algorithm in iterations $K,\ K+1,\ldots,\ k_2-1$. This analysis establishes the basis of induction for (3) with $J=2$. Moreover, for $k \geq 4K$,

$$\Pr(D^k_{S-2} \cap D^k_{S-1} \cap D^k_S | \, \mathbf{H}^K) \geq \frac{1}{4N} \cdot \frac{1}{2N} = \frac{1}{8N^2},$$

establishing the basis of induction for (4) with $J=2$.

To show the inductive step, suppose (3) and (4) hold for $j=2,\ldots,J$. Consider stage $S-(J+1)$ and player $p^\star_{S-(J+1)}$. At any iteration of the algorithm, if this player is not in play, it will choose $a^\star_{S-(J+1)}$ as the best reply with probability at least $\frac{1}{N}$. Moreover, if $p^\star_{S-(J+1)}$ is in play, it will identify $a^\star_{S-(J+1)}$ as the best reply if $p^\star_{S-j}$ has drawn $a^\star_{S-j}$ in Step 1 of the iteration for $j=0,\,1,\,2,\ldots J$ — applying (4), the probability of this happening in iteration $k_J \geq 2^J K$ (conditional on the history at the beginning of iteration $K$) is at least $h(J)$. In other words, each entry of $H_{p^\star_{S-(J+1)}}$ beginning with $2^J K+1$ is equal to $a^\star_{S-(J+1)}$ with probability at least $h(J)$. So, for $k_{J+1} \geq 2 \cdot 2^J K = 2^{J+1} K$,

$$\begin{aligned}
\Pr(D^{k_{J+1}}_{S-(J+1)} | \, \mathbf{H}^K) &\geq \Pr(D^{k_{J+1}}_{S-(J+1)} \, | \, \mathbf{H}^K,\, H^{k_{J+1}}_{S-(K+1)}[2^J K+1 : k_{J+1}]) \times \\
&\qquad \times \Pr(H^{k_{J+1}}_{S-(K+1)}[2^J K+1 : k_{J+1}] | \, \mathbf{H}^K) \\
&\geq h(J) \cdot \frac{k_{J+1} - 2^J K}{k_{J+1}} \geq \frac{h(J)}{2} = \frac{1}{2^{2^J}} \cdot \frac{1}{N^{2^{J-1}}} = f(J+1),
\end{aligned}$$

establishing the inductive step for (3). The inductive step for (4) follows easily from the above, since, for $k \geq 2^{J+1} K$,

$$\begin{aligned}
\Pr\left( \bigcap_{j=0}^{J+1} D^k_{S-J} | \, \mathbf{H}^K \right) &= \Pr\left( \bigcap_{j=0}^{J} D^k_{S-J} | \, \mathbf{H}^K \right) \times \Pr(D^k_{S-(J+1)} | \, \mathbf{H}^K) \\
&\geq h(J) \cdot \frac{1}{2^{2^J}} \cdot \frac{1}{N^{2^{J-1}}} \\
&= \frac{2}{2^{2^{J+1}}} \cdot \frac{1}{N^{2^J}} = h(J+1).
\end{aligned}$$

Once again, the above probability estimate is calculated at the beginning of iteration $K$, i.e., conditioning is done only on the history up to iteration $K$, and not on the trajectory of the algorithm in iterations $K,\ K+1,\ldots,\ k-1$. $\square$

Let

$$P^k := \bigcap_{s=1}^{S} D_s^k$$

denote the event corresponding to drawing the optimal path in Step 1 of iteration $k$. Applying bound (4) of Proposition 1 for $J = S-1$, we have the following corollary.

COROLLARY 1. *At the beginning of iteration $K$,*

$$\Pr(P^k \,|\, \mathbf{H}^K) \geq q^\star := \frac{2}{2^{2^{S-1}}} \cdot \frac{1}{N^{2^{S-2}}} \text{ for } k \geq 2^{S-1}K. \tag{5}$$

Next, we will use Corollary 1 to show that parameter-free SFP will discover an optimal path in a finite number of iterations with probability 1.

Let us define

$$\hat{k}_j := 2^{j(S-1)},\ j = 0, 1, 2, \ldots. \tag{6}$$

This subsequence of $\mathbb{Z}_+$ has the following property: applying Corollary 1 for $K = \hat{k}_{j-1}$, we can estimate the probability of drawing an optimal path in iteration $\hat{k}_j$, conditional on the history up to iteration $K$, as

$$\Pr(P^{\hat{k}_j} \,|\, \mathbf{H}^{\hat{k}_{j-1}}) \geq q^\star,\ j = 1, 2, \ldots.$$

Leveraging this lower bound, in the following analysis we will first concentrate on the *subsequence* of iterations $\{\hat{k}_j,\ j = 1, 2, \ldots\}$ of SFP. In particular, this (constant) lower bound allows us to show that the number of iterations within the subsequence $\{\hat{k}_j\}_{j=1}^{\infty}$ required to draw the optimal path for the first time is stochastically bounded above by a geometric random variable with parameter $q^\star$. This implies that the optimal path will be drawn in a finite number of iterations within the subsequence with probability one. Finally, since the subsequence $\{\hat{k}_j\}_{j=1}^{\infty}$ tends to infinity, the same statement can be extended to the complete sequence of iterations of SFP.

