Multimedia-Programmierung
Übung 9

Ludwig-Maximilians-Universität München

Sommersemester 2009

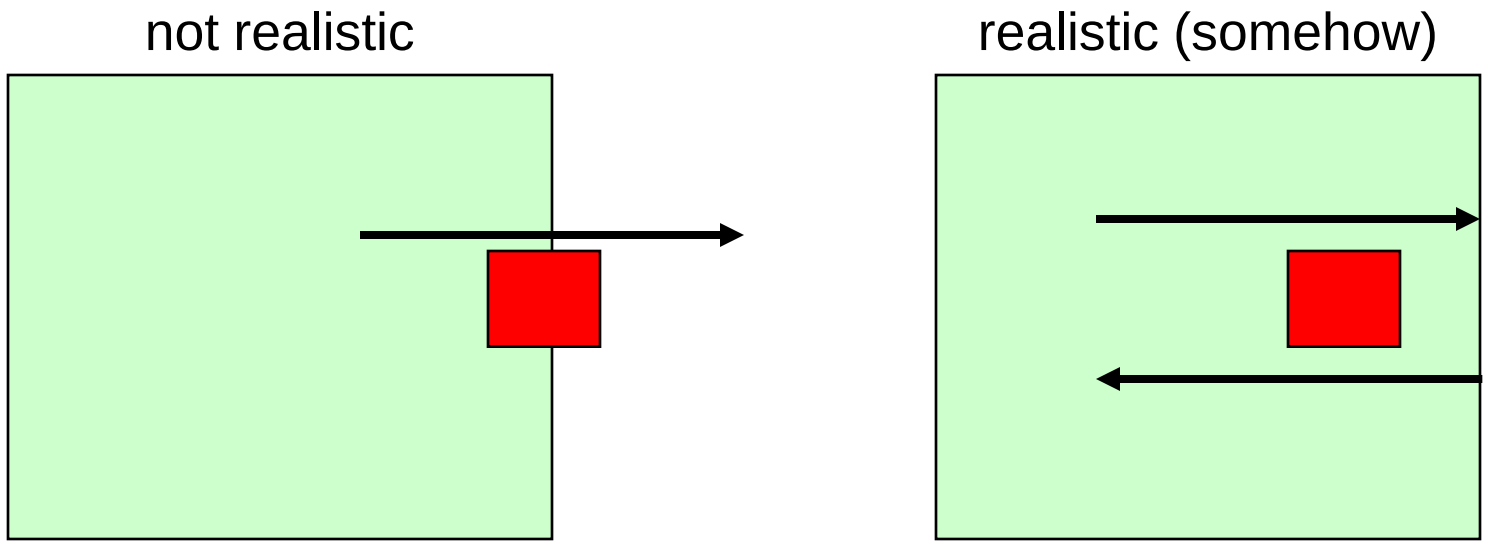
Today

- More on physics

Physics

How logical behaviour improves usability

- Users have specific expectations
- For example, if something hits a wall it should bounce or create some damage
- Adding physics to applications helps to improve usability



Physics

Examples I - Bumptop

- A physically enhanced Windows desktop

©bumptop.com

Physics

Examples II - Physics and Microsoft Surface

- Allows physically correct interaction with a tabletop device

Wilson, A. D., Izadi, S., Hilliges, O., Garcia-Mendoza, A., and Kirk, D. 2008. Bringing physics to the surface. In Proceedings of the 21st Annual ACM Symposium on User interface Software and Technology (Monterey, CA, USA, October 19 - 22, 2008). UIST '08. ACM, New York, NY, 67-76.

