COEN 5830, Fall 2024
Introduction to Robotics

Lecture 10
RRT and RRT*

Srikrishna Bangalore Raghu (srba2850@Colorado.edu)
Thursday, 9/26/2024

Sampling-Based Motion Planning

- Sampling-based planning is a popular graph-based approach used to generate robot motions by sampling discrete states and establishing connections between them via edges

- Their popularity is due to their simplicity and ability to rapidly explore high-dimensional spaces.

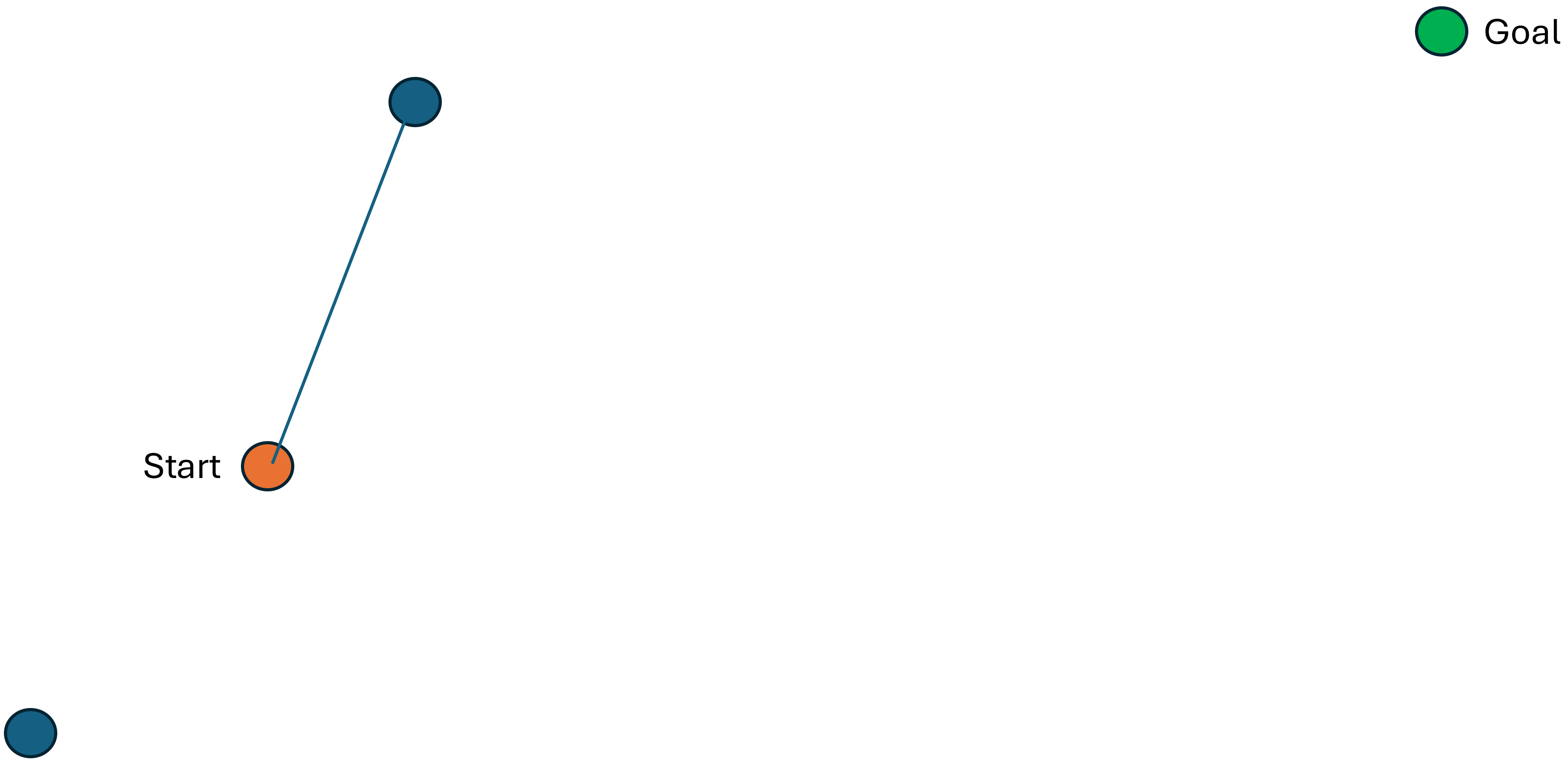
- Traditionally, these techniques employ a unidirectional tree that grows from the start state and expands towards the goal region

Rapidly Exploring Random Tree (RRT)

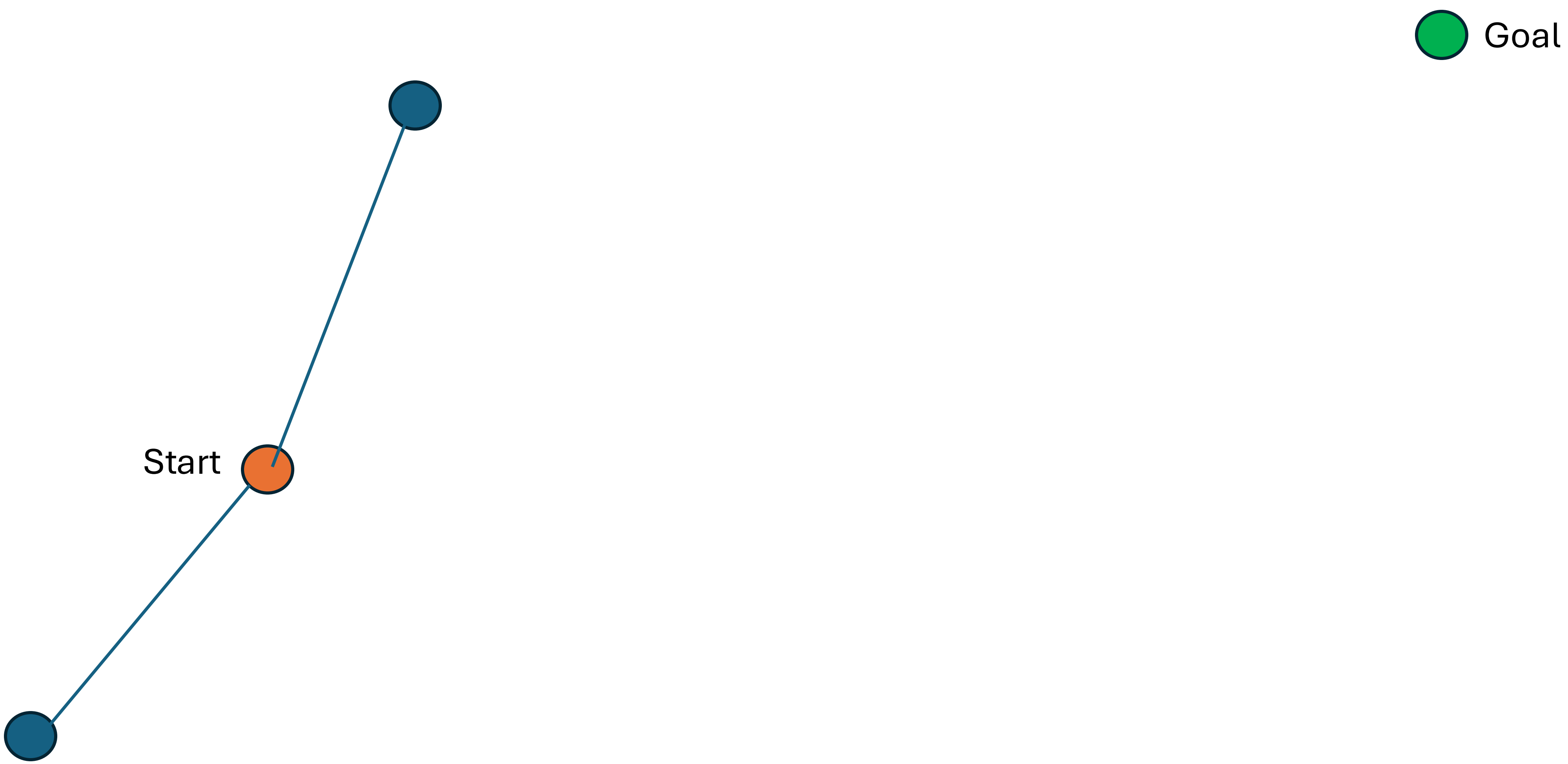
Rapidly Exploring Random Tree (RRT)

