PROFACTOR®

RRT Planner for the Binpicking Problem

Jose Capco

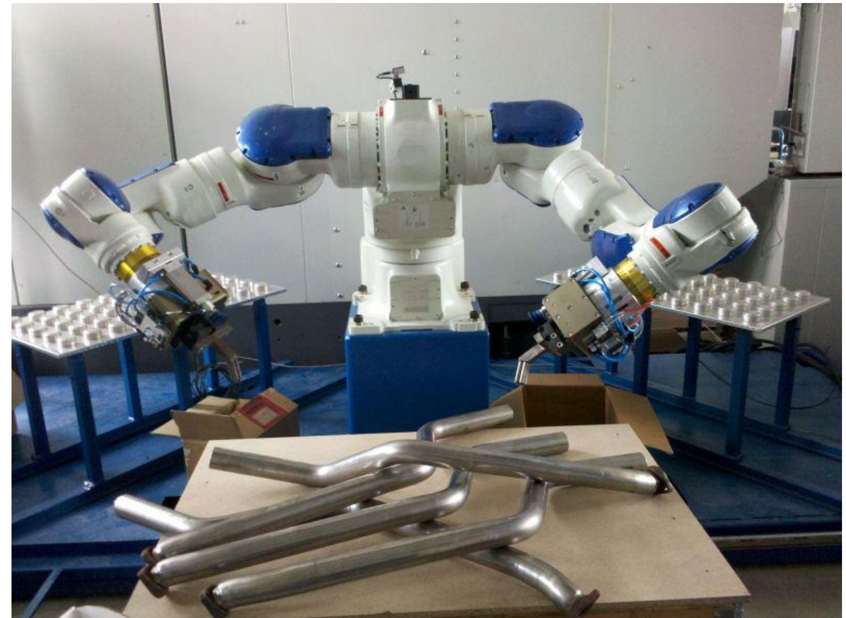
19.06.2013

LEADING
INNOVATIONS

WWW.PROFACTOR.AT

Overview

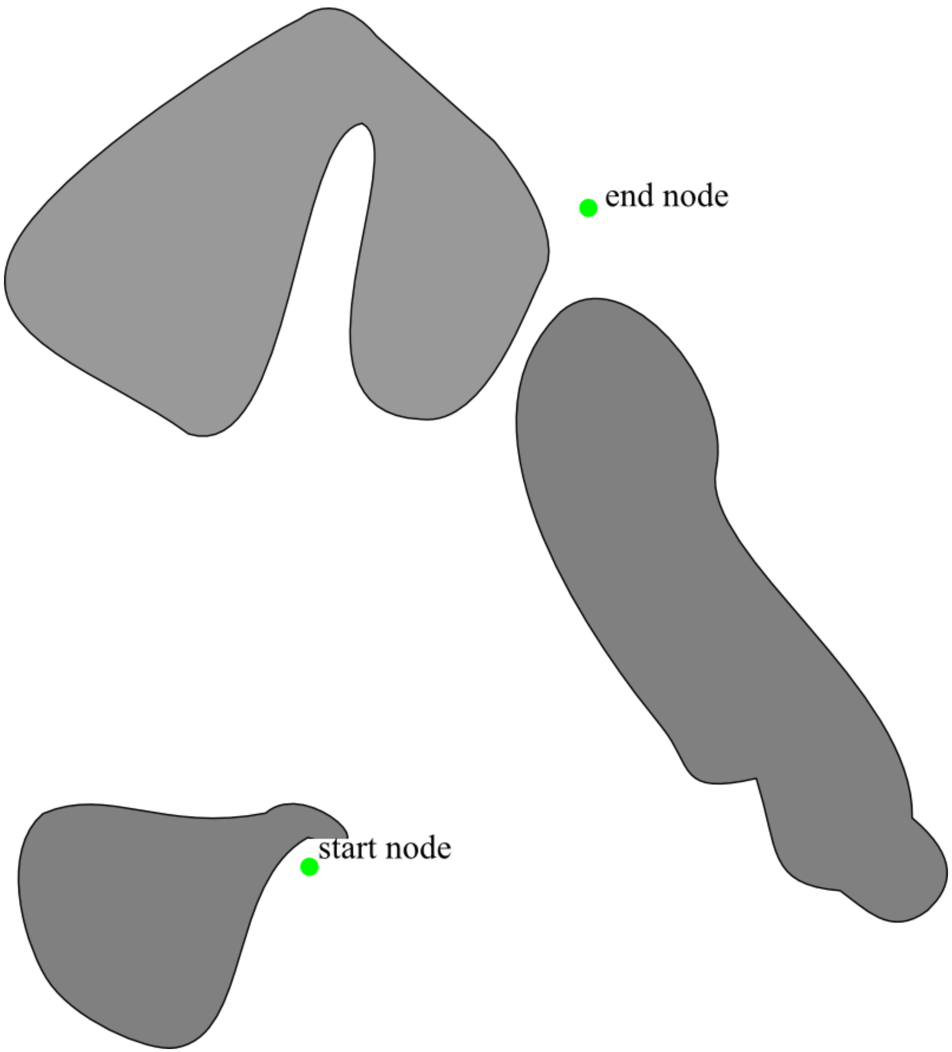
- Introduction on binpicking and planning
- Description of the RRT Planner
- Description of the RRT binpicking planner
- Improvements on RRTB: binary collision check, collinear path purging
- Videos and conclusions



