Rapport de PIR

Towards optimal path planning for autonomous sailboats using Upper Confidence Bounds for Trees (UCT)

by Paul Barde

ISAE Sup'Aéro
Projet Innovation Recherche - 2A
Yves Briere, Caroline P. Carvalho Chanel, Emmanuel Rachelson
June 6, 2023

Contents

Introduction

This report aims to be the logbook of my PIR. It will detail the processes of the research and justify the choices that were made. Augmented with the corresponding `Python` code it should permit to quickly take charge of the project.

1 Weather Forecasts

1.1 Generalities

We chose the open-source Global Forecast System (GFS) model. It exists several variants, hence we selected the most convenient one for our application. Because the boat is slow, we picked the one with the highest spatial resolution. Therefore, we have daily 0.25° spatial resolution outputs providing an eight days forecast. The time step of the forecasts is 3 hours. These grids are interpolated to get 0.05° and 1 hour resolution. A python class[1] as been implemented in order to download and process the data. It facilitates the daily connection to the GFS servers and downloads only the 10m above water surface wind on the selected longitudes and latitudes. This way we do not deal with the daily 400GB output. From this data we can extract the predicted wind (direction and speed) for given latitude, longitude and time. Figure 1.1 displays the predicted wind conditions over the Atlantic ocean.

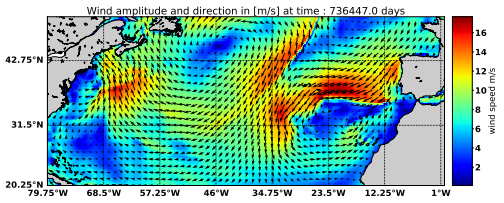


Figure 1.1: Predicted wind conditions over Atlantic ocean, time starts at year 0

1.2 Modeling the forecast uncertainty

We used the GFS to have an estimate of the encountered wind on the boat's track. This data is provided by NOAA server[2]. This model covers a 10 days horizon but do not give an measure of the uncertainty nor expresses how this uncertainty increases for long term estimations. In order to model degradation of the model precision with time we used the Global Ensemble Forecast System (GEFS) which is also provided by NOAA server[3]. GEFS runs several models from perturbed initial conditions and computes the ensemble spread of the models [1]. As shown in figure 1.2 the wind speed esemble spread increases as we look for longer term forecasts. This applies also for wind direction. Hence, in this work we assumed that GFS provides a good estimate of the forecast mean value and that GEFS provides a good measure of the uncertainty. Finally, it can be seen that beyond three days the forecast become rather unreliable.

[1] `https://github.com/PBarde/IBoat-Project`
[2] `http://nomads.ncep.noaa.gov:9090/dods/gfs_0p25/gfs`
[3] `http://nomads.ncep.noaa.gov:9090/dods/gens_bc/gens`

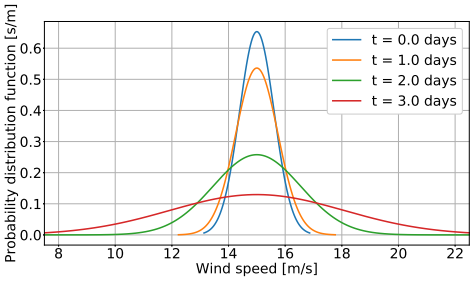


Figure 1.2: Gaussian wind speed distribution around GFS mean given GEFS standard deviation

The characteristic of GFS and GEFS outputs are summarized in the following table.

	Δt (hours)	Δx (°)	T (days)	**Coverage** (-)	**Data** (-)
GFS	3	0.25	10	Worldwide	10 m above surf. wind
GEFS	6	1	15.5	Worldwide	10 m above surf. wind std.

Table 1.1: GFS and GEFS characteristics summary

1.3 Possible improvements

A way to improve the use of the weather forecast would be to use, instead of a linear interpolation in space and time, a method that conserve the meteorological properties of the wind to interpolate the forecast. Same goes for the use of the GEFS data, we could think of a more meteorological way to use the standard deviation of the models.

2 Boat's and current's dynamics

Here we describe how we modeled the behaviour of the boat in order to create a simulator.

