9. Obične diferencijalne jednačine, problem početne vrednosti

1. Drugi Njutnov zakon

Zadatak 1

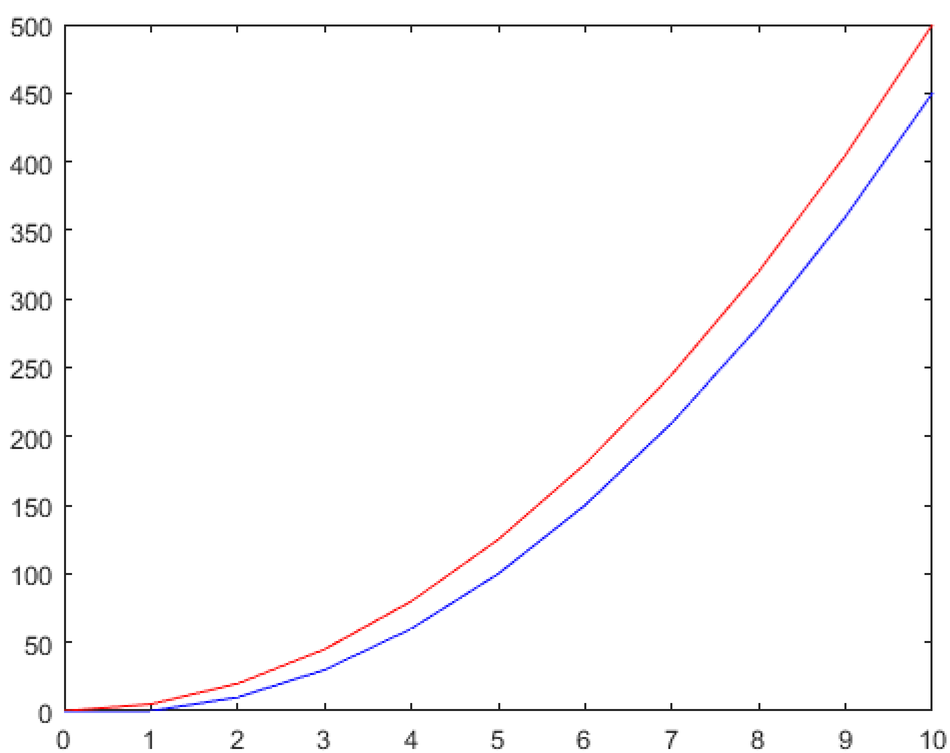
Ako na telo mase $1kg$, koje je u trenutku $0s$ imalo položaj $0m$ i brzinu $0\frac{m}{s}$, deluje konstantna sila od $10N$, naći položaj tela svake sekunde tokom narednih $10s$.

$$\frac{d^2s}{dt^2} = \frac{F}{m}$$
$$s''(t) = \frac{F}{m} = \frac{10}{1} = 10$$
$$s(0) = 0$$
$$s'(0) = v(0) = 0$$

Naći numeričko rešenje Ojlerovim metodom i metodom RK4 u 10 tačaka, zadajući korak `h`. Na istom grafiku nacrtati i uporediti 2 numerička i analitičko rešenje ($s(t) = v_0 t + \frac{F}{m}\frac{t^2}{2}$).
Uporediti nalaženje numeričkih rešenja u 10, 100, 1000, 10000 tačaka, menjajući korak `h`.

Rešenje (za 10 tačaka):

```
sTTrue =
     0     5    20    45    80   125   180   245   320   405   500

sTEuler =
     0     0    10    30    60   100   150   210   280   360   450

sTRK4 =
     0     5    20    45    80   125   180   245   320   405   500
```



Ojlerova metoda greši za veliki korak `h` (tj. za mali broj tačaka)!

