Iron and Feather

(Working title)

Urs Dönni
Martin Seiler
Julian Tschannen

March 17, 2008

Contents

1 Prototypes

We built two prototypes which show the basic working of the game. The prototypes are meant to lay a foundation which can be reused for the actual game, therefore they are both using *XNA*.

2 Game Prototype

2.1 Purpose

The game prototype shows how the game will be played. It's main focus is on the steering of the submarines. Since we intend to have ships with a turret on them, the player has to use both sticks for steering. One is for steering the submarine, and one is for using the turret.

The prototype implementation has two different options for steering the submarine body:

1. **Rotation:** This option let's the player rotate with the left-right movement of the stick and the move forward and backward with the up-down movement.
2. **Direction:** With this option, the submarine drives into the direction which the player moves the stick to. The submarine does a rotation on the spot until headed to the desired direction and then drives forward.

For the steering of the turret, four different options are implemented:

1. **Rotation:** The left-right movement of the stick changes the rotation of the turret relative to the body.
2. **Stable rotation:** In this mode, when the body rotates, the turret keeps it's heading relative to the playfield. The left-right movement of the stick changes the rotation of the turret.
3. **Free cursor:** The player uses the stick to move a cursor on the playfield. The turret is always headed towards the cursor.
4. **Restricted cursor:** In this mode, the cursor is restricted to a limited region around the submarine.

To simulate the game behaviour, shooting from the turret and from the body is also possible. This can be used to test how difficult aiming will be with the different steering alternatives.

In addition, the two submarines on the prototype field have different physical properties. The turning rate and the velocity are different. These values have a great effect on the gameplay and can be changed in the prototype to have a rough idea how they influence the gameplay.

2.2 Outcome

Insight given by this prototype:

- For steering the submarine body, the *direction* mode has proven superior over the *rotation* mode.
- Aiming in direction of the submarine is difficult. A help should be offered, for example a laser-pointer or auto-aim.
- Aiming with the turret will be difficult. The *free cursor* mode is a good choice. The best option may be player-dependent, so a setting could be offered to change this.
- Linking a gamepad to a submarine was pretty simple. The multiplayer part should therefore be easy to implement. The used concept naturally extends to artificial players which will help implementing them as well.

In addition, the source code of the prototype has been a great help to start implementing the actual game. After some cleanup, several parts have already been reused.

2.3 Screenshots

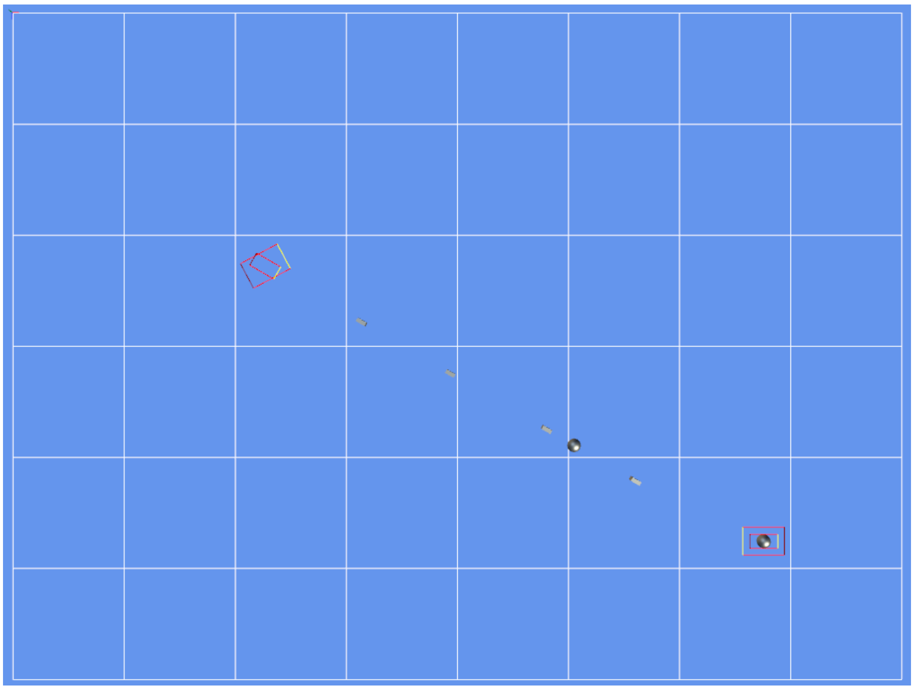

Figure 1: Prototype playfield. Two gamepads can be used to move the two submarines

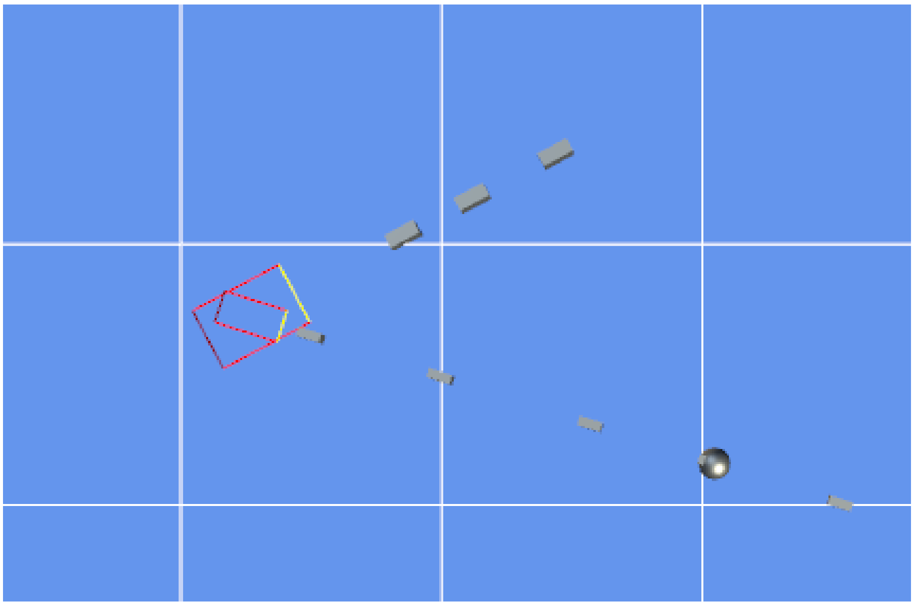

Figure 2: This screenshot shows the two weapon types: in the direction of the submarine or in the direction of the turret.

