16-350
Planning Techniques for Robotics

Planning Representations: Lattice-based Graphs

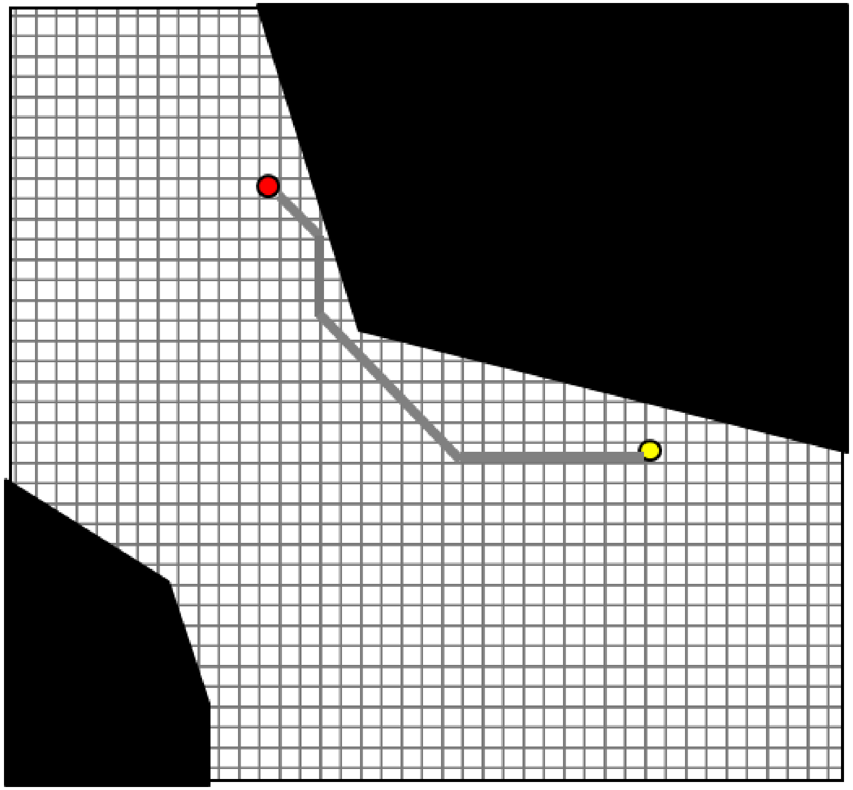
Maxim Likhachev

Robotics Institute

Carnegie Mellon University

Beyond Planning for Omnidirectional Robots

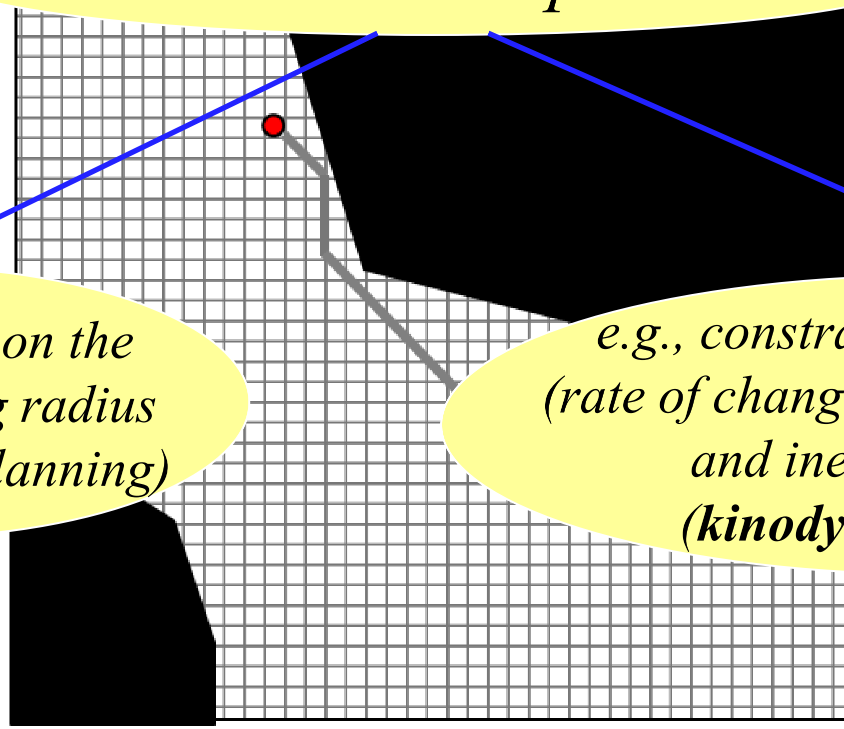
What's wrong with using
Grid-based Graphs when planning
for non-omnidirectional robots?

