Balancing Actuation Energy and Computing Energy in Motion Planning

Soumya Sudhakar, Vivienne Sze, Sertac Karaman
Low Energy Autonomy and Navigation (LEAN) Group

CICS Talk - May 5, 2021

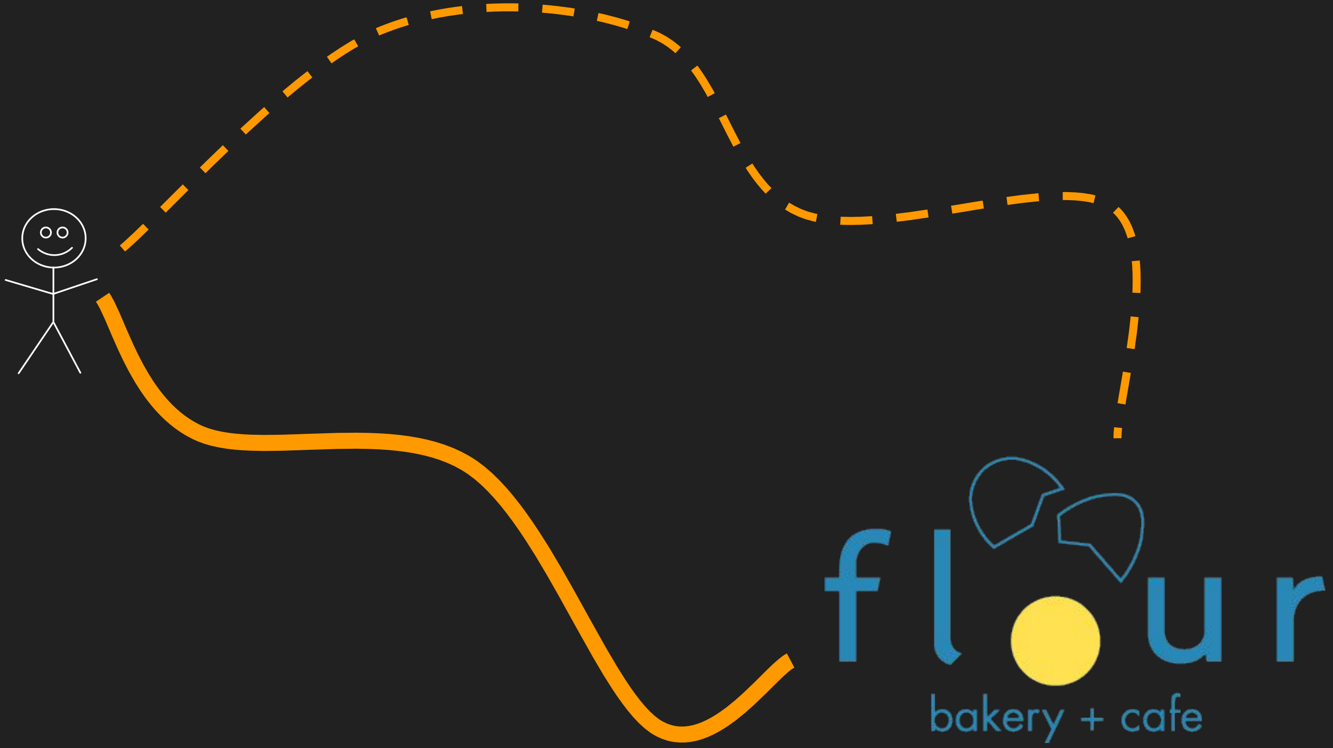
Planning a Path to the Coffeeshop

flour
bakery + cafe

Planning a Path to the Coffeeshop

flour
bakery + cafe

Don’t Think Too Hard



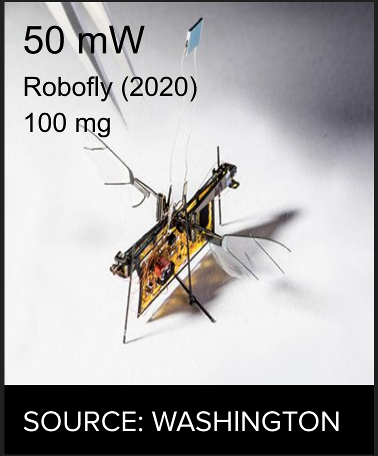
Don’t Think Too Hard

Don’t Think Too Hard

How do we get an energy-constrained robot to decide when it has computed enough?

flour
bakery + cafe

Miniature or Long-Duration Robotics are Power-Constrained



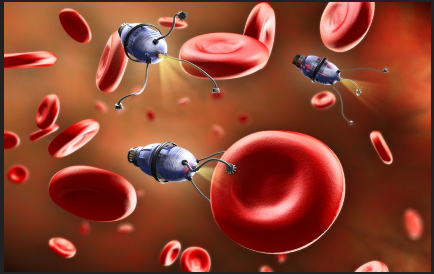
Power-constrained
due to size

Power-constrained
due to duration

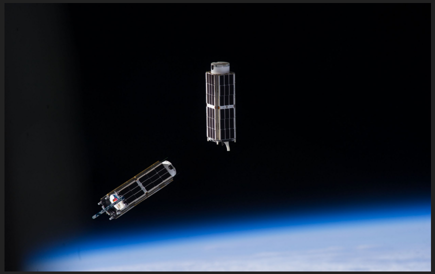
Miniature or Long-Duration Robotics Enable New Solutions

Infrastructure inspections

Persistent environmental monitoring



Noninvasive targeted drug delivery



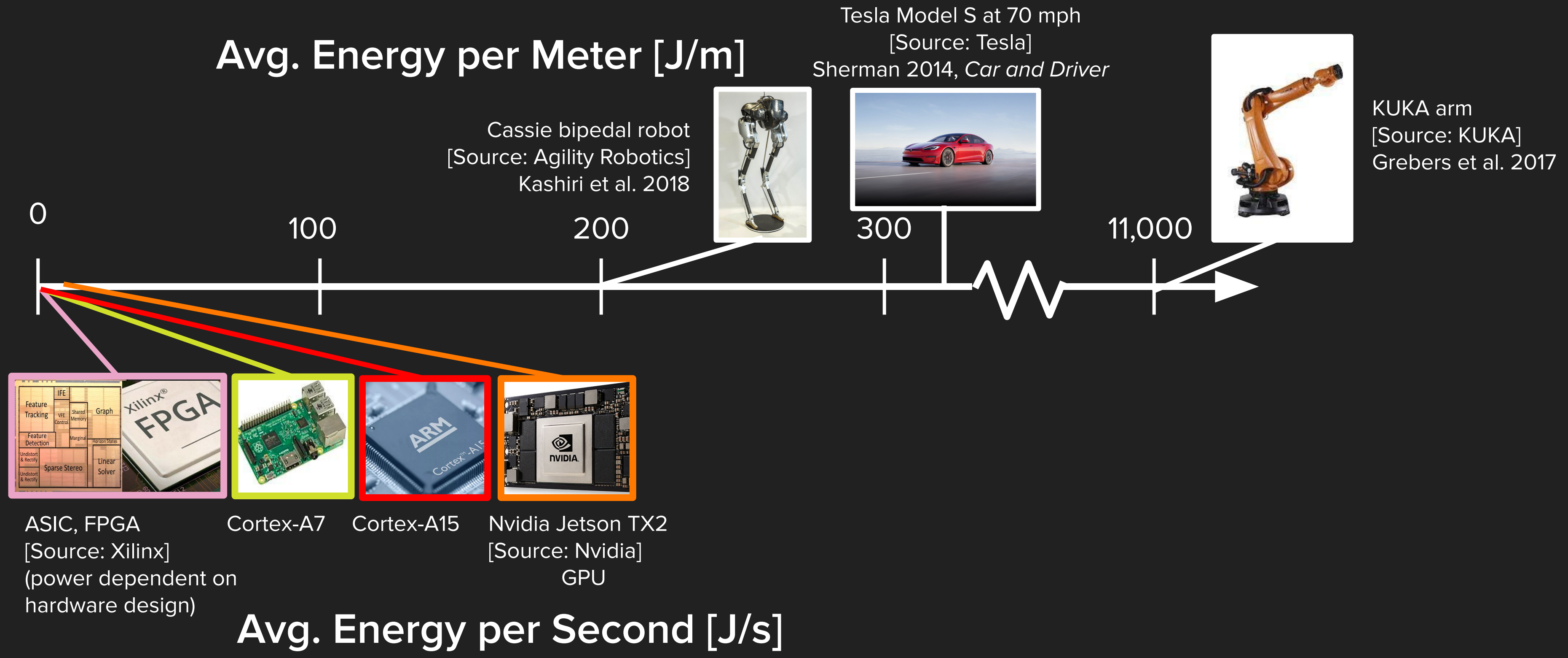
Space exploration

Many applications exist for miniature or long-duration robotic platforms that can intelligently navigate

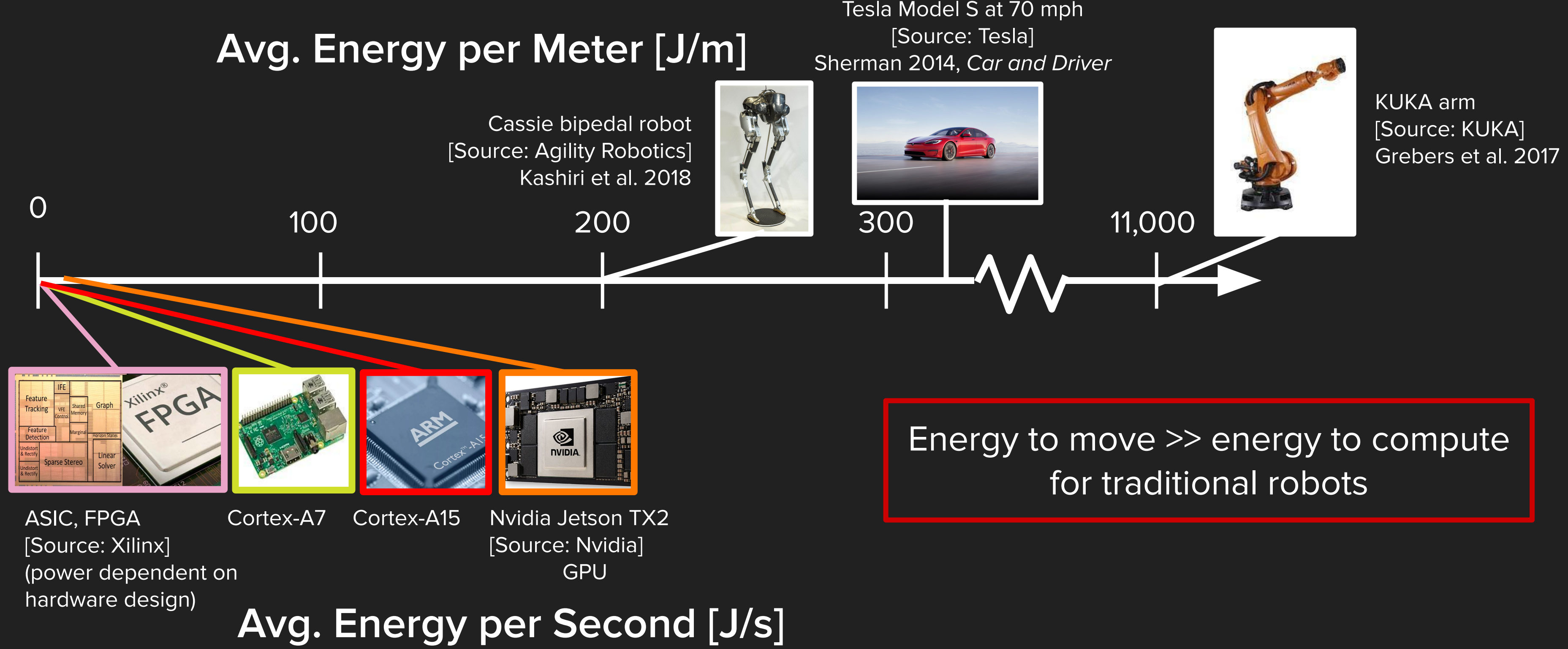
Recent Advances in Low-Power Robotic Platforms

Success in actuating miniature and long-duration robotics at low power in the lab and real-world

Less Attention Paid to Energy Computing Consumes



Less Attention Paid to Energy Computing Consumes



Energy to move >> energy to compute
for traditional robots

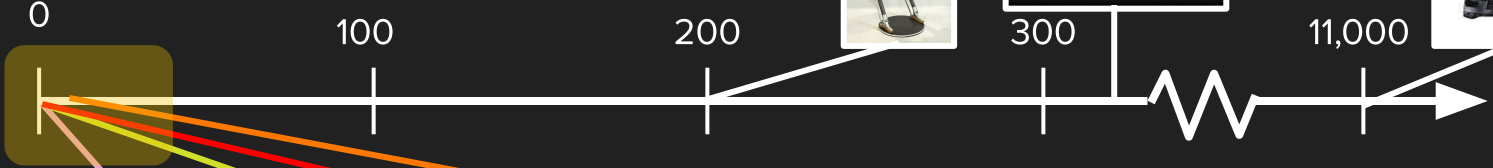
Less Attention Paid to Energy Computing Consumes

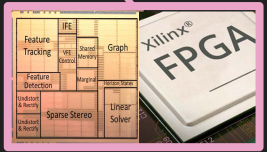
Avg. Energy per Meter [J/m]

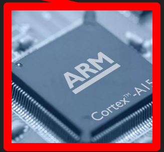
Tesla Model S at 70 mph
[Source: Tesla]
Sherman 2014, *Car and Driver*

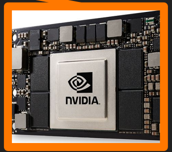
Cassie bipedal robot
[Source: Agility Robotics]
Kashiri et al. 2018

KUKA arm
[Source: KUKA]
Grebers et al. 2017









ASIC, FPGA
[Source: Xilinx]
(power dependent on hardware design)

Cortex-A7

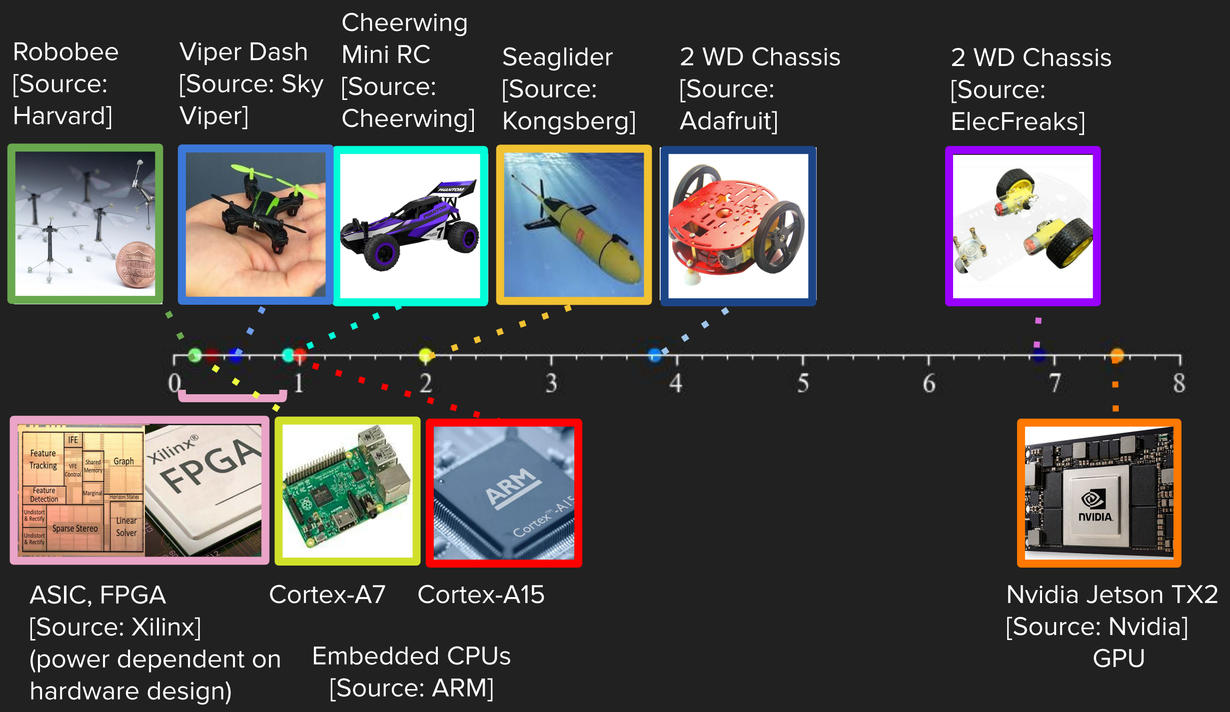
Cortex-A15

Nvidia Jetson TX2
[Source: Nvidia]
GPU

Avg. Energy per Second [J/s]

Energy to move >> energy to compute for traditional robots

Avg. Energy per Meter [J/m]



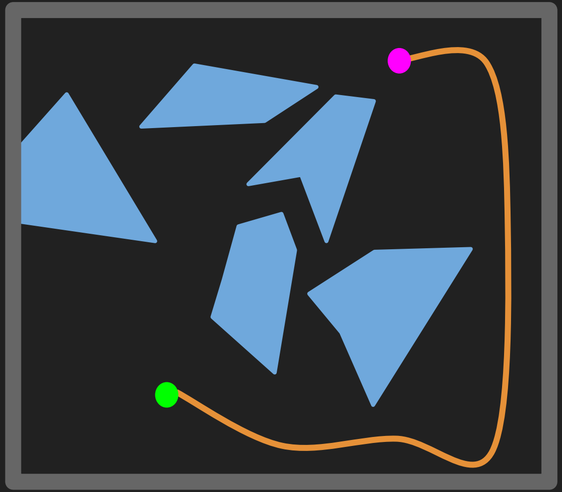
Avg. Energy per Second [J/s]

For low-power robotics, energy to move and energy to compute are on a similar magnitude

