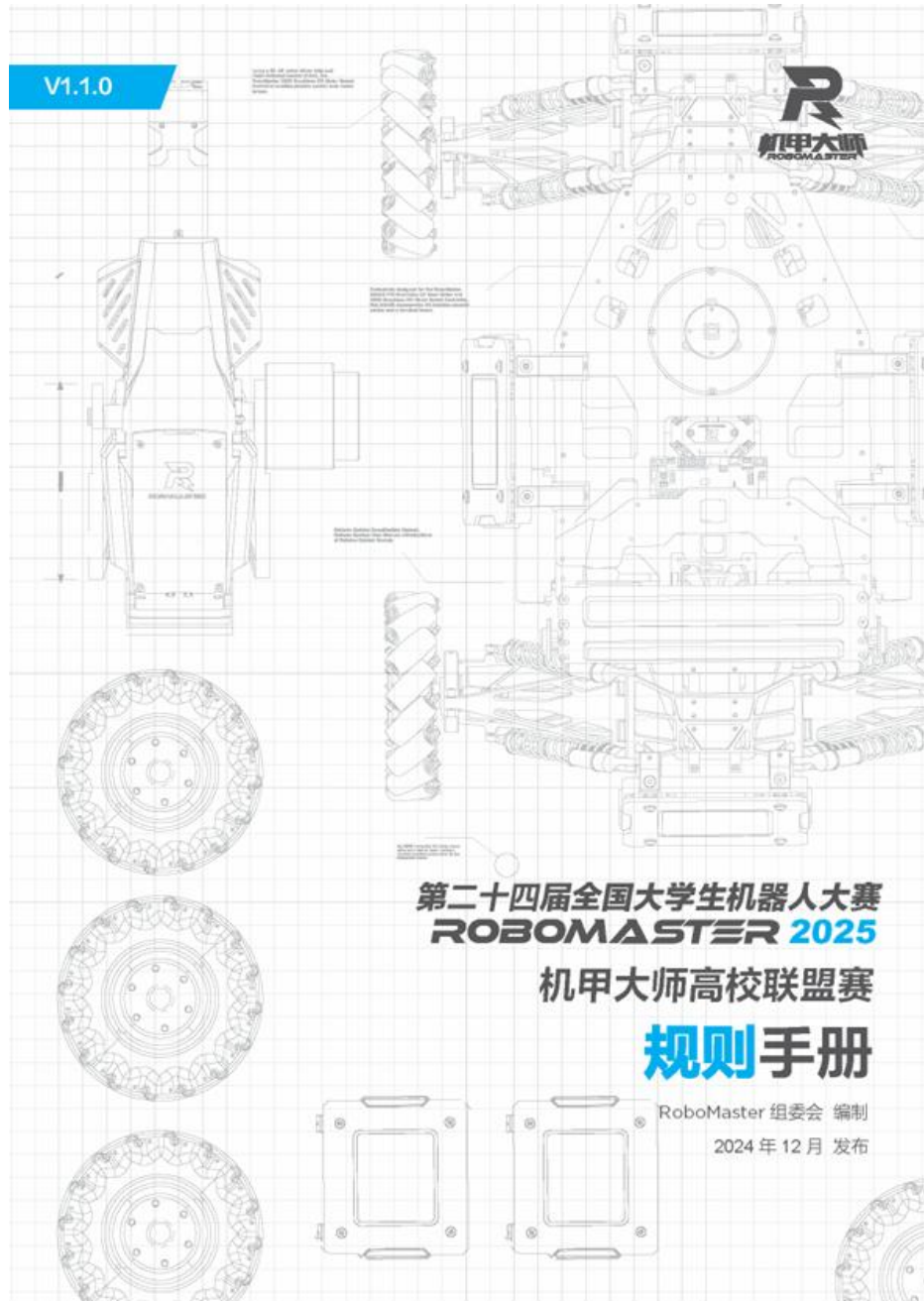


RoboMaster

This is a simulator to the robomaster



Game Overview

Map:

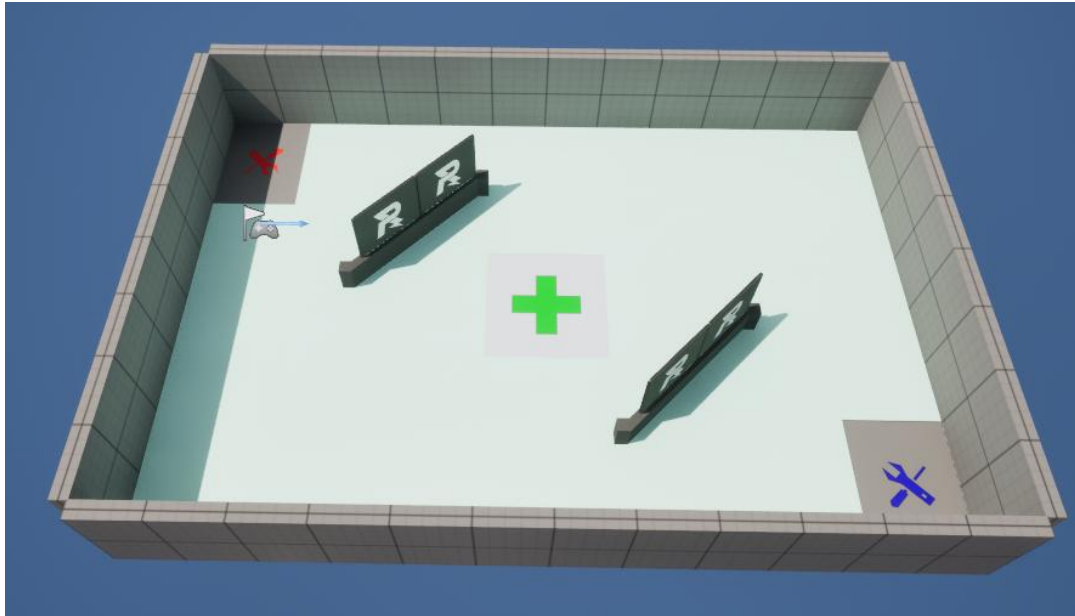


Figure 1: demo map

This game serves as a simulator, and at this stage only the most basic test arena is prepared. The arena is laid out strictly in accordance with the rulebook.

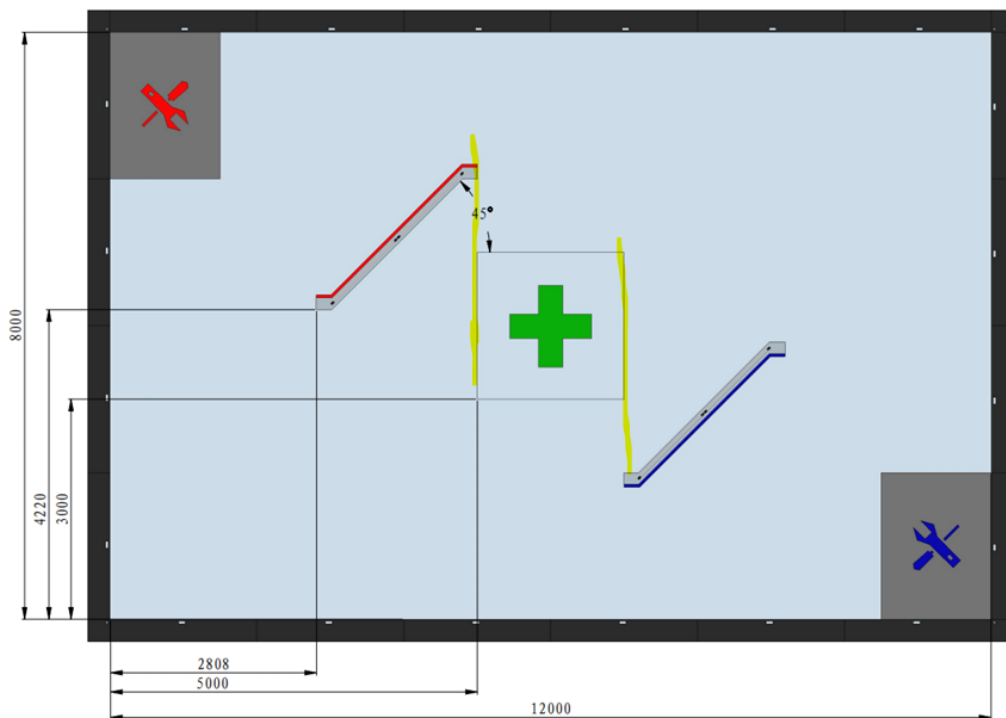


Figure2: rule book map

Model:

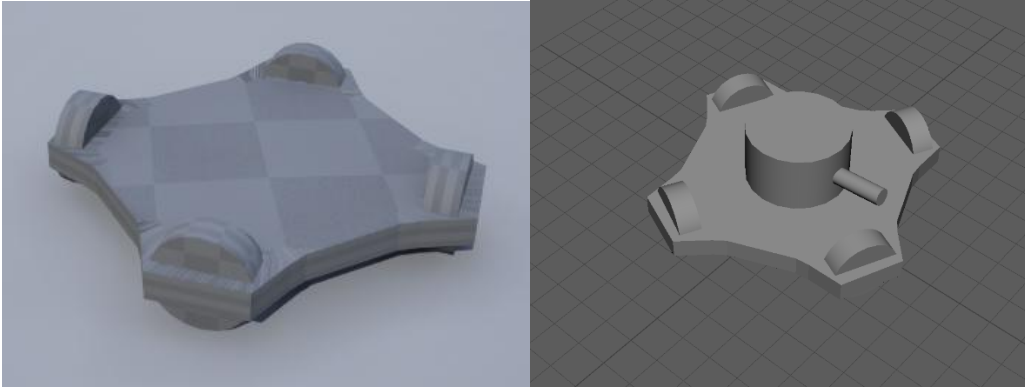


Figure 3: untextured body & turret

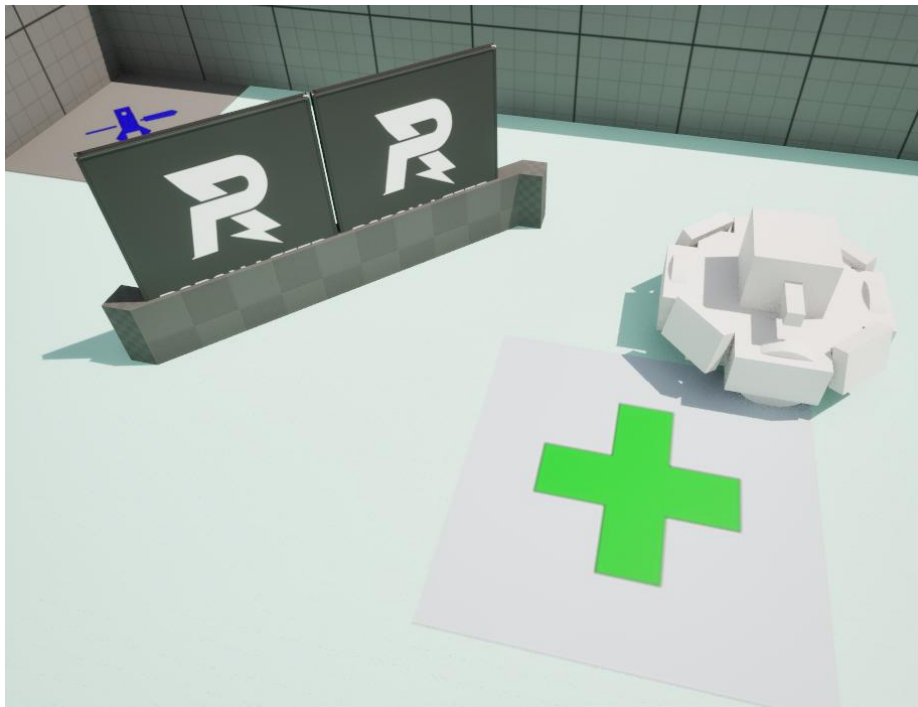


Figure 4: Combined and placed in the map

Demo used robo model. Just for placeholder.

Functional system

According to the rulebook, we have successfully completed the following features:

1. Base rotation

According to the rulebook and the competition videos, we discovered that the robot's base can rotate to achieve defensive objectives. This functionality has already been implemented (i.e., pressing a button starts the rotation, and pressing it again stops it).

