DRIFT — technische specificatie (solarpunk sky routing)

Nov 21, 2026 · @Lars

Doel en scope

DRIFT is een "Google Maps voor solarpunk": een webapp die voor een zonne-luchtschip de zuinigste route berekent op basis van een windvoorspelling, en die route daarna turn-by-turn navigeert. Alles speelt zich af in een fictieve kustregio, de Solenne Delta (6 × 6 km). Deze spec beschrijft het prototype zoals het nu werkt, zodat je het zelf kunt herbouwen of uitbreiden.

Wat er gebouwd moet worden:

- **Wereld**: procedurele delta met heuvelrug, rivier, stad, landbouw, eilanden, wegen en 8 aanlegtorens (hubs).
- **Windmodel**: windveld op 3 hoogtelagen (180, 320 en 460 m) dat per voorspellingsuur verandert.
- **Routing**: wind-aware A* op een 3D-grid met drie modi: Greenest, Fastest en Calmest.
- **Kaart**: twee kaarttypes. Een vectorkaart in Google-Maps-stijl (dag en nacht) en een gestileerde 3D-terreinweergave.
- **Navigatie**: autopilot of zelf vliegen, met banner, manoeuvres en automatisch herrouteren.
- **Video**: deterministische render van een conceptfilm van 39 s.

Het demoscenario is Copperleaf Farms naar Aster Island Pier, vertrek 14:00 (modeluur 4). De referentie-implementatie geeft deze uitkomsten; gebruik ze als regressietest.

Modus	Kruissnelheid (m/s)	Vluchttijd	Netto energie (kWh)	Gust-index
Greenest	11	5 min 26 s	1,21	0,43
Calmest	13	4 min 48 s	1,88	0,41
Fastest	18	3 min 37 s	3,29	0,53
Rechte lijn op 180 m (baseline)	11	—	1,95	—

Greenest gebruikt dus 38% minder energie dan een windblinde rechte lijn. Buiten scope: echte geodata, een echte weer-API, een backend en accounts. Eenheden overal: meter, seconde, m/s, kW en kWh; noorden is −Z, oosten is +X.

Tech stack en projectstructuur

Aanbevolen stack: Vite + TypeScript, React voor de UI, Three.js (of React Three Fiber) voor 3D, Zustand voor state en een Web Worker voor windgrids en A*. Het prototype zelf draait op TypeScript met een eigen WebGL2-engine zonder dependencies, omdat er in de bouwomgeving geen packages geïnstalleerd konden worden. De algoritmes in deze spec zijn los van de renderer en porten 1-op-1.

Laag	Prototype	Aanbevolen voor de rebuild
Taal en build	TypeScript 6, `tsc` naar ES modules	TypeScript + Vite
3D	Eigen WebGL2 (shaders, shadow map, MSAA)	Three.js `ShaderMaterial` of R3F met custom shaders
UI	Vanilla DOM + CSS	React + CSS modules of Tailwind
State	Object `S` in `main.ts`	Zustand store
Zwaar rekenwerk	Main thread (A* ≈ 70 ms per route)	Web Worker met transferable `Float32Arrays`
Tests	Node-scripts die routes vergelijken	Vitest voor `wind`, `router`, `nav`; Playwright voor de UI
Video	Headless Chrome + ffmpeg	Zelfde aanpak

Mappenstructuur (zelfde opdeling aanhouden):

- `engine/` — `math.ts` (mat4, projectie), `geometry.ts` (low-poly MeshBuilder), `gl.ts` (programs, meshes, shadow map).
- `world/` — `noise.ts` (geseede Perlin, fbm), `terrain.ts` (hoogte, landgebruik), `world.ts` (objecten, hubs, wegen, zones), `airship.ts` (het schip).
- `sim/` — `wind.ts` (windveld), `router.ts` (energiemodel en A*), `flight.ts` (vluchtmodel en autopilot).
- `render/` — `scene.ts` (renderpasses en thema's), `shaders.ts` (GLSL), `fx.ts` (windsporen, streaks).
- `app/` — `core.ts` (bootstrap), `nav.ts` (manoeuvres), `main.ts` (UI-prototype).
- `video/` — `director.ts` (tijdlijn van de film).

Alles is deterministisch: zelfde seed geeft dezelfde wereld, wind en routes. Gebruik overal een geseede RNG (mulberry32) en nooit `Math.random()`.

