DOZER: description of morphodynamic model and game mechanics

Eli D Lazarus

Environmental Dynamics Lab, School of Geography and Environmental Science, University of Southampton, Southampton, SO17 1BJ, UK

Correspondence: e.d.lazarus@soton.ac.uk

ORCiD: 0000-0003-2404-9661

How to cite this document:

Lazarus, E (2025) DOZER: code and documentation. *Zenodo* [model]: https://doi.org/10.5281/zenodo.15589917

Companion publications:

[pending]

GitHub repo: https://github.com/envidynxlab/dozer_release

1. Introduction

This is a detailed model description for DOZER – a single-player, arcade-style game motivated by observations of emergency road crews clearing sand from beachfront roads during coastal storms (Lazarus and Goldstein, 2019).

DOZER is also a fully coupled morphodynamic model, in which plowing actions by the player affect, and are affected by, patterns of storm-driven sediment deposition. The bulldozer functions as a distinctly anthropic physical process of sediment transport (Haff, 2010) – an adaptive agent of geomorphic change. But rather than being encoded, adaptive behaviour here is handled by a human player.

DOZER can be played for fun, or used as a heuristic tool for insight into the dynamics of deliberate human intervention in the physical processes of a natural hazard event.

2. Domain

The model domain **(Fig. 1)** is a discretized grid with a 4:3 aspect ratio (typical of a classic arcade screen). The domain represents the back-barrier floodplain of a coastal barrier (FitzGerald et al., 2008). The barrier shoreface, beach, and dune are not explicitly modelled, nor is the landward back-barrier boundary.

Cell colour in the domain represents the volume of sand in a given cell. Darker browns indicate greater sand volume. Empty cells are black.

The game screen that is visible to the player includes a two-lane road, but the road is for aesthetic purposes only. At the start of a game, DOZER sits idle in the middle of the lower third of the screen.

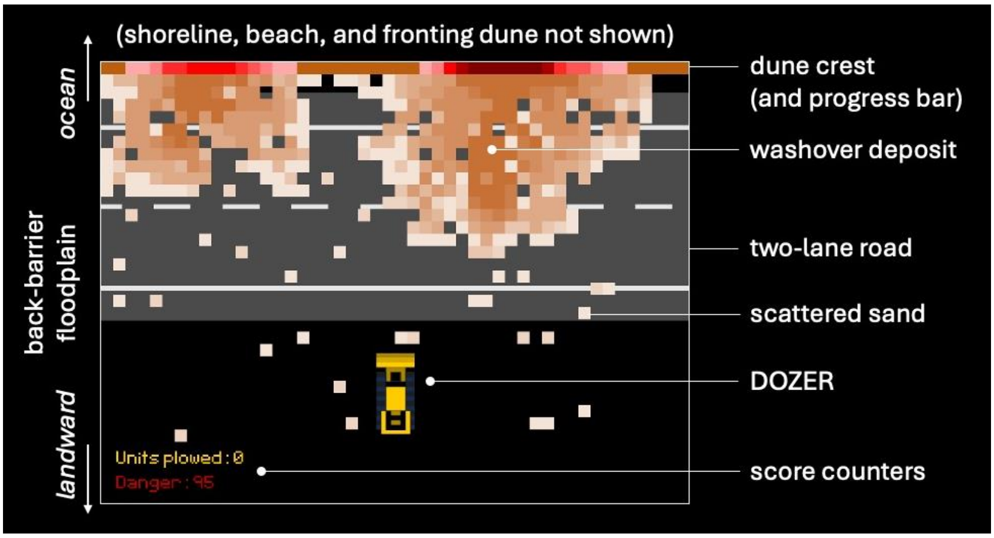

Figure 1. *Elements of DOZER game domain.*

2.1 Domain tiles

While game mechanics are handled with *pygame* modules, most mechanics related to numerical modelling are handled with *numpy* modules (see associated README). That means DOZER translates between player updates in the onscreen *pygame* interface and operable *numpy* arrays in the model code. The *tiles.py* script converts the *numpy* array, in units of array elements, to on-screen grid squares, in units of pixels. In *pygame* terminology, DOZER and each tile of the domain grid is rendered as an individual "sprite". This enables the fundamental pygame mechanic of "sprite collisions" between the DOZER sprite and tiles containing sand. Tile scaling relative to the size of the DOZER sprite is an aesthetic choice and can be specified by the user (see **Table 1**).

3. Overwash

In coastal barrier systems, overwash is a natural physical process of flow-driven sediment transport in which super-elevated water levels drive sediment from the shoreface and beach across the barrier crest onto the back-barrier floodplain (Donnelly et al., 2006). The sediment deposit that overwash leaves behind is called washover. Overwash flow may overtop, breach, and/or incise through a fronting dune in its cross-shore incursion. Overwash and washover are central to the premise of DOZER.

3.1 Determination of overwash sites and treatment of the dune crest

In DOZER, the dune crest is treated as a one-line model of height H in the alongshore dimension (*cols*), treated separately from the gridded domain. Treatment of the dune crest effectively establishes the external forcing for the rest of the model.

Overwash sites along the dune crest are determined by a subroutine that creates a pattern of flow paths from an array of directed random walks **(Fig. 2a)**, based on a simple model of self-organised critical random directed polymers by Jögi & Sornette (1998). This subroutine creates an array with the same alongshore dimension (*cols*) of the model domain but several times longer in the cross-shore dimension. The cells are assigned a uniformly random value between [0, 1]; cells in the first row of the domain are assigned a value = 1. These are "walker" cells. Each walker iteratively steps down the array into one of the three neighbouring cells (or two neighbouring cells, for edge cases) in the row below, occupying whichever cell has the minimum value. Each step is made independently of choices by other walking cells. When two walking cells step into the same neighbour cell, they merge into a single walker and their values are added: for example, two walkers = 1 become a single walker = 2. The subroutine iterates until remaining walkers reach the bottom of the array (here, ~2–5 remaining walkers after 100 iterations). The column-wise positions of these cells become the sites along the dune crest at which overwash initiates. Over many tens of runs of this subroutine, overwash sites tend toward a preferred (non-random) alongshore spacing (**Fig. 2b**), qualitatively consistent with observations of preferred overwash and washover spacing in physical experiments and field examples (Lazarus and Armstrong, 2015; Lazarus, 2016; Lazarus et al., 2020).

The directed random walks resemble an elongated drainage network: the value of each final walking cell, normalised by the total number of cells in the initial array (alongshore dimension of the model domain), indicates the fraction of flow capture at each overwash site. These fractional values in turn dictate the relative depth to which each overwash site incises into the dune crest. For example, a site that accounts for 40% of the forcing flow evolves toward an incision that is twice as deep as a site that channels 20%.

