Bauhaus-
Universität
Weimar

Stretching Time

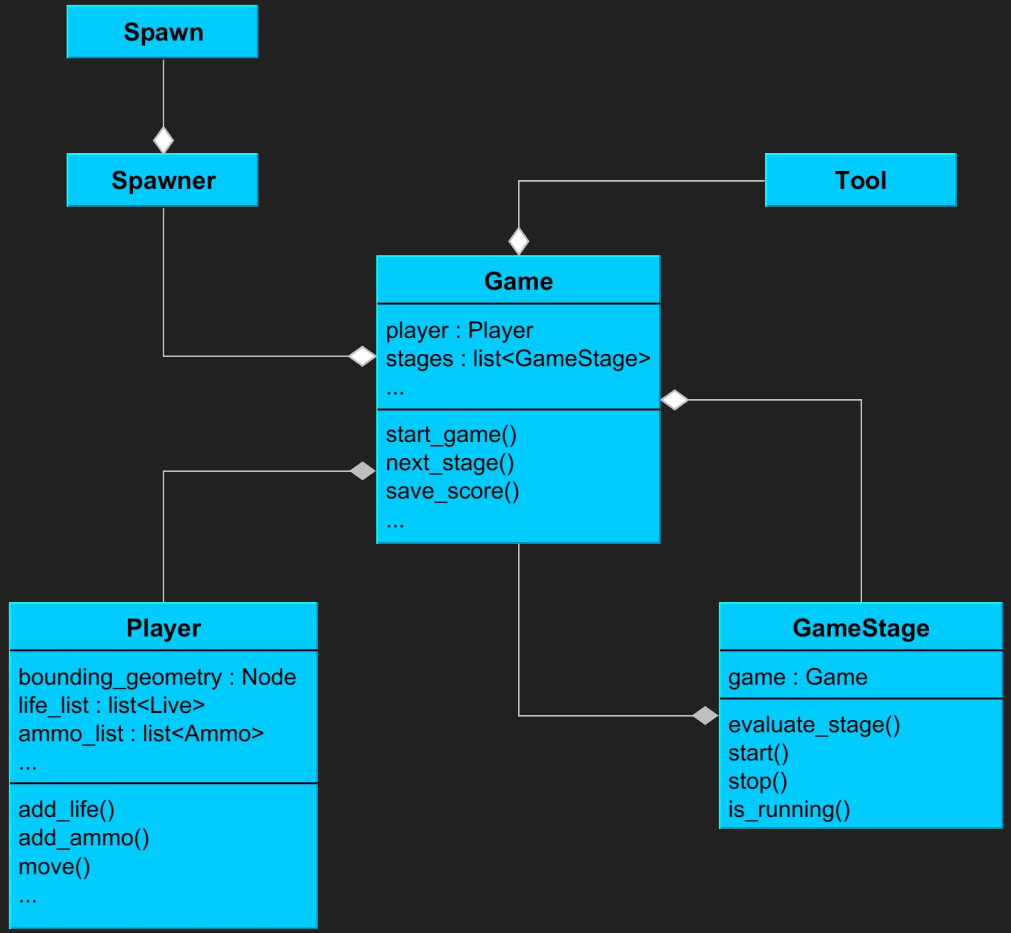
by Johannes Hartmann & Sebastian Stickert

Motivation

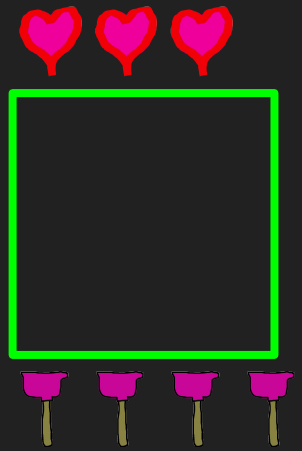
source: https://www.youtube.com/watch?v=n5lu4Gyb1KM

Let’s take a short look at our game

(see PowerWall)


Spawn
Spawner
Tool
Game
player : Player
stages : list<GameStage>
...
start_game()
next_stage()
save_score()
...
Player
bounding_geometry : Node
life_list : list<Live>
ammo_list : list<Ammo>
...
add_life()
add_ammo()
move()
...
GameStage
game : Game
evaluate_stage()
start()
stop()
is_running()

