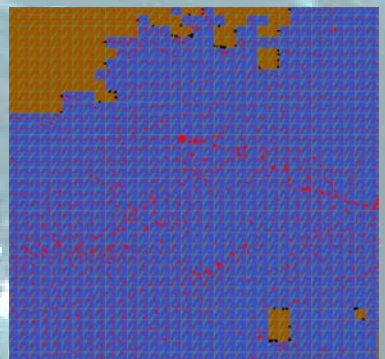
2nd Baltic Earth Conference “The Baltic Sea Region in transition”
Helsingør, Denmark. June 11-15, 2018

Last Reviewed: 13/6/2018 12:45

Modeling patchiness on the sea surface caused by the interplay of winds and currents in the Gulf of Finland

Giudici, A., **Kalda**, J., **Soomere**, T.



Presentation outline

- Introduction

- Overview of employed dataset

- Eulerian Simulation Model and Collision Detection

- Simulations Results

- Conclusions

Introduction - The Gulf of Finland

- Estuarine area which shows extremely complex dynamics
- High proportion of vertical motions of water masses [Leppäranta and Myrberg, 2009]
- Frequently hosts up- and downwelling phenomena [Lehmann and Myrberg, 2007]
- Extremely dense ship traffic throughout the year.

Introduction

- Patchiness in distributions of matter in the ocean is one of the oldest observed phenomena in oceanography

- Despite this, there is little consensus on its causes, and a general theory remains a distant goal ==> some studies even speculate whether such general theory actually exists

- Compressibility of surface velocity field is a convenient measure to indicate the possibility of gathering of floating litter into patches on the sea surface.

  - In some sense, it characterises the intensity of downwelling and upwelling motions near the surface, equivalently, whether water parcels can dive
  - It can be calculated from the 2D velocity pattern

- Its values are from 0 to 1. If it becomes large enough (>0.5 in the case of Kolmogorov turbulence), patch formation accelerates explosively [1]

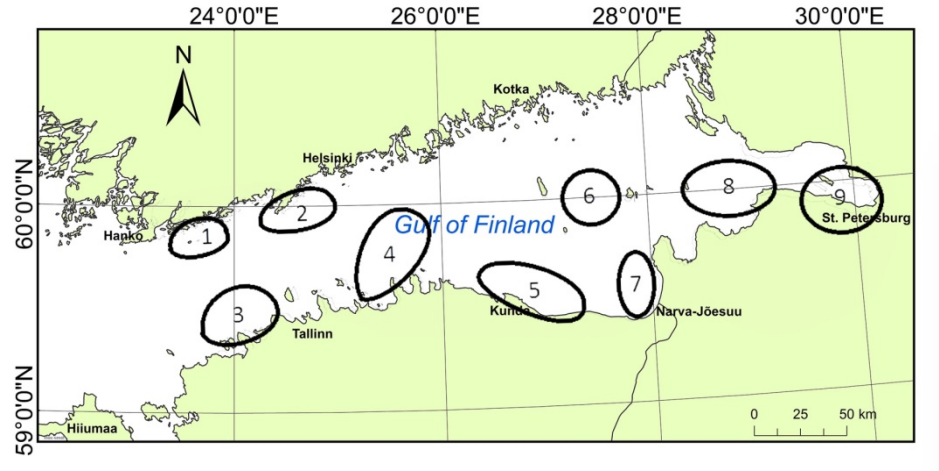
- In the case of quasi-2D flows (like marine flows in many cases), the surface flow compressibility is strongly suppressed.

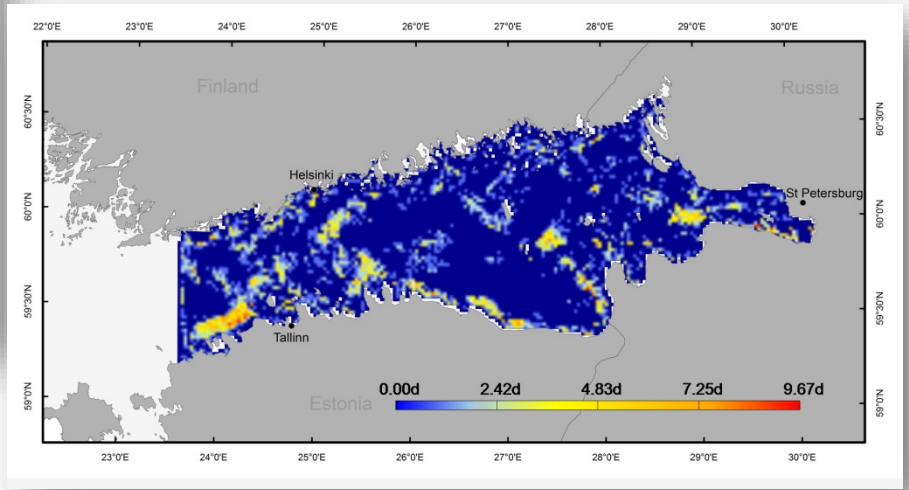
[1] Eckhardt, B. and Schumacher, J. “Turbolence and passive scalar transport in a free-slip surface”, Phys.Rev.E 64(1), 2001

Introduction

- Seabed features in specific regions (eg. Gulf of Finland) - may enhance patch generation by supporting up- and downwelling.

==> Previous work has highlighted regions where patches are likely formed in the Gulf of Finland by the field of currents only [2][3]





[1] Eckhardt, B. and Schumacher, J. “Turbolence and passive scalar transport in a free-slip surface”, Phys.Rev.E 64(1), 2001

[2]Kalda,J., Soomere, T. and Giudici, A. “On the finite-time compressibility of the surface currents in the Gulf of Finland, the Baltic Sea”,J.Marine Syst. 129, 2012

[3] Giudici,A., Soomere, T. “Finite-time compressibility as an agent of frequent spontaneous patch formation in the surface layer: a case study for the Gulf of Finland, the Baltic Sea”,Mar.Poll.Bull. 89, 2014

Introduction

- Other effects may also play a role in increasing effective compressibility, and therefore the clustering rate of floating litter:

==> Surface gravity waves: normally minor effect; they need to be strongly nonlinear in order to play a significant role [4][5][6]

==> Lateral stirring and mixing at sub-mesoscale, which were shown to have a direct effect on the patchiness of phytoplankton distribution [8]

==> Interplay of different phenomena, like waves and currents: an effective mechanism for providing patchiness of pollutants on the ocean surface [9]

[5] Denissenko, P., Falkovich and G., Lukashuck, S. “How waves affect the distribution of particles that float on a liquid surface”, Phys. Rev.Lett. 97(1), 2006
[6] Balk, A.M., Falkovich, G. and Stepanov, G. “Growth of density inhomogeneities in a flow of wave turbolence”,Phys.Rev.Lett. 92(1), 2004
[7] Vucelja, M., Falkovich, G. and Fouxon, I. “Clustering of matter in waves and currents
[8] Martin, A.P., “Phytoplankton patchiness: the role of lateral stirring and mixing”, Prog.Oceanog. 57(2), 2003
[9] Vucelja, M., Falkovich, G., Fouxon, I. “Clustering in waves and currents”, Phys.Rev.E Stat. Nonlin. Soft.Matter Phys., 75(6-2), 2007

Introduction

- A potential factor that may increase the effective compressibility is **the interplay of currents and winds**.

- Specifically, the effect of the differences in the wind drift speed of different patches as a function of their size.

- This mechanism is expected to be coupled with the turbulent mixing and the intrinsic compressibility of the flow field.

This work presents an attempt to quantify the effect of such differences in drift speeds by means of an Eulerian tracking model.

Introduction

- The main idea is to **create a model which is able to integrate current data with geostrophic wind information**.

- We know the areas where patch formation usually occurs. We want to know what happens outside of these areas and how we can quantify the process.