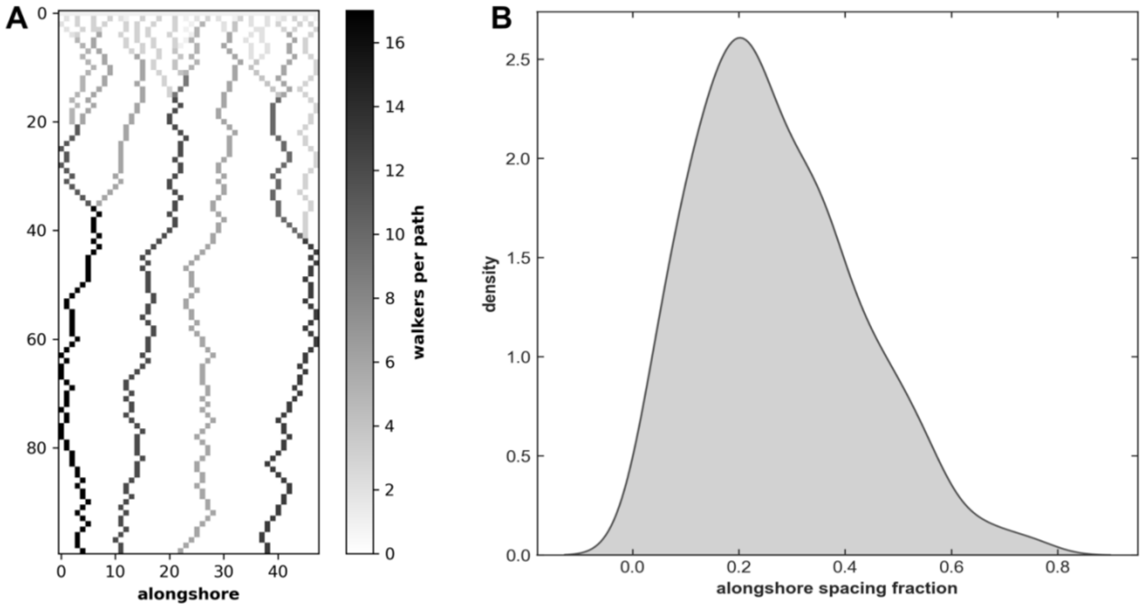

Figure 2. *Directed random walk routine by which overwash sites in the dune crest are selected.* ***(a)*** *Flow paths of "walkers" converge over a specified number of steps (here, 100 steps).* ***(b)*** *Over many iterations (here, n = 100), relative spacing between overwash sites tends to describe a preferred distance. Spacing fraction is shown relative to the total alongshore reach of the domain.*

3.2 Overwash incision

Incision through the dune at overwash sites is treated as inverted general Gaussian curves shaped by three parameters: M, the number of dune-crest cells that the Gaussian shape spans; p, a shape parameter; and σ, the standard deviation of the shape. A Gaussian curve is centred at each overwash site cell determined by the subroutine of directed random walkers.

The model delivers overwash flow to the back-barrier floodplain in pulses, which occur at randomised intervals every 3–10 seconds. With each pulse through an overwash site, M increases by +1 cell and σ by +0.1, gradually enlarging the Gaussian shape (**Fig. 3**). The model starts with a dune crest of uniform height H alongshore; the model (and game) ends when an overwash site incises through the full height of the dune to the floodplain floor.

Cell colour across the top row of the floodplain indicates the relative integrity of the dune alongshore. Reaches of crest that are at or near their initial height are the brown colour of maximum sand volume; the more deeply incised the dune is at a given cell, the redder its colour becomes. A "danger" meter at the bottom left of the game screen shows the maximum incision (as a percentage of initial crest height) along the dune (**Fig. 1**).

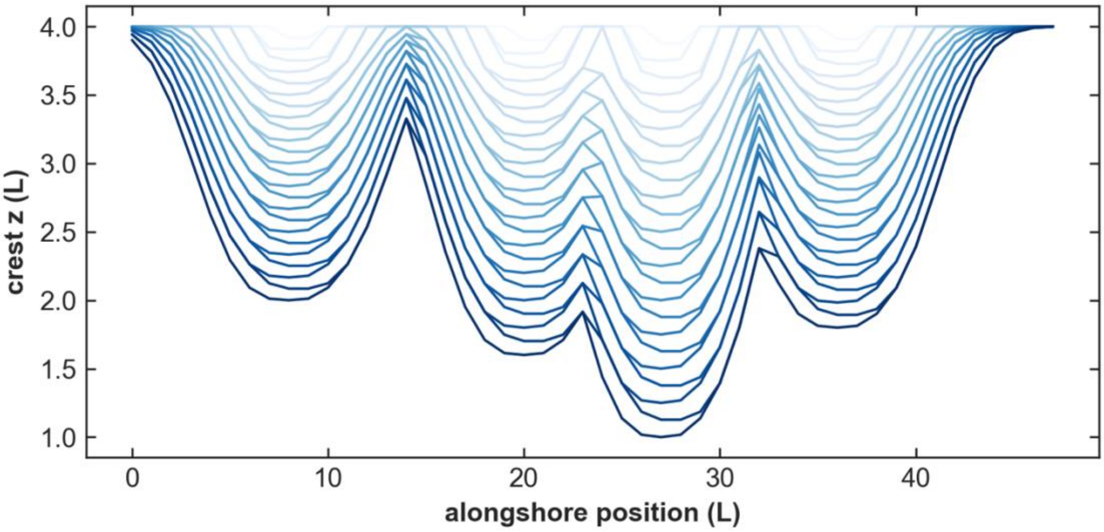

Figure 3. *Example of dune crest evolution, absent any DOZER interventions, with series of overwash pulses (colour darkens with elapsed game time).*

3.3 Forcing flow and perched overwash sites

The water level of forcing flow relative to the dune crest determines whether or not an overwash site is active. Initially, the water level of the forcing flow is equal to the height of the dune crest. Once overwash sites are established, the forcing water level is lowered by the equivalent total cross-sectional area removed from the dune by overwash. When an overwash pulse is triggered, only those sites where forcing flow exceeds a threshold depth over the crest profile have their Gaussian shapes expanded by an additional increment and transfer sand to the floodplain. An overwash site associated with a small proportion of flow capture will create a commensurately shallow incision in the dune crest. This means it may be left perched as the forcing water level drops with the progress of other, deeper incisions. The fraction of forcing flow associated with a perched site is distributed proportionally to neighbouring sites as a function of alongshore distance (i.e., a closer neighbour receives a higher proportion of the perched forcing flow).

4. Washover

With each overwash pulse, sand eroded from the dune at a given overwash site propagates onto the floodplain from the top edge of the domain, forming washover deposits. Sand in the model is conserved: all volume lost from the dune crest is transferred to the floodplain. Sand cannot be transported off the domain edges.

Instead of simulating hydrodynamics, the model uses a rule-based flow-routing routine to redistribute sand into washover. The routine operates on two versions of the floodplain: a temporary scratch domain; and the active game domain. When an overwash pulse delivers new sand, cells containing new sand are copied onto the scratch domain, characterised by uniformly random surface roughness and a linear slope that dips away from the dune toe toward the bottom of the domain. Each cell in the scratch domain containing a volume of fresh sand greater than a defined threshold (analogous to a sediment lag) compares its base elevation against those of its eight adjoining neighbours. Sand in exceedance of the volume threshold (which remains as a lag deposit) is then portioned from the cell to any lower neighbours according to local slope. Ordinal neighbours have a longer run (= $\sqrt{2}$) than cardinal neighbours (= 1). Local cell-to-cell slopes are sensitive to the scale of the random roughness imposed on the scratch domain; the background slope also imposed on the scratch domain imparts a subtle directional preference to flow. Low roughness relative to the background slope will tend to produce elongated, finger-like of washover with a smooth perimeter (low perimeter-to-area ratio); higher roughness relative to background slope tends to produce blunter, splay-like washover deposits with more rugose perimeters (high perimeter-to-area ratio).

When all sandy cells in the scratch domain have checked their neighbours and apportioned their transportable volumes, their collective additions (to neighbours) and subtractions (from themselves) are summed, the scratch domain is updated with the net result, and the neighbour-checking step repeats. This operation loops until all sand transferred from the dune by an overwash pulse has been redistributed, such that no cell in the scratch domain contains more sand than the threshold lag volume. If a depositional cell gets "stuck", such that it has an excess of movable sand but sits at a lower elevation than its neighbours, the routine allows for redistribution of sand upslope (to the neighbour with the smallest slope difference).