2. Diskretne jednačine

Zadatak 2

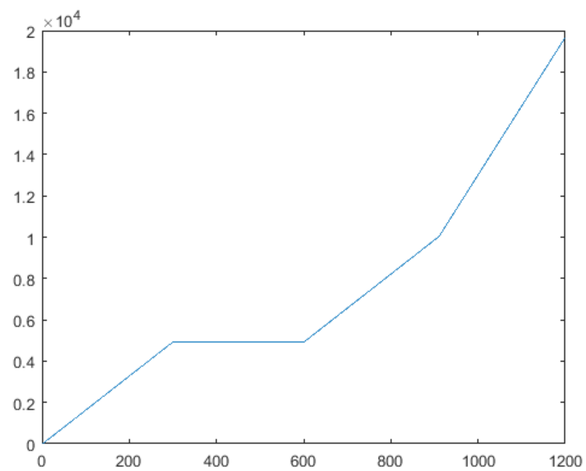
Pri dodavanju gasa vozlio ubrzava $5\frac{m}{s^2}$. Pri kočenju vozlio usporava $-10\frac{m}{s^2}$. Ako se vozilo iz stanja mirovanja kretalo po deonici puta dužine $10km$ sa ograničenjem brzine $60\frac{km}{h}$, a zatim ostatak puta sa ograničenjem brzine od $120\frac{km}{h}$ i ako je napravilo pauzu između 5. i 10. min., koliki put je prešlo nakon 20min?

a) Formulisati diferencijalnu jednačinu u posebnoj MATLAB datoteci (uz pomoć `if` selekcije definisati diskretne uslove kretanja):

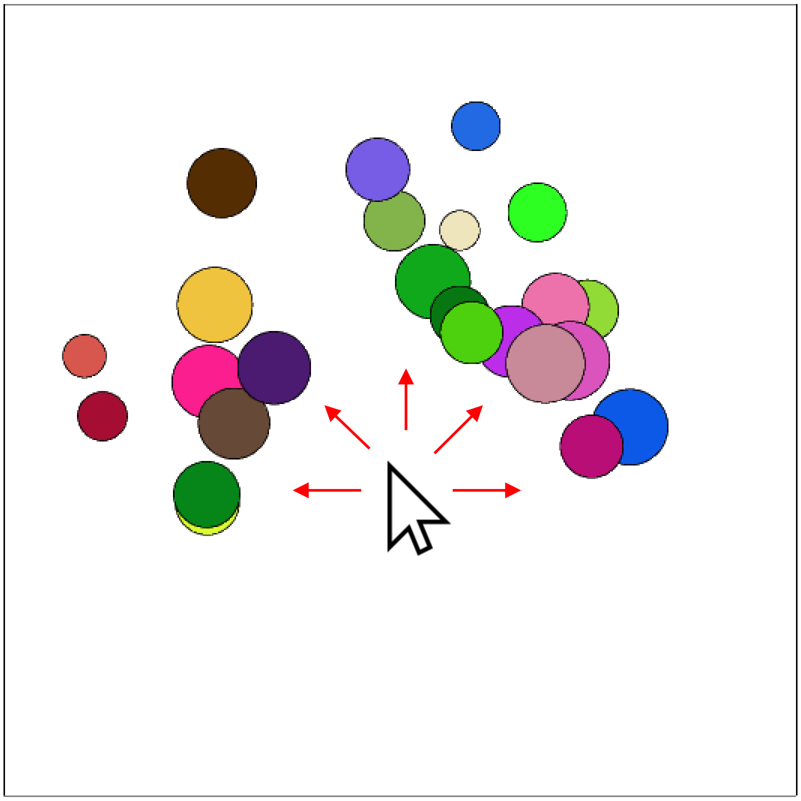
```
function ddsT = fp2(t, sT, dsT)
    % t - vreme proteklo od početka putovanja
    % sT - pređeni put
    % dsT - trenutna brzina vozila
    v = dsT;

    % ograničenje brzine
    speedLimit = 60*1000/3600; % 60km/h
    if sT > 10*1000 % nakon 10km
        speedLimit = 120*1000/3600; % 120km/h
    end

    acc = 5; % 5m/(s^2)
    dec = -10; % -10m/(s^2)
    if t > 5*60 && t < 10*60 % pauza između 5. i 10. min.
        if v >= 0 % usporavati sve dok vozilo ne stane
            a = dec;
        else % ako vozilo "krene u nazad" usled negativnog ubrzanja, ubrzati ponovo do 0
            a = acc;
        end
    else
        if v <= speedLimit % ubrzavati ako je brzina ispod ograničenja
            a = acc;
        else
            a = dec; % usporiti ako je brzina preko ograničenja
        end
    end

    ddsT = a;
end
```

b) Rešiti PPV

rešenje: $s(20min) = 19.678km$



3. Fizika u video igrama (interaktivno kretanje)



Slika 1. Interaktivno kretanje

Poznavajući fizičke karakteristike objekata i početne uslove kretanja, potrebno je simulirati njihovo kretanje u proizvoljnom vremenskom intervalu, pri čemu uslovi kretanja mogu da se menjaju u svakom vremenskom trenutku (slika 1). Radi jednostavnosti primera odabrane su sfere.