2.1 Estimation of the boat speed from a Velocity Prediction Program (VPP)

The Velocity Prediction Program Based on [2], [3] and [4], we used a Velocity Prediction Program to determine the velocity of the boat for a given wind magnitude and orientation. To this end, we made use of the Matlab script `gvpp` provided by Gianluca Meneghello[4] in which we inputted the geometrical properties of the boat that we got from blue-prints. This method is based on the work of [5] and relies on the equilibrium between the aerodynamic and hydrodynamic forces. This forces mostly derive from empirical considerations. The VPP assumes that all the controllable factors (such as sail configuration) are set so as to maximize the ship speed under the given wind conditions. Hence, from the wind 10m above the water surface, it outputs the velocity V of the boat in the direction of heading (thus neglecting the drifting of the vessel). Figure 2.1 shows the output of the VPP.

[4] `https://sourceforge.net/projects/gvpp/`

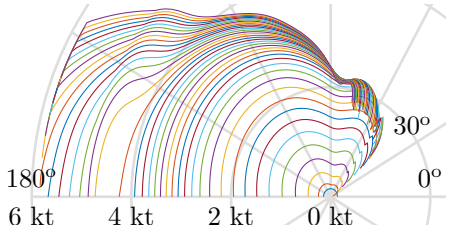


Figure 2.1: Polar plot of boat's velocity in knots wrt. the true wind angle for true wind speeds from 0.1 to 19.1 m/s (increment of 0.5 m/s)

2.2 Making use of the VPP

Speeding up computation To speed up computations and gain flexibility, these VPP results are fitted with a two-dimensional five-order polynomial. The relative error between the fit and the data is less than 3.5%. This polynomial fit is displayed in figure 2.2, where the surface is the fit and the dot the VPP outputs.

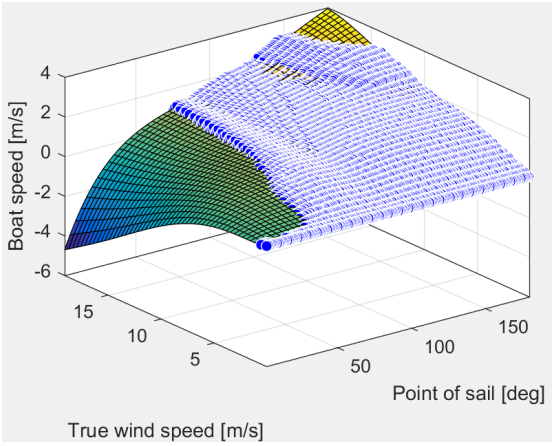


Figure 2.2: Velocity polar from VPP fitted with a fifth order two variable polynomial

Simplifying the physic In reality, a boat can not sail for all possible point of sail α (the angle between heading and wind direction). There is two no go zones : one for $0° <$ point of sail $< \alpha_{min} = 35°$ (the boat cannot sail directly into the wind) and another for $160° <$ point of sail $< \alpha_{max} = 180°$. Tacking trajectories are required to go toward this directions. However, in our case we will consider long period of time with constant point of sail, hence, the time lost during a tacking maneuver is negligible. Following this logic we will assume that the boat can sail toward the no-go zones but with a reduced speed. As a matter of fact, we will consider that the speed is equal to the Velocity Made Good (VMG) that is the actual velocity projected toward the goal. In addition, the VPP as been computed up to winds of magnitude 19 m/s, however it can be seen from figure 2.1 that, for high wind magnitudes, the curves tend to collapse. Thus we assume that the boat behaves for winds above 19 m/s like it behaves for a wind of 19 m/s. The final polar velocity that we obtain is displayed in the following figure.

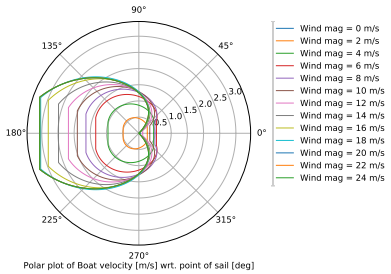


Figure 2.3: Modified boat's velocity polar

The resulting velocity polar is coherent with the boat's dynamics. The velocity is maximal for running points of sail but diminishes if the point of sail is greater than α_{max}. Finally, the vessel cannot sail into the wind if this one is too strong.

