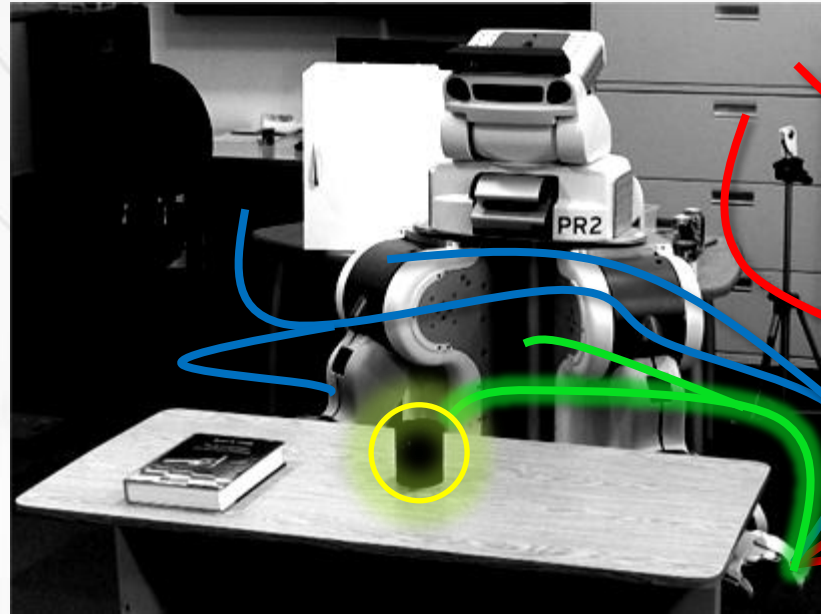


Asymptotically-optimal Manipulation Planning using Incremental Sampling-based Algorithms



Alejandro Perez¹

Sertac Karaman², Alexander Shkolnik¹,

Emilio Frazzoli², Seth Teller¹ and Matt Walter¹

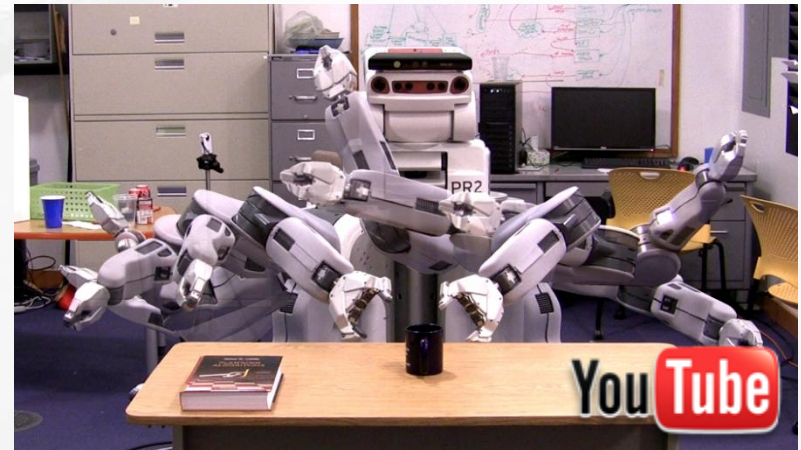
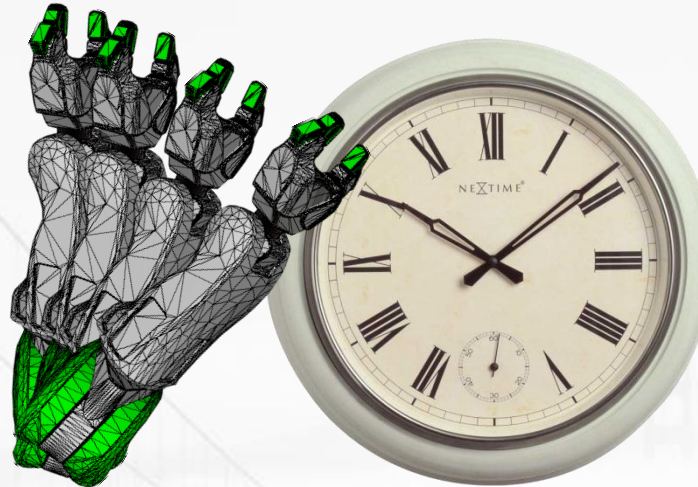
¹ MIT/CSAIL

² MIT/LIDS

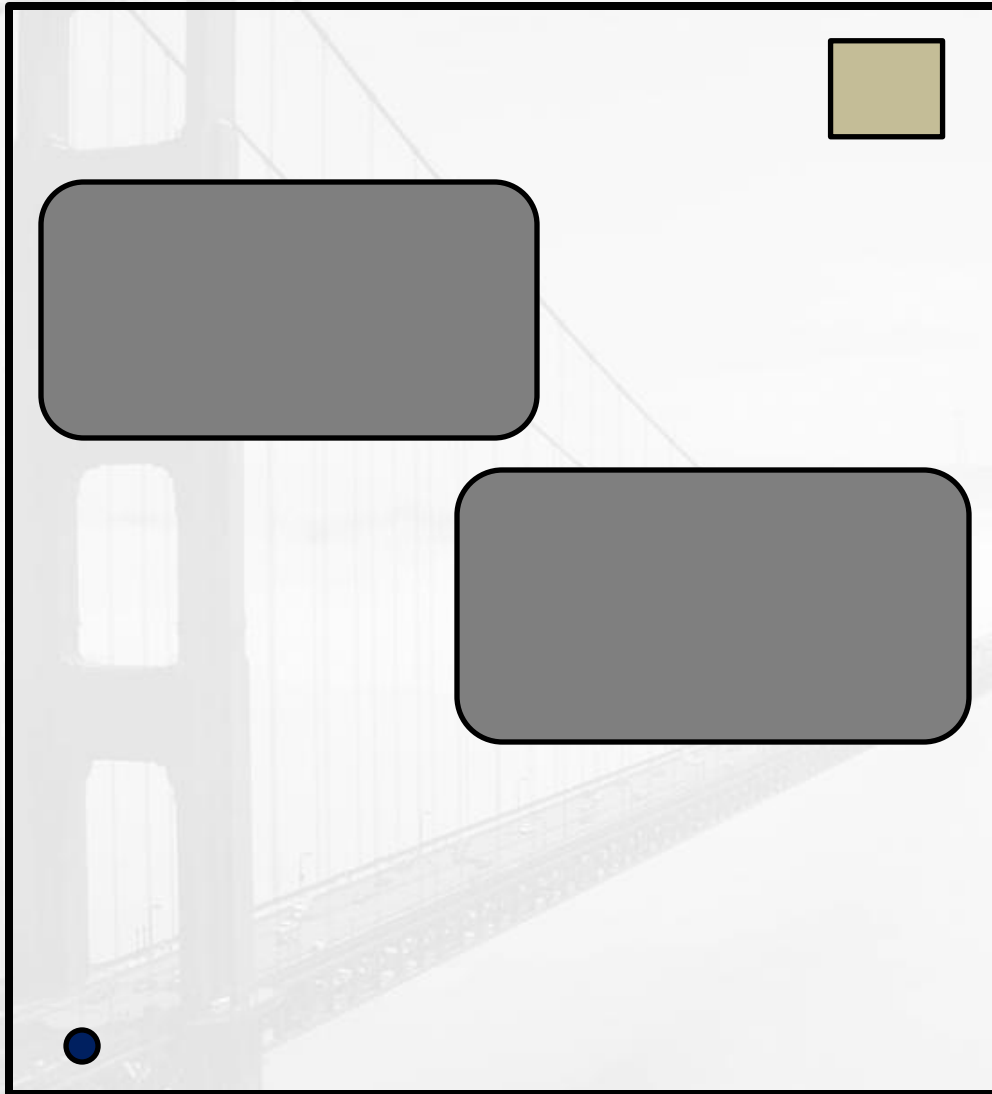
IEEE/RSJ International Conference on Intelligent
Robots and Systems 2011



Manipulation Planning



RRT* with Ball Trees



The RRT* Algorithm

```

1  $V \leftarrow \{x_{init}\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3    $x_{new} \leftarrow \text{Sample}(i);$ 
4    $X_{near} \leftarrow \text{Near}(V, x_{new});$ 
5   if  $X_{near} = \emptyset$  then
6      $X_{near} \leftarrow \text{Nearest}(V, x_{new});$ 
7    $L_{near} \leftarrow \text{PopulateSortedList}(X_{near}, x_{near});$ 
8    $x_{parent} \leftarrow \text{FindBestParent}(L_{near}, x_{new});$ 
9   if  $x_{parent} \neq \text{NULL}$  then
10     $V.\text{add}(x_{new});$ 
11     $E.\text{add}((x_{parent}, x_{new}));$ 
12     $E \leftarrow \text{RewireVertices}(E, X_{near}, x_{new});$ 
13 return  $T = (V, E).$ 

```

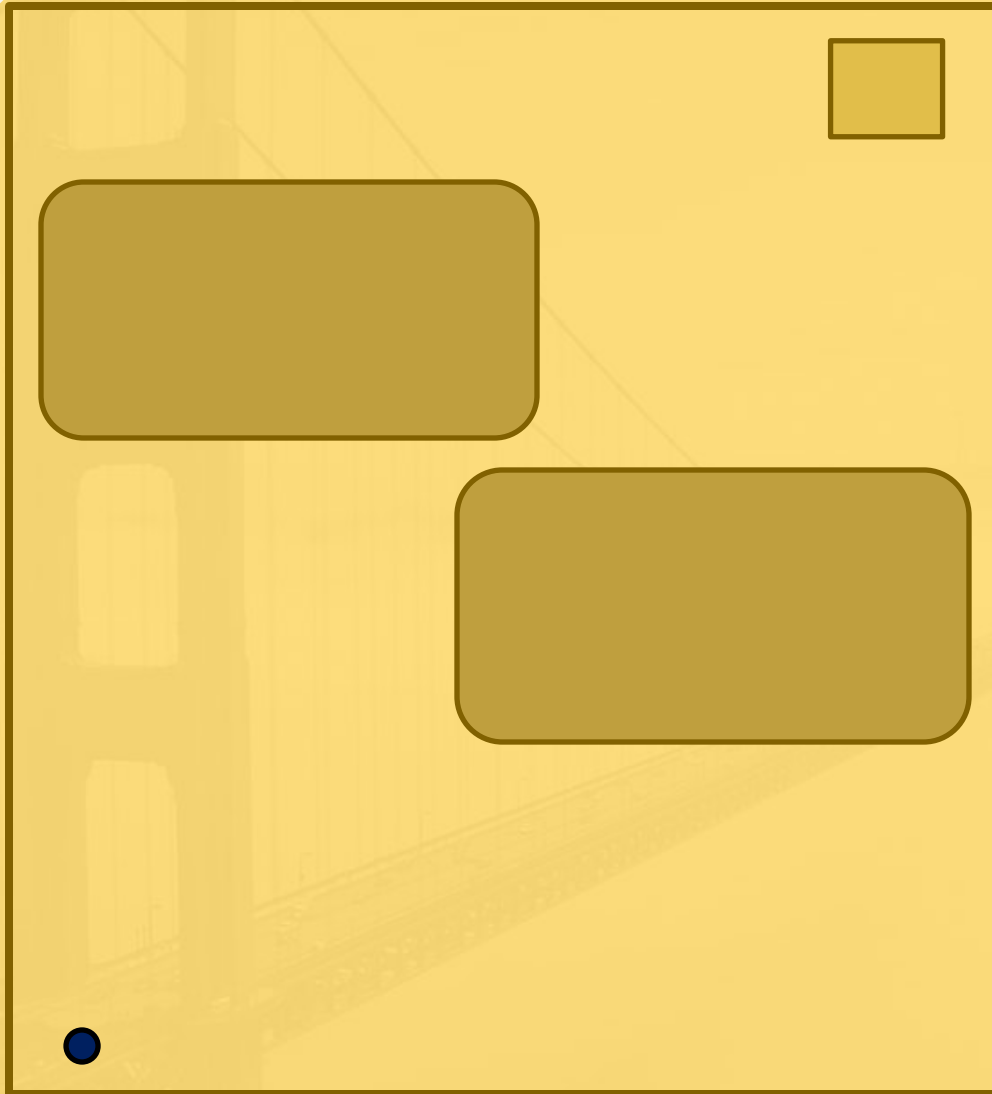
The Ball Tree Algorithm

```

1  $V \leftarrow \{x_{init}, r = 0\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3   while true do
4      $x_{new} \leftarrow \text{Sample}(i);$ 
5     if  $\text{InsideBall}(x_{new}, T)$  then
6       if  $\neg \text{CollisionFree}(x_{new})$  then
7          $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
8          $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
9       else
10        break;
11     $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
12     $\sigma \leftarrow \text{Steer}(x_{new}, x_{nearest});$ 
13    if  $\text{CollisionFree}(\sigma)$  then
14       $V.\text{add}(x_{new}, r_0);$ 
15       $E.\text{add}((x_{nearest}, x_{new}));$ 
16    else
17       $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
18 return  $T = (V, E).$ 

```

RRT* with Ball Trees



The RRT* Algorithm

```

1  $V \leftarrow \{x_{init}\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3    $x_{new} \leftarrow \text{Sample}(i);$ 
4    $X_{near} \leftarrow \text{Near}(V, x_{new});$ 
5   if  $X_{near} = \emptyset$  then
6      $X_{near} \leftarrow \text{Nearest}(V, x_{new});$ 
7    $L_{near} \leftarrow \text{PopulateSortedList}(X_{near}, x_{near});$ 
8    $x_{parent} \leftarrow \text{FindBestParent}(L_{near}, x_{new});$ 
9   if  $x_{parent} \neq \text{NULL}$  then
10     $V.\text{add}(x_{new});$ 
11     $E.\text{add}((x_{parent}, x_{new}));$ 
12     $E \leftarrow \text{RewireVertices}(E, X_{near}, x_{new});$ 
13 return  $T = (V, E).$ 