Background: Motion Planning

Plan the shortest path from the start to the goal avoiding all obstacles

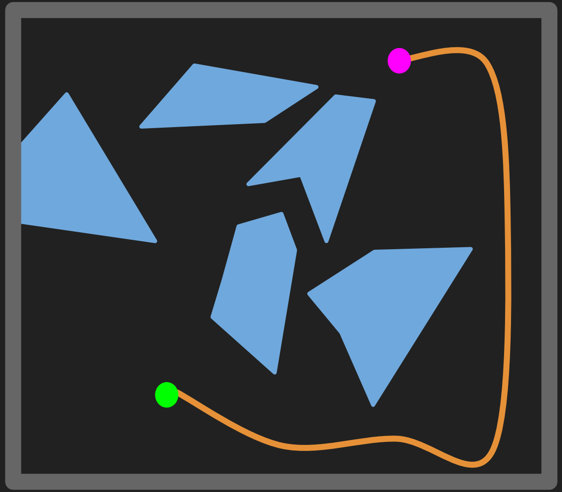
The motion planning problem



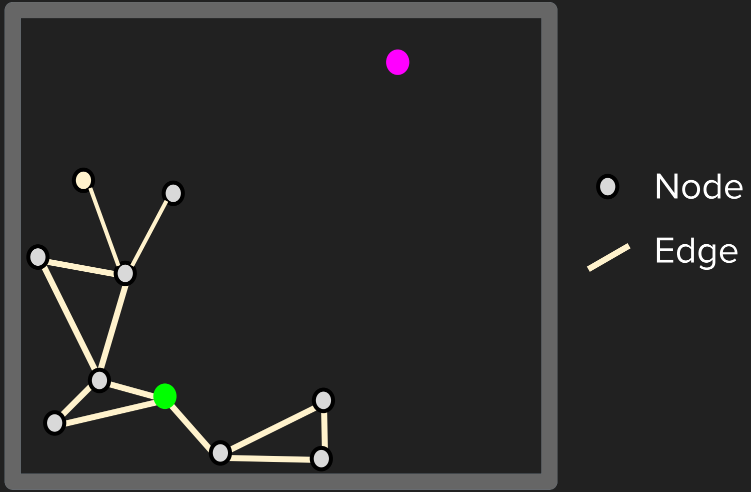
Background: Motion Planning

Plan the shortest path from the start to the goal avoiding all obstacles

The motion planning problem



Sampling-based motion planner

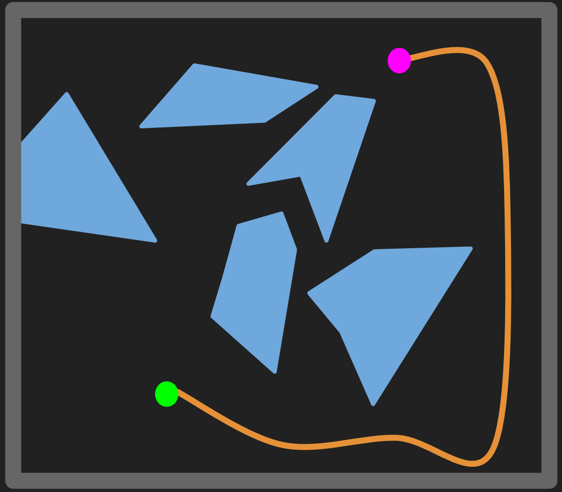


Sampling-based motion planner find paths by sampling and connecting nodes in free space

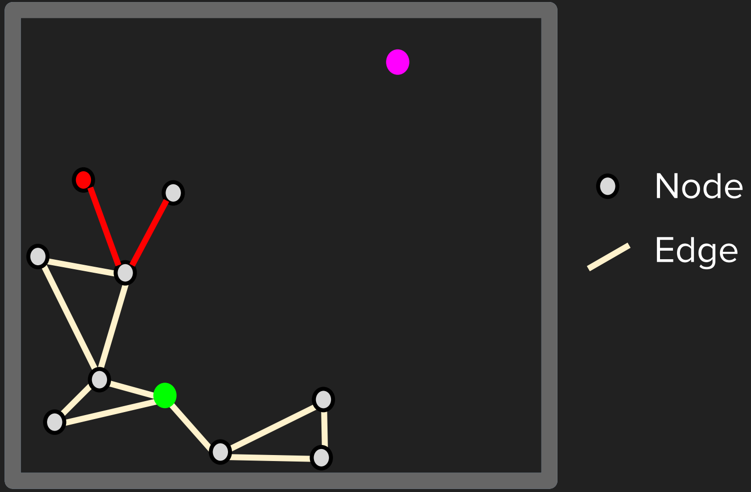
Background: Motion Planning

Plan the shortest path from the start to the goal avoiding all obstacles

The motion planning problem



Sampling-based motion planner

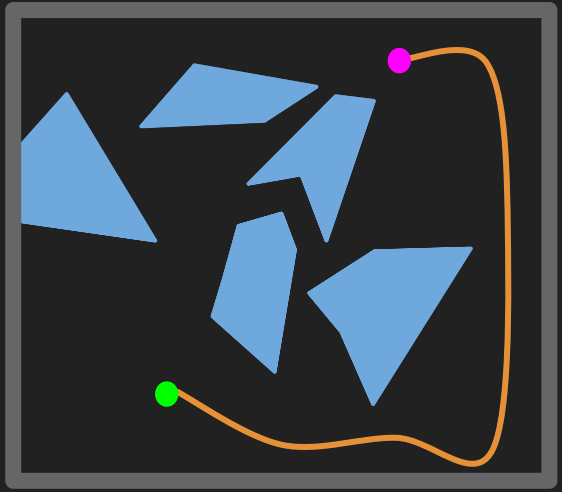


Sampling-based motion planner find paths by sampling and connecting nodes in free space

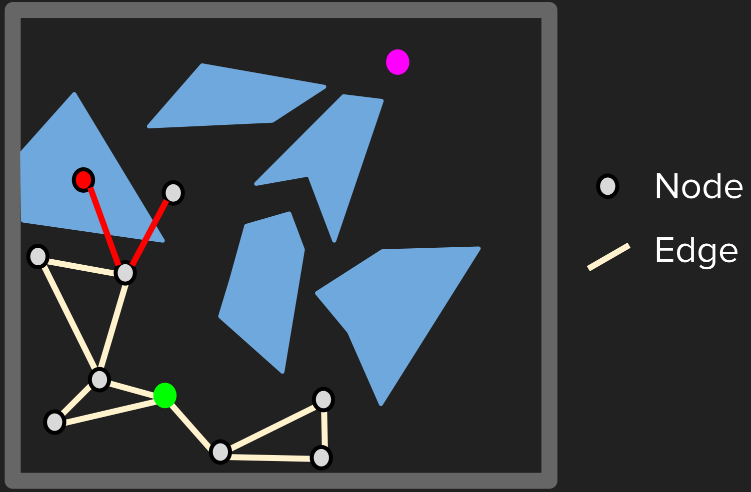
Background: Motion Planning

Plan the shortest path from the start to the goal avoiding all obstacles

The motion planning problem



Sampling-based motion planner

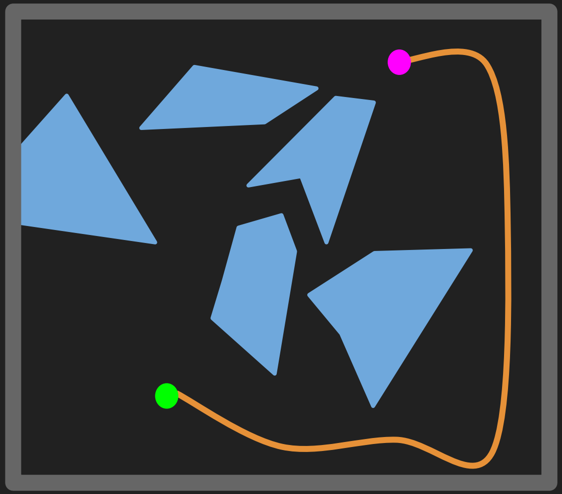


Sampling-based motion planner find paths by sampling and connecting nodes in free space

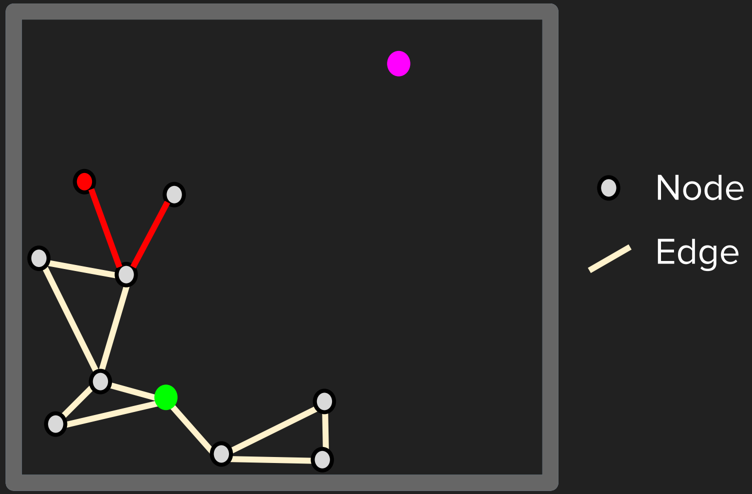
Background: Motion Planning

Plan the shortest path from the start to the goal avoiding all obstacles

The motion planning problem



Sampling-based motion planner

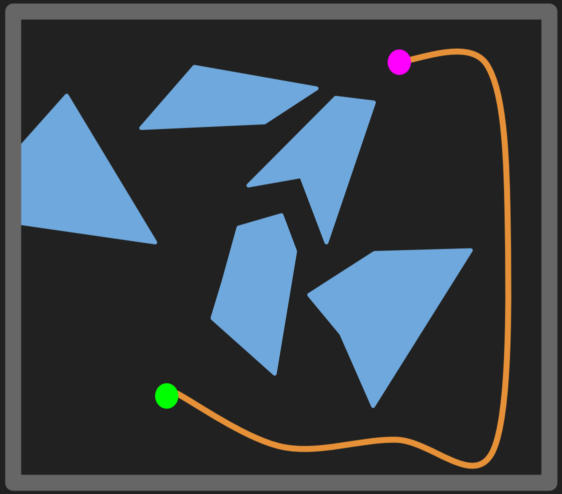


Sampling-based motion planner find paths by sampling and connecting nodes in free space

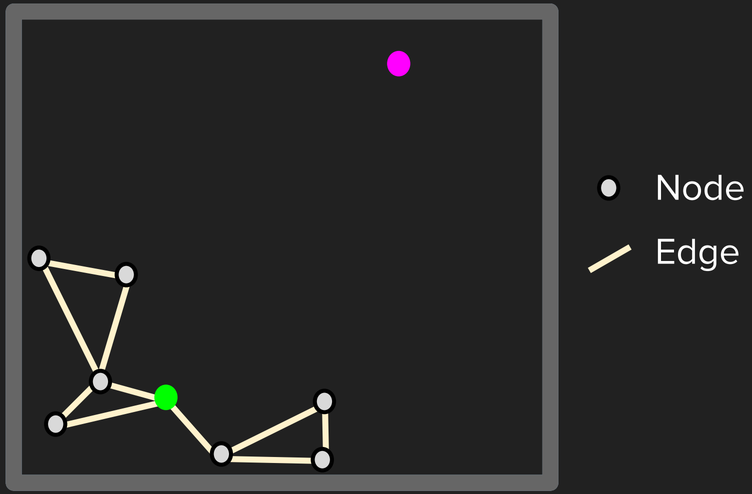
Background: Motion Planning

Plan the shortest path from the start to the goal avoiding all obstacles

The motion planning problem



Sampling-based motion planner

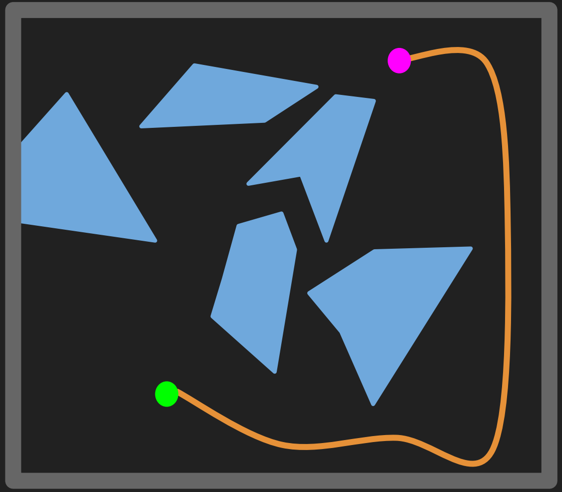


Sampling-based motion planner find paths by sampling and connecting nodes in free space

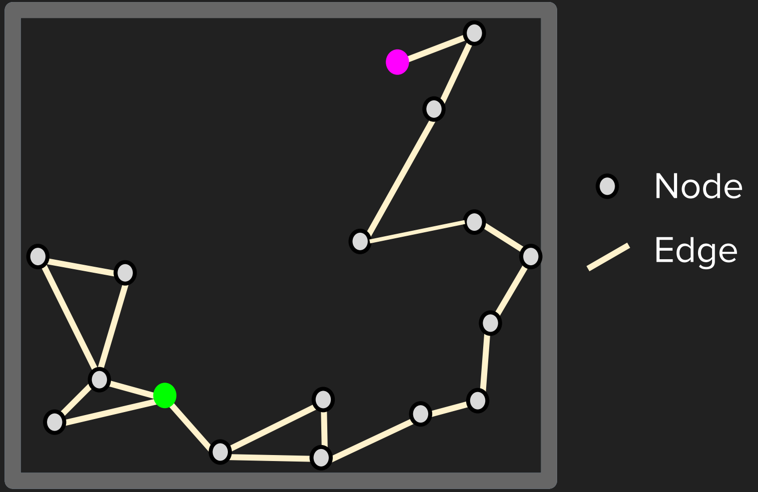
Background: Motion Planning

Plan the shortest path from the start to the goal avoiding all obstacles

The motion planning problem



Sampling-based motion planner

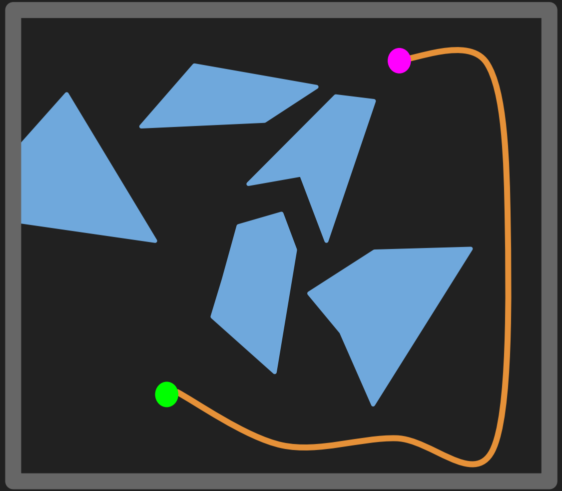


Sampling-based motion planner find paths by sampling and connecting nodes in free space

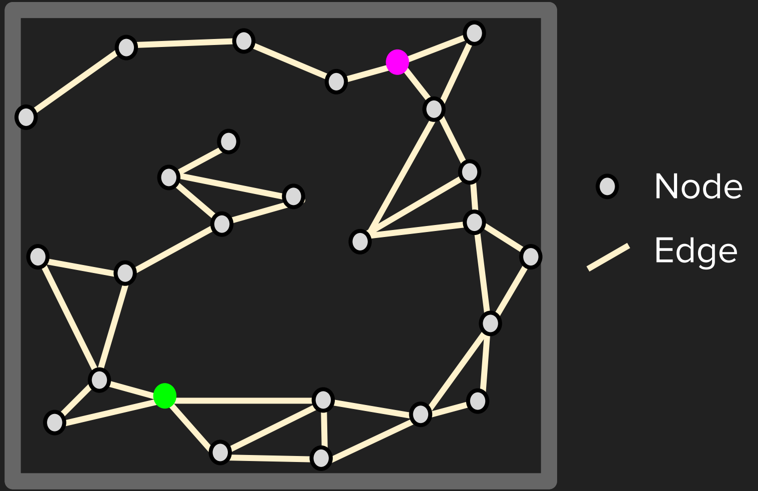
Background: Motion Planning

Plan the shortest path from the start to the goal avoiding all obstacles

The motion planning problem



Sampling-based motion planner

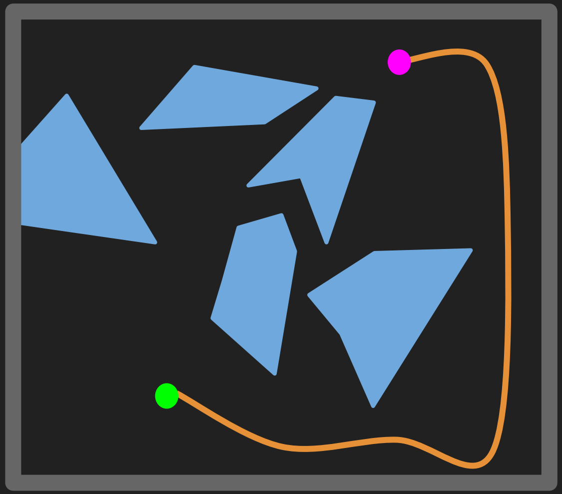


Sampling-based motion planner find paths by sampling and connecting nodes in free space

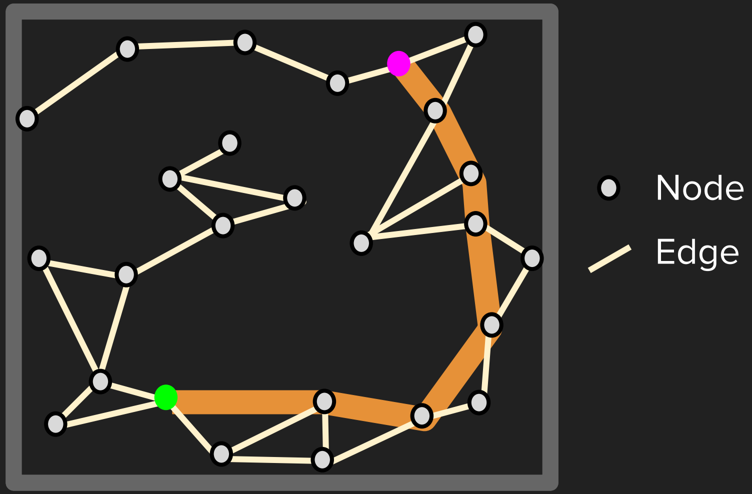
Background: Motion Planning

Plan the shortest path from the start to the goal avoiding all obstacles

The motion planning problem



Sampling-based motion planner



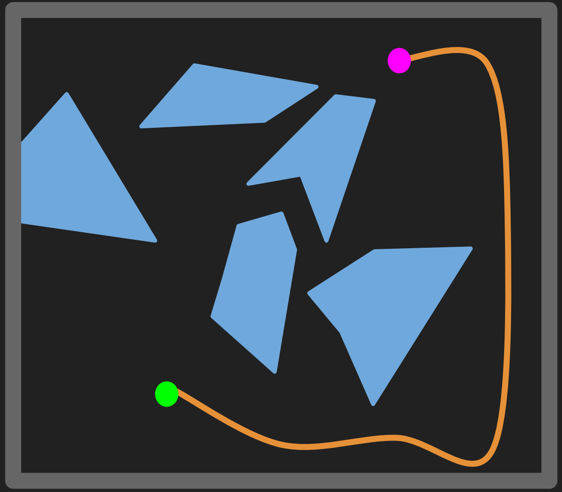
Sampling-based motion planner find paths by sampling and connecting nodes in free space

