

VG101 LAB6 - TankWar

Background

Tank War is a classic computer game. In this game, player acts as a tank and they compete to survive the game. In LAB6, we're going to realize a simple version of Tank War game and write an AI for it. Through this LAB, you will learn how to build data structures in C++, and learn some basic thing in game programming and traditional AI programming.

Rules

To make your life easier, we created the following rules, which are easy to implement using C++.

- Two players battle in spare map.
- One player owns one tank.
- Tank can move one meter each turn.
- Tank can choose to move left, right or straightly each turn.

- Tank will shoot a bullet every three turns (starting from the first turn), the bullet will generate 1m ahead of the tank's position. (It keeps the direction of the tank and is generated after the tank moves in his turn, or the tank will be shoot by itself)
- Bullet will move straightly at the speed 2 meter/round.
- Tank have 5 life points.
- Once a tank is hit by a bullet, two life point will be deducted. To make your life easier, just assume that the bullet moves after tanks, and this will avoid geometry calculations. Notice that there will be three possibilities where it could hit a tank each time in your calculation.
- Once a bullet hits a tank, it will not go further but explode.

- Initially, the the size of the map is 20*20. (A player can go outside the map)
- The map will shrink by 1 block to the center every 16 turns. (Notice that the first shrink is on the 16th turn), which means map size is 20x20-18x18-...
- As long as a tank is outside of the map, one life point will be deducted each turn.

- The tank whose life points are first deducted to 0 will lose.
- If two tanks have there life points deducted to 0 at the same time, they will have a draw competition.
- When two tanks crash, the tank with high life point wins.

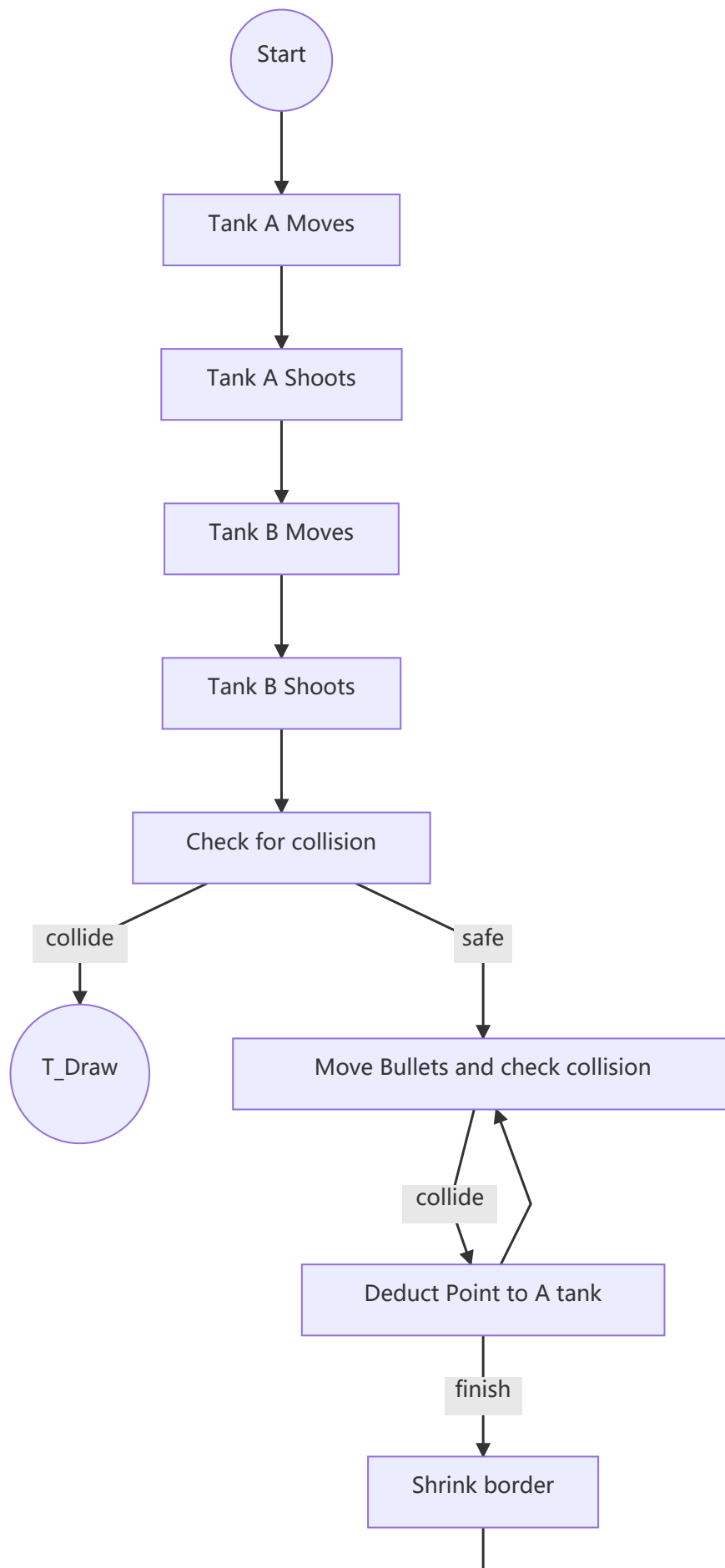
- To make your life easier, calculation will take place after two tanks move one turn.