Jednačina položaja takvog kretanja (2. Njutnov zakon) je:

$$\frac{d^2\vec{p}(t)}{dt^2} = \frac{\vec{F}(t)}{m} \tag{1}$$

, gde su $\vec{p}$ trenutni položaj tela, m je masa tela, $\vec{F}$ je sila koja deluje na telo, a t proteklo vreme. $\vec{F}(t)$ je veličina koja menja uslove kretanja u svakom vremenskom trenutku. $\vec{F}(t)$ je nepoznata funkcija i zavisi od korisničke interakcije. Nije moguće izraziti ovu funkciju analitički.

Funkciju položaja je međutim u svakom vremenskom trenutku $(t + \Delta t)$ moguće razviti u Tejlorov red:

$$\vec{p}(t+\Delta t) = \vec{p}(t) + \Delta t\frac{d\vec{p}(t)}{dt} + \frac{\Delta t^2}{2}\frac{d^2\vec{p}(t)}{dt^2} + \cdots + \frac{\Delta t^n}{n!}\frac{d^n\vec{p}(t)}{dt^n}$$

Ograničavanjem beskonačne sume Tejlorovog reda na konačnu, moguće je numeričkim putem doći do rešenja diferencijalne jednačine u uzastopnim vremenskim trenucima.

Ojlerov metod uzima u obzir prva 2 člana reda:

$$\vec{p}(t+\Delta t) \approx \vec{p}(t) + \Delta t\frac{d\vec{p}(t)}{dt}$$

Diferenciranjem obe strane jednačine dobija se:

$$\vec{p}(t + \Delta t) = \vec{p}(t) + \Delta t \frac{d\vec{p}(t)}{dt}$$

$$\frac{d\vec{p}(t + \Delta t)}{dt} = \frac{d\vec{p}(t)}{dt} + \Delta t \frac{d^2\vec{p}(t)}{dt^2}$$

Uvođenjem smene $\frac{d\vec{p}(t)}{dt} = \vec{v}(t)$, dobija se:

$$\vec{p}(t + \Delta t) = \vec{p}(t) + \Delta t \vec{v}(t)$$

$$\vec{v}(t + \Delta t) = \vec{v}(t) + \Delta t \frac{d^2\vec{p}(t)}{dt^2} \tag{2}$$

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

$$\vec{p}(t + \Delta t) = \vec{p}(t) + \Delta t \vec{v}(t)$$

$$\vec{v}(t + \Delta t) = \vec{v}(t) + \Delta t \frac{\vec{F}(t)}{m}$$

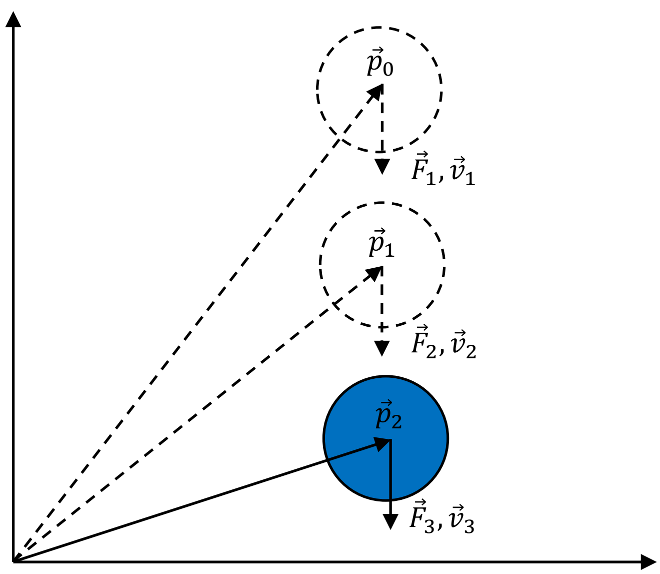
Prevođenjem u iterativni zapis, dobija se:

$$\vec{p}_i = \vec{p}_{i-1} + \Delta t \vec{v}_{i-1}$$

$$\vec{v}_i = \vec{v}_{i-1} + \Delta t \frac{\vec{F}_{i-1}}{m}$$

Masa tela m je konstantna. Ako je silu $\vec{F}_{i-1}$ u svakom trenutku moguće izraziti i izračunati i ako su početni položaj $\vec{p}_0$ i početna brzina $\vec{v}_0$ tela poznati, tada je **Ojlerovom integracijom** moguće naći položaj $\vec{p}_i$ i brzinu $\vec{v}_i$ u svakom vremenskom trenutku $(0 + \mathrm{i}\Delta t)$:

$\mathbf{0}$	$\mathbf{0 + \Delta t}$	$\mathbf{0 + 2\Delta t}$	$\mathbf{0 + i\Delta t}$
$\vec{p}_0$	$\vec{p}_1 = \vec{p}_0 + \Delta t \vec{v}_0$	$\vec{p}_2 = \vec{p}_1 + \Delta t \vec{v}_1$	$\vec{p}_i = \vec{p}_{i-1} + \Delta t \vec{v}_{i-1}$
$\vec{v}_0$	$\vec{v}_1 = \vec{v}_0 + \Delta t \frac{\vec{F}_0}{m}$	$\vec{v}_2 = \vec{v}_1 + \Delta t \frac{\vec{F}_1}{m}$	$\vec{v}_i = \vec{v}_{i-1} + \Delta t \frac{\vec{F}_{i-1}}{m}$



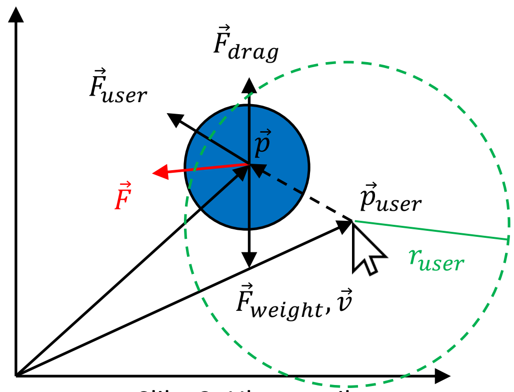
Slika 2. Ojlerova integracija

RK4 metoda podrazumeva iste parametre, pa se može upotrebiti na **isti način**.

Ako su poluprečnik r i gustina sfere ρ poznati (za gumu npr. $\rho_{rubber} = 1522 \frac{kg}{m^3}$), masa sfere se može izračunati na sledeći način:

$$m = \rho_{rubber} V = \rho_{rubber} \frac{4}{3} r^3 \pi$$

Ukupna sila $\vec{F}$ koja deluje na telo u svakom trenutku se može izračunati kao linearna kombinacija različitih komponenata:



Slika 3. Ukupna sila

$$\vec{F} = \vec{F}_{weight} + \vec{F}_{drag} + \vec{F}_{user}$$

Težina tela $\vec{F}_{weight}$ se izračunava na sledeći način:

$$\vec{F}_{weight} = m\vec{g}$$

, gde je m masa tela, a $\vec{g} = [0 \quad -9.81]\frac{m}{s^2}$ je gravitaciono ubrzanje.

Sila otpora vazduha $\vec{F}_{drag}$ se izračunava na sledeći način:

$$\vec{F}_{drag} = -\vec{v}^2\frac{1}{2}\rho_{air}C_dA == -v|\vec{v}|\frac{1}{2}\rho_{air}C_dA$$

, gde je $\rho_{air} = 1.225\frac{kg}{m^3}$ gustina vazduha, $\vec{v}$ je brzina tela, C_d je koeficijent aerodinamičnosti tela (za sfere $C_d = 0.47$), a A je poprečni presek tela normalan na pravac kretanja (za sfere $A = r^2\pi$). Primetiti da je sila otpora vazduha po pravcu ista, a po smeru suprotna brzini kretanja.

