Kapitel 13.1: Hitboxen – Einführung

Programmiert man Spiele mit Objekten, die sich bewegen, muss man normalerweise feststellen können, ob 2 Objekte miteinander kollidieren / gegeneinander prallen.

- Berühren sich 2 Autos beim Autorennen, gibt es einen Unfall, und beide Autos werden aus der Bahn geworfen.
- Trifft eine Kugel den Gegner, ist dieser verletzt und muss erst geheilt werden.
- Stößt eine Kugel gegen eine Wand, geht es nicht mehr weiter, oder man hat das Spiel verloren.
- Mit einer Spielfigur kann man verschiedene Items einsammeln, die dann irgendeinen Effekt haben.
- ...

Häufig haben diese Objekte unterschiedliche Größen und Formen, was das Erkennen von Kollisionen sehr erschwert. Aus diesem Grund arbeiten Spieleentwickler meistens mit sogenannten Hitboxen, die für den Benutzer aber nicht sichtbar sind. Im folgenden Beispiel wird die Hitbox in hellgrau mitgezeichnet, um das Prinzip besser zu veranschaulichen.

main.py

```
import pygame
from kreis import Kreis

pygame.init()
pygame.display.set_mode((500, 500))
pygame.display.set_caption("Kollisionen")

kreise = []
for i in range(4):
    k = Kreis(500, 500)
    if not k.detect_collision(kreise):
        kreise.append(k)

bRun = True
while bRun:
    pygame.time.delay(100)

    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            bRun = False

    for k in kreise:
        k.move()
        if k.detect_collision(kreise):
            k.change_direction()

    win = pygame.display.get_surface()
    win.fill((0, 0, 0))
    for k in kreise:
        k.draw(win)
```

```
    pygame.display.update()

pygame.quit()
```

kreis.py

```
import pygame
import random


class Kreis:
    def __init__(self, max_x, max_y):
        self.max_x = max_x
        self.max_y = max_y

        self.color = (random.randint(0, 255), random.randint(0, 255),
random.randint(0, 255))
        self.radius = 20

        self.x = random.randint(self.radius, max_x - self.radius)
        self.y = random.randint(self.radius, max_x - self.radius)

        self.move_x = random.randint(-10, 10)
        self.move_y = random.randint(-10, 10)

    def move(self):
        self.x += self.move_x
        self.y += self.move_y

        if self.x - self.radius <= 0:
            self.x = self.radius
            self.move_x *= -1
        elif self.x + self.radius >= self.max_x:
            self.x = self.max_x - self.radius
            self.move_x *= -1

        if self.y - self.radius <= 0:
            self.y = self.radius
            self.move_y *= -1
        elif self.y + self.radius >= self.max_y:
            self.y = self.max_y - self.radius
            self.move_y *= -1

    def draw(self, win):
        pygame.draw.circle(win, self.color, (self.x, self.y), self.radius)
        pygame.draw.rect(win, (100, 100, 100), self.get_hitbox(), 1)

    def get_hitbox(self):
        return pygame.Rect(self.x - self.radius, self.y - self.radius, 2 *
self.radius, 2 * self.radius)
```

```
    def detect_collision(self, kreise):
        for k in kreise:
            if not k == self:
                hitbox1 = self.get_hitbox()
                hitbox2 = k.get_hitbox()
                if hitbox1.colliderect(hitbox2):
                    return True
        return False

    def change_direction(self):
        self.move_x *= -1
        self.move_y *= -1
```