2.3 Introducing a stochastic behaviour

For now the boat's dynamics are deterministic, a given wind and point of sail produce a unique boat velocity. This is not feasible since we did not fully model the boat or the sea state. It is known for example that wind gusts or waves drastically impact the boats performances. To account for this physic we did not modelled we add noise to the velocity obtained by the speed polar. From the deterministic boat velocity $\mathcal{V}$ and the boat heading γ we can compute the boat velocity toward East U and toward North V as follow :

$$U = \mathcal{V}\sin(\gamma), \qquad V = \mathcal{V}\cos(\gamma) \tag{2.1}$$

Then we define a variance σ^2 proportional to the boat deterministic speed :

$$\sigma^2 = \kappa\mathcal{V} \tag{2.2}$$

where κ is an arbitrary constant set to 20%. Finally, the stochastic velocities u' and v' are picked from a normal distribution :

$$u' \sim \mathcal{N}(U, \sigma^2), \qquad v' \sim \mathcal{N}(V, \sigma^2) \tag{2.3}$$

2.4 Modeling the current

The VPP model considers static water surface, hence to get the actual velocity of the boat we must add the water surface velocity. The surface current can be approximated to be K=3% of the 10m above water surface wind[6]. Here we used a simplified current model that only takes into consideration the wind induced surface current. We do not account for under water, climatological currents. Hence for arbitrary wind velocities u_w and v_w, the total boat velocity is given by :

$$u = u' + Ku_w, \qquad v = v' + Kv_w \tag{2.4}$$

2.5 Possible improvement

On the VPP As said previously, the VPP does not consider the drifting of the vessel but provides the heel angle ϕ. Hence, the leeway angle θ (quantifying the drifting) can be estimated as follow[5] :

$$\theta = \phi\frac{K}{\mathcal{V}^2} \tag{2.5}$$

[5]https://www.boatdesign.net/threads/thoughts-of-resistance-and-leeway.51130/page-2

Where K is a boat-specific value (around 10) if speed is in knots and the leeway angle is in degrees.

On the current modelling We could make use of current forecasts or consider under-surface and surface current interaction.

On uncertainty We could use sea state and wind gust forecast to define the uncertainty magnitude.

3 Implementation of the simulator

In this section we describe how we implemented the simulator that for a given action (heading/bearing γ) and a given boat state $\mathbf{s}$ generates a new state $\mathbf{s}'$.

3.1 The different blocks of the simulator

Boat states : a boat state is defined by a date t, a latitude φ and a longitude λ :

$$\mathbf{s} = (t, \varphi, \lambda)^{\mathrm{T}} \tag{3.1}$$

Wind conditions : the GFS and GEFS files are linearly interpolated over the domain. Hence, we get two functions $\mathbf{W}_{avg}(t, \varphi, \lambda)$ and $\mathbf{W}_{spr}(t, \varphi, \lambda)$ that outputs the mean wind and the ensemble spread respectively. Thus, for a given state $\mathbf{s}$, we can build the observed wind velocities u'_w and v'_w as follow :

$$\begin{aligned} \mathbf{W}_{avg}(\mathbf{s}) = (U_w, V_w)^{\mathrm{T}}, \qquad \mathbf{W}_{spr}(\mathbf{s}) = (\sigma_u, \sigma_v)^{\mathrm{T}} \\ (u'_w, v'_w) \sim (\mathcal{N}(U_w, \sigma_u^2), \mathcal{N}(V_w, \sigma_w^2))^{\mathrm{T}} \end{aligned} \tag{3.2}$$

Boat dynamic : From last paragraph we get a stochastic wind condition for a given state. Coupling this wind condition with the boat and current model of section 2 we then get a noisy boat velocity for a given boat heading. We consider this velocity constant and trake its modulus and angles with respect to the true North. Then, given a time-step we get a covered distance ΔL toward a direction ψ (that may differ from the boat heading due to the noisy wind and dynamic and the surface current), this step is thus the simplest **integration** possible.