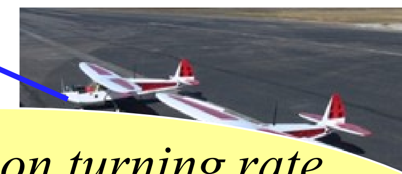


Beyond Planning for Omnidirectional Robots

What's wrong with using
Grid-based Graphs when planning
for non-omnidirectional robots?

"Can't turn in place"





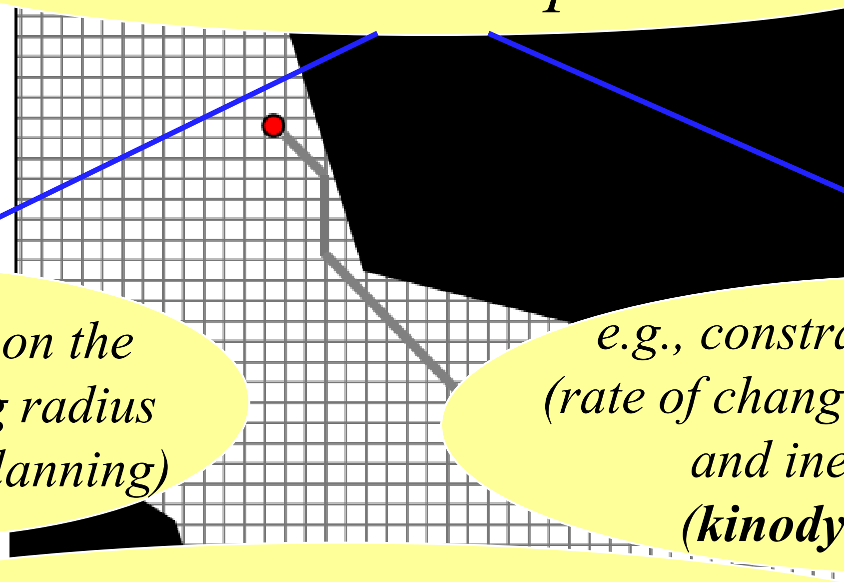
*e.g., constraints on the minimum turning radius (still **kinematic** planning)*

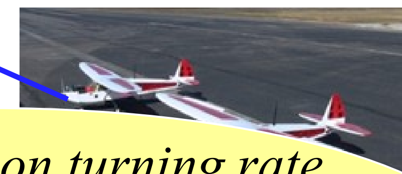
*e.g., constraints on turning rate (rate of change in wheel orientation) and inertial constraints (**kinodynamic** planning)*

Beyond Planning for Omnidirectional Robots

What's wrong with using
Grid-based Graphs when planning
for non-omnidirectional robots?