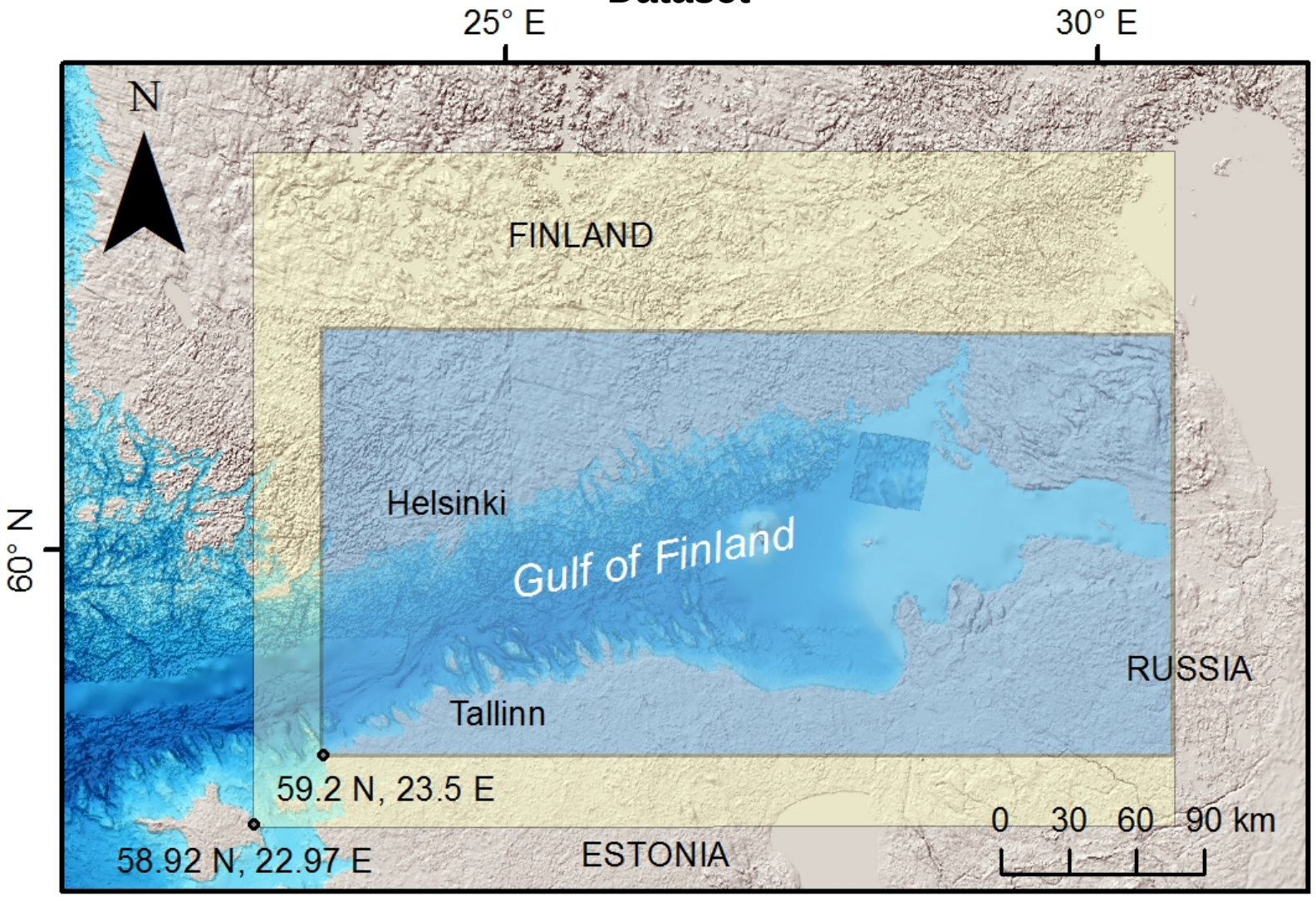
- The overlapping region of two data sets (current and wind), covering a large portion of the GoF, is gridded using a uniform 1NM grid.

- Simulated small patches of floating sticky litter (e.g. remnants of plastic bags, or small oil spills) are scattered uniformly across the Gulf

- The behavior of such patches is simulated in terms of their:
  => relocation under the effect of winds and currents
  => colliding trajectories

- We aim at describing the resulting distribution of patches, after they are left at sea for various periods of time.

Dataset



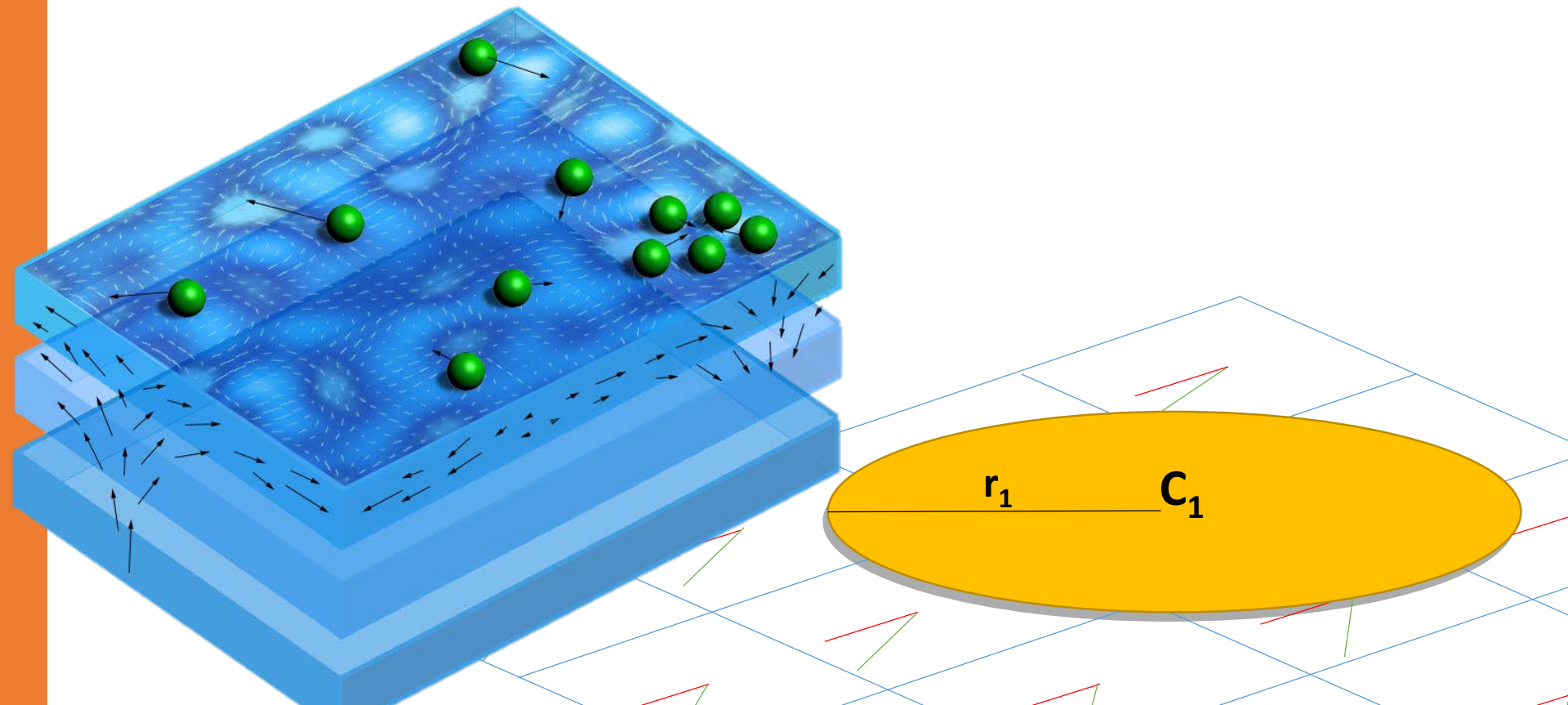
Currents data BL (59.2N, 23.5E) @ 0.03dx, 0.016dy → 202x100 grid 1NM res OAAS model

Winds data: BL (58.92, 22.97E) @ 0.19528dx 0.09754dy → 38x26 grid (flipped) calculated from air pressure gradients and rescaled at 10m height

Simulation Model - Patches

- A **patch** object is defined by the following properties:

  - **Center Coordinates**
  - **Radius** (Horizontal Section)
  - Vertical Section --> expressed as a function of horizontal section $V_{section} = H_{section}^{\alpha<1}$
  - '**Alive**' status --> whether the patch is at sea of hits the coast
  - **Lifetime** --> number of hours the patch is alive for



Simulation Model

- Patches are relocated, once per timestep (3 h) throughout the time window, under the combined effect of the interpolated Wind- and Current-fields
  ==> **Patch relocation model**
- At each timestep, collisions between patches are calculated
  ==> **Patch collision model**

- Patches properties (lifespan, Hsize, Vsize, coastal hit) are updated

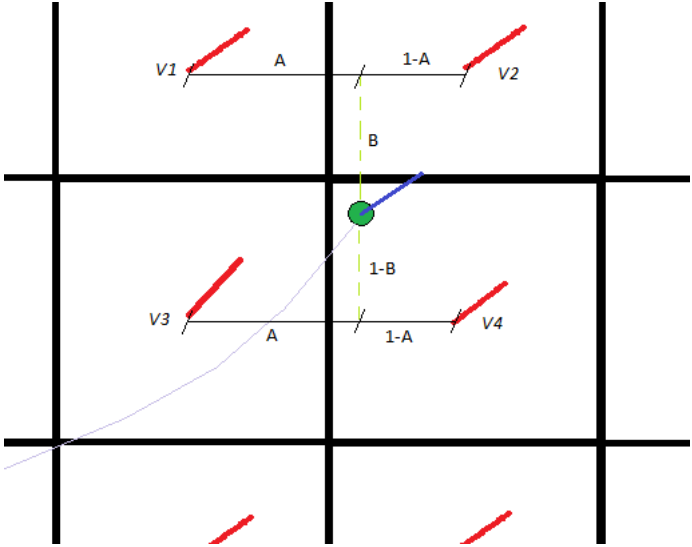
- Set of patches is updated based on collisions (total nr. of patches always diminishes)