Spherical geometry : Since the vessel is sailing on the spherical Earth surface we use the Geodesic formulae to compute distances and headings. The reader is referred to `http://www.movable-type.co.uk/scripts/latlong.html` for more informations on the subject. These formulae make the link between two points on the Earth surface $x_1 = (\varphi_1, \lambda_1)$, $x_2 = (\varphi_2, \lambda_2)$ and the distance D between them as well as the bearing θ from x_1 to x_2 :

$$(x_1, x_2) \to (D, \theta), \qquad (x_1, D, \theta) \to x_2 \tag{3.3}$$

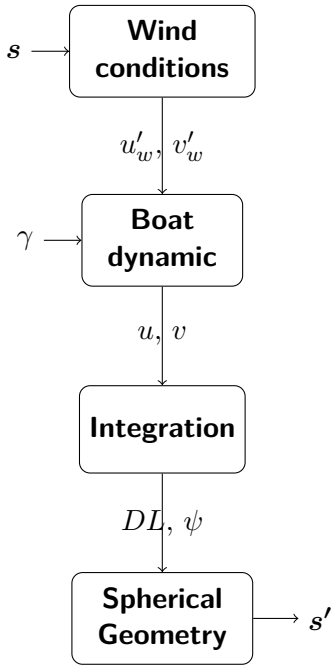


Figure 3.1: Simulator's steps

3.2 Testing the Simulator

To test the simulator we launch 30 boats from the same departure state $\boldsymbol{s} = (0, 47.5, 356.5)^{\mathrm{T}}$ with a constant heading of 225° for a 6 days navigation. They all sail with the same GFS and GEFS data. To represent the trajectories we chose an Azimuthal Equidistant Projection centred on the departure state. This means that the distances from the departures are preserved and that all points that lie on a circle around the departure are equidistant on the surface of the earth.

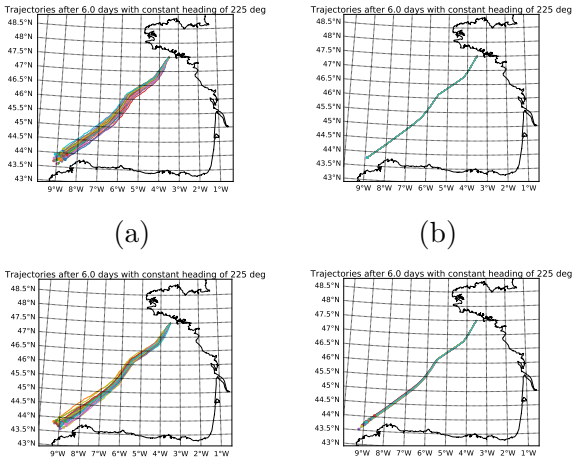


Figure 3.2: 30 boats simulations with (a) stochastic wind and dynamics, (b) deterministic wind and dynamics, (c) deterministic wind and stochastic dynamics, (d) stochastic wind and deterministic dynamics

From figure 3.2 we can conclude that the simulator is coherent. Stochastic has to

effects : it spreads the boats' trajectory and it varies the distance between the arrival points and the departure. Without the stochastic modelling of the wind and the boat's dynamic all trajectories are identical. It should be noted that the stochastic dynamic of the boat impacts more the spreading of the trajectory than the covered distance. The opposite occurs regarding the stochastic wind, the trajectories are close to one another but vary in length. This is due to the fact that the wind affects only the direction of motion of the boat through surface current (with a coefficient of 3%) but plays a major role in the speed of the vessel. Whereas the boat's dynamic affect boat its direction of motion and its velocity.

If we now zoom on a specific trajectory under stochastic wind and dynamic we get the following figure.

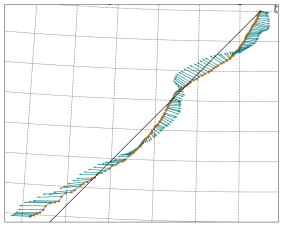


Figure 3.3: One boat trajectory under uncertain wind and boat's dynamic

A yellow dot represents a state, the solid black line represents the constant 225° heading and the blue arrows are a quiver of the mean wind exerted on the boat for the corresponding state. This trajectory is coherent, we notice that the boat is deported by the wind along form the constant heading line. Finally the boat sails faster (dots are more spaced out) when the wind come from behind (running point of sails).

