Factor Graphs and "World Models"

Frank Dellaert

April 21, 2026

Author: Frank Dellaert

There is a lot of excitement right now around **world models**: learned latent-state representations that support prediction, imagination, and planning. Recent work around *JEPA* is especially interesting in this regard, as it aims to learn useful latent dynamics that can support downstream decision making and planning, as in the recent *LeWorldModel* paper, and more broadly in Yann LeCun's 2022 position paper *A Path Towards Autonomous Machine Intelligence*. This connects to Lecun's earlier work on energy-based models, where the central idea is not to generate outputs one token at a time, but to define an energy landscape and then minimize it.

The jury is still out, but I'm a fan . What's more, from a factor-graph point of view, this feels very familiar.

1. Factor graphs are energy-based models

A factor graph defines a probability distribution up to normalization:

$$p(x \mid z) \propto \prod_i \phi_i(x_i),$$

where each factor ϕ_i expresses a local probabilistic relationship among some subset of variables x_i, conditioned on the measurements z. Taking the negative log gives an energy:

$$E(x) = -\log p(x \mid z) = \sum_i E_i(x_i) + \text{constant}.$$

Hence, **maximum a posteriori inference is energy minimization**.

That observation has been the basis of state estimation, smoothing, and SLAM for a long time. In robotics, we already know how to represent dynamics, measurements, priors, and constraints as factors. We already know how to optimize the resulting energy efficiently. In that sense, factor graphs are a very concrete

and highly structured form of energy-based world model, exactly the perspective developed in Dellaert and Kaess's Factor Graphs for Robot Perception.

2. From estimation to planning

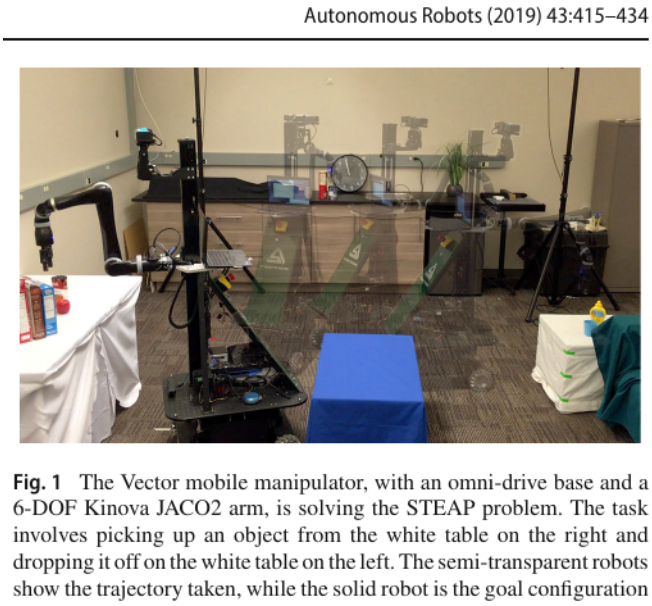


Fig. 1 The Vector mobile manipulator, with an omni-drive base and a 6-DOF Kinova JACO2 arm, is solving the STEAP problem. The task involves picking up an object from the white table on the right and dropping it off on the white table on the left. The semi-transparent robots show the trajectory taken, while the solid robot is the goal configuration

Figure 1: Figure 1 from the STEAP paper showing the Vector mobile manipulator solving the STEAP problem

In our earlier *STEAP* work, simultaneous trajectory estimation and **planning**, we already extended the same factor-graph machinery to planning. There, estimation and planning were treated in a unified optimization framework over an entire trajectory, as described by Mukadam et al. in STEAP: Simultaneous Trajectory Estimation and Planning. The same graph could encode what the robot had observed, how it could move, and what trajectory it should follow to get from point a to point b.

That was an important step, but it still assumes a **fixed interval** $[a, b]$: there is a start, there is a goal, and we optimize over the trajectory connecting the two. An interesting question is what happens if we anchor the whole construction at **the current time** t, rather than thinking in terms of one fixed offline interval.