Rapidly Exploring Random Tree (RRT)

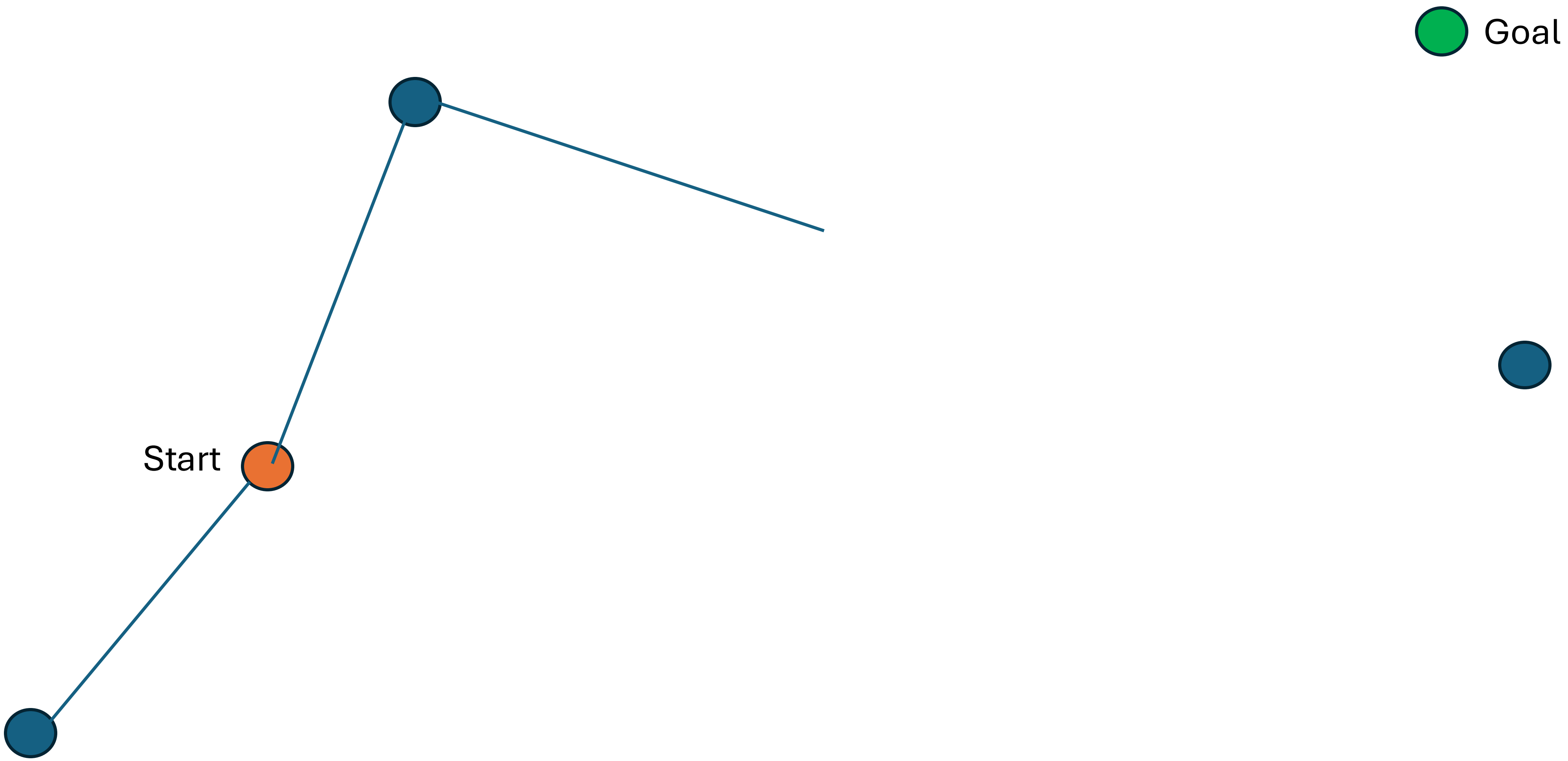
Rapidly Exploring Random Tree (RRT)