"Can't turn in place"





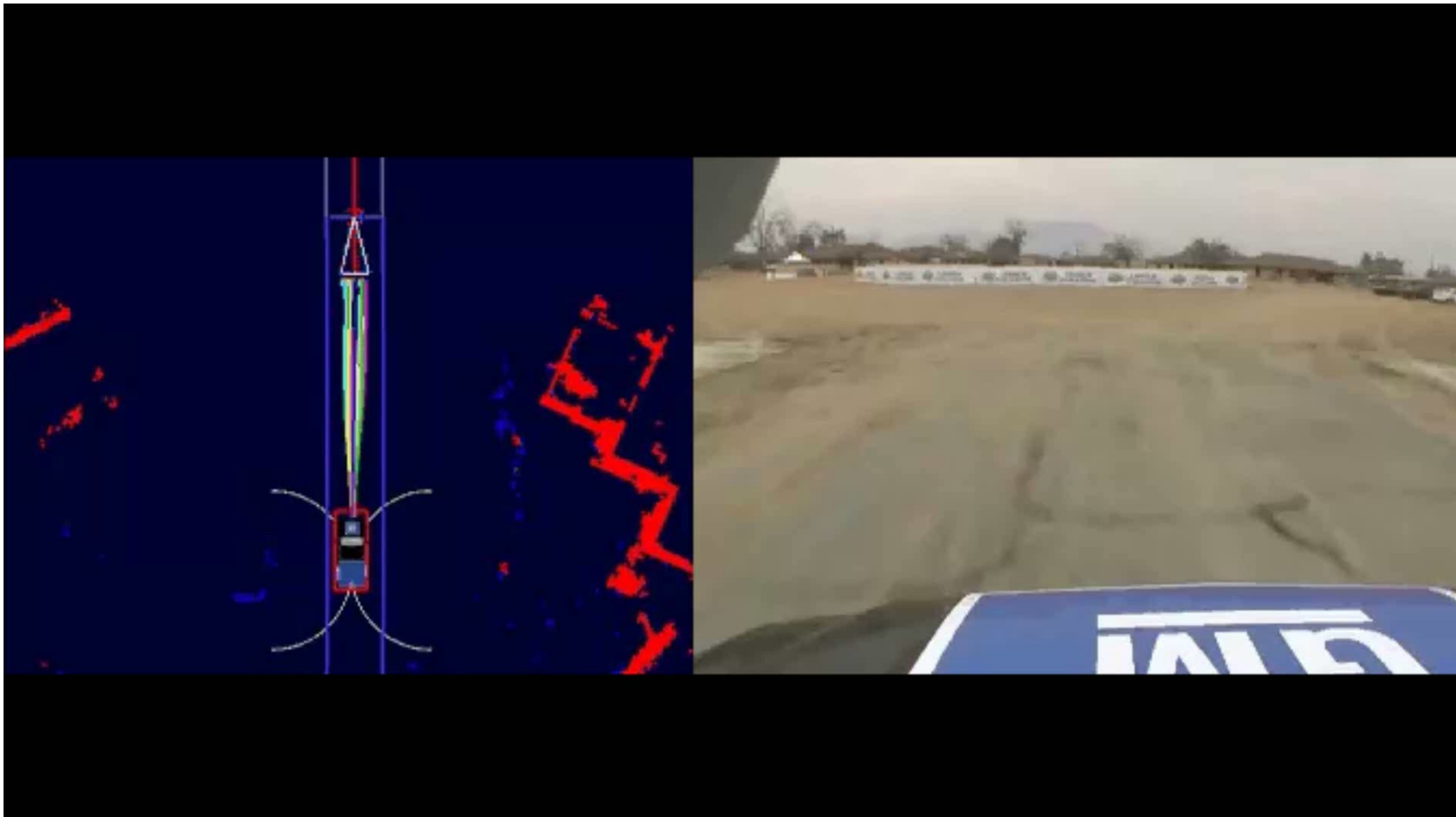
*e.g., constraints on the minimum turning radius (still **kinematic** planning)*

*e.g., constraints on turning rate (rate of change in wheel orientation) and inertial constraints (**kinodynamic** planning)*

***Kinodynamic planning**:*
Planning representation includes $\{X, \dot{X}\}$,
where X-configuration and $\dot{X}$-derivative of X (dynamics of X)

