ИКТ в НОС

Гледна точка

Тема №16

Гледна точка

Гледна точка

Гледна точка

- Всяка нарисувана сцена включва гледна точка
- Явна или неявна, винаги я има

Използване

- За определяне как ще се вижда сцената
- За движение из тримерен свят
- За плъзгане на образа, приближаване и т.н.

Съдържание на гледната точка

Не е просто 3D точка

- Определя местоположението на потребителя (зрителя) спрямо виртуалното пространство
- Определя посоката на гледане
- Определя ориентацията на образа

Математическа обусловеност

- Гледната точка е проективна матрица
- Трябва да не е с нулева дискриминанта


БГ

Гледна точка в СУИКА

Функция lookAt

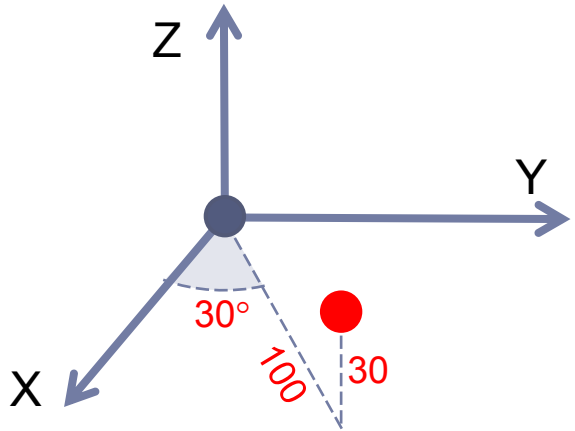
- Дефинира гледна точка
- Функция lookAt (*позиция*, *цел*, *нагоре*)

И още

- Функцията demo използва вътрешно lookAt
- Проекцията с perspective или orthographic е независима от гледната точка

Позиция е тримерна точка

- Определя мястото, където е окото на зрителя
- По подразбиране е [86.6, 50, 30] или 100[cos 30°,sin 30°,0.3]

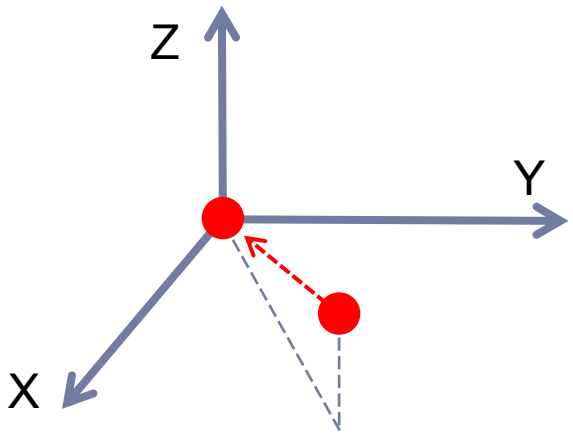


Изисквания

- Да е достатъчно далеч от гледаната сцена
- Да не е прекалено далеч от сцената

Целта е тримерна точка

- Определя точката, към която се гледа
- Тя се появява в средата на графичния прозорец
- По подразбиране е [0,0,0]

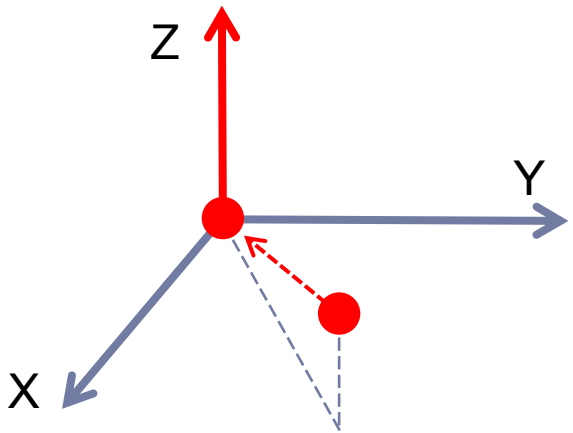


Изисквания

- Да е различна от позицията на гледаната точка

Посоката нагоре е вектор

- Определя как е завъртяна сцената
- Сочи посоката „нагоре“ по екрана
- По подразбиране е [0,0,1]

