Navigating the Tumultuous Seas of AI

From: **Joseph.Maxwell**
Joseph.Maxwell02@proton.me

To: **Joseph.Maxwell02@proton.me**
Joseph.Maxwell02@proton.me

On: **Wednesday, 17 June 2026 at 0:09:04**

How to stay captain of your own vessel and turn an LLM into a research assistant rather than an answer machine

Most advice about AI teaches people how to speak to the machine.

Be specific. Give examples. Define the tone. Break the task into steps. Ask it to act as an expert.

That advice is useful, but it begins too late.

The more important question is not:

How should I prompt the AI?

It is:

What role have I given the AI inside my work?

The same language model can be used as a shortcut around thinking, an answer machine, a ghostwriter, a coding assistant, a critic, a research partner, or part of a larger system.

The model may be the same.

The role changes everything.

I do not experience working with an LLM as standing outside a machine and issuing commands to it.

It feels closer to navigating a vessel from inside it.

I hold a direction in my head. Often I do not yet know the final form of the thing, but I have a sense of where it is heading. I use the model to help carry the work in that direction while paying attention to how the system responds beneath my feet.

The AI is the vessel currently carrying the work.

The ideas, evidence, code, files, experiments and unfinished structures are the cargo.

The wider environment is the sea: uncertainty, incomplete information, time, complexity and changing conditions.

I remain the captain.

My job is not simply to point at a destination.

My job is to choose routes the current vessel can survive.

The same ocean is not the same problem for every vessel

A small sailing boat and a cargo ship may cross the same ocean, but they do not experience the same hazards.

A small vessel may be threatened by individual waves, sudden weather, exposure and limited supplies.

A cargo ship may be threatened by inertia, turning radius, shallow water, structural loading and the difficulty of changing direction once committed.

The obstacle is not simply the ocean.

It is the relationship between:

- the environment;
- the vessel;
- the load;
- the destination;
- the available recovery options.

The same is true of AI.

A short chat is light and manoeuvrable. It can change direction quickly, but it carries little reliable state and loses continuity easily.

A long-running research workflow can carry much more:

- documents;
- code;
- datasets;
- evidence;
- terminology;
- version history;
- decisions;
- unresolved questions.

But it also develops inertia.

A bad assumption introduced early can become embedded in later work. A private vocabulary can become difficult to translate. A model can become so fluent in the project’s language that it stops noticing the assumptions underneath it.

As the vessel grows, the method of navigation has to change.

You cannot steer a cargo ship as though it were a canoe.

I hold the direction, not a finished answer

When I begin working with an LLM, I often do not hold a complete solution.

What I hold is a directional model.

I may sense that:

- a distinction matters;
- two parts of a system are related;
- a repeated failure is exposing a missing layer;
- the current explanation is too shallow;
- several different problems share a structural form;
- the work is moving towards something more precise.

I then use the model to help travel towards that structure.

This is different from asking the model to decide the destination.

The model can suggest routes, expose contradictions, generate alternatives, write code, organise evidence and help build the vessel while it is already moving.

But I remain responsible for deciding whether the work still points towards the thing I intended to make.

That is where the human contribution sits.

The originality is not simply in the final words.

It lies in:

- the direction held;
- the distinctions selected;
- the standards imposed;
- the connections made;
- the failures recognised;
- the claims accepted or rejected.

The model can help carry and shape the work.

It does not inherit authorship merely because it helped produce the surface.

Reading the vessel

A captain does not steer only by looking at the horizon.

They also read the vessel.

They feel changes through the deck. They notice resistance in the water. They watch

the load shift and pay attention to the delay between turning the wheel and changing direction.

I use an LLM in a similar way.

I pay attention to:

- which concepts it repeatedly groups together;
- which distinctions it loses;
- where it becomes too confident;
- where explanation turns into analogy;
- where an implementation begins to sound more validated than it is;
- what it omits;
- when its language becomes smoother than the evidence;
- where it starts repeating my terminology without testing it.

The answer is not only an answer.

It is also feedback from the vessel.

That does not mean every unusual phrase is meaningful.

One strange word may be noise.

A repeated wording pattern is weak evidence.

A repeated structural failure is more useful.

A failure that appears across controlled variations is something worth designing around.

The aim is not to imagine that the model has hidden feelings or intentions.

It is to inspect the behaviour of the system I am relying on.

One voyage

One of the clearest examples came from building a materials-engineering model-audit system.

The initial idea was straightforward: instead of only asking whether a model predicts well, test whether the model's important features remain stable when the data or descriptors are perturbed.

The LLM helped me turn that idea into:

- an experimental plan;
- software architecture;
- perturbation logic;
- result formats;
- validation reports;
- technical documentation.

The work moved quickly.

Too quickly in places.

At one stage, the system architecture looked mature enough that the language around it began sounding as though the science had already been validated.

It had not.

Some parts were only specified.

Some were scaffolded.

Some had run on fixtures.

Some had run on real native data.

The model was not intentionally misleading me. It was doing what language models are very good at doing: completing the shape of the story faster than the evidence had completed the work.

That failure changed the project.

I began separating states explicitly:

- architecture defined;
- scaffold implemented;
- contract integrated;
- runtime executed;
- validated on native data;
- validated on official external data;
- publication ready;
- external-use ready.

Those labels became waypoints.

They stopped forward motion from being mistaken for arrival.

The same process later shaped how I handled memory, authority, dataset size, cross-domain transfer and failure states.

A weakness in the interaction became a design requirement.

That is one of the most valuable things AI has given me.

It does not only accelerate answers.

It can reveal the hidden structure of the work, including the structure of its own failure.

The logbook matters

A long conversation can feel like a stable workspace, but it is not a reliable database.

