

# Adding a Bouncing Sphere Mode to the Lucifer Cannon in Tremulous

The first goal for me was to avoid the need of binding a key to switch the weapon to the new mode; thanks to the fact that the normal fire mode of the Lucifer Cannon charges a sphere, we're able to determine at any time if the player is holding down the left mouse button AND the right one!

First of all, we need to modify the LCChargeFire function inside g\_weapon.c:

```
void LCChargeFire( gentity_t *ent, qboolean secondary )
{
    gentity_t *m
    ...

    // Bouncing mode
    if ((ent->client->ps.stats[STAT_MISC]) && (secondary))
    {
        // Create the energy sphere
        m = fire_luciferCannon(ent, muzzle, forward, ent->client->ps.stats[STAT_MISC],
                               LCANNON_RADIUS);
        ent->client->ps.weaponTime = LCANNON_CHARGEREREPEAT;

        // At every bounce, the sphere will lose half of its current speed
        m->s.eFlags |= EF_BOUNCE_HALF;
        m->iBounceLimit = LUCIFER_BOUNCES_LIMIT;

        // Set half the damage of the normal fully-charged sphere
        m->damage /= 2;

        // Reset the lucifer charge
        ent->client->ps.stats[STAT_MISC] = 0;

        // Set the fire delay time
        ent->client->ps.weaponTime = LCANNON_CHARGEREREPEAT;

        return;
    }

    if( secondary )
    {
        m = fire_luciferCannon(ent, muzzle, forward, LCANNON_SECONDARY_DAMAGE,
                               LCANNON_SECONDARY_RADIUS);
        ent->client->ps.weaponTime = LCANNON_REPEAT;
    }
    else
    {
        m = fire_luciferCannon(ent, muzzle, forward, ent->client->ps.stats[ STAT_MISC ],
                               LCANNON_RADIUS);
        ent->client->ps.weaponTime = LCANNON_CHARGEREREPEAT;
    }
    ent->client->ps.stats[ STAT_MISC ] = 0;
}
```

Now, you have to define the LUCIFER\_BOUNCES\_LIMIT constant (i set it to 2)

```
#define LUCIFER_BOUNCES_LIMIT 2
```

In order to count the number of times an energy sphere bounced on walls, we need to add a counter in the `gentity_s` structure (`g_local.h`):

```
struct gentity_s
{
    entityState_t s;
    entityShared_t r;
    ...
    ...

    int iBounceCount;
    int iBounceLimit;    // This is not really needed now, because we use a constant to set the limit, but
                        // as you'll soon see, it can be handy for further modifications.
};
```

Now that we have the the variables, we have to handle the impacts between the energy spheres and the walls. Open `g_missile.c` and search for `G_MissileImpact`:

```
void G_MissileImpact(gentity_t *ent, trace_t *trace)
{
    gentity_t *other, *attacker;
    qboolean returnAfterDamage = qfalse;
    vec3_t dir;

    other = &g_entities[ trace->entityNum ];
    attacker = &g_entities[ ent->r.ownerNum ];

    if(!other->takedamage && (ent->s.eFlags & (EF_BOUNCE | EF_BOUNCE_HALF)) &&
        (ent->iBounceCount < ent->iBounceLimit))
    {
        G_BounceMissile(ent, trace);

        ent->iBounceCount += 1;

        if (!(ent->s.eFlags & EF_NO_BOUNCE_SOUND))
            G_AddEvent(ent, EV_GRENADE_BOUNCE, 0);

        return;
    }

    if (!strcmp(ent->classname, "grenade"))
        ...
}
```

The mod is ready! Equip the Lucifer Cannon and then charge a sphere by holding down the left mouse button; when it's ready, press the right mouse button without releasing the left one!

Do you remember the `iBounceLimit` variable `i` set in the structure? Since it's there, we can dynamically set the limit based on the current sphere power! Here's an example:

```
#define LUCIFER_BMODIFIER 1
m->iBounceLimit = ( LUCIFER_BMODIFIER * ent->client->ps.stats[STAT_MISC]);
```

A fully charged sphere will be able to bounce on solids for 10 times!

And now, the most important part: enjoy your new modification!