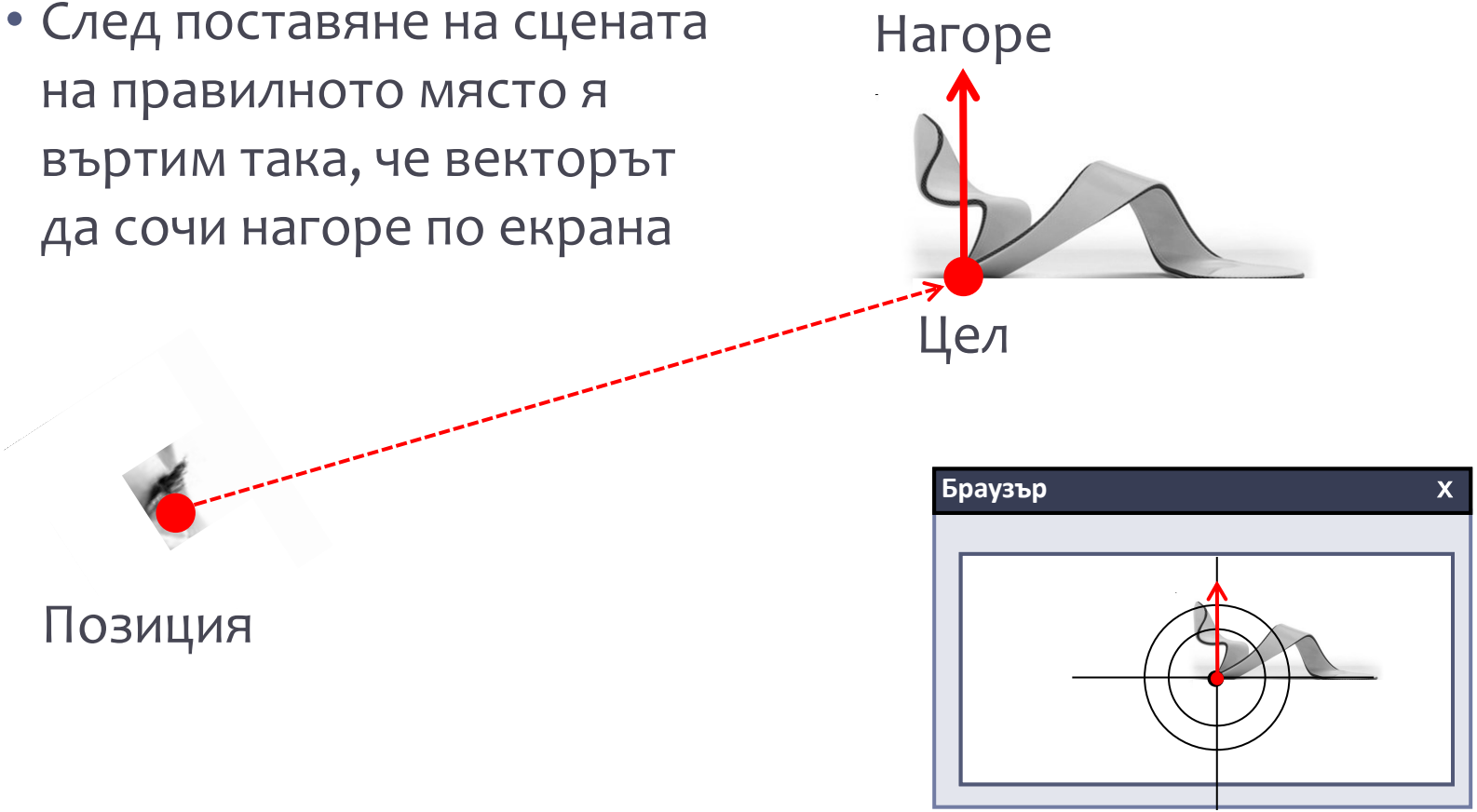


Изисквания

- Да не се вижда като нулев вектор от дадената позиция

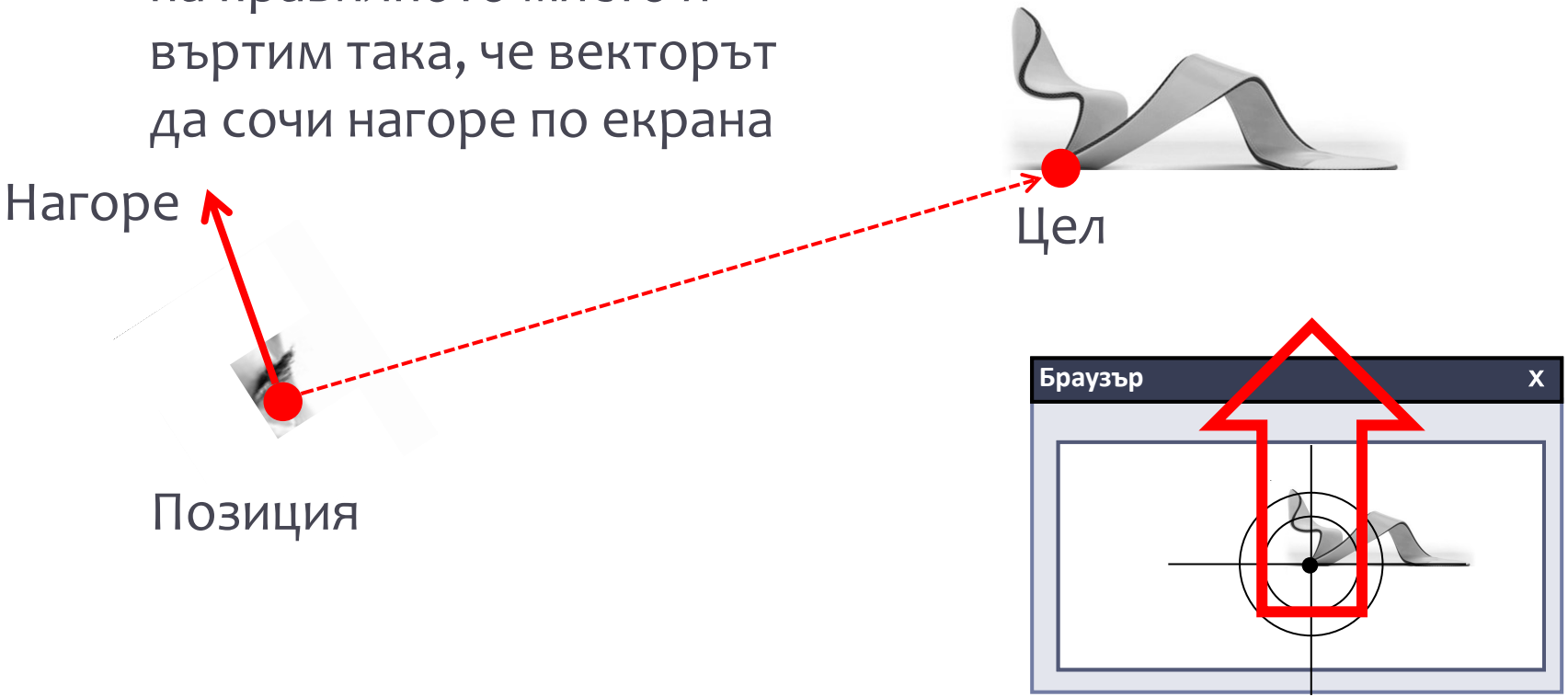
Интерпретация №1

- След поставяне на сцената на правилното място я въртим така, че векторът да сочи нагоре по екрана



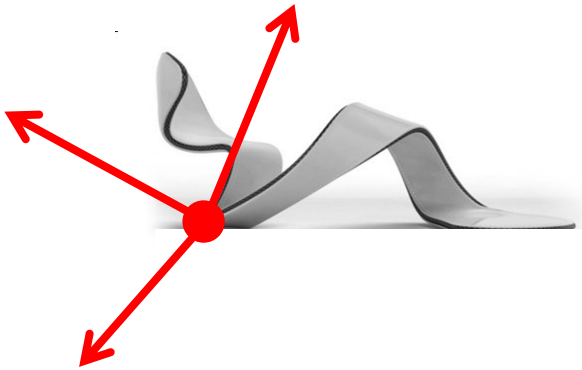
Интерпретация №2

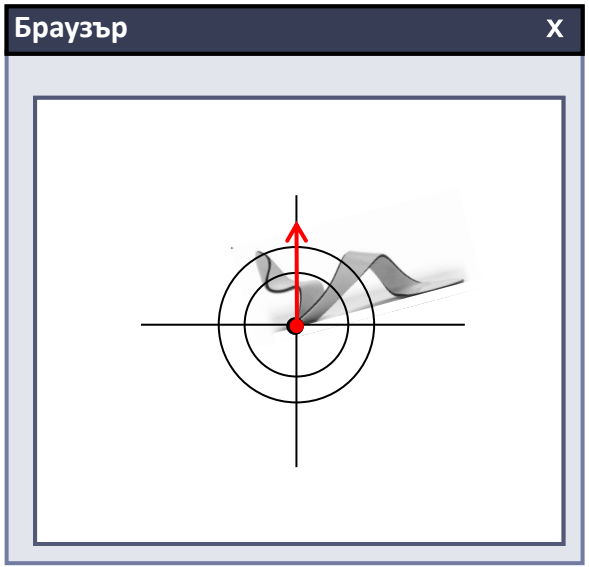
- След поставяне на сцената на правилното място я въртим така, че векторът да сочи нагоре по екрана