Earlier decisions can be compressed. Boundaries can drift. The latest wording can quietly reshape the older work.

So I began moving important state outside the chat.

I use:

- specifications;
- versioned documents;
- software manifests;
- object registries;
- dataset records;
- evidence matrices;
- patch reports;
- file indexes;
- test outputs;
- decision logs.

These are the ship’s logbook and cargo manifest.

They record:

- what exists;
- where it came from;
- what changed;
- what has actually been tested;
- what remains unresolved;
- what the current work is allowed to claim.

The model helps me operate against that external state.

It is not expected to remember the entire voyage perfectly.

This has been especially important because my health and available energy fluctuate heavily.

There are days when I can produce a huge amount and days when simply re-entering the project is difficult.

Externalising the work means I do not have to rebuild the whole world in my head every time I return.

The vessel retains the cargo.

The logbook tells me where I was.

The model helps me get back on deck.

What this has allowed me to achieve

The obvious benefit is speed.

I can move from an idea to:

- a technical specification;
- a software package;
- an experimental design;
- an evidence report;
- a public explanation;

far faster than I could alone.

But speed is not the main benefit.

The deeper benefit is continuity and structural visibility.

Using this approach, I have been able to build and connect:

- a materials-engineering audit engine;
- a governed scientific runtime;
- regime and phase-mapping tools;
- a bounded navigation engine;
- a persistent memory architecture;
- evidence and claim-status systems;
- technical papers and public essays;
- long-running research programmes across several domains.

Not all of that work is equally mature.

Some parts are conceptual.

Some are implemented.

Some have run on real data.

Some remain experimental.

That distinction is part of the achievement, not a weakness in it.

The goal is not to make every part sound finished.

It is to know what each part actually is.

When AI becomes a cheat

AI becomes a cheat when it hides the true state of the user's understanding.

That can happen even when no formal rule has technically been broken.

The model produces a polished answer.

The user can submit, publish or present it.

But they no longer know:

- which assumptions were inspected;
- which parts they could independently explain;
- where the evidence came from;
- what remains uncertain;
- what the model invented;
- what would falsify the claim.

The work appears complete while the thinking underneath it is not.

A research assistant plays a different role.

A good research assistant helps:

- organise evidence;
- identify missing information;
- compare explanations;
- challenge assumptions;
- design experiments;
- document decisions;
- expose limitations;
- prepare work for review.

They do not silently become the researcher.

The same should apply to an LLM.

Capability is not authority

One of the most important rules I developed is simple:

Capability is not authority.

A model may be capable of writing code.

That does not mean the code should be run.

A system may be capable of processing millions of records.

That does not mean consuming the whole dataset is justified.

An explanation may be coherent.

That does not mean it is supported.

I separate:

- what the model can do;
- what it is allowed to do;
- what the evidence supports;
- what may be promoted into the wider project.

That distinction improves the work because the model is no longer forced to pretend to be the final authority.

The goal is not to avoid dependence

I do depend on AI for parts of this work.

Pretending otherwise would be dishonest.

It carries structure, continuity, translation and implementation load that would otherwise be difficult or impossible for me to sustain alone.

The goal is not to eliminate dependence.

It is to prevent invisible dependence.

A healthy dependence is:

- visible;
- documented;
- inspectable;
- reversible where possible;
- understood by the human using it.

An unhealthy dependence is one where the user no longer knows what the system is carrying on their behalf.

Sometimes the right move is to hold position

Language models are biased towards completion.

They want to answer.

They want the paragraph to resolve.

They want the system to feel whole.

But serious work often requires valid non-completion.

Some of the most useful outputs in my work are:

- hold;
- blocked;
- not estimable;
- partially supported;
- data unavailable;

- not tested.

These are not failed answers.

They are decisions not to sail into conditions the current vessel cannot safely handle.

A trustworthy research assistant must be able to stop.

A model that must always arrive will invent land.

Six rules for navigating with AI

1. Hold the direction yourself

Begin with:

- the real question;
- the intended direction;
- the current stage;
- what would count as genuine progress;
- what would count as pretending.

Do not hand the destination to the model.

2. Define the model's role

Decide whether it is being asked to:

- generate possibilities;
- organise evidence;
- critique a claim;
- write code;
- compare alternatives;
- translate language;
- propose a decision.

Do not let one response silently perform every role.

3. Separate proposal, implementation and evidence

Keep distinct:

- what is suggested;
- what is built;
- what has run;
- what the results support.

Do not use "working" as a single status.

4. Keep the logbook

Preserve:

- original files;
- sources;
- versions;
- configurations;
- test results;
- generated artefacts;
- decision records;
- claim status.

The final summary should never be the only surviving record.

5. Use bounded voyages and explicit waypoints

Start with the smallest test capable of answering the current question.

Then grow.

Mark the difference between:

- proposed;
- built;
- run;
- tested;
- externally validated;
- ready to publish.

6. Let the system stop

Allow:

- unknown;
- hold;
- insufficient evidence;
- not estimable;
- source unavailable.

A system that cannot stop cannot be trusted with difficult navigation.

The deeper lesson

The best use of AI is not finding the perfect prompt.

It is creating a relationship in which the model can carry more of the work without taking ownership of its meaning.

The AI is not the destination.

It is not the ocean.

It is not the captain.

It is the vessel currently carrying the work.

The human task is to hold the direction, read the feedback honestly, understand the limits of the vessel, preserve the logbook and choose routes that match both the sea and the cargo.

Used as an answer machine, an LLM can help people avoid thinking while producing the appearance of thought.

Used as a research assistant, it can help people think across more material, preserve more structure and build more ambitious work without surrendering responsibility for what that work means.

The problem is not the prompt.

It is the role you give the AI.