Wereldmodel

De wereld is een vierkant servicegebied van 6 × 6 km (`H = 3000`, coördinaten van −3000 tot 3000 m) met daarrond een grove rand tot ±9 km voor de horizon. Hoogte is een pure functie `rawHeight(x, z)`; het mesh en alle queries gebruiken dezelfde functie.

Terreinmesh. Binnen: 192 × 192 cellen van 31,25 m. Buiten: niet-uniforme as-coördinaten (−9000, −7000, −5600, −4600, −3950, −3550, −3300, −3150, −3060) zodat er geen naden zijn. De diagonaal wisselt per cel; `heightAt()` interpoleert op exact dezelfde driehoek, zodat objecten nooit zweven. Flat shading, kleur per driehoek uit het biome.

Hoogtefunctie (nx = x/H, nz = z/H), in deze volgorde:

1. Land-masker: `land = −0,48·nx − 0,82·nz + 0,2 + 0,24·fbm(1,3·nx + 3,1; 1,3·nz − 1,7)`, min de baai `0,34·gauss(x−1050, z−1200, r = 760)`. Land als `land > 0`: `h = 135·land` plus heuvels `22·fbm(5,5·nx, 5,5·nz)`. Zee: `h = max(−70, 380·land)`.
2. Windmill Ridge: `215·exp(−(d/560)²)·(0,75 + 0,45·ridged)` met d de afstand tot de ridge-polyline.
3. Drie eilanden: `h = max(h, −32 + (topH + 32)·exp(−2,3·d²/r²))`.
4. Stadsplateau rond (330, 380), straal 720: blend naar 7 m.
5. Zoutmoeras rond (−1050, 1180), straal 620: blend naar 0,9 + 2,6·fbm, wat vijvers geeft.
6. Rivier (Silverthread): vallei verbreden (`h ×= 0,45…1` tussen 90 en 650 m) en een geul van −3,5 m binnen 28–85 m van de polyline.

Landgebruik (`landuse(x, z)` → farm, city, forest, marsh, elk 0…1) bepaalt kleuren, objectplaatsing en de kaartstijl. Bak het in een RGBA-texture van 512².

Objecten: stadsblokken op een grid van 78 m (hoek 0,21 rad) met terrastuinen en zonnedaken, de Sunspire-toren (172 m), ongeveer 2.500 bomen, windturbines elke 340 m op de ridge, zonneparken in de landbouwzone, kassen, dorpen, boten, een vuurtoren en bruggen. Bewaar van elk gebouw een footprint `{x, z, w, d, h, yaw, round}`; de kaartstijl extrudeert die.

Hubs (aanlegtorens van 58 m, top = grond + 58):

Hub	x (m)	z (m)	Gebied
Copperleaf Farms	1250	−1680	North Plains
Aster Island Pier	1860	1650	Aster Island
Lumen Bay Central	560	250	Lumen Bay
Mossgate Dock	−1880	−230	Fern Valley
Windmill Ridge Station	−1780	−1630	Windmill Ridge
Kelp Docks	1700	−520	East Shore
Heron Point	−2350	1050	Saltmarsh Coast
Saltglass Isle	2540	400	Outer Isles

No-fly zones: Saltmarsh Bird Sanctuary (centrum −1050, 1180; straal 560 m; alle hoogtes) en de Lumen Bay quiet zone (290, 410; straal 430 m; verboden onder 260 m). Gemelde gust-cellen komen er tijdens het vliegen bij (max. 4 zones in de shader).

Wegen (alleen voor de kaartstijl): klasse 0 snelweg, 1 hoofdweg, 2 straat, 3 landbouwpad. Hoofdwegen = controlepunten + Perlin-wiggle + Catmull-Rom, geknipt waar het terrein onder −0,5 m zakt (behalve binnen 140 m van de rivier: bruggen). Verder het stadsraster, landbouwpaden op het veldraster en lussen rond dorpen. Wegnummers: N12, N4, N9, R21, R7, R3, R15 en R8.

Windmodel

Het windveld is een som van zes analytische componenten, voorgerekend op een grid van 100 × 100 cellen (60 m) per hoogtelaag (180, 320 en 460 m). Per cel worden `wx`, `wz` (m/s, richting waarheen de wind waait) en een gust-index 0…1 opgeslagen. Het voorspellingsuur in het model is UI-uur + 4; `compute(hour)` herberekent alle grids (< 100 ms).