Introduction to Binpicking and Planning

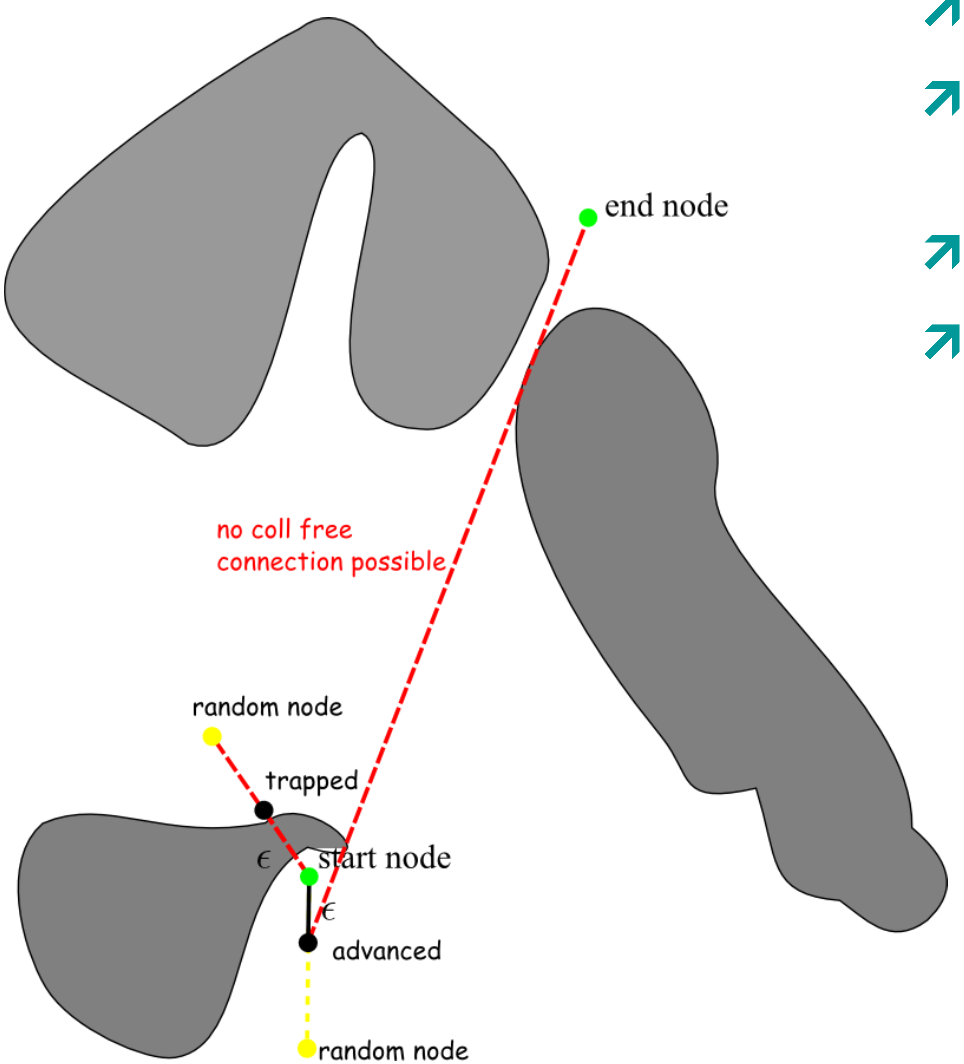
- Given a robot and a container containing (even randomly placed) workpieces, the robot should perform the following actions:
  - travel collision-free with an empty DAF from a start-point to a grip-point of a selected workpiece in the container
  - grip the workpiece in the container
  - travel collision-free with the workpiece (now attached to its DAF) to a certain goal-point and release the workpiece at the goal-point.
  - repeat the last three steps until either the container is empty or a collision-free path by the latter steps are not possible anymore.