```

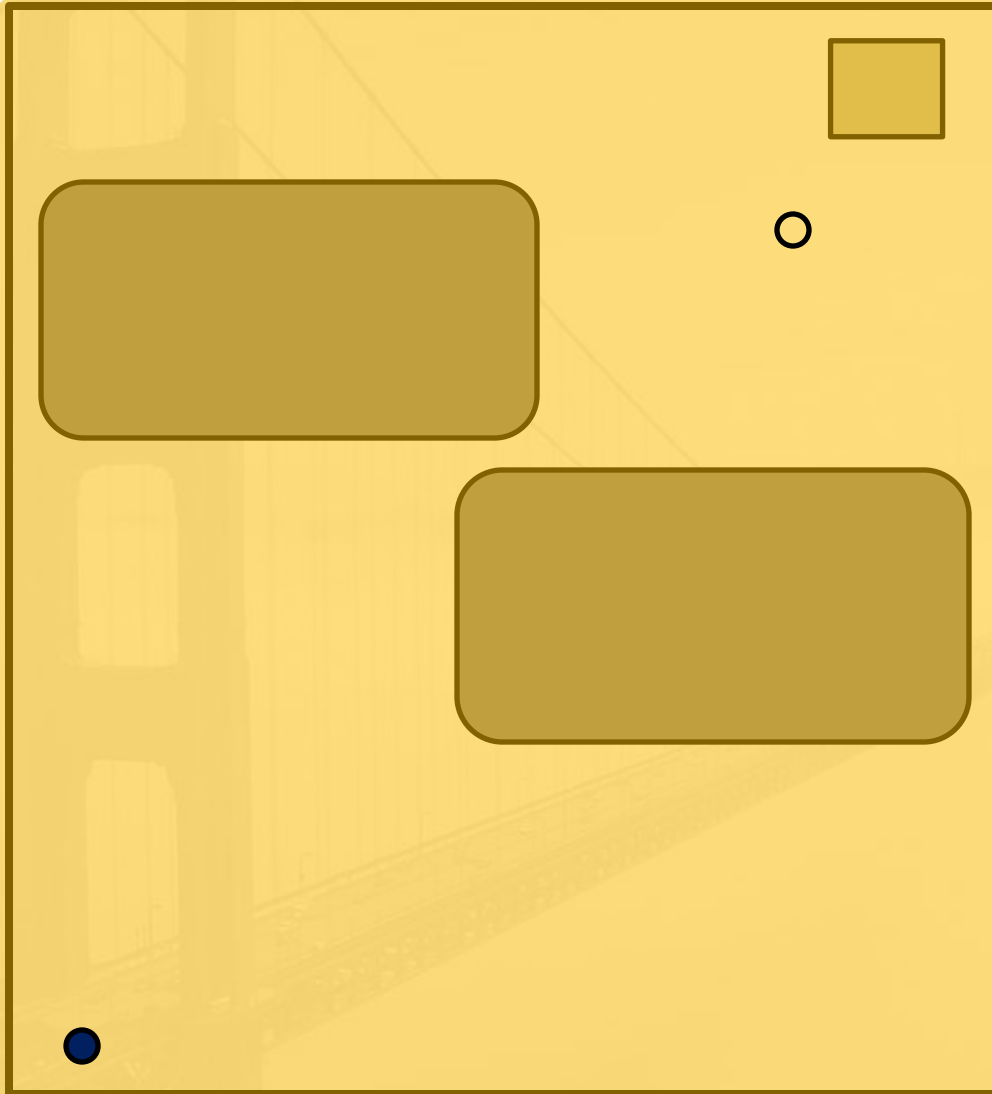
The Ball Tree Algorithm

```

1  $V \leftarrow \{x_{init}, r = 0\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3   while true do
4      $x_{new} \leftarrow \text{Sample}(i);$ 
5     if  $\text{InsideBall}(x_{new}, T)$  then
6       if  $\neg \text{CollisionFree}(x_{new})$  then
7          $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
8          $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
9       else
10        break;
11     $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
12     $\sigma \leftarrow \text{Steer}(x_{new}, x_{nearest});$ 
13    if  $\text{CollisionFree}(\sigma)$  then
14       $V.\text{add}(x_{new}, r_0);$ 
15       $E.\text{add}((x_{nearest}, x_{new}));$ 
16    else
17       $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
18 return  $T = (V, E).$ 

```

RRT* with Ball Trees



The RRT* Algorithm

```

1  $V \leftarrow \{x_{init}\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3    $x_{new} \leftarrow \text{Sample}(i);$ 
4    $X_{near} \leftarrow \text{Near}(V, x_{new});$ 
5   if  $X_{near} = \emptyset$  then
6      $X_{near} \leftarrow \text{Nearest}(V, x_{new});$ 
7    $L_{near} \leftarrow \text{PopulateSortedList}(X_{near}, x_{near});$ 
8    $x_{parent} \leftarrow \text{FindBestParent}(L_{near}, x_{new});$ 
9   if  $x_{parent} \neq \text{NULL}$  then
10     $V.\text{add}(x_{new});$ 
11     $E.\text{add}((x_{parent}, x_{new}));$ 
12     $E \leftarrow \text{RewireVertices}(E, X_{near}, x_{new});$ 
13 return  $T = (V, E).$ 

```

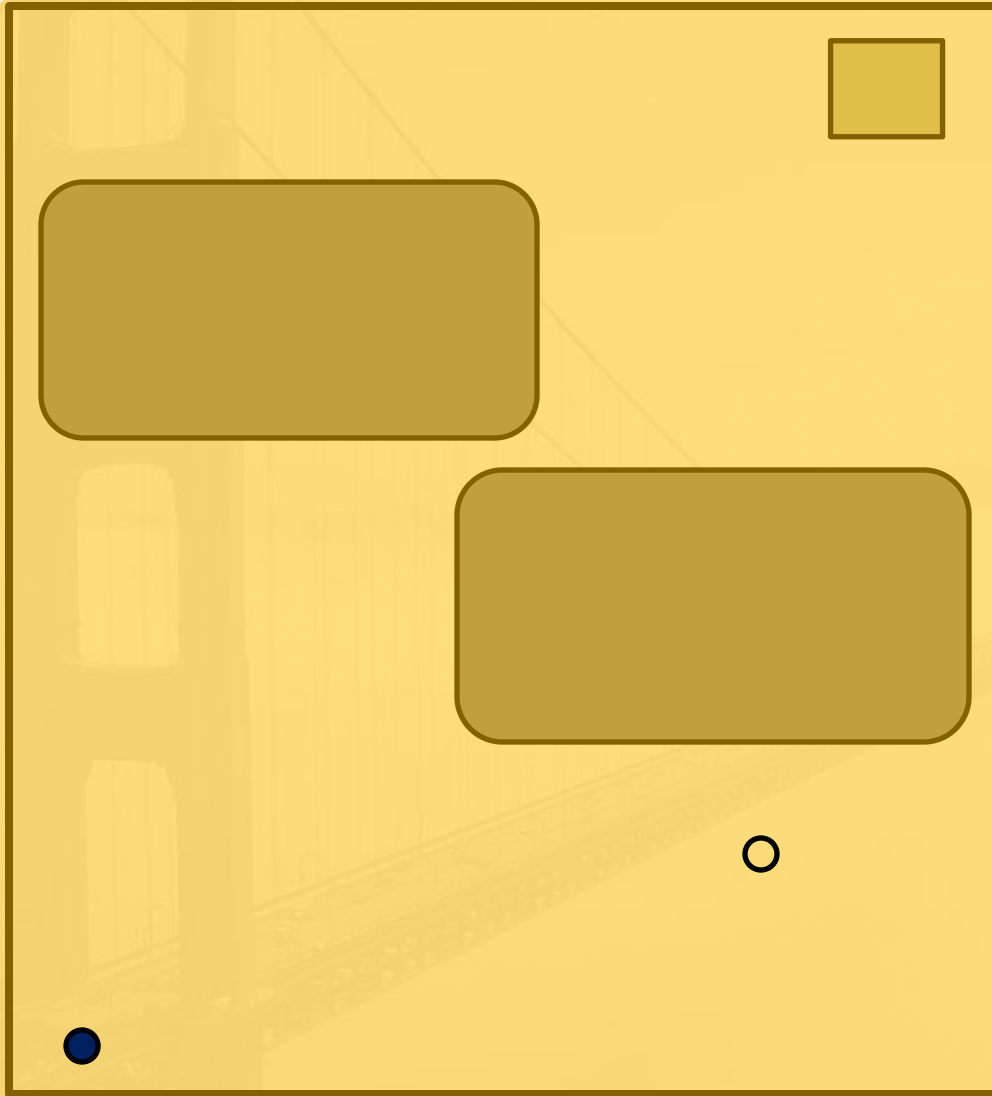
The Ball Tree Algorithm

```

1  $V \leftarrow \{x_{init}, r = 0\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3   while true do
4      $x_{new} \leftarrow \text{Sample}(i);$ 
5     if  $\text{InsideBall}(x_{new}, T)$  then
6       if  $\neg \text{CollisionFree}(x_{new})$  then
7          $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
8          $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
9       else
10        break;
11     $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
12     $\sigma \leftarrow \text{Steer}(x_{new}, x_{nearest});$ 
13    if  $\text{CollisionFree}(\sigma)$  then
14       $V.\text{add}(x_{new}, r_0);$ 
15       $E.\text{add}((x_{nearest}, x_{new}));$ 
16    else
17       $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
18 return  $T = (V, E).$ 

```

RRT* with Ball Trees



The RRT* Algorithm

```

1  $V \leftarrow \{x_{init}\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3    $x_{new} \leftarrow \text{Sample}(i);$ 
4    $X_{near} \leftarrow \text{Near}(V, x_{new});$ 
5   if  $X_{near} = \emptyset$  then
6      $X_{near} \leftarrow \text{Nearest}(V, x_{new});$ 
7    $L_{near} \leftarrow \text{PopulateSortedList}(X_{near}, x_{near});$ 
8    $x_{parent} \leftarrow \text{FindBestParent}(L_{near}, x_{new});$ 
9   if  $x_{parent} \neq \text{NULL}$  then
10     $V.\text{add}(x_{new});$ 
11     $E.\text{add}((x_{parent}, x_{new}));$ 
12     $E \leftarrow \text{RewireVertices}(E, X_{near}, x_{new});$ 
13 return  $T = (V, E).$ 

```

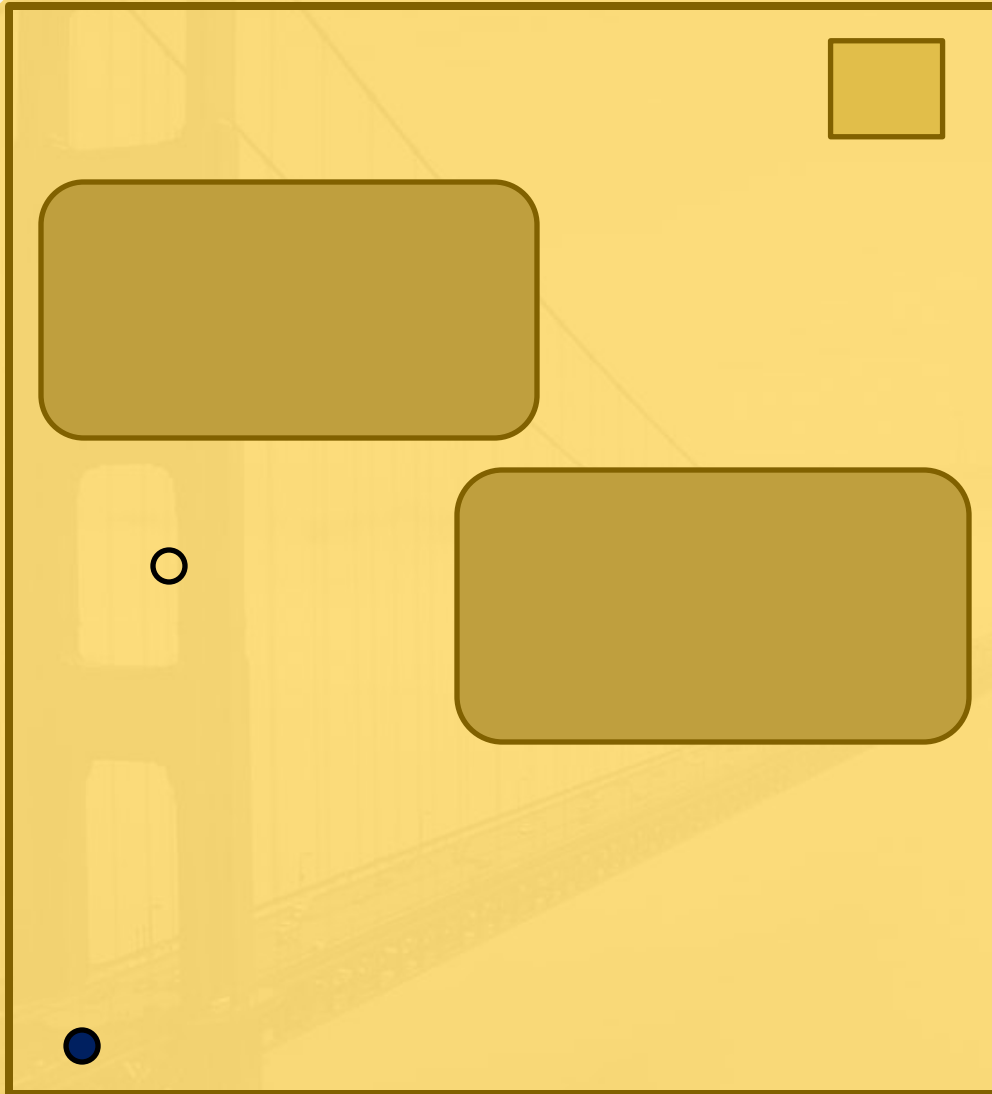
The Ball Tree Algorithm

```

1  $V \leftarrow \{x_{init}, r = 0\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3   while true do
4      $x_{new} \leftarrow \text{Sample}(i);$ 
5     if  $\text{InsideBall}(x_{new}, T)$  then
6       if  $\neg \text{CollisionFree}(x_{new})$  then
7          $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
8          $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
9       else
10        break;
11     $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
12     $\sigma \leftarrow \text{Steer}(x_{new}, x_{nearest});$ 
13    if  $\text{CollisionFree}(\sigma)$  then
14       $V.\text{add}(x_{new}, r_0);$ 
15       $E.\text{add}((x_{nearest}, x_{new}));$ 
16    else
17       $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
18 return  $T = (V, E).$ 