Once distributed, the layer of sand delivered by a given overwash pulse is added to the active floodplain domain, and to the base elevation of the scratch domain, establishing the floodplain condition for the next overwash pulse. The active floodplain domain has a slope of zero. In the game interface, all grid cells containing transportable sand are temporarily rendered in blue so that the player sees the pattern of the overwash pulse. Cell-to-cell deposition increments at the visual frame rate of the game (60 frames per second).

5. Bulldozer agent

A player controls the bulldozer using keyboard inputs. The bulldozer can move forward or backward, and rotate in a full circle to the right or left. The player also determines whether the plow blade of the bulldozer is up or down. (The default position is up.) Keyboard inputs are nonexclusive, so that the bulldozer can travel, turn, and plow simultaneously.

If the bulldozer is traveling forward with the blade down and encounters a sandy cell, the volume of sand at that grid cell is subtracted from the floodplain domain and added to the plow blade. In the code, this is achieved by checking for sprite collisions: each domain cell is a sprite, as is the bulldozer; the plow blade is itself a separate sprite that tracks with the face of the bulldozer rectangle. The blade can collect and push sand up to a maximum volume (the blade capacity), beyond which the blade stays full but skims over the floodplain surface without picking up any more sand. Sand will stay on the blade until the player deposits it by releasing the blade (or reversing the bulldozer). The bulldozer cannot scrape sand when moving in reverse, nor can it push sand off the edges of the domain. By default, DOZER movements are slower than the rate of sediment delivery to the floodplain.

Beyond the specific mechanics of how the bulldozer moves, there are no fixed rules of play directing how a player operates the bulldozer. The player is given no explicit instructions for what to do. The player might choose to plow washover to the verges of the road, or to the right and left of the screen, instead of plugging it back into gaps in the dune. The player could make the bulldozer churn donuts, or try to plow their name in the sand.

Even if the player takes no action and leaves the bulldozer idle, overwash and washover will progress until the dune crest is fully incised.

6. Morphodynamic coupling

Morphodynamic coupling in the model between overwash flow, washover deposition, and bulldozer actions is affected in two ways.

First, by rearranging the shape of washover deposits as they develop over successive pulses, plowing actions change local paths of steepest descent for overwash flow, steering washover deposition into accommodation spaces different from those it would occupy in the absence of any intervention. Meanwhile, bulldozer actions are informed by the evolving pattern of washover deposition. This kind of coupling occurs even if the player never directly interacts with the dune crest along the top of the screen.

Second, a different mode of coupling occurs if the player chooses to plow sand back into incised reaches of the dune. Because the model assumes constant forcing, if any of the incised sites in the dune line become partially infilled by plowing, any blocked portion of forcing flow is redirected laterally to the nearest neighbouring overwash site and added to the existing fraction of forcing flow at that site. If a partially infilled site has two neighbours, the blocked portion of flow is diverted to both neighbours as a function of lateral distance (more flow goes to the closer neighbour). Competition for, and lateral redistribution of, overwash flow among near-neighbour overwash sites has been observed in physical models of overwash morphology (Lazarus, 2016; Lazarus et al., 2020). No new overwash sites are created during a game, but a site that gets infilled by plowing can reactivate if a sufficient portion of forcing flow gets diverted back to it. This is a contrivance of design, but analogous to assuming that a newly plugged gap in the dune is weaker than an existing intact section that has not been plowed.

I refer to this second mode of coupling as "whack-a-mole" dynamics: plugging one gap worsens incision and washover intrusion elsewhere along the dune line. This dynamic is consistent with one associated with levees in flood-prone settings, in which interventions to prevent frequent, small-scale floods drive the system toward infrequent, large-scale, disastrous events (Criss and Shock, 2001; Werner and McNamara, 2007).

7. Dummy model for comparative analysis

To enable direct quantitative comparison between the outcome of a game and the result that would have manifest in the absence of any plowing action under the same initial conditions, a dummy model of the overwash and washover routines, without the bulldozer, runs in parallel

(**Fig. 4**). The dummy model is not visible to the player and does not affect game function. However, the inbuilt capacity to analyse human-altered versus natural morphodynamic outcomes is what makes DOZER a tool for systemic insight.

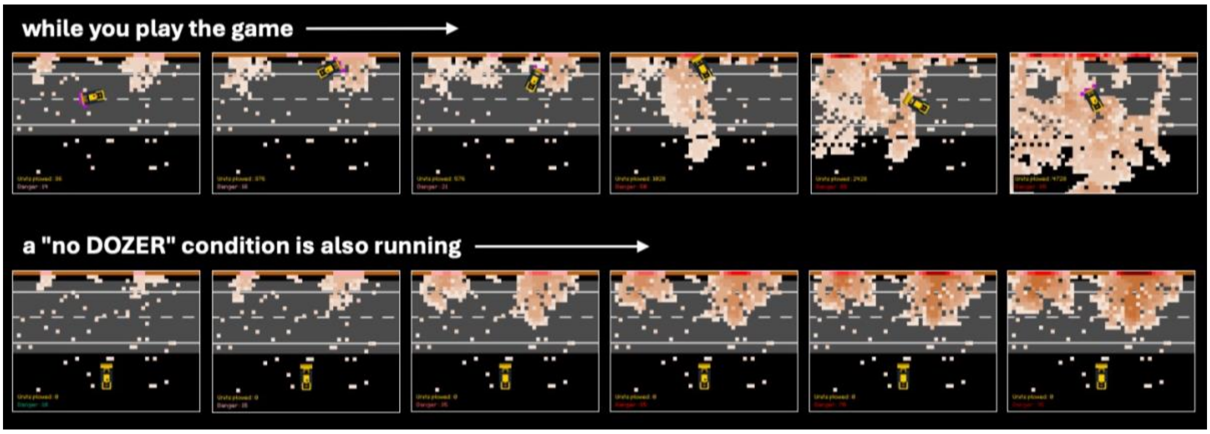

Figure 4. *Sequence of screenshots from a typical game of DOZER (top), and the "natural" (dummy) result for the same conditions in the absence of any plowing intervention (bottom). Comparison shows the divergence between human-altered versus natural morphodynamic outcomes.*

8. Analytics

8.1 Exported files

DOZER exports a number of outputs to the project *data* folder.

The primary data outputs for analysis are *.csv* files (**Table 2**). These are named using the *YYYYMMDD-HHMMSS* datetime format along with an identifying trial number. Suffixes '*_D*' and '*_ND*' refer to DOZER and "NO DOZER" (dummy model) files, respectively.

Variables recorded in the *gamedata* file are captured at a regular interval from the start of the game clock. Here, the default interval is every 2.5 seconds. Files specifically related to overwash pulses – *Qmove_series, ow_footprint, ow_sand* – are captured when overwash events occur.

The user also has the option of capturing screenshots at regular intervals during the game, and capturing screenshots of the overwash events. These are saved as *.png* files, and are named using the same *YYYYMMDD-HHMMSS* datetime format along with an identifying trial number.

8.2 Analytic outputs

The *DOZER_analytics_release.ipynb* file is a jupyter notebook that helps users process and plot game data from the exported files. The notebook file works from the same folder as the stored game data. The following outputs are illustrative examples, derived from the supporting materials included in this repository.

8.2.1 Compare final domains

Visually comparing the final depositional patterns across the DOZER and dummy domains, captured when the game ends (**Fig. 5**), helps the user gain physical intuition with the model before any further abstraction.