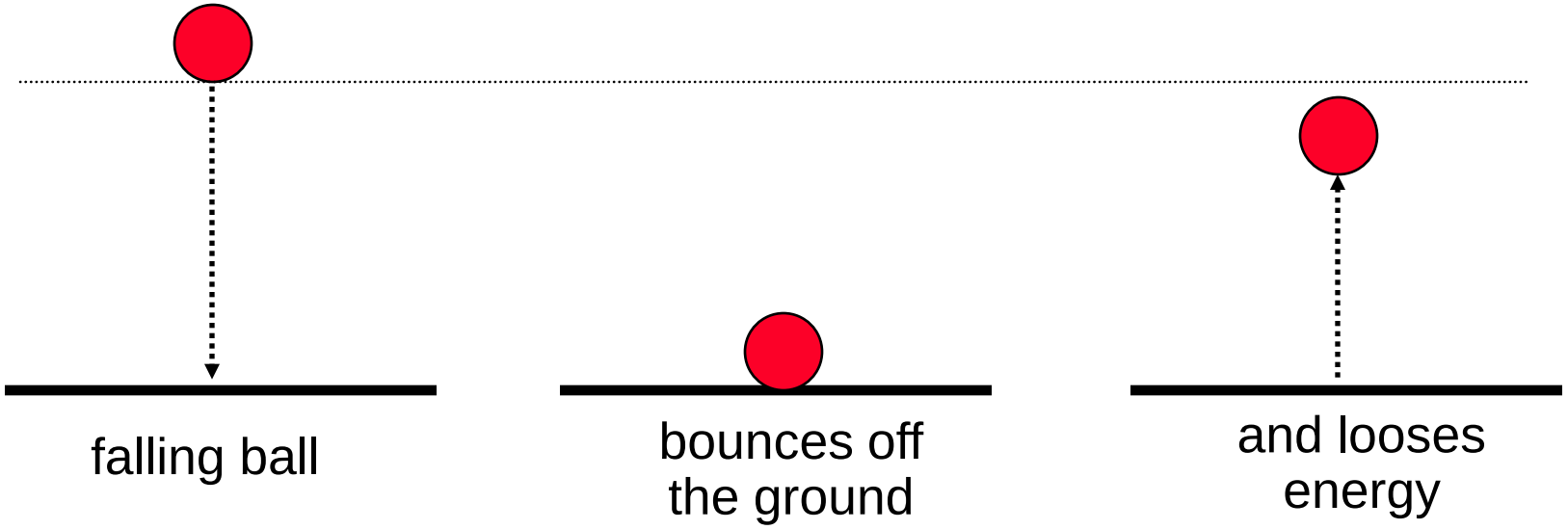
Programming Physics

- Frameworks, APIs, development tools etc. often offer physics engines (e.g. 3D game engines, Interpolators in Flash)

- In Python, **WE** do the physics!!

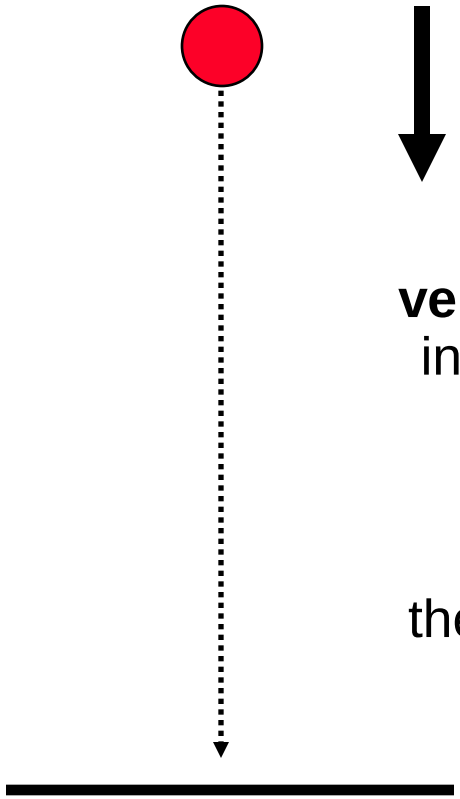
Bouncing Ball Example 1

- Let’s make a ball bounce in a realistic way
- 1. We need a concept:



Bouncing Ball Example 2

- 2. What makes the ball fall and bounce?



gravity makes the ball fall

velocity depends on gravity and increases/decreases over time

the material of the ball influences how far it will **bounce** back

Bouncing Ball Example 3

```
class Ball(pygame.sprite.Sprite):
    def __init__(self, color, initial_position):
        pygame.sprite.Sprite.__init__(self)
        size = 20
        self.gravity = 900
        self.velocity = 0
        self.bounce = 0.9

        self.image = pygame.Surface((size,size),pygame.SRCALPHA,32)
        pygame.draw.circle(self.image,color,(size/2,size/2),size/2)
        self.rect = self.image.get_rect()
        self.rect.center = initial_position

    def update(self, time_passed, size):

        self.velocity += (self.gravity * time_passed)
        self.rect.bottom += int(self.velocity * time_passed)

        if self.rect.bottom >= size[1]:
            self.rect.bottom = size[1]
            self.velocity = -self.velocity * self.bounce
```