Beyond Planning for Omnidirectional Robots

(x,y,Θ,v) planning
with Anytime D (Anytime Incremental A*) on Lattice Graphs*



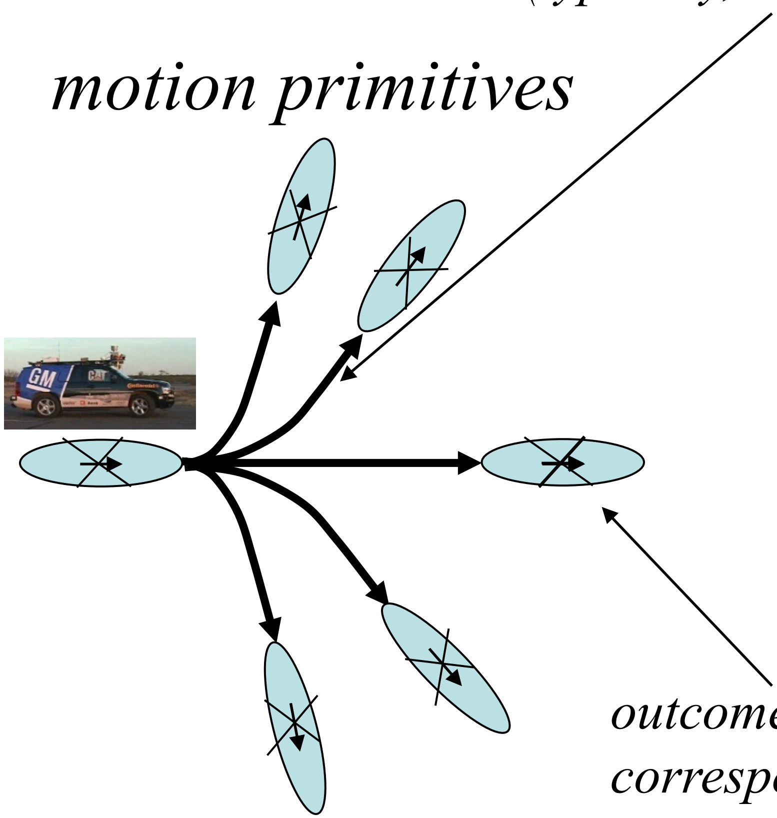
Beyond Planning for Omnidirectional Robots

(x,y,Θ) planning with ARA-based algorithm on Lattice Graphs*

Joint work with V. Kumar (Upenn), I. Kaminer (NPS) and V. Dobrokhodov (NPS)
[thakur et al., '13]

Lattice Graphs

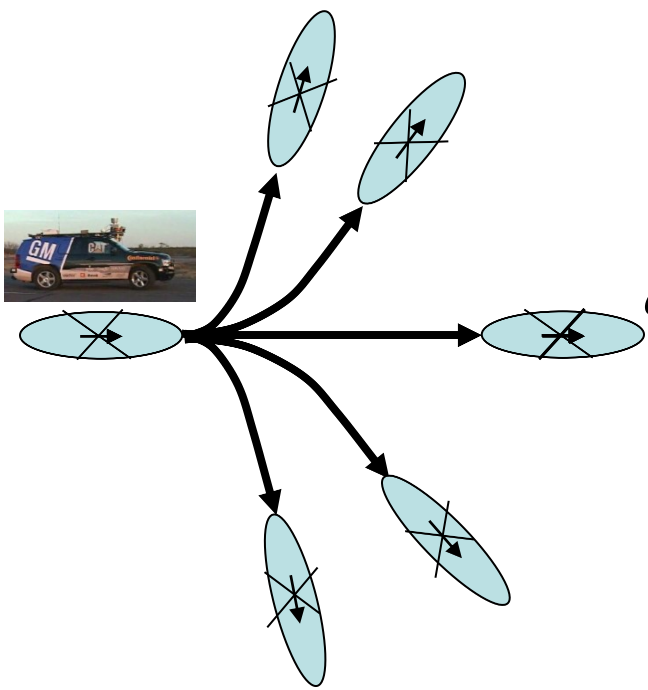
- Graph *{V, E}* where
  - *V*: centers of the grid-cells
  - *E*: motion primitives that connect centers of cells via short-term **feasible** motions