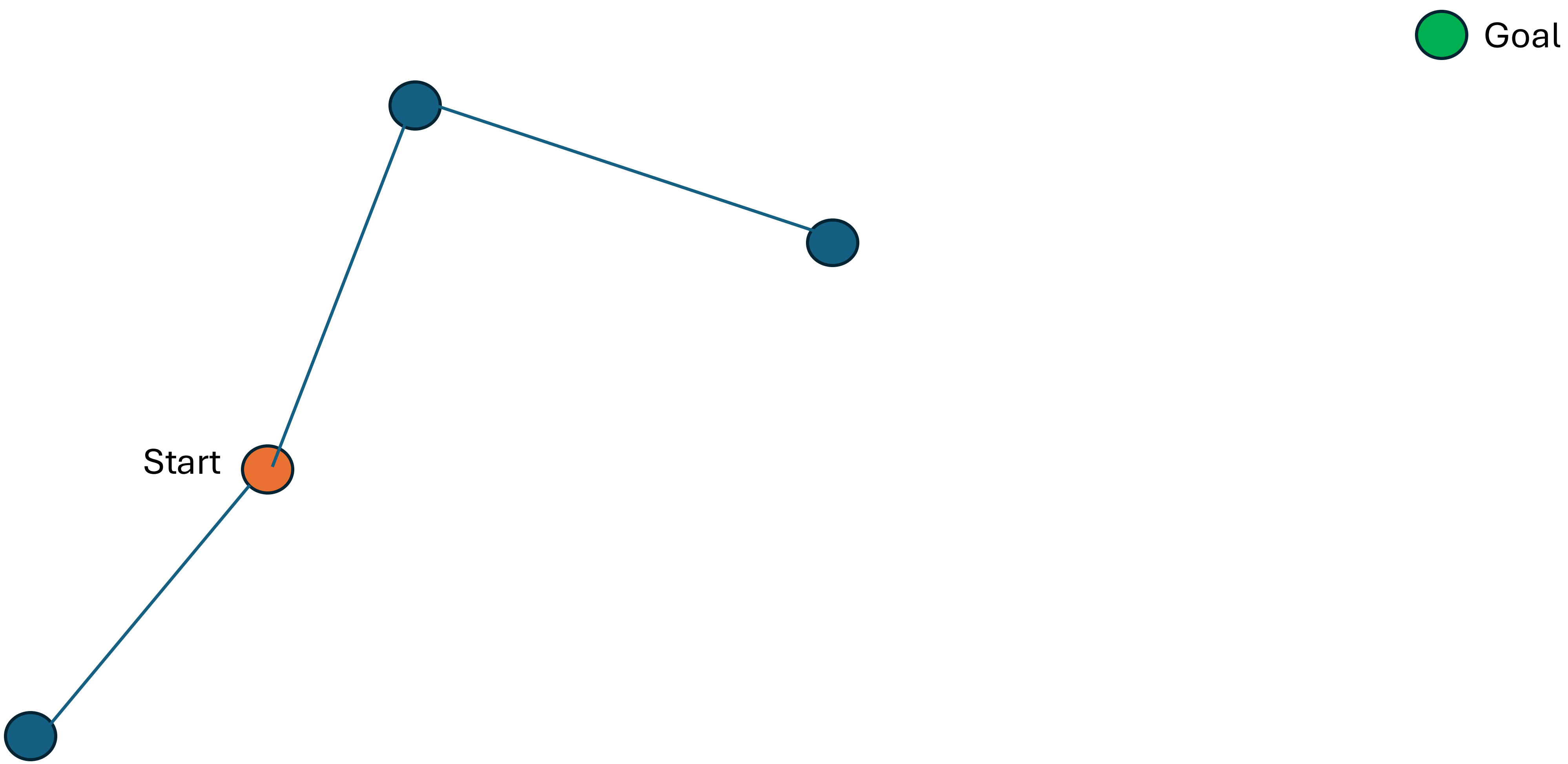
Rapidly Exploring Random Tree (RRT)



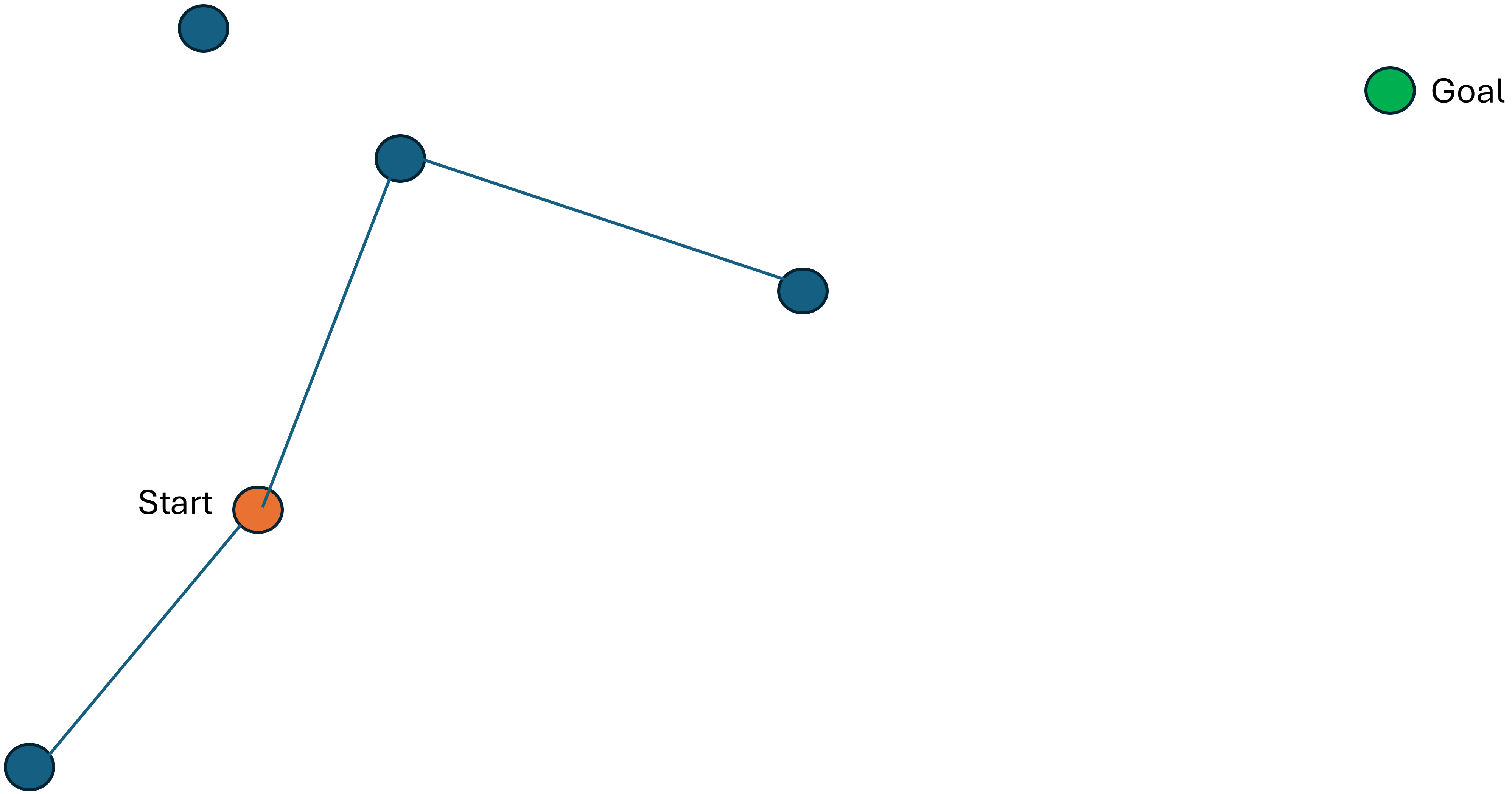
Rapidly Exploring Random Tree (RRT)