PROPOSITION 2. *For any $j = 1, 2, \ldots$, $\Pr(P^{\hat{k}_j}) \geq q^\star$. Moreover, the random variable $\hat{X}$, defined as the number of iterations within the subsequence $\{\hat{k}_j\}_{j=1}^{\infty}$ required to draw the optimal path for the first time, is stochastically bounded above by a geometric random variable with parameter $q^\star$. That is,*

$$\Pr(\hat{X} \leq m) \geq q^\star \cdot \sum_{j=1}^{m} (1 - q^\star)^{j-1},\ m = 1, 2, \ldots \tag{7}$$

*Proof.* Consider $P^{\hat{k}_j}$ for any integer $j \geq 1$. Then, invoking the law of total probability (see, e.g., Ross 1995, pp. 23–24 or Zwillinger and Kokoska 1999, p. 31),

$$\Pr(P^{\hat{k}_j}) = \mathrm{E}_{\mathbf{H}^{\hat{k}_{j-1}}} \left[ \Pr(P^{\hat{k}_j} \mid \mathbf{H}^{\hat{k}_{j-1}}) \right] \geq q^\star, \tag{8}$$

where the inequality is the result of the lower bound on the value of all conditional probabilities proved in Corollary 1. In light of (8),

$$\Pr(\hat{X} \leq m) = 1 - \Pr(\hat{X} > m) \geq 1 - (1 - q^\star)^m. \tag{9}$$

Rewriting the right-hand side of (7) using the formula for geometric series, we get

$$q^\star \cdot \sum_{j=1}^{m} (1 - q^\star)^{j-1} = q^\star \, \frac{1 - (1 - q^\star)^m}{1 - (1 - q^\star)} = 1 - (1 - q^\star)^m. \tag{10}$$

Substituting (10) into the right-hand side of (9) completes the proof. □

Returning to the full sequence of SFP iterations, consider a random variable $X$, defined as the number of iterations of SFP required to draw the optimal path for the first time. Clearly, $\Pr(X \leq \hat{k}_j) \geq \Pr(\hat{X} \leq j)$ for $j = 1, 2, \ldots$, which leads to the following corollary of Proposition 2.

Corollary 2. *Random variable $X$, defined as the number of iterations of parameter-free SFP required to draw the optimal path for the first time, satisfies*

$$\Pr(X \leq m) \geq \Pr(\hat{X} \leq k(m)) \geq q^\star \cdot \sum_{j=1}^{k(m)} (1 - q^\star)^{j-1}, \;\; m = \hat{k}_1, \hat{k}_1 + 1, \hat{k}_1 + 2, \ldots, \tag{11}$$

*where $k(m) := \max\{j : \hat{k}_j \leq m\}$.*

Now we establish the key result of this paper.

Theorem 1. *The optimal path will be drawn by the parameter-free SFP algorithm in a finite number of iterations with probability one. That is, $\Pr(X < \infty) = 1$. Furthermore, with probability one, the number of iterations until, for the first time, the players' reply in Step 2 of the SFP algorithm will correspond to the optimal path is finite.*

*Proof.* Combining the proof of Proposition 2 and the statement of Corollary 2 we have:

$$\Pr(X < m) = \Pr(X \leq m-1) \geq 1 - (1-q^\star)^{k(m-1)}.$$

Note that as $m$ approaches infinity, so does $k(m-1)$. Then, letting $m$ approach infinity and noting that $(1-q^\star) < 1$, we obtain

$$\Pr(X < \infty) = \lim_{m\to\infty} \Pr(X < m) \geq \lim_{m\to\infty} \left(1 - (1-q^\star)^{k(m-1)}\right) = 1,$$

which implies that $\Pr(X < \infty) = 1$.

To prove the second part of the theorem, observe that if all players $p_s^\star,\ s = 1, 2, ..., S$, draw an optimal path in Step 1 of some iteration of the algorithm, then none of these players will change their actions in Step 2 of this iteration. □

In implementations of SFP it is typical to keep track of the *incumbent solution*, i.e., the shortest path discovered so far in the algorithm, either by sampling or computing best replies for nodes that are in play. Once the optimal path is discovered by any of the above means, the incumbent solution becomes, and stays, equal to this optimal path. The following corollary is an immediate consequence of Theorem 1.

Corollary 3. *Let the incumbent solution be equal to the best solution discovered in the parameter-free SFP algorithm by sampling or computing best replies. Then, with probability one, the incumbent will be set to the optimal path in a finite number of iterations.*

It should be noted that probabilistic bounds derived in Corollaries 1 and 2 are very pessimistic. They are sufficient to provide convergence of parameter-free SFP and easy to derive; however, in practice the algorithm tends to perform much better than the above analysis may suggest.

## 3. Minimum-motion Path Finding Problem in Maritime Navigation

To demonstrate the computational performance of the parameter-free SFP algorithm proposed in this paper, we used a novel optimal path-finding (vessel routing) problem arising in maritime navigation as a computational testbed. We describe the problem and formulate its DP model in this section, and apply parameter-free SFP to solve the resulting DP in Section 4.

### 3.1. Path-finding in Maritime Navigation

Optimal path finding problems have been studied in various settings and applications for decades. Within the vessel routing setting, Zermelo (1931) introduced a problem of steering a ship along a minimum-time path through a strong current region, which became to be known as the Zermelo Navigation Problem, one of the classical problems in optimal control and calculus of variations. A significant volume of work in naval navigation has followed (e.g., Faulkner 1963a,b, Papadakis and Perakis 1990, Perakis and Papadakis 1988, 1989, Kimball and Story 1998), the majority of which has assumed that a closed form speed function is available to the user, thus facilitating analytical solution approaches. Path planning problems are also studied in computational geometry (e.g., see survey by Mitchell 2000), robotics (e.g., Lanthier et al. 1999, Rowe 1997, Rowe and Ross 1990, Sun and Rief 2005), air traffic management (e.g., Nilim et al. 2001, Nilim and El Ghaoui 2004), and many other applications. However, the majority of work to date focuses on minimizing travel time, distance, energy consumed, or other additive objective functions, whereas our goal is more complex: we aim to reduce a measure of vessel roll (discussed below) without significantly increasing travel time to destination compared to the most direct path.