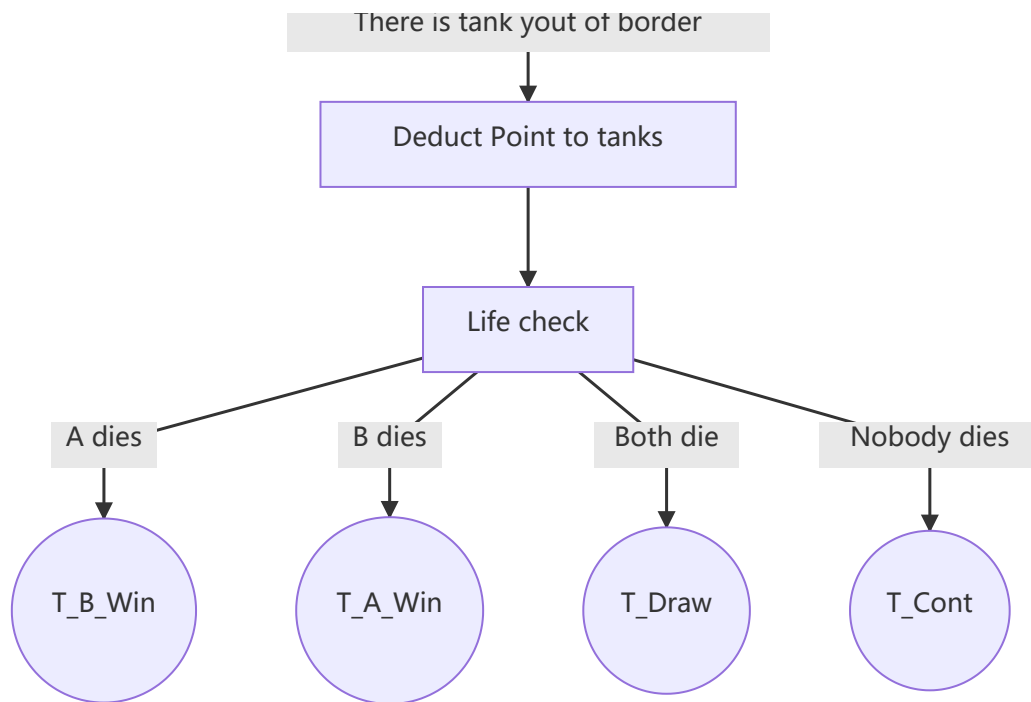
Objective I

Based on `Lab6_Related_Files` on canvas, realize a Tank War Game based on the provided structure.

To implement the Tank War game, you need to come up with a data structure so that you can store the position of tanks and bullets. With the data structure, you may then have a method so that you can update the tank and bullets each turn.

Flow chart inside one turn is shown below:





You can define your own data structure and write the code in `1ab6.cpp` but you may not modify other related files because you could only submit `1ab6.cpp` to the OJ platform later in Objective II and other files will be provided by TAs.

Example I/O module is provided in `main.cpp`.

Sample STDIN:

```


0 0 19 19 2 0
2 0
1 0
0 0
0 0
0 0
0 0
0 0
0 0
0 0
0 0
0 0
2 0
0 0
0 0
0 0
0 0
1 2
1 2
0 0
1 1
1 1
0 0
0 0
0 0
0 0
0 0
0 0
0 2
0 0
0 0
2 0

```

```
0 0
0 0
0 0
0 0
0 0
0 0
2 0
0 0
0 0
0 0
0 0
0 0
0 0
```

Sample STDOUT:

```
Tank A Wins
```

 2020-11-17 09-28-18

Objective II

Based on `Lab6_Related_Files` on canvas, write an AI for your Tank War game in `MyBrain` class.

In `MyBrain` class, an AI framework is provided, we have obtained a reference of `MyGame` object, by providing interface in your `MyGame` class, your AI could have access to the content of your current game.

Submit your AI to <http://106.15.125.102> and compete with other AIs.

Hints

1. You do not need to understand the class inheritance and virtual functions in `Tab6.h`, it simply enables you to write your own structures without `Tab6.h` being changed.
2. A simple 2D vector class is provided and you may want to use that to realize your Tank War game and AI.

3. How to build an AI?

- Decision Tree Strategy

Theoretically, you could deduce all the possible state of game several turns after by listing all the possible actions. However, because the phenomenon of exponential explosion, we cannot do so in programming.

However, by assuming that your opponent will also take the best strategy, you could reduce the possible states that the game would fall into.

Also, you could give evaluation to certain intermediate case. For example,

```
| |>|-
|>| |-
>| |T|-
-----
```

This case will certainly cause the tank to lose one life point. So this case will be evaluated higher.

- State Machine Strategy

We could set up some tasks that a tank should do in certain stage. We could let the tank to behave different in different stages so that it could have different effects.

Grading

- Objective I worth 150 pts.
- The OJ have a system ranking for each player, we will assign each player up to 150 pts according to the ranking.
- If you found bugs in the OJ platform or the game rules, we will consider to give you bonus to this lab.