2. Hitboxes

When a target area is hit, points are awarded.

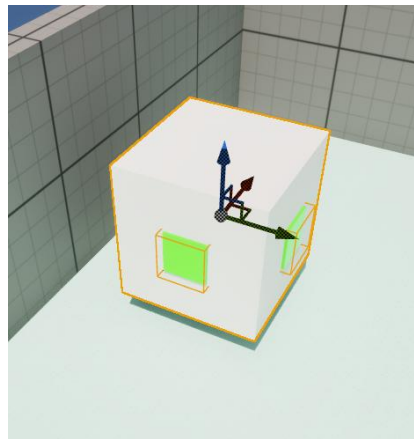


Figure 5: hitbox

3. Bullet firing / Camera movement

Pressing a button fires the bullets, and the camera movement is adjusted accordingly to ensure there is no difference from actual operation (still needs fine-tuning).

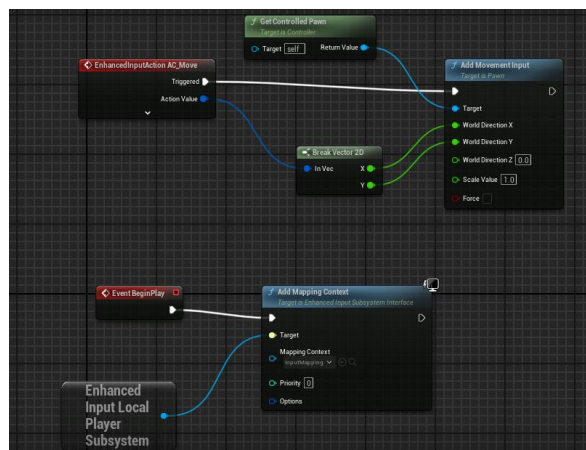


Figure 6: Camera movement(we use C++ in this part)

4. A simple health bar UI

A simple health bar UI is in place (only a basic system that reduces health when taking damage, without a death/respawn system).

5. Overheat system / Reload / Overload

The overheat system has been set up in strict accordance with the rulebook, but since it is not yet connected to any death/respawn system, it remains at the planning (blueprint) stage.

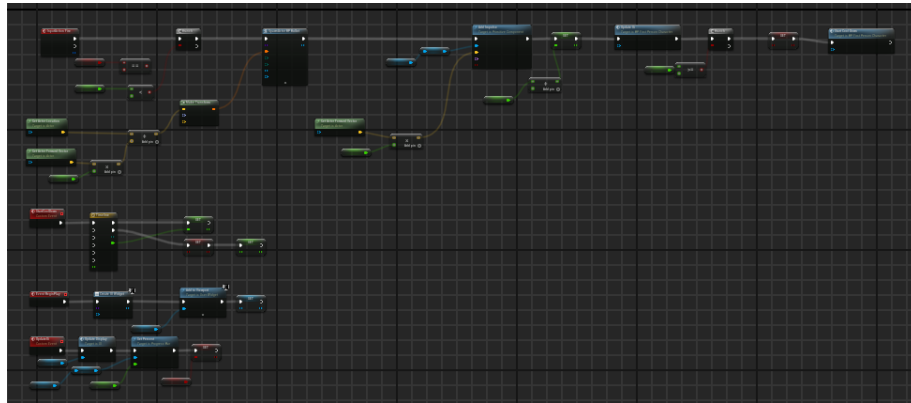


Figure 7: over heat system

Problem Facing

Of course, we've also encountered some difficulties, but we are working hard to find solutions, hoping to overcome them and complete the project.

1. **UI**

The widget system in Unreal 5 isn't easy to customize, and it doesn't integrate very well with certain design elements of this project. Many times, progress on other systems stalls because of issues with the widget system.

2. **Specific Data Adaptation**

We don't know the exact data for machine rotation; currently, these values are placeholders. In future iterations, we may make key parameters—such as rotation speed and movement speed—configurable so that users can adjust them as needed.

3. **Code Issues**

These are straightforward technical challenges that we can only tackle step by step as they arise.