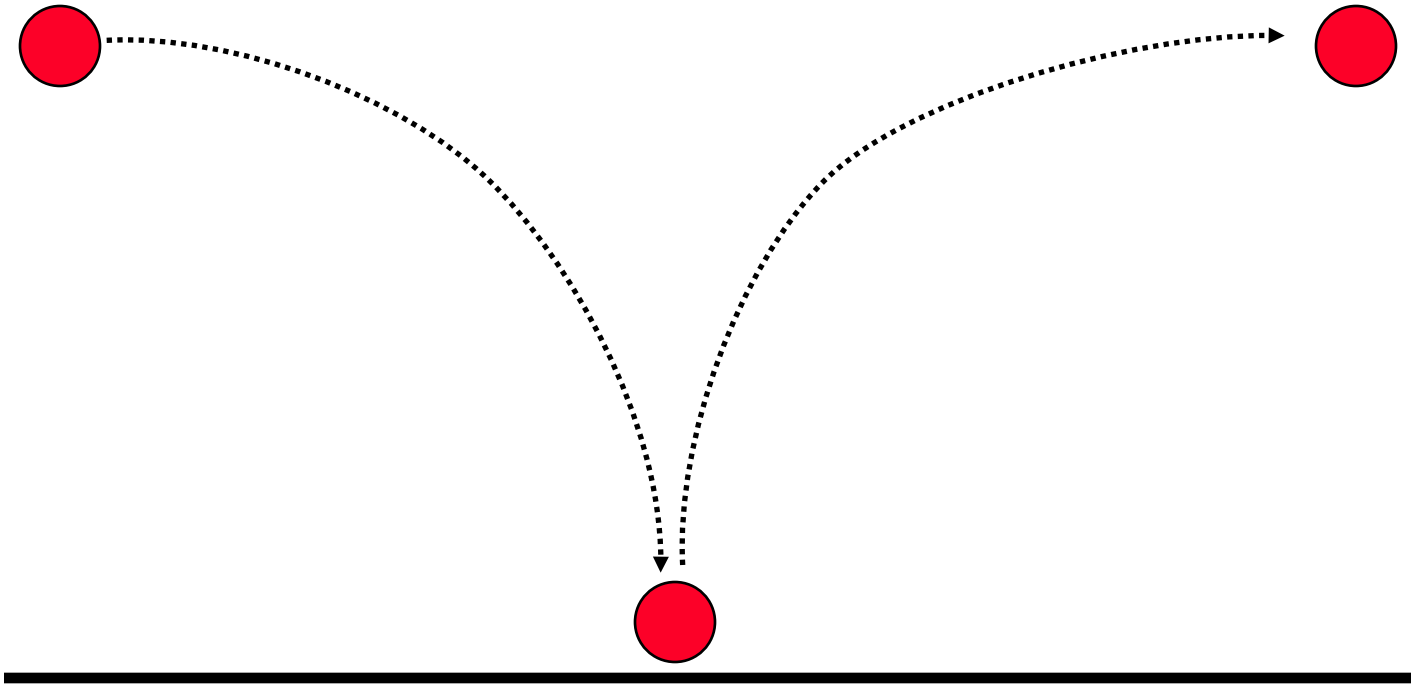
gravity per second, current velocity and bounce factor of the material

velocity is increased/decreased by the gravity

if the ball hits the ground, reduce velocity based on the bounce factor

Bouncing Ball Example 4

- Making the ball bounce and move vertically



Bouncing Ball Example 5

```
class Ball(pygame.sprite.Sprite):
    def __init__(self, color, initial_position):
        pygame.sprite.Sprite.__init__(self)
        size = 20
        self.gravity = 900
        self.vx = 0
        self.vy = 0
        self.bounce = 0.9
        …
    def update(self, time_passed, size):
        self.velocity += (self.gravity * time_passed)
        ydistance = int(self.vy * time_passed)
        self.rect.bottom += ydistance
        if ydistance == 0 and self.rect.bottom == size[1]: self.vx = 0
        self.rect.left += int(self.vx * time_passed)
        if self.rect.right >= size[0]:
            self.rect.right = size[0]
            self.vx = -self.vx
        if self.rect.left <= 0:
            self.rect.left = 0
            self.vx = -self.vx
        if self.rect.bottom >= size[1]:
            self.rect.bottom = size[1]
            self.vy = -self.vy* self.bounce
```

x and y velocity

clumsy way to make the ball stop

if the ball hits the sidewalls, make it change the direction

Arrival Angle = Angle of Reflection

- What if the Ball doesn't drop perfectly vertically?