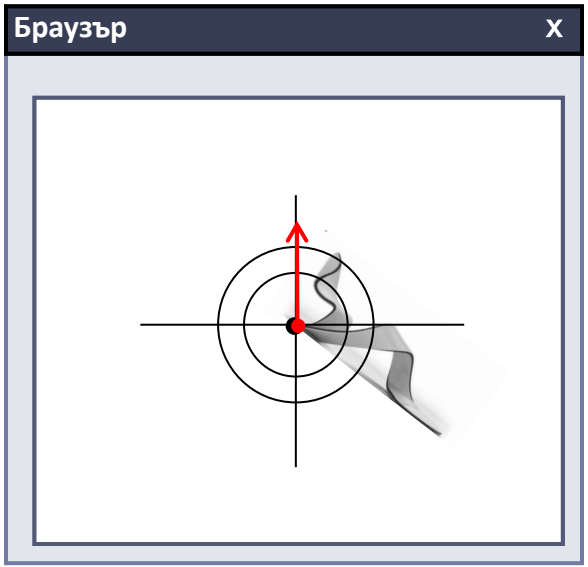


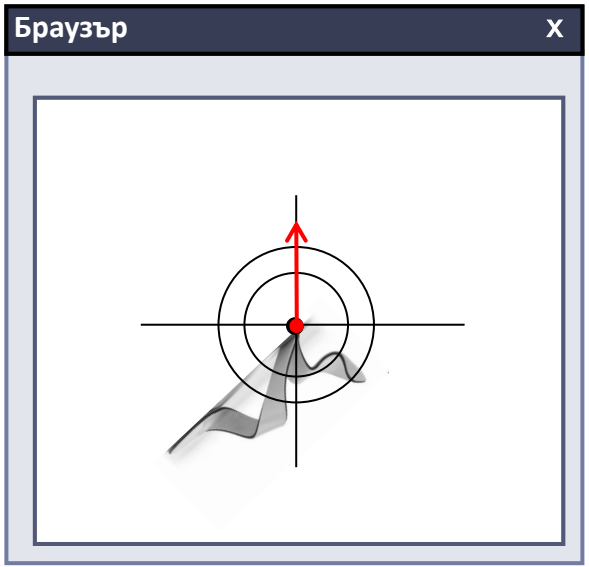
Роля на посоката нагоре

- Една и съща позиция
- Една и съща цел
- Различна посока нагоре





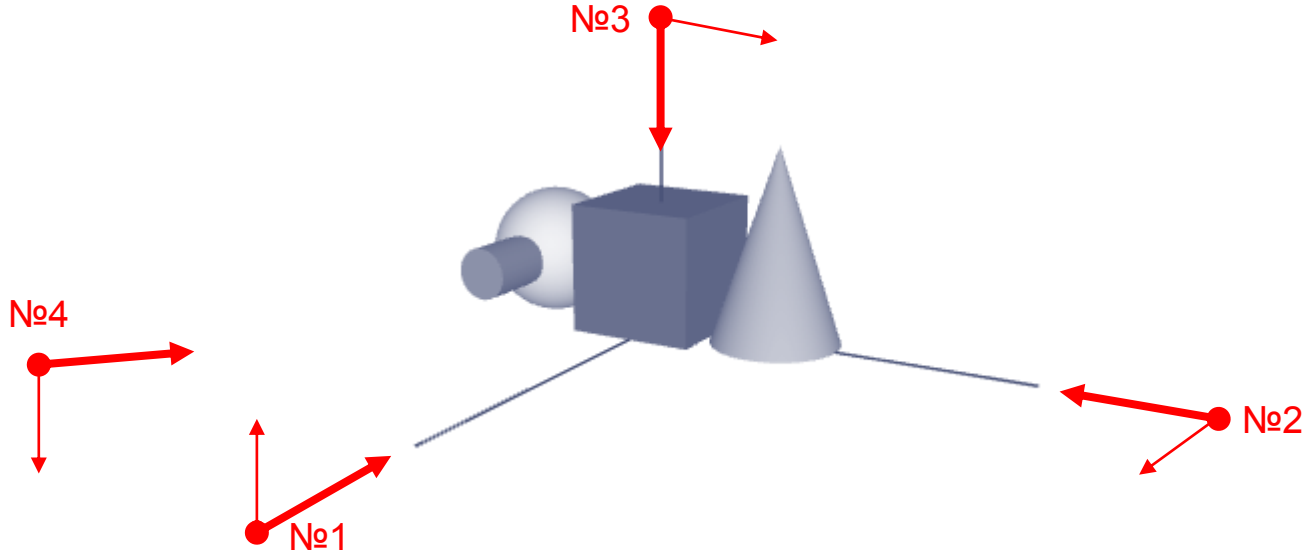




Пример

Различни гледни точки

- Сцена от няколко фигури + координатна система
- Различни гледни точки (тънкият вектор е посоката нагоре)