Background: Motion Planning

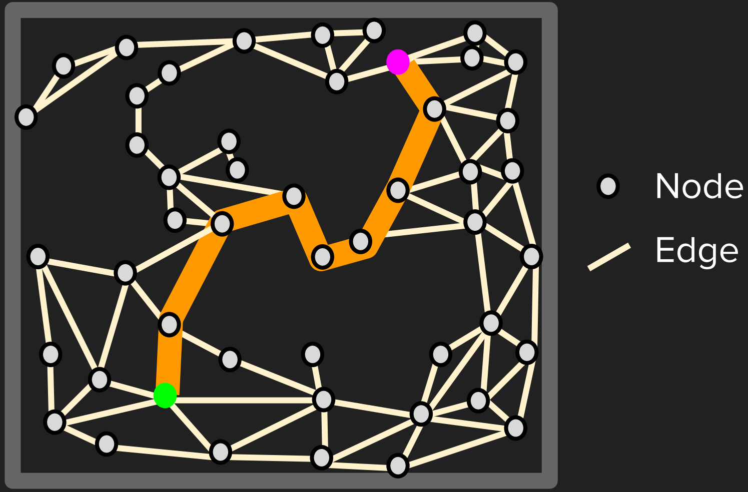
Plan the shortest path from the start to the goal avoiding all obstacles

Computing more nodes → find shorter paths

The motion planning problem



Sampling-based motion planner

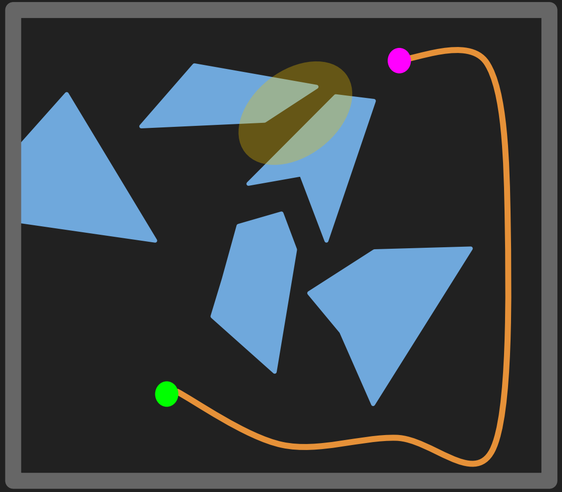


Sampling-based motion planner find paths by sampling and connecting nodes in free space

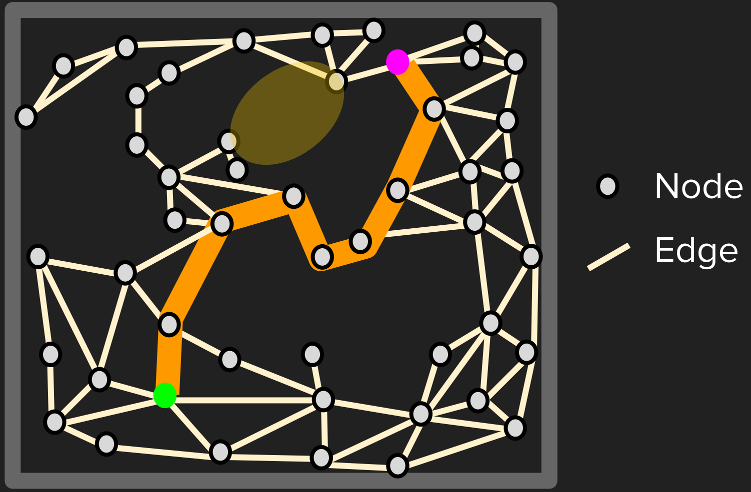
Background: Motion Planning

Plan the shortest path from the start to the goal avoiding all obstacles

The motion planning problem



Sampling-based motion planner

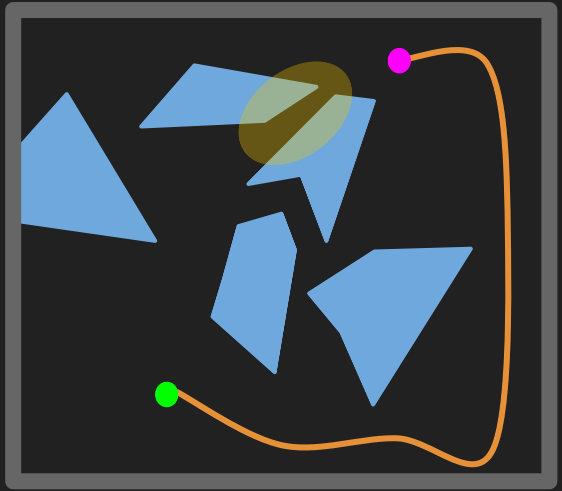


Sampling-based motion planner find paths by sampling and connecting nodes in free space

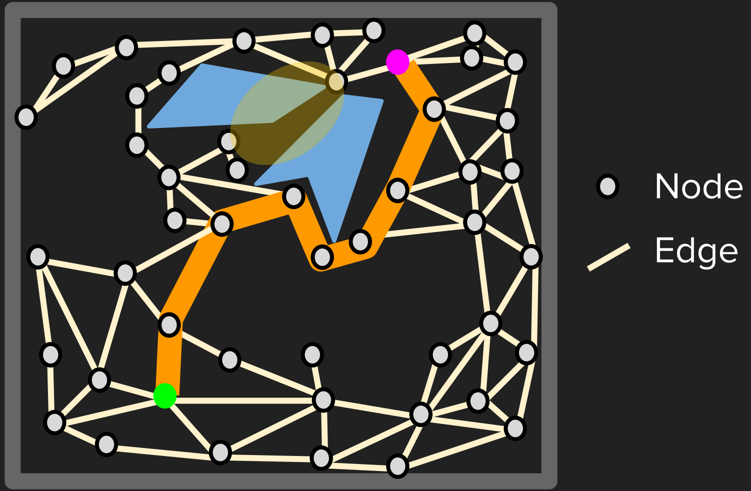
Background: Motion Planning

Plan the shortest path from the start to the goal avoiding all obstacles

The motion planning problem



Sampling-based motion planner

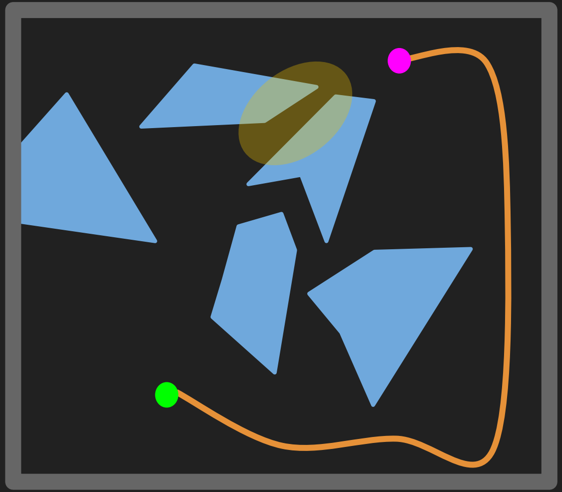


Sampling-based motion planner find paths by sampling and connecting nodes in free space

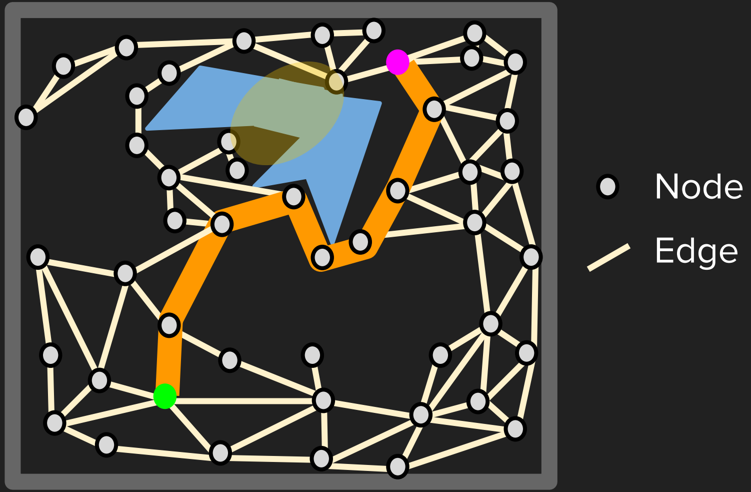
Background: Motion Planning

Plan the shortest path from the start to the goal avoiding all obstacles

The motion planning problem



Sampling-based motion planner

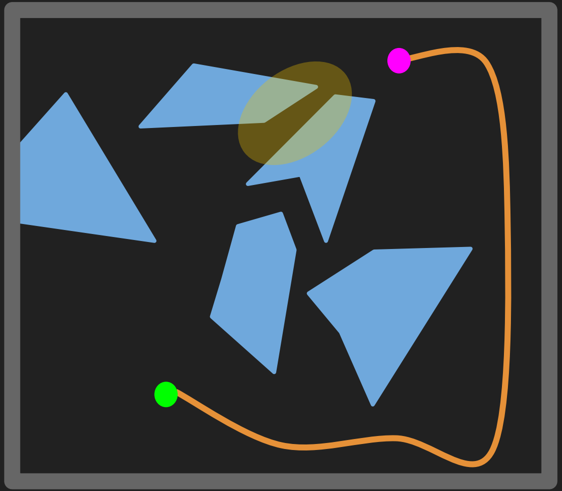


Sampling-based motion planner find paths by sampling and connecting nodes in free space

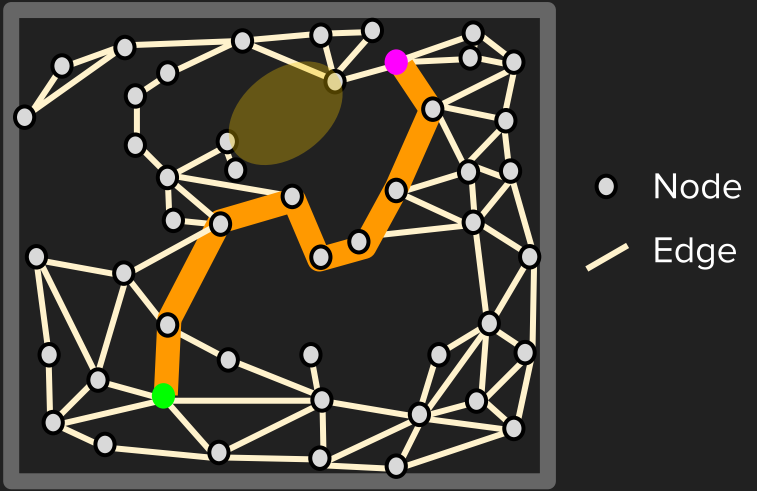
Background: Motion Planning

Plan the shortest path from the start to the goal avoiding all obstacles

The motion planning problem



Sampling-based motion planner

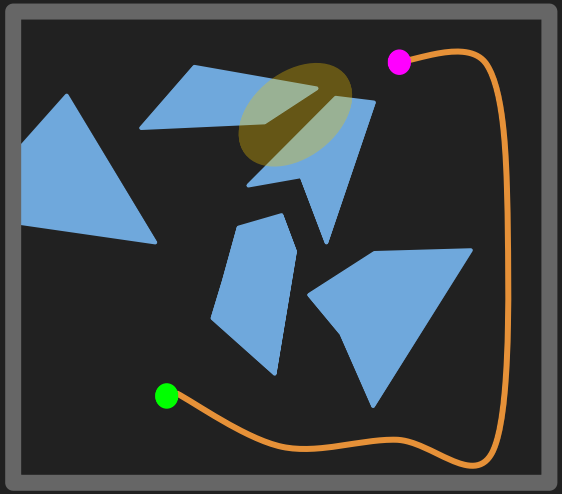


Sampling-based motion planner find paths by sampling and connecting nodes in free space

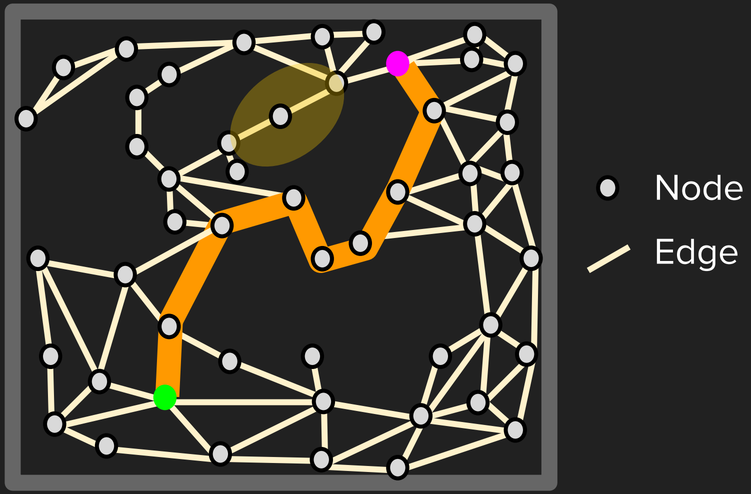
Background: Motion Planning

Plan the shortest path from the start to the goal avoiding all obstacles

The motion planning problem



Sampling-based motion planner



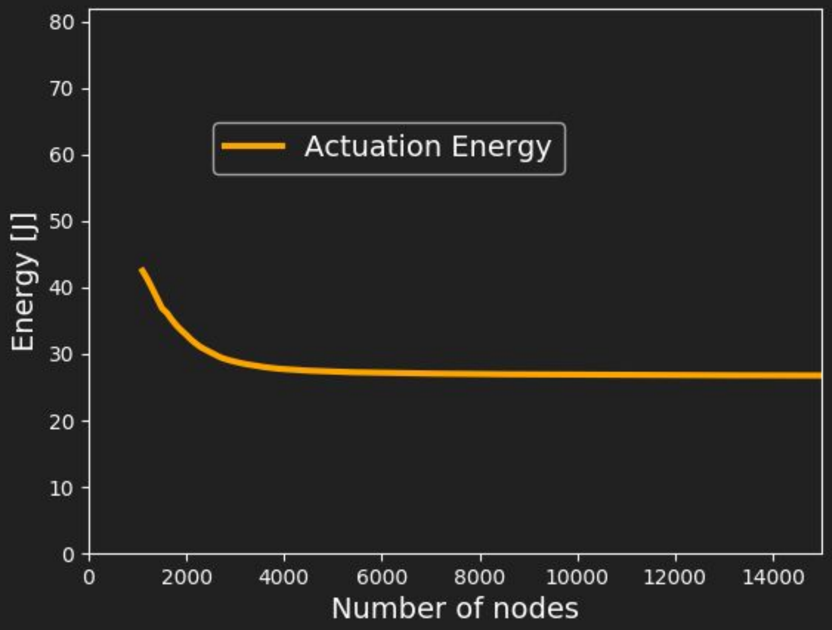
Sampling-based motion planner find paths by sampling and connecting nodes in free space

Background: Motion Planning

E_a Actuation energy = energy consumed by vehicle's actuators (e.g., motors) to move along path

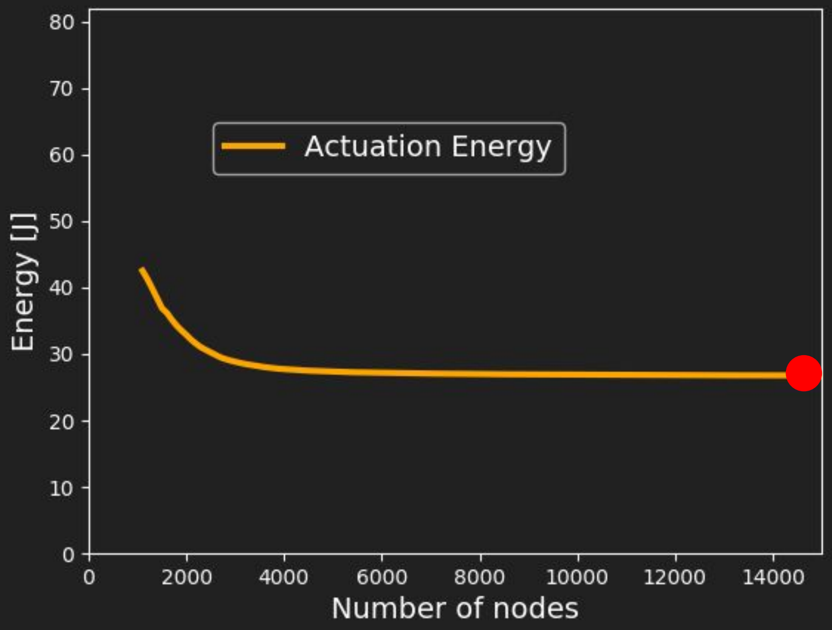
E_c Computing energy = energy consumed by computer onboard vehicle to compute the path

Total Energy of Actuation and Computing



Simulated vehicle that can travel 1 m/s at 1 Watt, computing on a Cortex A15 (embedded CPU)

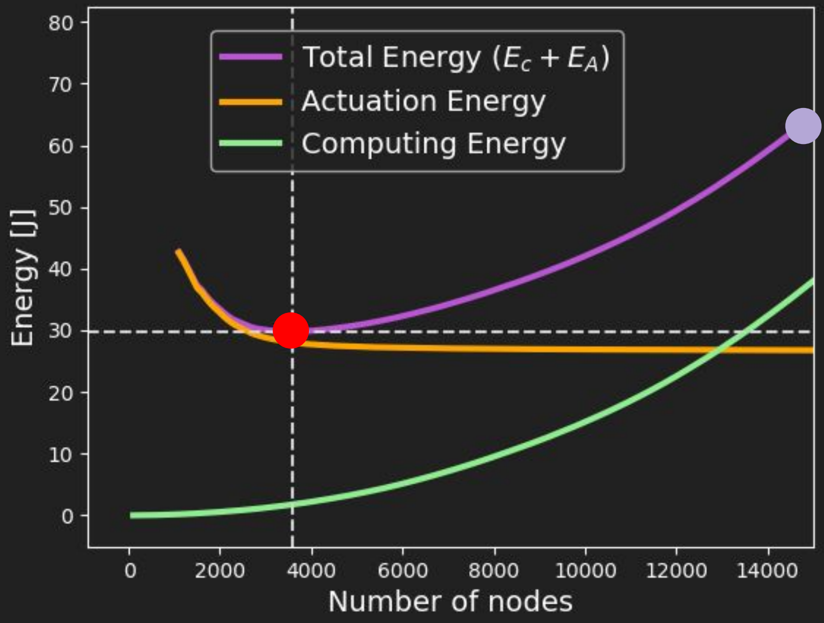
Total Energy of Actuation and Computing



Simulated vehicle that can travel 1 m/s at 1 Watt, computing on a Cortex A15 (embedded CPU)

$$\overbrace{E_t}^{\text{Total energy}} = \underbrace{E_a}_{\text{Actuation energy (energy to move along path)}}$$

Total Energy of Planning and Moving



Simulated vehicle that can travel 1 m/s at 1 Watt, computing on a Cortex A15 (embedded CPU)

$$\overbrace{E_t}^{\text{Total energy}} = \underbrace{E_a}_{\substack{\text{Actuation energy} \\ \text{(energy to move} \\ \text{along path)}}} + \underbrace{E_c}_{\substack{\text{Computing energy} \\ \text{(energy to} \\ \text{compute path)}}}$$

Reducing total energy is now an early stopping problem

The Work of Actuation and Computation

Actuation	Computing
-	Num. of nodes [nodes], n
Path length [m], $l_a(n)$	Num. of operations [ops], $l_c(n)$