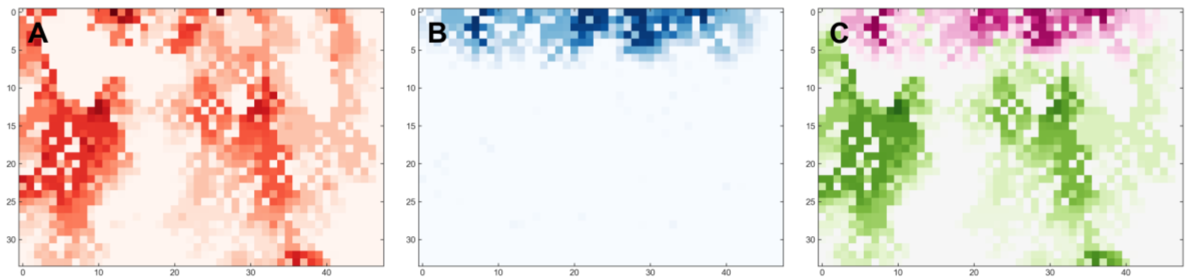

Figure 5. *Comparative deposition patterns.* ***(a)*** *Deposited floodplain with typical DOZER result (reds);* ***(b)*** *the dummy counterpart (blues); and* ***(c)*** *difference map (DOZER minus dummy), showing patterns of deposition deficit (pinks) and surplus (greens) relative to expected volumes.*

8.2.2 Compare crests

Visualising the evolution of the dune crest under DOZER and dummy conditions shows how infilling by plowing actions alter the spatial pattern and depths of incisions at overwash sites alongshore (**Fig. 6a, 6b**). Tracking the evolution of the forcing flow shows how partial infilling by DOZER can affect the waterline. Sufficient infilling will cause the waterline to rise, albeit temporarily – but potentially enough to reactivate a perched overwash site.

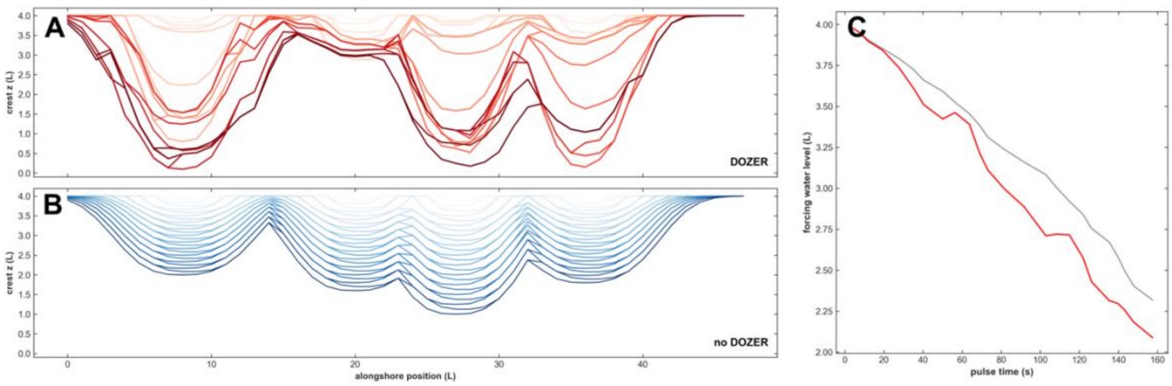

Figure 6. *Dune crest and waterline evolution for a typical game.* ***(a)*** *DOZER dune (reds);* ***(b)*** *Corresponding dummy condition. Colours in (a) and (b) darken with elapsed time.* ***(c)*** *Dynamic waterline of forcing flow over time for DOZER (red) and dummy (grey) conditions.*

8.2.3 Morphometric scaling and dynamic allometry

Physical laboratory experiments suggest that washover deposits demonstrate dynamic allometry: their geometric characteristics not only scale relative to each other in a systematic way, but those scaling relationships also evolve in a systematic way over time (Lazarus et al., 2020). Despite its stylised mechanics, DOZER nevertheless generates washover patterns that reflect morphometric scaling relationships and dynamic allometry (**Fig. 7**), as found in field examples and physical experiments (Lazarus, 2016; Lazarus et al., 2020, 2021, 2022). Tracking the morphometric evolution of individual washover deposits shows well-organised washover growth in the dummy model: a scaling relationship between volume and area, for example, reveals an inflected trajectory, in which the footprint of the deposit rapidly expands, and then gradually gains volume. By comparison, washover allometry under DOZER conditions is more variable and excursive, as plowing disrupts or breaks intrinsic morphometric scaling. In any given game, plowing tends to affect some deposits more than others. The DOZER sprite can only move so quickly: while the player works to plow away one deposit, another might develop with little interference. In such a case, the morphometric result of the DOZER and dummy conditions may overlap. Morphometric trajectories in the DOZER condition can also be recursive, such as if a player plows away a deposit at an overwash site that later reactivates.

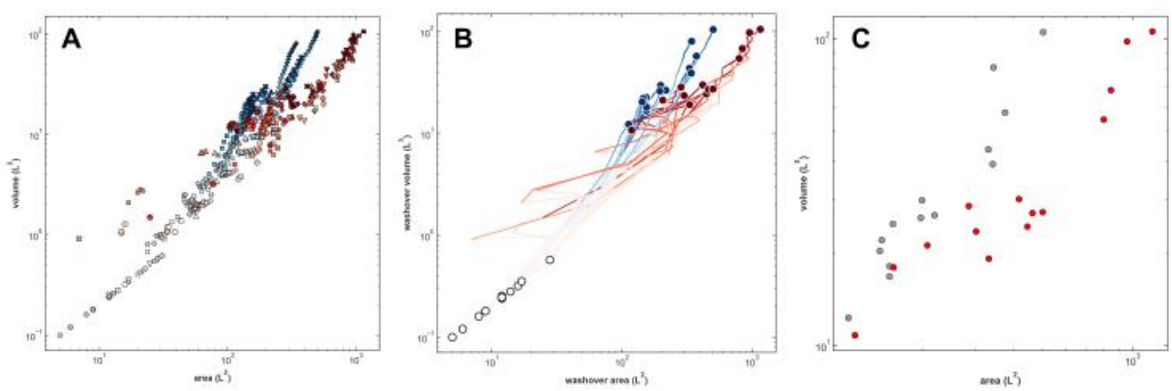

Figure 7. *Example of morphometric scaling and dynamic allometry for washover volume as a function of area.* ***(a)*** *Geometric characteristics of individual DOZER (red) and dummy (blue) washover deposits, sampled every 2.5 seconds during game play, for ten games with different initial conditions (symbols). Colour darkens with elapsed game time.* ***(b)*** *Results in (a) plotted as trajectories in partial phase space; open circles indicate initial deposit and filled circles the final geometry.* ***(c)*** *Morphometric scaling relationship between washover volume and area for final washover deposits only – same as filled circles in (b) – where red points are from DOZER conditions and grey from dummy conditions, respectively. In general, DOZER intervention tends to yield washover deposits that are larger in area and lower in volume relative to the dummy model.*

8.2.4 Divergence between predicted and observed conditions

The evolution of corresponding DOZER and dummy conditions can also be compared directly, as a relationship between prediction and observation (**Fig. 8**). Here, the dummy model is treated as the prediction (what one would expect to happen) and the DOZER result as the observation (what actually happens). The user can compare area or volume over the whole domain (**Fig. 8**) or compare individual washover deposits (**Fig. 9**).