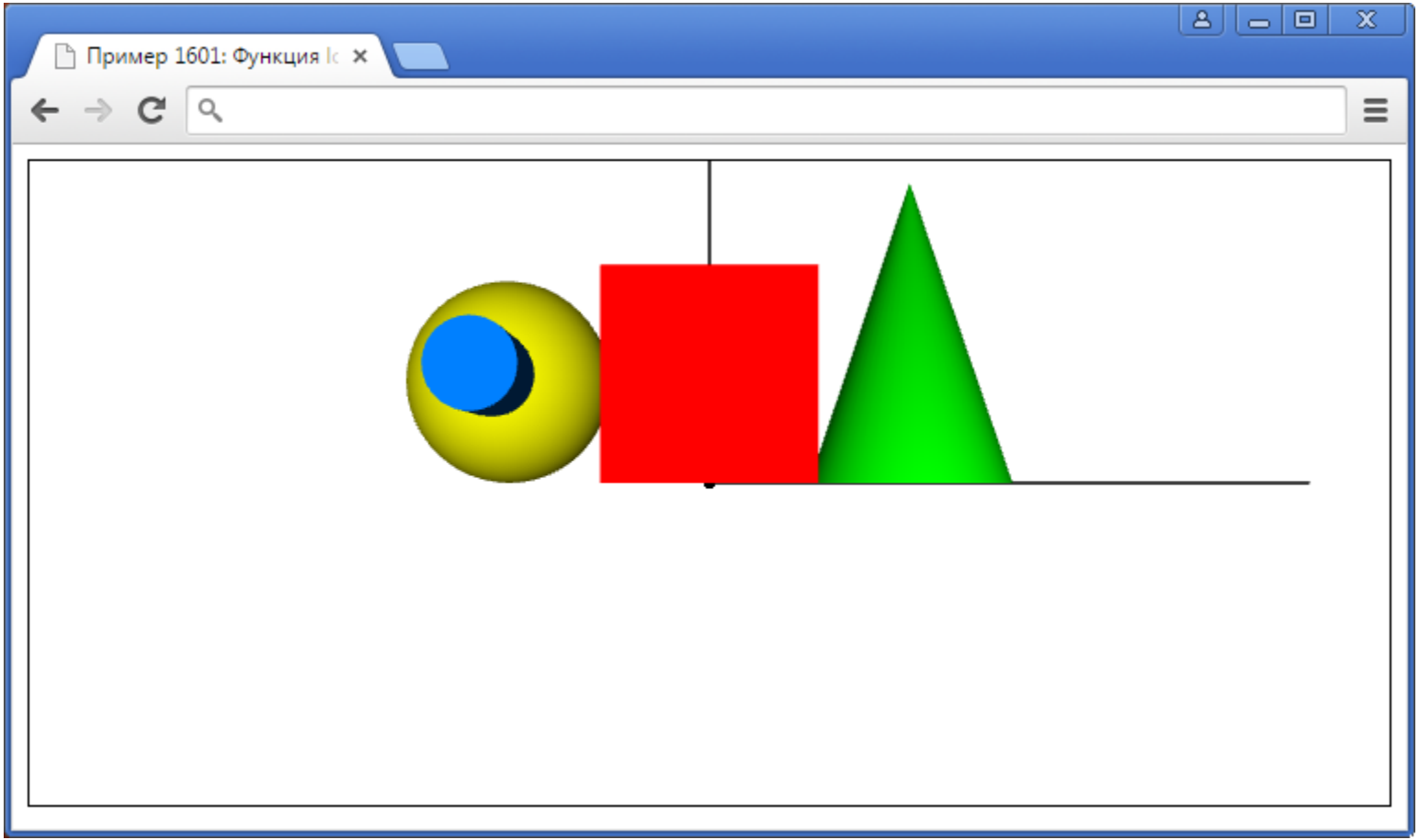
Реализация

- Началото на големите вектори определят позициите
- Целта и при четирите е все (0,0,0)
- Малките вектори определят посоката нагоре

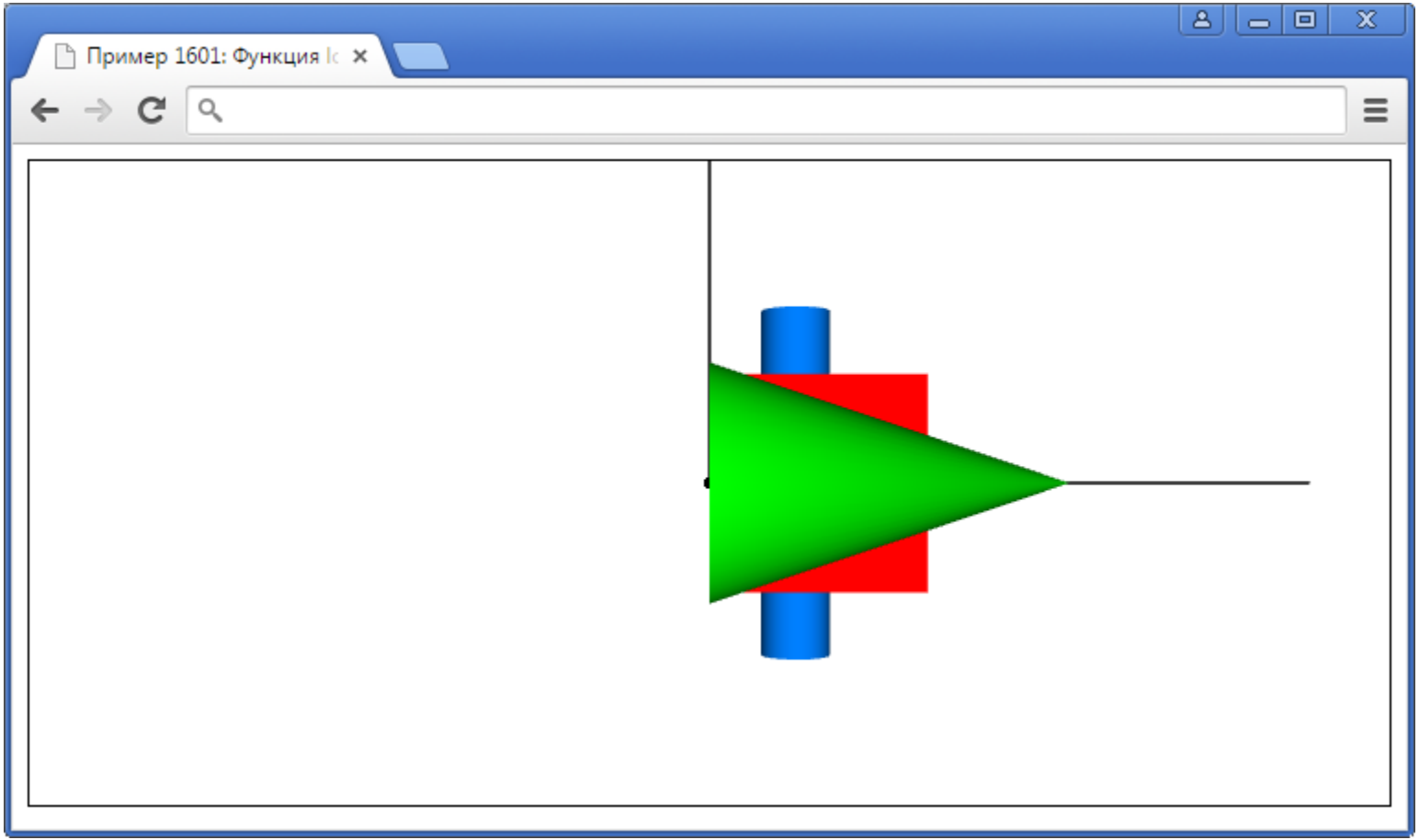
Изискванията

- Началата и целите да са различни точки
- Векторите нагоре да не са нулеви и да не се виждат като нулеви