Componenten (p = eenheidsvector van de grondwind, h = modeluur):

1. **Heersende richting**: hoek `atan2(−0,45; −1) + 0,3·sin(0,42·h + 0,6)`, dus een zeebries naar het west-noordwesten.
2. **Zeebriescel**: aan de grond `p × 3,4 m/s`; in de hoogte een *return flow* van 7,2 m/s, p gedraaid over π − 0,38 rad. Mengfactor `smoothstep(190, 450, alt)`. Hierdoor hangt de beste hoogte af van je koers: zeewaarts hoog, landinwaarts laag.
3. **Wervels (curl noise)**: stroomfunctie `ψ = 2500·fbm(x/1700 + 0,09h, z/1700 − 0,06h)`, snelheid `(∂ψ/∂z, −∂ψ/∂x)` met centrale differenties (stap 45 m). Divergentievrij, dus realistisch ogende draaiingen.
4. **Silverthread Jet**: een dalwind langs de rivier, doorgetrokken over de baai tot (2250, 1880). Sterkte `9 m/s × exp(−(d/400)²)`, langs de richting van het dichtste segment, afnemend met hoogte (`1 − 0,9·smoothstep(170, 470, alt)`).
5. **Ridge speed-up**: factor `1 + 0,45·exp(−max(0, vrije hoogte − 60)/140)` boven hoog terrein.
6. **Squall line**: een band loodrecht op p, positie `−4200 + 720·h` m, profiel `exp(−(s/380)²)`, voegt 4,5 m/s toe. Ze trekt over de kaart tussen ongeveer +2 h en +9 h.

Gust-index = 0,2 + noise (0,32) + squall (0,62) + jet (0,2) + lijzijde van de ridge (0,42, onder 250–520 m) + stad (0,22, onder 180–360 m), vermenigvuldigd met `1,08 − alt/1600`, geklemd op 0…1.

Bewolking (voor zonne-opbrengst en wolkschaduw): een tileable fbm-texture van 128² over 6,4 km die met de return flow meedrijft (offset `−p × 6 m/s × t`).

API

Functie	Doet
`compute(hour)`	Herbouwt de drie grids voor een voorspellingsuur
`sample(x, z, alt)`	Bilineair per laag, lineair tussen lagen; onder 180 m snelheid × alt/180 (min. 0,35)
`sampleLayer(l, x, z)`	Bilineair in één laag (voor de windsporen)
`cloudAt(x, z, ox, oz)`	Bewolking 0…1 met drift-offset
`prevailing(h)`, `squallPos(h)`	Voor UI-teksten ("squall line due in 2 h")

Hou wind en terrein strikt gescheiden van rendering: de router, de autopilot en de windsporen lezen allemaal dezelfde `sample()`.

Energiemodel en routing

De router is een A* over een 3D-grid (100 × 100 cellen × 3 lagen = 30.000 nodes) waarbij elke edge gekost wordt door een stukje vlucht door het windveld te simuleren. De drie modi verschillen alleen in kruissnelheid en kostfunctie.

Energiemodel (constanten in `ENERGY`): accu 180 kWh, start op 86%.

```
P(v, g) = 3 + 0{,}0085\,v^3 + 6\,g \quad [\mathrm{kW}]
```

```
S(c) = 9 \cdot 0{,}75 \cdot (1 - 0{,}78\,c) \quad [\mathrm{kW}]
```

P is het verbruik bij luchtsnelheid v en gust-index g; S is de zonne-opbrengst bij bewolking c. Klimmen kost 0,12 kWh per 100 m (ballonets), dalen 0,04 kWh per 100 m.

Grondsnelheid langs eenheidsrichting d met wind w en luchtsnelheid v: ontbind w in een component langs d en een dwarscomponent. Is de dwarswind groter dan v, dan is de edge onhaalbaar. Anders:

```
v_g = (\vec w \cdot \vec d) + \sqrt{v^2 - |\vec w_\perp|^2}
```

Is v_g < 2,5 m/s, probeer dan opnieuw met v + 6 m/s (gas geven tegen de wind in, aan hoger verbruik). Blijft v_g onder 1,5 m/s, dan is de edge geblokkeerd.