The work of actuation and the work of computing have analogous variables

The Work of Actuation and Computation

Actuation	Computing
-	Num. of nodes [nodes], n
Path length [m], $l_a(n)$	Num. of operations [ops], $l_c(n)$
Vehicle speed [m/s], v_a	Processing speed [ops/s], v_c

The work of actuation and the work of computing have analogous variables

The Work of Actuation and Computation

Actuation	Computing
-	Num. of nodes [nodes], n
Path length [m], $l_a(n)$	Num. of operations [ops], $l_c(n)$
Vehicle speed [m/s], v_a	Processing speed [ops/s], v_c
Actuation power [W], $P_a(v_a)$	Computing power [W], $P_c(v_c)$

The work of actuation and the work of computing have analogous variables

The Work of Actuation and Computation

Actuation	Computing
-	Num. of nodes [nodes], n
Path length [m], $l_a(n)$	Num. of operations [ops], $l_c(n)$
Vehicle speed [m/s], v_a	Processing speed [ops/s], v_c
Actuation power [W], $P_a(v_a)$	Computing power [W], $P_c(v_c)$
Actuation energy [J], E_a	Computing Energy [J], E_c

The work of actuation and the work of computing have analogous variables

The Work of Actuation and Computation

Actuation	Computing
-	Num. of nodes [nodes], n
Path length [m], $l_a(n)$	Num. of operations [ops], $l_c(n)$
Vehicle speed [m/s], v_a	Processing speed [ops/s], v_c
Actuation power [W], $P_a(v_a)$	Computing power [W], $P_c(v_c)$
Actuation energy [J], E_a	Computing Energy [J], E_c

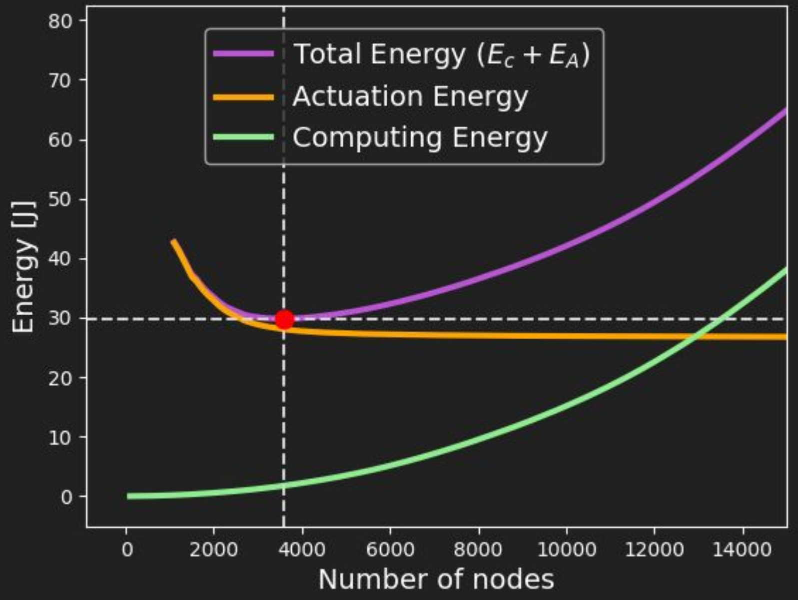
The work of actuation and the work of computing have analogous variables

The Work of Actuation and Computation

Actuation	Computing
-	Num. of nodes [nodes], n
Path length [m], $l_a(n)$	Num. of operations [ops], $l_c(n)$
Vehicle speed [m/s], v_a	Processing speed [ops/s], v_c
Actuation power [W], P_a	Computing power [W], P_c
Actuation energy [J], E_a	Computing Energy [J], E_c

The work of actuation and the work of computing have analogous variables

Total Energy of Planning and Moving



Simulated vehicle that can travel 1 m/s at 1 Watt, computing on a Cortex A15 (embedded CPU)

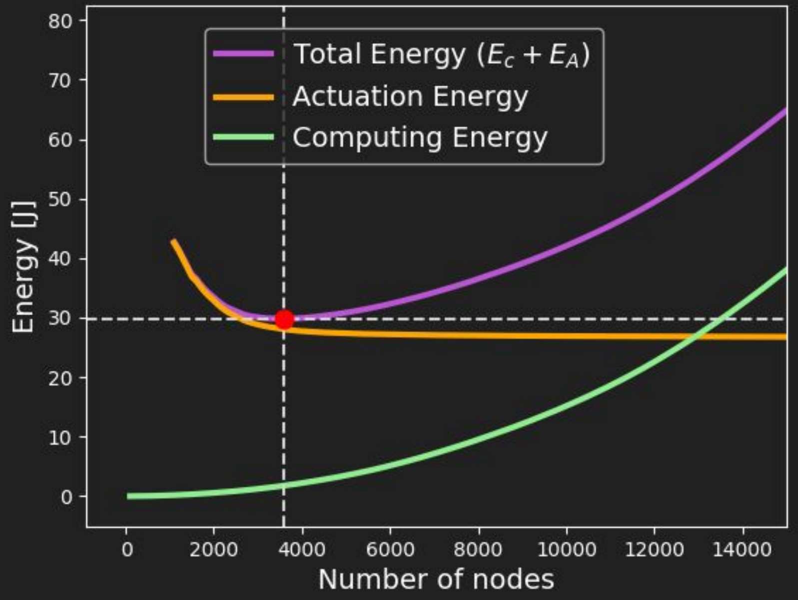
Actuation	Computing
-	Num. of nodes [nodes], n
Path length [m], $l_a(n)$	Num. of operations [ops], $l_c(n)$
Vehicle speed [m/s], v_a	Processing speed [ops/s], v_c
Actuation power [W], $P_a(v_a)$	Computing power [W], $P_c(v_c)$
Actuation energy [J], E_a	Computing Energy [J], E_c

$$E_t = E_a + E_c$$

$$= \underbrace{\frac{P_a}{v_a} l_a(n)}_{\text{Actuation energy (energy to move along path)}} + \underbrace{\frac{P_c}{v_c} l_c(n)}_{\text{Computing energy (energy to compute path)}}$$

- ⬆ # of nodes n
- ⬇ actuation energy
- ⬆ computing energy

Total Energy of Planning and Moving



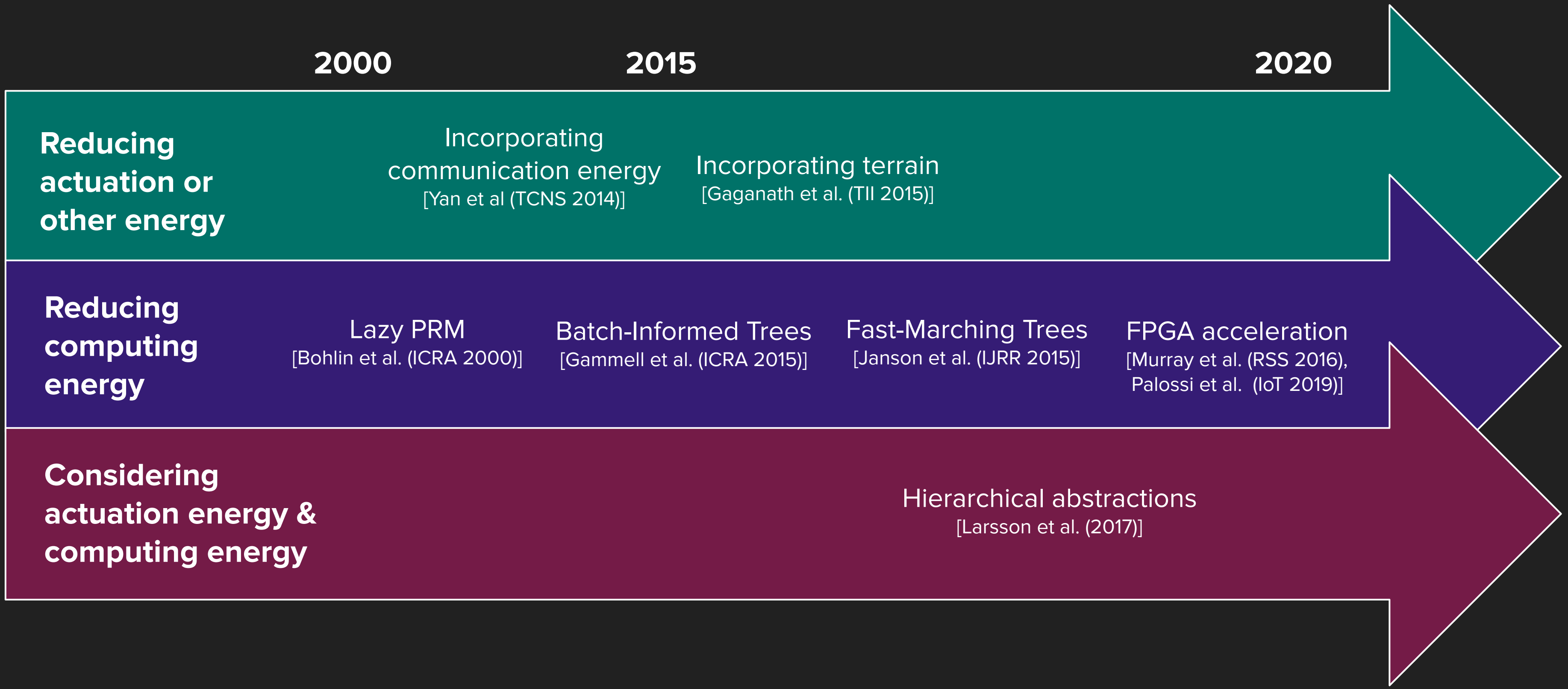
Simulated vehicle that can travel 1 m/s at 1 Watt, computing on a Cortex A15 (embedded CPU)

Actuation	Computing
-	Num. of nodes [nodes], n
Path length [m], $l_a(n)$	Num. of operations [ops], $l_c(n)$
Vehicle speed [m/s], v_a	Processing speed [ops/s], v_c
Actuation power [W], $P_a(v_a)$	Computing power [W], $P_c(v_c)$
Actuation energy [J], E_a	Computing Energy [J], E_c

$$E_t = E_a + E_c$$
$$= \underbrace{\frac{P_a}{v_a} l_a(n)}_{\text{Actuation energy (energy to move along path)}} + \underbrace{\frac{P_c}{v_c} l_c(n)}_{\text{Computing energy (energy to compute path)}} + \underbrace{\frac{P_c}{v_c} \overline{l_c}(n)}_{\text{Overhead energy (energy to decide when to stop)}}$$

Performance metric includes overhead we introduce

Related Work



Related Work

2000 | 2015 | 2020

Reducing actuation or other energy

- Incorporating communication energy [Yan et al (TCNS 2014)]
- Incorporating terrain [Gaganath et al. (TII 2015)]

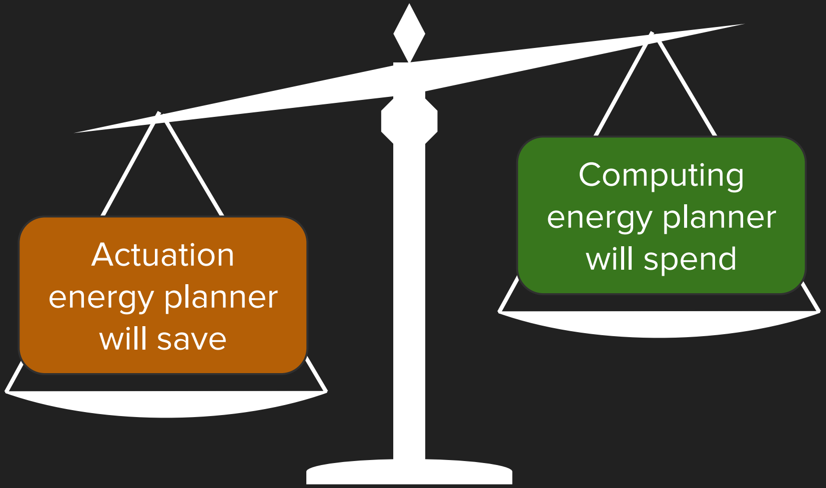
Reducing computing energy

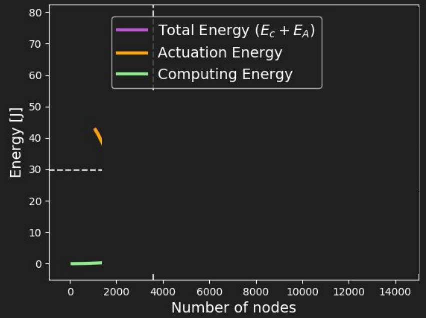
- Lazy PRM [Bohlin et al. (ICRA 2000)]
- Batch-Informed Trees [Gammell et al. (ICRA 2015)]
- Fast-Marching Trees [Janson et al. (IJRR 2015)]
- FPGA acceleration [Murray et al. (RSS 2016), Palossi et al. (IoT 2019)]

Considering actuation energy & computing energy

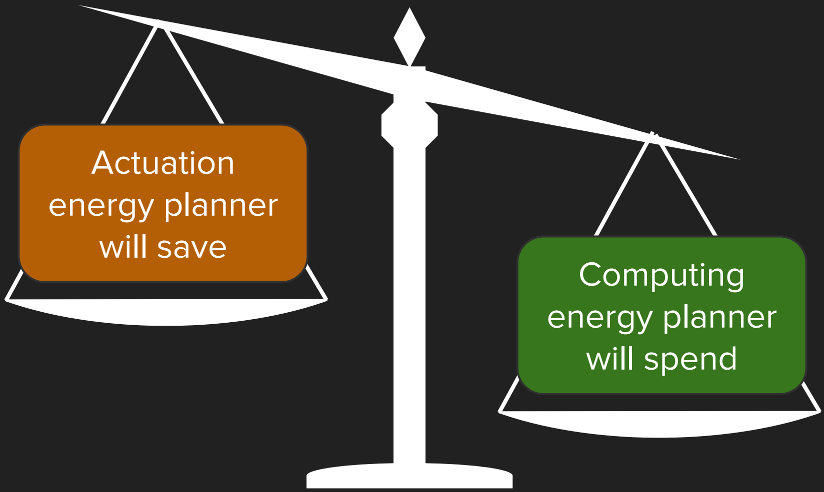
- Hierarchical abstractions [Larsson et al. (2017)]
- **CEIMP (this work)** [Sudhakar et al. (2020)]

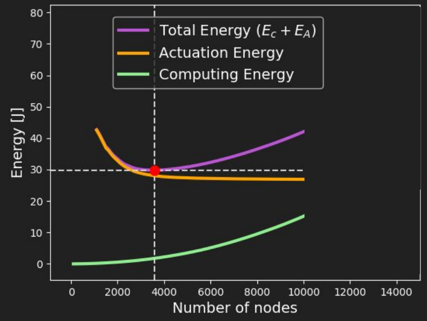
Technical Gap: Knowing How Much Computing is Enough



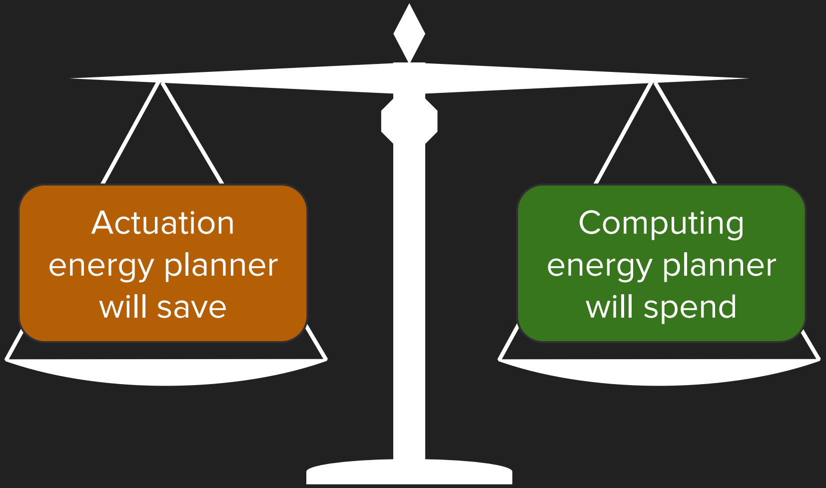


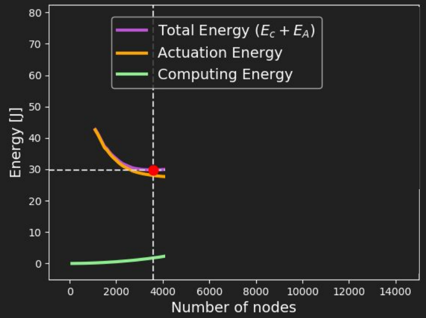
Technical Gap: Knowing How Much Computing is Enough



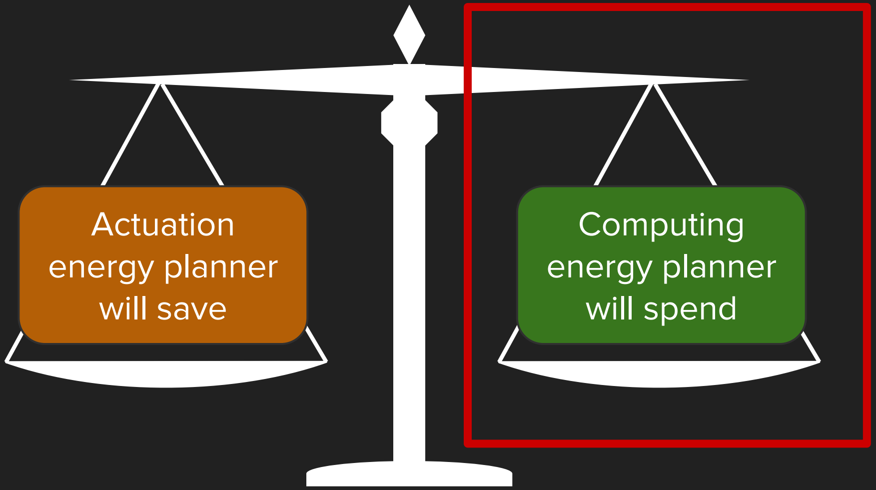


Technical Gap: Knowing How Much Computing is Enough



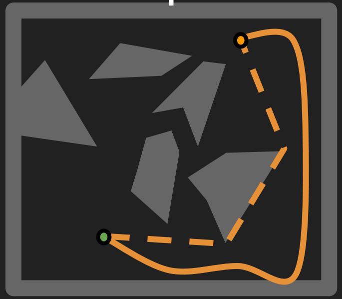


Technical Gap: Knowing How Much Computing is Enough

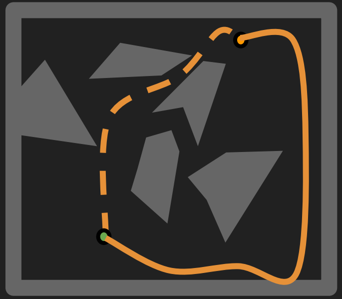


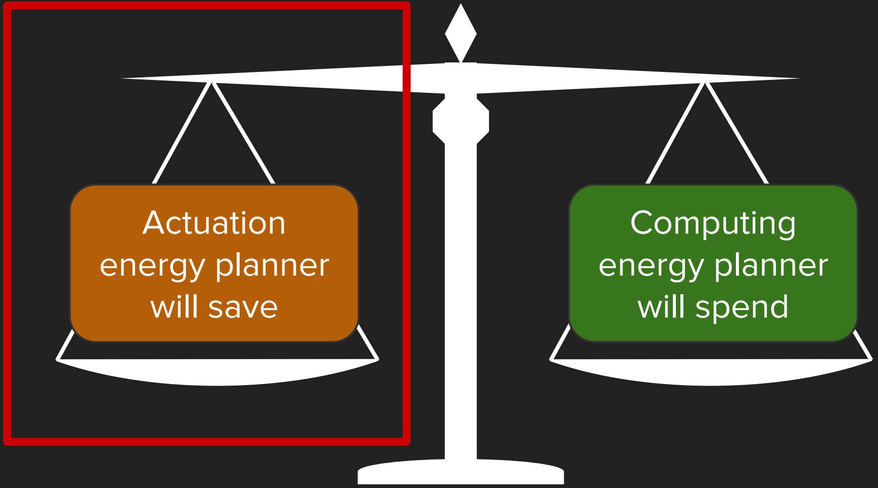
Technical Gap: Knowing How Much Computing is Enough