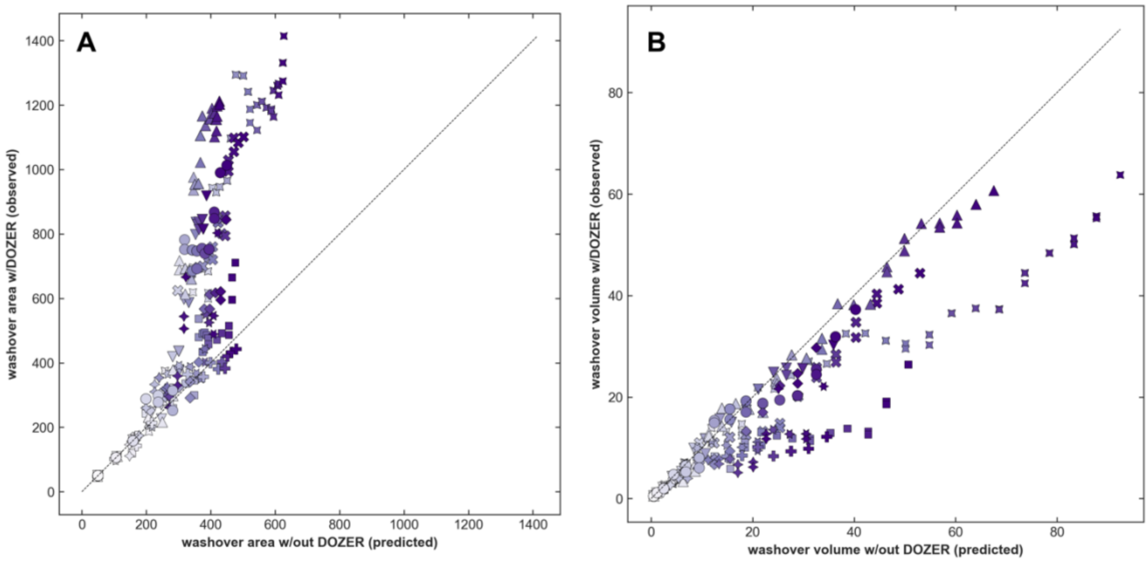

Figure 8. *Observed (DOZER) versus predicted (dummy) comparisons of total washover* ***(a)*** *area and* ***(b)*** *volume, from ten games with different initial conditions (symbols). Reference line (1:1) denotes perfect agreement. Colour darkens with elapsed game time. Plots show DOZER forces a rapid divergence in washover area, ultimately resulting in areas larger than those predicted. Washover volume is comparatively less sensitive, but DOZER forces the system toward washover volumes lower than those predicted.*

To compare individual washover deposits, a threshold contour is used (here, sand cell volume = 0.02) to delineate the footprints of deposits in the final domain of the dummy model. Each deposit is labelled with a number. Transient, evolving patterns of deposition in the timestamped DOZER and dummy domains, respectively, are then delineated with the same threshold contour and compared to the labelled footprints. Each transient deposit is assigned the label of the final dummy deposit with which it most overlaps. Deposit morphometry (area, perimeter, and volume) is calculated from the contoured footprint.

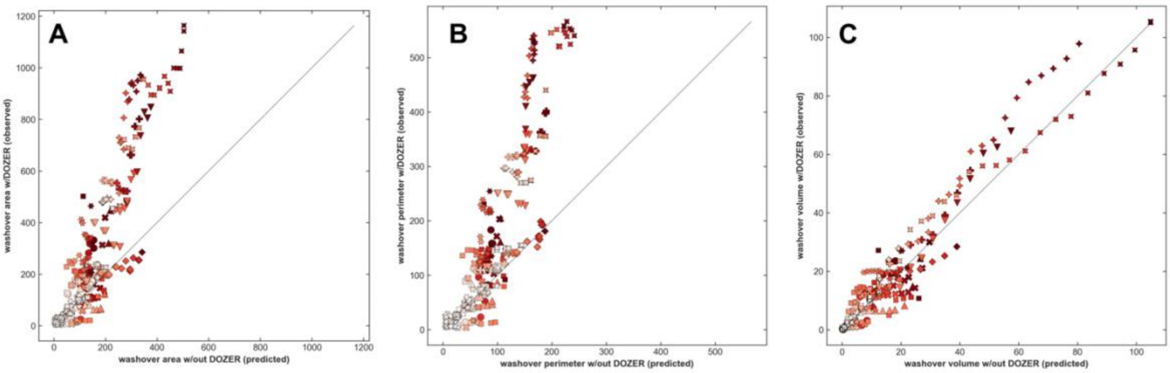

Figure 9. *Observed (DOZER) versus predicted (dummy) comparisons of individual washover deposit* ***(a)*** *area,* ***(b)*** *perimeter, and* ***(c)*** *volume, from ten games with different initial conditions (symbols). Reference line (1:1) denotes perfect agreement. Colour darkens with elapsed game time. As in* ***Fig. 8****, plots show DOZER forces a rapid divergence in washover area, ultimately resulting in areas and perimeters larger than those predicted. Washover volume shows a more complicated relationship for individual deposits. DOZER actions can pile sand into volumes larger than expected in specific regions of the domain, at the volumetric expense of other regions.*

8.2.5 Dynamical attractors

The dynamics of DOZER can also be described in terms of attractors.

For example, comparing two state variables through time – here, mean normalised dune crest elevation (the integrity of the fronting dune) versus total depositional area – demonstrates trajectories into distinct regions of the partial phase space (**Fig. 10**). Even for games with different initial conditions, trajectories for the dummy model tend to be tightly grouped and uniformly directed: the mean elevation of the dune line only decreases and depositional area only increases, and change in both variables is smooth and continuous. DOZER trajectories, however, are more excursive and recursive, as bulldozing actions complicate both the morphology of the dune line and reorganise the accommodation space of the floodplain.

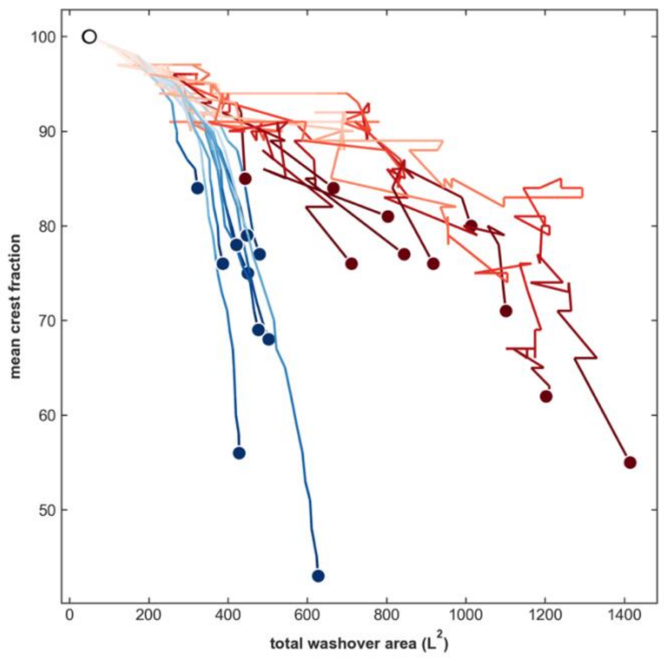

Figure 10. *Partial phase space described by mean normalised crest elevation versus total washover area, from ten games with different initial conditions. Red trajectories show DOZER results; blue trajectories show the dummy model. DOZER trajectories tend to be nonlinear and recursive, with notable variability. DOZER interventions drive the system toward comparatively higher washover areas for a given extent of crest erosion. All games start from the open circle (fully intact dune; no washover); closed circles indicate final states. Colour darkens with elapsed game time.*

The game also yields a version of the "flood enhancement through flood control" dynamic described by Criss & Shock (2001) and Werner & McNamara (2007), in which interventions to prevent frequent, minor hazard events unintentionally drive the human–landscape system toward infrequent, major hazard events (**Fig. 11**). In DOZER, if the player chooses to plow sand back into the dune gaps, the mechanic that redirects and compounds the forcing flow through neighbouring overwash sites is what affects this progressive exacerbation.