3 Fluid Prototype

3.1 Purpose

Since the game will be played under water, the fluid simulation is a vital part of the gameplay. Therefore we built a prototype of a fluid simulation. The grid-based simulation allows for arbitrary boundaries.

3.2 Outcome

The fluid simulation exposed a major problem: performance. The simulation on a 50 x 50 is already problematic on the Xbox. Since the planned grid resolution is 128 x 96, this is clearly a problem.

For this reason, we changed the focus of the fluid simulation to the GPU. Since code on the GPU is the only native code we can execute, the fluid simulation will be done on the graphics card.

3.3 Screenshots

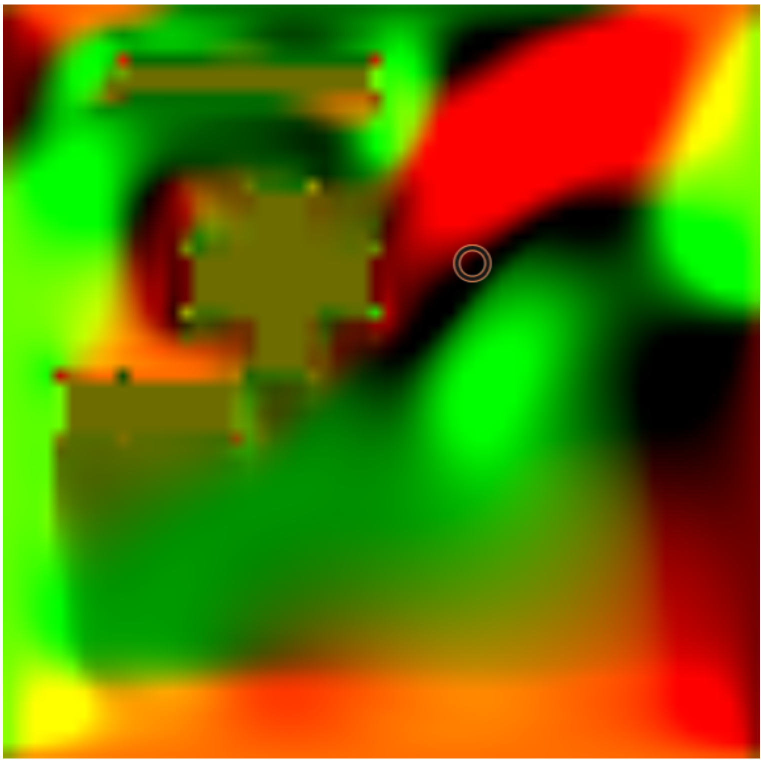

Figure 3: The velocity field of the fluid simulation

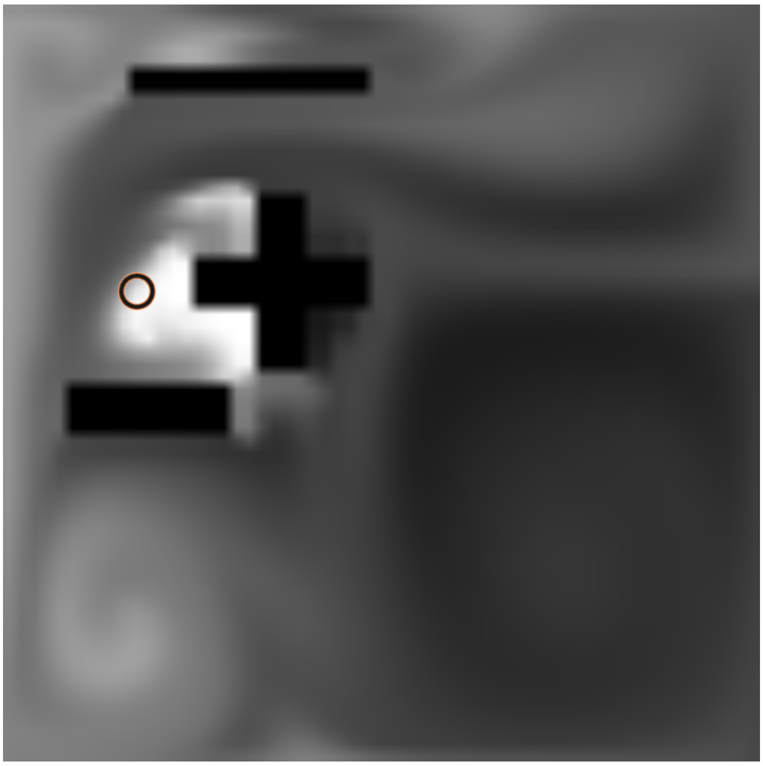

Figure 4: The density field of the fluid simulation

4 Development Status

Since the implementation of the prototpyes was done with *XNA*, we can reuse parts of the code. This contains the input handling and the basic graphics. For the basic gameplay to work, collision detection is still missing.