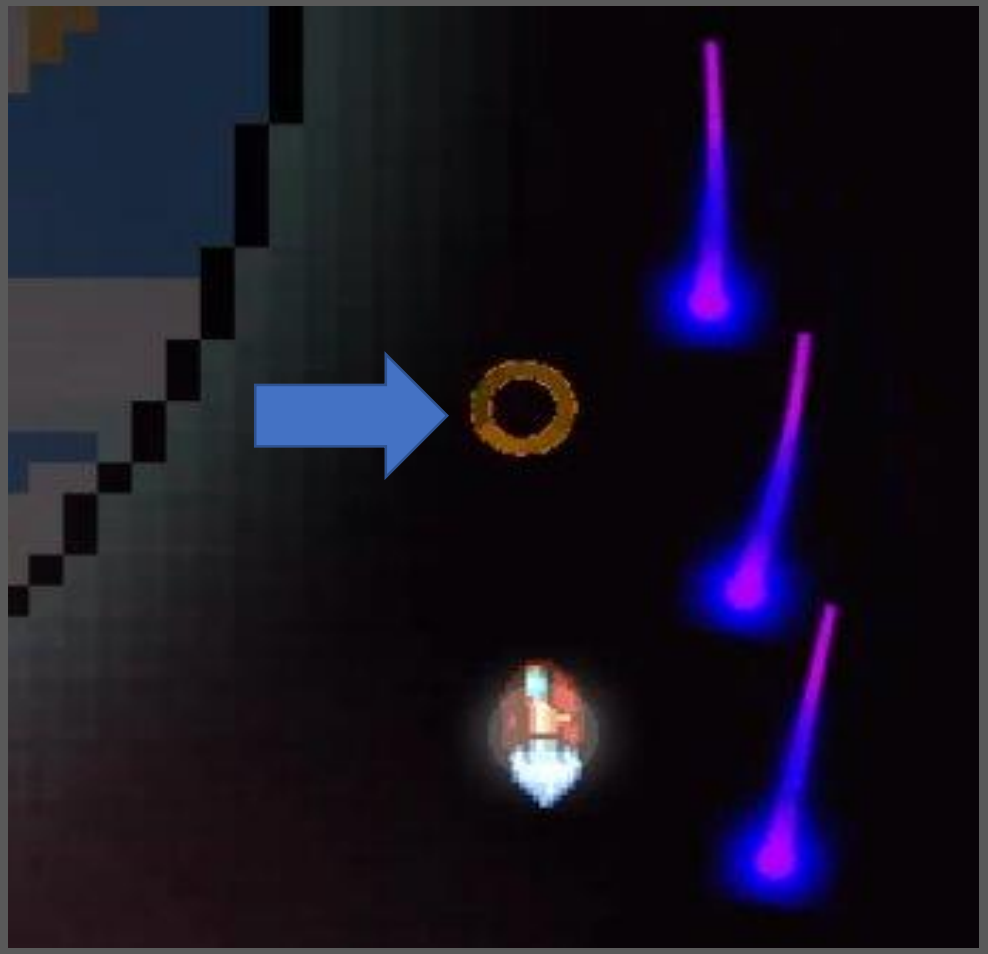
「誰にでも楽しんでもらえる対戦シューティング」を実現させるためにこだわったポイント

照準

自機から離れる最大距離を決め、
斜辺をとり半径を掛け算して決めています。
自機が動いても照準はついてきます。

```
float maxlen = 100.0f;//照準が自機から離れる最大距離 1
//auto pPos1 = Player;
auto diffX = reti_.pos.x - pos_.x;//行った先から行った元
auto diffY = reti_.pos.y - pos_.y;
auto dis  = hypotf(diffX, diffY); // 斜辺7

if (dis > maxlen){
    //プレイヤーから照準への方向
    Vector2 diff = reti_.pos;
    diff.x -= pos_.x;
    diff.y -= pos_.y;
    diff.x /= dis;
    diff.y /= dis;
    auto dirV3 = diff;
    // 方向に対して、円の半径を掛け算し、
   //プレイヤーからの移動量を取る(相対座標となる)
    Vector2 movePow =
    {
        static_cast<int>(roundf(dirV3.x * maxlen)),
        static_cast<int>(roundf(dirV3.y * maxlen))
    };
    //プレイヤー座標に移動量を加えて、
  // 照準の座標を更新する
    Vector2 newPos = pos_;
    newPos.x += movePow.x;
    newPos.y += movePow.y;
    reti_.pos = newPos;
}
```



ホーミング弾1/2

狙って撃つのが慣れない人でも楽しめて、
慣れてくると戦略に使えるホーミング弾。

一定の時間はホーミングするようにして
います。

```
    //少しの間だけホーミングする
    if (b.count <= 40)
    {
        HomingPos(false);
        b.vel = (b.vel + (homingPos - b.pos).Normalized()).Normalized() * 12.0f;
        b.count++;
    }
    if (b.pos.x + 6 < 0 || b.pos.x - 6 > 1024 ||
        b.pos.y - 6 < 0 || b.pos.y - 6 > 768) {
        b.isActive = false;
        b.history.clear();
    }
    b.pos += b.vel;
}
```

ホーミング弾2/2

弾の座標をhistoryとして 毎回
取っておき、一定数たまり次第
破棄しています

```
108     for (auto& b : bullets) {
109         if (!b.isActive) {
110             b.count = 0;
111             continue;
112         }
113         b.history.push_front(b.pos);
114         //7個たまったら一番古いものを捨てる
115         if (b.history.size() > 5) {
116             b.history.pop_back();
117         }
```

新しい順に太く古いものは細くしていき
軌跡を表現しています

```
        float thickness = 5.0;
        for (const auto& h : b.history) {
            DrawLineAA(pos.x, pos.y, h.x, h.y, 0xffffff, thickness);
            thickness *= 0.9;
            pos = h;
        }
```

軌跡の表現