靶场题：
这里使用暴力生成来解决，省去了看模型代码的苦恼。

```
import json
import random
import math

FIELD_SIZE = 50
OBSTACLE_RADIUS = 1.0
OBSTACLE_DIST = 3.0
START_END_DIST = 2.5
MAX_OBSTACLES = 30
```

这里按照题目约束条件来构建判断是否有效的函数

```
def dist(a, b):
    return math.hypot(a[0]-b[0], a[1]-b[1])
def is_valid(new, obstacles, start, end):
    for obs in obstacles:
        if dist((new[0], new[1]), (obs["x"], obs["y"])) < OBSTACLE_RADIUS*2 + OBSTACLE_DIST:
            return False
    if dist((new[0], new[1]), start) < OBSTACLE_RADIUS + START_END_DIST:
        return False
    if dist((new[0], new[1]), end) < OBSTACLE_RADIUS + START_END_DIST:
        return False
    return True
```

将环境随机生成

```
def generate_env():
    start = [random.uniform(2, 10), random.uniform(2, 10)]
    while True:
        end = [random.uniform(40, 48), random.uniform(40, 48)]
        if dist(start, end) >= 20:
            break
    obstacles = []
    tries = 0
    while len(obstacles) < MAX_OBSTACLES and tries < 10000:
        x = random.uniform(OBSTACLE_RADIUS, FIELD_SIZE-OBSTACLE_RADIUS)
        y = random.uniform(OBSTACLE_RADIUS, FIELD_SIZE-OBSTACLE_RADIUS)
        if is_valid((x, y), obstacles, start, end):
            obstacles.append({"x": round(x, 2), "y": round(y, 2)})
        tries += 1
    return {
        "start": [round(start[0],2), round(start[1],2)],
        "end": [round(end[0],2), round(end[1],2)],
        "obstacles": obstacles
    }
```

```
if __name__ == "__main__":
    env = generate_env()
    with open("example_env.json", "w") as f:
        json.dump(env, f, indent=2)
```

这里，我们生成器也就写好了，接下来就是不断运行这个脚本来检查 10/10 的情况什么时候出现。下面写了一个小脚本。

```
import subprocess
import sys
while True:
    subprocess.run([sys.executable, "generator.py"])
    result = subprocess.run(
        [sys.executable, "test_scoring.py", "example_env.json"],
        capture_output=True, text=True
    )
    print(result.stdout)
    for line in result.stdout.splitlines():
        if line.strip().startswith("Failures:"):
            if "10/10" in line:
                print("找到 10 次失败的障碍布局，已停止。")
                exit(0)
```