4 Incompatibility with Markov Decision Process work-frame

Like it has been discussed in the literature review, Markov Decision Processes (MDP) would have been useful work-frame to derive and solve the problem. Unfortunately, the stochastic modelling we derived is not handled by this formalism. Indeed, MDP requires explicit probabilistic transition function of the form :

$$P_a(\boldsymbol{s}, \boldsymbol{s'}) = \mathcal{P}(\boldsymbol{s}_{t+1} = \boldsymbol{s'} | \boldsymbol{s}_t = \boldsymbol{s}, a_t = a) \tag{4.1}$$

which is the probability that action a in state $\boldsymbol{s}$ at time t will lead to state $\boldsymbol{s'}$ at time $t+1$.

In our case we have a stochastic generator (the simulator) that outputs a state $\boldsymbol{s'}$ from a state $\boldsymbol{s}$ and an action a. Hence we will explore another frame-work : the Monte-Carlo Tree Search (MCTS) and the Upper Bound Confidence for Tree (UBT).

5 Monte Carlo Tree Search (MCTS) and Upper Confidence Bound for Tree (UCT) work-frame

5.1 Monte Carlo Tree Search :

The definition of the MCTS given by [7] is the following :

> Monte Carlo Tree Search is a method for finding optimal decision in a given domain by taking random samples in the decision space and building a search tree according to the results.

In brief, for large decision space, where an exhaustive search is not practicable, we pick random samples to explore the decision space. In order to estimate the value of the picked decision we run a simulation the Default Policy that can be a random or statistically biased sequence of action applied to the state resulting from the picked decision until a terminal condition is reached. The terminal state reached after the Default Policy is used to evaluate a reward that is incorporated in the value of the picked state. A tree is build in an asymmetric and incremental manner in order to explore the decision space. Each node of the tree represent a sequence of decision and posses a value, function of the reward obtained from the Default Policy.

5.2 The Upper Confidence Bounds for Trees :

In order to built the tree in an asymmetrical manner, i.e. to explore only promising decisions and to avoid loosing time and computational power with inefficient decision, we need a clever Tree Policy. A Tree Policy is a method that computes the values of tree nodes and that select the more promising node. Thus, at each iteration of the search the Tree Policy selects the node to expand and thus dictates how the decision space is explored. In this work we will use the most popular algorithm in the MCTS family : The Upper Confidence Bounds for Trees. Stating at the root node of the tree, we select the child node from its UCT value. If this child node is non-terminal and not fully expanded, we expand it. Otherwise, we look at the UCT values of its children. The UCT value of a child node j is given by :

$$UCT = \bar{X_j} + 2C_p\sqrt{\frac{2\log(n)}{n_j}} \tag{5.1}$$

Where $\bar{X_j}$ is the mean reward of all the decisions sequences that passed through node j. n_j is the number of decisions sequences that involved the child node j, whereas n is the number of decisions sequences that involved the parent node. C_p is a positive coefficient called the exploration coefficient. The left hand term represents the value obtained from a node, it is the exploitation part. And the right side represents the exploration term. Indeed, if a child node is not explored, the right hand side increases as $\sqrt{\log(n)}$ with n the number of time the parent node has been explored. Thus, the UCT of the unexplored node will eventually grow and have the Tree Policy select it. A visual representation of a step of the MCTS-UCT algorithm is shown in figure 5.1.

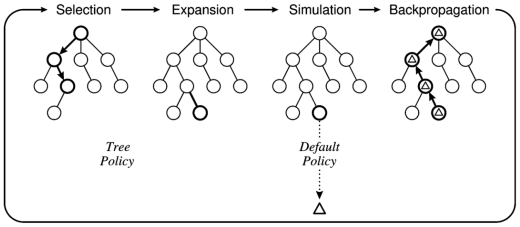


Figure 5.1: One step of the MCTS-UCT algorithm generating a reward Δ, extracted from [7]

6 Our design of MCTS-UCT

6.1 Defining the objects :

The available actions : The purpose of the tree-search is to find the optimal sequence of actions to go from a departure point to a destination point. The optimal sequence of action is the one that minimizes in average the date of arrival. The actions are the boat's headings and thus range from 0° to 360°. In order to simplify the problem and reduce the branching factor of the tree we will use only a discrete set of headings with a step of 45°. The work done here could be generalized to continuous sets of actions using Progressive Widening as it has been proposed by [8].