Korisnička sila $\vec{F}_{user}$ se izračunava na sledeći način:

$$\vec{F}_{user} = \begin{cases} 0 & , \quad |\vec{p} - \vec{p}_{user}| > r_{user} \\ \dfrac{\vec{p} - \vec{p}_{user}}{|\vec{p} - \vec{p}_{user}|} f_{user} & , \quad |\vec{p} - \vec{p}_{user}| \le r_{user} \end{cases}$$

, gde je $\vec{p}$ položaj tela, $\vec{p}_{user}$ je položaj pokazivača miša, r_{user} poluprečnik kruga delovanja korisničke sile, a f_{user} je intenzitet korisničke sile.

Zadatak 3

Simulirati slobodan pad gumenih sfera nasumično generisanih položaja iz stanja mirovanja kroz vazduh. Omogućiti da korsinik pokazivačem miša može da unese dodatnu silu u simulaciju.

Konstante prostora:

dimenzije prostora	$(width, height) = [10 \quad 10]m$
gravitaciono ubrzanje	$\vec{g}(x,y) = [0 \quad -9.81]\frac{m}{s^2}$
gustina vazduha	$\rho_{air} = 1.225\frac{kg}{m^3}$
gustina gume	$\rho_{rubber} = 1522\frac{kg}{m^3}$
koeficijent aerodinamičnosti sfere	$C_d = 0.47$

Sfere:

veličina	min. nasumično generisana vrednost	maks. nasumično generisana vrednost
broj	25	25
poluprečnik	$r_{min} = 0.25m$	$r_{max} = 0.5m$
početne brzine sfera	$\vec{v}_{min}(x,y) = [0 \quad 0]\frac{m}{s}$	$\vec{v}_{max}(x,y) = [0 \quad 0]\frac{m}{s}$
položaji sfera	$\vec{p}_{min}(x,y) = [2.5 \quad 7.5]m$	$\vec{p}_{max}(x,y) = [7.5 \quad 12.5]m$

Korisnička sila:

poluprečnik kruga delovanja korisničke sile	$r_{user} = 2m$
intenzitet korisničke sile	$f_{user} = 15000N$

c) Definisati konstante prostora:

```
worldSize = [10.0 10.0]; % [m]; dimenzije prostora

g = 9.81; % [m/s^2]; gravitaciono ubrzanje
airDensity = 1.225; % [kg/m^3]; gustina vazduha
rubberDensity = 1522; % [kg/m^3]; gustina gume
dragCoefficient = 0.47; % koeficijent aerodinamičnosti sfera
```

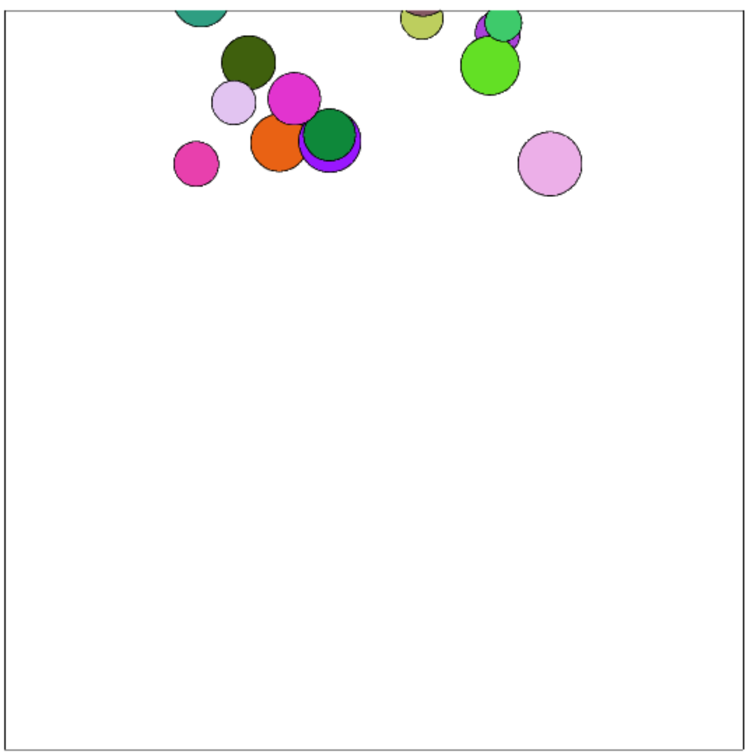
d) Definisati sfere:

```
% sfere
sphereCount = 25;
r = (0.5 + rand(sphereCount, 1)*0.5)*0.5; % [m]; dimenzije sfera
A = r.^2*pi; % [m^2]; poprečni preseci sfera
m = rubberDensity*4/3*r.^3*pi; % [kg]; mase (gumenih) sfera
v = zeros(sphereCount, 2); % [m/s]; trenutne brzine sfera
p = [(0.25 + rand(sphereCount, 1)*0.5)*worldSize(1) (0.75 + rand(sphereCount, 1)*0.5)*worldSize(2)]; %
[m/s]; trenutni položaji sfera
colors = rand(sphereCount, 3); % (R,G,B); boje sfera
```