Input Mapping



Player

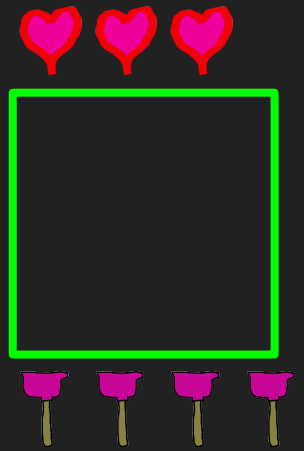
[1]

Tool

[1] source: https://www.google.de/search?q=hand&source=lnms&tbm=isch&sa=X&ved=0ahUKEwiykb2x7_HUAhVDJ1AKHT5IBQcQ_AUIBigB&biw=1536&bih=755#imgrc=HKJolux7KjqzwM:

Input Mapping - Player

```
# in Game.my_constructor()
p_offset = avango.gua.make_scale_mat(0.1,0.1,0.1)
self.player = Player()
self.player.my_constructor(
        PARENT_NODE = self.screen_node,
        OFFSET_MAT = p_offset,
        MAX_LIFE_COUNT=3
)

def _move_player(self):
        ''' Moves player by offset between this and last frame's head position. '''
        head_m = self.head_node.WorldTransform.value
        pos = head_m.get_translate()
        pos.z = 0.0
        self.player.set_transform(pos)
```



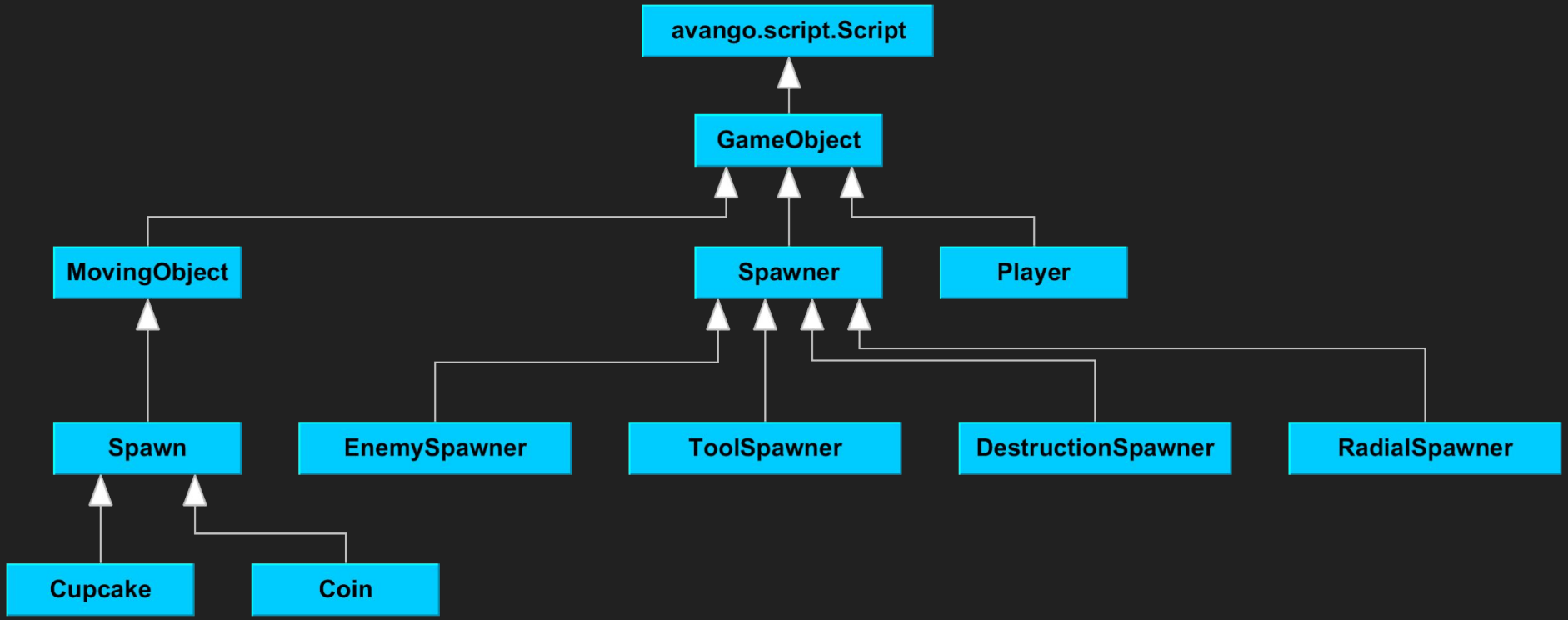
Input Mapping - Tool

```
@field_has_changed(sf_hand_mat)
def sf_hand_mat_changed(self):
        t = self.sf_hand_mat.value.get_translate() * 0.5
        t.z = min(0.0, t.z-1.0)
        m_t = avango.gua.make_trans_mat(t)
        m_r = avango.gua.make_rot_mat(self.sf_hand_mat.value.get_rotate())
        m = m_t * m_r
        self.bounding_geometry.Transform.value = m * self._offset_mat
```