RRT Planner



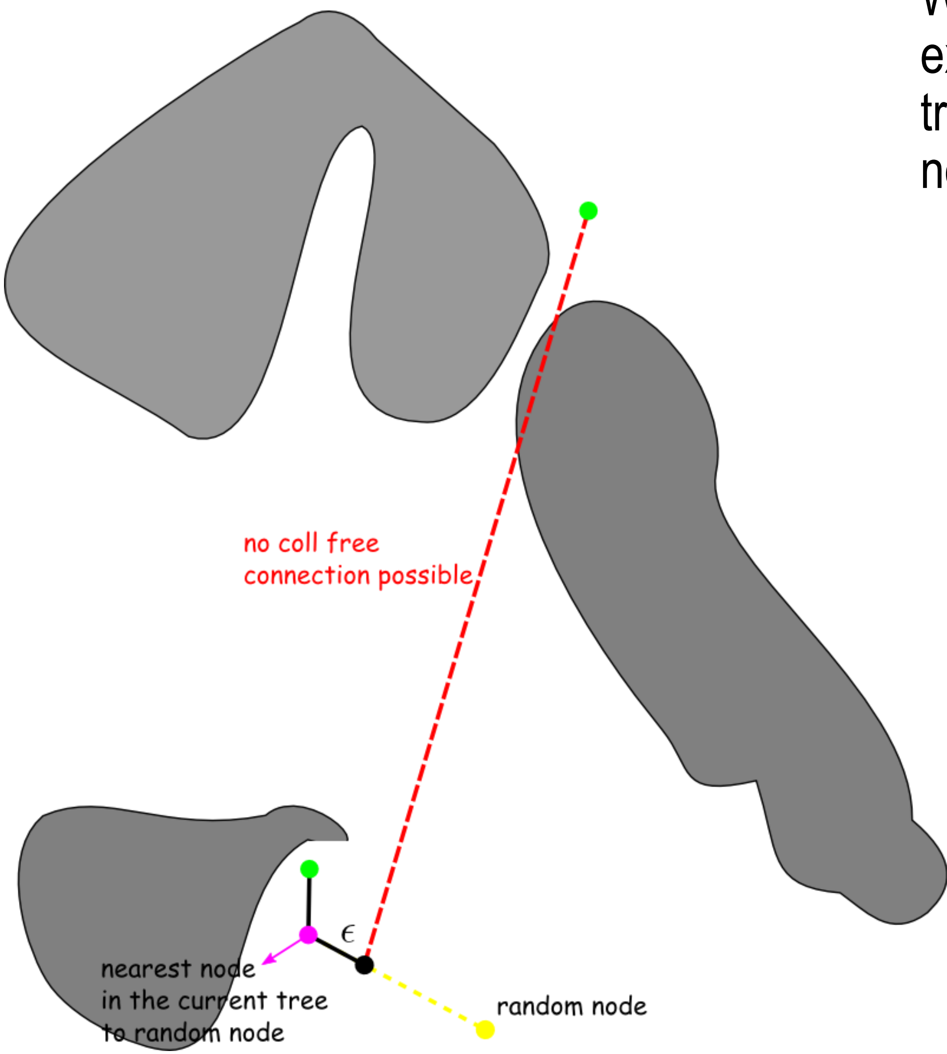
The initial setup as viewed in the configuration space of the robot.

RRT Planner



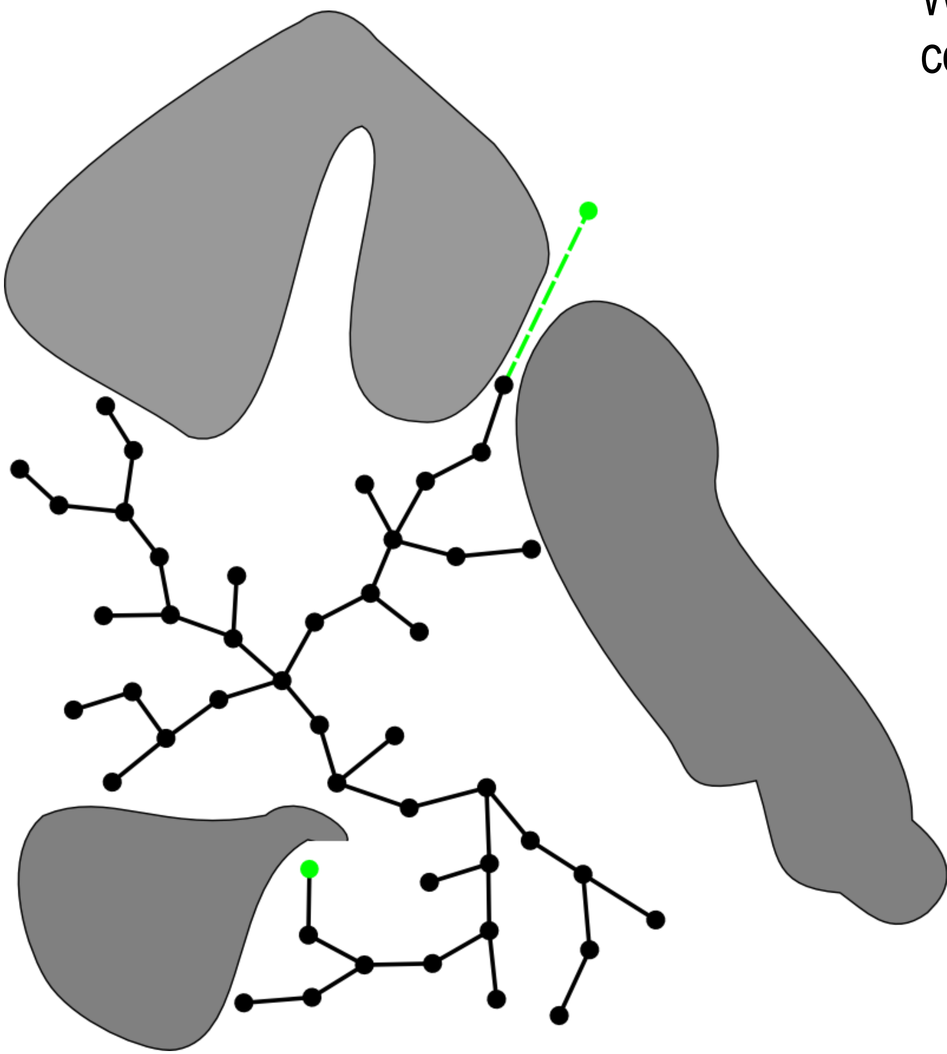
- probe random nodes in configuration space
- extend start node by a length ε (the extend length).
- if extend is colliding retry the above
- if extend not colliding path between last node and goal node and add to tree.