```

RRT* with Ball Trees



The RRT* Algorithm

```

1  $V \leftarrow \{x_{init}\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3    $x_{new} \leftarrow \text{Sample}(i);$ 
4    $X_{near} \leftarrow \text{Near}(V, x_{new});$ 
5   if  $X_{near} = \emptyset$  then
6      $X_{near} \leftarrow \text{Nearest}(V, x_{new});$ 
7    $L_{near} \leftarrow \text{PopulateSortedList}(X_{near}, x_{near});$ 
8    $x_{parent} \leftarrow \text{FindBestParent}(L_{near}, x_{new});$ 
9   if  $x_{parent} \neq \text{NULL}$  then
10     $V.\text{add}(x_{new});$ 
11     $E.\text{add}((x_{parent}, x_{new}));$ 
12     $E \leftarrow \text{RewireVertices}(E, X_{near}, x_{new});$ 
13 return  $T = (V, E).$ 

```

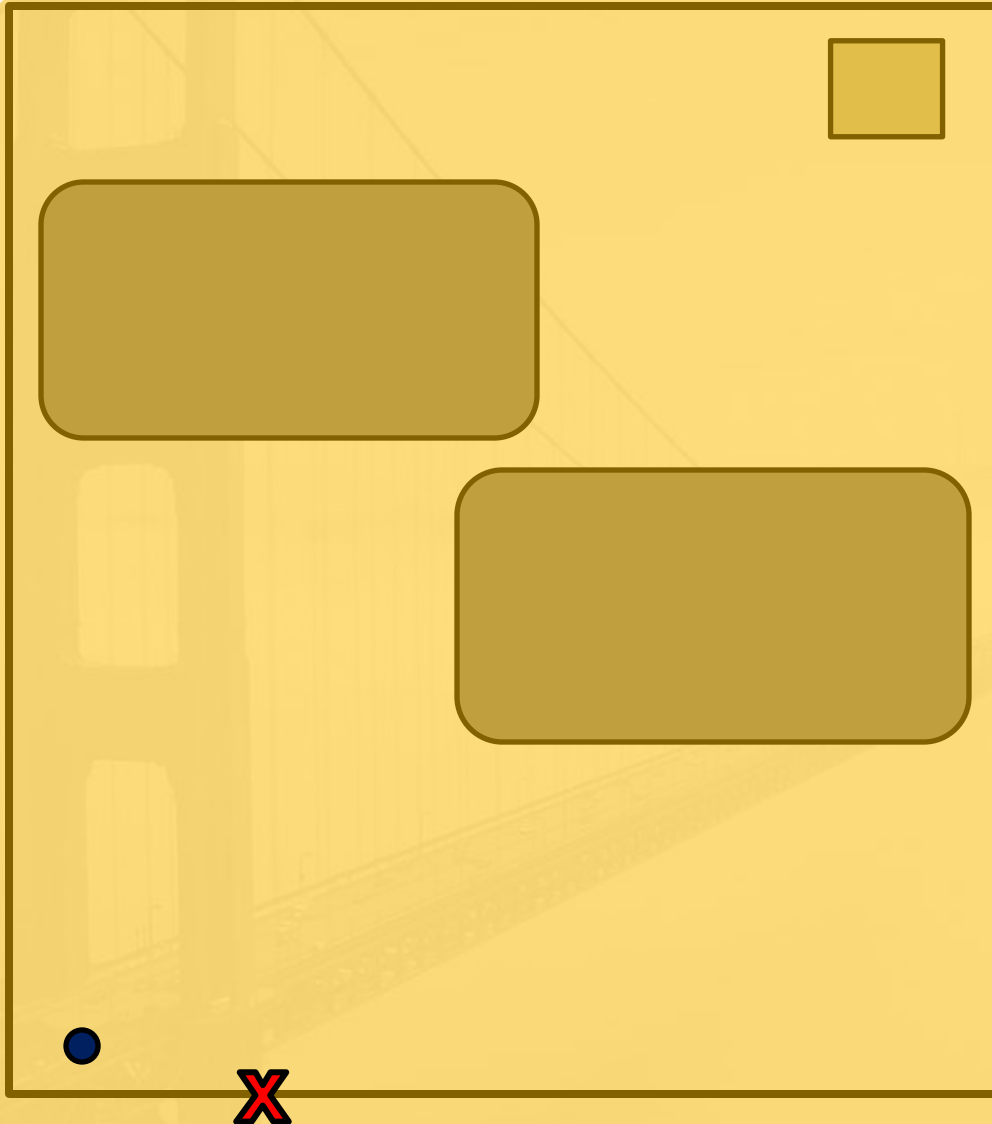
The Ball Tree Algorithm

```

1  $V \leftarrow \{x_{init}, r = 0\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3   while true do
4      $x_{new} \leftarrow \text{Sample}(i);$ 
5     if  $\text{InsideBall}(x_{new}, T)$  then
6       if  $\neg \text{CollisionFree}(x_{new})$  then
7          $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
8          $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
9       else
10        break;
11     $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
12     $\sigma \leftarrow \text{Steer}(x_{new}, x_{nearest});$ 
13    if  $\text{CollisionFree}(\sigma)$  then
14       $V.\text{add}(x_{new}, r_0);$ 
15       $E.\text{add}((x_{nearest}, x_{new}));$ 
16    else
17       $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
18 return  $T = (V, E).$ 

```

RRT* with Ball Trees



The RRT* Algorithm

```

1  $V \leftarrow \{x_{init}\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3    $x_{new} \leftarrow \text{Sample}(i);$ 
4    $X_{near} \leftarrow \text{Near}(V, x_{new});$ 
5   if  $X_{near} = \emptyset$  then
6      $X_{near} \leftarrow \text{Nearest}(V, x_{new});$ 
7    $L_{near} \leftarrow \text{PopulateSortedList}(X_{near}, x_{near});$ 
8    $x_{parent} \leftarrow \text{FindBestParent}(L_{near}, x_{new});$ 
9   if  $x_{parent} \neq \text{NULL}$  then
10     $V.\text{add}(x_{new});$ 
11     $E.\text{add}((x_{parent}, x_{new}));$ 
12     $E \leftarrow \text{RewireVertices}(E, X_{near}, x_{new});$ 
13 return  $T = (V, E).$ 

```

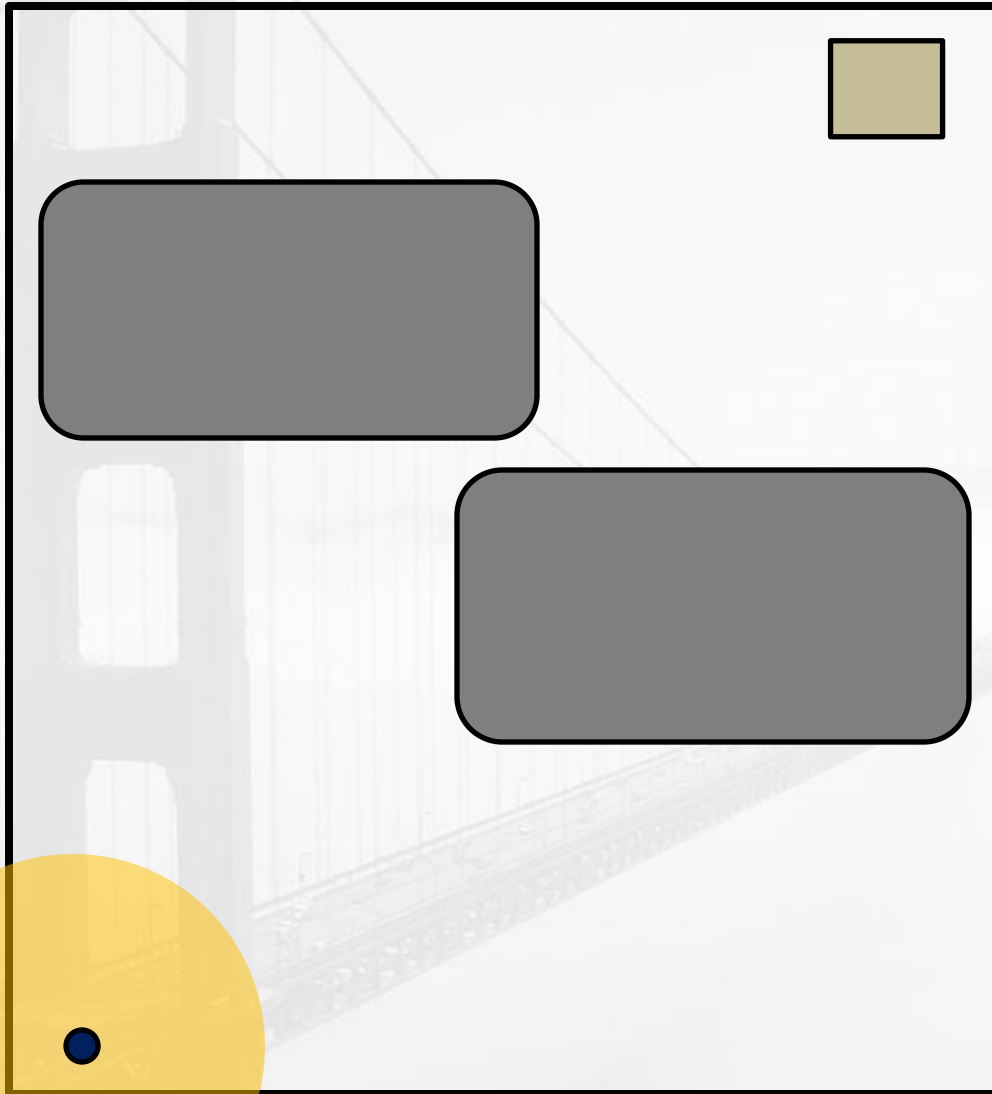
The Ball Tree Algorithm

```

1  $V \leftarrow \{x_{init}, r = 0\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3   while true do
4      $x_{new} \leftarrow \text{Sample}(i);$ 
5     if  $\text{InsideBall}(x_{new}, T)$  then
6       if  $\neg \text{CollisionFree}(x_{new})$  then
7          $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
8          $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
9       else
10        break;
11     $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
12     $\sigma \leftarrow \text{Steer}(x_{new}, x_{nearest});$ 
13    if  $\text{CollisionFree}(\sigma)$  then
14       $V.\text{add}(x_{new}, r_0);$ 
15       $E.\text{add}((x_{nearest}, x_{new}));$ 
16    else
17       $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
18 return  $T = (V, E).$ 

```

RRT* with Ball Trees



The RRT* Algorithm

```

1  $V \leftarrow \{x_{init}\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3    $x_{new} \leftarrow \text{Sample}(i);$ 
4    $X_{near} \leftarrow \text{Near}(V, x_{new});$ 
5   if  $X_{near} = \emptyset$  then
6      $X_{near} \leftarrow \text{Nearest}(V, x_{new});$ 
7    $L_{near} \leftarrow \text{PopulateSortedList}(X_{near}, x_{near});$ 
8    $x_{parent} \leftarrow \text{FindBestParent}(L_{near}, x_{new});$ 
9   if  $x_{parent} \neq \text{NULL}$  then
10     $V.\text{add}(x_{new});$ 
11     $E.\text{add}((x_{parent}, x_{new}));$ 
12     $E \leftarrow \text{RewireVertices}(E, X_{near}, x_{new});$ 
13 return  $T = (V, E).$ 

```

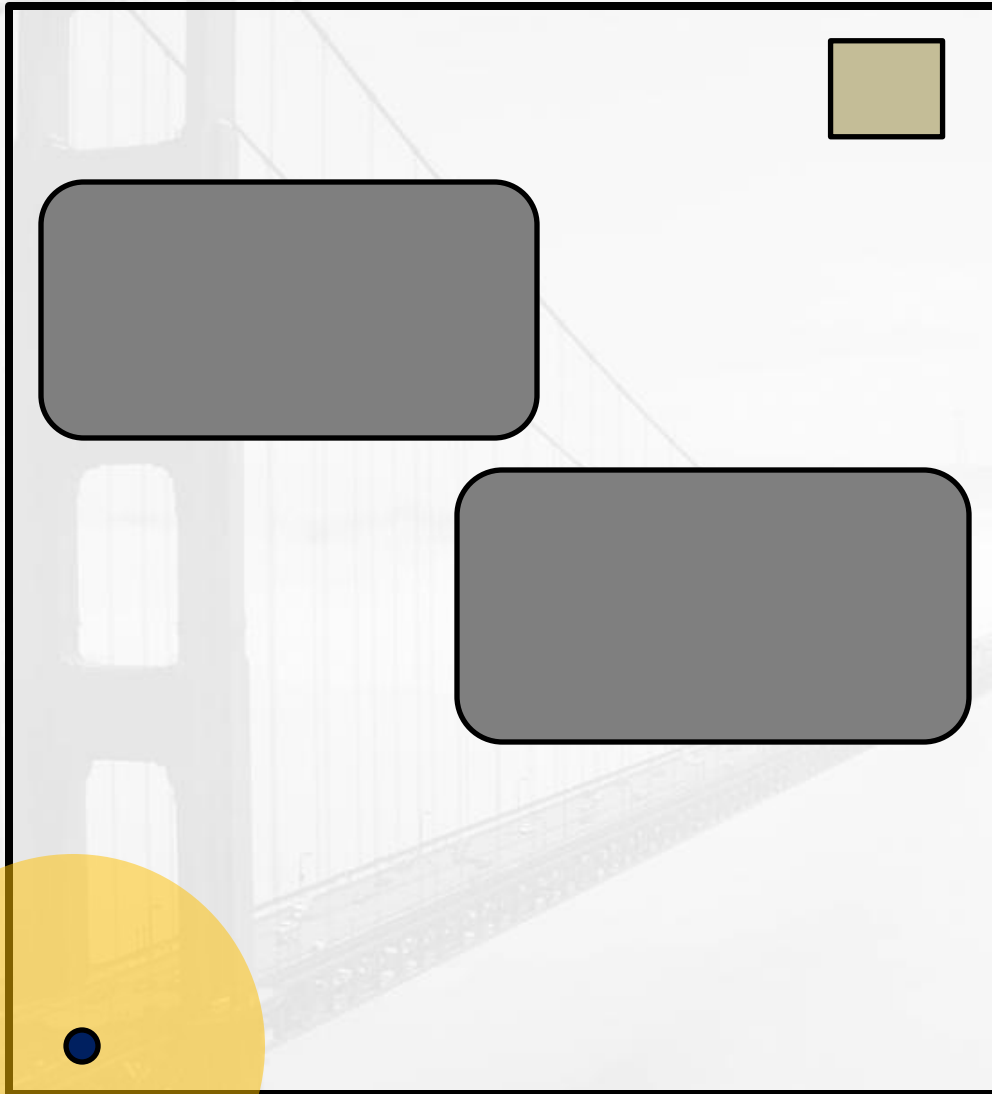
The Ball Tree Algorithm

```

1  $V \leftarrow \{x_{init}, r = 0\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3   while true do
4      $x_{new} \leftarrow \text{Sample}(i);$ 
5     if  $\text{InsideBall}(x_{new}, T)$  then
6       if  $\neg \text{CollisionFree}(x_{new})$  then
7          $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
8          $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
9       else
10        break;
11     $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
12     $\sigma \leftarrow \text{Steer}(x_{new}, x_{nearest});$ 
13    if  $\text{CollisionFree}(\sigma)$  then
14       $V.\text{add}(x_{new}, r_0);$ 
15       $E.\text{add}((x_{nearest}, x_{new}));$ 
16    else
17       $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
18 return  $T = (V, E).$ 