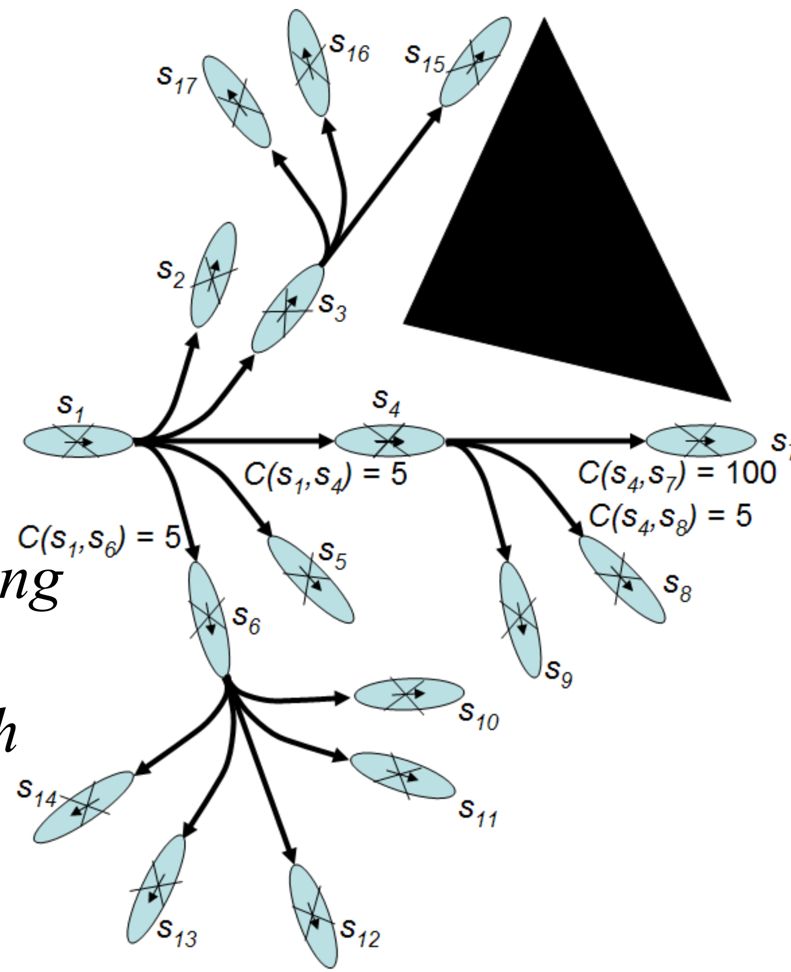
Lattice Graphs

- Graph *{V, E}* where
  - *V*: centers of the grid-cells
  - *E*: motion primitives that connect centers of cells via short-term **feasible** motions

