**Algorithm 2：Dynamic-RRT-Connect**

**Input:** Xinit, Xgoal**,** old graph node set Gold**,** new obstacle set Onew

**Output:** path **from** Xinit to Xgoal **in** the updated map

Gnew **=** Gold

**for** each obstacle o **in** Onew do

**for** each node n **in** Gold do

**if** check\_collision**(**n, o**) then**

remove\_node\_and\_edge**(**n**,** Gnew**)**

end **if**

Ginit **=** reachable\_nodes\_and\_edges(Xinit**,** Gnew)

Ggoal **=** reachable\_nodes\_and\_edges(Xgoal**,** Gnew)

**while not** terminated **do**

Xrand **=** sample a random configuration **in** C\_free

Xnearest\_init **=** nearest neighbor to Xrand **in** Ginit

Xnew\_init **=** steer**(**Xnearest\_init**,** Xrand**)**

**if not** collision**(**Xnearest\_init**,** Xnew\_init**) and not**

obstacle\_collision**(**Xnearestt\_init**,** Xnew**) then**

add Xnew\_init **as** a node **with** edge(Xnew\_init**,** Xnearest\_init**)** to Ginit

Xnearest\_goaltoinit **=** nearest neighbor to Xnew\_init **in** Ggoal

Xprim\_goal **=** steer**(**Xnearest\_goaltoinit **,** Xnew\_init**)**

Xlast\_node = Xprim\_goal

**if not** collision**(**Xprim\_goal**,** Xnearest\_goaltoinit**) and not**

obstacle\_collision**(**Xprim\_goal**,** Xgoal**) then**

add Xprim\_goal **as** a node **with** edge (Xprim\_goal**,** Xnearest\_goaltoinit**)** to Ggoal

**while not** same\_nodes(Xlast\_node, Xnew\_init) **do**

        Xprim\_goal\_iter **=** steer**(**Xlast\_node **,** Xnew\_init**)**

**if not** collision**(**Xprim\_goal**,** Xprim\_goal\_iter**) and not**

obstacle\_collision**(**Xprim\_goal**,** Xprim\_goal\_iter**) then**

add Xprim\_goal\_iter **as** a node **with** edge(Xprim\_goal\_iter**,** Xlast\_node**)** to Ggoal

Xlast\_node = Xprim\_goal\_iter

**if** Xlast\_node **is** close to Xnew\_init **then**

add Xlast\_node **as** a node **with** edge (Xnew\_init**,** Xlast\_node**)** to Ggoal

Gold = union\_graph(Ggoal**,** Ginit)

**return** the path **from** Xinit to Xgoal

end **if**

end **if**

end **if**

**for** each obstacle o **in** Onew do

**for** each node n **in** Ginit do

**if** check\_collision**(**n**,** o**) then**

remove\_node\_and\_edge**(**n**,** Ginit**)**

end **if**

**for** each node n **in** Ggoal do

**if** check\_collision**(**n**,** o**) then**

remove\_node\_and\_edge**(**n**,** Ggoal**)**

end **if**

**if** size\_bigger(Ggoal**,** Ginit) then

swap(Ggoal**,** Ginit)

end **if**