```

RRT* with Ball Trees



The RRT* Algorithm

```

1  $V \leftarrow \{x_{init}\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3    $x_{new} \leftarrow \text{Sample}(i);$ 
4    $X_{near} \leftarrow \text{Near}(V, x_{new});$ 
5   if  $X_{near} = \emptyset$  then
6      $X_{near} \leftarrow \text{Nearest}(V, x_{new});$ 
7    $L_{near} \leftarrow \text{PopulateSortedList}(X_{near}, x_{near});$ 
8    $x_{parent} \leftarrow \text{FindBestParent}(L_{near}, x_{new});$ 
9   if  $x_{parent} \neq \text{NULL}$  then
10     $V.\text{add}(x_{new});$ 
11     $E.\text{add}((x_{parent}, x_{new}));$ 
12     $E \leftarrow \text{RewireVertices}(E, X_{near}, x_{new});$ 
13 return  $T = (V, E).$ 

```

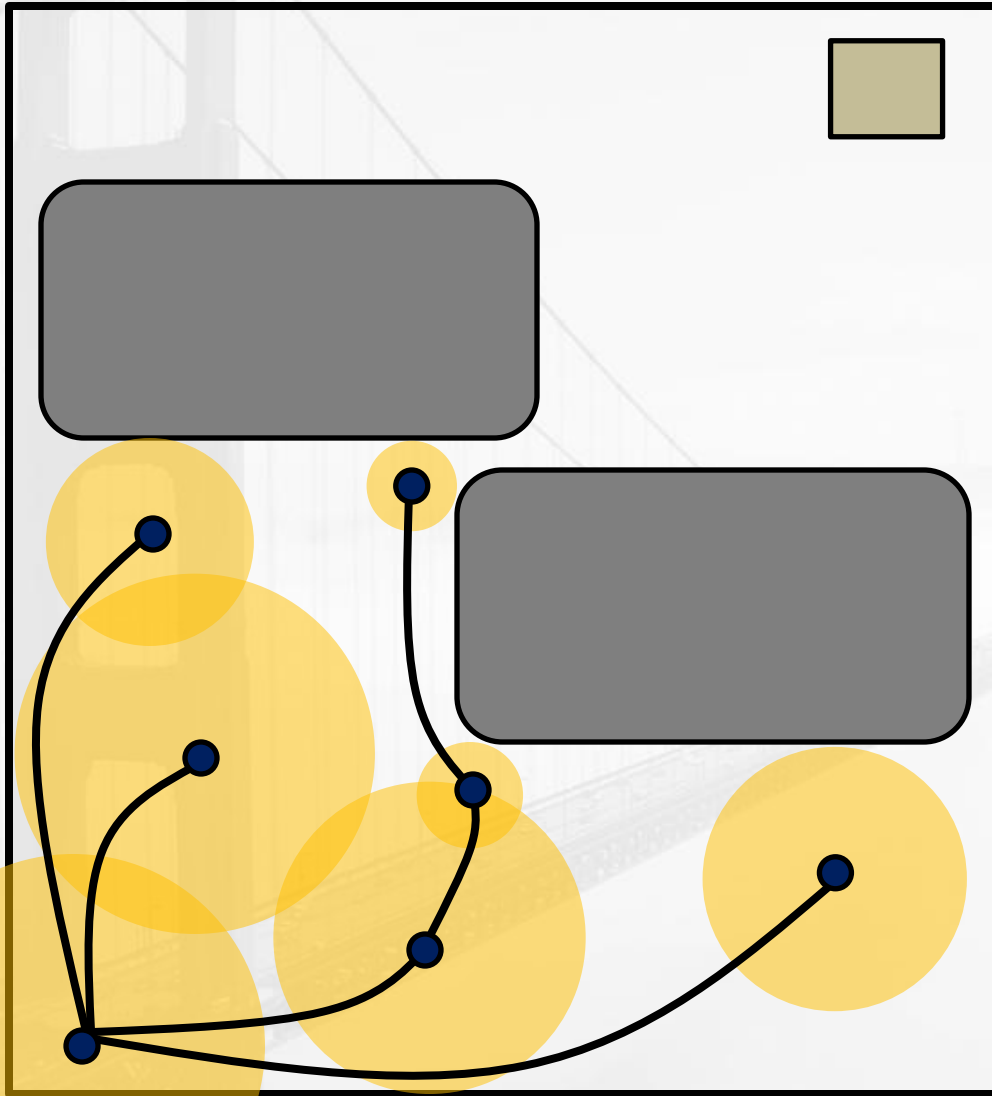
The Ball Tree Algorithm

```

1  $V \leftarrow \{x_{init}, r = 0\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3   while true do
4      $x_{new} \leftarrow \text{Sample}(i);$ 
5     if  $\text{InsideBall}(x_{new}, T)$  then
6       if  $\neg \text{CollisionFree}(x_{new})$  then
7          $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
8          $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
9       else
10        break;
11     $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
12     $\sigma \leftarrow \text{Steer}(x_{new}, x_{nearest});$ 
13    if  $\text{CollisionFree}(\sigma)$  then
14       $V.\text{add}(x_{new}, r_0);$ 
15       $E.\text{add}((x_{nearest}, x_{new}));$ 
16    else
17       $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
18 return  $T = (V, E).$ 

```

RRT* with Ball Trees



The RRT* Algorithm

```

1  $V \leftarrow \{x_{init}\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3    $x_{new} \leftarrow \text{Sample}(i);$ 
4    $X_{near} \leftarrow \text{Near}(V, x_{new});$ 
5   if  $X_{near} = \emptyset$  then
6      $X_{near} \leftarrow \text{Nearest}(V, x_{new});$ 
7    $L_{near} \leftarrow \text{PopulateSortedList}(X_{near}, x_{near});$ 
8    $x_{parent} \leftarrow \text{FindBestParent}(L_{near}, x_{new});$ 
9   if  $x_{parent} \neq \text{NULL}$  then
10     $V.add(x_{new});$ 
11     $E.add((x_{parent}, x_{new}));$ 
12     $E \leftarrow \text{RewireVertices}(E, X_{near}, x_{new});$ 
13 return  $T = (V, E).$ 

```

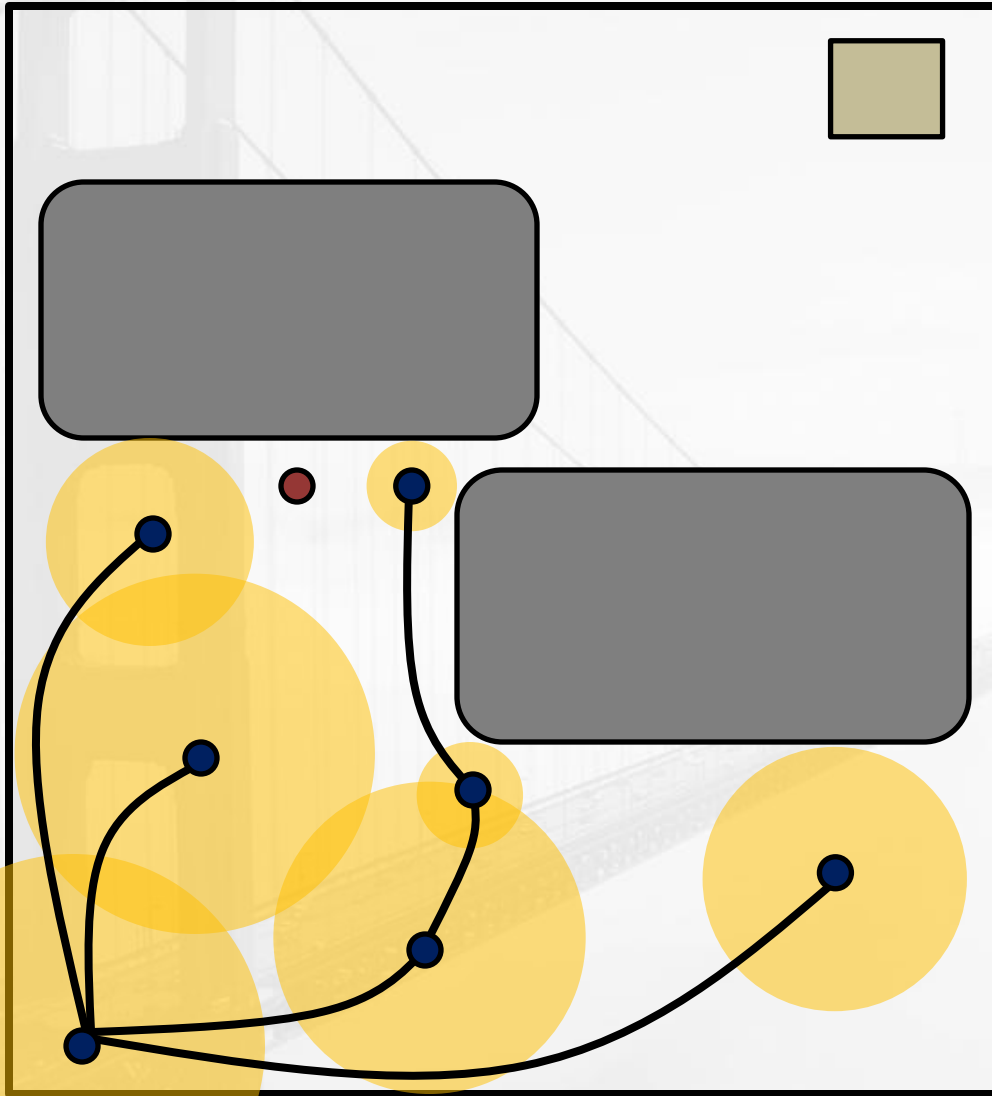
The Ball Tree Algorithm

```

1  $V \leftarrow \{x_{init}, r = 0\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3   while true do
4      $x_{new} \leftarrow \text{Sample}(i);$ 
5     if  $\text{InsideBall}(x_{new}, T)$  then
6       if  $\neg \text{CollisionFree}(x_{new})$  then
7          $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
8          $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
9       else
10        break;
11     $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
12     $\sigma \leftarrow \text{Steer}(x_{new}, x_{nearest});$ 
13    if  $\text{CollisionFree}(\sigma)$  then
14       $V.add(x_{new}, r_0);$ 
15       $E.add((x_{nearest}, x_{new}));$ 
16    else
17       $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
18 return  $T = (V, E).$ 

```

RRT* with Ball Trees



The RRT* Algorithm

```

1  $V \leftarrow \{x_{init}\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3    $x_{new} \leftarrow \text{Sample}(i);$ 
4    $X_{near} \leftarrow \text{Near}(V, x_{new});$ 
5   if  $X_{near} = \emptyset$  then
6      $X_{near} \leftarrow \text{Nearest}(V, x_{new});$ 
7    $L_{near} \leftarrow \text{PopulateSortedList}(X_{near}, x_{near});$ 
8    $x_{parent} \leftarrow \text{FindBestParent}(L_{near}, x_{new});$ 
9   if  $x_{parent} \neq \text{NULL}$  then
10     $V.add(x_{new});$ 
11     $E.add((x_{parent}, x_{new}));$ 
12     $E \leftarrow \text{RewireVertices}(E, X_{near}, x_{new});$ 
13 return  $T = (V, E).$ 