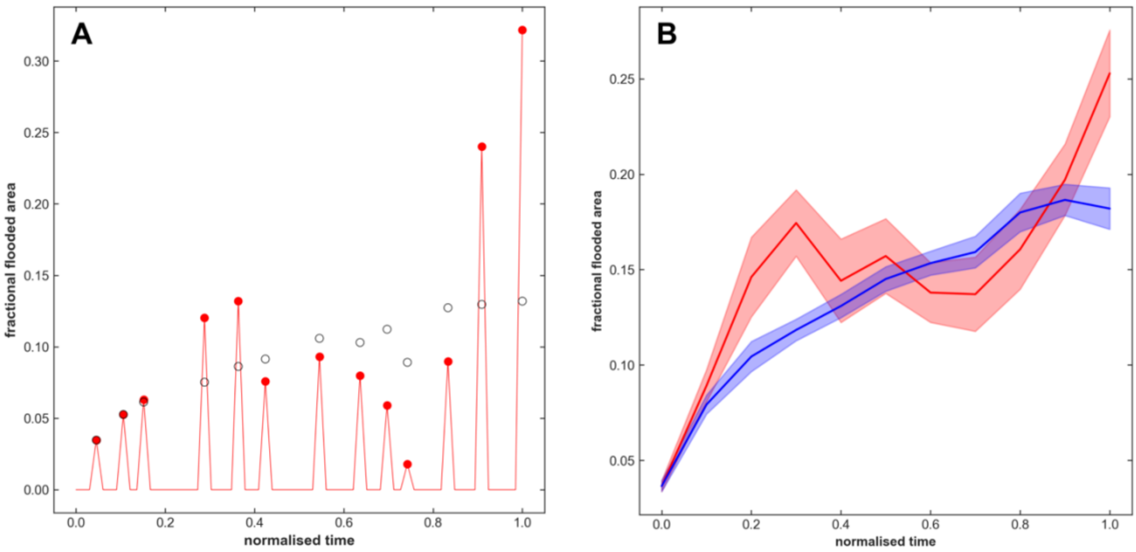

Figure 11. *Examples of the "flood enhancement through flood control" dynamic (Criss & Shock, 2001; Werner & McNamara, 2007) in DOZER.* ***(a)*** *Fractional flooded area of the domain per overwash pulse for a single game. Red circles denote DOZER condition; open circles denote dummy model.* ***(b)*** *Ensemble results from ten games with different initial conditions. Red curve denotes DOZER condtions, blue curve denotes the dummy model; envelopes capture ± 1 SE.*

8.2.6 Fixed versus randomised conditions

Users can toggle on or off a line of code in the *main.py* file that allows the script to generate a new set of initial conditions for each game, or fixes a single set of conditions for reproducibility. The dynamics that emerge from randomised and fixed-conditions games are the same (**Fig. 12**). Even repeated plays of the same fixed conditions generates different results, because DOZER actions are inevitably different in each game.

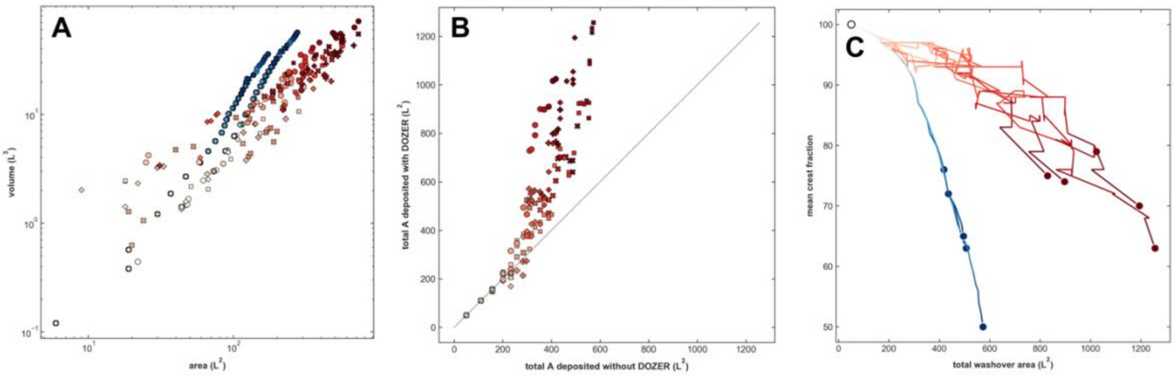

Figure 12. *Examples of game output from five games (symbols) with the same initial conditions.* ***(a)*** *Washover deposit volume versus area for DOZER (reds) and dummy (blue) conditions (see* ***Fig. 7****).* ***(b)*** *Divergence between predicted (dummy model) and observed (DOZER) washover area; 1:1 line denotes perfect correspondence (see* ***Fig. 9****).* ***(c)*** *DOZER (red) and dummy (blue) trajectories in partial phase space defined by mean normalised mean crest elevation and total washover area (see* ***Fig. 10****). In each panel, colour darkens with elapsed game time.*

8.2.7 Path strategy

DOZER cannot capture why a player chose to plow sand in one area versus another, but the model can record where DOZER moved (and when and how much they plowed), which provides at least a narrow window into player behaviour. During game play the model logs DOZER position (*x*, *y*) on the floodplain, much the way GPS is used to track the movement patterns of heavy machinery for insight into construction and mining operations (Fu et al., 2017). These positions can be mapped to show where a player directed DOZER as the integrity of the dune line deteriorated (**Fig. 13**).

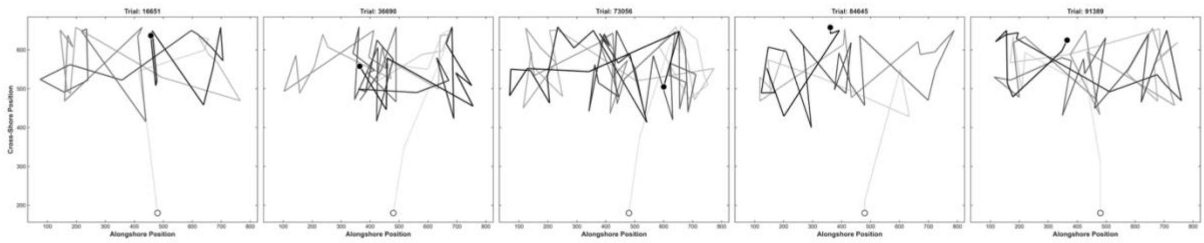

Figure 13. *Examples of DOZER position, captured every 2.5 seconds, from five different plays of the same game condition. Open (white) and closed (black) circles indicates initial and final positions, respectively; colour indicates elapsed game time.*

For repeated attempts a given condition, mapping DOZER positions over many trials reveals aggregate patterns of movement, and can be interpreted as an indicator of emergent strategy

(**Fig. 14**) – even if the player's logical motivation behind that strategy is unknown, and regardless of what a given strategy might achieve. For example, one player might labour to keep the dune intact; another might focus on keeping the road clear. Different player strategies may open regions of the model phase space otherwise inaccessible to deterministic, probabilistic, and genetic approaches.