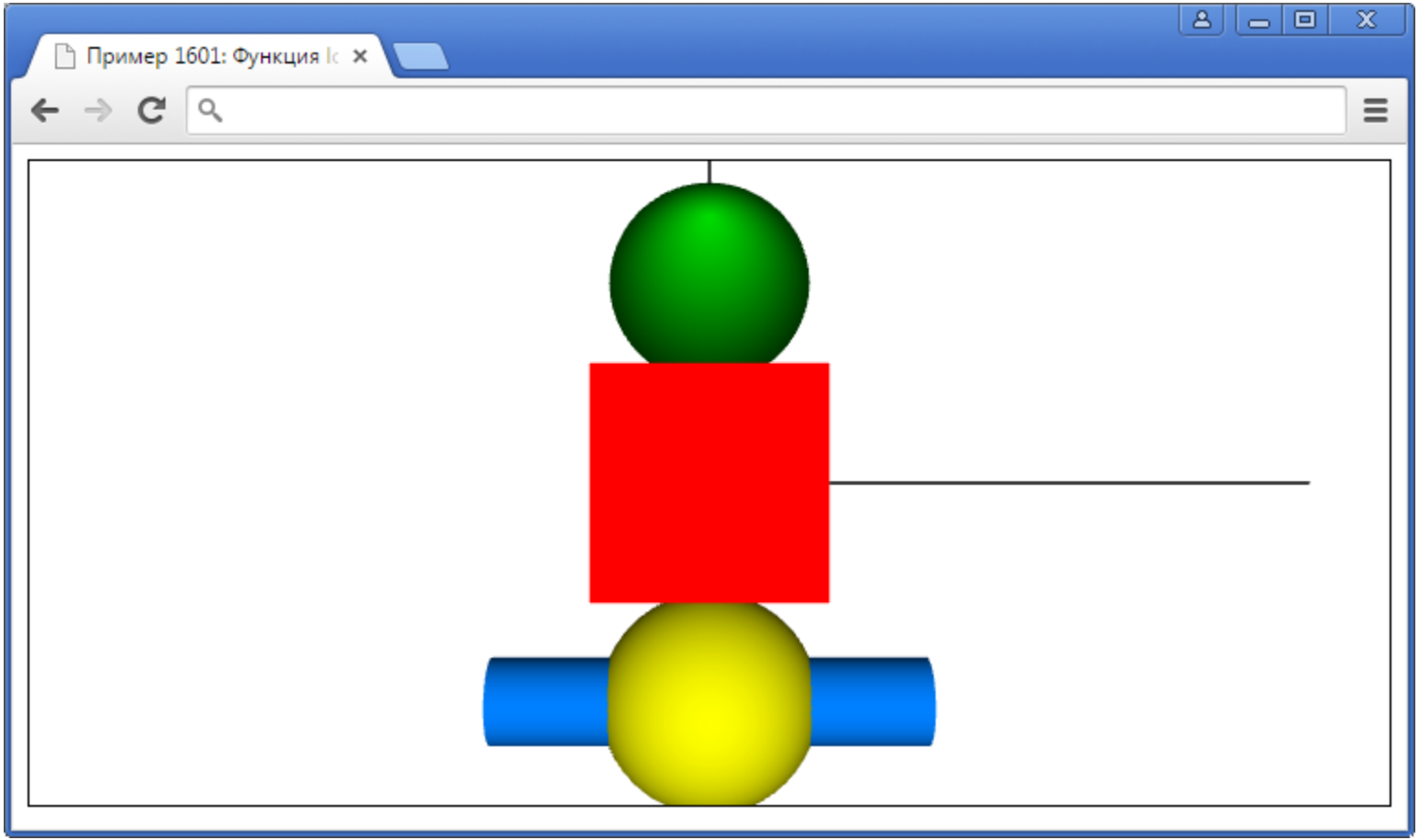
```
lookAt( [60,0,0], [0,0,0], [0,0,1] );
lookAt( [0,60,0], [0,0,0], [1,0,0] );
lookAt( [0,0,60], [0,0,0], [0,1,0] );
lookAt( [50,50,0], [0,0,0], [0,0,-1] );
```