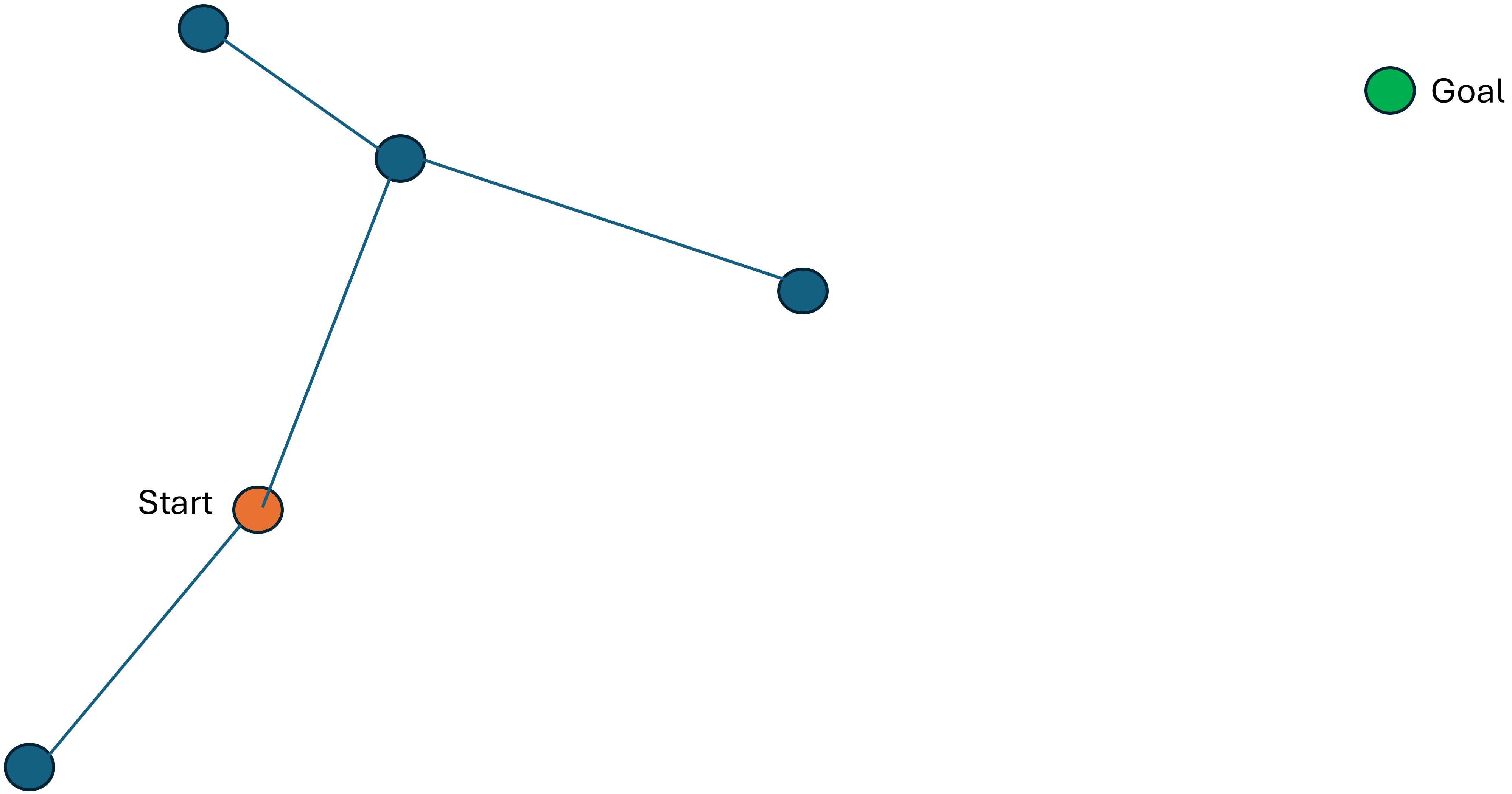
Rapidly Exploring Random Tree (RRT)



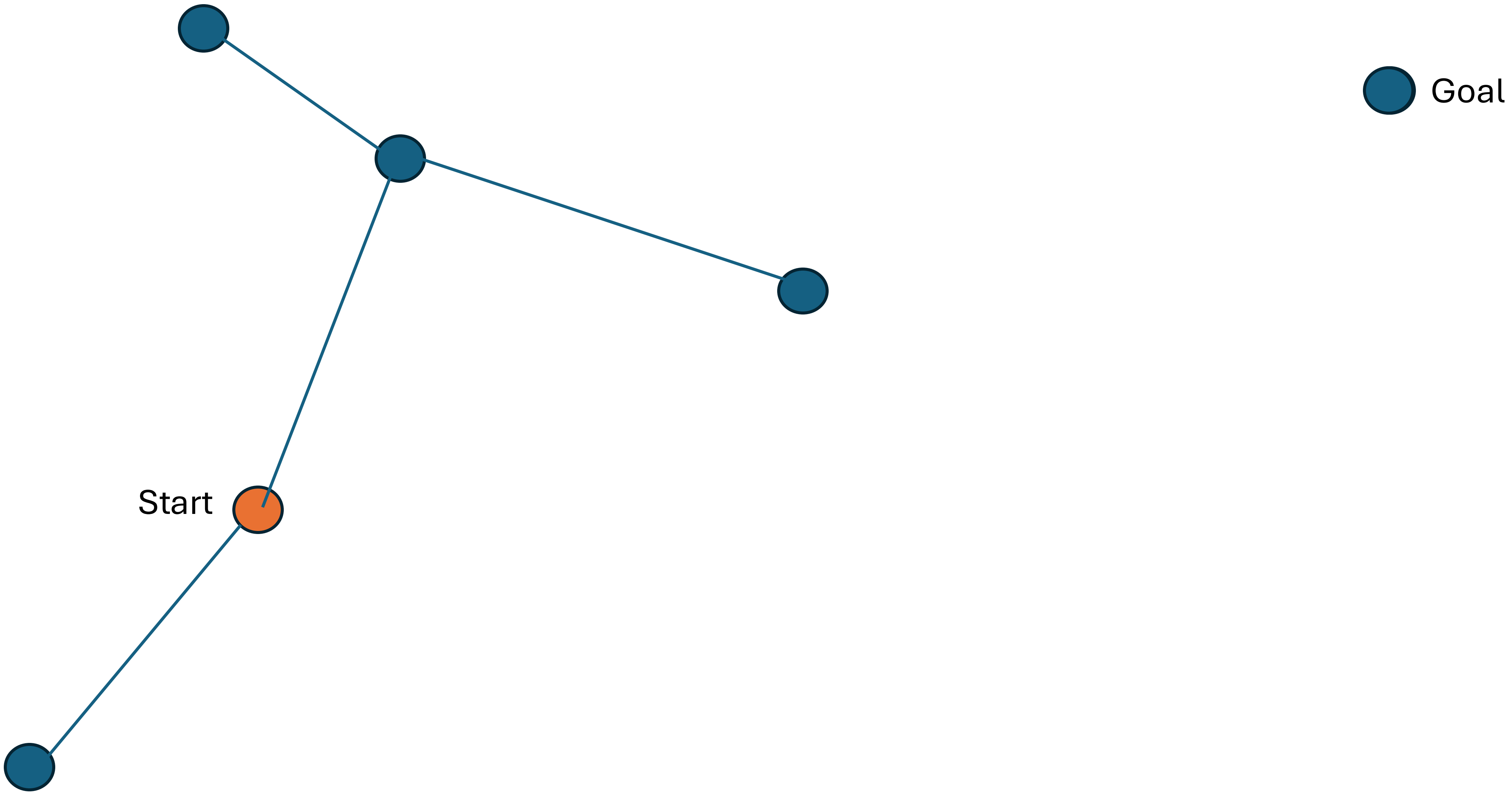
Rapidly Exploring Random Tree (RRT)



