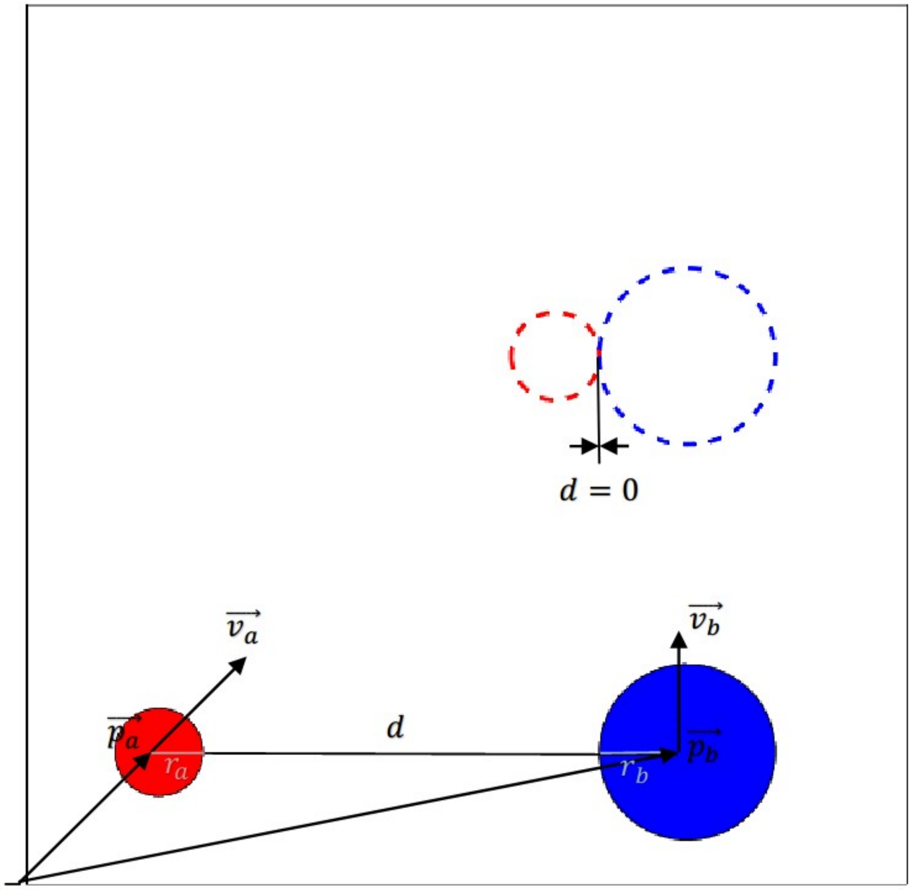
4. Rešavanje nelinearnih jednačina

Predviđanje sudara



Slika 1. Predviđanje sudara

Poznavajući fizičke karakteristike objekata i početne uslove kretanja, potrebno je odrediti trenutak sudara 2 objekta (slika 1). Radi jednostavnosti primera odabrani su krugovi i konstantne brzine kretanja. Funkcija položaja takvog kretanja je:

$$\vec{p}(t) = \vec{p}_0 + \vec{v}\, t \qquad (1)$$

, gde su $\vec{p}$ trenutni položaj tela, $\vec{p}_0$ početni položaj tela, $\vec{v}$ konstantna brzina tela, a t proteklo vreme. Funkcija rastojanja između 2 kruga je:

$$d(t) = \sqrt{\left(\vec{p}_b(t) - \vec{p}_a(t)\right)^2} - \left(r_a + r_b\right)$$

, gde su d trenutno rastojanje između 2 kruga, $\vec{p}_a$ položaj kruga A, r_a poluprečnik kruga A, $\vec{p}_b$ položaj kruga B, r_b poluprečnik kruga B, a t proteklo vreme.

Cilj je naći t, kada se krugovi dodiruju, tj. kada je rastojanje $d = 0$:

$$\sqrt{\left(\vec{p}_b(t) - \vec{p}_a(t)\right)^2} - \left(r_a + r_b\right) = 0 \qquad (2)$$

Zamenom jednačine (1) u (2) dobija se:

$$a t^2 + bt + c = 0 \quad (3)$$

$$a = \left\| \overrightarrow{v_{ab}} \right\|^2$$

$$b = 2\left(\overrightarrow{p_{ab}} \cdot \overrightarrow{v_{ab}} \right)$$

$$c = \left\| \overrightarrow{p_{ab}} \right\|^2 - r_{ab}^{\ 2}$$

$$\overrightarrow{v_{ab}} = \vec{v_b} - \vec{v_a}$$

$$\overrightarrow{p_{ab}} = \overrightarrow{p_{0b}} - \overrightarrow{p_{0a}}$$

$$r_{ab} = r_a + r_b$$

, gde su $\overrightarrow{p_{0a}}$ početni položaj kruga A, $\vec{v_a}$ brzina kruga A, r_a poluprečnik kruga a, $\overrightarrow{p_{0b}}$ početni položaj kruga B, $\vec{v_b}$ brzina kruga B, r_b poluprečnik kruga B, a $\overrightarrow{p_{ab}} \cdot \overrightarrow{v_{ab}}$ je skalarni proizvod vektora.