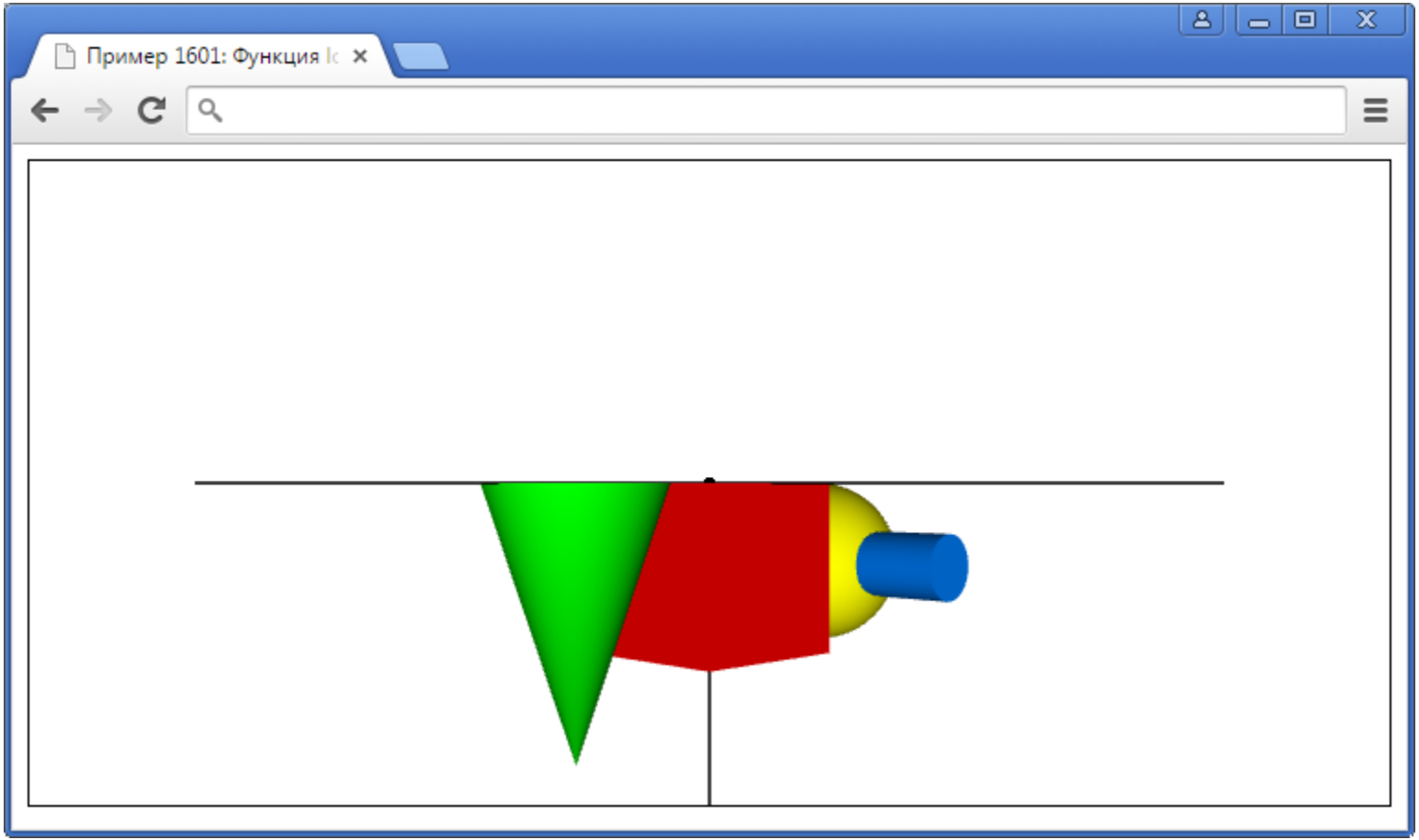
ПРОБА



ПРОБА



ПРОБА



ПРОБА

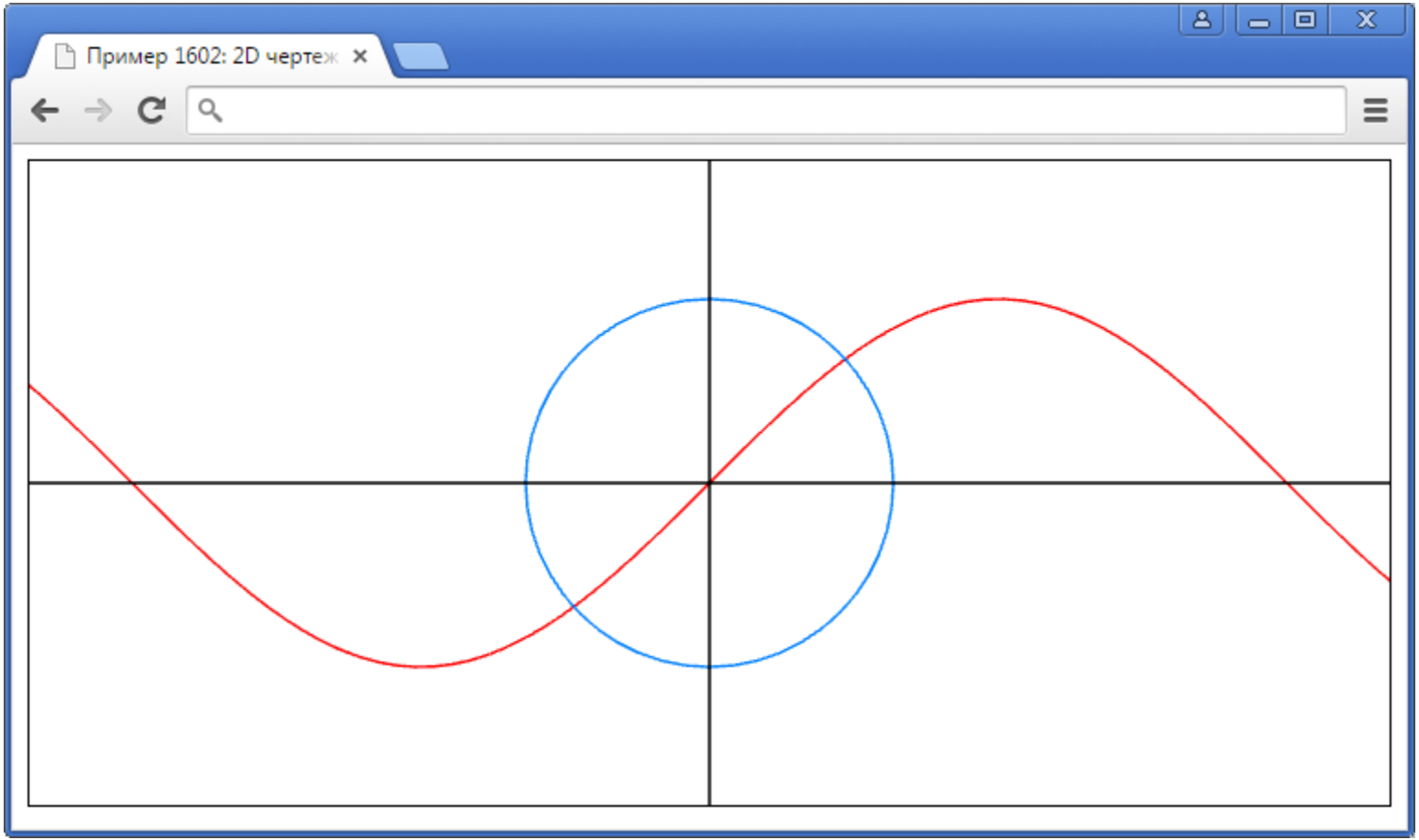
Традиционен 2D чертеж

Синусоида като в учебниците

- В средата е (0,0), оста X сочи надясно, а Y сочи нагоре
- Гледна точка с позиция по Z, цел в (0,0) и посока нагоре Y

```
orthographic(-100,100);
lookAt( [0,0,10], [0,0,0], [0,1,0] );

o = [0,0,0];
line(o,[1,0,0]).custom({color:[0,0,0]});
line(o,[0,1,0]).custom({color:[0,0,0]});
...
```