- At the end of each time window, statistics are produced (min,max,avg Size, min, max, avg Life, size distribution, N/S coastal hits)

- The process is then repeated by sliding the time-window ahead by an interval (1 day)

- After the last time window is simulated, statistics are averaged over all the available realizations

Relocation Model



- The model has the following rules for relocating the generated patches:

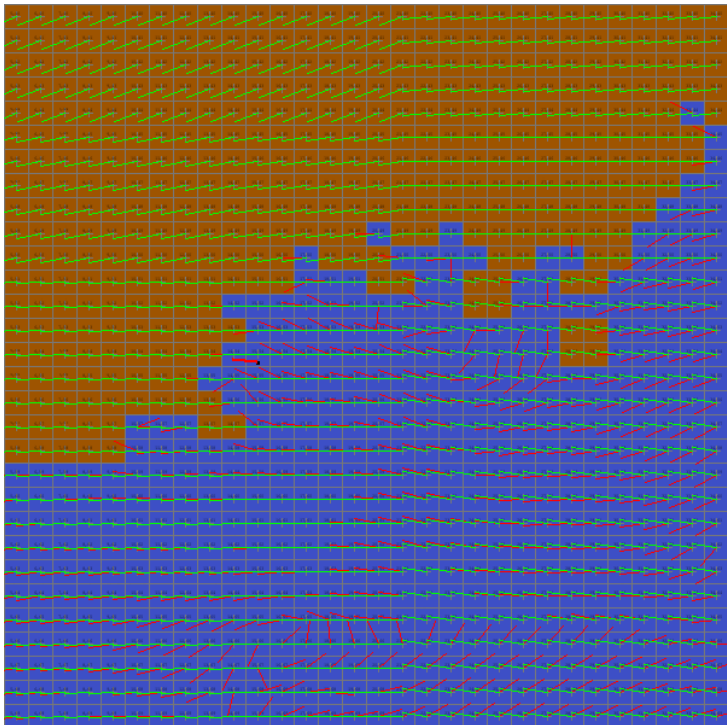
FOR EACH time step
  FOR EACH Patch
    GET Interpolated Current Vector **C**
    GET Interpolated Wind Vector **W**
    x (t + 1) = f (**x**(t), **W, C**)

$$V_{Res}{}^{T} = B * \left(A * V_1{}^{T} + (1-A) * V_2{}^{T}\right) + (1-B) * \left(A * V_3{}^{T} + (1-A) * V_4{}^{T}\right)$$

$$x(t+1) = x(t) + v_c \cdot \Delta t + \Delta M$$

$$\frac{\Delta M}{|\Delta M|} = \frac{v_c - v_w}{|v_c - v_w|}$$

$$\Delta M = v_\alpha \cdot \Delta t = \sqrt{\frac{H_{section}}{V_{section}}} \cdot \sqrt{\frac{\rho_{air}}{\rho_{water}}} \cdot (v_c - v_w) \cdot \Delta t$$



Time Subdivision Scheme

- A time interval of interest $\mathbf{t_D}$ is chosen (1 month)

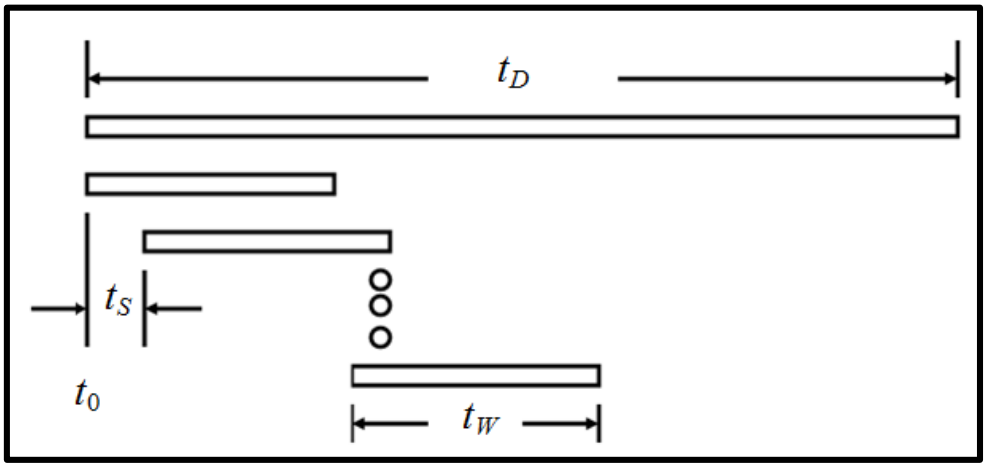
- The time interval is divided into time windows $\mathbf{t_W}$ (3 days)

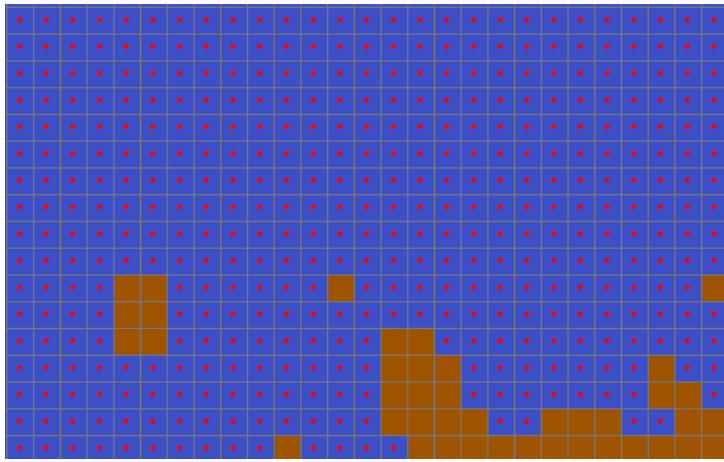
- At the beginning of each time window, a patch of initially small [relative to the grid cell size] size (50, 100, 150, 200, 250, 300 meters) is positioned in the center of each wet grid cell

==> A +/- 15% randomness is added to the initial size of the patches

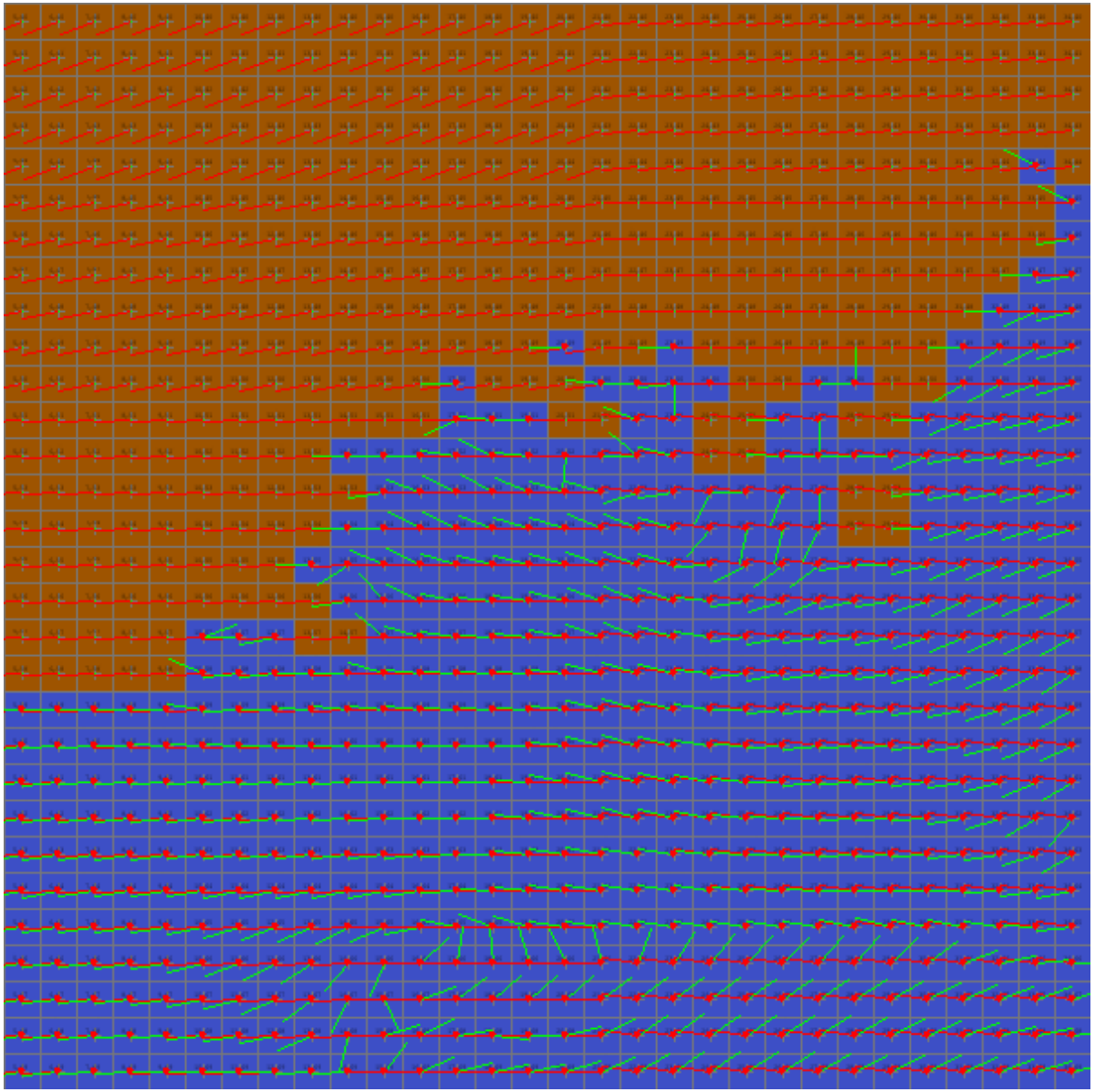
- The bahavior of each patch is simulated throughout the current time window