Modi en kostfuncties (een edge wordt in stukken van max. 60 m geïntegreerd, dt = stuklengte / v_g):

Modus	v (m/s)	Kost per stuk	Verticale edge
Greenest	11	(P·dt − S·dt)/3600 + 0,25·g²·km + klimenergie	klimenergie + 0,02
Fastest	18	dt (+ klimhoogte/12 s)	0,2 × Δh/3
Calmest	13	dt · (1 + 7·g²)	0,2 × Δh/3

Grid en buren. Een cel is geblokkeerd als `max. terrein + 75 m > laaghoogte`, als ze in een no-fly zone ligt onder de maxhoogte van die zone, of aan de rand van het grid. Buren: 16 horizontale richtingen (8 + paardensprongen, met corner-cut-check) plus één laag omhoog of omlaag. Heuristiek: afstand × minimale kost per meter, met een maximale meewind van 16 m/s (Greenest: `(P(v,0) − S(0)) / 3600 / (v + 16)`, de andere: `1 / (v + 16)`). Start en doel zijn de dichtste vrije node bij de hub, laagste laag eerst. Bewaar de volgorde van geëxpandeerde nodes: de video toont die als scananimatie.

Nabewerking van het pad, in deze volgorde:

1. Kostbewuste string pulling: sla nodes over als het rechte stuk vrij is én hoogstens 0,4% duurder (venster 14 nodes, niet over een laagwissel heen).
2. Begin en eind aan de torentop (top + 6 m) met een tussenpunt als oprit.
3. Chaikin-smoothing, 3 iteraties; daarna resamplen op 20 m.
4. Hellingslimiet 0,24 (een voorwaartse en een achterwaartse pass die punten enkel verlagen), dan terreinvrijheid van minstens 40 m.

Metrics (`evaluate`) over het gladde pad: vluchttijd (dt = max(L/v_g, |Δh|/3,5)), verbruik, zonne-opbrengst, netto energie = max(0, verbruik − zon), tijdsgewogen gust-index, gemiddelde meewind, max.

dwarswind, klimhoogte en afstand. De **baseline** is een windblinde rechte lijn op max(180 m, terrein + 80 m) aan 11 m/s, met dezelfde hellingslimiet; de UI toont "−X% vs straight line" t.o.v. die waarde.

Herrouteren gebruikt dezelfde `route()` met een tijdelijke pseudo-hub `{x, z, top = huidige hoogte}` als vertrekpunt.

Vluchtmodel en autopilot

Het schip is een puntmassa met luchtsnelheid, koers en verticale snelheid; de wind verplaatst het. Daardoor vliegt het schip schuin (crab) om zijn grondspoor te houden, wat je in beeld ziet.

Toestand (`Ship`): `pos`, `heading` (kompaskoers, 0 = noord, met de klok mee), `track` (richting van de grondsnelheid), `airspeed`, `target`, `vs`, `roll`, `pitch`, `battery`, `used`, `harvested`, `distance`, `time`, `pathIndex`, `docked`, `warn`.

Stap `step(dt, controls)`, in substappen van max. 0,1 s:

Grootheid	Regel
Doelsnelheid	`target += throttle · 4 · dt`, geklemd op 0…22 m/s
Luchtsnelheid	naar target met max. +1,2 m/s² en −1,6 m/s²
Draaisnelheid	manueel `yaw · 0,24` rad/s; autopilot `clamp(Δkoers · 0,9; ±0,22)`; gladgestreken (factor 3/s)
Verticale snelheid	manueel `climb · 4` m/s; autopilot `clamp((doelhoogte − y) · 0,35; ±4)`; gladgestreken (1,5/s)
Grondsnelheid	`airspeed · (sin h, −cos h) + wind(x, z, y)`
Grenzen	min. 28 m boven terrein (waarschuwing "TERRAIN"), plafond 900 m, waarschuwing in een no-fly zone
Vermogen	`P(airspeed, gust) + max(0, vs) · 4,3` kW; zon `S(bewolking)`; accu `−= (P − S)·dt/3600`
Visueel	roll naar `−0,9 · draaisnelheid`, pitch naar `0,012 · vs`, propellers draaien met de snelheid mee

Autopilot `autopilot(dt, path, speed)`:

1. Zoek het dichtste padpunt vooruit vanaf `pathIndex` (venster 60 punten) en kies een lookahead-punt 7 punten verder (≈ 140 m).
2. Gewenst grondspoor d = richting naar dat punt. Luchtsnelheid = snelheid van de modus, + 6 m/s als de grondsnelheid onder 2,5 m/s zakt.
3. Binnen 260 m van het doel vertragen naar `speed · afstand/260`, minimaal 6,5 m/s.
4. Crab-hoek: luchtvector `a = v_g·d − w`, koers = `atan2(a.x, −a.z)`. Doelhoogte = hoogte van het lookahead-punt.
5. Aangelegd als het schip binnen 30 m horizontaal en 25 m verticaal van de torentop is; zet het dan vast op torentop + 3 m.