3. Anchor the graph at "now"

Suppose we center everything around the present time t. Then we can think of the robot's world as split into two closely related factor graphs:

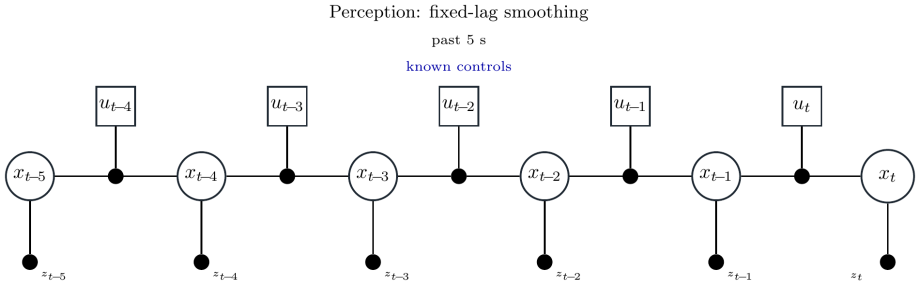
- a graph over the **recent past**, used for perception and state estimation

- a graph over the **near future**, used for planning and control

Both graphs share the same state representation and the same dynamics model. The difference is what information they are tied to: the past is tied to **measurements**. The future is tied to **objectives**.

This is exactly the kind of structure one would expect from a practical world model: infer the hidden state from recent evidence, then optimize the likely future under the same transition model.

4. Perception as fixed-lag smoothing



A fixed-lag smoothing graph for perception. Each dynamics factor connects neighboring states together with a known control input, while unary measurement factors summarize the sensor evidence over the last five seconds.

For perception, consider a factor graph over the last five seconds:

$$x_{t-5}, x_{t-4}, \dots, x_t.$$

On this graph we place:

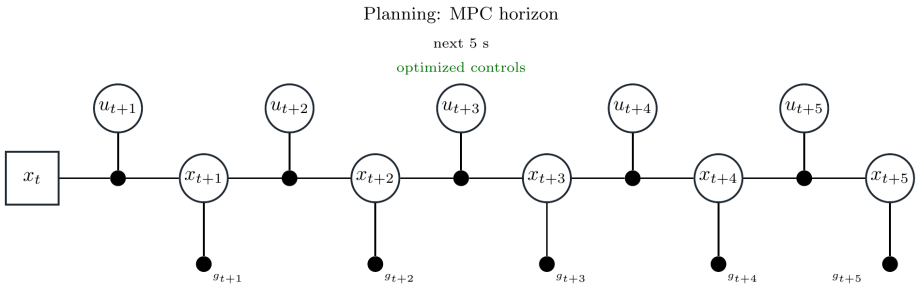
- **dynamics factors** between successive states, each coupled to a known control input u_k
- **unary measurement factors** that connect each state to the relevant sensor observations

Optimizing this graph gives a **fixed-lag smoother**. Rather than filtering only forward, we re-estimate the recent trajectory using all measurements within the lag window. That gives a better estimate of the current state x_t, while still keeping the computation bounded and online.

In this inference problem, the controls over the recent past are part of the observed history: commanded wheel velocities, torques, or thrusts are already known, and so they enter the graph as known variables attached to the dynamics factors.

This is the standard factor-graph view of perception: the robot's recent past is inferred by minimizing an energy defined by dynamics plus measurements.

5. Planning as a future factor graph



A model-predictive control (MPC) graph for planning. The same* dynamics factors connect the future states, but now the controls are optimization variables, and the unary terms are goal or objective factors rather than measurements.*

Now look at the next five seconds:

$$x_t, x_{t+1}, \dots, x_{t+5}.$$

Again, we use the **same dynamics factors**. That is important: the robot should "think" about the future using the same transition structure with which it interprets the past.

But in the planning graph, the controls are no longer known. They are optimized jointly with the future states over the horizon. This is the familiar **collocated optimal control** view: states and controls live on the same time grid and are solved for together.