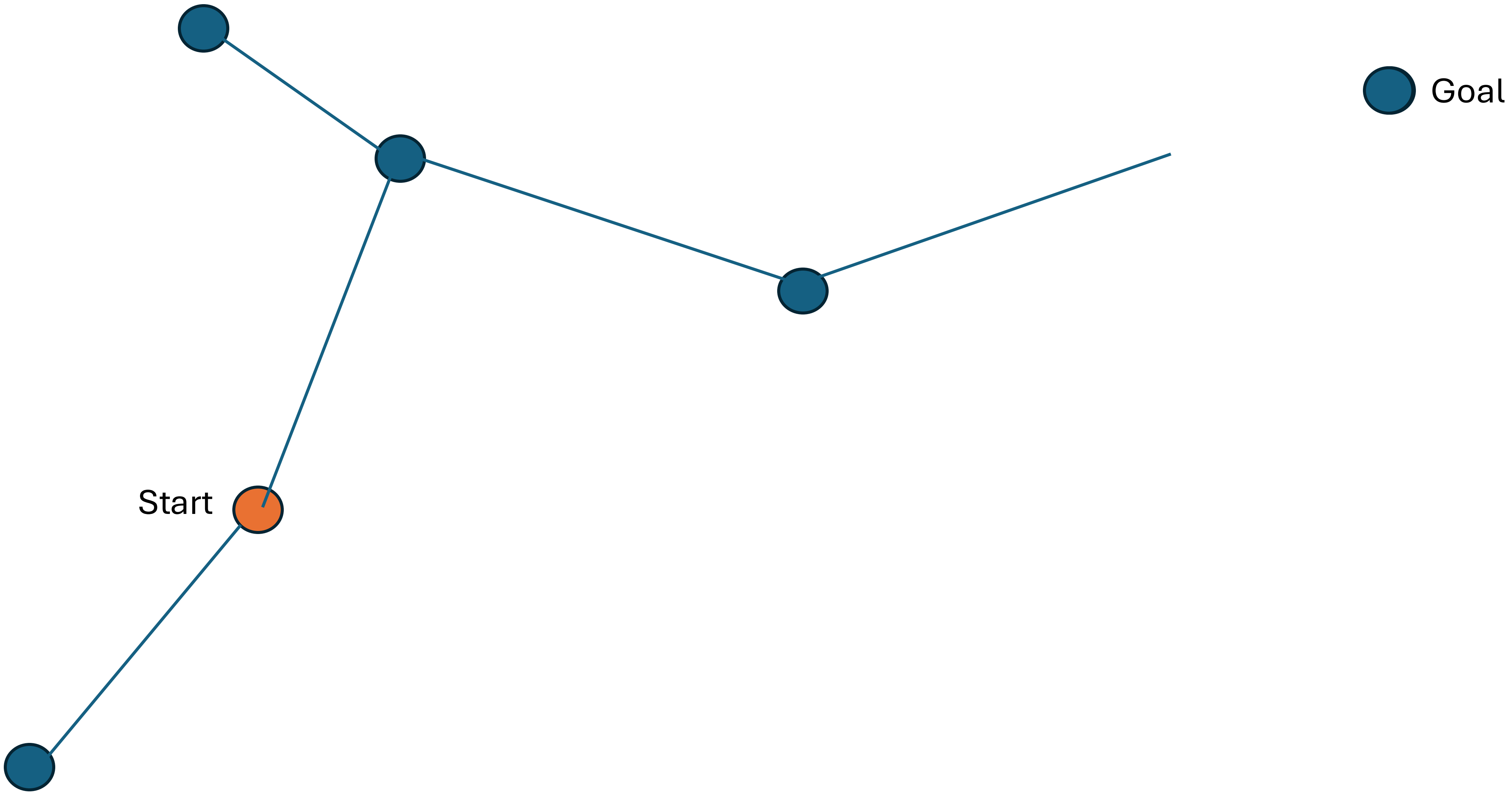
Rapidly Exploring Random Tree (RRT)



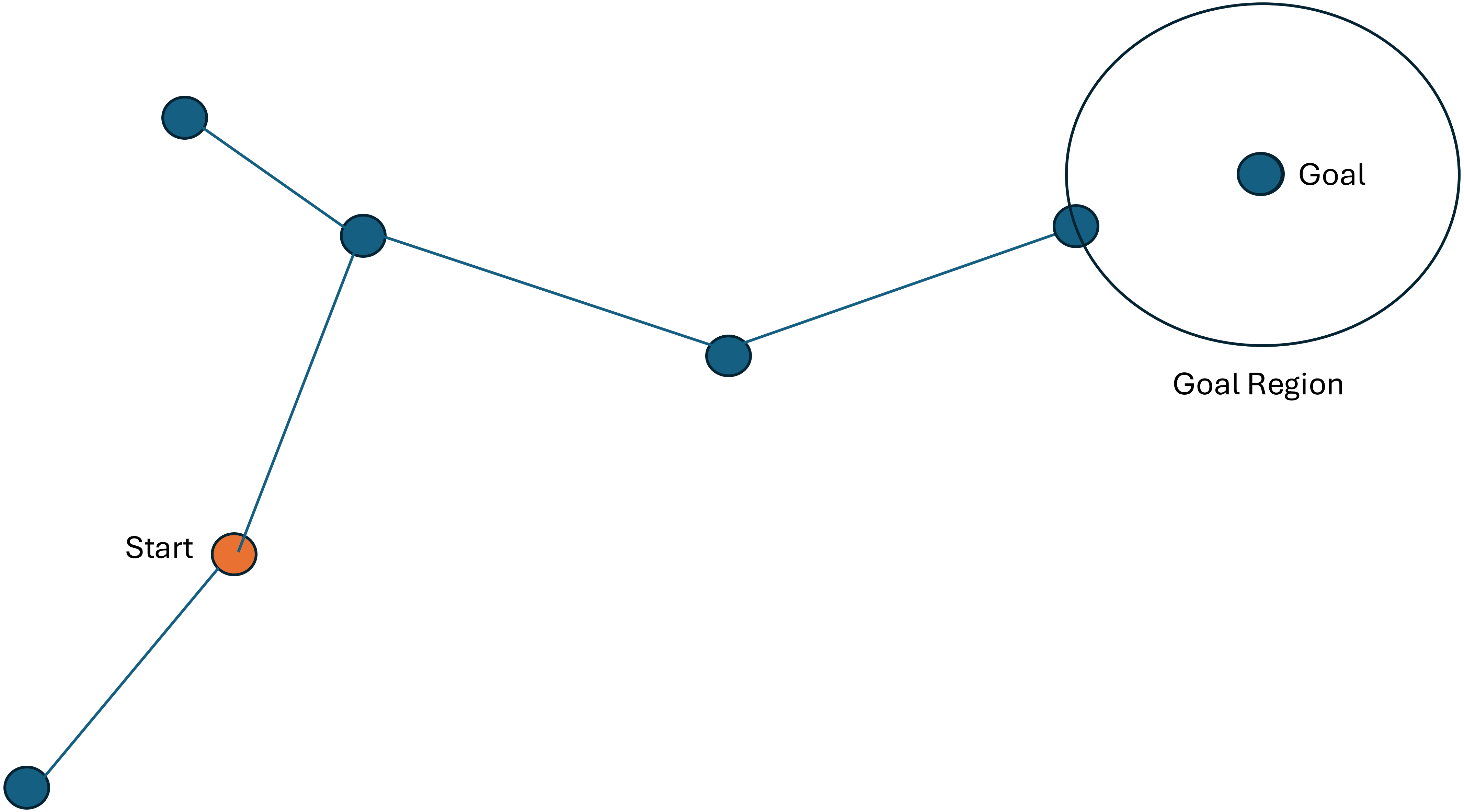
Rapidly Exploring Random Tree (RRT)