Improvement in **same homotopic class**



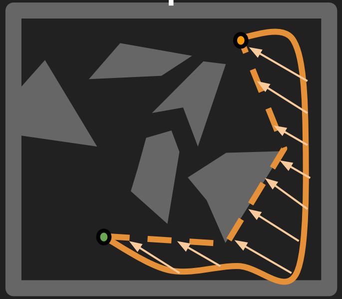
Improvement from **homotopic class change**



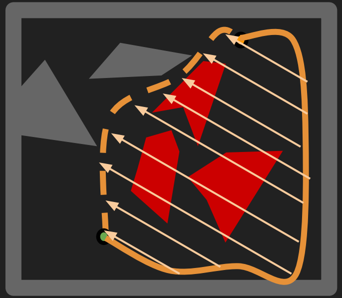


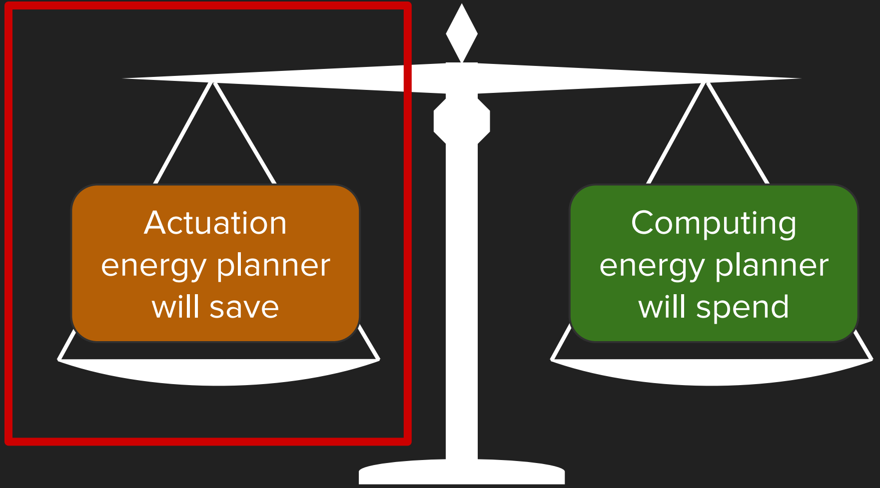
Technical Gap: Knowing How Much Computing is Enough

Improvement in **same homotopic class**



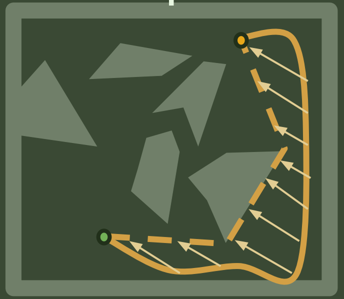
Improvement from **homotopic class change**



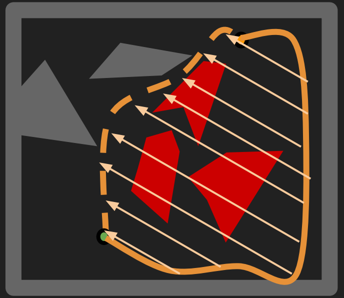


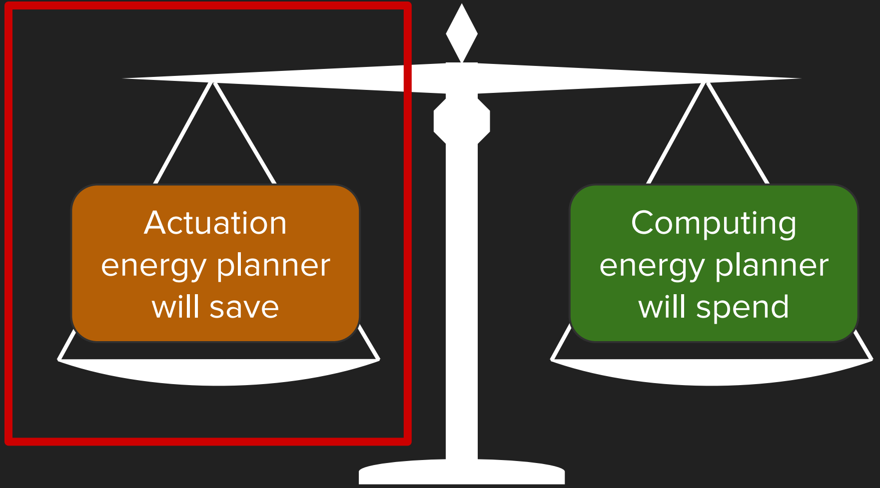
Technical Gap: Knowing How Much Computing is Enough

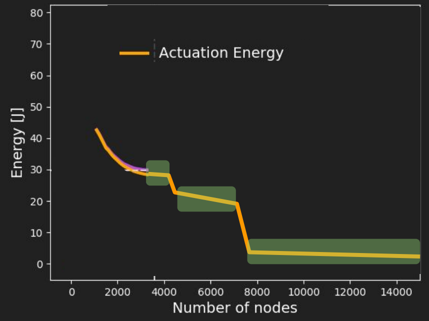
Improvement in **same homotopic class**



Improvement from **homotopic class change**

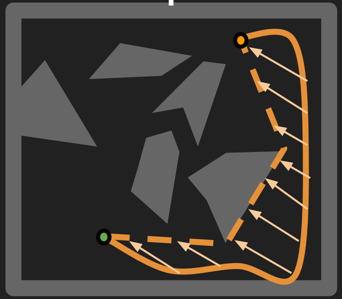




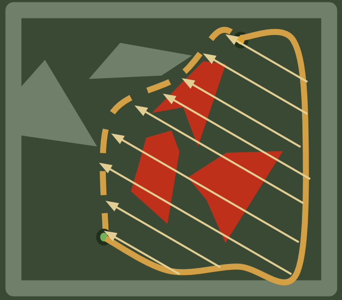


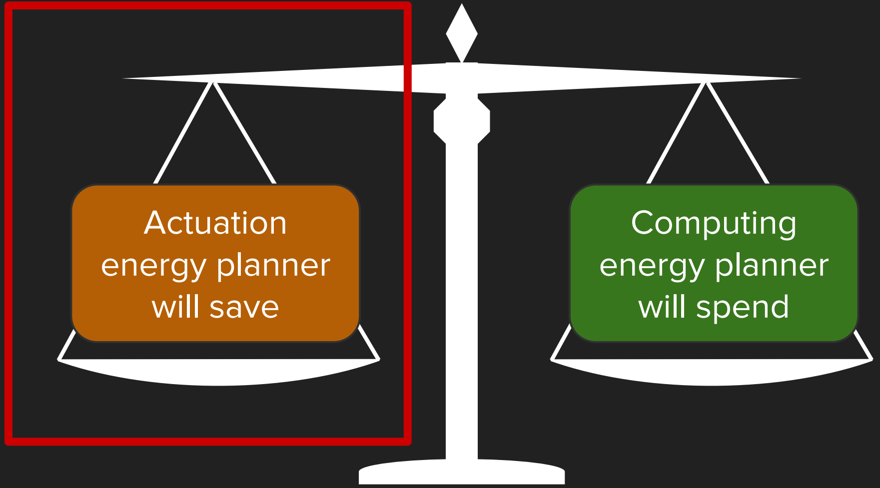
Technical Gap: Knowing How Much Computing is Enough

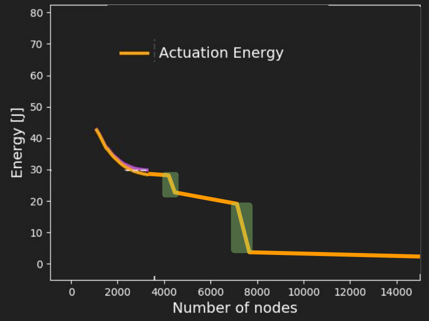
Improvement in **same homotopic class**



Improvement from **homotopic class change**

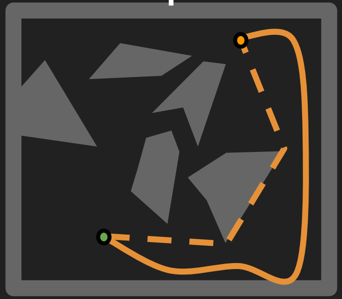




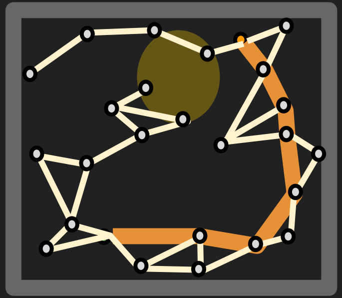


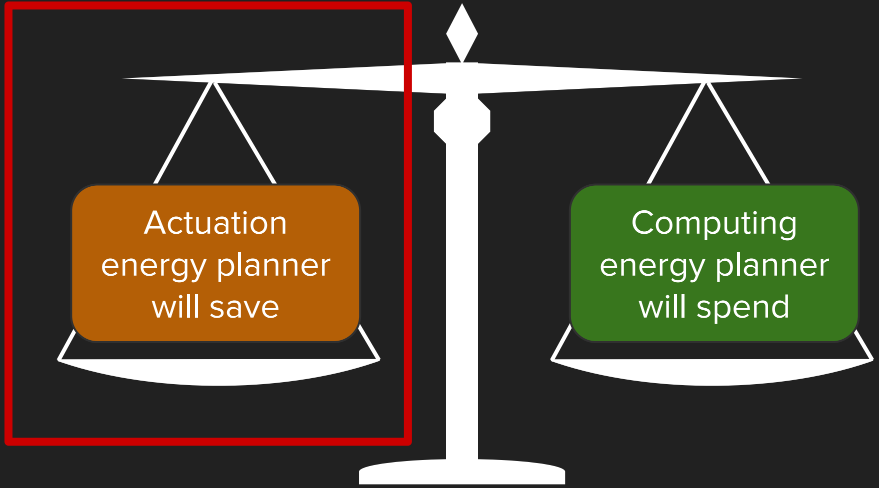
Technical Gap: Knowing How Much Computing is Enough

Improvement in **same homotopic class**



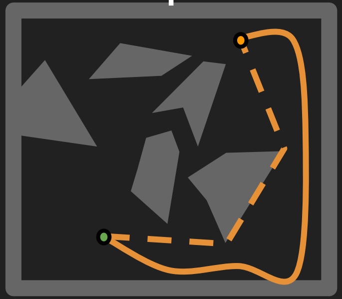
Improvement from **homotopic class change**



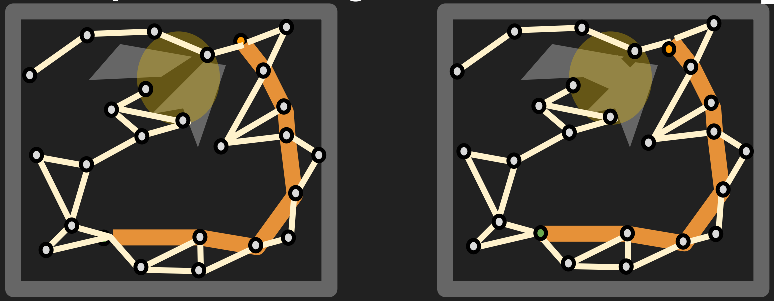


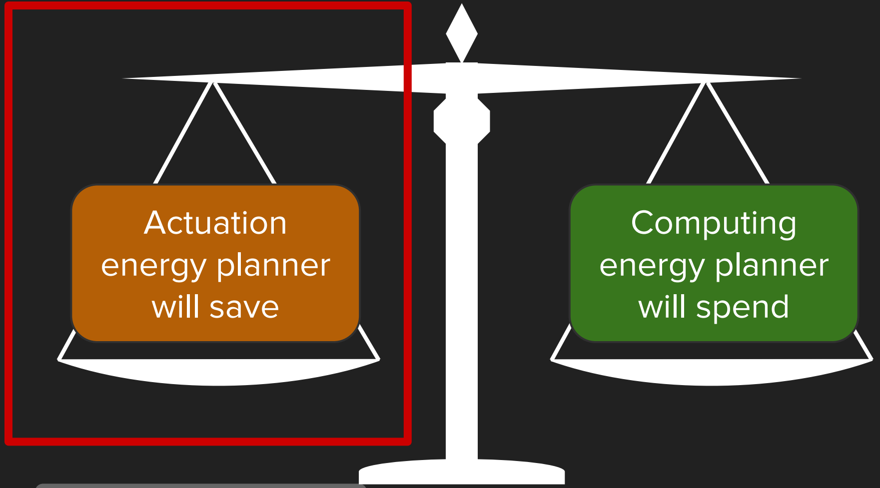
Technical Gap: Knowing How Much Computing is Enough

Improvement in **same homotopic class**



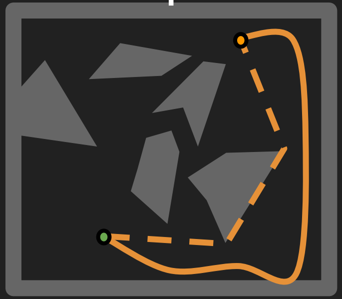
Improvement from **homotopic class change**



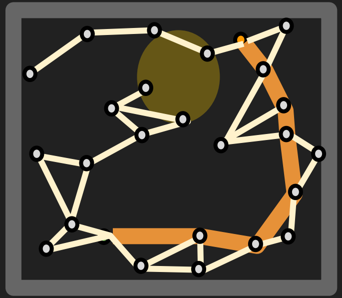


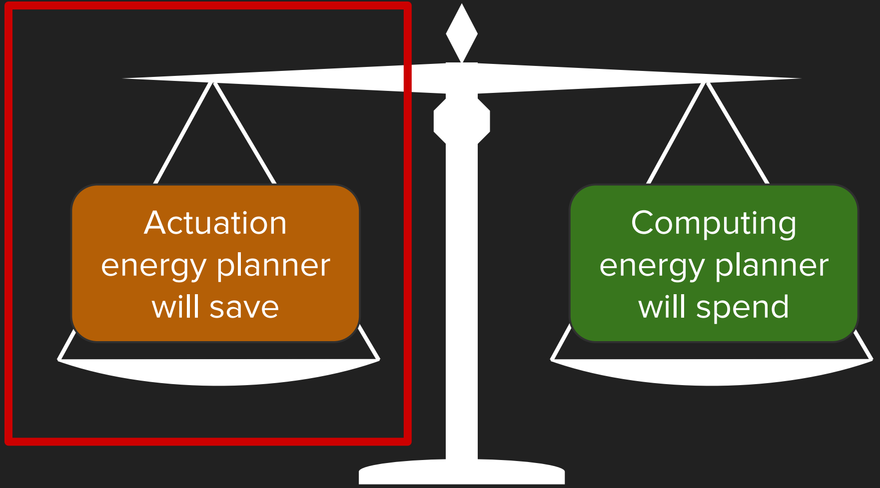
Technical Gap: Knowing How Much Computing is Enough

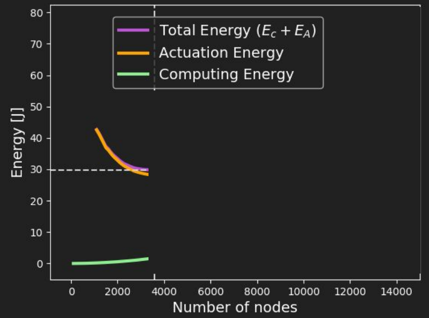
Improvement in **same homotopic class**



Improvement from **homotopic class change**

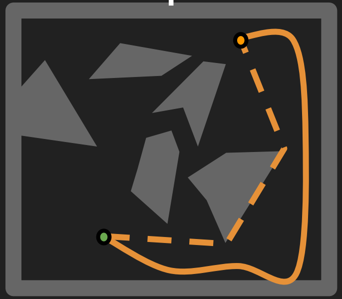




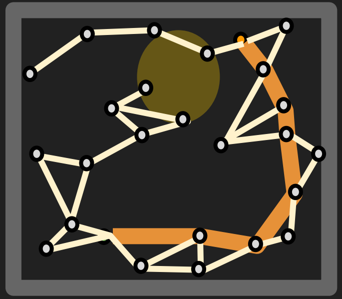


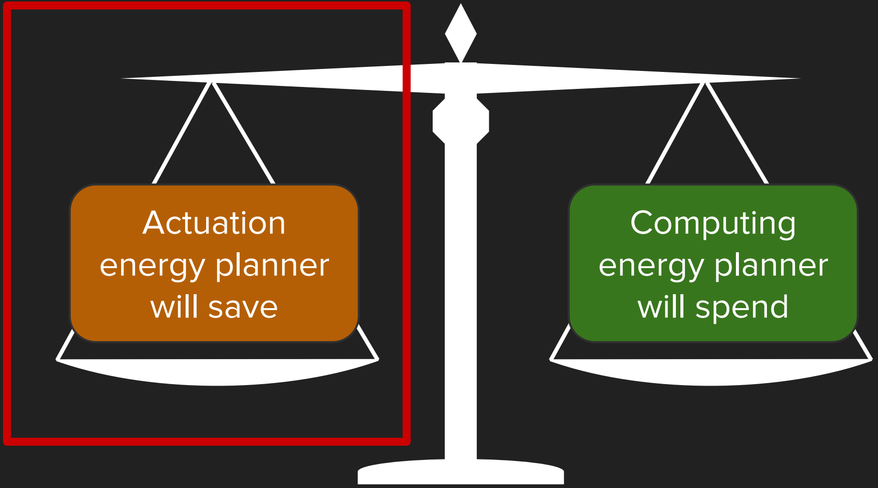
Technical Gap: Knowing How Much Computing is Enough