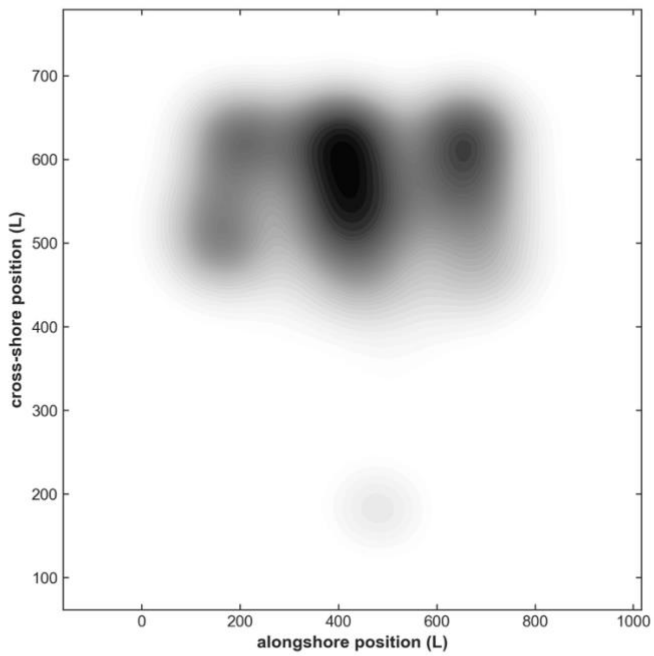

Figure 14. *Bivariate kernel density estimate (KDE) plot of DOZER positions from five games shown individually in Fig. 13. Darkness indicates a greater number of points in a given area of the domain, or the areas of the floodplain in which DOZER spent the most time.*

9. Model parameters and their effects

The default parameters used in this version of DOZER are tuned for game aesthetics, not for morphodynamic fidelity. (Even so, the model produces realistic scaling relationships.) These parameters generate washover deposits of a size that gradually fills the specified screen without overwhelming it: the deposits develop at a rate that is faster than, but broadly commensurate with, the rate at which DOZER can respond.

There are a number of ways in which the user can adjust the geometry and scale of washover deposits. Some of these adjustments can be considered as proxies for grain size. While the model does not include any explicit dependence on grain size, the version described here assumes a sandy barrier. To simulate washover more typical of a gravel barrier, for example, the user would tune the defaults to generate blunter, rounder deposits. A non-exhaustive summary of parameter adjustments and their effects is provided in **Table 5**.

10. End-game

The "whack-a-mole" mechanic, which also drives the dynamic of pulse exacerbation, is a design choice: an element that lends dramatic tension to the game, and ensures that it ends. If DOZER plugged gaps in the dune line just enough to reduce overwash flow but never fully stop it, the model would cycle forever and the player would eventually quit – out of necessity or boredom. (In its current form, a typical game of DOZER lasts 2–4 minutes.) An alternative set of rules could have made DOZER completely effective at repairing the dune in real time, and the game would end once DOZER had plowed back all the washover sand from the floodplain. Both alternatives might be fine for an autonomous model, but would lack the requisite tension to hold a player's attention.

Data availability

Code for DOZER model and analytics is freely available under MIT license (Lazarus, 2025). All game art is original by the author. Game audio is available under CC0 license (see README file in "audio" folder in project file directory.

Acknowledgements

I am grateful to Evan Goldstein, Alida Payson, Dylan McNamara, Darren Page, Chris Cooper, Ben Weeks, Steve Darby, Justin Sheffield, and Alvise Finotello for helpful discussions, and to Marjorie Lazarus for beta-testing. Web tutorials by Christian Koch (@clearcode) for coding in *pygame* were immensely instructive. This work was supported in part by NERC grant NE/X011496/1.

References

Criss, R. E., & Shock, E. L. (2001) Flood enhancement through flood control. *Geology*, 29(10), 875–878.

Donnelly, C., Kraus, N., & Larson, M. (2006) State of knowledge on measurement and modeling of coastal overwash. *Journal of Coastal Research*, 22(4), 965–991.

FitzGerald, D. M., Fenster, M. S., Argow, B. A., & Buynevich, I. V. (2008) Coastal impacts due to sea-level rise. *Annual Review of Earth and Planetary Science*, 36(1), 601–647.

Fu, J., Jenelius, E., & Koutsopoulos, H. N. (2017) Identification of workstations in earthwork operations from vehicle GPS data. *Automation in Construction*, 83, 237–246.

Haff, P. K. (2010) Hillslopes, rivers, plows, and trucks: mass transport on Earth's surface by natural and technological processes. *Earth Surface Processes and Landforms*, 35(10), 1157–1166.

Jögi, P., & Sornette, D. (1998) Self-organized critical random directed polymers. *Physical Review E*, 57(6), 6936–6943.

Lazarus, E. D. (2016) Scaling laws for coastal overwash morphology. *Geophysical Research Letters*, 43(23), 12113–12119.

Lazarus, E. D. (2025) DOZER release. GitHub repository: https://github.com/envidynxlab/dozer_release [accessed July 2025]

Lazarus, E. D., & Armstrong, S. (2015) Self-organized pattern formation in coastal barrier washover deposits. *Geology*, 43(4), 363–366.

Lazarus, E. D., & Goldstein, E. B. (2019) Is there a bulldozer in your model? *Journal of Geophysical Research: Earth Surface*, 124(3), 696–699.

Lazarus, E. D., Davenport, K. L., & Matias, A. (2020) Dynamic allometry in coastal overwash morphology. *Earth Surface Dynamics*, 8(1), 37–50.

Lazarus, E. D., Goldstein, E. B., Taylor, L. A., & Williams, H. E. (2021) Comparing patterns of hurricane washover into built and unbuilt environments. *Earth's Future*, 9(3), e2020EF001818.

Lazarus, E. D., Williams, H. E., & Goldstein, E. B. (2022) Volume estimation from planform characteristics of washover morphology. *Geophysical Research Letters*, 49(22), e2022GL100098.

Werner, B. T., & Mcnamara, D. E. (2007) Dynamics of coupled human-landscape systems. Geomorphology, 91(3-4), 393–407.

Table 1. Key user-defined parameters in DOZER.

Variable	Coded	Default	File	Notes
random seed	*r_seed*	19680807	*settings.py*	set a specific seed to reproduce a specific set of game conditions
number of columns in domain	*COLS*	48	*settings.py*	width of domain, in grid cells
number of rows in domain	*ROWS*	36	*settings.py*	height of domain, in grid cells
number of rows used in directed random walk subroutine	*ROWS_DRW*	100	*settings.py*	sets number of steps used in directed random walk subroutine to determine overwash sites; a low number (e.g., < 20) will yield a greater number of overwash sites
size of game grid tiles	*tile_size*	20	*settings.py*	side length of a game grid cell in pixels
screen width	*SCREEN_WIDTH*	*960*	*settings.py*	game screen width, in pixels – note that default is a 4:3 aspect ratio, typical of arcade screens
screen height	*SCREEN_HEIGHT*	*720*	*settings.py*	game screen height, in pixels
height of dune crest	*H*	4	*settings.py*	user's choice; higher *H* means a greater volume of sediment in the dune, and therefore more sediment flux onto floodplain (and vice versa)
domain roughness	*Rmax*	0.6**H*	*settings.py*	higher roughness coefficient lends a more rugose perimeter to the washover deposits (high perimeter-to-area ratio); lower value results in smoother-edged lobes (low perimeter-to-area ratio)
sediment lag	*Vmin*	0.02	*settings.py*	lower values result in washover deposits large in area but low in volume; higher values result in blunter, deeper deposits
threshold water depth through overwash site	*thresh*	2**Vmin*	*settings.py*	minimum water depth through overwash site to initiate activity