ПРОБА

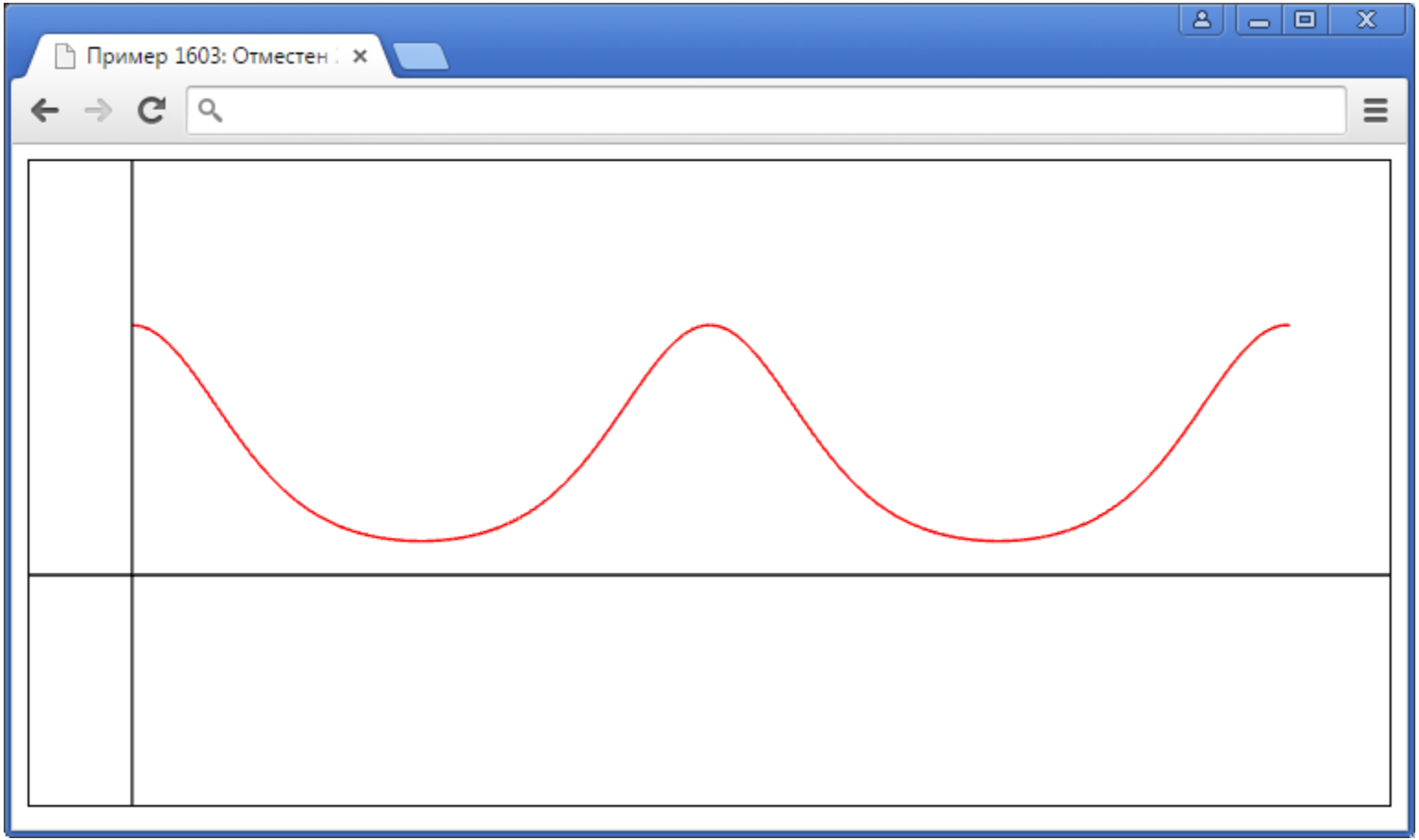
Графика на $f(x)=e^{\cos(x)}$ за $x \in [0,2\pi]$

- Позицията и целта са отместени заедно, за да може посоката на гледане да остане перпендикулярна на равнината XY

```
lookAt([100*Math.PI,50,10],[100*Math.PI,50,0],[0,1,0]);

function f(x) { return Math.exp(Math.cos(x)); }

dX = 0.1;
for (var x=0; x<4*Math.PI; x+=dX)
{
   p = [50*x,50*f(x),0];
   q = [50*(x+dX),50*f(x+dX),0];
   segment(p,q).custom({color:[1,0,0]});
}
```



ПРОБА

Анимация на гледната точка

Движение на гледната точка

Гледната точка като графичен обект

- Може да се променя с времето
- Създава илюзия за движение

  преместване на целия чертеж

  преместване на зрителя спрямо чертежа

Например

- Промяна на посоката се вижда като въртене

Следене на 2D чертеж

Графика на *sin*(*x*)

- Абсцисата е разграфена (до някъде)
- При рисуване се следи точката на рисуване

Потенциален проблем

- При създаване на много нови точки се получава забавяне

Решение

- Работим само с n точки
- Вместо да създаваме нова точка, преместваме някоя, излязла извън екрана, на желаното място в екрана