Any vessel moving in a seaway is subjected to the effects of surrounding waves, which impact its motions, especially roll. This is due to the fact that roll exhibits resonant behavior, and a typical natural period of roll coincides with waves frequencies often occurring in sea. Significant roll motions affect crew's and passengers' comfort, and limit their productivity and capacity. The motions can also damage cargo, jeopardize safety of the vessel and negatively impact the completion of the ship's missions (e.g., helicopter landing, rescue and recovery operation). In Fang and Luo (2007), the authors point out that "the serious roll motion generally affects the ship stability, comfort and efficiency of crews, accuracy of electrical mechanism, and ship course. Therefore reducing roll motion is advantageous for ships in waves." To date, there has been significant research in the field of naval architecture and marine engineering delivering a variety of roll motion control devices since "roll is the largest and most undesirable component ship motion" (Treakle et al. 2000). Examples

of such devices (varying between passive and active controllers) are bilge keels, anti-roll tanks, moving weights, gyroscopic stabilizers, and stabilizing fins (Smith and Thomas 1990). In contrast, the goal of our work is to find an alternative path to the vessel's destination to reduce the roll motion, while avoiding a significant increase in travel time. This is particularly important in the settings where simply delaying travel until unfavorable conditions pass is not an option, such as in the case of rescue and military missions, as well as transportation of time sensitive and perishable goods.

To enable accurate, up to date modeling of the vessel's environment, it is important to utilize recent developments in meteorological and oceanographic sensing and forecasting technologies, which have made it possible for maritime vessels to have access to more accurate and current environmental data than ever before (e.g., Plant et al. 2005, Johnson et al. 2009). Advances in modeling vessel interaction with time-varying nonlinear wave-fields allows navigation systems to take advantage of this accurate data (Zhang et al. 2010). In our earlier work (Dolinskaya 2012), we have focused on the fastest path finding problem capturing the detailed real-time information with the vessel satisfying a set of operability and dynamics constraints. Here, we extend that work to the minimum-motion path finding problem, and integrate detailed, real-time information about the surrounding environment and include vessel's dynamics restrictions (such as minimum turning radius) and innovative motion prediction model to deliver a high fidelity model. Our high fidelity model calls for a new solution method, since small computational time is critical to real-time navigation in dynamic environments. The proposed parameter-free SFP algorithm meets this need, as demonstrated by our computational study.

### 3.2. Dynamic Programming Formulation of the Minimum-Motion Path Finding Problem

We consider the problem of minimizing the root-mean-squared (RMS) value of the motion experienced by a vessel during travel from a given starting point $s \in \mathbb{R}^2$ to a specified final target point $f \in \mathbb{R}^2$, leaving $s$ at time $t_s = 0$. To account for constraints on minimum turning radius, we add the starting heading angle, $\theta_s \in S^1$, and, if appropriate, the final heading angle, $\theta_f \in S^1$, to the set of

input parameters. (Here, $S^1 := [0,\ 2\pi]$.) We let $r > 0$ denote the minimum turning radius and $v > 0$ denote vessel speed, which is assumed to be constant. We focus our discussion on the *roll*, which is a rotational motion about the longitudinal axis passing through the vessel. Let $\phi(a, \theta_a, t_a) \in \mathbb{R}$ denote the roll (measured in degrees) at point $a \in \mathbb{R}^2$ at time $t_a \geq 0$ when vessel's heading direction is $\theta_a \in S^1$. To measure the RMS roll along a path from $s$ to $f$, we discretize the path into a set of equally spaced points, $\Delta t$ time units apart (and $v\Delta t$ distance units apart), and let $\phi_i$ denote the roll experienced by a vessel at the $i^{th}$ such point. The discretization is a necessary approach since roll motion is evaluated using simulation models and no analytical form of $\phi(a, \theta_a, t_a)$ is available. Then our goal is to minimize the following quantity:

$$\text{RMS}_{\text{Roll}} = \sqrt{\frac{\phi_1^2 + ... + \phi_T^2}{T}}, \tag{12}$$

where $T$ is the number of discretization points on this path. Root-mean-squared roll is a discretized version of the control performance measure function widely used in optimal vessel control when assessing ship's exposure to motions, especially roll (see Chapter 6, Automatic Control of Ships, in Fossen 1994). This is a well-established discretization due to its applicability in practice, when closed form control equations of the roll are replaced with more accurate and realistic computer simulations, as it is in our case.

Our general problem statement is as follows: Find a path starting at time $t_s = 0$ from configuration $(s, \theta_s) \in \mathbb{R}^2 \times S^1$ to the final state $(f, \theta_f) \in \mathbb{R}^2 \times S^1$ that minimizes RMS roll *without significant increase in travel time from a fastest (direct) path*, where path curvature is constrained by $r > 0$ and vessel speed is $v > 0$.

Note that the above problem description does not rigorously define the constraint "without significant increase in travel time." The general idea is that we are not interested in making a drastic detour from a direct (and fastest) path with only a minor improvement in the experienced roll motion. Thus, we want to integrate a trade-off between motion and travel time. Our DP-based approach to the minimum-motion path finding problem will incorporate this tradeoff through several modeling choices.

**Objective function:** The overall goal of our path finding problem is to minimize the RMS roll defined in (12). However, to apply dynamic programming, the objective function has to be additive, so, instead of minimizing $\mathrm{RMS}_{\mathrm{Roll}}$, we focus on minimizing the total sum of squared roll values along a path from $(s, \theta_s)$ to $(f, \theta_f)$:

$$\mathrm{Sq}_{\mathrm{Roll}} = \phi_1^2 + ... + \phi_T^2. \tag{13}$$

Recall that to discretize the path we spaced the motion measurement points at equal time intervals $\Delta t$ (and equal distance intervals $v\Delta t$), so a longer path would have a larger number of terms ($T$) in equation (13). At the same time, when the values of these additional summation terms are significantly smaller than along a more direct path, $\mathrm{Sq}_{\mathrm{Roll}}$ would be able to capture this phenomenon and would favor the longer path. Consequently, by minimizing $\mathrm{Sq}_{\mathrm{Roll}}$, our path-finding model implicitly incorporates a trade-off between total vessel roll and total travel time.