Traženjem nule funkcije $a t^2 + bt + c = 0$ (pri čemu su a, b i c unapred izračunate konstante) se dobija trenutak sudara t dva kruga.

Zadatak 1

Naći trenutak sudara između 2 kruga:

krug	**A**	**B**
r	$0.05\,m$	$0.1\,m$
$\vec{v}$	$(x, y) = (0.1, 0.1)\frac{m}{s}$	$(x, y) = (0.0, 0.1)\frac{m}{s}$
$\vec{p_0}$	$(x, y) = (0.0, 0.0)\,m$	$(x, y) = (0.75, 0.0)\,m$

a) Definisati fizičke konstante:

```
world_size = [1.0, 1.0] # [m]; dimenzije prostora

# krugovi
circle_count = 2
r = np.array([0.05, 0.10]) # [m]; dimenzije krugova
v = np.array([[0.1, 0.1],
              [0.0, 0.1]]) # [m/s]; brzine krugova
p0 = np.array([[0.05, 0.05],
               [0.85, 0.10]]) # [m]; pocetni polozaji krugova
colors = np.array([[1.0, 0.0, 0.0],
                   [0.0, 0.0, 1.0]]) # (R,G,B); boje krugova

f_motion = lambda p0, v, t: p0 + v*t # funkcija kretanja
```

b) Inicijalizovati grafički interfejs:

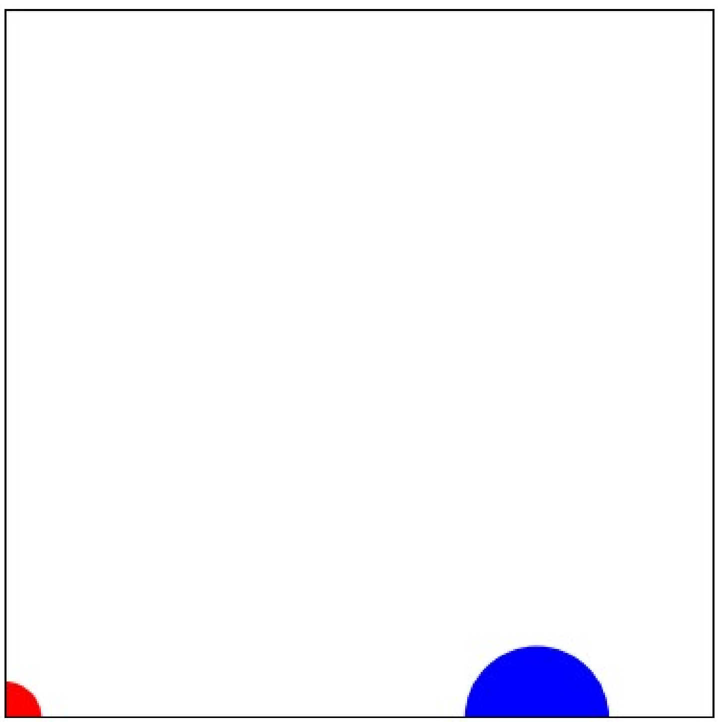
```
# GUI
fig, ax = plt.subplots()
plt.axis((0, world_size[0], 0, world_size[1])) # ogranicavanje prikaza u okviru dimenzija prostora
ax.set_aspect('equal') # sprecavanje reskaliranja prikaza
plt.axis('off') # sakrivanje osa

# ivice
plt.plot([0, world_size[0]], [0, 0], c='k')
plt.plot([0, world_size[0]], [world_size[1], world_size[1]], c='k')
plt.plot([0, 0], [0, world_size[1]], c='k')
plt.plot([world_size[0], world_size[0]], [0, world_size[1]], c='k')


# krugovi
circles = []
for circle in range(circle_count): # za svaki krug (circle)
    location = p0[circle]
    radius = r[circle]
    diameter = 2*radius
    x = location[0] - radius
    y = location[1] - radius

    c = plt.Circle((x, y), radius, color=colors[circle])
    circles.append(c)
```

Rezultat:



Slika 2. Početak simulacije

c) Simulirati kretanje krugova po funkciji kretanja (jednačini (1) iz uvoda):