```

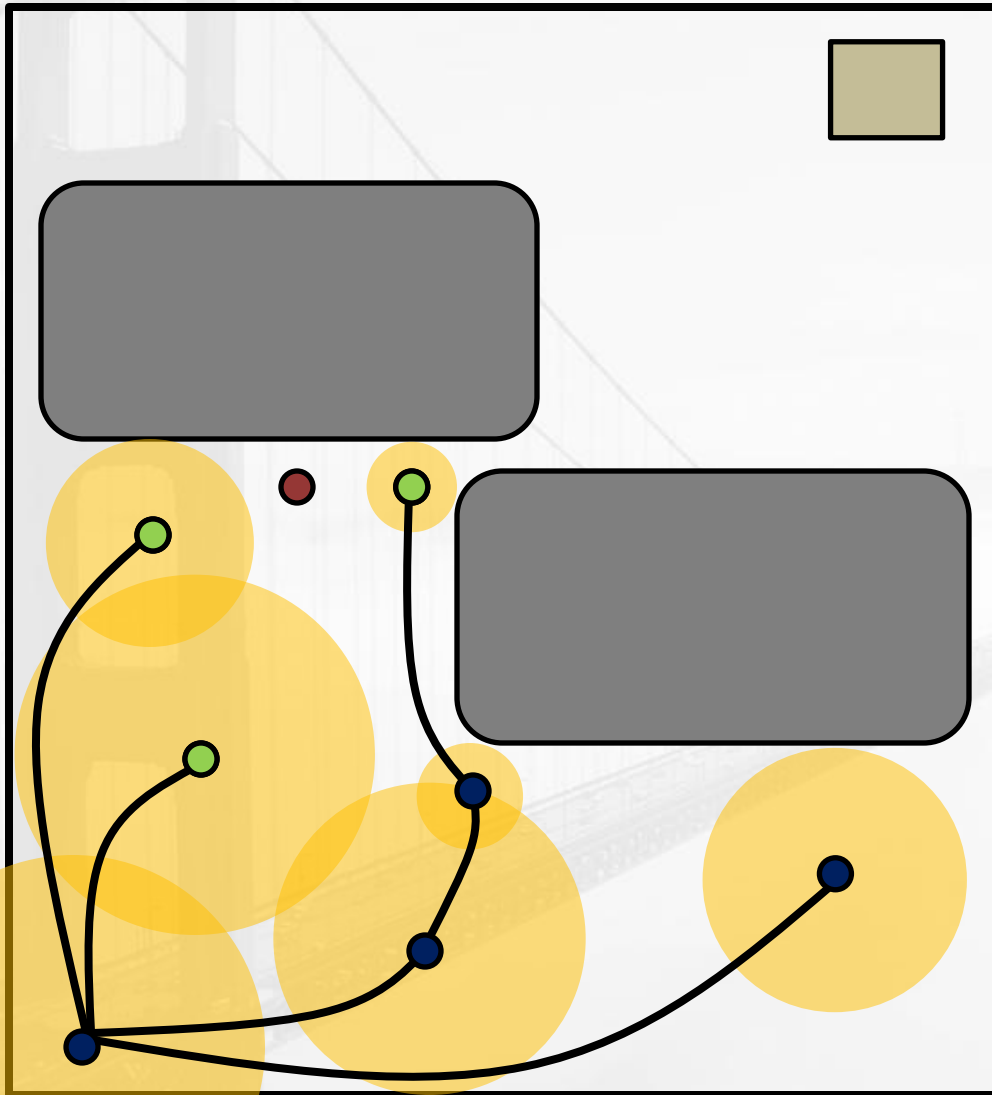
The Ball Tree Algorithm

```

1  $V \leftarrow \{x_{init}, r = 0\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3   while true do
4      $x_{new} \leftarrow \text{Sample}(i);$ 
5     if  $\text{InsideBall}(x_{new}, T)$  then
6       if  $\neg \text{CollisionFree}(x_{new})$  then
7          $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
8          $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
9       else
10        break;
11     $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
12     $\sigma \leftarrow \text{Steer}(x_{new}, x_{nearest});$ 
13    if  $\text{CollisionFree}(\sigma)$  then
14       $V.add(x_{new}, r_0);$ 
15       $E.add((x_{nearest}, x_{new}));$ 
16    else
17       $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
18 return  $T = (V, E).$ 

```

RRT* with Ball Trees



The RRT* Algorithm

```

1  $V \leftarrow \{x_{init}\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3    $x_{new} \leftarrow \text{Sample}(i);$ 
4    $X_{near} \leftarrow \text{Near}(V, x_{new});$ 
5   if  $X_{near} = \emptyset$  then
6      $X_{near} \leftarrow \text{Nearest}(V, x_{new});$ 
7    $L_{near} \leftarrow \text{PopulateSortedList}(X_{near}, x_{near});$ 
8    $x_{parent} \leftarrow \text{FindBestParent}(L_{near}, x_{new});$ 
9   if  $x_{parent} \neq \text{NULL}$  then
10     $V.add(x_{new});$ 
11     $E.add((x_{parent}, x_{new}));$ 
12     $E \leftarrow \text{RewireVertices}(E, X_{near}, x_{new});$ 
13 return  $T = (V, E);$ 

```

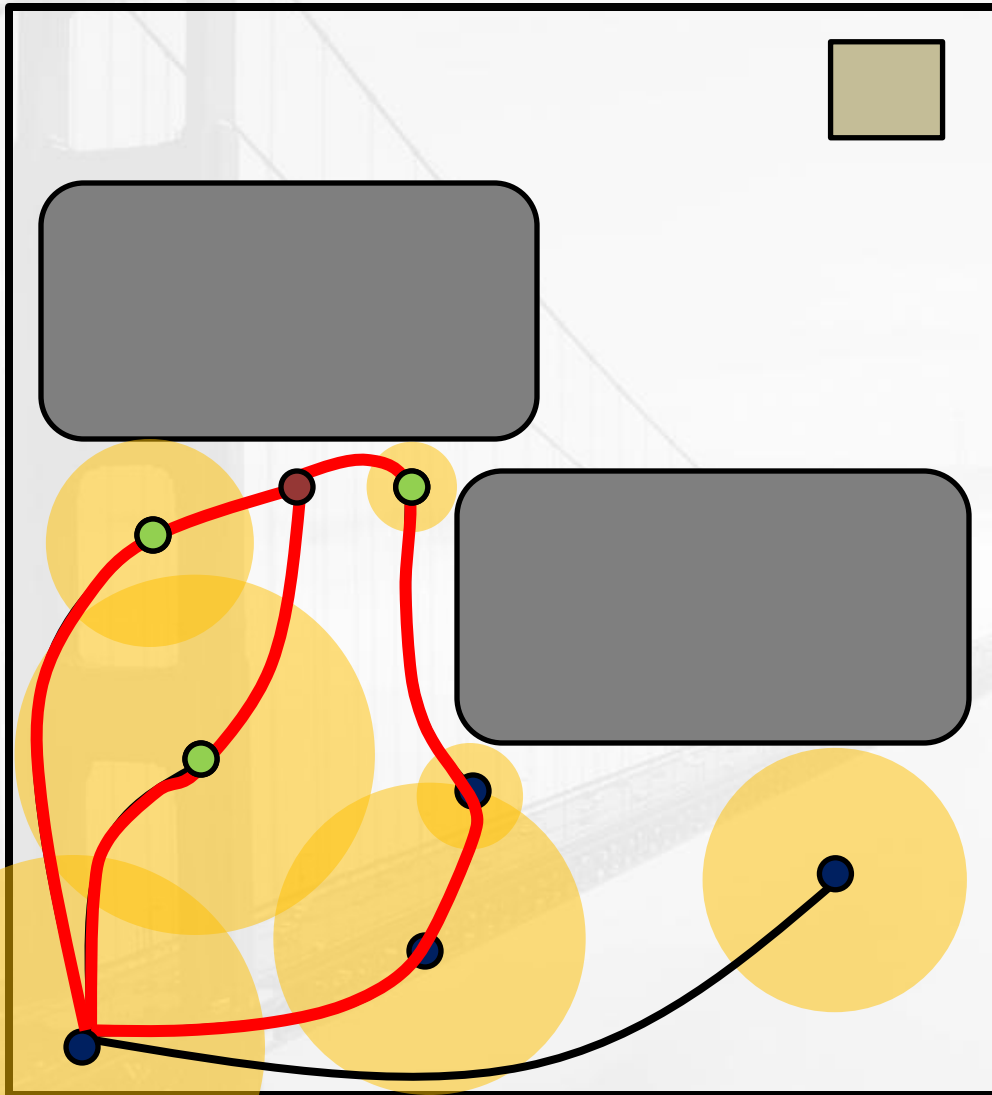
The Ball Tree Algorithm

```

1  $V \leftarrow \{x_{init}, r = 0\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3   while true do
4      $x_{new} \leftarrow \text{Sample}(i);$ 
5     if  $\text{InsideBall}(x_{new}, T)$  then
6       if  $\neg \text{CollisionFree}(x_{new})$  then
7          $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
8          $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
9       else
10        break;
11     $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
12     $\sigma \leftarrow \text{Steer}(x_{new}, x_{nearest});$ 
13    if  $\text{CollisionFree}(\sigma)$  then
14       $V.add(x_{new}, r_0);$ 
15       $E.add((x_{nearest}, x_{new}));$ 
16    else
17       $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
18 return  $T = (V, E);$ 

```

RRT* with Ball Trees



The RRT* Algorithm

```

1  $V \leftarrow \{x_{init}\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3    $x_{new} \leftarrow \text{Sample}(i);$ 
4    $X_{near} \leftarrow \text{Near}(V, x_{new});$ 
5   if  $X_{near} = \emptyset$  then
6      $X_{near} \leftarrow \text{Nearest}(V, x_{new});$ 
7    $L_{near} \leftarrow \text{PopulateSortedList}(X_{near}, x_{new});$ 
8    $x_{parent} \leftarrow \text{FindBestParent}(L_{near}, x_{new});$ 
9   if  $x_{parent} \neq \text{NULL}$  then
10     $V.add(x_{new});$ 
11     $E.add((x_{parent}, x_{new}));$ 
12     $E \leftarrow \text{RewireVertices}(E, X_{near}, x_{new});$ 
13 return  $T = (V, E).$ 

```

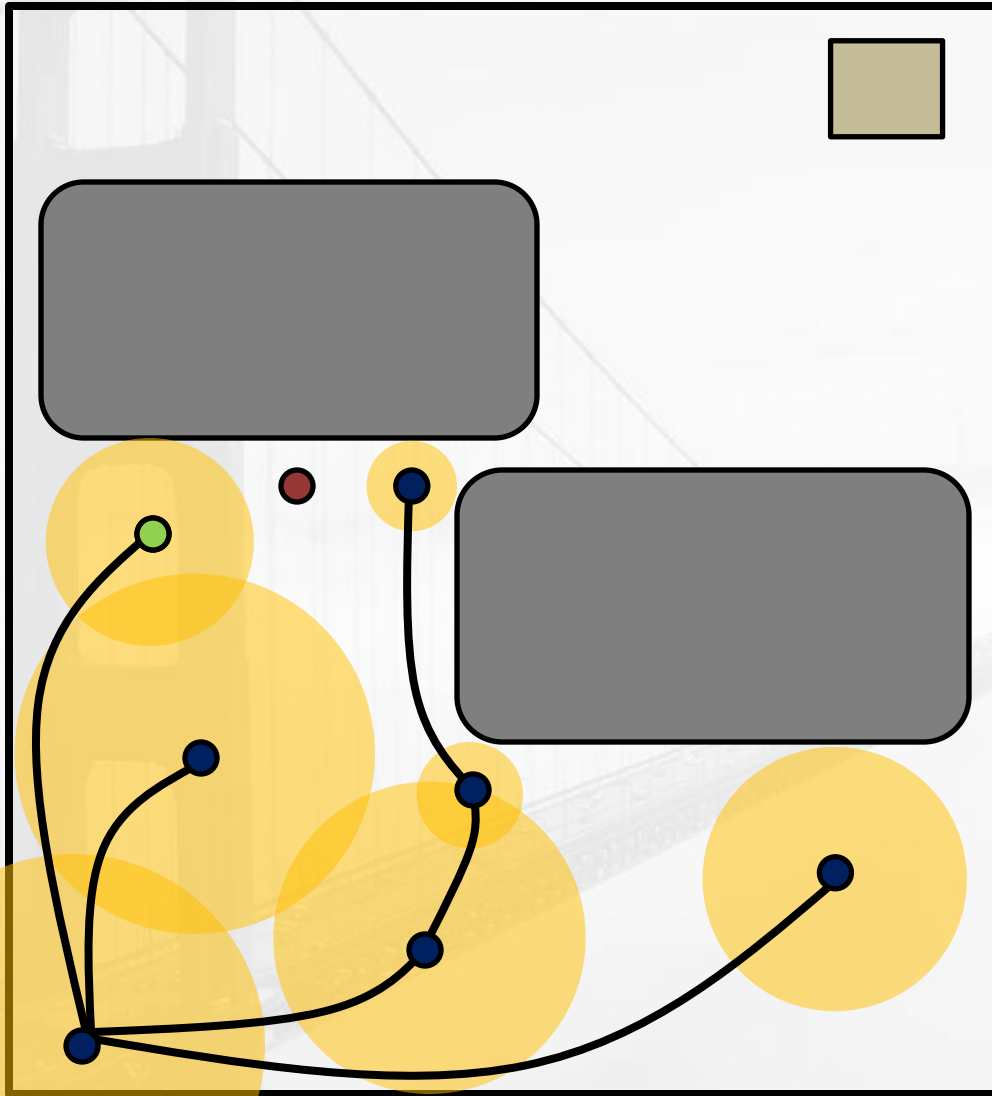
The Ball Tree Algorithm

```

1  $V \leftarrow \{x_{init}, r = 0\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3   while true do
4      $x_{new} \leftarrow \text{Sample}(i);$ 
5     if  $\text{InsideBall}(x_{new}, T)$  then
6       if  $\neg \text{CollisionFree}(x_{new})$  then
7          $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
8          $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
9       else
10        break;
11     $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
12     $\sigma \leftarrow \text{Steer}(x_{new}, x_{nearest});$ 
13    if  $\text{CollisionFree}(\sigma)$  then
14       $V.add(x_{new}, r_0);$ 
15       $E.add((x_{nearest}, x_{new}));$ 
16    else
17       $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
18 return  $T = (V, E).$ 

```

RRT* with Ball Trees



The RRT* Algorithm

```

1  $V \leftarrow \{x_{init}\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3    $x_{new} \leftarrow \text{Sample}(i);$ 
4    $X_{near} \leftarrow \text{Near}(V, x_{new});$ 
5   if  $X_{near} = \emptyset$  then
6      $X_{near} \leftarrow \text{Nearest}(V, x_{new});$ 
7    $L_{near} \leftarrow \text{PopulateSortedList}(X_{near}, x_{near});$ 
8    $x_{parent} \leftarrow \text{FindBestParent}(L_{near}, x_{new});$ 
9   if  $x_{parent} \neq \text{NULL}$  then
10     $V.add(x_{new});$ 
11     $E.add((x_{parent}, x_{new}));$ 
12     $E \leftarrow \text{RewireVertices}(E, X_{near}, x_{new});$ 
13 return  $T = (V, E).$ 