**DP state definition:** An important characteristic of vessel navigation is the constraint on the sharpest turn it can feasibly make. Thus, in addition to capturing the current location of the vessel, our DP model state keeps track of the vessel heading angle. This enables us to implement *control-feasible* paths whose minimum turning radius is bounded by $r$. Also, in the case of evolving wave-fields presented in this paper, vessel motions depend on time as well as its location and heading angle. Thus, we let $(a, \theta_a, t_a)$ be the DP state, where $a \in \mathbb{R}^2$ corresponds to the location of the vessel, $\theta_a$ denotes it heading angle, and $t_a$ is the time. (In the fastest-path finding version of this problem in Dolinskaya 2012 we were able to omit the time variable from the DP state space without loss of optimality. However, this technique cannot be applied to the minimum motion model. Note that adding time to the state description exponentially increases the state space and computational demands of the model.)

**DP model:** To formulate the problem using dynamic programming, we discretize vessel paths into waypoints, with $l > 0$ denoting the Euclidean distance between each pair of adjacent waypoints. This construction is illustrated in Figure 1. Here, $a \in \mathbb{R}^2$ is the location of the current waypoint, and $\theta_a \in S^1$ is the heading of the vessel at $a$; for modeling purposes we discretize $S^1$ into a finite set

of values $\Delta\theta$ degrees apart. To generate waypoints adjacent to $a$, we select a discrete set of points (with angular distance $\delta$) on the circle of radius $l$ centered at $a$; however, only points that can be reached from $(a, \theta_a)$ along control-feasible paths that are contained within the circle of radius $l$ centered at $a$ are included, again, to avoid excessively long paths. At each waypoint $b$ on the perimeter of the circle, we indicate which headings $\theta_b$ are achievable by control-feasible paths from $(a, \theta_a)$.

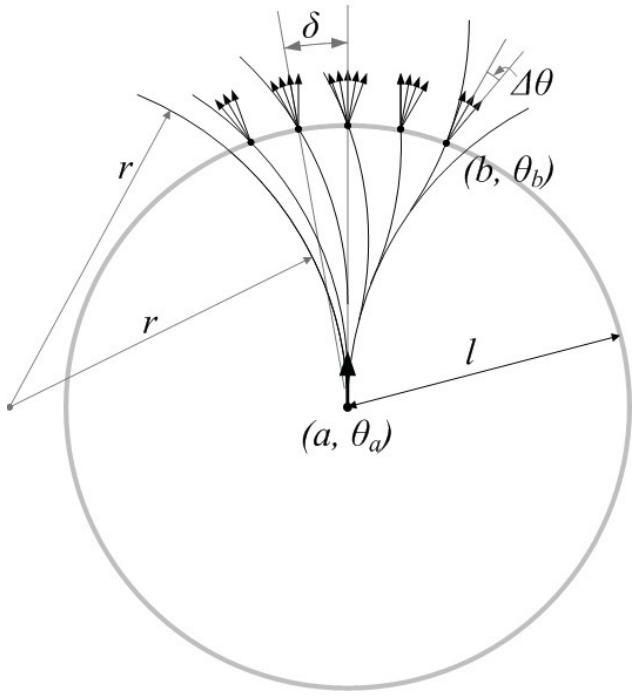



**Figure 1 Construction of adjacent waypoints**

Recall that it is important to keep track of the total travel time along the chosen path in order to avoid long detours with only marginal benefits to the total motion. In light of this, we will implement the shortest (and fastest) path between each pair of waypoints along the path. We use results from the well-studied Dubins car problem (Dubins 1957, Sussmann and Tang 1991, Boissonnat et al. 1994) that deliver a shortest path with bounded curvature from one point to another taking into account the initial and final heading angles. We let $\tau(a, \theta_a, b, \theta_b)$ denote the travel time along the shortest control-feasible path from configuration $(a, \theta_a)$ to the adjacent configuration $(b, \theta_b)$, and let

$\mathcal{R}(a,\theta_a,b,\theta_b,t_a)$ denote the sum of squared rolls experienced by a vessel along this path departing $a$ at time $t_a$. This approach ensures continuity of vessel heading along the path and enforcement of the system dynamics restrictions.

For each DP state $(a,\theta_a,t_a)$ we define the optimal value function $g(a,\theta_a,t_a)$ to be the minimum sum of squared rolls experienced by the vessel over all the paths from the initial state $(s,\theta_s,t_s=0)$ to point $a$ that arrive at $a$ with the heading angle $\theta_a$ at time $t_a$. Then, we have the following DP forward functional equation: $g(s,\theta_s,t_s=0)=0$ and

$$g(b,\theta_b,t_b)=\min_{a,\theta_a\,:\,b \text{ adjacent to } a} g(a,\theta_a,t_b-\tau(a,\theta_a,b,\theta_b))+\mathcal{R}(a,\theta_a,b,\theta_b,t_b-\tau(a,\theta_a,b,\theta_b)) \qquad (14)$$

for $(b,\theta_b,t_b)\neq(s,\theta_s,0)$.

Recursively solving equation (14) until we reach the final state $(f,\theta_f)$ results in the optimal path.

### 3.3. Test Problem Instances for Computational Experiments

In our computational experiments detailed in the following section we considered multiple instances of the minimum-motion path finding problem using the S-175 containership model. In these test instances, the vessel speed was set to $v=10$ meters/sec (approximately 20 knots), and the minimum turning radius was set to $r=300$ meters. We generated a simulated wave-field by specifying the wave distribution parameters: significant wave height 3.25 meters, peak period 9.53 sec, and dominant wave direction 30 degrees (from SWW). The wave propagation model (Nwogu 2009, Nwogu and Lyzenga 2010) forecasts the evolution of the created wave-field over time, while the quasi-steady seakeeping and maneuvering MotionSim program (Zhang et al. 2010) evaluates the containership roll. Discretization parameter $\Delta t$ used for evaluation of $\mathrm{Sq}_{\mathrm{Roll}}$ and $\mathrm{RMS}_{\mathrm{Roll}}$ was set to 1 second.