danger score counter	*danger*	99	*settings.py*	a progress indicator for player; taken from the maximum percent incision along dune crest
frame count	*FPS*	60	*settings.py*	frames per second; default is a standard frame count; FPS clock defines the speed of the morphodynamic model processes
data collection interval	*t_collect*	2500 milli-seconds	*settings.py*	interval at which game play data are collected (in milliseconds)
speed at which DOZER rotates	*rotation*	1 radians	*player.py*	DOZER rectangle rotates to right or left by 1 radian with player input
speed at which DOZER moves forward or backward	*velocity*	150 pixels	*player.py*	DOZER centroid moves forward or backward by 150 pixels with player input
plow carrying capacity	*blade_MAX*	1	*main.py*	maximum sand capacity for DOZER plow blade
width of overwash site	*M*	≥ 2	*morphodynamics.py*	Gaussian kernel window; number of cells (integer) in the output window; coded as (1 + *temp_inc*), where initial *temp_inc* = 1 (shape widens over time)
overwash site shape parameter	*p*	1.5	*morphodynamics.py*	Gaussian kernel window shape parameter, where *p* = 1 is Gaussian; *p* > 1 makes blunter curve
overwash site standard deviation	*sig*	≥ 2	*morphodynamics.py*	Gaussian kernel window standard deviation (sigma), coded as (2 + *temp_perc*), where initial *temp_perc* = 0.1 (shape widens over time)
time between overwash pulses	*period*	3–10 seconds	*main.py*	random interval (in seconds) between successive overwash pulses

Table 2. DOZER data output files.

File	**Notes**
gamedata	Returns a variety of game variables stored at regular intervals during game play (see **Table 3**)
Qmove_series	Returns information regarding overwash pulses (see **Table 4**)
final_sand (D, ND)	domain-sized array of final distribution of sand volume per grid cell at the end of play
waterlines (D, ND)	array in which each row represents berm height at time of each overwash pulse (e.g., **Fig. 3**)
temp_sand (D, ND)	domain-sized array of distribution of sand volume per grid cell recorded at regular intervals of play (e.g., **Fig. 4**)
ow_footprint (D, ND)	domain-sized binary array of 'wet' grid cells during an overwash event, where wet = 1
ow_sand (D, ND)	domain-sized array of sand transported per grid cell during an overwash event
screenshots	optional screenshot *.png* files during game play and/or overwash pulses

Table 3. Variables exported in '…_gamedata.csv'

Variable	**Header**	**Notes**
time stamp	*datetime_id*	datetime identifier in 'YYYYMMDD-HHMMSS'
random seed	*randseed*	records random seed used to initialise game conditions, for reproducibility
trial ID	*trial*	randomised trial identifier for each game
sediment lag	*Vmin*	minimum volume of sediment left behind in a given grid cell by overwash (see **Table 1**)
flow depth threshold	*threshold*	minimum water volume (elevation difference) between water level of forcing flow and dune crest height for flow to transport sediment
height of dune crest	*H*	height of dune crest, initially uniform alongshore (see **Table 1**)
game clock	*run_time*	running time elapsed during game, in seconds
DOZER position in *x*	*dozer_x*	Cartesian *x* position of centre of DOZER sprite rectangle, in units pixels (i.e., relative to game screen width and height – see **Table 1**)
DOZER position in *y*	*dozer_y*	Cartesian *y* position of centre of DOZER sprite rectangle, in units pixels (i.e., relative to game screen width and height – see **Table 1**)
volume of sand actively being moved by DOZER at that time step	*dozer_Qs*	captures whatever volume of sand is actively on DOZER plow blade at time of data collection
cumulative total volume of sand moved by DOZER	*dozer_Qs_tot*	cumulative total volume of sand moved by DOZER; difference of *dozer_Qs_tot* will yield a time series of plowing activity more complete than *dozer_Qs*
total volume of sand on floodplain (DOZER condition)	*washover_V_D*	total volume of sand on floodplain (DOZER condition)
total volume of sand on floodplain (dummy condition)	*washover_V_ND*	total volume of sand on floodplain (dummy condition)
total area of sand on floodplain (DOZER condition)	*washover_A_D*	total area of sand on floodplain (DOZER condition), in number of grid cells

total area of sand on floodplain (dummy condition)	*washover_A_ND*	total area of sand on floodplain (dummy condition), in number of grid cells
flow redistribution among overwash sites	*lateral_disp*	fraction of total forcing flow redistributed among overwash sites as a result of plowing actions that infill incised portions of the dune crest
danger score	*danger_score*	game play indicator; reports extent of maximum incision through dune crest as a percentage of total dune height
mean dune crest height (DOZER condition)	*crest_mu_D*	mean dune crest elevation alongshore
mean dune crest height (dummy condition)	*crest_mu_ND*	mean dune crest elevation alongshore
overwash pulse time	*pulse_time*	time on running game clock when last overwash pulse occurred; can be used to mark overwash events relative to other game-play data

Table 4. Variables exported in '…_Qmove_series.csv'

Variable	**Header**	**Notes**
pulse time	*pulse_time*	Time on game clock at which overwash pulse occurs. Note that these pulse times also correspond to the arrays of dune crest elevation alongshore in the '…_waterlines.csv' file.
sand flux (DOZER condition)	*Qm_D*	Volume of sand fluxed through fronting dune and onto floodplain by a given overwash pulse – letter *m* indicates sand "to move". Note that volume shown in game-score counters is x100.
sand flux (dummy condition)	*Qm_ND*	(as above)
number of "wet" grid cells (DOZER condition)	*wet_D*	Count of total "wet" grid cells at full extent of an overwash pulse. These are cells that receive and/or redistribute overwashing sand.
number of "wet" grid cells (dummy condition)	*wet_ND*	(as above)

Table 5. Model parameters (see **Table 1**) and their effects.

Effect	Variables	Notes
larger (smaller) washover deposits; *NB* – defaults set relative to a domain array of 36 x 48 (*rows* x *columns*)	↑ (↓) *H*	changes effective forcing by increasing height of dune crest; more available sand means larger deposits on floodplain
	↓ (↑) *Vmin*	reducing the sediment lag will produce larger, thinner washover deposits; increasing the lag will produce smaller, thicker deposits
	↓ (↑) *threshold*	reducing the flow-depth threshold to move sediment through overwash sites results in sediment flux at all active sites (equal to or lower than water level); higher threshold increases time to site activation and amount of sediment moved during a given pulse (e.g., effectively fewer pulses, but more sediment per pulse)
	Gaussian window parameters (*M, p, sig*)	changing the shape of overwash incision can deliver more or less sediment to the floodplain: a wider, blunter window will yield larger washover; a narrower, peakier window will yield smaller washover
qualities of washover expression (morphometry)	↑ (↓) *Rmax*	roughness of scratch domain; higher values produce washover with more rugose perimeters (more like splays); lower values produce footprints with smoother perimeters (more like lobes)
	↑ (↓) *yv*	variable (in *morphodynamics.py*) that changes slope of scratch; default = 1; steeper slopes produce washover with greater intrusion lengths (narrower, longer, more finger-like footprints); shallower slopes produce blunter footprints
DOZER effectiveness relative to washover deposition	↑ (↓) *velocity* (also *rotation*)	faster DOZER movement makes player more responsive to washover deposition, and potentially more effective at clearing washover; could result in more exaggerated "exacerbation" dynamic (**Fig. 11**)
	↑ (↓) *blade_MAX*	increasing carrying capacity of DOZER blade makes player movements more effective at clearing sand
	timings between overwash pulses	less time between overwash pulses accelerates game, gives player less time to respond to any given pulse; more time makes the player more effective at moving sand relative to a given pulse