- At the end of each time window, patch position is reset and the initial time for the next simulation $\mathbf{T_0}$ is advanced by $\mathbf{T_s}$ (1 day)

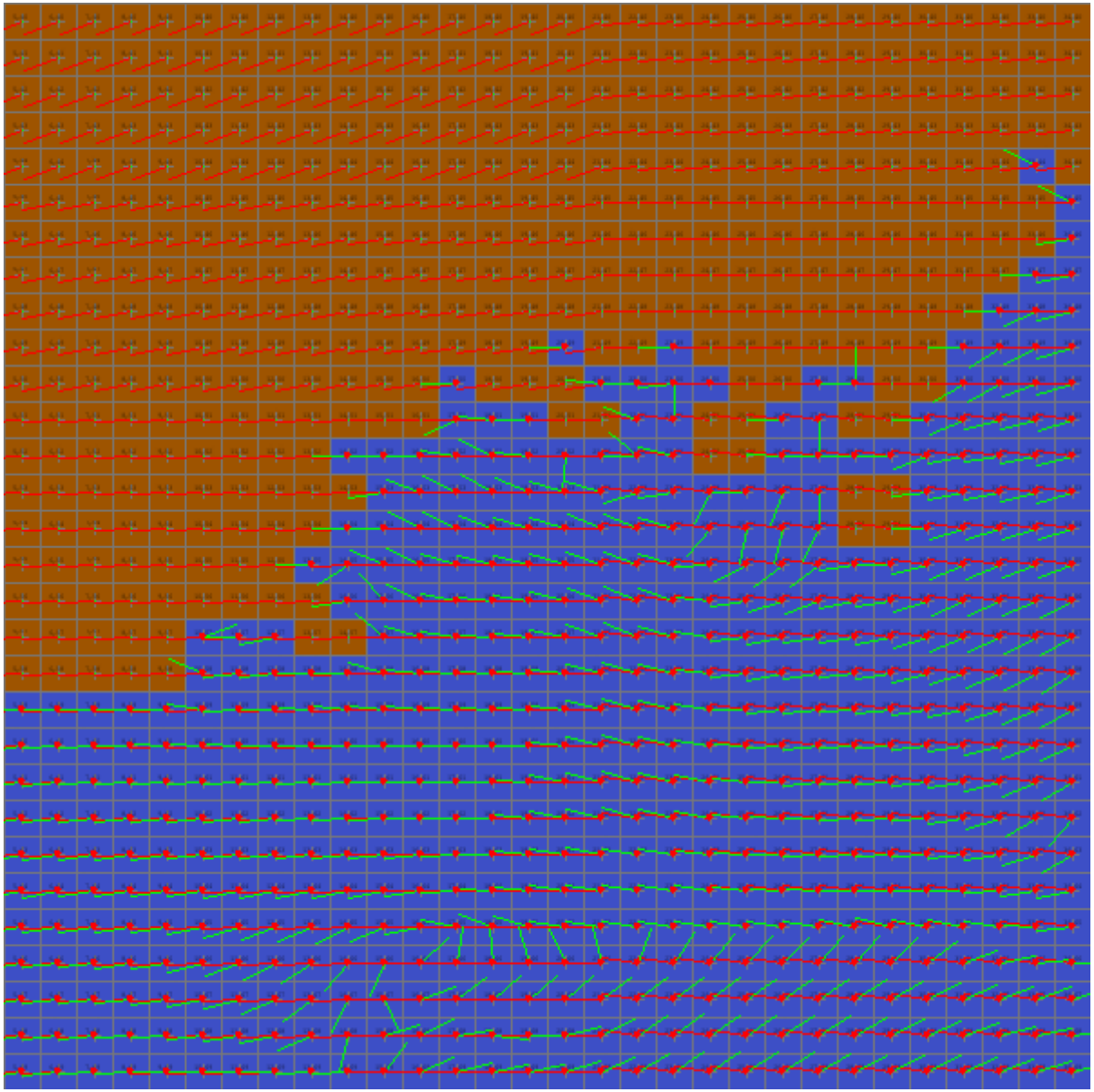




Running Simulation [Collisions Disabled]



Running Simulation [Collisions Enabled]



Collision Model

- The model has the following rules for collisions among patches:

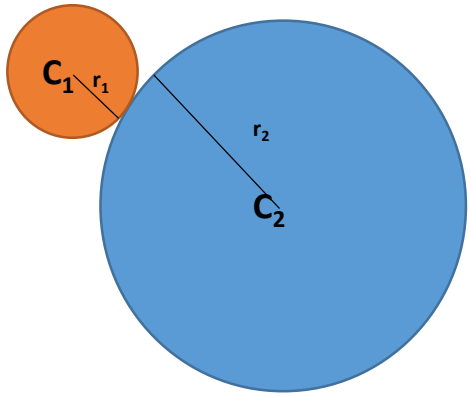
1) When two patches collide, they merge into one patch
2) The collision happens whenever the distance between the two patche's centers is less or equal than the sum of the two patches' radii:

$$\mathbf{d(C_1,C_2) < r_1 + r_2}$$

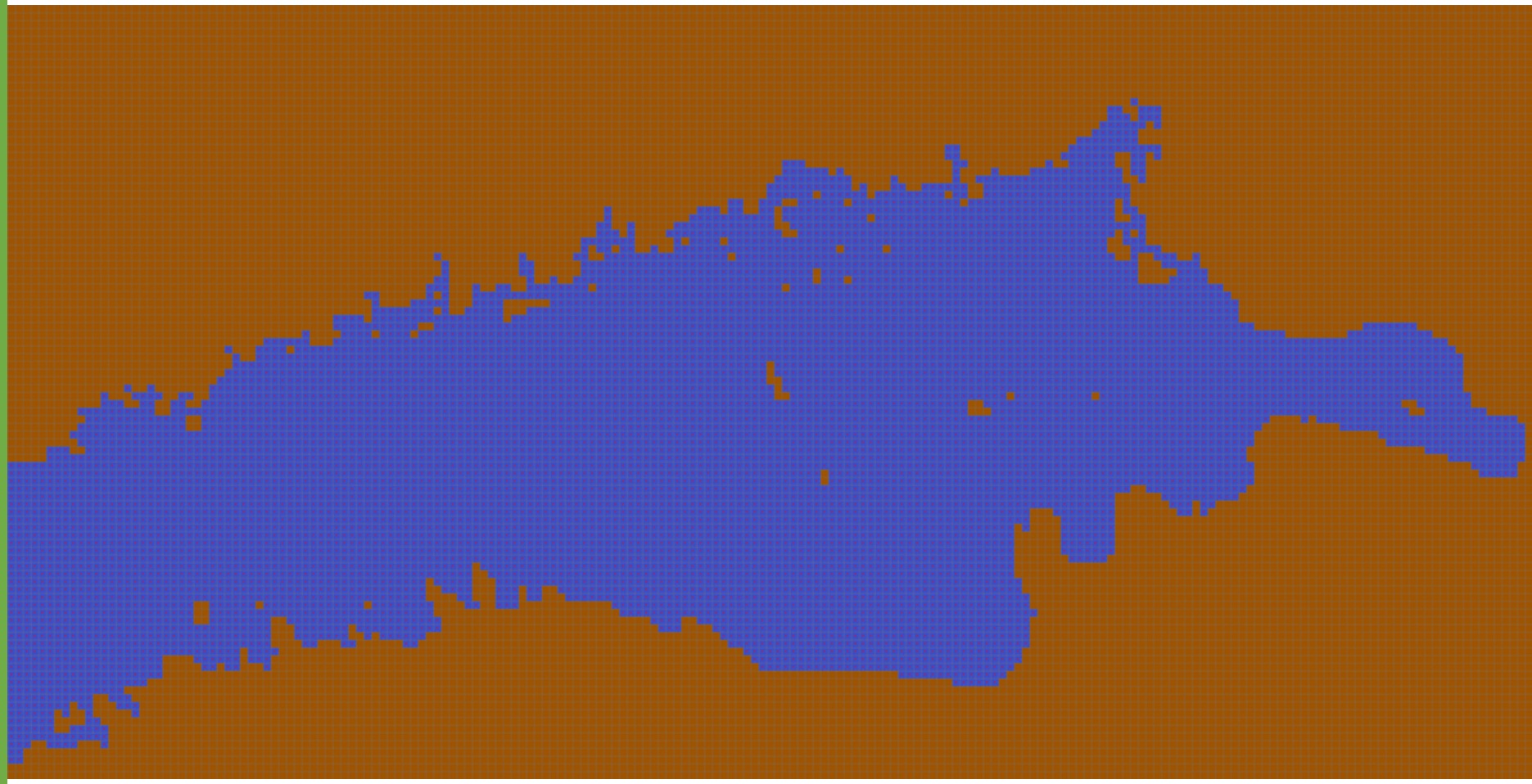
3) The patch resulting from the collision has the same area, as the sum of the areas of the two colliding patches
4) The bigger patch 'absorbs' the smaller patch and survives -> the resulting patch has the same center and lifespan as the biggest and oldest, of the two colliding patches