The starting point $s$ and finish points $f$ in each instance were selected about 2200 meters apart, and we discretized the set of potential values of headings $\theta$ into 36 angles 10° apart. The waypoint distance parameter $l$ was set to 250 meters, and waypoints were selected $\delta=10^\circ$ apart; i.e., each point could have no more than 36 adjacent waypoints. To regularize the set of waypoints, we

rounded their coordinates to the nearest point on a 10 meter by 10 meter grid. Since the size of the grid is small compared to the value of $l$, only a small perturbation of the waypoints was required. Moreover, locating the waypoints on the grid works hand in hand with the wave field simulation engine, which also uses a grid of 10 by 10 meters.

## 4. Numerical Results

We will now provide the details of our implementation of parameter-free SFP for the DP model of the minimum-motion path finding problem discussed in Section 3.2, and discuss its excellent empirical performance (including a comparison with Dijkstra's algorithm).

### 4.1. Problem Characteristic and Challenges; Comparison with Dijkstra's Algorithm

In order to compare the performance of the parameter-free SFP and Dijkstra's algorithms, we ran a number of initial numerical tests for the S-175 containership. The DP network in our application was prohibitively large for a straightforward application of the traditional Dijkstra's algorithm (Dijkstra 1959) to be tractable. We attempted to mitigate this problem by implementing a rolling (receding) horizon procedure that "looks out" only a few (two to four) waypoints ahead before deciding on the next waypoint (for discussion of rolling horizon approaches to DP and shortest path problems see, e.g., Alden and Smith 1992, Lee and Denardo 1986, Ovacikt and Uzsoy 1994). However, due to the specific characteristics of the problem's objective function (discussed below), we observed that short rolling horizon guided the vessel in a locally-optimal direction and the resulting path drifted away from the target point without a good return strategy, while a longer rolling horizon lead to an exponential increase in computational time of the algorithm, quickly exceeding the practical threshold for real-time implementation. The specific challenging characteristics of this problem are:

1. **Expensive arc cost computation:** Motion prediction model, MotionSim, that evaluates function $\phi(\cdot)$ is computationally demanding. Thus, evaluating the cost of each arc in the DP network is a computationally expensive operation and the majority of computing time of any path-finding algorithm is spent running the MotionSim program. Since Dijkstra's algorithm is a breadth-focused

search algorithm, it would require evaluating arc costs for almost the entire network, making its implementation prohibitively time-demanding.

2. **Highly direction-dependent objective:** Due to the specific nature of the vessel roll motion, function $\phi(a, \theta_a, t_a)$ is highly direction-dependent. Thus, short look-ahead horizon results in the algorithm selecting a locally-optimal heading direction without giving significant weight to the need to eventually reach a pre-specified final point.

3. **Ergodic environment:** While the wave-field in our problem is a dynamic environment with direction, location and time dependent short-crested waves impacting the vessel motions, the underlying distribution of the waves has a single dominant wave direction. As a result, we are dealing with an ergodic system, and the limited vessel maneuverability, as reflected by the minimum turning radius $r$, reinforces the ergodic property (since the vessel encounters several waves before it can change its course). This characteristic further enhances the "drifting off" effect of a rolling-horizon type solution approach.

In particular, in our computational experiments with rolling horizon Dijkstra's algorithm we observed that (i) while the smaller rolling horizon (two waypoints look ahead) had a smaller run time (typically, 300 — 500 seconds), it often resulted in the computed path guiding the vessel in a locally optimal heading direction away from the target point $f$ which was never reached; (ii) on the other hand, increasing the rolling horizon to four, five or six waypoints ahead quickly increased the run time (typically, to 1800 — 2500 seconds), and in most cases, the algorithm timed out after two hours without finding a solution. In one instance, we ran the algorithm for 48 hours with six waypoints look ahead horizon and were not able to find a solution within that timeframe. Moreover, this problem is unlikely to benefit from other popular modifications of Dijkstra's algorithm, such as the $A^*$ heuristic (see, e.g., Pearl 1984), since the network lacks an accurate guidance function, i.e., a heuristic for estimating total cost of travel from the current to the final state, again, due to the challenging characteristics described above.

For comparison, we tested the computational performance of parameter-free SFP for 10, 20 and 30 iterations of the algorithm applied to the entire network of DP states between $s$ and $f$. As

discussed in Section 2, at every iteration the algorithm keeps track of the best solution (an $s$-to-$f$ path in the DP network) discovered so far, either by sampling or computing best replies. The run time of the algorithm grows linearly with the number of iterations performed, from around 500 seconds (for 10 iterations) to 1800 seconds (for 30 iterations). In most cases, the most significant improvement in the best path found occurred within the first 4 to 8 iterations. In addition, in most of the instances where the rolling horizon Dijkstra's algorithm found a solution before timing out, the solution found by the parameter-free SFP after 10 iterations was better. Thus, we chose to run the complete set of numerical experiments to evaluate the performance of our minimum-motion model using 10 iterations of the parameter-free SFP algorithm. As our experiments show, a strength of the parameter-free SFP algorithm is its ability to find a high-quality solution in just a few iterations.