RRT Planner



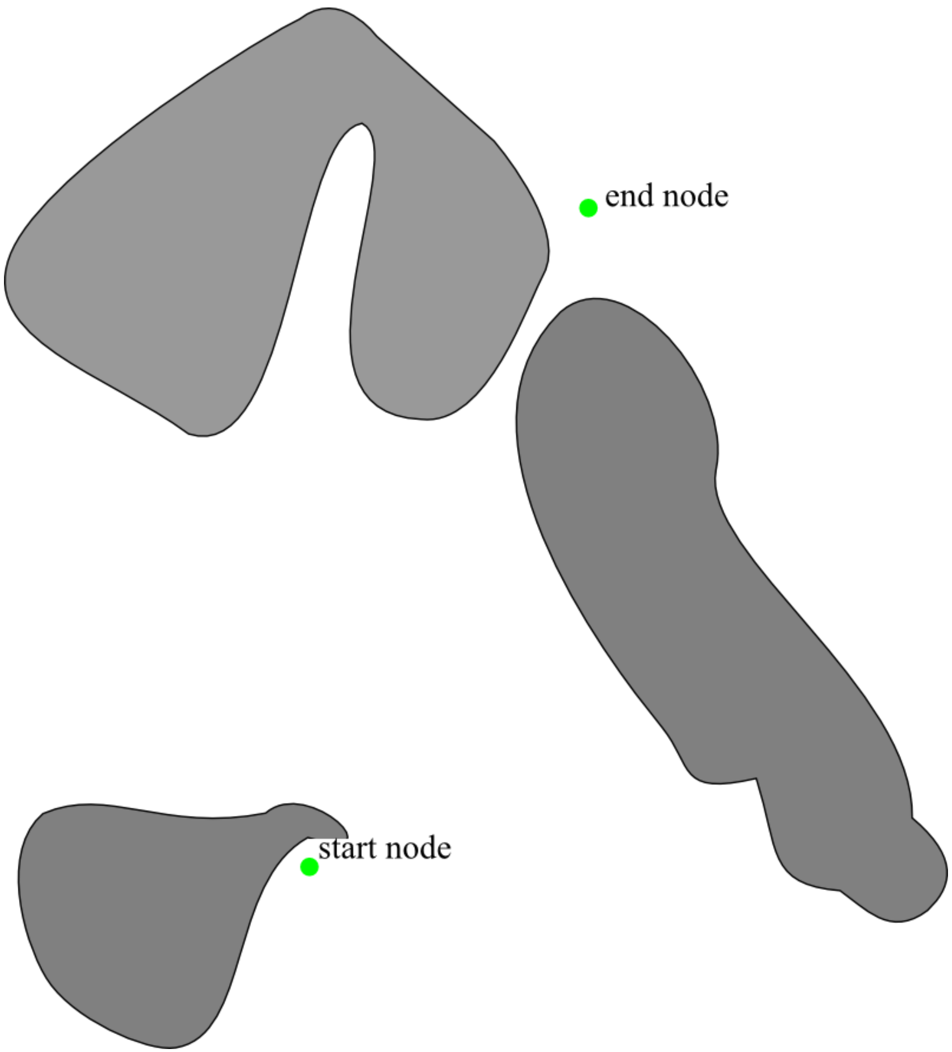
We continue the same procedure as in the first step, except that we chose the nearest node of the current tree to the random node to connect to the random node.