5) If a patch hits the land, it is marked as landed (N/S) and removed from the simulation

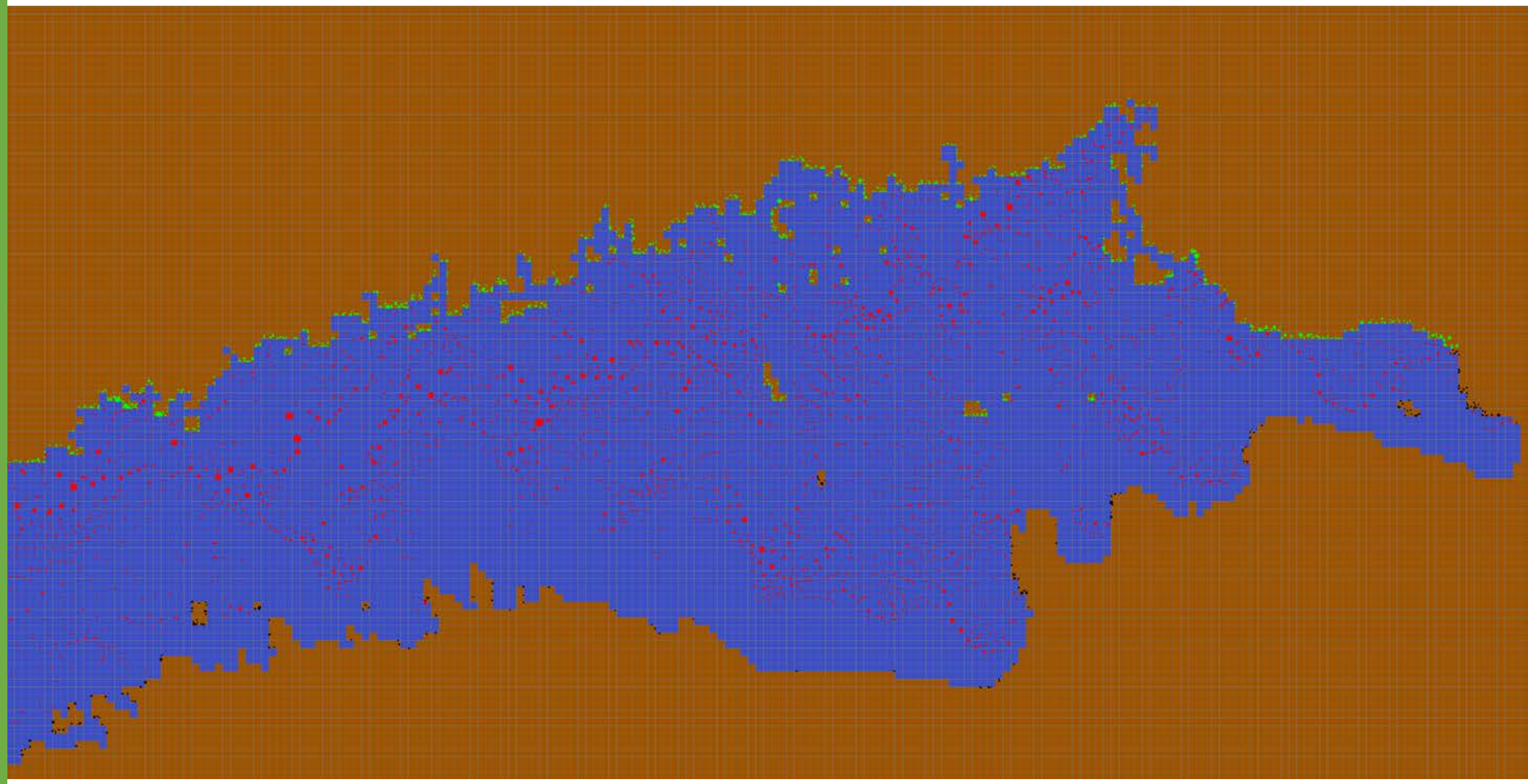
6) The N-body problem of collisions is solved by means of a BSP algorithm employing Quadtrees [collisions resolved in **O(log N)**].



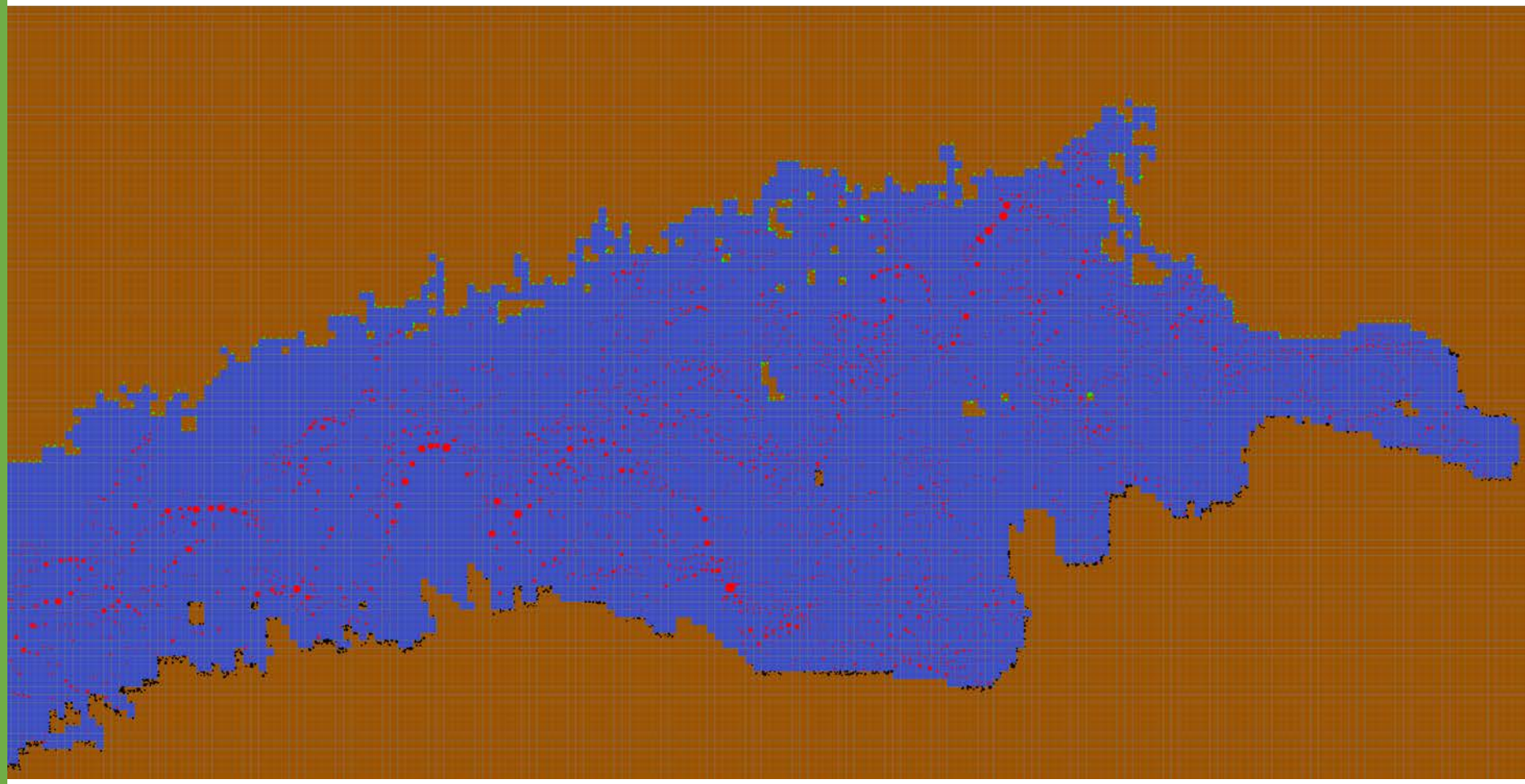
Gulf-wide 3 days run animation



(3 days run final distribution w/ majority of coastal hits to the N coast)