Our implementation of parameter-free SFP took advantage of several inherent efficiencies possible in the algorithm. Note that the players (nodes) do not need to explicitly update their histories in those iterations when they are not in play. Instead, for each player, we only record the actions chosen in iterations when it was in play; we can sample an action from its history by conditioning on whether the action comes from the "in play" or "not in play" portion of history, and in the latter case sampling uniformly from its action space. The size and the almost tree-like structure of our problem network imply that, in any iteration, only a small fraction of players will be on the active path and thus computing their best replies; moreover, only a small fraction of feasible paths will need to be considered for each best reply computation. Thus, in each iteration, we need to sample actions for only a fraction of the players and evaluate only a small fraction of network arcs.

It should be pointed out that SFP also easily lends itself to parallelization. Indeed, since each player computes a best reply in Step 2 of the algorithm independently, and moreover, evaluations of each action available to a player are also independent, those computations can be performed in parallel. In the instances tested in our experiments, a typical $s$-to-$f$ path consisted of approximately 10 waypoints, and on average each player had 17 action options. Therefore, with sufficient

parallel processors available, we anticipate that the runtimes we report for our non-parallelized implementation can be reduced by 2 orders of magnitude. This would bring the computation time down to a few seconds, making it suitable for real-time implementation.

### 4.2. Further Numerical Tests with parameter-free SFP

To further test the performance of parameter-free SFP, we designed a collection of 18 test instances in which the angle of the vector $\overrightarrow{sf}$ spanned the entire $S^1$, to capture the effect of direction on the vessel roll motion. We set $\theta_s$ to the angle of vector $\overrightarrow{sf}$, and did not restrict the value of the final heading $\theta_f$. For each instance, we ran 10 iterations of parameter-free SFP, keeping track of the incumbent solution, and compared the best path found in 10 iterations to the shortest (straight line) path from $s$ to $f$.

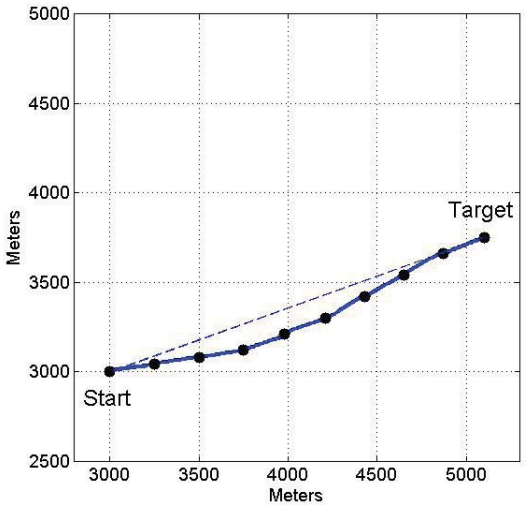
Table 1 summarizes the results for our experiments. The first column (labeled "$\theta_s$") lists the value of the angle $\overrightarrow{sf}$ in each of our 18 test instances. The second column reports iteration number at which the best found path was discovered by SFP, and the third column contains run times for 10 iterations of the algorithm. The fourth column displays the value of the DP objective function, i.e., $\mathrm{Sq}_{\mathrm{Roll}}$, measured in degrees squared, along the best path found by the parameter-free SFP. In the final two columns we compare this path to the shortest (straight line) path from $s$ to $f$: column 5 reports percentage change in $\mathrm{RMS}_{\mathrm{Roll}}$ (the original measure of roll), and column 6 — percentage change in total travel time.

According to Table 1, in most instances the path found by applying parameter-free SFP to the DP model of the problem significantly (by up to 77.8%) improved RMS roll compared to the shortest path, with increase in travel time ranging between 4% and 91%. The improvement in roll is particularly significant for the heading directions where the vessel experiences high roll values (greater than 700 degrees squared in total squared roll). However, in instances where motion values are small (see rows corresponding to $\theta_s = 180^\circ$, $200^\circ$, and $220^\circ$ in Table 1), our approach finds paths that are slightly worse than the shortest path. This phenomenon can most likely be explained by the fact that in those instances the DP network's arc costs are all similarly small in magnitude.

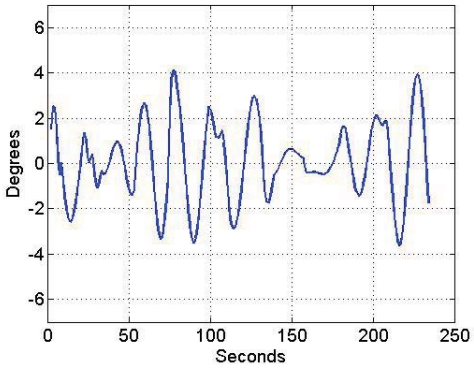
| $\theta_s$ | Iter. Best Found | SFP ComputTime (s) | $\mathrm{Sq}_{\mathrm{Roll}}$ | $\mathrm{RMS}_{\mathrm{Roll}}$ | Travel Time |
|---|---|---|---|---|---|
| 0° | 3 | 469 | 1174.80 | -62.6% | 45% |
| 20° | 5 | 501 | 709.62 | -34.8% | 4% |
| 40° | 8 | 529 | 974.95 | -39.0% | 7% |
| 60° | 2 | 739 | 1697.42 | -64.7% | 71% |
| 80° | 5 | 477 | 2157.44 | -65.4% | 55% |
| 100° | 10 | 503 | 1159.45 | -74.7% | 83% |
| 120° | 3 | 465 | 1167.47 | -47.7% | 51% |
| 140° | 1 | 451 | 527.10 | -19.2% | 16% |
| 160° | 10 | 484 | 262.52 | -6.7% | 5% |
| 180° | 1 | 476 | 124.05 | 0.4% | 7% |
| 200° | 1 | 407 | 36.33 | 3.7% | 6% |
| 220° | 5 | 469 | 20.13 | 1.5% | 4% |
| 240° | 1 | 473 | 69.01 | -6.0% | 6% |
| 260° | 10 | 522 | 194.73 | -13.8% | 12% |
| 280° | 4 | 433 | 495.83 | -5.0% | 7% |
| 300° | 10 | 499 | 1256.30 | -27.1% | 26% |
| 320° | 10 | 719 | 965.40 | -77.8% | 91% |
| 340° | 2 | 455 | 1494.61 | -75.0% | 76% |