```

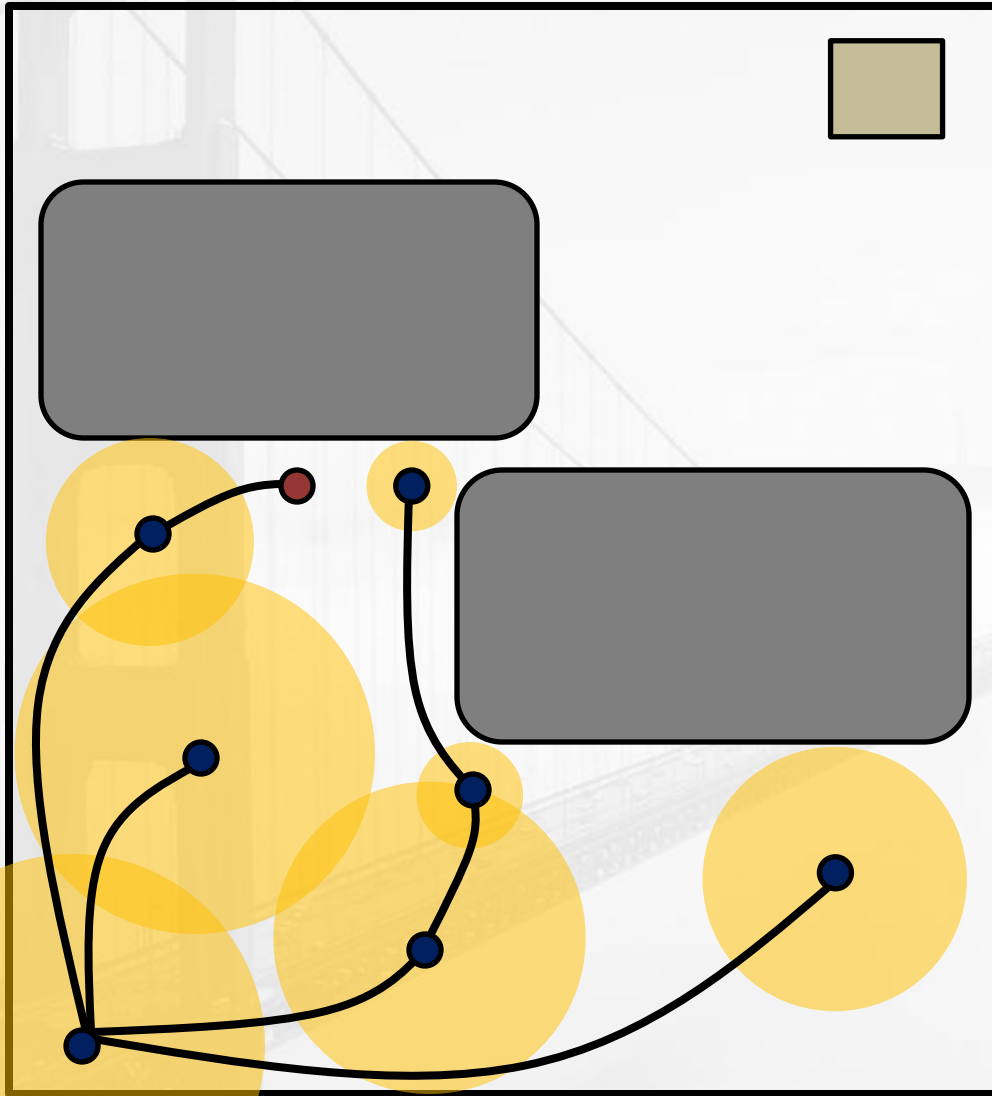
The Ball Tree Algorithm

```

1  $V \leftarrow \{x_{init}, r = 0\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3   while true do
4      $x_{new} \leftarrow \text{Sample}(i);$ 
5     if  $\text{InsideBall}(x_{new}, T)$  then
6       if  $\neg \text{CollisionFree}(x_{new})$  then
7          $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
8          $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
9       else
10        break;
11     $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
12     $\sigma \leftarrow \text{Steer}(x_{new}, x_{nearest});$ 
13    if  $\text{CollisionFree}(\sigma)$  then
14       $V.add(x_{new}, r_0);$ 
15       $E.add((x_{nearest}, x_{new}));$ 
16    else
17       $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
18 return  $T = (V, E).$ 

```

RRT* with Ball Trees



The RRT* Algorithm

```

1  $V \leftarrow \{x_{init}\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3    $x_{new} \leftarrow \text{Sample}(i);$ 
4    $X_{near} \leftarrow \text{Near}(V, x_{new});$ 
5   if  $X_{near} = \emptyset$  then
6      $X_{near} \leftarrow \text{Nearest}(V, x_{new});$ 
7    $L_{near} \leftarrow \text{PopulateSortedList}(X_{near}, x_{near});$ 
8    $x_{parent} \leftarrow \text{FindBestParent}(L_{near}, x_{new});$ 
9   if  $x_{parent} \neq \text{NULL}$  then
10     $V.add(x_{new});$ 
11     $E.add((x_{parent}, x_{new}));$ 
12     $E \leftarrow \text{RewireVertices}(E, X_{near}, x_{new});$ 
13 return  $T = (V, E).$ 

```

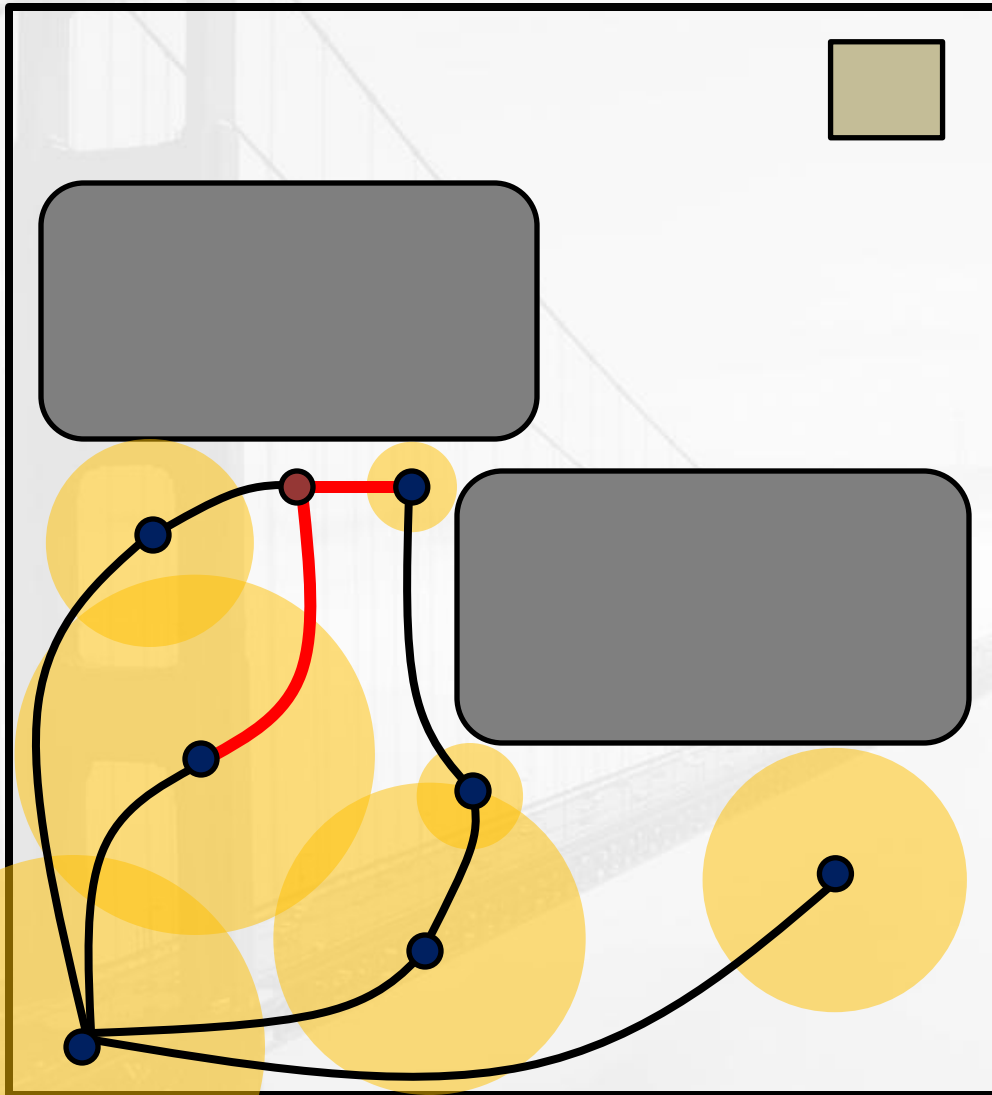
The Ball Tree Algorithm

```

1  $V \leftarrow \{x_{init}, r = 0\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3   while true do
4      $x_{new} \leftarrow \text{Sample}(i);$ 
5     if  $\text{InsideBall}(x_{new}, T)$  then
6       if  $\neg \text{CollisionFree}(x_{new})$  then
7          $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
8          $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
9       else
10        break;
11     $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
12     $\sigma \leftarrow \text{Steer}(x_{new}, x_{nearest});$ 
13    if  $\text{CollisionFree}(\sigma)$  then
14       $V.add(x_{new}, r_0);$ 
15       $E.add((x_{nearest}, x_{new}));$ 
16    else
17       $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
18 return  $T = (V, E).$ 

```

RRT* with Ball Trees



The RRT* Algorithm

```

1  $V \leftarrow \{x_{init}\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3    $x_{new} \leftarrow \text{Sample}(i);$ 
4    $X_{near} \leftarrow \text{Near}(V, x_{new});$ 
5   if  $X_{near} = \emptyset$  then
6      $X_{near} \leftarrow \text{Nearest}(V, x_{new});$ 
7    $L_{near} \leftarrow \text{PopulateSortedList}(X_{near}, x_{near});$ 
8    $x_{parent} \leftarrow \text{FindBestParent}(L_{near}, x_{new});$ 
9   if  $x_{parent} \neq \text{NULL}$  then
10     $V.add(x_{new});$ 
11     $E.add((x_{parent}, x_{new}));$ 
12     $E \leftarrow \text{RewireVertices}(E, X_{near}, x_{new});$ 
13 return  $T = (V, E).$ 

```

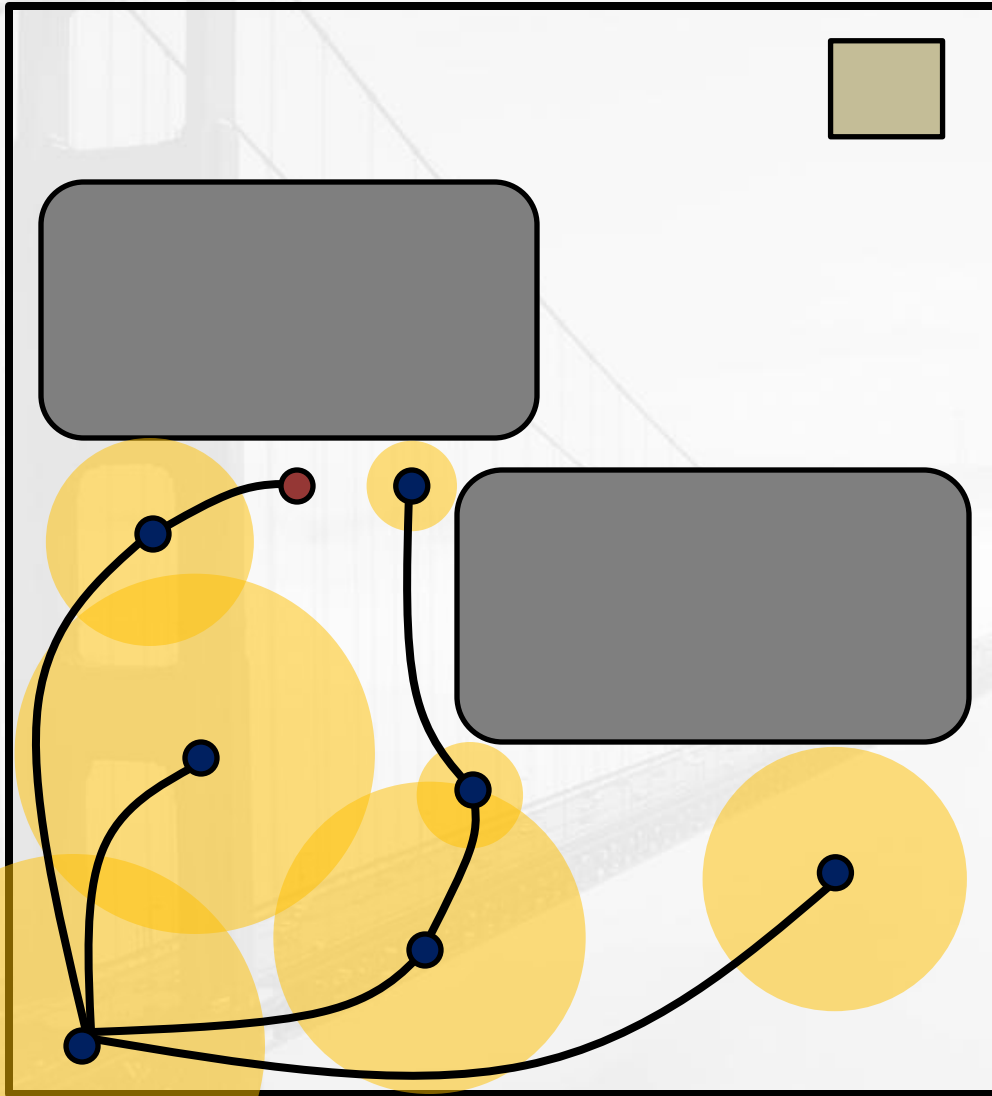
The Ball Tree Algorithm

```

1  $V \leftarrow \{x_{init}, r = 0\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3   while true do
4      $x_{new} \leftarrow \text{Sample}(i);$ 
5     if  $\text{InsideBall}(x_{new}, T)$  then
6       if  $\neg \text{CollisionFree}(x_{new})$  then
7          $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
8          $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
9       else
10        break;
11     $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
12     $\sigma \leftarrow \text{Steer}(x_{new}, x_{nearest});$ 
13    if  $\text{CollisionFree}(\sigma)$  then
14       $V.add(x_{new}, r_0);$ 
15       $E.add((x_{nearest}, x_{new}));$ 
16    else
17       $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
18 return  $T = (V, E).$ 