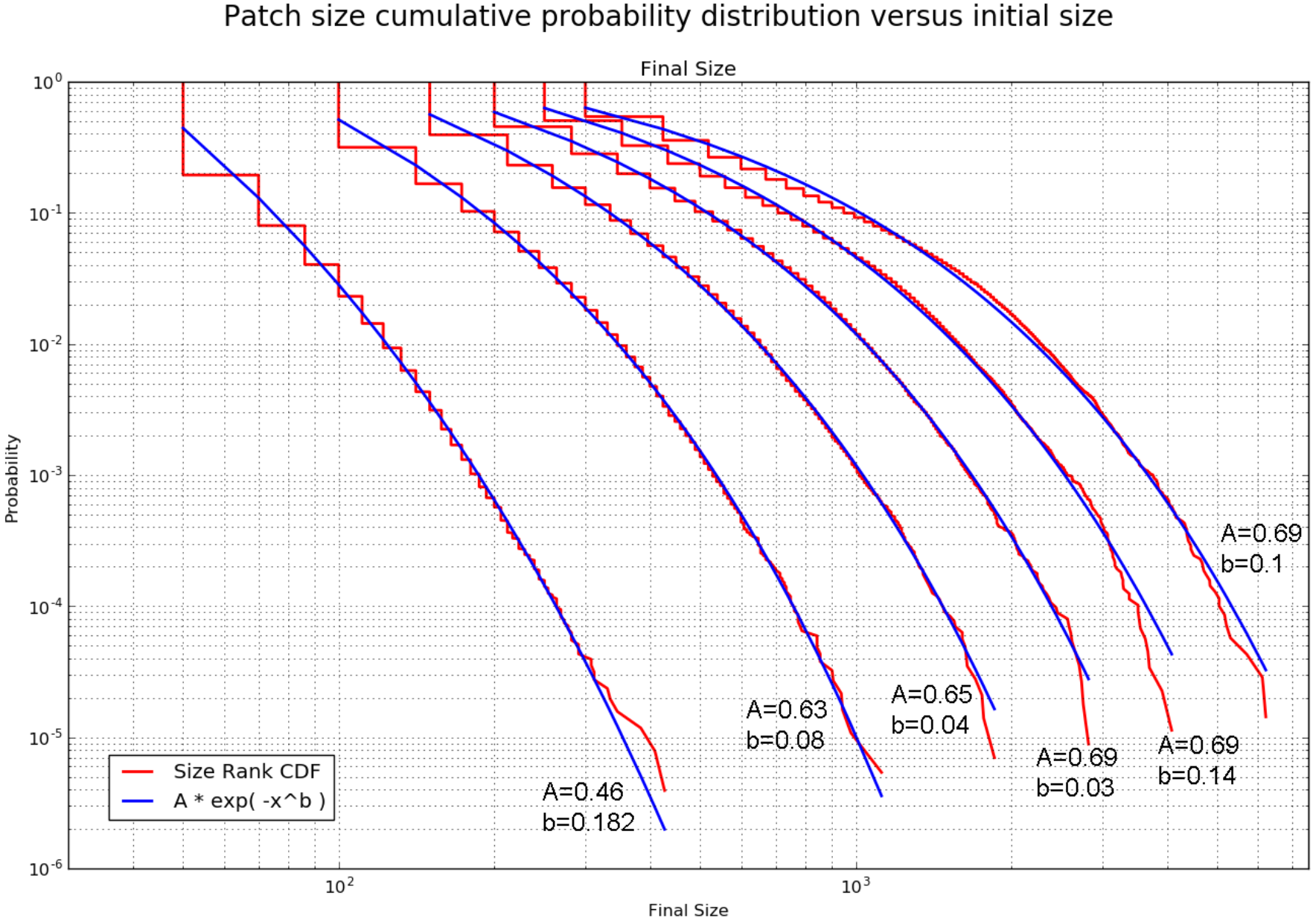
(3 days run final distribution w/ majority of coastal hits to the S coast)



Simulation Statistics (1 month time interval T_d , 31 time windows of 3 days each)

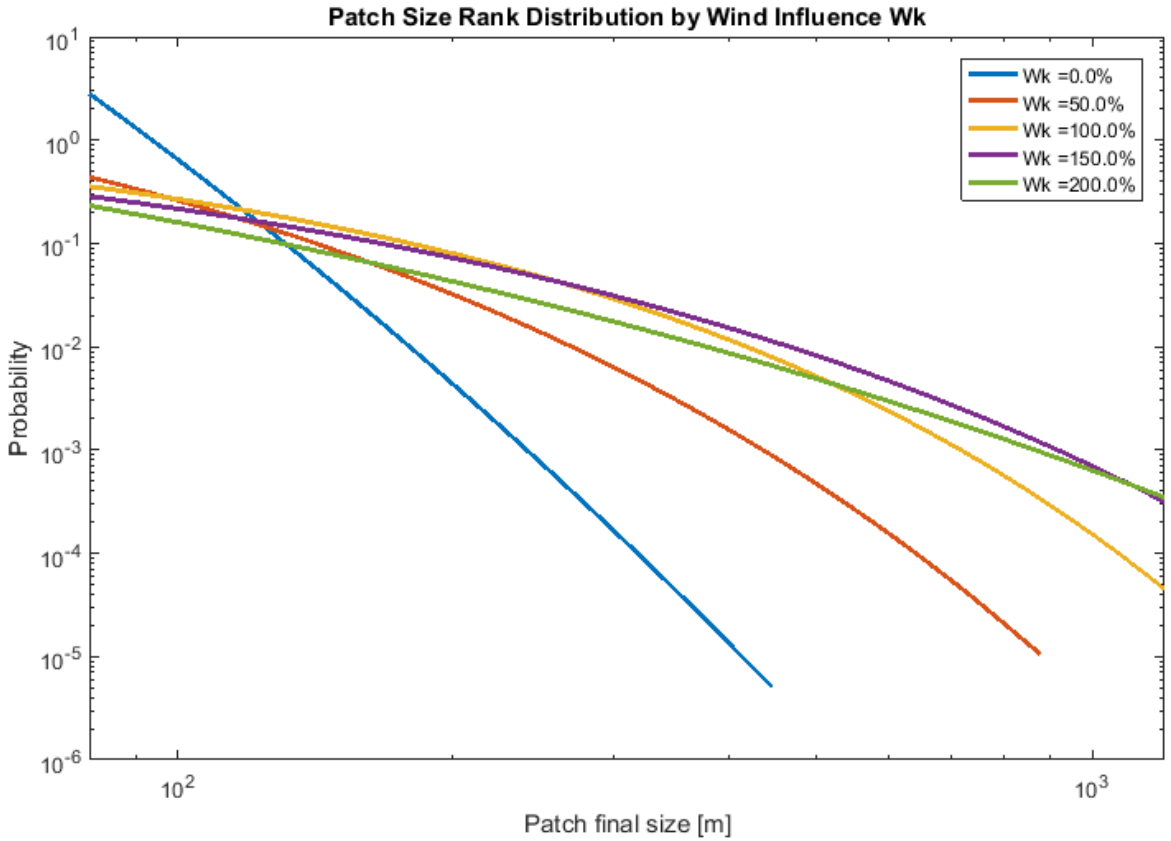
Parameter	Value
Min Min size [m]	150
Max Max size [m]	1573 (+1048%)
Avg Avg size [m]	197 (+131%)
Min Min life span [h]	3
Max Max life span [h]	72
Avg Avg life span [h]	53.2
Avg Coast hits (North)	947
Avg Coast hits (South)	545

Patch rank CDF (1 month time interval T_d , 31 time windows of 3 days each)