A node : In this architecture only the root node represents a state. All other nodes represent a sequence of action executed from the root node. Thus the attributes of a node are :

- *state :* Only for the root node. The state of the root node is the initial state corresponding to $t = 0$ and the departure point.
- *parent :* A reference toward the parent node.
- *origin :* The action taken from the parent node to expand it and create the node.
- *children :* A list of references toward the children nodes.
- *actions :* A shuffled list of action available. Every time the node is expanded using one of the actions, this action is not available any more.
- R : Sum of all the reward that passed through the node.
- n : Number of times the node as been involved in a decision sequence.
- *depth :* Corresponds to the number of action taken from the root node to arrive to the node.

A tree : A tree is the object that collects the created nodes and posses the methods implementing the MCTS.

- *rootNode :* A reference pointing toward the root node object.
- *ite :* The number of nodes that have been created.
- *budget :* The total number of node that will be created during the search.
- *Simulator :* A reference toward a simulator object like the one described in 3.
- *destination :* The latitude and longitude of the destination provided by the initialization of the tree.
- *TimeMax :* The temporal horizon on which the search is carried out. Also the horizon of the simulator.
- *TimeMin :* An arbitrary estimate of the time required by the boat to go from the departure to the destination. Is provided by the tree initialization.
- *depth :* Total depth of the tree.

- *nodes :* A list of reference toward the created nodes to keep track of the chronology of the tree's growth.
- *rewards :* A list of all the reward that were obtained during the search.

6.2 Defining the processes

Initialization : This steps converts a mission heading and time horizon to a destination point and a estimated time of arrival. It can be divided in several sub-steps :

- *a) Estimating a covered distance :* To estimate, given the wind condition, the distance covered by the boat we simulate $N_b = 50$ boats over the whole time horizon and sailing straight toward the mission heading.
- *b) Computing the corresponding destination :* Due to the drifting and the discrete headings that are available, the N_b boats do not actually sail with the correct heading. Therefore, we take a fraction c_d of the estimated distance and we project it toward the actual heading. This gives us the destination.
- *c) Estimating a date of arrival :* Then, to get an estimate of the date at which the boat will arrive to destination, we simulate N_b other boat. These boats modify their heading to match the destination bearing, hence they eventually get to the destination. The fastest boat is taken as a reference.

We show the trajectories resulting from the initialisation for a time horizon of 2 days and a mission heading of 235° in figure 6.1.

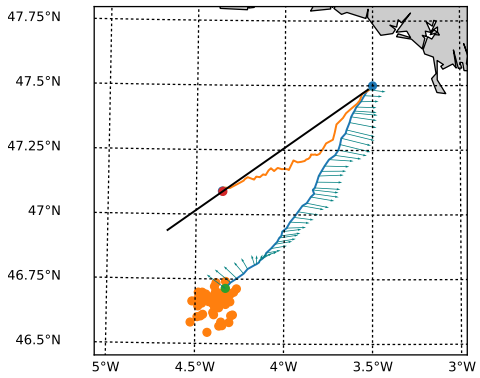


Figure 6.1: Initializations trajectories

The blue point is the departure. All the yellow points are the arrivals of the N_b boats simulated in *a)*. The blue trajectory is one of them. We see that the boat drifted toward the East due to the lateral wind and the fact that it did not adapt its heading. The red point is the destination obtained from *b)*. To compute it we took 3/4 of the minimal distance covered in *a)*. The yellow trajectory is one of the boat sent to estimate the time of arrival. Finally the the dark line indicates the mission heading.