```

RRT* with Ball Trees



The RRT* Algorithm

```

1  $V \leftarrow \{x_{init}\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3    $x_{new} \leftarrow \text{Sample}(i);$ 
4    $X_{near} \leftarrow \text{Near}(V, x_{new});$ 
5   if  $X_{near} = \emptyset$  then
6      $X_{near} \leftarrow \text{Nearest}(V, x_{new});$ 
7    $L_{near} \leftarrow \text{PopulateSortedList}(X_{near}, x_{near});$ 
8    $x_{parent} \leftarrow \text{FindBestParent}(L_{near}, x_{new});$ 
9   if  $x_{parent} \neq \text{NULL}$  then
10     $V.add(x_{new});$ 
11     $E.add((x_{parent}, x_{new}));$ 
12     $E \leftarrow \text{RewireVertices}(E, X_{near}, x_{new});$ 
13 return  $T = (V, E);$ 

```

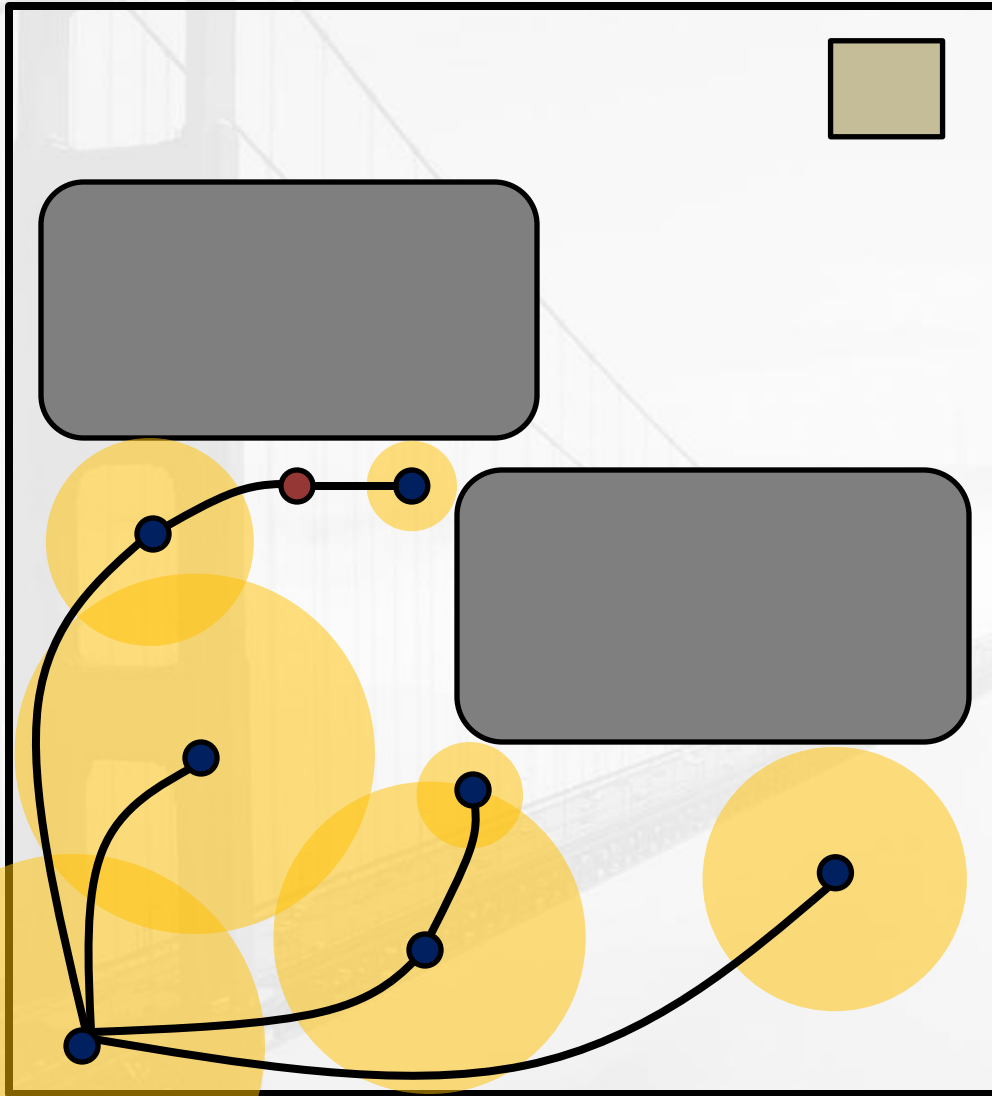
The Ball Tree Algorithm

```

1  $V \leftarrow \{x_{init}, r = 0\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3   while true do
4      $x_{new} \leftarrow \text{Sample}(i);$ 
5     if  $\text{InsideBall}(x_{new}, T)$  then
6       if  $\neg \text{CollisionFree}(x_{new})$  then
7          $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
8          $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
9       else
10        break;
11     $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
12     $\sigma \leftarrow \text{Steer}(x_{new}, x_{nearest});$ 
13    if  $\text{CollisionFree}(\sigma)$  then
14       $V.add(x_{new}, r_0);$ 
15       $E.add((x_{nearest}, x_{new}));$ 
16    else
17       $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
18 return  $T = (V, E);$ 

```

RRT* with Ball Trees



The RRT* Algorithm

```

1  $V \leftarrow \{x_{init}\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3    $x_{new} \leftarrow \text{Sample}(i);$ 
4    $X_{near} \leftarrow \text{Near}(V, x_{new});$ 
5   if  $X_{near} = \emptyset$  then
6      $X_{near} \leftarrow \text{Nearest}(V, x_{new});$ 
7    $L_{near} \leftarrow \text{PopulateSortedList}(X_{near}, x_{near});$ 
8    $x_{parent} \leftarrow \text{FindBestParent}(L_{near}, x_{new});$ 
9   if  $x_{parent} \neq \text{NULL}$  then
10     $V.add(x_{new});$ 
11     $E.add((x_{parent}, x_{new}));$ 
12     $E \leftarrow \text{RewireVertices}(E, X_{near}, x_{new});$ 
13 return  $T = (V, E).$ 

```

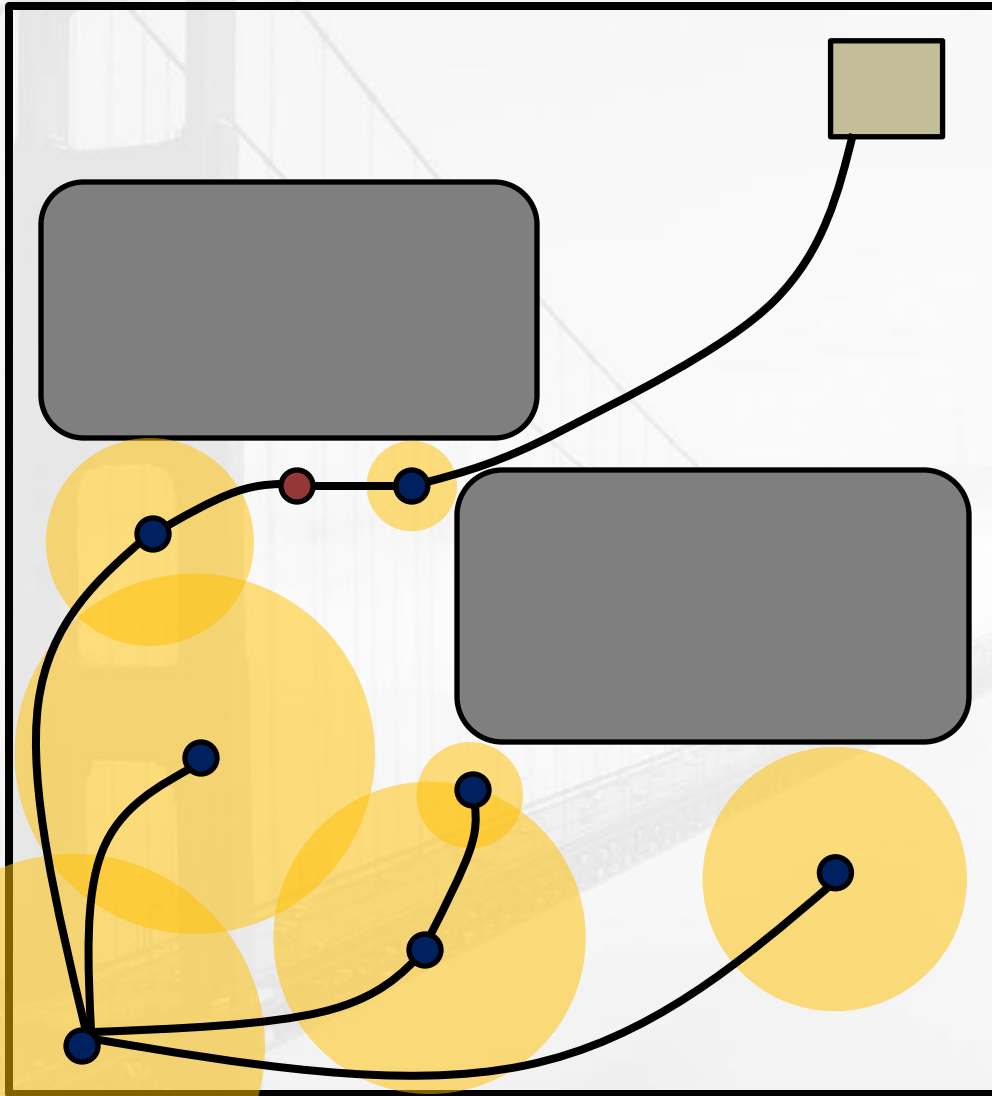
The Ball Tree Algorithm

```

1  $V \leftarrow \{x_{init}, r = 0\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3   while true do
4      $x_{new} \leftarrow \text{Sample}(i);$ 
5     if  $\text{InsideBall}(x_{new}, T)$  then
6       if  $\neg \text{CollisionFree}(x_{new})$  then
7          $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
8          $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
9       else
10        break;
11     $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
12     $\sigma \leftarrow \text{Steer}(x_{new}, x_{nearest});$ 
13    if  $\text{CollisionFree}(\sigma)$  then
14       $V.add(x_{new}, r_0);$ 
15       $E.add((x_{nearest}, x_{new}));$ 
16    else
17       $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
18 return  $T = (V, E).$ 

```

RRT* with Ball Trees



The RRT* Algorithm

```

1  $V \leftarrow \{x_{init}\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3    $x_{new} \leftarrow \text{Sample}(i);$ 
4    $X_{near} \leftarrow \text{Near}(V, x_{new});$ 
5   if  $X_{near} = \emptyset$  then
6      $X_{near} \leftarrow \text{Nearest}(V, x_{new});$ 
7    $L_{near} \leftarrow \text{PopulateSortedList}(X_{near}, x_{near});$ 
8    $x_{parent} \leftarrow \text{FindBestParent}(L_{near}, x_{new});$ 
9   if  $x_{parent} \neq \text{NULL}$  then
10     $V.add(x_{new});$ 
11     $E.add((x_{parent}, x_{new}));$ 
12     $E \leftarrow \text{RewireVertices}(E, X_{near}, x_{new});$ 
13 return  $T = (V, E).$ 

```

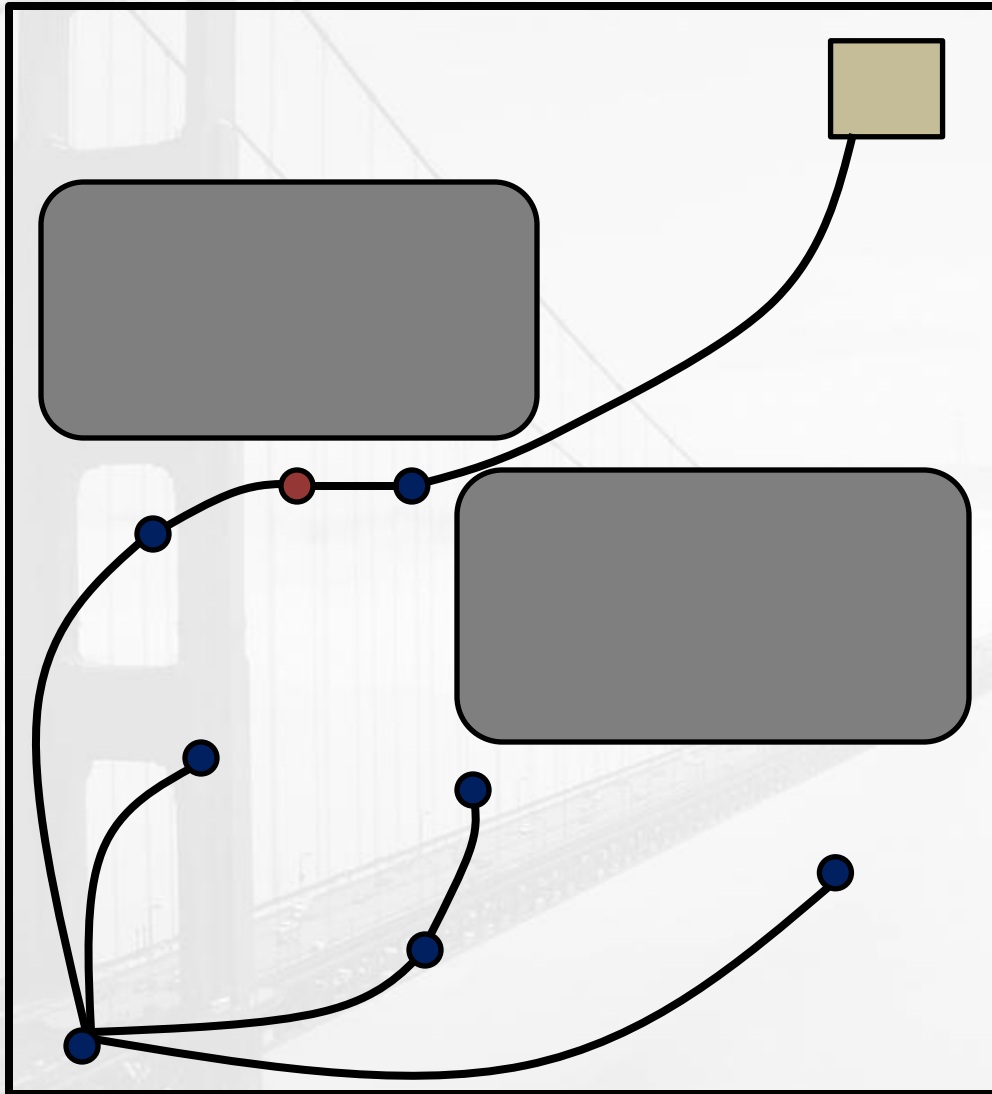
The Ball Tree Algorithm

```

1  $V \leftarrow \{x_{init}, r = 0\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3   while true do
4      $x_{new} \leftarrow \text{Sample}(i);$ 
5     if  $\text{InsideBall}(x_{new}, T)$  then
6       if  $\neg \text{CollisionFree}(x_{new})$  then
7          $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
8          $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
9       else
10        break;
11     $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
12     $\sigma \leftarrow \text{Steer}(x_{new}, x_{nearest});$ 
13    if  $\text{CollisionFree}(\sigma)$  then
14       $V.add(x_{new}, r_0);$ 
15       $E.add((x_{nearest}, x_{new}));$ 
16    else
17       $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
18 return  $T = (V, E).$ 