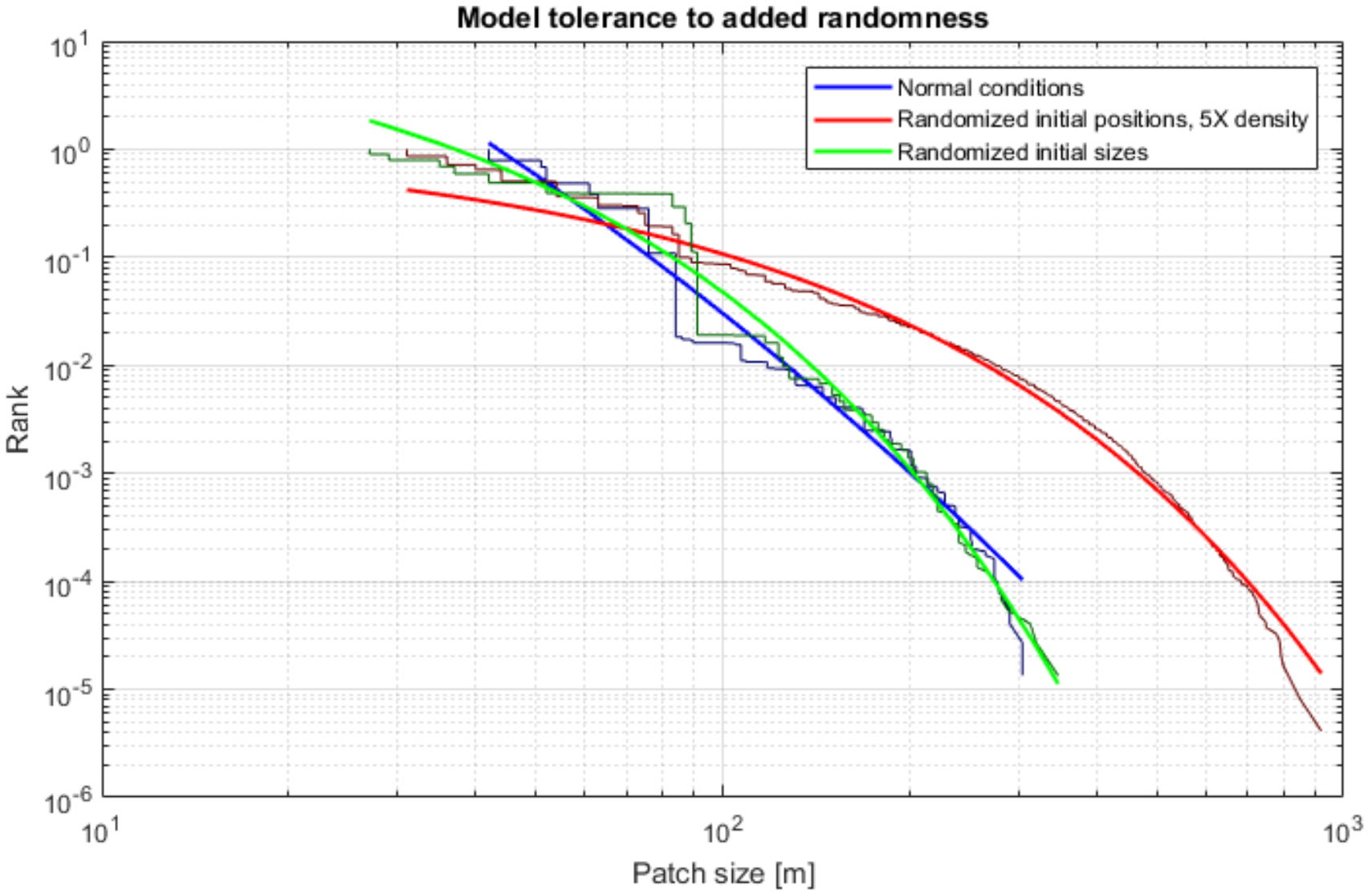


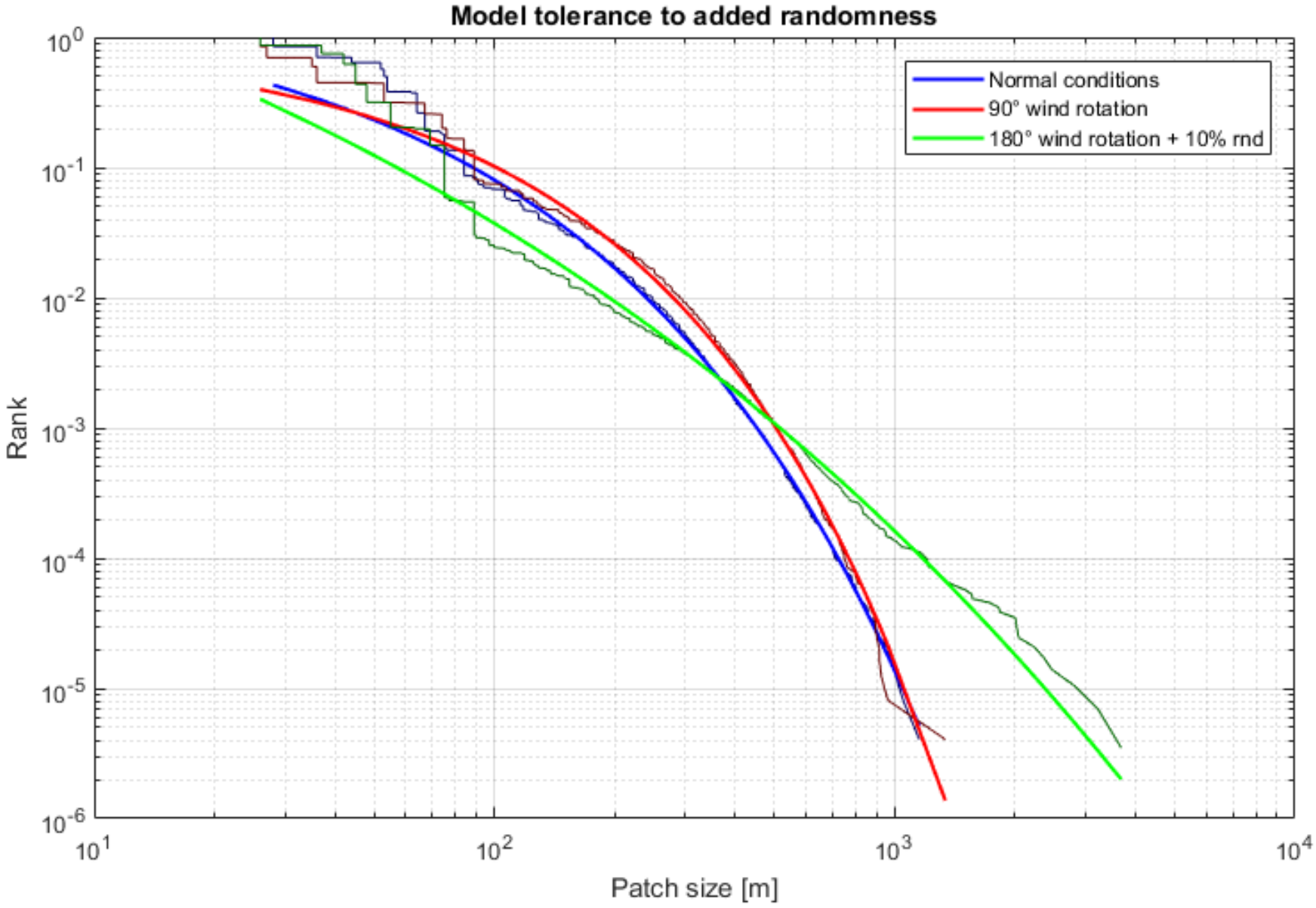
Some important questions are:

- How sensitive is this process with respect to some parameters?
- Does it show features similar to a phase transition?



- It seems evident that the wind does play a key role in the patch formation mechanism
- The collision-enhancing effect of wind drag seems to reach a saturation level when the wind intensity is increased by 150%
- This saturation value likely is not a universal number


Model tolerance to added randomness
Normal conditions
Randomized initial positions, 5X density
Randomized initial sizes
Rank
Patch size [m]
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
10^{1}
10^{2}
10^{3}


Model tolerance to added randomness
Normal conditions
90° wind rotation
180° wind rotation + 10% rnd
Rank
Patch size [m]
10^0
10^-1
10^-2
10^-3
10^-4
10^-5
10^-6
10^1
10^2
10^3
10^4

Conclusions

- A model is introduced, to simulate the behavior of patches of floating material on the sea surface, under the combined effect of currents and winds.

- The model employs current data from the OAAS model and wind data calculated from geostrophic wind information.

- The model employs binary-space partitioning of the region of interest to achieve adequate performance when computing collisions between patches in a large-size simulation area.

- **The empirical distribution of patch sizes seems universal and follows a stretched exponential power-law $f_\beta(t)=\exp(-t^\beta)$**
- This law is independent from:
  - Variations to the initial size and position of patches
  - Variations to wind direction

- **The effect of wind in augmenting the patch formation process is very clear**, and it shows the existence of a saturation level.

- We regard this as an indicator that such processes on the sea surface are not really random; instead, they exhibit classic self-organisation and scale-invariance properties

Thank you.

Simulation Model - Collision Detection [Naive Approach]

- The easiest way is to check each patch, for collision against every other patch of the set.
  ==> This means that if there are N total patches, at every time-step, N*N collision checks must be performed
  ==> In this case the collision-checking algorithm would require a time in the order of

$O(N^2)$ *[big-O notation, quantifies the amount of time required by a program to run, as a function of the length of the input, and exludes all coefficients and lower order terms]*

- While this is perfectly fine for a small number of patches (i.e. a limited sub-region of the area of interest), it becomes computationally unfeasible as the number of patches grows.