e) Inicijalizovati grafički interfejs:

```
% GUI
fig = figure('Name', 'Physics', 'Units', 'normalized', 'Position', [0.2 0.2 0.6 0.6]); % prozor
axis([0 worldSize(1) 0 worldSize(2)]) % ograničavanje prikaza u okviru dimenzija prostora
axis square % sprečavanje reskaliranja prikaza
axis off % sakrivanje osa

% inicijalizacija grafičkih objekata
graphics = gobjects(4 + sphereCount);
% ivice
graphics(1) = line([0 worldSize(1)], [0 0], 'color', 'black');
graphics(2) = line([0 worldSize(1)], [worldSize(2) worldSize(2)], 'color', 'black');
graphics(3) = line([0 0], [0 worldSize(2)], 'color', 'black');
graphics(4) = line([worldSize(1) worldSize(1)], [0 worldSize(2)], 'color', 'black');
% sfere
for sphere = 1:sphereCount % za svaku sferu(sphere)
    location = p(sphere, :);
    radius = r(sphere);
    diameter = 2*radius;
    x = location(1) - radius;
    y = location(2) - radius;

    position = [x y diameter diameter];
    color = colors(sphere, :);
    % grafički objekti od 1 do 4 su ivice
    graphics(4 + sphere) = rectangle('Position', position, 'Curvature', [1 1], 'EdgeColor', 'black',
'FaceColor', color);
end
```

Rezultat:



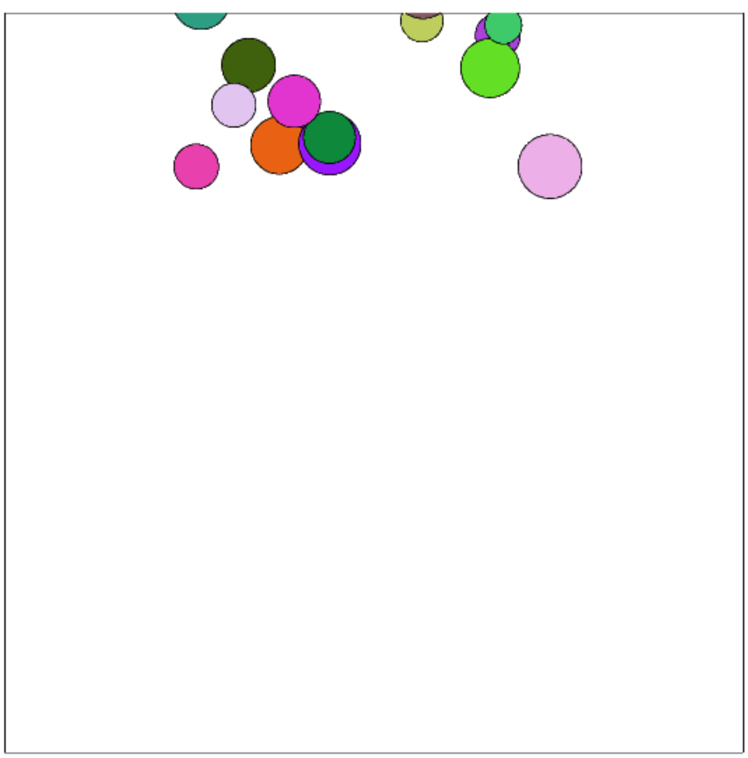
Slika 2. Početak simulacije

f) Simulirati kretanje sfera Ojlerovom integracijom:

```
fps = 60; % broj osvežavanja prikaza u sekundi
timeScale = 1.0; % brzina simulacije

t1 = 0; % [s]; početni vremenski trenutak
dt = 1/fps; % [s]; vremenska razlika između koraka
while ishandle(fig) % dokle god je prozor otvorent
    t2 = t1 + dt*timeScale; % naredni vremenski trenutak

    % ažuriranje položaja i iscrtavanje
    for sphere = 1:sphereCount
        % sile
        F = [0 0];

        % integracija
        % -----------------------------------------------------------------
        ddpX = @(t, p, v) F(1)/m(sphere); % (p"(t) = F(t)/m) funkcija kretanja (x)
        ddpY = @(t, p, v) F(2)/m(sphere); % (p"(t) = F(t)/m) funkcija kretanja (y)
        [~, pnX] = eulerN(t1, t2, t2 - t1, [p(sphere, 1) v(sphere, 1)], ddpX, 0.0); % integracija (x)
        [~, pnY] = eulerN(t1, t2, t2 - t1, [p(sphere, 2) v(sphere, 2)], ddpY, 0.0); % integracija (y)
        p(sphere, :) = [pnX(1, end) pnY(1, end)]; % trenutni položaj
        v(sphere, :) = [pnX(2, end) pnY(2, end)]; % trenutna brzina

        % prikaz
        % -----------------------------------------------------------------
        location = p(sphere, :);
        radius = r(sphere);
        diameter = 2*radius;
        x = location(1) - radius;
        y = location(2) - radius;

        position = [x y diameter diameter];
        set(graphics(4 + sphere), 'Position', position); % ažuriranje položaja grafičkih objekata
    end

    t1 = t2; % prošli vremenski trenutak
    pause(dt);
end
```

$$p_x(t) = \begin{bmatrix} p_x(t_1) & p_x(t_2) \\ v_x(t_1) & v_x(t_2) \end{bmatrix}$$

Rezultat:

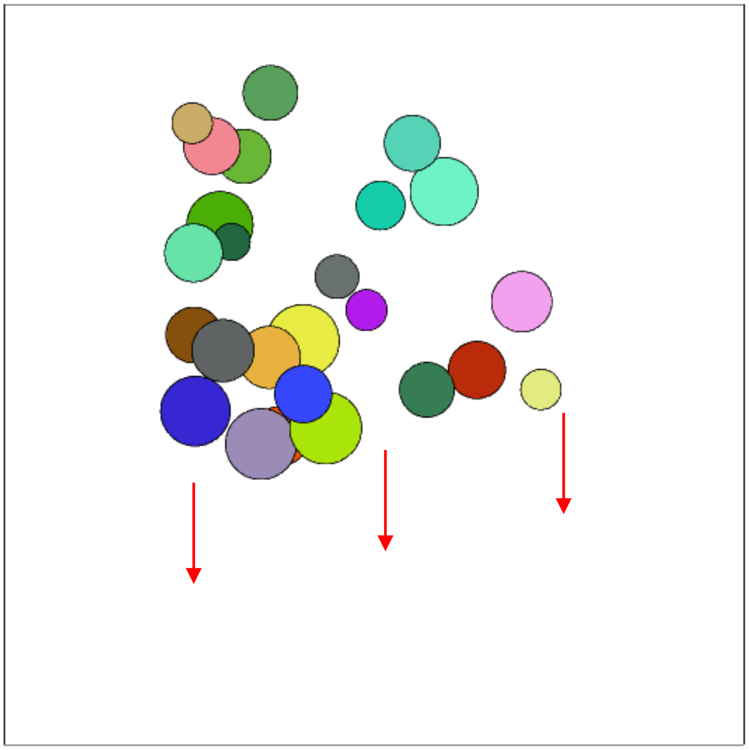


Slika 3. Bez sile kretanje nije moguće

g) Uvesti sile težine tela i otpora vazduha:

```
.
.
.

% ažuriranje položaja i iscrtavanje
for sphere = 1:sphereCount
    % sile
    % ---------------------------------------------------------------------
    FWeight = [0 -m(sphere)*g]; % težina tela (x, y)

    velocity = v(sphere, :);
    referenceArea = r(sphere)^2*pi;
    FDrag = -velocity*norm(velocity)*0.5*airDensity*dragCoefficient*A(sphere); % otpor vazduha (x, y)

    F = FWeight + FDrag;

    % integracija
    % ---------------------------------------------------------------------
    .
    .
    .
```

