RBE 550 – HW4 Section 6.2: Planner Algorithm

Mohamed Eljahmi
Worcester Polytechnic Institute

Hybrid * A Planner Overview

The project employs a single planner—Hybrid A*—for all vehicle types: the differential-drive robot, the Ackermann-steered car, and the articulated truck–trailer.

Hybrid A* operates in continuous configuration space SE (2) rather than on a discrete grid.
From each state, the planner generates a small set of short, dynamically feasible motion primitives that respect the vehicle's nonholonomic constraints.

Each primitive is simulated using the vehicle's kinematic model to produce a successor pose.
Successors that result in geometric overlaps between the vehicle's oriented rectangular footprint and any obstacle polygon, detected via the Separating Axis Theorem (SAT), are discarded.

The algorithm expands the most promising state first, where $f = g + h$ combines the traveled cost g with a heuristic h based on Euclidean distance and heading difference to the goal.

When a pose lies within a specified positional and angular tolerance of the parking-bay target, the final trajectory is reconstructed by tracing stored parent links.

Although the environment is derived from a 12 × 12 cell layout for obstacle generation, all motion propagation, cost evaluation, and collision checking occur in continuous metric space; the grid is used only to form coarse hash keys for visited-state detection.

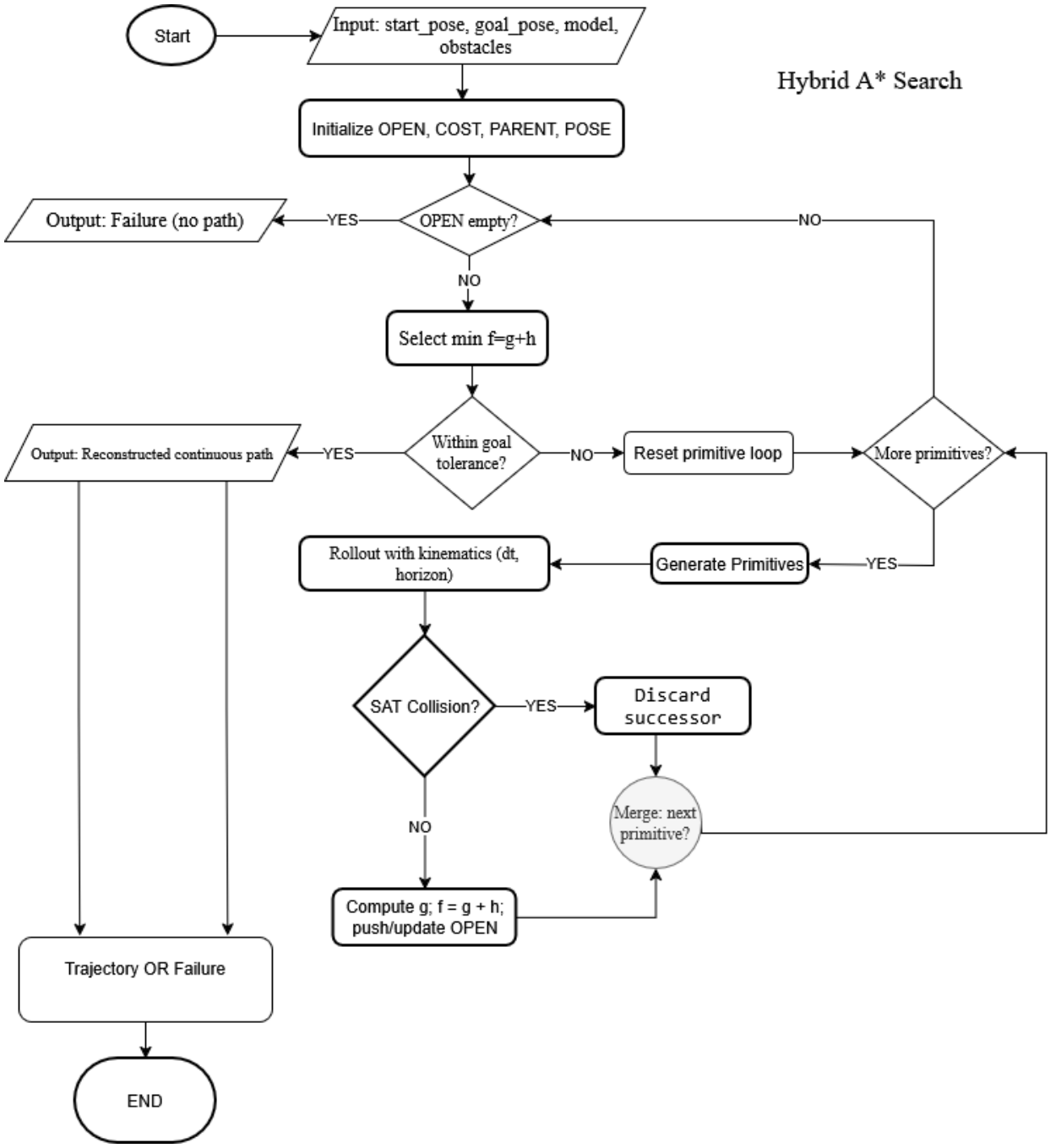
*1. Hybrid * A pseudocode*

Hybrid A* search algorithm

1. **Initialize** the open list with the start pose (store g=0, f=h(start), and parent = None).
2. **While** the open list is **not empty**:
   a. **Select** the node with minimum total cost $f = g + h$.
   b. **Goal test**: if the node is within position/yaw tolerance of the goal, **reconstruct the path and return the trajectory** (continuous poses).
   c. **For each** feasible motion primitive (forward/reverse, left/straight/right):
      – **Roll out** the kinematic model to get a successor pose (continuous SE(2)).
      – **Reject** if any sample on the segment collides (SAT).
      – **Update** cost $g'$and total $f' = g' + h$, set parent, and **push/update** in OPEN.
3. **Failure**: if the open list is exhausted, **return failure** (no path).

Output: **trajectory** (list of continuous poses) **or failure**.

2. ***Flow Chart***


Hybrid A* Search
Start
Input: start_pose, goal_pose, model, obstacles
Initialize OPEN, COST, PARENT, POSE
OPEN empty?
YES
Output: Failure (no path)
NO
NO
Select min f=g+h
Within goal tolerance?
YES
Output: Reconstructed continuous path
NO
Reset primitive loop
More primitives?
YES
Generate Primitives
Rollout with kinematics (dt, horizon)
SAT Collision?
YES
Discard successor
NO
Merge: next primitive?
Compute g; f = g + h; push/update OPEN
Trajectory OR Failure
END