Example: computing collisions in the model using this approach takes around 3s/timestep for ~200 patches. Computing the same thing for the entire Gulf of Finland (~8000 patches) takes 3 minutes
==> **Not acceptable**, too many time-steps to compute in a reasonable time
(1 run = 3min x 24timesteps x 30days = 2160 min = 36 hrs run time)

- This approach has the only advantage of offering a simple way to 'store' the patches inside a data structure -> a simple 2D array can be used.

Simulation Model - Collision Detection [BSP Approach]

- A slightly more complex way of storing the patches' positions, allows for a *much* faster way to obtain the collision detection.

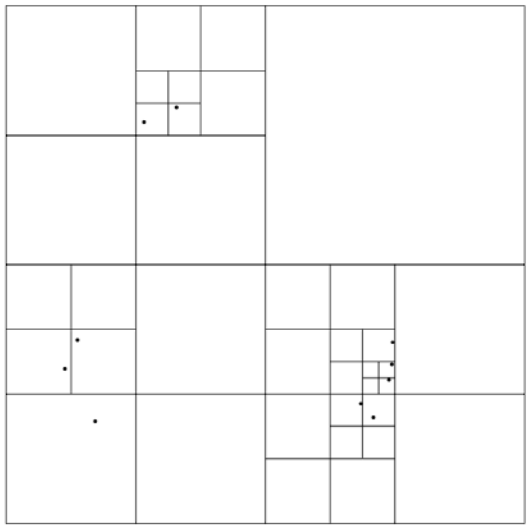
- BSP => Recursively subdivide a space into *convex sets* by using *hyperplanes*

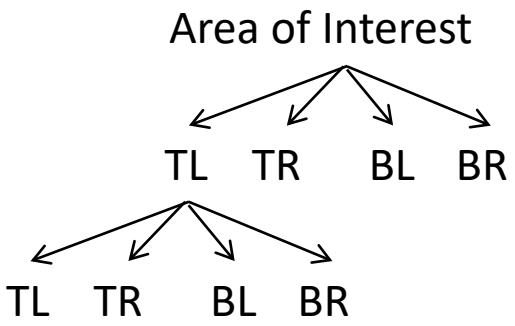
[a region such that, for every pair of points within the region, every point on the straight line segment that joins the pair of points is also within the region]

[subspace of one dimension less than its ambient space]

- A **quadtree** structure is used. A quadtree is a 2D-BSP data structure for organizing points on a plane. It's represented tree-shaped data structure, where eah internal node has exactly 4 children.

- A region of the 2D plane is decomposed into four equal quadrants, subquadrants, and so on, with each leaf node containing only information about a patch, belonging to a specific subregion.





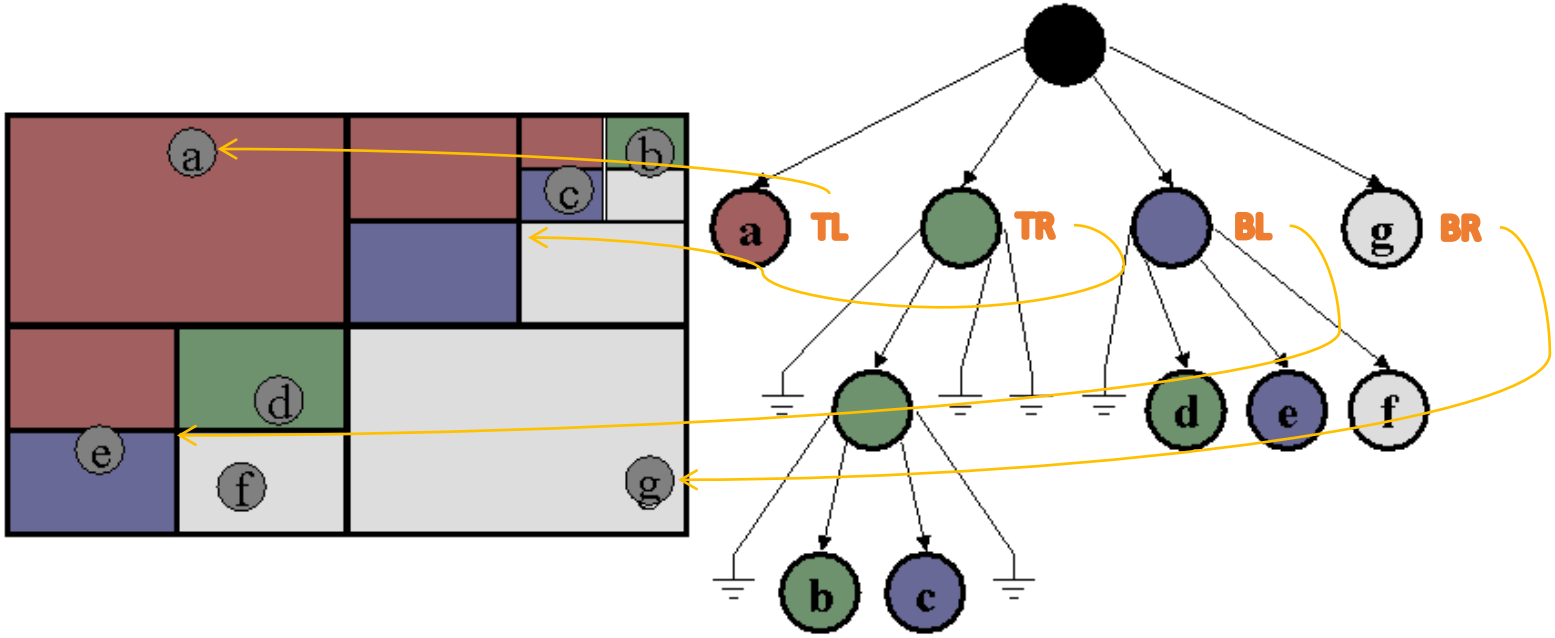
Simulation Model - Collision Detection - Quadtrees

- A quadtree decomposes space into adaptable cells

- Each cell has a maximum capacity of patches. When the max. capacity is reached, a cell is split into 4 sub-elements (a new level is added to the tree)

- The tree directory follows the spatial decomposition of the quadtree

- Notes on balancing the tree / bad cases



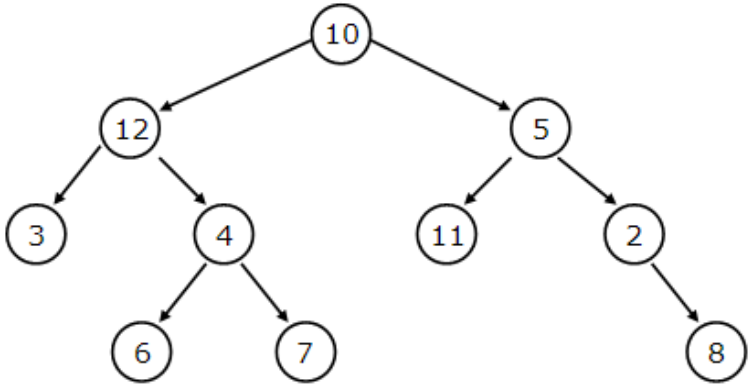
Simulation Model - Collision Detection - Quadtrees

- With this approach, some 'performance' (= time) is lost while adding patches to the structure, as the tree needs to be traversed every time, to determine which 'leaf' the new patch will be in. This takes **O(N log N)** at worst [with the *naive* approach, it takes constant time **O(1)**]

- A lot of time is saved though, when computing collisions, as this can be done only between each patch, and its *closest neighbor* within the spatial structure.

- Note on Recursive Tree Traversing

- **Nearest Neighbor Search (NN) ->** it's an algorithm which aims at finding the point in the tree, that is nearest to a point given as input.
  - This search can be done *efficiently* using the properties of the tree to quickly eliminate large portions of the search space:



Levelorder tree traversal
10, 12, 5, 3, 4, 11, 2, 6, 7, 8

Inorder tree traversal
3, 12, 6, 4, 7, 10, 11, 5, 2, 8

Preorder tree traversal
10, 12, 3, 4, 6, 7, 5, 11, 2, 8

Postorder tree traversal
3, 6, 7, 4, 12, 11, 8, 2, 5, 10

Simulation Model - Collision Detection - Quadtrees

1) Starting from the root note, the algorithm moves down the tree recursively -> it goes left or right depending on whether the point is lesser or greater than the current node in the split dimension
2) Once the algorithm reaches a leaf node, it marks the corresponding patch as the 'current best'
3) The algorithm unwinds the recursion on the tree, performing the following steps at each node:

    3a) If the current node is closer than the current best, then it becomes the current best

    3b) It checks whethere there could be any points on the side of the splitting plane that are closer to the search point than the current best.

4) When the process is complete starting from the root node, the search is complete.

The nearest point is found in **O (log N)** time, given a completely random distribution of points.

$$\mathbf{O (\log N) << O (N^2), N > K}$$

Example: computing collisions in the model using this approach takes around 3s/timestep for ~200 patches. Computing the same thing for the entire Gulf (~8000 patches) takes 3 minutes !!

3 seconds / timestep
for the *entire* GoF