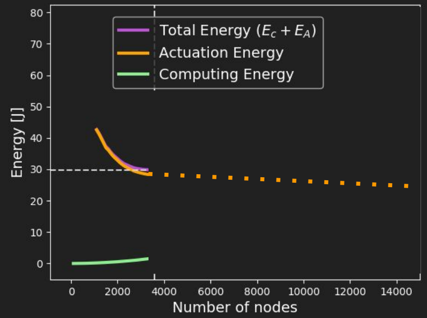
Improvement in **same homotopic class**



Improvement from **homotopic class change**

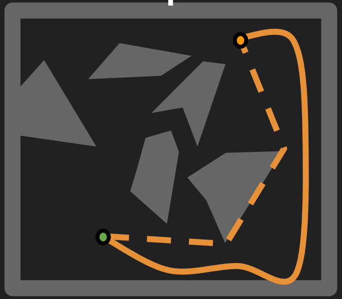




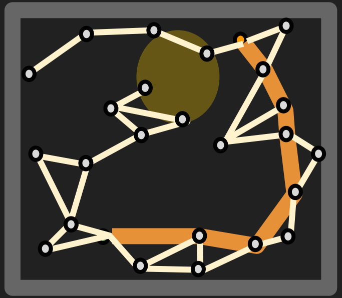


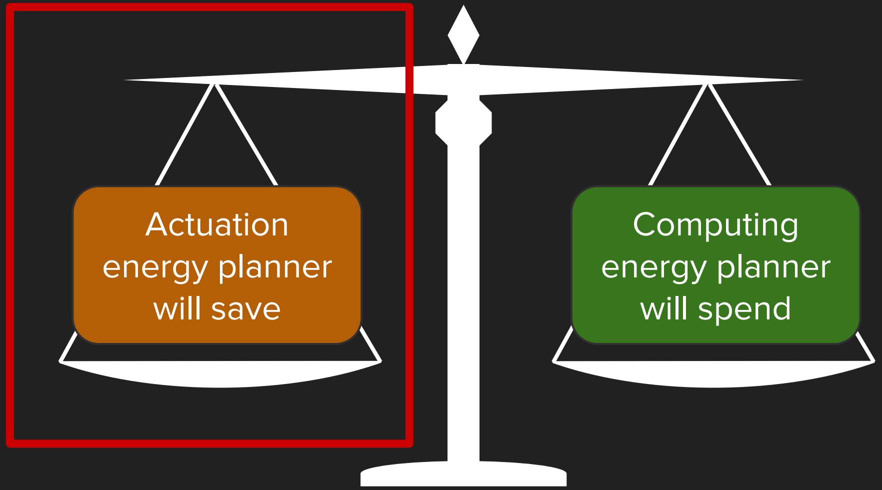
Technical Gap: Knowing How Much Computing is Enough

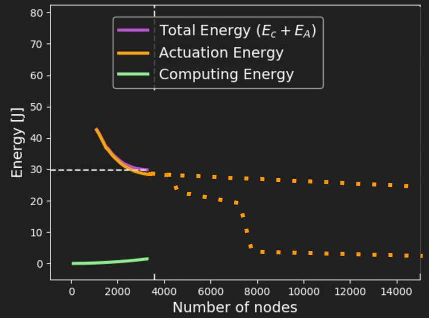
Improvement in **same homotopic class**



Improvement from **homotopic class change**

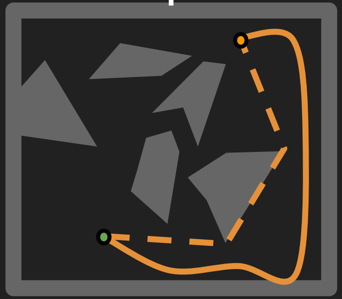




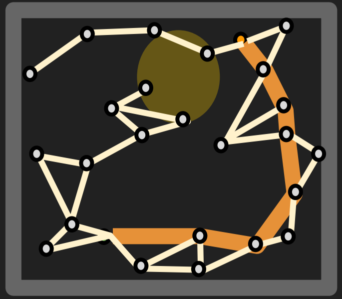


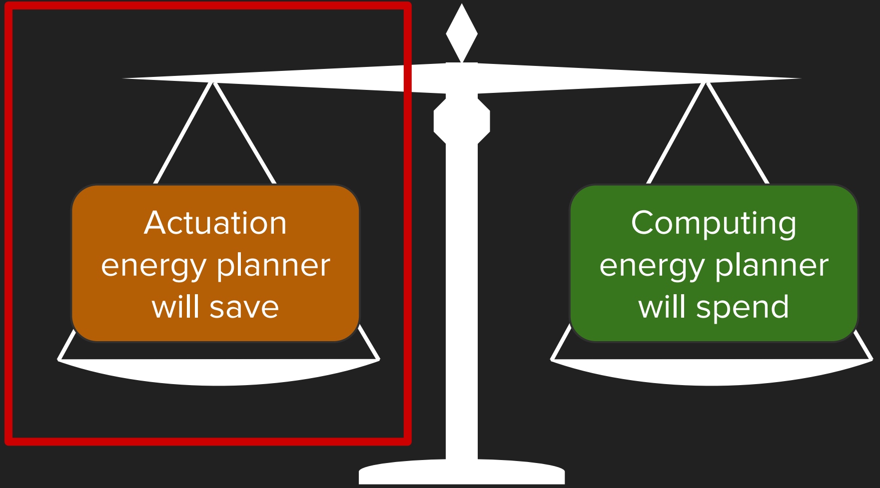
Technical Gap: Knowing How Much Computing is Enough

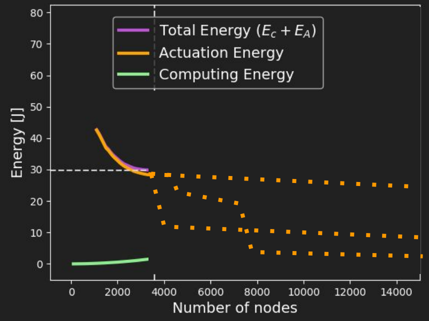
Improvement in **same homotopic class**



Improvement from **homotopic class change**

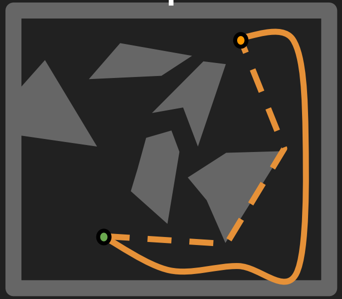




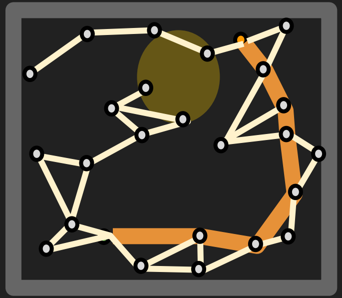


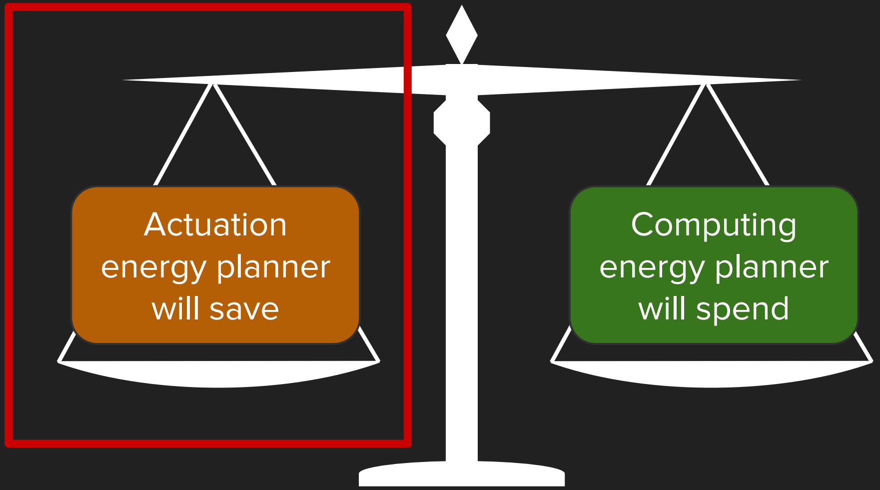
Technical Gap: Knowing How Much Computing is Enough

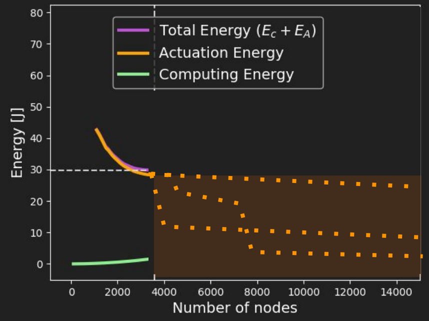
Improvement in **same homotopic class**



Improvement from **homotopic class change**

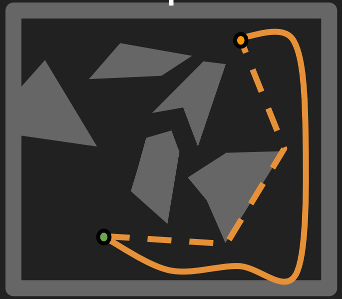




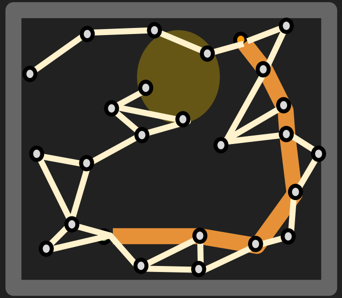


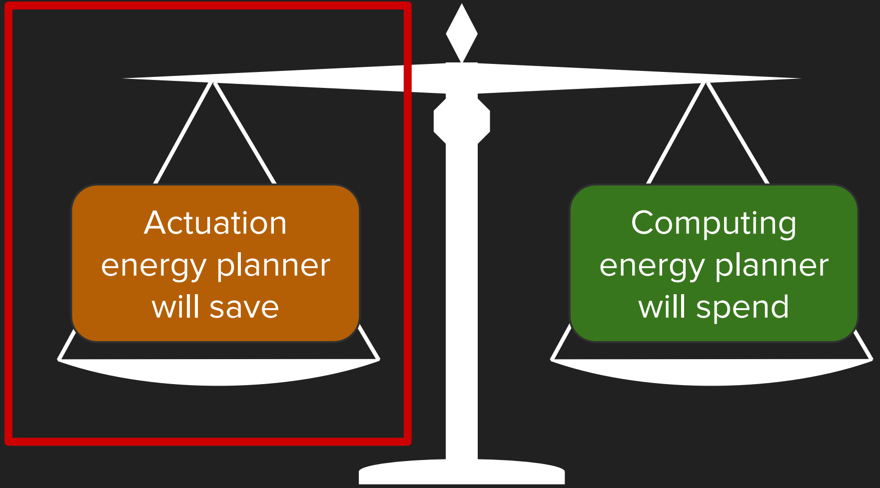
Technical Gap: Knowing How Much Computing is Enough

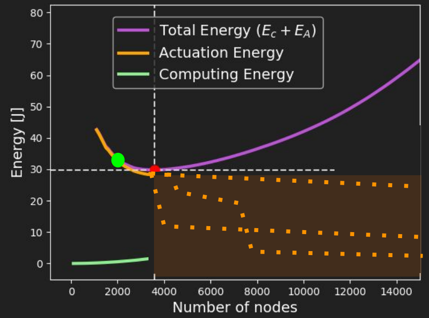
Improvement in **same homotopic class**



Improvement from **homotopic class change**

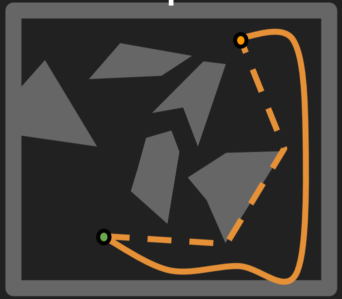




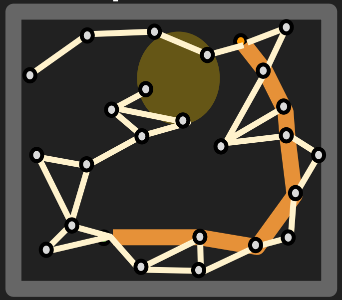


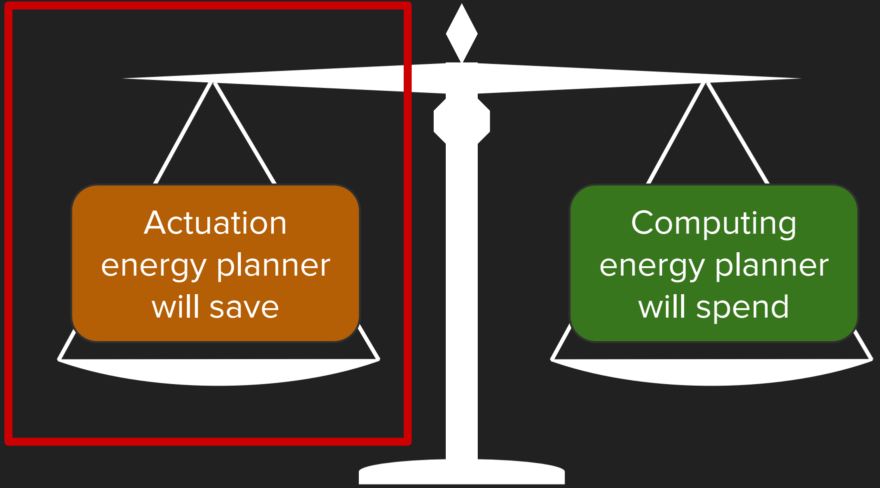
Technical Gap: Knowing How Much Computing is Enough

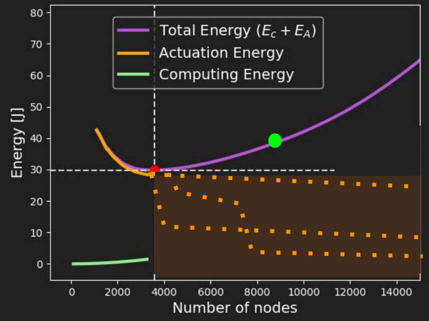
Improvement in **same homotopic class**



Improvement from **homotopic class change**

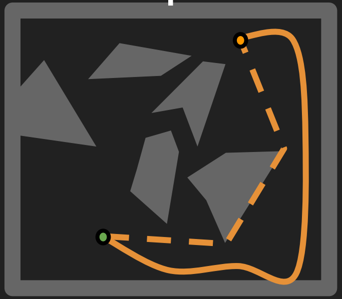




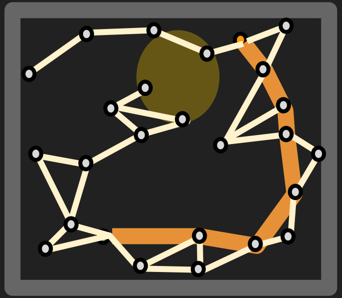


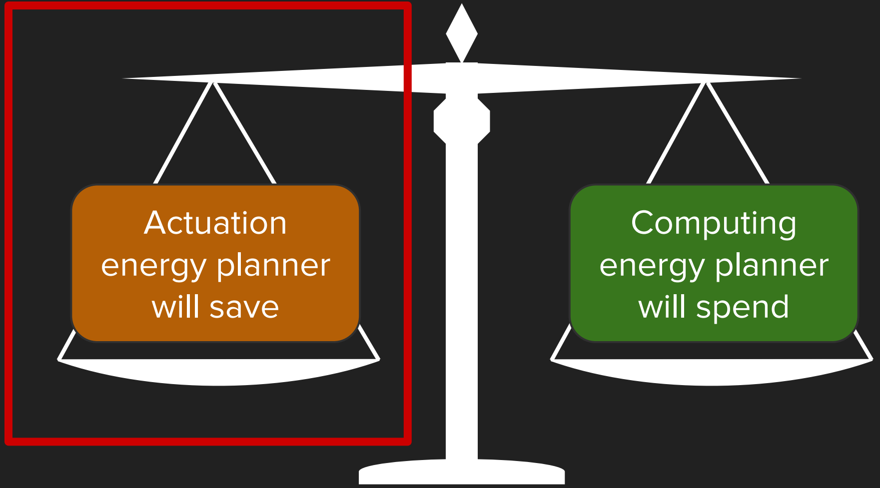
Technical Gap: Knowing How Much Computing is Enough

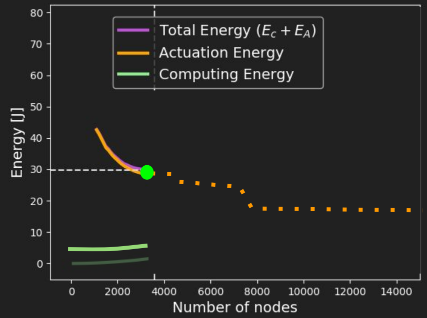
Improvement in **same homotopic class**



Improvement from **homotopic class change**

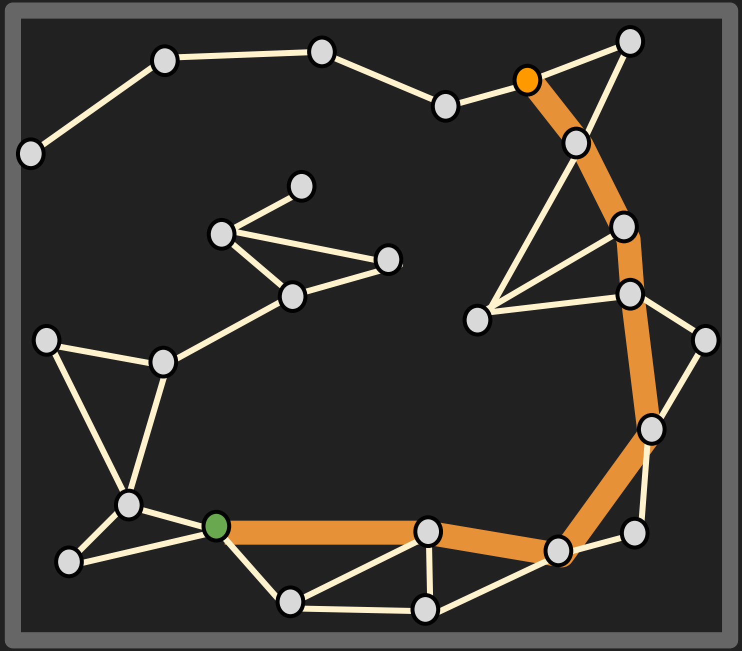




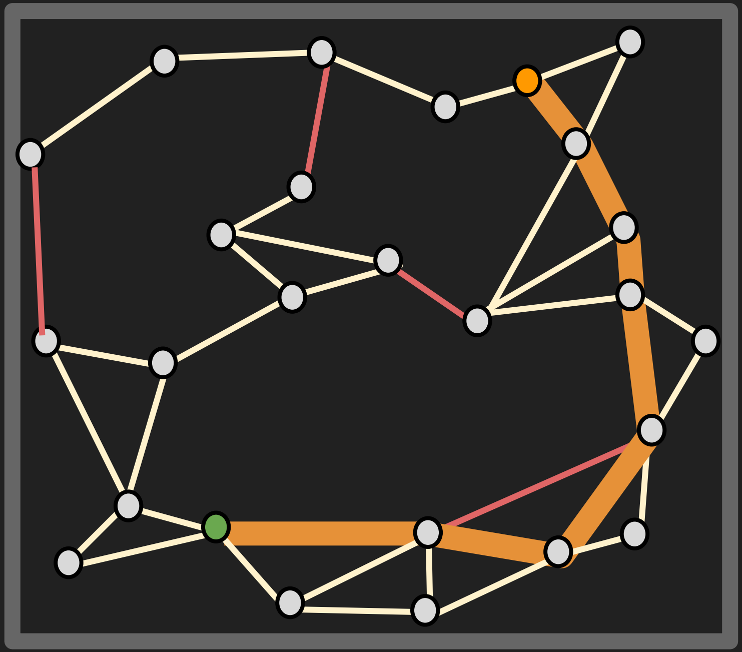


This work:
Computing Energy Included Motion Planning (CEIMP)