motion primitives

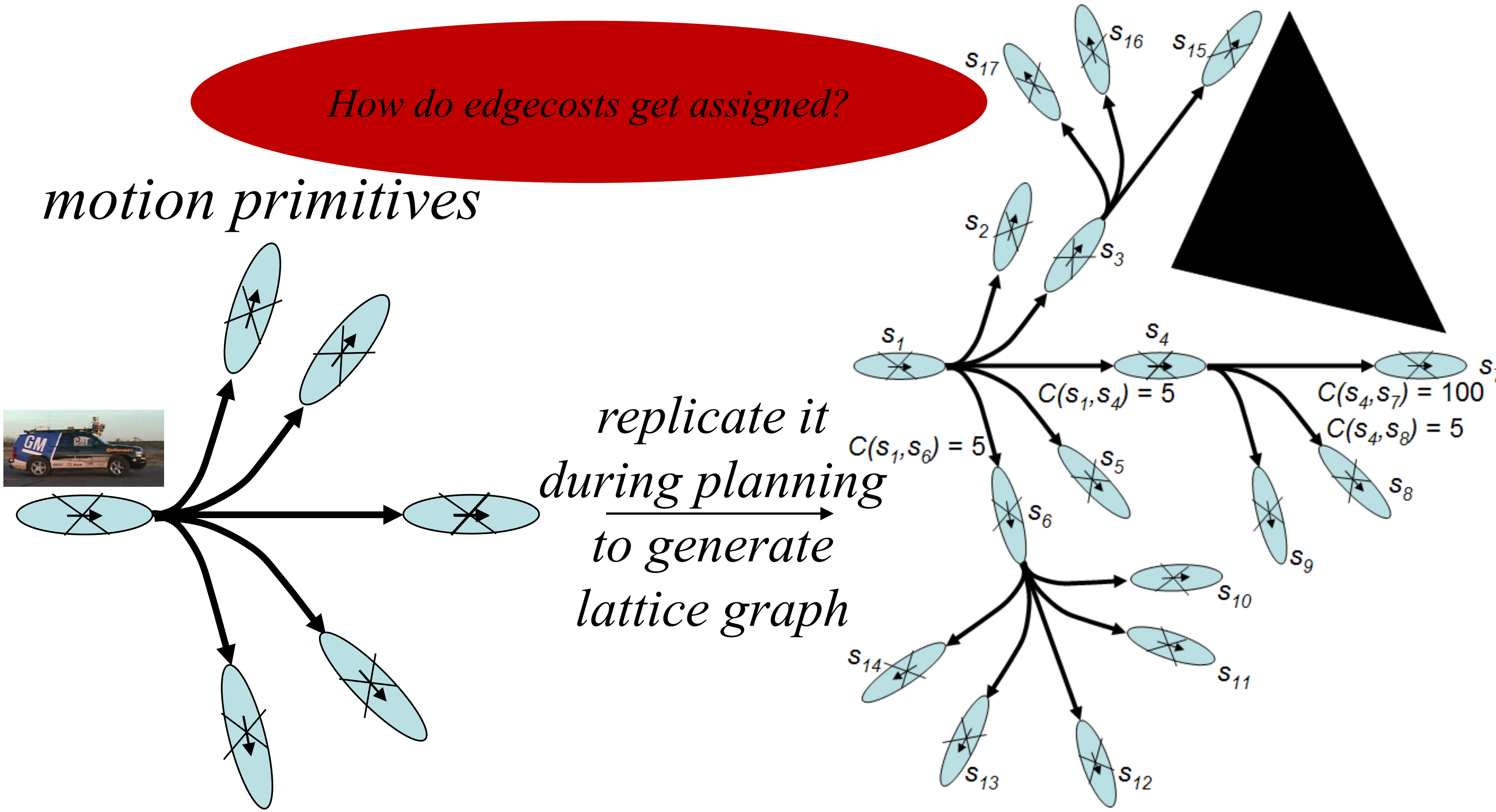


replicate it during planning to generate lattice graph



Lattice Graphs

- Graph *{V, E}* where
  - *V*: centers of the grid-cells
  - *E*: motion primitives that connect centers of cells via short-term **feasible** motions