Simulatiesnelheid in de UI: 1×, 5×, 15× (standaard) en 40×. De vlucht van het demoscenario duurt 5 min 24 s (gevlogen, iets meer dan de geplande 5 min 26 s door de autopilot) en verbruikt 1,13 kWh netto.

Het luchtschip (eigen ontwerp): ellipsoïde envelop van 45 m lang (straal 8,6 m) met zonnefolie op de bovenkant, vier staartvinnen in X-opstelling, een bootvormige gondel van 23 m met daktuin en cabine, twee propellers op uitleggers en een duwpropeller achteraan. Oorsprong = dek van de gondel, voorwaarts = −Z. In de kaartstijl wordt het schip opgeschaald (× 1,3 tot 9, volgens camera-afstand) en op 20% van de echte hoogte getekend, met een zachte schaduw eronder.

Navigatie

Navigatie zet het gladde pad om in een lijst manoeuvres `{s, idx, kind, text, sub, name, pos}`, waarbij `s` de afstand langs het pad is (horizontaal, in meter). Voor het demoscenario komen er zes uit: Head south-west, Climb to Skyway 320, Bear left toward Sunspire, Join the Silverthread Jet, Descend to Skyway 180 en Dock at Aster Island Pier.

Namen. Hoogtelagen heten "Skyway 180", "Skyway 320" en "Skyway 460" (dichtste laag). Binnen 260 m van de jet-corridor en onder 400 m heet de weg "Silverthread Jet". Die naam staat in de pil onder het schip, zoals een straatnaam.

Extractie (`buildManeuvers`):

1. **Vertrek** op s = 0: "Head <kompasrichting>", sub "Leave <hub> and climb".
2. **Bochten**: venster van 6 punten (120 m). `Δ = koers(i→i+6) − koers(i−6→i)`. Kandidaat als |Δ| > 0,5 rad en lokaal maximum binnen ±6 punten; minstens 260 m na de vorige bocht, niet in de eerste 160 m of laatste 180 m. Soort: > 1,9 rad sharp, > 0,95 rad turn, anders slight ("Bear"). Tekst "Turn left toward <plaats>": de hub of het gebied op 400–4200 m in de nieuwe richting (cos > 0,82, score cos/(1 + d/2500)). Sub: meewind of tegenwind in m/s.
3. **Hoogtewissels**: als de hoogte over 12 punten meer dan 45 m verandert. Tekst "Climb to Skyway 320", sub met de doelhoogte en de extra meewind daarboven (anders "smoother air above").
4. **Jet**: het eerste punt in de corridor: "Join the Silverthread Jet", sub met de meewind langs het spoor.
5. **Aankomst**: "Dock at <hub>", sub met de hoogte van het platform.
6. Sorteer op s en voeg events samen die minder dan 140 m uit elkaar liggen; hou het belangrijkste (aankomst > vertrek > jet > hoogte > bocht).

Runtime

Element	Regel
Positie op de route	dichtste padindex in het venster `pathIndex − 30 … +90`
Banner	volgende manoeuvre met s > s_schip + 15 m
Afstand	≥ 1000 m in km met 1 decimaal; ≥ 100 m afgerond op 50 m; anders op 10 m
"Then"-kaartje	als de manoeuvre daarna binnen 700 m volgt

Element	Regel
Callout op de kaart	naam van de volgende weg, op het manoeuvrepunt, binnen 1,6 km
Resterende tijd	geplande `times[]` van het pad; ETA = 14:00 + vertrekuur + verstreken + resterend
Spraak (optioneel)	`speechSynthesis` zodra de volgende manoeuvre binnen 450 m komt

Herrouteren. Ligt het schip meer dan 190 m naast het pad gedurende ± 1,2 s (geschaald met de simulatiesnelheid), dan wordt een nieuwe route berekend vanaf de huidige positie en hoogte. De banner wordt 1,6 s grijs met "Rerouting…". De vertrekmanoeuvre valt weg in de nieuwe lijst.