RRT Planner



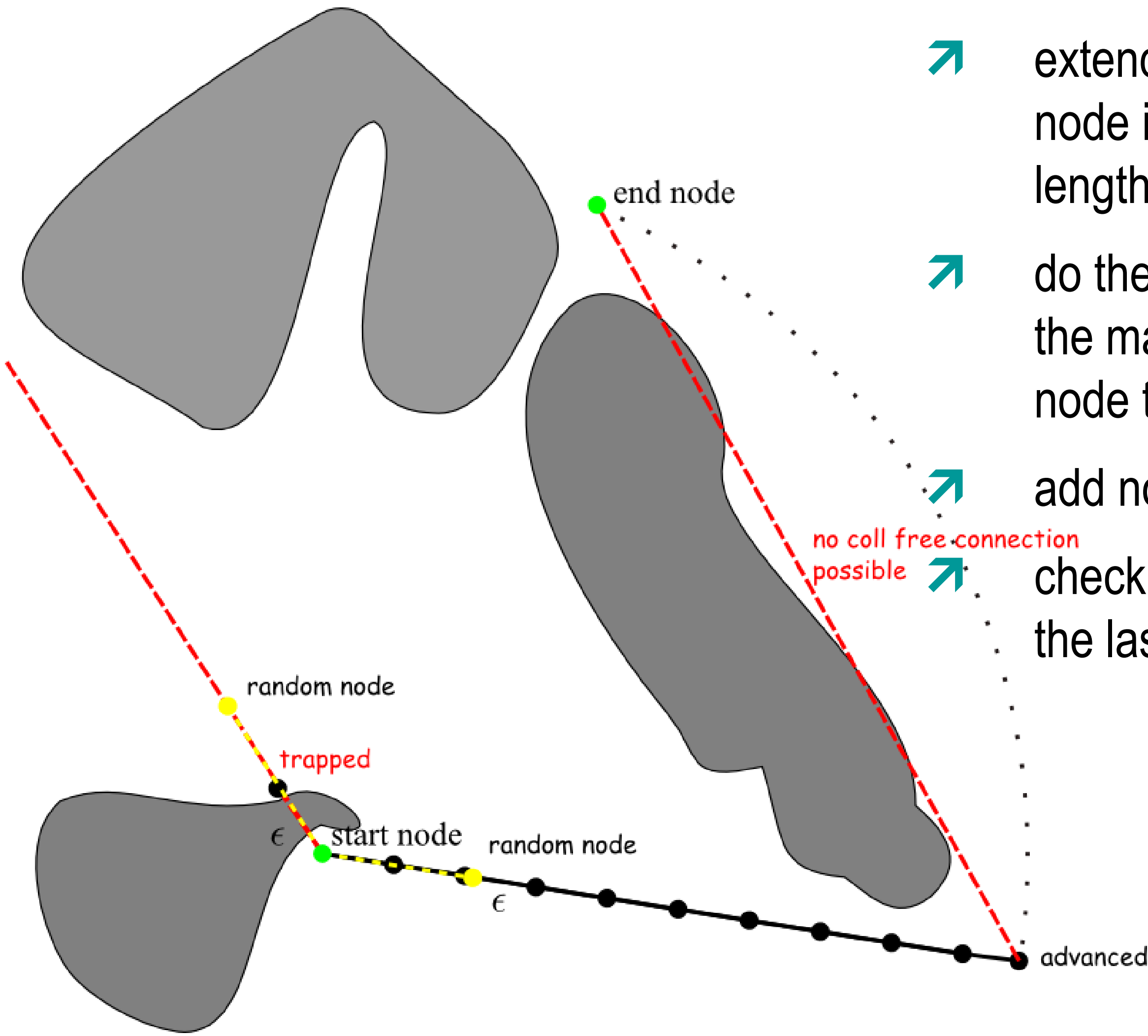
We reach a point wherein the last node of the tree connects to the goal node without collision.

RRT Binpicking



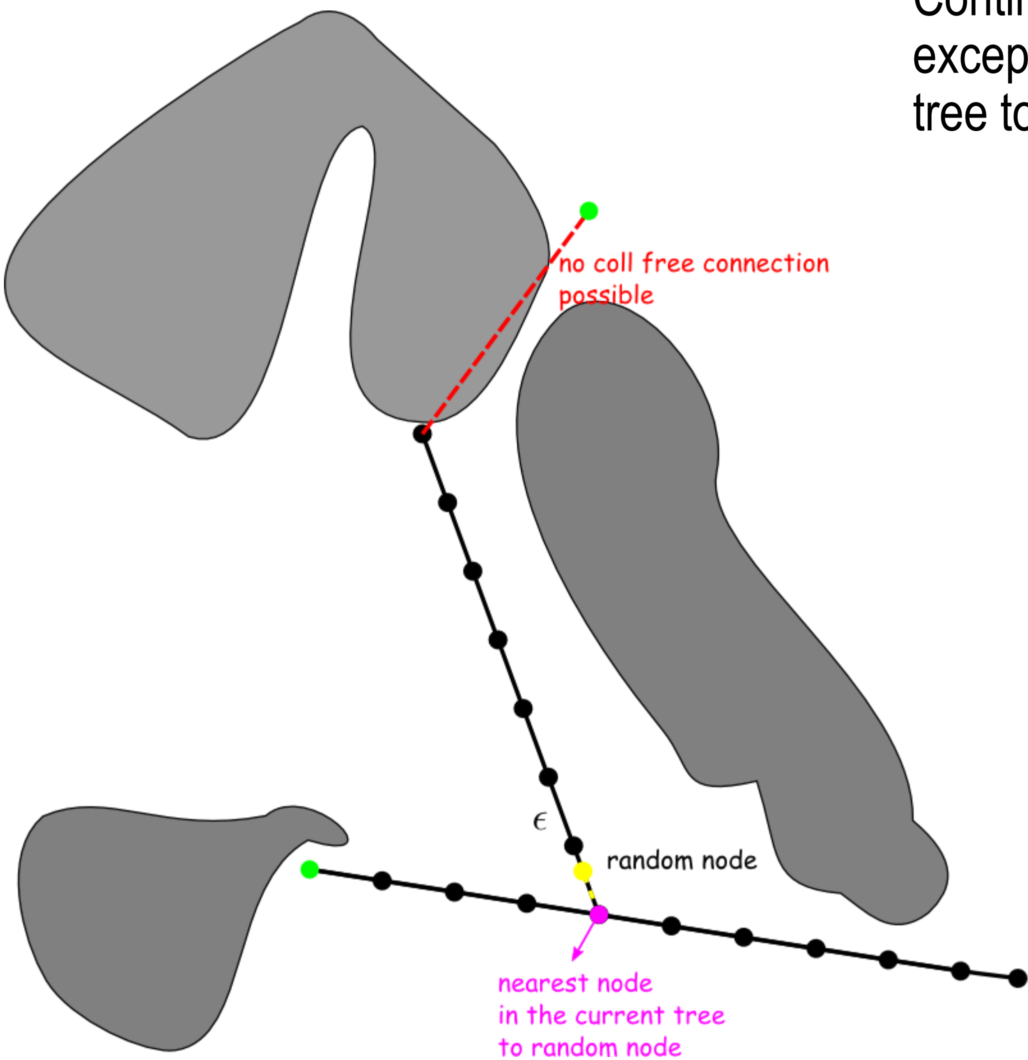
The initial setup as viewed in the configuration space of the robot.