Input Mapping - Time Stretching

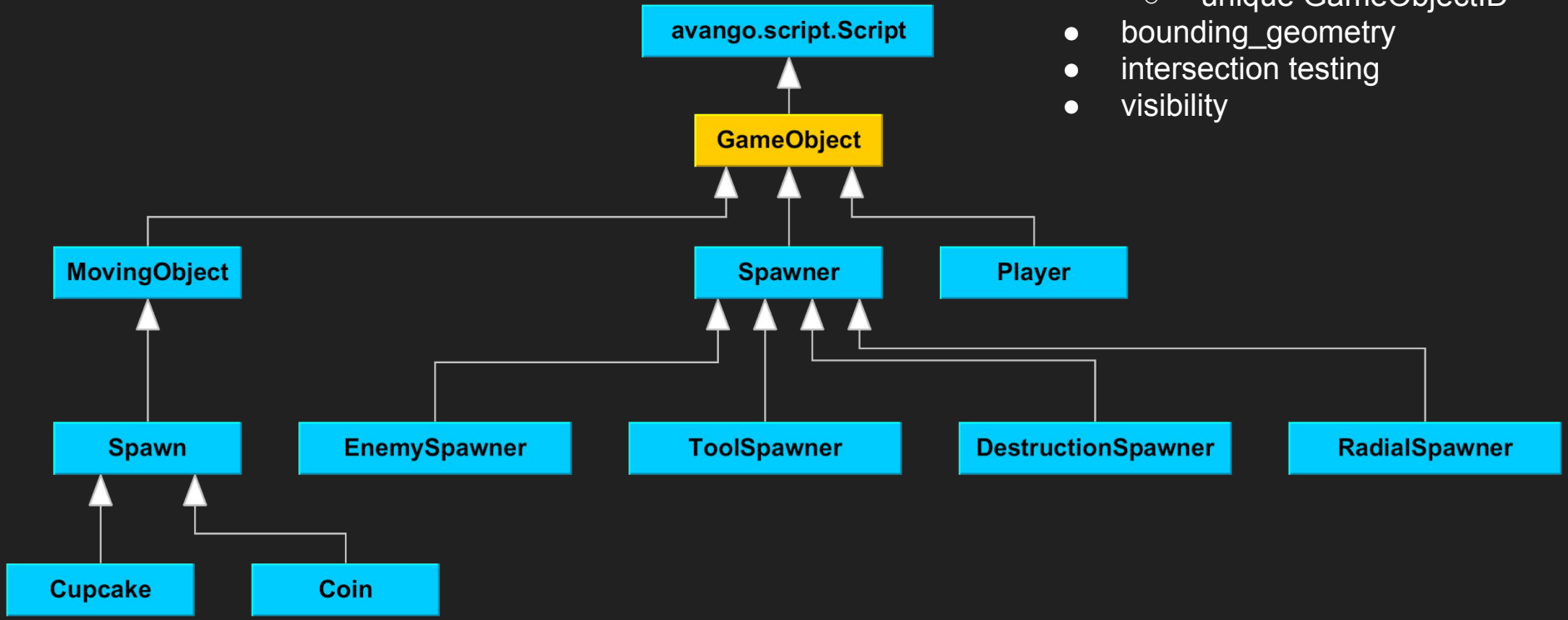
```
def _calc_time_stretch(self):
        ''' calculates a global factor for all time based animations. '''
        head_pos = self._game.head_node.WorldTransform.value.get_translate()
        self._game.head_pos_buffer.append(head_pos)
        if len(self._game.head_pos_buffer) == 10:
                self._game.head_pos_buffer.pop(0)
        lib.game.Globals.TIME_FACTOR = self._game.debug_stretch_factor * self._calc_velocity_factor()

def _calc_velocity_factor(self):
        ''' calculates a velocity value from average movement speed of the head_node. '''
        sum_velocity = sum([(b-a).length() for a,b in zip(self._game.head_pos_buffer, self._game.head_pos_buffer[1:])])
        velocity_avg = sum_velocity / len(self._game.head_pos_buffer)
        return velocity_avg / self._game.velocity_norm
```