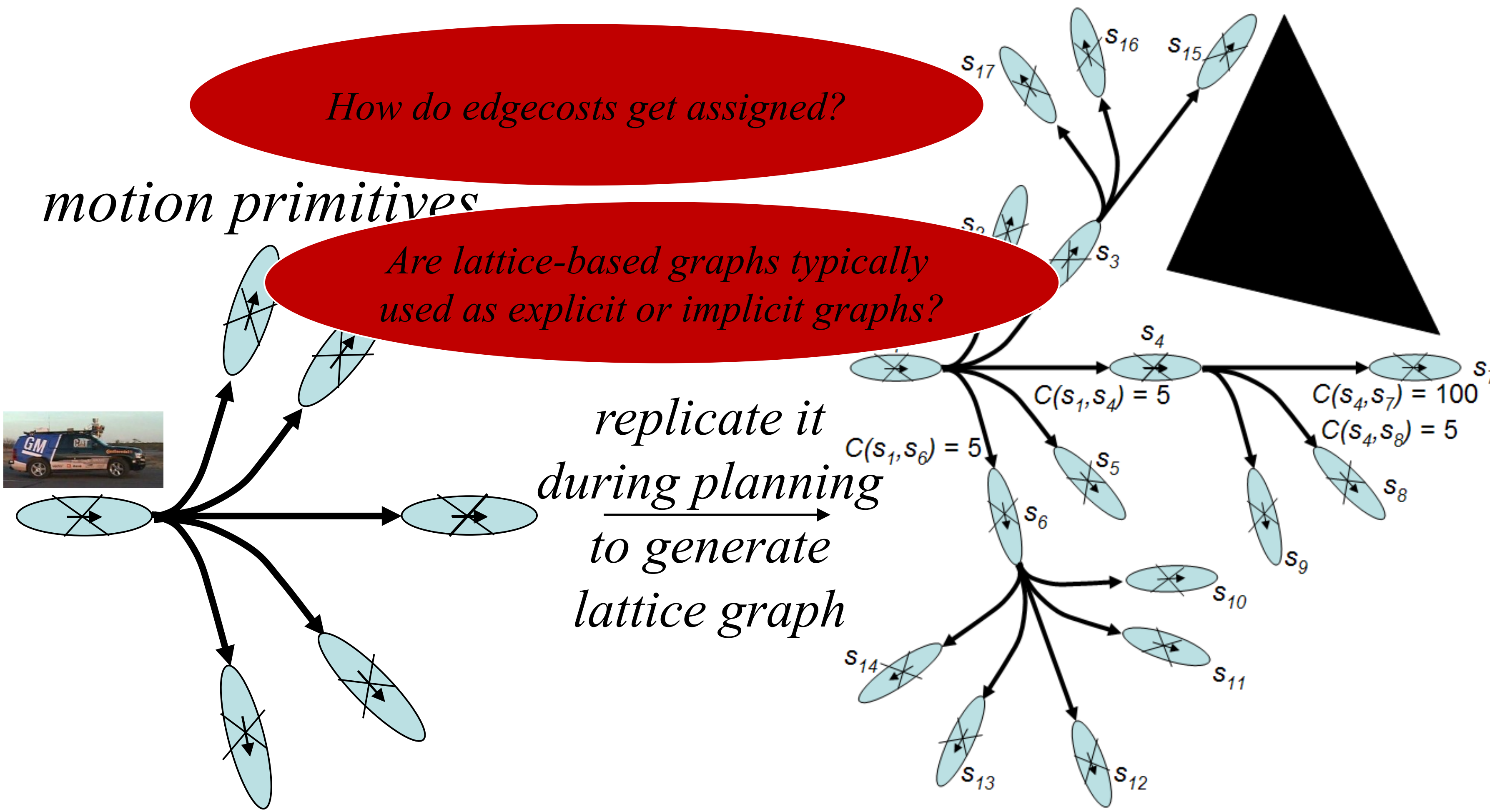


Lattice Graphs

- Graph *{V, E}* where
  - *V*: centers of the grid-cells
  - *E*: motion primitives that connect centers of cells via short-term **feasible** motions

How do edgecosts get assigned?

Are lattice-based graphs typically used as explicit or implicit graphs?



Lattice Graphs

- **Board example** for *(x,y,Θ)* planning for a unicycle model (minimum turning radius)

What You Should Know…

- What are Lattice graphs and how they get constructed