RRT Binpicking



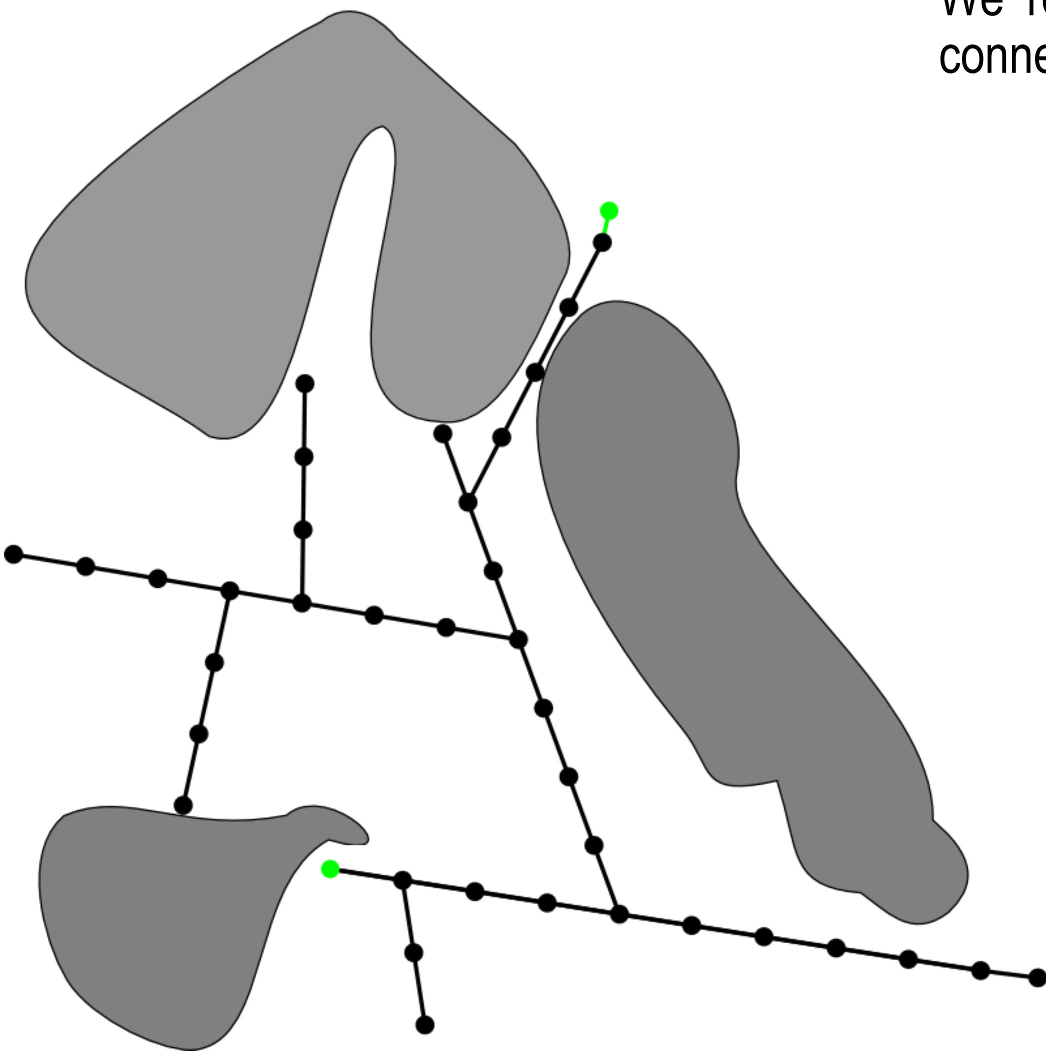
- probe random nodes in configuration space
- extend start node to the direction of the random node iteratively by a length ε (minimum extend length).
- do the above until either colliding or we reached the maximum extend length (length from start node to goal node).
- add nodes and extension to tree.
- check if non-colliding direct path exists between the last node of the tree and the goal node.