Gust melden. De knop met de gele driehoek legt een tijdelijke zone (straal 260 m) op het padpunt ± 700 m vooruit, herbouwt de router met die zone en herroutert meteen. Op de kaart verschijnt een geel windicoon.

Aankomst. Banner wordt groen: "Docked at <hub>" met het verbruik, de zonne-opbrengst en de besparing t.o.v. de rechte lijn. De Exit-knop wordt "Done".

Rendering

Er is één renderer met twee stijlen die via een uniform `uMap` (0 = 3D-terrein, 1 = vectorkaart) in elkaar overvloeien; het wisselen duurt ± 0,3 s. Elke shader berekent beide kleuren en mengt ze, zodat de overgang vloeiend is (de video gebruikt dat als effect).

3D-terreinlaag

- Licht: hemisferisch ambient + zon (richting (0,5; 0,62; 0,62) genormaliseerd, kleur `#ffeccd` × 2,35), rim light, two-sided normals.
- Schaduw: shadow map van 2048², 3×3 PCF, licht-frustum rond het camera-doel (halve breedte 1,15 × camera-afstand, 140–4600 m), gesnapt op texels tegen flikkeren. Plus wolkschaduw uit de bewolkingstexture, geprojecteerd langs de zon.
- Mist met hoogtedemping, ACES-tonemapping, lichte S-curve, saturatie 1,06. Lucht: verloop `#3f95d1` naar `#cadfe0` met zonneschijf.
- Water: som van sinusgolven + noise-normaal, fresnel, diepte uit een hoogtetexture (R16F, 512² over 12 km) en schuim aan de kust.
- Windsporen: 4.200 deeltjes die in `sampleLayer` meedrijven, getekend op een 2D-canvas van 1024² dat elke frame 8,5% vervaagt. Kleur per snelheid (< 6, < 10, < 14, ≥ 14 m/s). Het canvas wordt als texture over terrein en water gelegd.
- Routes: een buis waarvan de straal een uniform is (blijft zichtbaar bij elke zoom), een doorzichtig gordijn naar de grond en bewegende streepjes.

Kaartstijl (Google-Maps-look)

- Terrein: alle vertices naar y = 0,6. Kleur uit de landgebruik-texture (drempels `smoothstep(0,35; 0,65)`), kustlijn uit de hoogtetexture met `fwidth`-antialiasing. Geen belichting.
- Wegen: per klasse één ribbon-mesh. Elke vertex bevat het middenpunt in `aPos` en de miter-vector in `aNor`; de vertex shader blaast op: `pos + aNor · breedte`. Breedte = `max(5,5;`

`0,0042 · camera-afstand`) × [1,25; 0,95; 0,7; 0,42] per klasse, zodat wegen even breed blijven op het scherm. Eerst alle randen (+0,42), dan alle vullingen, zonder depth write.

- Gebouwen: dozen en cilinders uit de footprints, dak en zijkant in themakleur; `uExtrude` (0…1) laat ze uit de grond groeien.
- Routes: platte ribbons, als laatste getekend zonder depth test zodat ze boven alles liggen. Breedte `max(7; 0,0068 · afstand)`, rand × 1,32. Geselecteerd in themablauw, alternatieven grijs en 0,8× zo breed. Het afgelegde stuk verdwijnt via `uStart`.
- Waas naar de horizonkleur vanaf 35% van `max(4200; 3,4 × afstand)`.

Thema's

Element	Nacht	Dag
Land	`#1f3c55`	`#efece5`
Stad	`#283e63`	`#e7e3db`
Park en bos	`#1b4752`	`#c8e8c4`
Landbouw	`#22445b`	`#eef1d9`
Water	`#0c2442`	`#9fd4f6`
Strand	`#2a4964`	`#f6edcf`
Gebouw dak / zijkant	`#506697` / `#36497a`	`#f4f1ec` / `#d8d3cb`
Snelweg vulling / rand	`#8193d2` / `#33468c`	`#fdd982` / `#e2b452`
Straat vulling / rand	`#4f6299` / `#253563`	`#ffffff` / `#dbd6ce`
Lucht boven / horizon	`#0a1730` / `#16304e`	`#bcdcf2` / `#e6eff2`
Route vulling / rand	`#22cdff` / `#0b77a8`	`#1a73e8` / `#1456b5`
Alternatieve route	`#6c7ba8` / `#3a4870`	`#9fbdec` / `#7597d4`

Labels zijn DOM-elementen die elke frame via `project(vp, p)` gepositioneerd worden: gebiedsnamen, waternamen (cursief), wegschilden (N12 enz., op 30% en 72% van de weg), hub-iconen (vertrek = witte ring, bestemming = rode pin), tijdsbubbels per route, de wegpil onder het schip en de callout bij de volgende manoeuvre. Een routebubbel staat op het punt dat het verst van de andere routes én van de al geplaatste bubbels ligt.