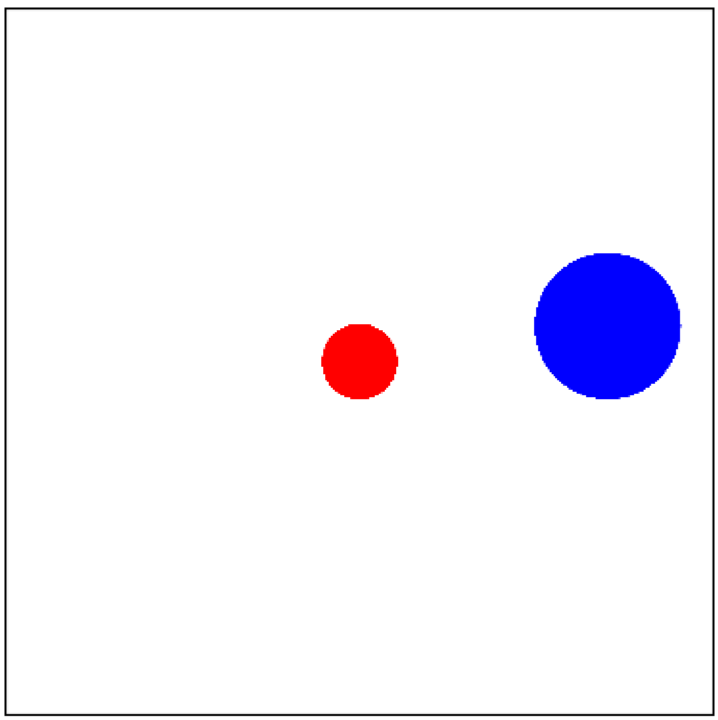
```
def init(): # inicijalizacija početnih pozicija
    for circle in circles:
        ax.add_patch(circle)
    return circles

def animate(t): # ažuriranje položaja
    for idx, circle in enumerate(circles):
        circle.center = f_motion(p0[idx], v[idx], t)
        ax.add_patch(circle)
    return circles

anim=animation.FuncAnimation(fig,animate,init_func=init,frames=60,blit=True)

plt.show()
```

Rezultat:



Slika 3. Simulacija

d) <u>Pre procedure</u> iz koraka c), pročitati vrednosti $\overrightarrow{p_{0a}}$, $\vec{v}_a$, r_a, $\overrightarrow{p_{0b}}$, $\vec{v}_b$ i r_b:

```
rA = r[0] # poluprecnik 1. kruga
vA = v[0] # brzina 1. kruga
p0A = p0[0] # pocetni polozaj 1. kruga

rB = r[1] # polupre?nik 2. kruga
vB = v[1] # brzina 2. kruga
p0B = p0[1] # pocetni polozaj 2. kruga
```

e) Pre procedure iz koraka c), naći nulu funkcije rastojanja (definisane jednačinom (3) iz uvoda) i rezultat sačuvati u promenljivu `t_collision`. Ona predstavlja trenutak sudara. Skalarni proizvod dva vektora A i B računati na sledeći način: `np.sum(A.conj()*B, axis=0)`.

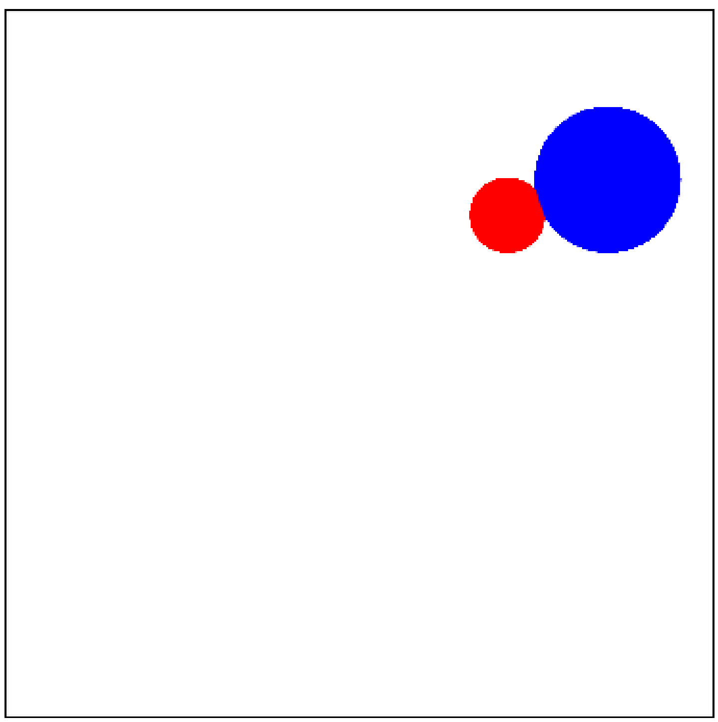
f) Sprečiti da se simulacija odvija nakon izračunatog trenutka sudara:

```
.
.
.

def animate(t): # azuriranje polozaja
    if t > t_collision: # ograniciti vreme na trenutak sudara
        t = t_collision
    .
    .
    .
    return circles

    .
    .
    .
```

Rezultat:



Slika 4. Sudar