GameObject hierarchy



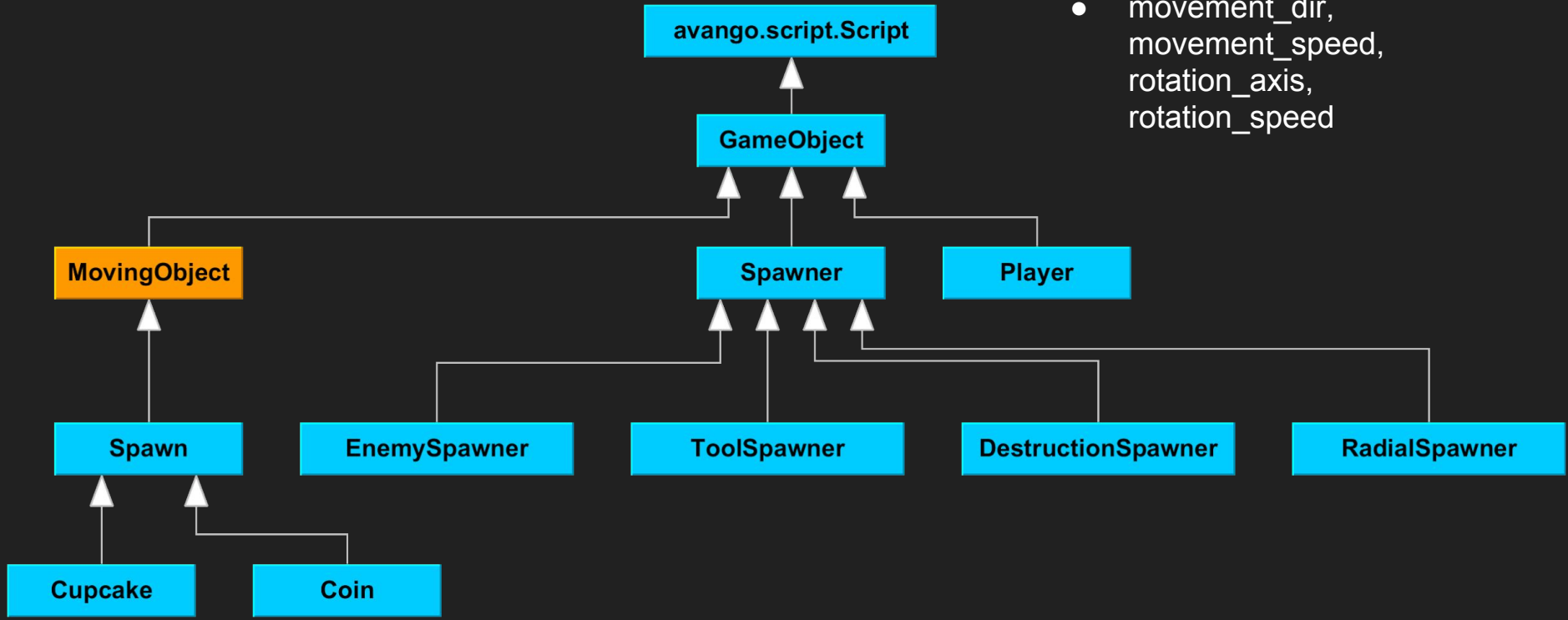
GameObject hierarchy

- metadata for game entities
  - unique GameObjectID
- bounding_geometry
- intersection testing
- visibility



GameObject hierarchy

- framerate independent, time stretched movement
- movement_dir, movement_speed, rotation_axis, rotation_speed



MovingObject

```
def _move(self):
        ''' calculates movement update. '''
        # compute translation
        t = self.movement_dir * self.movement_speed * \
            lib.game.Globals.TIME_FACTOR * self._get_fps_scale()

        # compute rotation angle
        a = self.rotation_speed * \
             lib.game.Globals.TIME_FACTOR * self._get_fps_scale()

        # create transformation matrix (global)
        m = avango.gua.make_trans_mat(t.x, t.y, t.z) * \
              self.bounding_geometry.WorldTransform.value * \
              avango.gua.make_rot_mat(a, self.rotation_axis.x, self.rotation_axis.y, self.rotation_axis.z)

        # bounding_geometry.Transform.value = inv(Parent.WorldTransform.value) * m
```