Rezultat:



Slika 4. Slobodan pad

h) Definisati funkciju `out = normalize(in)` koja za ulazni vektor `in` vraća jedinični vektor `out` istog pravca i smera:

```
function out = normalize(in)
    magnitude = norm(in);
    if magnitude == Inf || magnitude <= 0
        out = [1, 0];
    else
        out = in/magnitude;
    end
end
```

i) Definisati funkciju `guiMouseMove(~, ~)` koja na događaj pomeranja pokazivača miša čuva položaj miša u globalnoj promenljivoj `'mouseLocation'`:

```
function guiMouseMove(~, ~)
    mouseLocation = get(gca, 'CurrentPoint');
    setappdata(gcf, 'mouseLocation', mouseLocation)
end
```

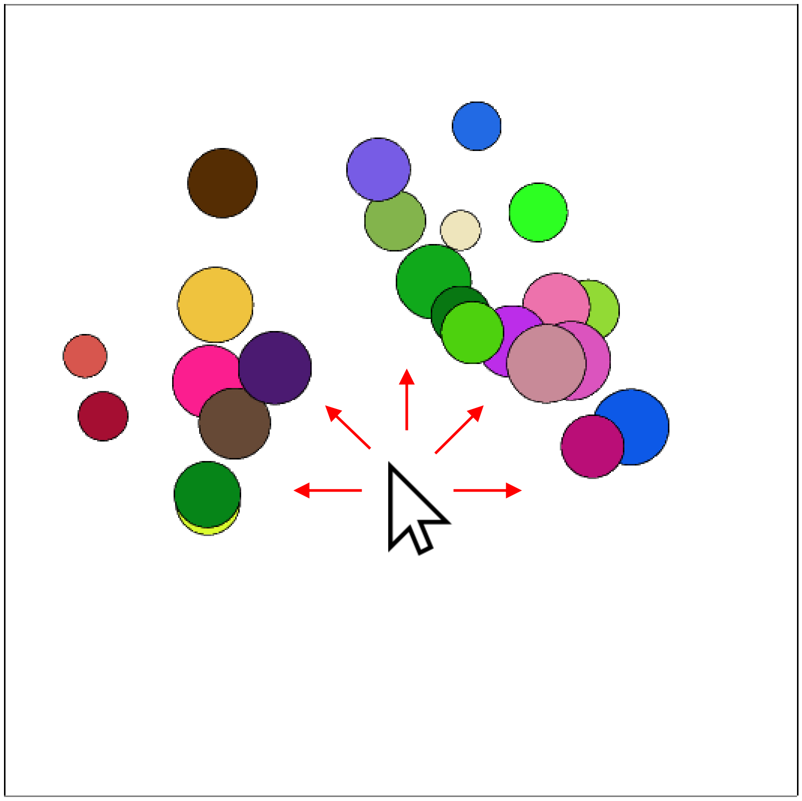
j) Pre procedure iz koraka d), definisati parametre korisničke sile i registrovati funkciju `guiMouseMove` da reaguje na događaj pomeranja pokazivača miša:

```
% user force
rUser = min(worldSize)*0.2; % [m]
fUser = 15000; % [N = kg*m/s^2]
pUser = [-Inf -Inf]; % [m]
setappdata(gcf, 'mouseLocation', pUser) % inicijalizacija globalne promenljive

set(gcf, 'WindowButtonMotionFcn', @guiMouseMove); % registrovanje funkcije
```

k) Uvesti delovanje korisničke sile:

```
.
.
.

pUser = getappdata(gcf, 'mouseLocation'); % čitanje vrednosti globalne promenljive

% ažuriranje položaja i iscrtavanje
for sphere = 1:sphereCount
    % sile
    % ------------------------------------------------------------------
    .
    .
    .

    FUser = [0 0];
    direction = p(sphere, :) - pUser(1, 1:2);
    if norm(direction) <= rUser
        FUser = normalize(direction)*fUser; % korisnička sila (x, y)
    end

    F = FWeight + FDrag + FUser;

    % integracija
    % ------------------------------------------------------------------
    .
    .
    .
```

Rezultat:



Slika 5. Interaktivno kretanje