The Default Policy : The Default Policy is going straight toward the destination. At each time step we compute the destination bearing to adapt the boat heading and reach the destination. A Default Policy from the departure will look like the

yellow trajectory in figure 6.1. For an arbitrary node we first need to determine the corresponding state. We can estimate it by starting from the root node and executing the sequence of actions represented by the node. Then the reward is computed as :

$$R = \exp\left(\frac{TimeMax \times c - date}{TimeMax \times c - TimeMin}\right) \tag{6.1}$$

Where $TimeMax$ is the search time horizon, *date* is the date of arrival of the boat following the default policy, $TimeMin$ is obtained from step *c)* on the initialization and c is a constant to avoid a division by zero. In our case $c = 1.001$. If the boat cannot reached the destination during the time horizon following the Default Policy the reward is set to zero. Thus the reward is positive but can be greater than 1. We used an exponential reward instead of a linear one to encourage and have a better distinction between the solutions that are near the best one that has been found during the initialisation.

Terminal and fully expanded nodes : A terminal state is attained if it is at destination or if it reached the time horizon. A node is fully expanded if it has no more available actions.

6.3 The Algorithm

We implemented the algorithm number 2 presented in [7].

7 Results

In this section we will present the results we obtained. We will propose explanations for what is observed. The limitations of the work will be discussed and our model criticised. Possible hints for improvement are to be offered.

7.1 Discussion of the exploration coefficient

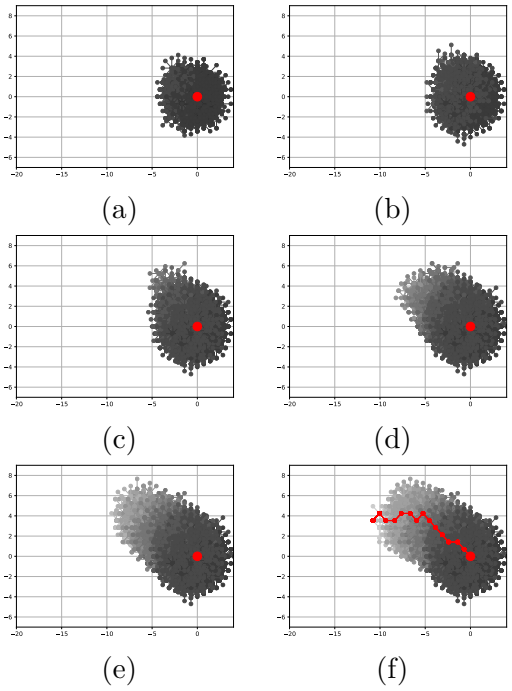
The exploration coefficient C_p in equation 5.1 is a critical parameter of the UCT. As a matter of fact, it drives how much the tree search is greedy and looks for states that give "immediate" reward. For example, with a low exploration coefficient, nodes that in the beginning did not generate high rewards are abandoned and the tree quickly grows in the direction of the nodes that provide value. This is positive in the fact that with such a logic we dig into strategies that pay and we do not loose time with the strategies that have low "short-term" reward. The danger is to miss an optimal solution that would require some initial investments (take some action that do not result in a short arrival time using the default policy) but that ends up giving optimal results in the long run.

There is therefore a trade-off to find between reaching an important depth for a given computational budget and not missing the optimal solution. [7] advise a value of $1/\sqrt{2}$ but indicates that C_p is greatly problem dependent.

Asymmetrical tree growth : We show here the tree growth for a search with the following parameters :

ite	$TimeMax$	time step	heading mission	c_d	C_p
15000	2 days	1 hour	235°	3/4	$1/\sqrt{2}$

Table 7.1: Parameters of simulation number 1



(a) (b) (c) (d) (e) (f)

7.2 Discussion on problem sizing and computing resources

7.3 Critic of model

Bias in the decision making because the "tree trajectory" have discrete action (heading step of 45°). MCTS is for on-line planning, not long-term off-line planning. We should investigate the depency with the time step. Because, we take the instantneous wind at the time and position at the begining of the time step and we assume that it is constant. we should take a time average at least, but this might be complicated. Rem: boat with 3h time step go less far than for 1h (it seems...). Also we should with large time step we have a large distance that is covered in one time step and we could keep turning around the destination and never be inside the arrival radius.

7.4 Algo improvement

Instead of having a estimate state method we should do this during the Tree policy !