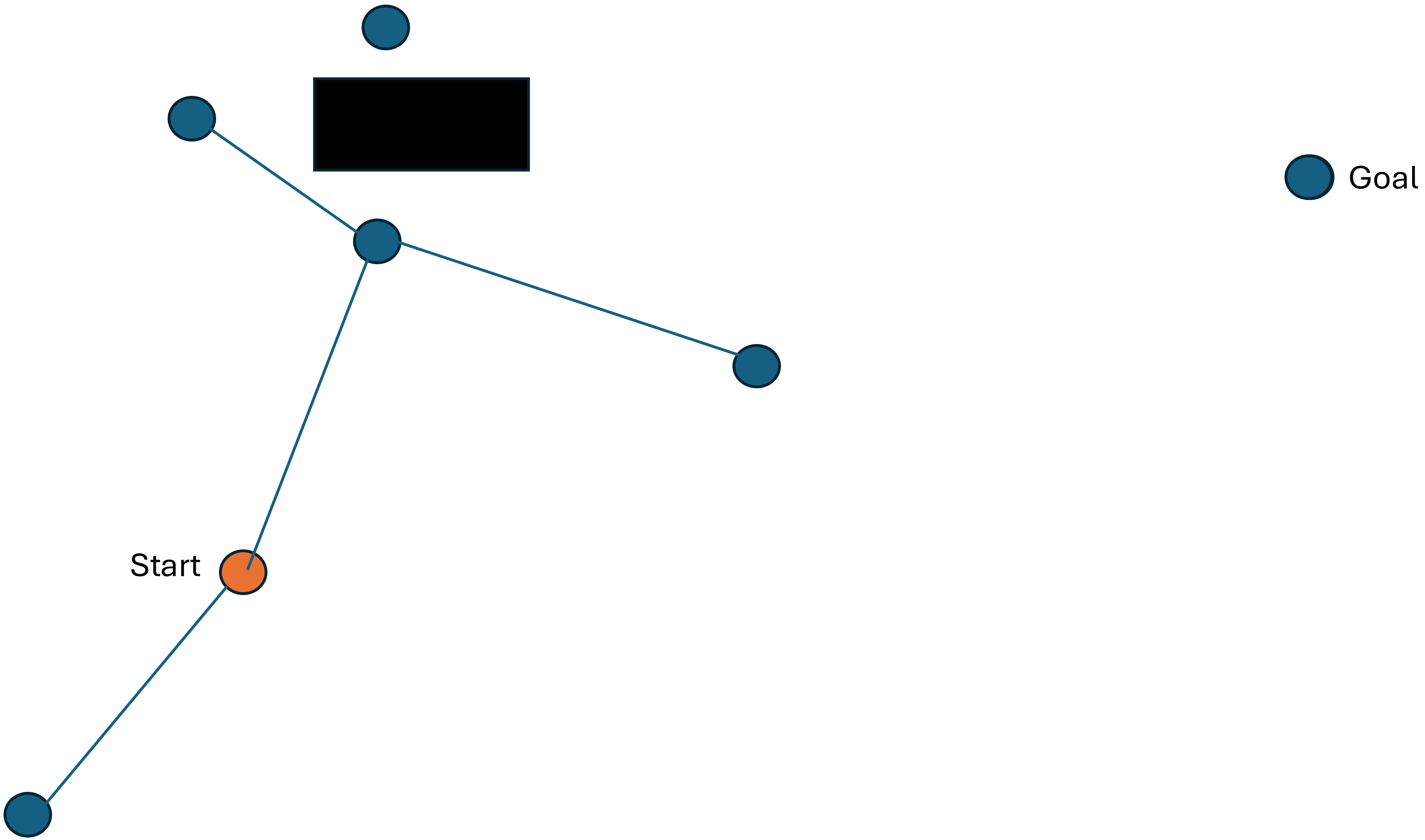
Rapidly Exploring Random Tree (RRT)



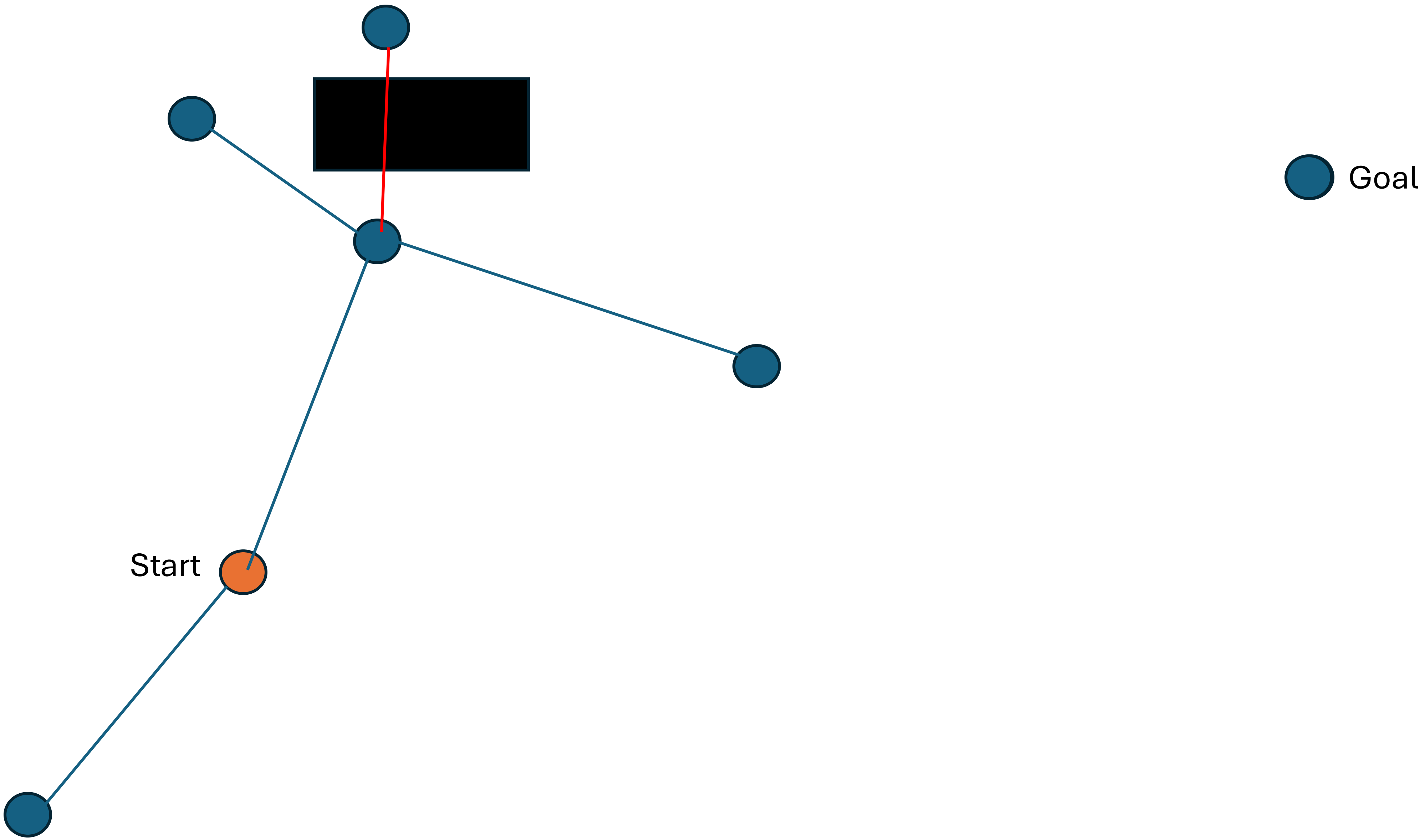
Rapidly Exploring Random Tree (RRT)