UI-specificatie

De app heeft twee schermen, Plannen en Navigeren, en volgt de conventies van Google Maps (zijpaneel met routes, tijdsbubbels, turn-by-turn banner, ronde knoppen, bottom sheet) met een eigen merk: DRIFT-logo, geen Google-logo's of -iconen. Fonts: Figtree voor koppen en grote cijfers, Roboto voor tekst, Roboto Mono voor meetwaarden. Standaardthema is nacht.

Plannen (desktop: paneel links van 404 px; onder 760 px breed: bottom sheet van 62% hoogte)

- Kop: logo, "Sky routing for the Solenne Delta", knop dag/nacht.
- Van en Naar: velden die een zoekbare lijst van de 8 hubs openen, plus een wisselknop.
- Modus-chips: Greenest (blad), Fastest (bliksem), Calmest (golf), elk met de reistijd in minuten.
- Vertrek: "Leave now, 14:00" of +1 tot +8 uur. Wijzigen herberekent wind en de drie routes.
- Routelijst, één kaart per modus: reistijd, afstand en snelheid, uitleg ("Via Skyway 180, Skyway 320 and the Silverthread Jet…"), een tag ("Lowest energy, −38% vs straight line", "+69% energy" of de gust-index) en rechts netto kWh en gemiddelde wind. Klikken selecteert.
- Windnotitie onderaan: richting van de zeebries en wanneer de squall line komt.
- Knoppen: Preview (camera vliegt 9 s langs de route) en Start.
- Op de kaart: drie routes (geselecteerd in blauw), tijdsbubbels met blad-icoon voor Greenest, hub-iconen. Klik op een hub: menu "Fly from here" / "Fly to here".
- Rechtsonder: lagenknop, kompas (terug naar noord) en zoom. Onderaan een legenda voor wind, no-fly en gusts.

Lagenpaneel: kaarttype Map of 3D terrain; schakelaars Wind flow, Gust risk, No-fly zones, Solar (cloud cover) en 3D buildings; keuze van de windlaag (180, 320 of 460 m).

Navigeren

- Banner linksboven (teal `#0c5a53`, max. 560 px): manoeuvre-icoon, afstand, instructie in 34 px en een subregel. Daaronder het "Then"-tabje (`#0a4a44`). Grijs bij herrouteren, groen bij aankomst.
- Rechts vijf ronde knoppen van 64 px: kompas (heading-up of north-up), zoeken (terug naar het overzicht), stem aan/uit, windsporen aan/uit, gust melden.
- Links: snelheidscirkel (grondsnelheid in km/h) en de pil "Re-centre" zodra de gebruiker de kaart versleept.
- Bottom sheet (max. 720 px): resterende minuten in groen (blad-icoon bij Greenest), "4.2 km, arrive 14:05", knop Zelf vliegen en een rode Exit-knop. Daaronder chips: accu, verbruikt, zonne-opbrengst, mee- of tegenwind, hoogte, en de simulatiesnelheid.
- Waarschuwingspil in het midden bij terrein of een no-fly zone.
- Camera: volgt het grondspoor (heading-up), afstand 520 m (kaart) of 260 m (3D), pitch 0,92 rad of 0,5 rad, kijkpunt 32% of 25% van de afstand voor het schip uit. Windsporen staan standaard uit tijdens het navigeren.

Besturing

Invoer	Actie
Linkermuis slepen / één vinger	Pannen (straal snijdt het grondvlak)
Rechtermuis of Shift + slepen	Draaien en kantelen (pitch 0,32–1,52 rad)
Scrollwiel / pinch	Zoomen naar de cursor (250–16.000 m)
Twee vingers draaien	Kaart draaien
W / S of ↑ / ↓	Gas bij / gas terug (schakelt automatisch naar Zelf vliegen)
A / D of ← / →	Links / rechts draaien

Invoer	Actie
R / F of PageUp / PageDown	Klimmen / dalen
Esc	Navigatie verlaten

Op touch-schermen verschijnt tijdens Zelf vliegen een pad met zes knoppen (gas, draaien, klimmen, dalen).