**Table 1 Results of computational experiments on 18 test instances. Fifth and sixth columns contain percentage change in RMS roll and travel time of the path found by the parameter-free SFP algorithm versus the direct (straight line) path.**

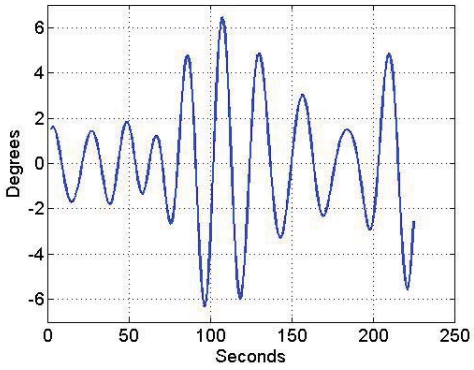
As a result, there is no significant difference between various feasible paths, and the path found by the algorithm in 10 iterations is slightly suboptiomal.




(a) Path found by SFP (solid) vs. the direct path (dashed).




(b) Roll motions along the path found by SFP




(c) Roll motions along the straight line path

**Figure 2** **Results for $\theta_s = 20°$.**

Figure 2 illustrates our findings for the instance with $\theta_s = 20°$: Figure 2(a) plots the path found by the SFP algorithm and the direct $s$-to-$f$ path, while Figures 2(b) and 2(c) plot the roll motions experienced by the containership following these two paths, respectively.

## 5. Conclusion and Future Work

We presented a parameter-free variation of the Sampled Fictitious Play algorithm for solving deterministic Dynamic Programming problems with finite state and action spaces, including general shortest path problems in acyclic networks. The algorithm eliminates the need for parameters present in the preceding versions of the algorithm that require fine-tuning and further customiza-

tion. Instead, the random tie-braking procedure of the parameter-free SFP algorithm provides a natural and sufficient randomness to guarantee the discovery of an optimal path in a finite number of iterations. Our algorithm is well-suited to very large-scale problems with computationally expensive action (arc) cost evaluations, as demonstrated by our numerical experiments on the minimum-motion vessel path finding problem, where it found high quality solutions in just a few iterations.

We also presented a novel approach to roll-minimizing path finding in marine navigation by presenting a DP model that (1) integrates dynamic real-time information about the vessel's surroundings, (2) includes minimum turning radius constraint, and (3) captures motion versus travel-time trade off for the considered path. Our numerical results demonstrate that the presented model and algorithm can significantly reduce vessel root-mean-squared motion (roll) on turbulent routes with an overall acceptable increase in total travel time.

In our future work, we plan to further study the behavior and properties of the presented parameter-free SFP algorithm. More specifically, we are interested to characterize a class of problems for which our algorithm performs especially well and develop a tight bound on the convergence rate. We believe the parameter-free SFP is an excellent solution method for a wide class of problems, and further investigation is necessary to fully quantify its benefits.

## Acknowledgments

The authors would like to thank Okey Nwogu and Fernando Tavares for their assistance with implementation and numerical results. This work was supported in part by the Office of Naval Research through the Multidisciplinary University Research Initiative (MURI) Optimum Vessel Performance in Evolving Nonlinear Wave Fields grant (N00014-05-1-0537).

## References

Alden, J. M., R. L. Smith. 1992. Rolling horizon procedures in nonhomogeneous Markov decision processes. *Oper. Res.* **40**(suppl. 2) S183–S194.

Boissonnat, J.-D., A. Cérézo, J. Leblond. 1994. Shortest paths of bounded curvature in the plane. *J. of Intelligent and Robotic Systems* **11**(1-2) 5–20.

Brown, G. W. 1951. Iterative solution of games by fictitious play. T. C. Koopmans, ed., *Activity Analysis of Production and Allocation*, chap. XXIV. Wiley, New York, 374–376.

Cheng, S.-F., M. A. Epelman, R. L. Smith. 2006. CoSIGN: A parallel algorithm for coordinated traffic signal control. *IEEE Trans. on Intelligent Transportation Systems* **7**(4) 551–564.

Denardo, E. V. 2003. *Dynamic programming*. Dover Publications Inc., Mineola, NY.

Dijkstra, E. W. 1959. A note on two problems in connexion with graphs. *Numerische Mathematik* **1**(1) 269–271.

Dolinskaya, I. S. 2012. Optimal path finding in direction, location and time dependent environments. *Nav. Res. Logist. Quart.* **59**(5) 325–339.

Dubins, L. E. 1957. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *Amer. J. Math.* **79** 497–516.

Epelman, M. A., A. Ghate, R. L. Smith. 2011. Sampled fictitious play for approximate dynamic programming. *Comput. and Oper. Res.* **36**(12) 1705–1718.

Fang, M.-C., J.-H. Luo. 2007. On the track keeping and roll reduction of the ship in random waves using different sliding mode controllers. *Ocean Engineering* **34** 479–488.

Faulkner, F. D. 1963a. A general numerical method for determining optimum ship routes. *Navigation* **10**(2) 143–148.

Faulkner, F. D. 1963b. Numerical methods for determining optimum ship routes. *Navigation* **10**(4) 351–367.

Fossen, T. I. 1994. *Guidance and Control of Ocean Vehicles*. Wiley & Sons, England.

Garcia, A., S. D. Patek, K. Sinha. 2007. A decentralized approach to discrete optimization via simulation: Application to network flow. *Oper. Res.* **55**(4) 717–732.

Garcia, A., D. Reaume, R. L. Smith. 2000. Fictitious play for finding system optimal routing in dynamic traffic networks. *Transportation Res. B* **34**(2) 147–156.