Tool

- We use a generic tracked tool in hand
- As mentioned before, the mapping is not a 1:1 mapping
- Moving the tool does *not* stretch time
- Apart from movement, the tool’s button press is the only user interaction

SwordDyrion

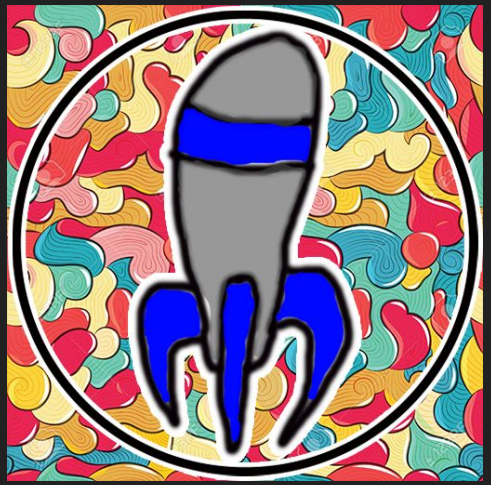
- Our interpretation of melee combat
- Bounding box intersection to destroy cupcakes
- Proximity test to collect power-ups

PewPewGun

- Shoots projectile along pointing direction
- RadialSpawner assures projectiles vanish
- Proximity test to collect power-ups

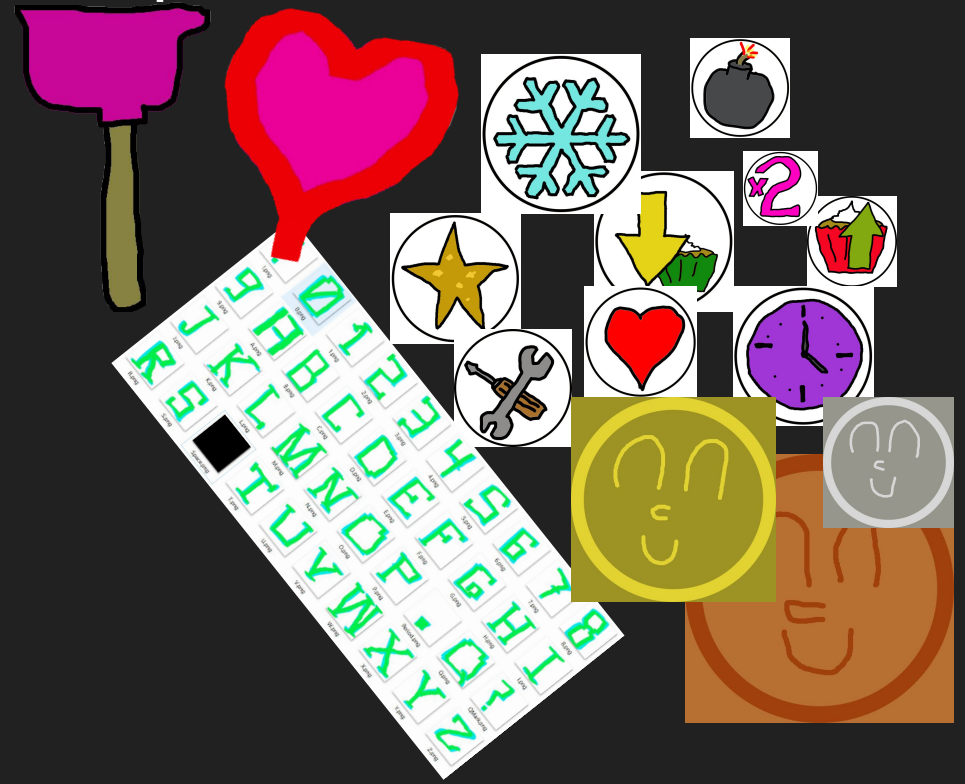
HomingGun

- Can only be fired if aimed at a cupcake target
  - Selection via cone selection
    - Ratio of distances used for disambiguation
- Shot projectile adjusts movement vector
  - Fraction of angle between movement dir and to-target dir
- Selection visualization for user feedback



Cupcakes, Plungers, Powerups and Schmeckles

- attention to detail
- creating an enjoyable virtual environment



Future Work

- more tools and powerups
  - magnet gun
  - ...
- improve time stretch
- more reasonable menu interaction before and after game start
- more interactive enemies
- ...

TIME TO SET YOUR SCORE!