RRT Binpicking



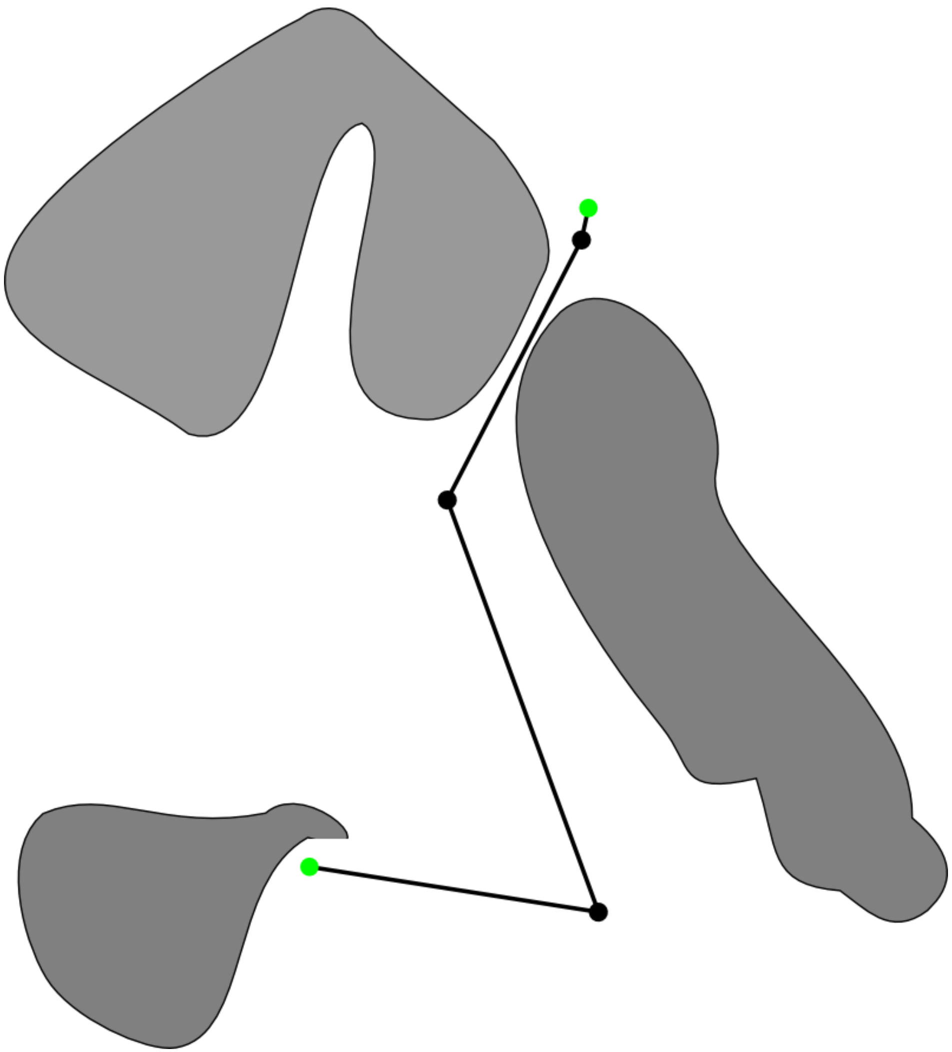
Continue the same procedure as in the first step, except that we extend from the nearest node of the tree to the random node.

RRT Binpicking



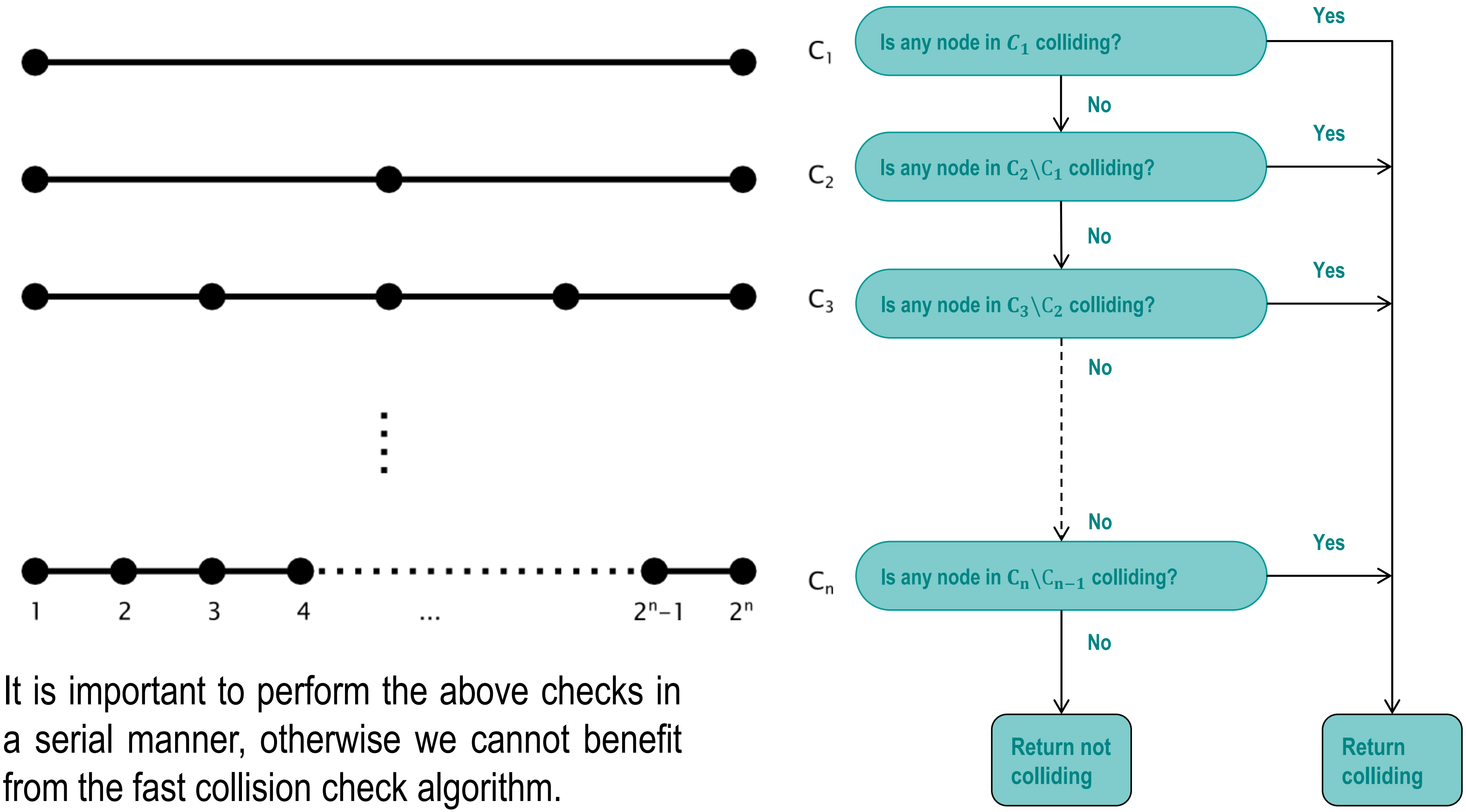
We reach a point wherein the last node of the tree connects to the goal node without collision.