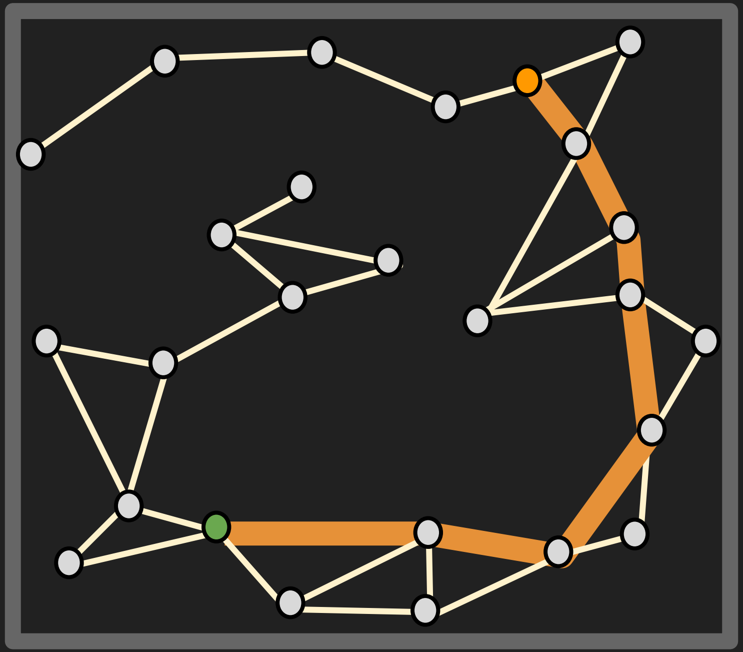
CEIMP: Underlying Motion Planner



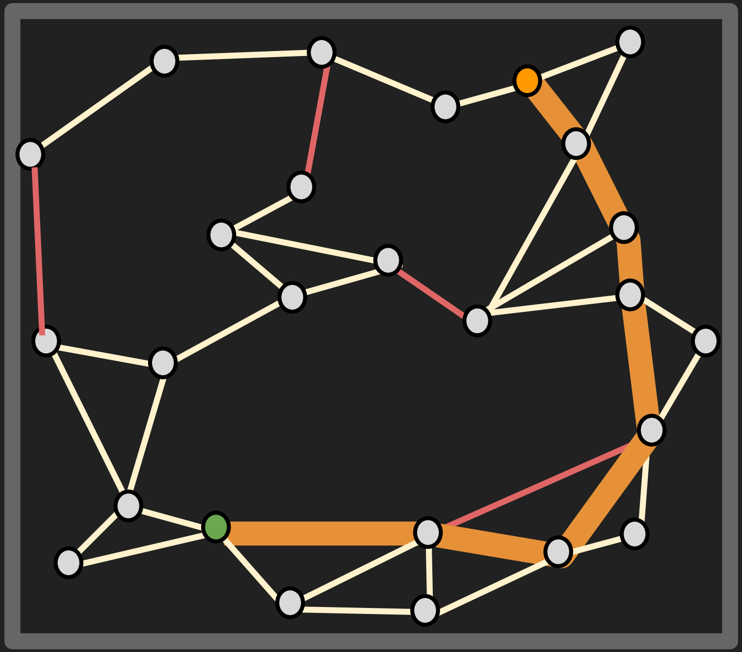
CEIMP: Track edges in collision with the obstacles to "probe" the environment for new homotopic classes



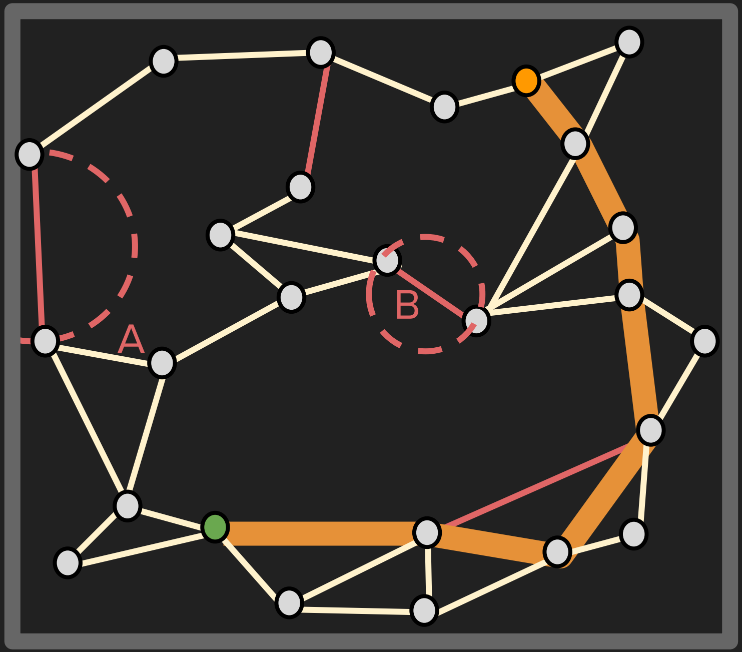
CEIMP: Track edges in collision with the obstacles to "probe" the environment for new homotopic classes



CEIMP: Track edges in collision with the obstacles to “probe” the environment for new homotopic classes

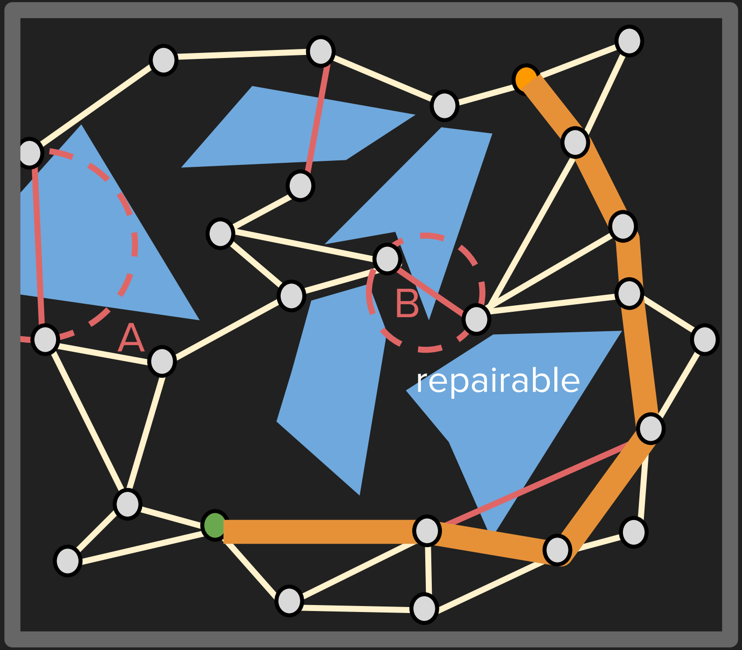


CEIMP: Model the state of each currently edge in collision as ‘repairable’ or ‘unrepairable’

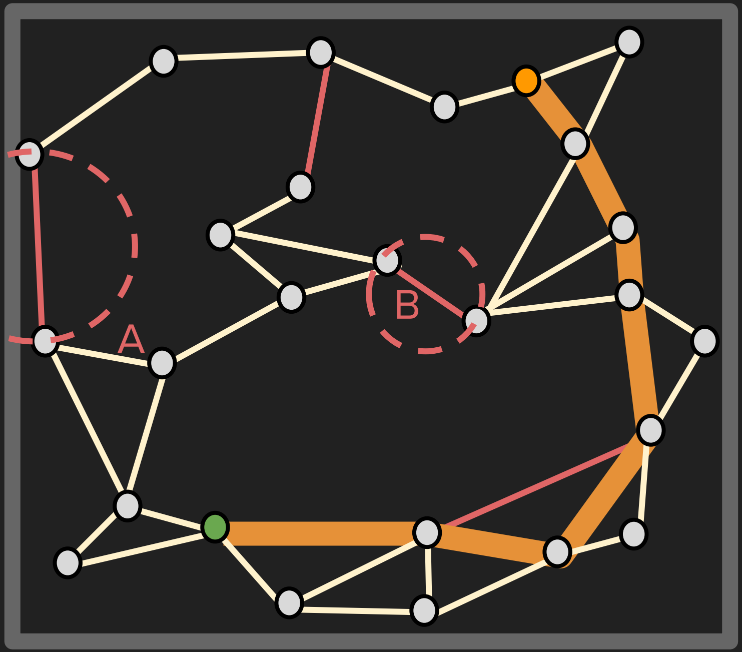


CEIMP: Model the state of each currently edge in collision as ‘repairable’ or ‘unrepairable’

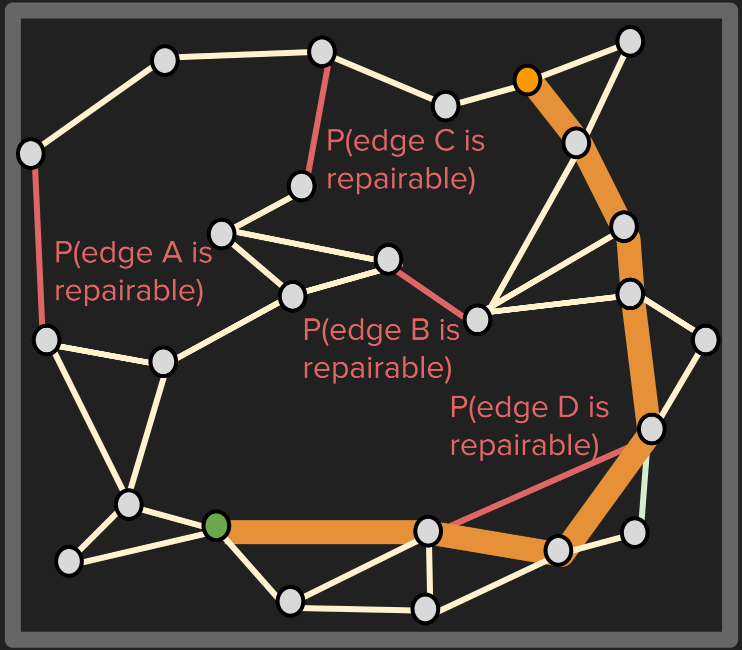
unrepairable



CEIMP: Model the state of each currently edge in collision as ‘repairable’ or ‘unrepairable’

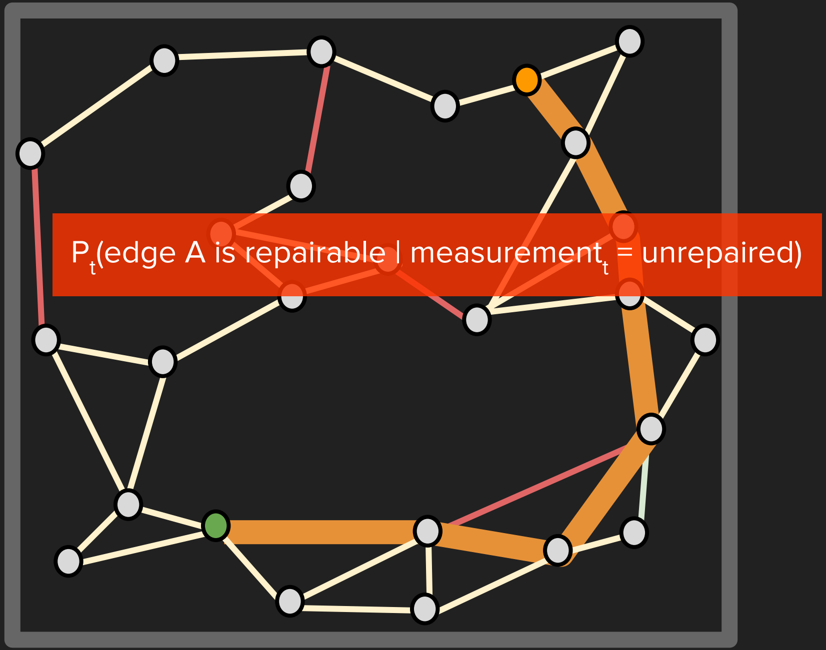


CEIMP: Estimate the probability an edge in collision's state is 'repairable'



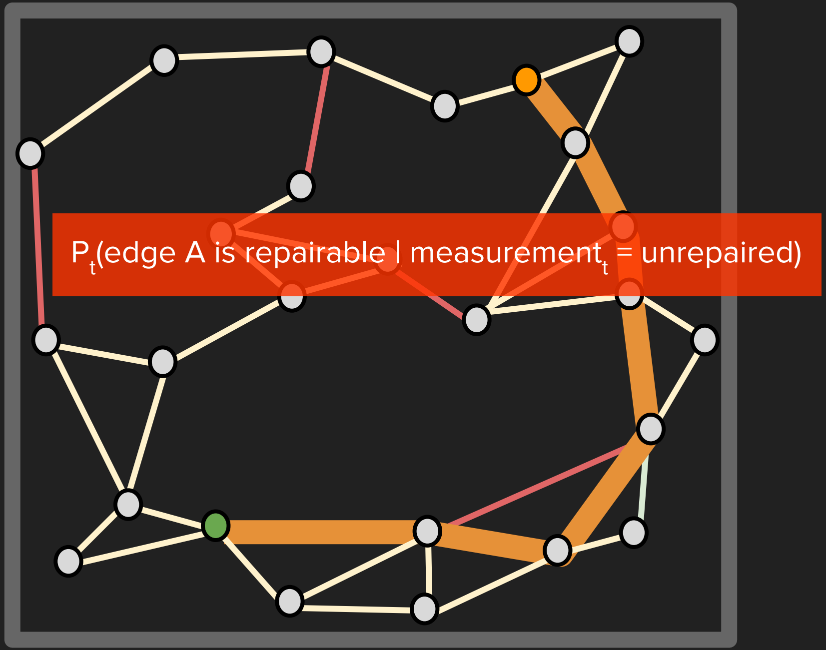
CEIMP: Estimate the probability an edge in collision's state is 'repairable'

Model computing a batch of nodes as a noisy sensor that returns a reading of **repaired** or **unrepaired**



CEIMP: Estimate the probability an edge in collision's state is 'repairable'

Model computing a batch of nodes as a noisy sensor that returns a reading of **repaired** or **unrepaired**



What is the probability an edge in collision is a repairable edge, given we haven't been able to repair it yet?

Computing as a Measurement

edge $\in$ {**repairable**, **unrepairable**}
measurement from sampling a batch of nodes $\in$ {**repaired**, **unrepaired**}

Computing as a Measurement

edge ∈ {**repairable**, **unrepairable**}
measurement from sampling a batch of nodes ∈ {**repaired**, **unrepaired**}

$$P(\text{measurement} = \textbf{unrepaired}) = \begin{cases} a & \text{if edge is } \textbf{repairable} \\ 1 & \text{if edge is } \textbf{unrepairable} \end{cases}$$

Computing as a Measurement

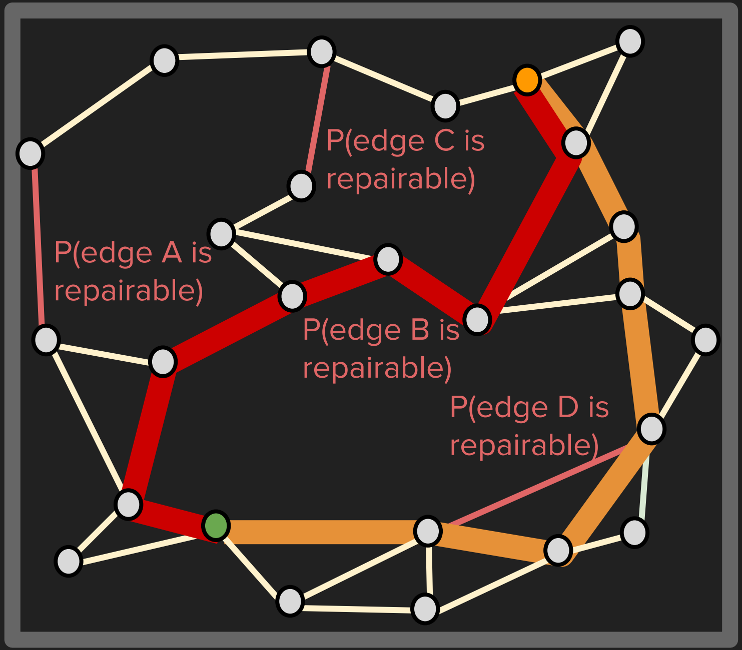
edge $\in$ {**repairable**, **unrepairable**}
measurement from sampling a batch of nodes $\in$ {**repaired**, **unrepaired**}

$$P(\text{measurement} = \textbf{unrepaired}) = \begin{cases} a & \text{if edge is } \textbf{repairable} \\ 1 & \text{if edge is } \textbf{unrepairable} \end{cases}$$

Binary Bayesian filtering

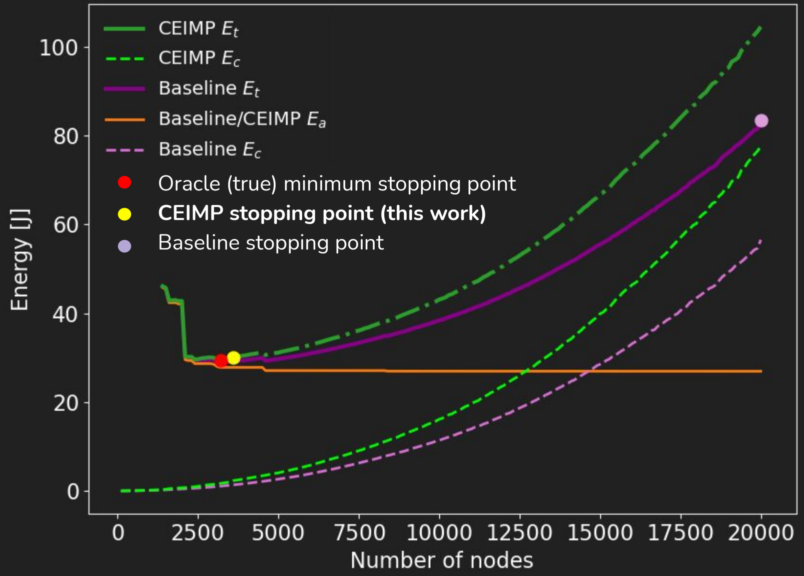
$$P_t(\text{edge A is } \textbf{repairable} \mid \text{measurement}_t = \textbf{unrepaired})$$
$$= \eta P(\text{measurement} = \textbf{unrepaired} \mid \text{edge A is } \textbf{repairable}) P_{t-1}(\text{edge A is } \textbf{repairable})$$