Videopipeline

De film (39 s, 1920 × 1080, 30 fps) wordt frame per frame gerenderd door dezelfde engine in een aparte pagina `video.html`, zodat beeld en app altijd overeenkomen.

1. `video.html` bouwt wereld, wind en routes, en rekent de volledige vlucht vooraf met de autopilot (dt = 0,05 s).
2. `window.renderFrame(i)` zet camera, kaartstijl, overlays en schip voor tijd i/30 en rendert. De pagina gebruikt `preserveDrawingBuffer`.
3. Een Node-script (Playwright, headless Chrome met SwiftShader) roept elke frame op en maakt een JPEG-screenshot (kwaliteit 92), inclusief de HTML-overlays.
4. Per scène gaan de JPEGs via een pipe naar ffmpeg (`-f image2pipe -c:v mjpeg` → `libx264 -crf 18 -preset veryfast -pix_fmt yuv420p`), eerst naar een tijdelijke `.mp4`, hernoemd als ffmpeg met 0 eindigt.
5. De scènes worden samengevoegd met de concat demuxer en `-c copy`.

De vluchttijd wordt op videotijd afgebeeld met een ease-in-out time-lapse (tot ± 75×), daarna 26 s vlucht in 5 s voor de aanlegmanoeuvre.

Tijd (s)	Scène	Wat er gebeurt
0–8	Hero	Schip over de zonneparken, titel "Ride the wind.", callouts voor zonnefolie, envelop, propellers en gondel
8–14	De wereld wordt een kaart	Camera trekt omhoog; `uMap` gaat van 0 naar 1 tussen 10,2 en 12,6 s, gebouwen groeien uit de grond, windsporen en gust-zones verschijnen
14–22	Routeren	Scan van 13.831 A*-nodes, drie routes tekenen zich, tijdsbubbels en het planningspaneel schuiven in
22–32	Navigeren	Heading-up camera achter het schip met banner, "Then", wegpil en bottom sheet, time-lapse
32–39	Aankomst	Terug naar 3D, aanleggen op Aster Island Pier, eindkaart met tijd, kWh en besparing

Rendertijd: ± 1,4 s per frame op 2 CPU-cores zonder GPU (± 27 min voor de hele film); met een GPU gaat het in real time.

Mijlpalen, acceptatiecriteria en referentie

Bouw in vijf stappen, elk afgesloten met een demo; de acceptatiecriteria hieronder zijn de definitie van klaar.

Acceptatiecriteria

- ☐Demoroute Copperleaf Farms naar Aster Island Pier (vertrek 14:00): Greenest ≈ 1,2 kWh netto (± 10%) en minstens 30% zuiniger dan de rechte-lijn-baseline; de drie routes zijn zichtbaar verschillend.
- ☐Zelfde seed geeft identieke routes (snapshot-test in Vitest op `route()` en `buildManeuvers()`).
- ☐Voor alle 56 hub-paren en alle vertrekuren geeft Greenest een route, of de UI meldt duidelijk "No safe route with this wind".
- ☐Vertrekuur wijzigen herberekent wind en drie routes in minder dan 300 ms (Web Worker), en de squall line verschuift zichtbaar.
- ☐Kaartstijl: wegen blijven even breed op het scherm bij elke zoom, kustlijn zonder trapjes, gebouwen groeien uit de grond, dag en nacht werken.
- ☐60 fps op een laptop-GPU op 1440p, in beide kaarttypes.
- ☐Navigatie: de banner toont de juiste volgende manoeuvre en afstand, "Then" verschijnt, de ETA telt mee af, aankomst geeft de groene banner met het verbruik.
- ☐Zelf vliegen: meer dan 190 m van de route geeft binnen ± 1 s "Rerouting…" en een nieuwe route; een gemelde gust wordt omzeild.
- ☐Onder 760 px breed wordt het planningspaneel een bottom sheet; touch-pad werkt.
- ☐De video rendert deterministisch: dezelfde frame-index geeft telkens hetzelfde beeld.

Referentie-implementatie. Het werkende prototype en de volledige TypeScript-broncode (± 3.500 regels, mappen zoals in de sectie over de projectstructuur) horen bij deze spec. Bij twijfel geldt de code; de getallen in deze spec komen er rechtstreeks uit.