RRT Binpicking



In binpicking applications, final path with collinear purging has usually less nodes than RRT paths that are almost optimized.

Binary Path Collision Check for path with 2^n nodes



It is important to perform the above checks in a serial manner, otherwise we cannot benefit from the fast collision check algorithm.

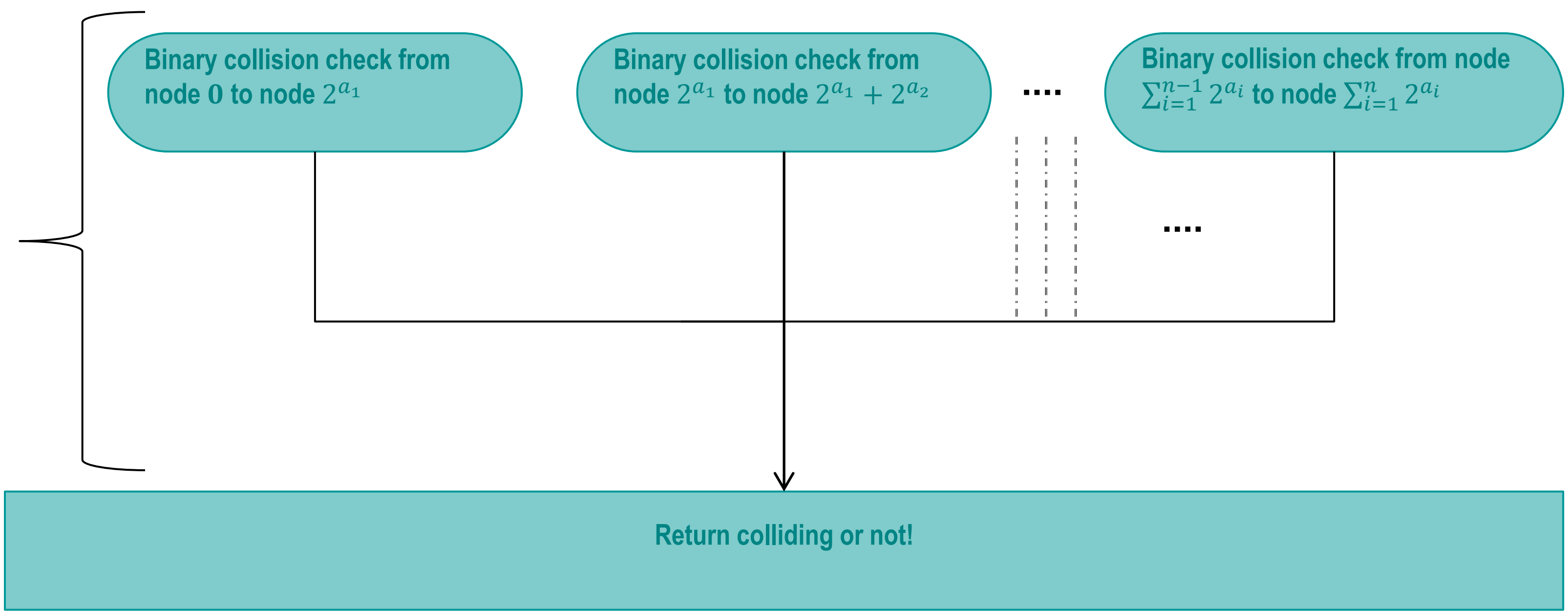
Binary Path Collision Check for path with k nodes

Of course *k>0*. We first write *k* using its 2-adic representation:

$k = 2^{a_1} + 2^{a_2} + \cdots + 2^{a_n}$ where $0 \leq a_1 < a_2 < \cdots < a_n$

Order the nodes in the path (linearly) from *1* to *k*.

Parallel binary collision checks for paths with power of 2 number of nodes. First process to return colliding breaks the parallel process.



Conclusion

- If the grip-points of most workpieces in the bin satisfy bottleneck-condition, RRTB can perform up to 200% better (2x faster cycle times) than RRT.
- If the above condition is not satisfied RRT performs around 25% faster than RRTB
- To decide which planner to choose is still something that experiments can prove. Though, in most of our experimental cases RRTB planners has better or good performance compared to other planners.
- From our experiments, binary path collision check improves the speed of (linear) collision check up to 6-times. This is more prefered than linear discrete path collision checks.