Ghate, A., S.-F. Cheng, S. Baumert, D. Reaume, D. Sharma, R. L. Smith. 2014. Sampled fictitious play for multi-action stochastic dynamic programs. *IIE Transactions* **46**(7) 742–756.

Johnson, J. T., R. J. Burkholder, J. V. Toporkov, D. R. Lyzenga, W. J. Plant. 2009. A numerical study of the retrieval of sea surface height profiles from low grazing angle radar data. *IEEE Trans. on Geoscience and Remote Sensing* **47**(6) 1641–1650.

Kimball, J. C., H. Story. 1998. Fermat's principle, Huygens' principle, Hamilton's optics and sailing strategy. *European J. of Physics* **19** 15–24.

Lambert, III, T. J., M. A. Epelman, R. L. Smith. 2005. A fictitious play approach to large-scale optimization. *Oper. Res.* **53**(3) 477–489.

Lanthier, M., A. Maheshwari, J.-R. Sack. 1999. Shortest anisotropic paths on terrains. *Automata, languages and programming (Prague, 1999)*, *Lecture Notes in Comput. Sci.*, vol. 1644. Springer, Berlin, 524–533.

Lee, C.-Y., E. V. Denardo. 1986. Rolling planning horizons: Error bounds for the dynamic lot size model. *Math. Oper. Res.* **11**(3) 423–432.

Mitchell, J. S. B. 2000. Geometric shortest paths and network optimization. *Handbook of computational geometry*. North-Holland, Amsterdam, 633–701.

Monderer, D., L. S. Shapley. 1996. Fictitious play property for games with identical interests. *J. Econ. Theory* **68**(14) 258–265.

Nilim, A., L. El Ghaoui. 2004. Algorithms for air traffic flow management under stochastic environments. *Proceedings of American Control Conference*, vol. 4. 3429–3434.

Nilim, A., L. El Ghaoui, M. Hansen, V. Duong. 2001. Trajectory-based air traffic management (TB-ATM) under weather uncertainty. *Proceedings of the Fourth International Air Traffic Management R&D Seminar ATM*. Santa Fe, New Mexico.

Nwogu, O. G. 2009. Interaction of finite-amplitude waves with vertically-sheared current fields. *J. of Fluid Mechanics* **627** 179–213.

Nwogu, O. G., D. R. Lyzenga. 2010. Surface wavefield estimation from coherent marine radars. *IEEE Geoscience and Remote Sensing Letters* **7**(4) 631–635.

Ovacikt, I. M., R. Uzsoy. 1994. Rolling horizon algorithms for a single-machine dynamic scheduling problem with sequence-dependent setup times. *International J. of Production Research* **32**(6) 1243–1263.

Papadakis, N. A., A. N. Perakis. 1990. Deterministic minimal time vessel routing. *Oper. Res.* **38**(3) 426–438.

Pearl, J. 1984. *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley, MA.

Perakis, A. N., N. A. Papadakis. 1988. New models for minimal time ship weather routing. *Society of Naval Architecture and Marine Engineering Transactions* **96** 247–269.

Perakis, A. N., N. A. Papadakis. 1989. Minimal time vessel routing in a time-dependent environment. *Transportation Sci.* **23**(4) 266–276.

Plant, W. J., W. C. Keller, K. Hayes. 2005. Simultaneous measurement of ocean winds and waves with an airborne coherent real aperture radar. *J. of Atmospheric and Oceanic Technology — Special Section* **22** 832–846.

Robinson, J. 1951. An iterative method of solving a game. *Ann. Math.* **54**(2) 296–301.

Ross, S. M. 1995. *Stochastic Processes*. 2nd ed. Wiley.

Rowe, N. C. 1997. Obtaining optimal mobile-robot paths with nonsmooth anisotropic cost functions using qualitative-state reasoning. *The International J. of Robotics Research* **16**(3) 375–399.

Rowe, N. C., R. S. Ross. 1990. Optimal grid-free path planning across arbitrarily contoured terrain with anisotropic friction and gravity effects. *IEEE Trans. on Robotics and Automation* **6**(5) 540–553.

Sisikoglu, E. 2009. Distributed algorithms based on fictitious play for near optimal sequential decision making. Ph.D. thesis, The University of Michigan, Ann Arbor, MI.

Sisikoglu, E., M. A. Epelman, R. L. Smith. 2011. A sampled fictitious play based learning algorithm for infinite horizon markov decision processes. S. Jain, R. R. Creasey, J. Himmelspach, K. P. White, M. Fu, eds., *Proceedings of the 2011 Winter Simulation Conference*. 4086–4097.

Smith, T. C., W. L. Thomas, III. 1990. A survey of ship motion reduction devices. Departmental Report SHD-1338-01, David Taylor Research Center, Bethesda, Maryland 20084-5000.

Sun, Z., J. H. Rief. 2005. On finding energy-minimizing paths on terrains. *IEEE Trans. on Robotics* **21**(1) 102–114.

Sussmann, H. J., G. Tang. 1991. Shortest path for the Reeds-Shepp car: A worked out example of the use of geometric techniques in nonlinear optimal control. Tech. Rep. SYCON-91-10, Rutgers Center for Systems and Control.

Treakle, III, T. W., D. T. Mook, S. I. Liapis, A. H. Nayfeh. 2000. A time-domain method to evaluate the use of moving weights to reduce the roll motion of a ship. *Ocean Engineering* **27**(12) 1321–1343.

Zermelo, E. 1931. Über das navigationsproblem bei ruhender oder veränderlicher windverteilung. *Zeitschrift für Angewandte Mathematik und Mechanik* **11**(2) 114–124.

Zhang, X., P. Bandyk, R. F. Beck. 2010. Seakeeping computations using double-body basis flows. *Applied Ocean Research* **32**(4) 471–482.

Zwillinger, D., S. Kokoska. 1999. *CRC Standard Probability and Statistics Tables and Formulae*. CRC Press.