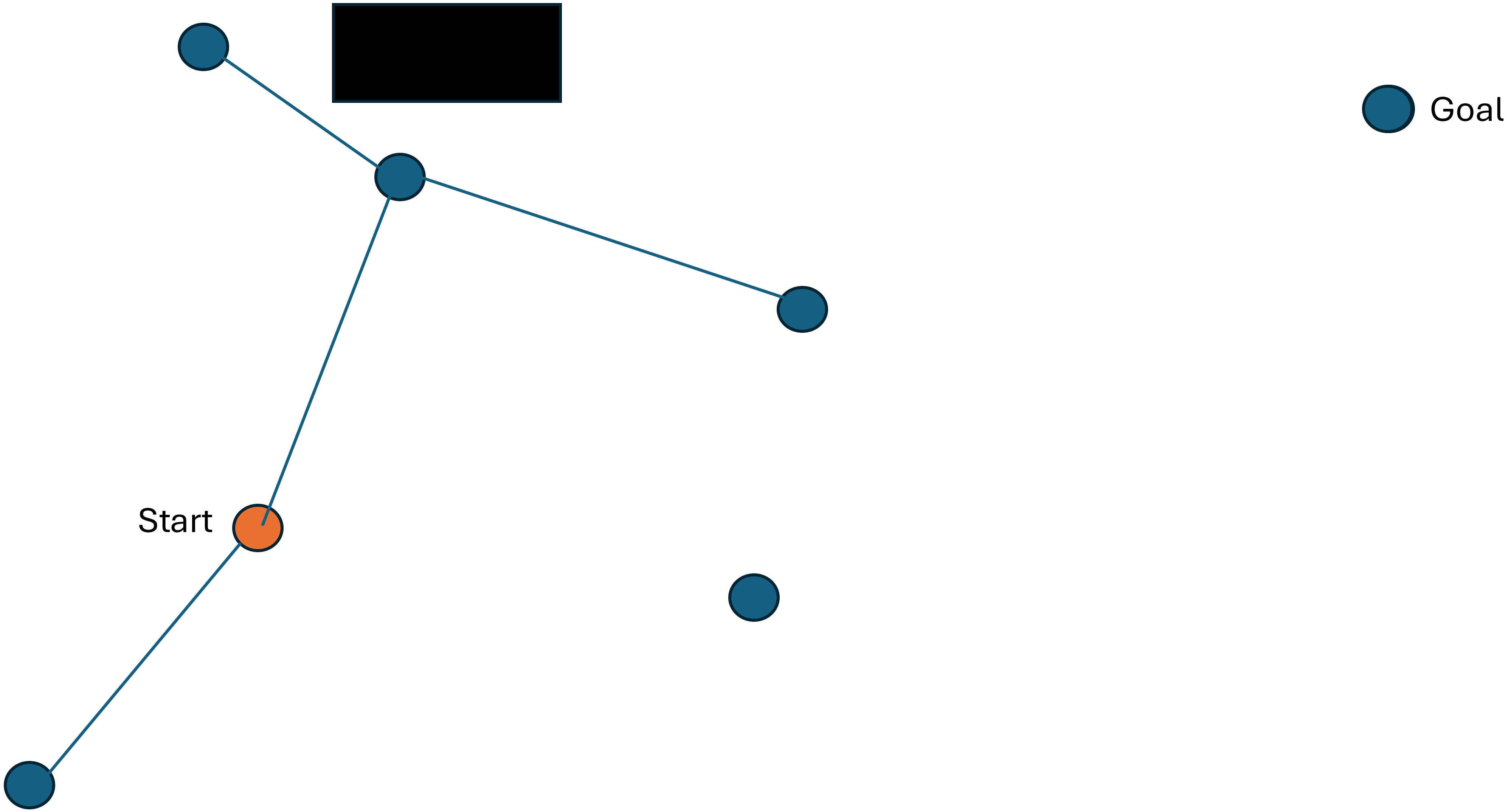
Rapidly Exploring Random Tree (RRT)



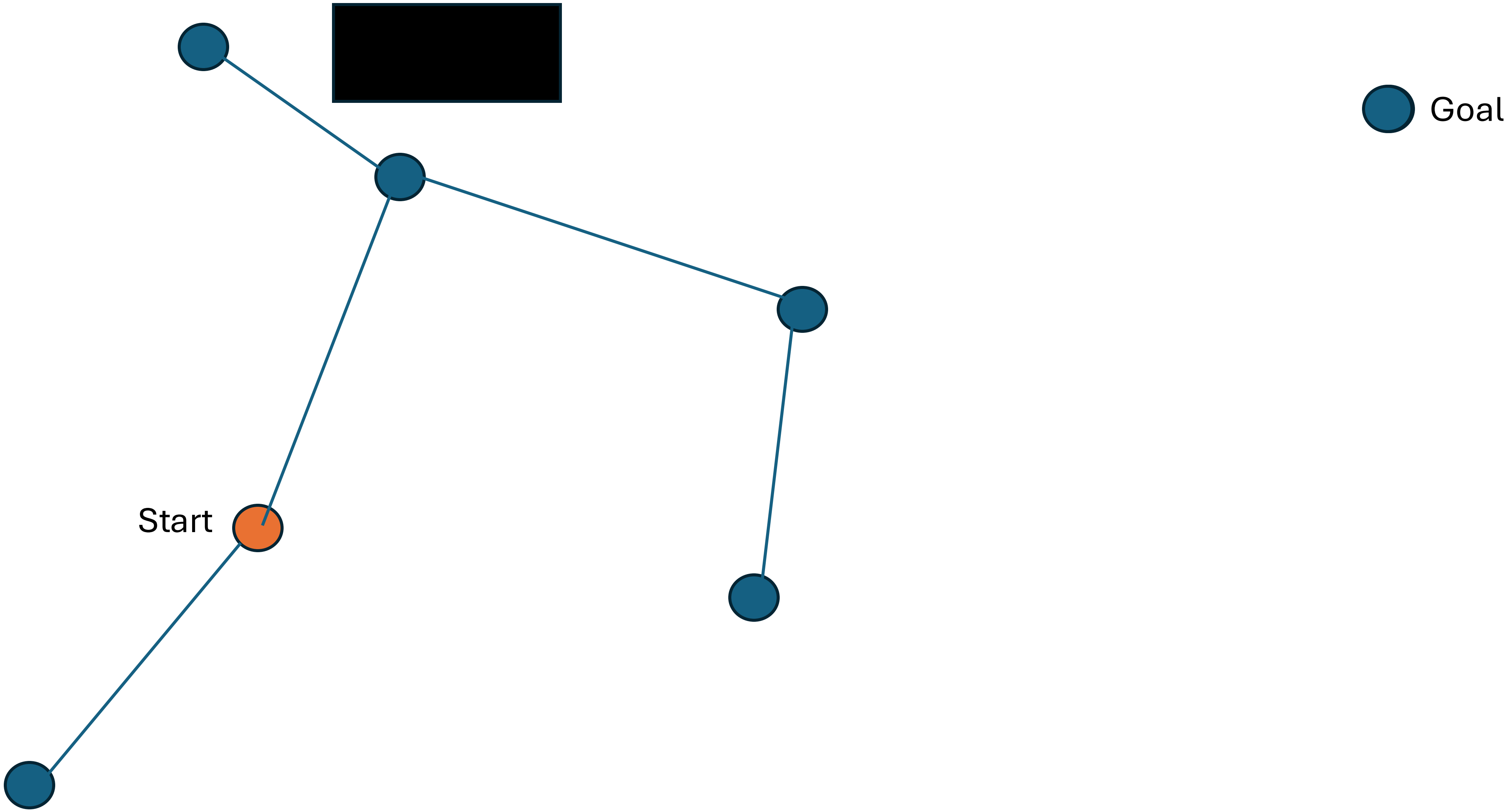
Rapidly Exploring Random Tree (RRT)