Improvements to the model would be to use surface current, wind gusts and waves forecasts. To have a better matching between the model and the boat behaviour it would also be preferable to use a velocity polar based on experiments. However, we already have solid foundations on which we can build the MDP model. This being done, we will implement it and test it on actual weather forecasts.

For the wind A way to improve the use of the weather forecast would be to use, instead of a linear interpolation in space and time, a method that conserve the me-

teorological properties of the wind to interpolate the forecast. Same goes for the use of the GEFS data, we could think of a more meteorological way to use the standard deviation of the models.

for the current Here we used a simplified current model that only takes into consideration the wind induced surface current. We do not account for under water, climatological currents.

7.5 Possible improvement

On the VPP As said previously, the VPP does not consider the drifting of the vessel but provides the heel angle ϕ. Hence, the leeway angle θ (quantifying the drifting) can be estimated as follow[6] :

$$\theta = \phi \frac{K}{\mathcal{V}^2} \tag{7.1}$$

Where K is a boat-specific value (around 10) if speed is in knots and the leeway angle is in degrees.

On the current modelling We could make use of current forecasts or consider under-surface and surface current interaction.

On uncertainty We could use sea state and wind gust forecast to define the uncertainty magnitude.

7.6 Algo improvement

Instead of having a estimate state method we should do this during the Tree policy !

to evaluate the results, run longer tests, on different trees generated from the same algo. See the effect on the std of the arrival Date. also find a method to determine the optimal number of actions to consider.

Continuous actions The work done here could be generalized to continuous sets of actions using Progressive Widening as it has been proposed by [8]. Bias in the decision taking because the "tree trajectory" have discrete action (heading step of 45°). MCTS is for on-line planning, not long-term off-line planning. We should investigate the dependency with the time step. Because, we take the instantaneous wind at the time and position at the beginning of the time step and we assume that it is constant. we should take a time average at least, but this might be complicated. Rem: boat with 3h time step go less far than for 1h (it seems...). Also we should with large time step we have a large distance that is covered in one time step and we could keep turning around the destination and never be inside the arrival radius.

We think that this time gain would be much greater if we used a Continuous Default Policy during the tree search. This way we would have the performance gain of the optimal policy without the loss of discrete heading during Default Policy.

[6] https://www.boatdesign.net/threads/thoughts-of-resistance-and-leeway.51130/page-2

References

[1] Yuejian Zhu and Zoltan Toth. 2.2 ensemble based probabilistic forecast verification. 2008.

[2] C. Pêtrès, M. A. Romero-Ramirez, and F. Plumet. Reactive path planning for autonomous sailboat. In *2011 15th International Conference on Advanced Robotics (ICAR)*, pages 112–117, June 2011. doi: 10.1109/ICAR.2011.6088585.

[3] Daniel S. Ferguson and Pantelis Elinas. *A Markov Decision Process Model for Strategic Decision Making in Sailboat Racing*, pages 110–121. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. ISBN 978-3-642-21043-3. doi: 10.1007/978-3-642-21043-3_14. URL http://dx.doi.org/10.1007/978-3-642-21043-3_14.

[4] Andy Philpott and Andrew Mason. Optimising yacht routes under uncertainty. In *The 15th Cheasapeake Sailing Yacht Symposium*, 2001.

[5] David Martin and Robert F Beck. Pcsail, a velocity prediction program for a home computer. In *15th Chesapeake Sailing Yacht Symposium*, volume 100, 2001.

[6] Jan Erik Weber. Steady wind- and wave-induced currents in the open ocean. *Journal of Physical Oceanography*, 13(3):524–530, 1983. doi: 10.1175/1520-0485(1983)013<0524:SWAWIC>2.0.CO;2. URL http://dx.doi.org/10.1175/1520-0485(1983)013<0524:SWAWIC>2.0.CO;2.

[7] Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1):1–43, 2012.

[8] Adrien Couëtoux, Jean-Baptiste Hoock, Nataliya Sokolovska, Olivier Teytaud, and Nicolas Bonnard. *Continuous Upper Confidence Trees*, pages 433–445. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. ISBN 978-3-642-25566-3. doi: 10.1007/978-3-642-25566-3_32. URL http://dx.doi.org/10.1007/978-3-642-25566-3_32.