```

RRT* with Ball Trees



The RRT* Algorithm

```

1  $V \leftarrow \{x_{init}\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3    $x_{new} \leftarrow \text{Sample}(i);$ 
4    $X_{near} \leftarrow \text{Near}(V, x_{new});$ 
5   if  $X_{near} = \emptyset$  then
6      $X_{near} \leftarrow \text{Nearest}(V, x_{new});$ 
7    $L_{near} \leftarrow \text{PopulateSortedList}(X_{near}, x_{near});$ 
8    $x_{parent} \leftarrow \text{FindBestParent}(L_{near}, x_{new});$ 
9   if  $x_{parent} \neq \text{NULL}$  then
10     $V.add(x_{new});$ 
11     $E.add((x_{parent}, x_{new}));$ 
12     $E \leftarrow \text{RewireVertices}(E, X_{near}, x_{new});$ 
13 return  $T = (V, E).$ 

```

The Ball Tree Algorithm

```

1  $V \leftarrow \{x_{init}, r = 0\}; E \leftarrow \emptyset; T \leftarrow (V, E);$ 
2 for  $i = 1$  to  $N$  do
3   while true do
4      $x_{new} \leftarrow \text{Sample}(i);$ 
5     if  $\text{InsideBall}(x_{new}, T)$  then
6       if  $\neg \text{CollisionFree}(x_{new})$  then
7          $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
8          $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
9       else
10        break;
11     $x_{nearest} \leftarrow \text{NearestBall}(V, x_{new});$ 
12     $\sigma \leftarrow \text{Steer}(x_{new}, x_{nearest});$ 
13    if  $\text{CollisionFree}(\sigma)$  then
14       $V.add(x_{new}, r_0);$ 
15       $E.add((x_{nearest}, x_{new}));$ 
16    else
17       $\text{TrimRadius}(x_{nearest}, \|x_{new} - x_{nearest}\|);$ 
18 return  $T = (V, E).$ 

```

Results (7DOF)

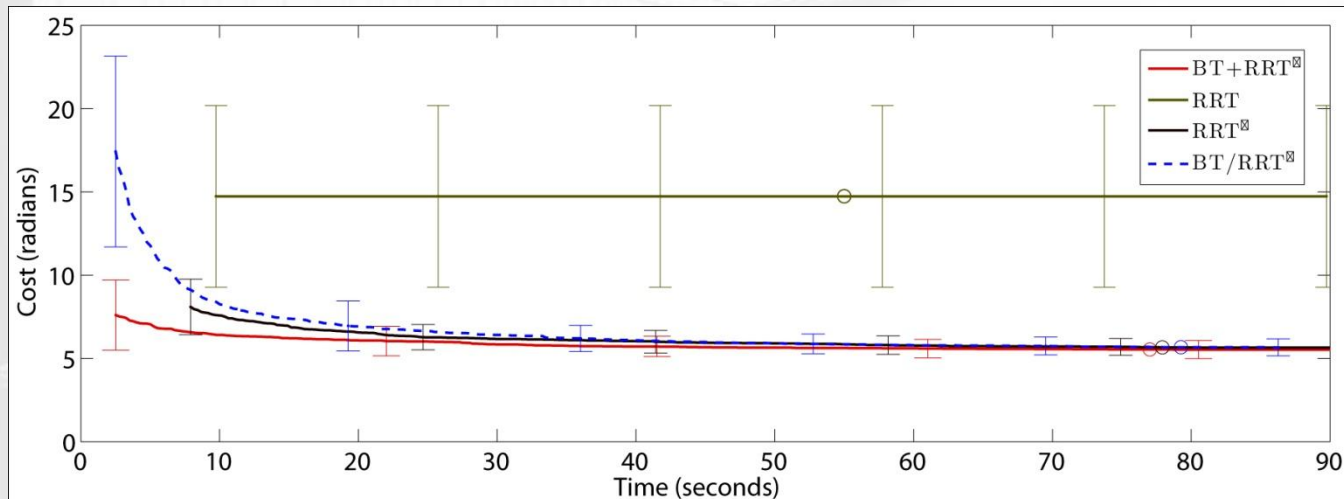


RRT

First solution time: 9.75 s
First solution cost: 14.73
Final solution time: 54.96 s
Final solution cost: 14.73

BT+RRT*

First solution time: 2.52 s
First solution cost: 7.61
Final solution time: 77.14 s
Final solution cost: 5.52



Results (7DOF)

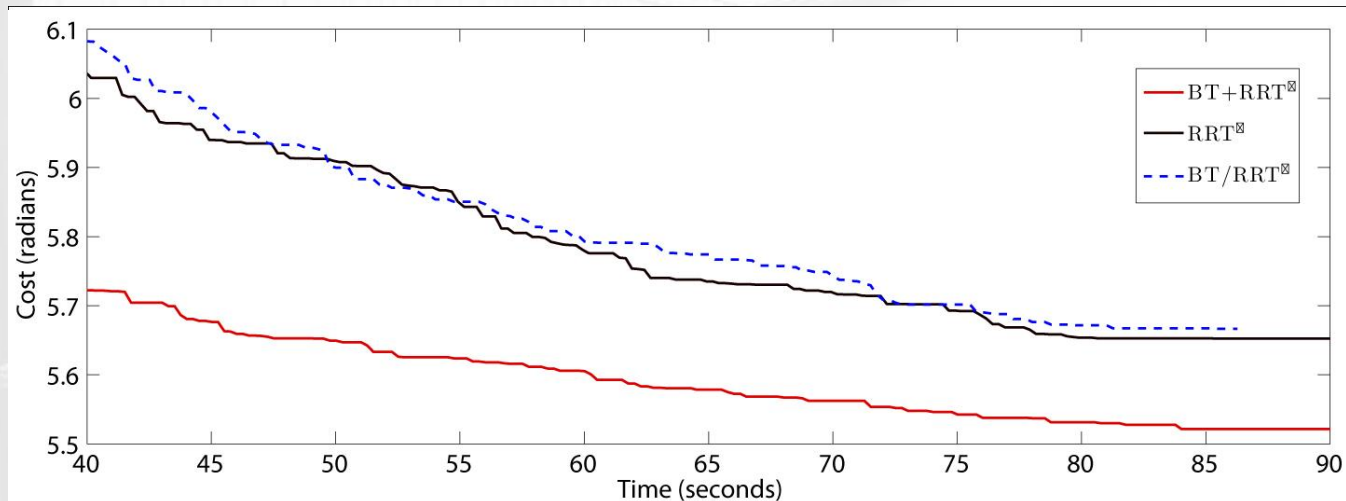


RRT

First solution time: 9.75 s
First solution cost: 14.73
Final solution time: 54.96 s
Final solution cost: 14.73

BT+RRT*

First solution time: 2.52 s
First solution cost: 7.61
Final solution time: 77.14 s
Final solution cost: 5.52



Results (7DOF)



RRT

First solution time: 9.75 s
 First solution cost: 14.73
 Final solution time: 54.96 s
 Final solution cost: 14.73

BT+RRT*

First solution time: 2.52 s
 First solution cost: 7.61
 Final solution time: 77.14 s
 Final solution cost: 5.52

TABLE I

SEVEN DEGREE OF FREEDOM MONTE CARLO RESULTS

		BT+RRT*	RRT	RRT*	BT/RRT*
Success Rate (100 runs)		100.00%	87.00%	99.00%	100.00%
First Solution	Time (s)	2.52 (3.07)	9.75 (12.52)	7.92 (10.97)	2.51 (2.48)
	Cost	7.61 (2.11)	14.73 (5.49)	8.11 (1.67)	17.99 (5.63)
Final Solution	Time (s)	77.14 (4.49)	54.96 (4.75)	77.85 (3.95)	79.21 (4.47)
	Cost	5.52 (0.53)	14.73 (5.49)	5.65 (0.50)	5.67 (0.51)
Time per Iteration (ms)		19.33 (1.13)	13.78 (1.19)	19.51 (0.99)	19.85 (1.12)

Results (12DOF)

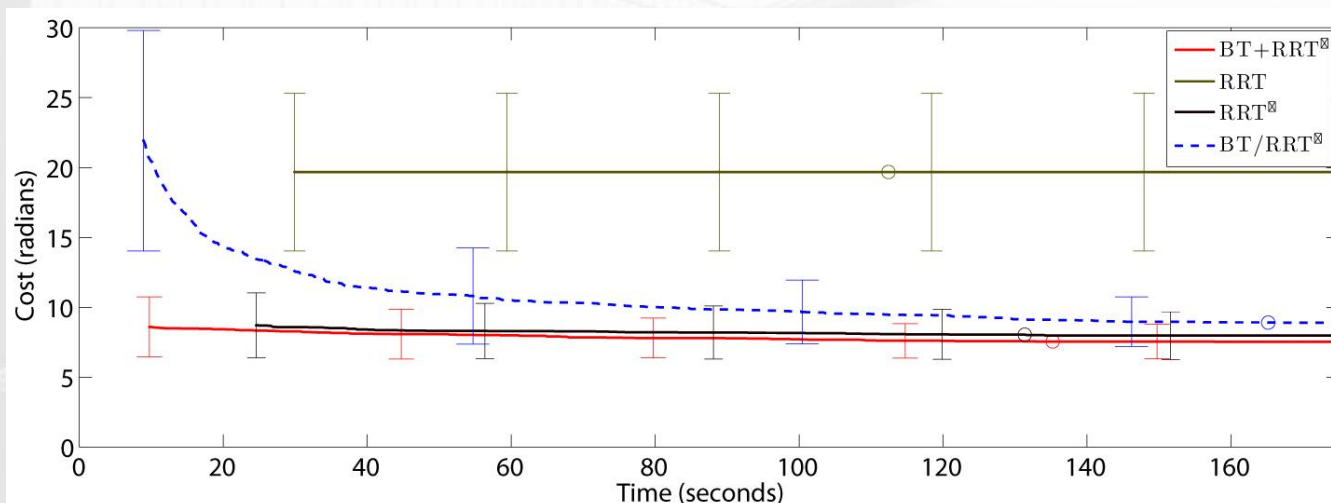


RRT

First solution time: 29.92 s
First solution cost: 19.76
Final solution time: 112.41 s
Final solution cost: 19.76

BT+RRT*

First solution time: 9.74 s
First solution cost: 8.59
Final solution time: 135.28 s
Final solution cost: 7.53



Results (12DOF)



RRT

First solution time: 29.92 s
 First solution cost: 19.76
 Final solution time: 112.41 s
 Final solution cost: 19.76

BT+RRT*

First solution time: 9.74 s
 First solution cost: 8.59
 Final solution time: 135.28 s
 Final solution cost: 7.53

TABLE II

TWELVE DEGREE OF FREEDOM MONTE CARLO RESULTS

		BT+RRT*	RRT	RRT*	BT/RRT*
Success Rate (100 runs)		100.00%	58.00%	85.00%	100.00%
First Solution	Time (s)	9.74 (12.84)	29.92 (34.05)	24.61 (32.09)	8.94 (11.06)
	Cost (rad)	8.59 (2.16)	19.76 (5.69)	8.71 (2.34)	22.13 (7.72)
Final Solution	Time (s)	135.28 (15.08)	112.41 (19.46)	131.38 (14.49)	165.28 (28.16)
	Cost (rad)	7.53 (1.21)	19.76 (5.69)	7.97 (1.71)	8.83 (1.73)
Time per Iteration (ms)		22.58 (2.52)	18.77 (3.25)	21.93 (2.42)	27.59 (4.70)

IROS 2011 Video



For more information:

Perez, A., Karaman, S., Shkolnik, A., Frazzoli E., Teller, S. and Walter, M.,
“[Asymptotically-optimal Manipulation Planning using Incremental Sampling-based Algorithms](#),” in IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2011.

http://ares.lids.mit.edu/manipulation_planning

Interactive Session (Golden Gate Room)

[The PR2 Workshop](#)

Also featured in:

[MIT News](#), [Slashdot](#), [New Scientist](#), [New England Post](#),
[Gizmag](#), [io9](#), [AzoRobotics](#) and [Indian Express](#)



IEEE/RSJ International Conference on Intelligent
Robots and Systems 2011