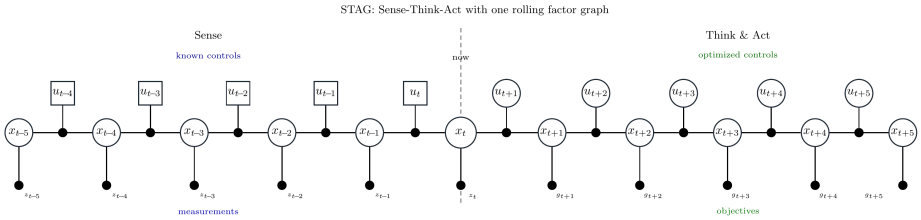
But now the unary factors are different. Instead of measurement factors, we introduce **goal or objective factors**. Let us denote these as $g_{t+1}, \dots, g_{t+5}$ over the horizon, and these could represent:

- goal-reaching costs
- collision-avoidance costs
- control-effort penalties
- stability or balance objectives
- task-specific preferences

Optimizing this future graph gives a trajectory in the familiar **model-predictive control** style: solve over a short horizon, execute the first action, move forward in time, then rebuild and re-optimize.

The structure is almost the mirror image of perception. In the past, unary factors tell us what the sensors observed. In the future, unary factors tell us what we want.

6. STAG: Sense-Think-Act with Graphs



The combined STAG picture: a single rolling factor graph centered on the current state. The left half explains the recent past from measurements and known controls, while the right half plans the near future with goal factors and optimized controls.

Put these two pieces together and a simple picture emerges:

- **Sense**: estimate the recent past and present with a fixed-lag smoothing graph
- **Think**: optimize over the latent state trajectory using factor-graph inference
- **Act**: execute the first part of the optimized future trajectory

This is really **Sense-Think-Act with factor Graphs**. Let us call it **STAG**.

The appealing part is that STAG does not require a sharp separation between "perception code" and "planning code". Both are instances of the same underlying idea:

- define a state
- write down dynamics
- add factors
- minimize energy

Past and future are handled by nearly the same computational machinery, just with different unary terms. One graph explains evidence. The other graph expresses intent.

The picture is clear: a single rolling graph spans both the recent past and near future, anchored at the current state x_t. On the left are measurement factors. On the right are goal factors $g_{t+1}, \ldots, g_{t+5}$. In the middle sits the current latent state, tying perception and planning together.

7. What remains robot-specific?

Once stated this way, the core questions for any robot become very clear. For a given platform, what do we need to define or *learn*?

1. **What is the state?** Is it pose and velocity? Joint angles and contact models? A full belief state over robot and environment? And can we learn this representation?

2. **What is the dynamics model?** Is it analytic, learned, or hybrid? Is it first-order, second-order, or manifold-valued?
3. **What is the measurement model?** How do cameras, IMUs, force sensors, wheel encoders, or tactile signals connect to the latent state?
4. **What are the objectives for the future?** What should the robot optimize for over the next few seconds? Is this where reinforcement learning enters the picture?

I don't have the answers to all these questions, but we've started to play around with this model, and invite others to do the same.

Summary

The current interest in world models, JEPA, and energy-based learning is exciting. But from the perspective of factor graphs, there is also a deep continuity with ideas that robotics and estimation have used for years.

Factor graphs are already energy-based models. Their negative log likelihood is already an energy. State estimation and SLAM have long exploited this fact. And STEAP and other papers showed that planning can live in the same optimization framework.

The next step is to anchor everything at the present and view the robot as carrying two tightly coupled graphs:

- a **past graph** for perception
- a **future graph** for planning

Using the same state and the same dynamics on both sides gives a simple and powerful recipe for online robot intelligence: Sense with factors. Think by optimization. Act on the resulting plan.

That is STAG.

References

1. "Energy-Based Models", Yann LeCun, Sumit Chopra, Raia Hadsell, Marc'Aurelio Ranzato, Fu-Jie Huang, in *Predicting Structured Data*, MIT Press, 2006.
2. *A Path Towards Autonomous Machine Intelligence*, Yann LeCun, position paper, version 0.9.2, June 27, 2022.
3. "LeWorldModel: Stable End-to-End Joint-Embedding Predictive Architecture from Pixels", Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, Randall Balestriero, arXiv, March 13, 2026.
4. "Factor Graphs for Robot Perception", Frank Dellaert, Michael Kaess, *Foundations and Trends in Robotics*, 2017, Vol. 6, No. 1-2, pp. 1-139.
5. "STEAP: Simultaneous Trajectory Estimation and Planning", Mustafa Mukadam, Jing Dong, Frank Dellaert, Byron Boots, *Autonomous Robots*, 2019, Vol. 43, No. 2, pp. 415-434.

Disclosure: AI was used to help draft this post.