Rapidly Exploring Random Tree (RRT)



Rapidly Exploring Random Tree (RRT)



RRT

Algorithm : RRT

$V \leftarrow \{x_{\text{init}}\}$; $E \leftarrow \emptyset$;

for $i = 1, \ldots, n$ **do**

  $x_{\text{rand}} \leftarrow \texttt{SampleFree}_i$;

  $x_{\text{nearest}} \leftarrow \texttt{Nearest}(G = (V, E), x_{\text{rand}})$;

  $x_{\text{new}} \leftarrow \texttt{Steer}(x_{\text{nearest}}, x_{\text{rand}})$;

  if $\texttt{ObtacleFree}(x_{\text{nearest}}, x_{\text{new}})$ **then**

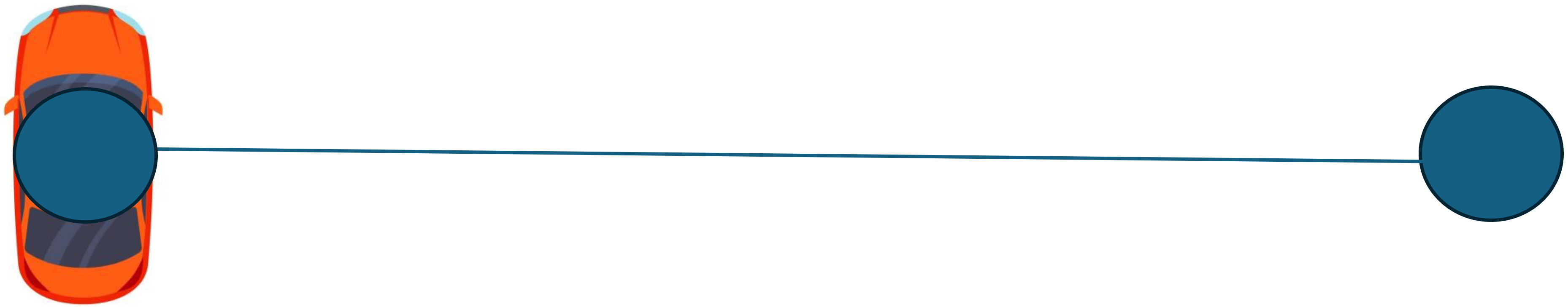
    $V \leftarrow V \cup \{x_{\text{new}}\}$; $E \leftarrow E \cup \{(x_{\text{nearest}}, x_{\text{new}})\}$;

return $G = (V, E)$;

Steering Function

Steering Function

Steering Function



RRT*

Algorithm 6: RRT*

$V \leftarrow \{x_{\text{init}}\}$; $E \leftarrow \emptyset$;
for $i = 1, \dots, n$ **do**
  $x_{\text{rand}} \leftarrow \texttt{SampleFree}_i$;
  $x_{\text{nearest}} \leftarrow \texttt{Nearest}(G = (V, E), x_{\text{rand}})$;
  $x_{\text{new}} \leftarrow \texttt{Steer}(x_{\text{nearest}}, x_{\text{rand}})$;
  if $\texttt{ObtacleFree}(x_{\text{nearest}}, x_{\text{new}})$ **then**
    $X_{\text{near}} \leftarrow \texttt{Near}(G = (V, E), x_{\text{new}}, \min\{\gamma_{\text{RRT}^*}(\log(\text{card}\,(V))/\,\text{card}\,(V))^{1/d}, \eta\})$;
    $V \leftarrow V \cup \{x_{\text{new}}\}$;
    $x_{\text{min}} \leftarrow x_{\text{nearest}}$; $c_{\text{min}} \leftarrow \texttt{Cost}(x_{\text{nearest}}) + c(\texttt{Line}(x_{\text{nearest}}, x_{\text{new}}))$;
    foreach $x_{\text{near}} \in X_{\text{near}}$ **do** // Connect along a minimum-cost path
      if $\texttt{CollisionFree}(x_{\text{near}}, x_{\text{new}}) \wedge \texttt{Cost}(x_{\text{near}}) + c(\texttt{Line}(x_{\text{near}}, x_{\text{new}})) < c_{\text{min}}$ **then**
        $x_{\text{min}} \leftarrow x_{\text{near}}$; $c_{\text{min}} \leftarrow \texttt{Cost}(x_{\text{near}}) + c(\texttt{Line}(x_{\text{near}}, x_{\text{new}}))$
    $E \leftarrow E \cup \{(x_{\text{min}}, x_{\text{new}})\}$;
    foreach $x_{\text{near}} \in X_{\text{near}}$ **do** // Rewire the tree
      if $\texttt{CollisionFree}(x_{\text{new}}, x_{\text{near}}) \wedge \texttt{Cost}(x_{\text{new}}) + c(\texttt{Line}(x_{\text{new}}, x_{\text{near}})) < \texttt{Cost}(x_{\text{near}})$
      then $x_{\text{parent}} \leftarrow \texttt{Parent}(x_{\text{near}})$;
      $E \leftarrow (E \setminus \{(x_{\text{parent}}, x_{\text{near}})\}) \cup \{(x_{\text{new}}, x_{\text{near}})\}$
return $G = (V, E)$;