CEIMP: Run a search algorithm on probabilistic graph to return the shortest expected path

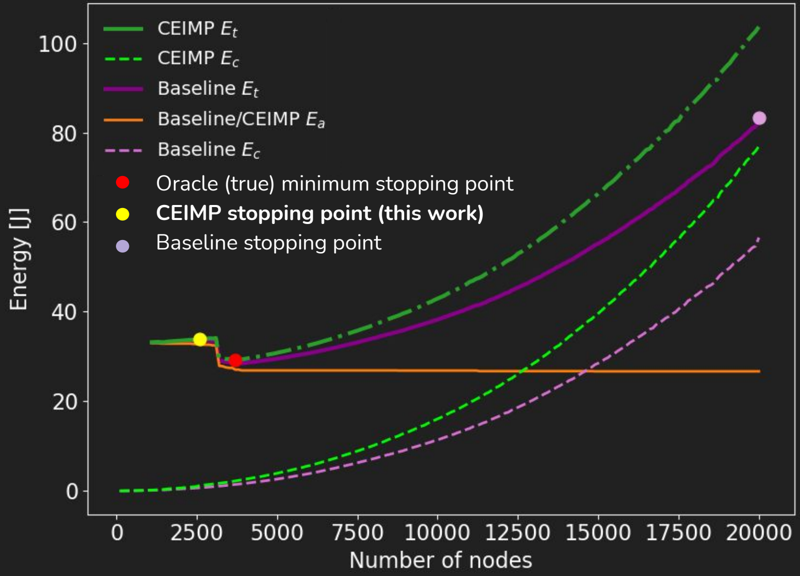


Experimental Results

Simulated vehicle that can travel 1 m/s at 1 Watt, computing on a Cortex A15 (embedded CPU)



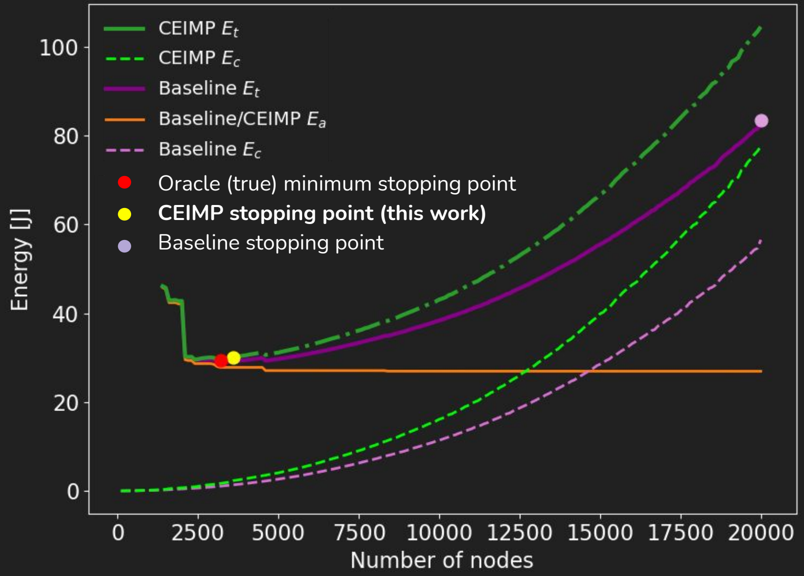
CEIMP successful at stopping close to true minimum



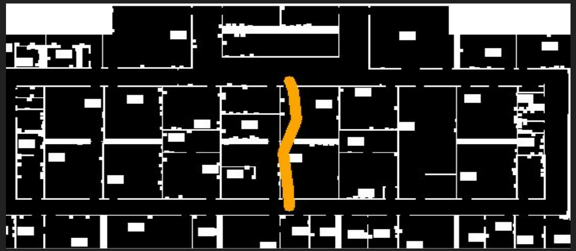
CEIMP stops too early, misses savings from homotopic class change

Experimental Results

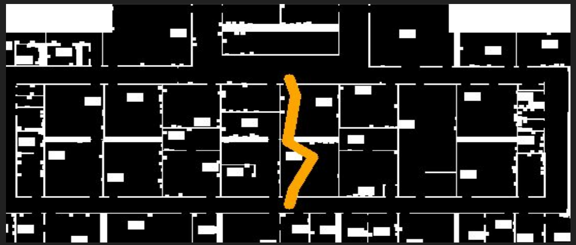
Simulated vehicle that can travel 1 m/s at 1 Watt, computing on a Cortex A15 (embedded CPU)



CEIMP successful at stopping close to true minimum



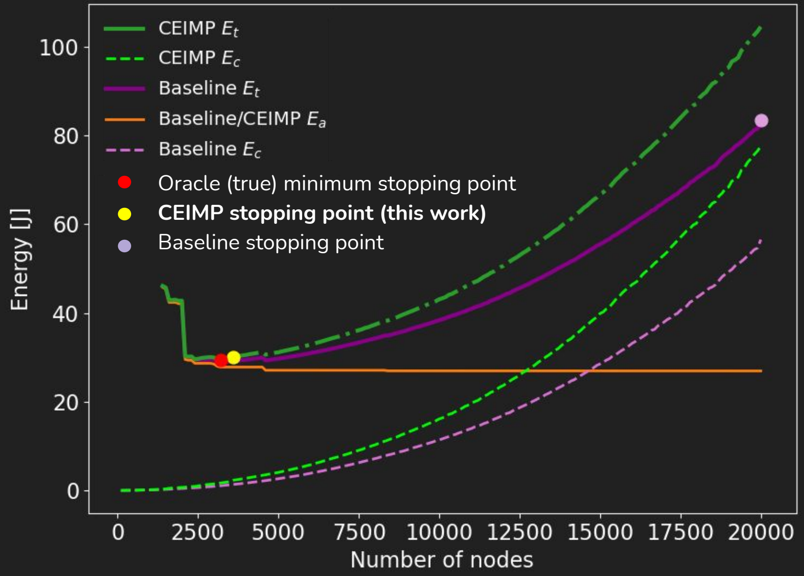
Example path returned by baseline



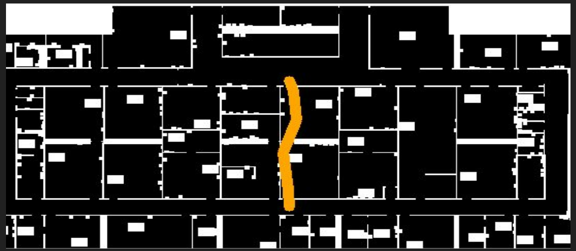
Example path returned by CEIMP

Experimental Results

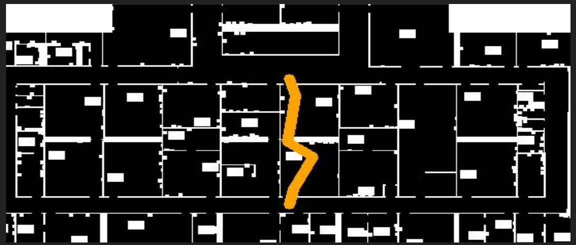
Simulated vehicle that can travel 1 m/s at 1 Watt, computing on a Cortex A15 (embedded CPU)



CEIMP successful at stopping close to true minimum



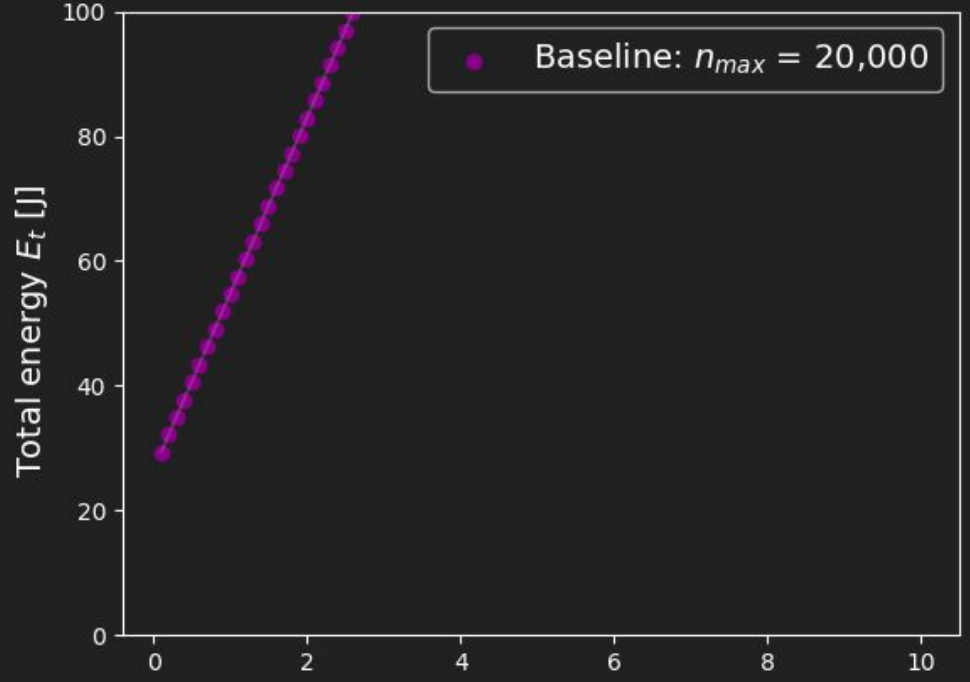
Example path returned by baseline



Example path returned by CEIMP

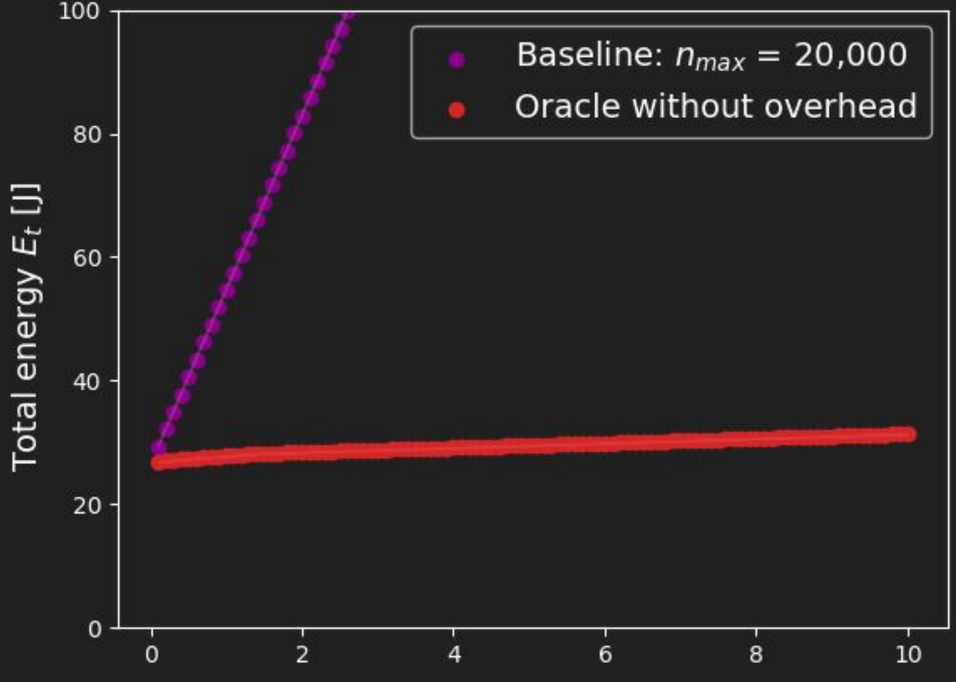
On average across 10 MIT floorplans, CEIMP saves **2.1x-8.9x** the energy compared to baseline

As the the energy to compute becomes more expensive relative to the energy to move, CEIMP will increase energy savings



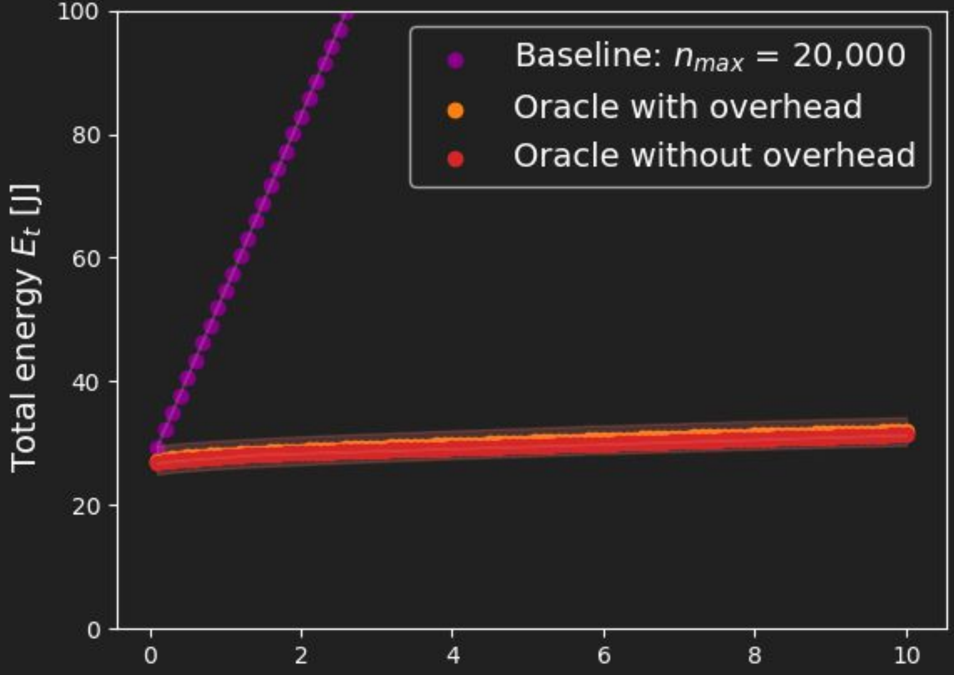
Energy to compute 1 sec relative to the energy to move 1 meter

As the the energy to compute becomes more expensive relative to the energy to move, CEIMP will increase energy savings



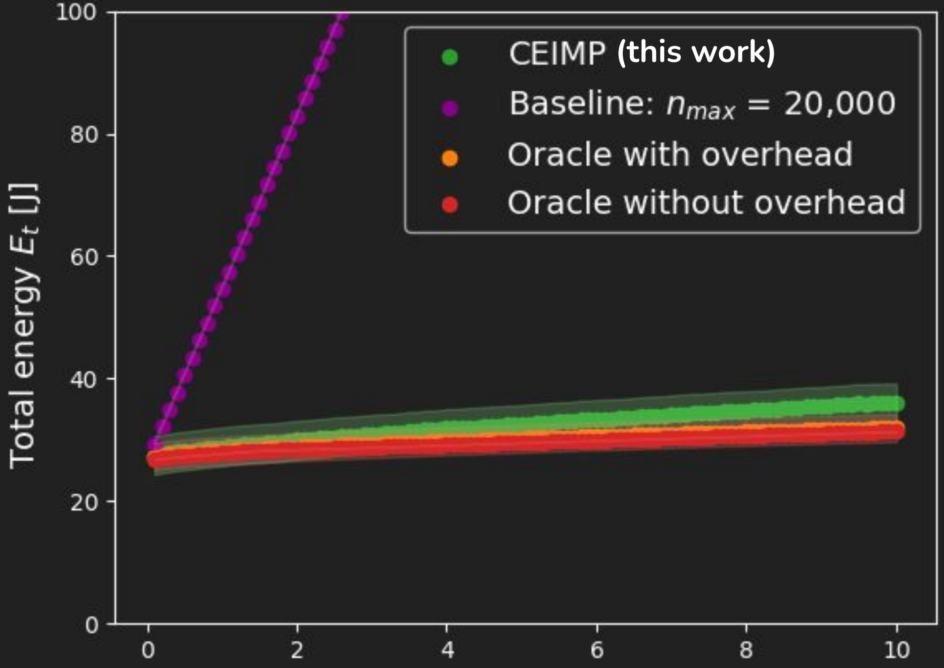
Energy to compute 1 sec relative to the energy to move 1 meter

As the the energy to compute becomes more expensive relative to the energy to move, CEIMP will increase energy savings



Energy to compute 1 sec relative to the energy to move 1 meter

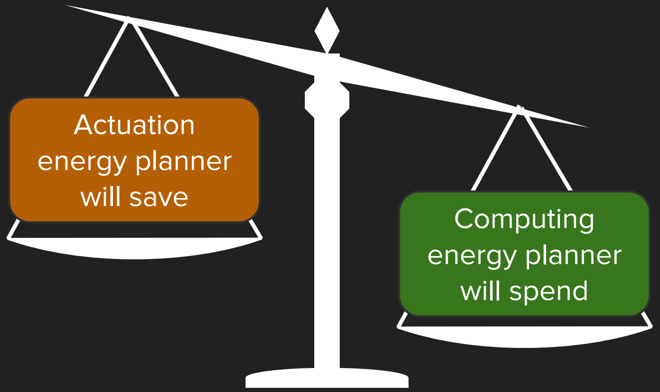
As the the energy to compute becomes more expensive relative to the energy to move, CEIMP will increase energy savings



Energy to compute 1 sec relative to the energy to move 1 meter

Key Takeaways

- **Don't think *too* hard:** A longer path that we have now can be better than a shorter path that we have to compute a long time to find

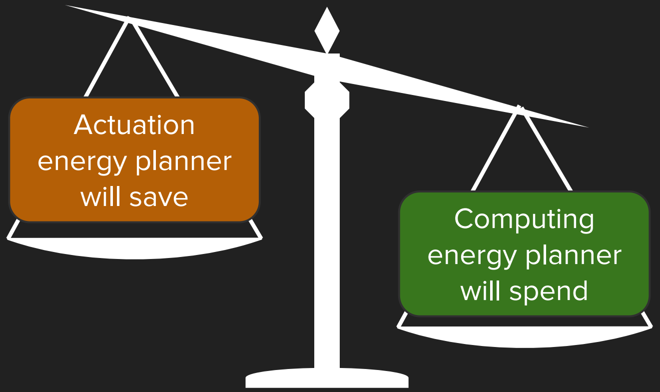


Sudhakar, Soumya, Sertac Karaman, and Vivienne Sze. "Balancing Actuation and Computing Energy in Motion Planning." *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020.

This work was funded by the National Science Foundation, Cyber-Physical Systems (CPS) program, through grant no. 1837212

Key Takeaways

- **Don't think *too* hard:** A longer path that we have now can be better than a shorter path that we have to compute a long time to find
- **Computing is (noisy) sensing:** Sampling nodes in a motion planner can be modeled as a noisy sensor that returns whether a path is open or closed



Sudhakar, Soumya, Sertac Karaman, and Vivienne Sze. "Balancing Actuation and Computing Energy in Motion Planning." *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020.

This work was funded by the National Science Foundation, Cyber-Physical Systems (CPS) program, through grant no. 1837212