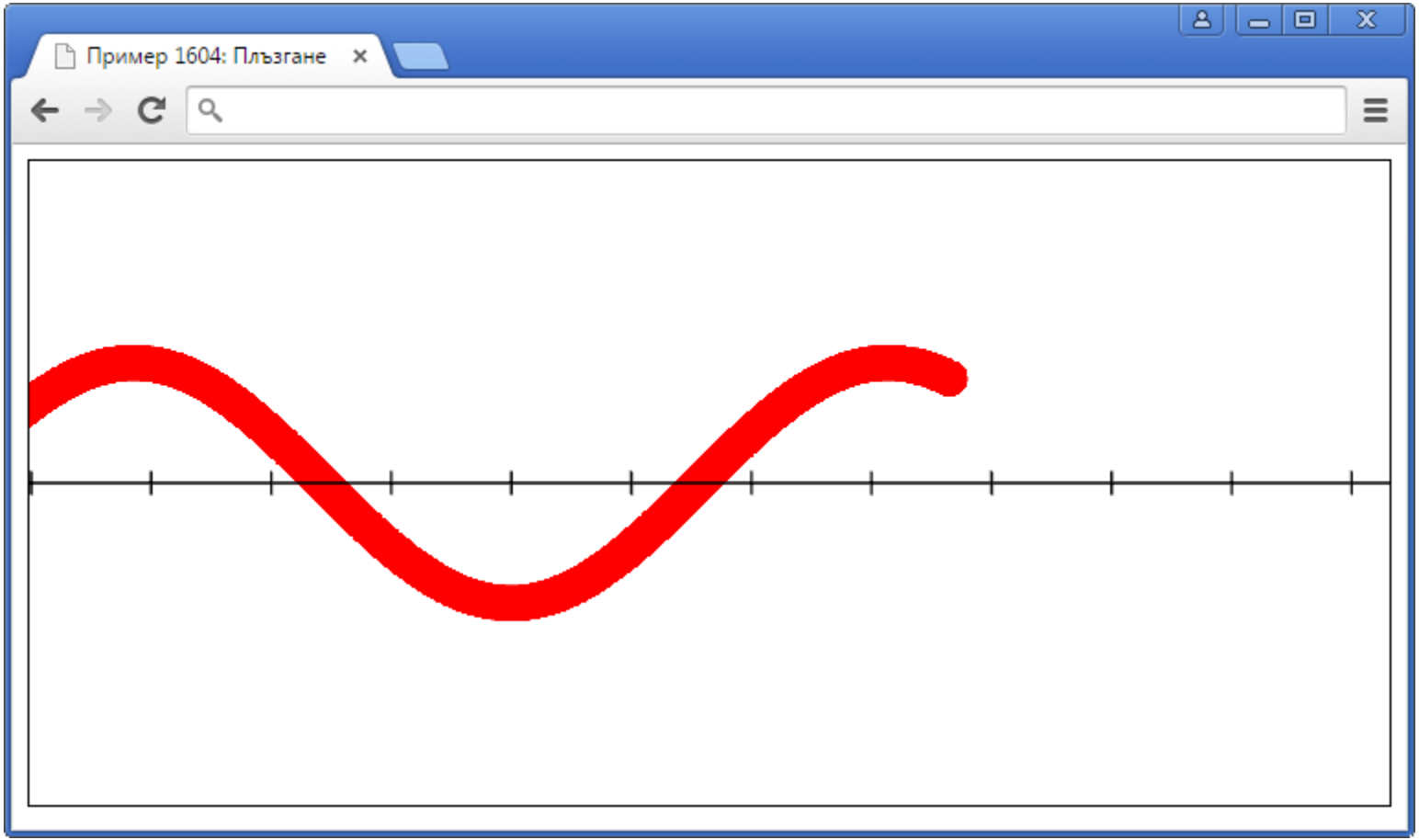
Реализация

- Масив от n предварително създадени точки

```
n = 160;
q = [];
for (var i=0; i<n; i++) q[i] = point([0,0,0])...
```

- Брояч i, който определя коя точка се генерира
- Гледаме на 2 единици зад генерираната точка
- С i%n определяме коя точка от масива променяме

```
x = i/20;
lookAt( [x-2,0,10], [x-2,0,0], [0,1,0] );
q[i%n].center = [x,Math.sin(x),0];
i++;
```



ПРОБА

Графика на спирала

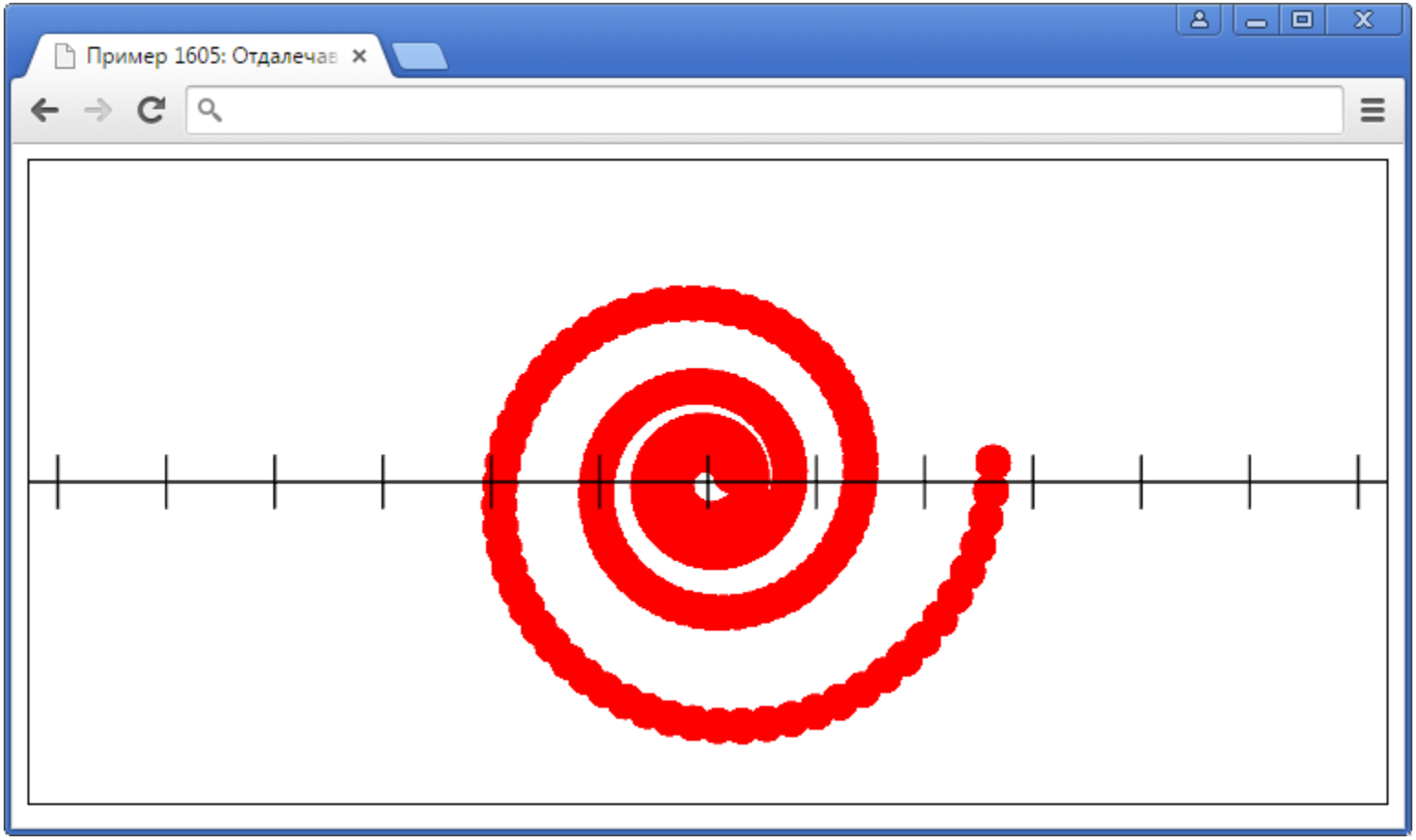
- Кръгово движение с променлив радиус r и ъгъл x
- Радиусът се увеличава плавно
- Позицията на гледната точка се отдалечава (зависи от r)

Въпрос

- Защо изведнъж графиката изчезва?

```
r = r*1.01;
lookAt( [0,0,1+4*r], [0,0,0], [0,1,0] );

x = i/10;
q[i%n].center = [r*Math.cos(x),r*Math.sin(x),0];
i++;
```



ПРОБА

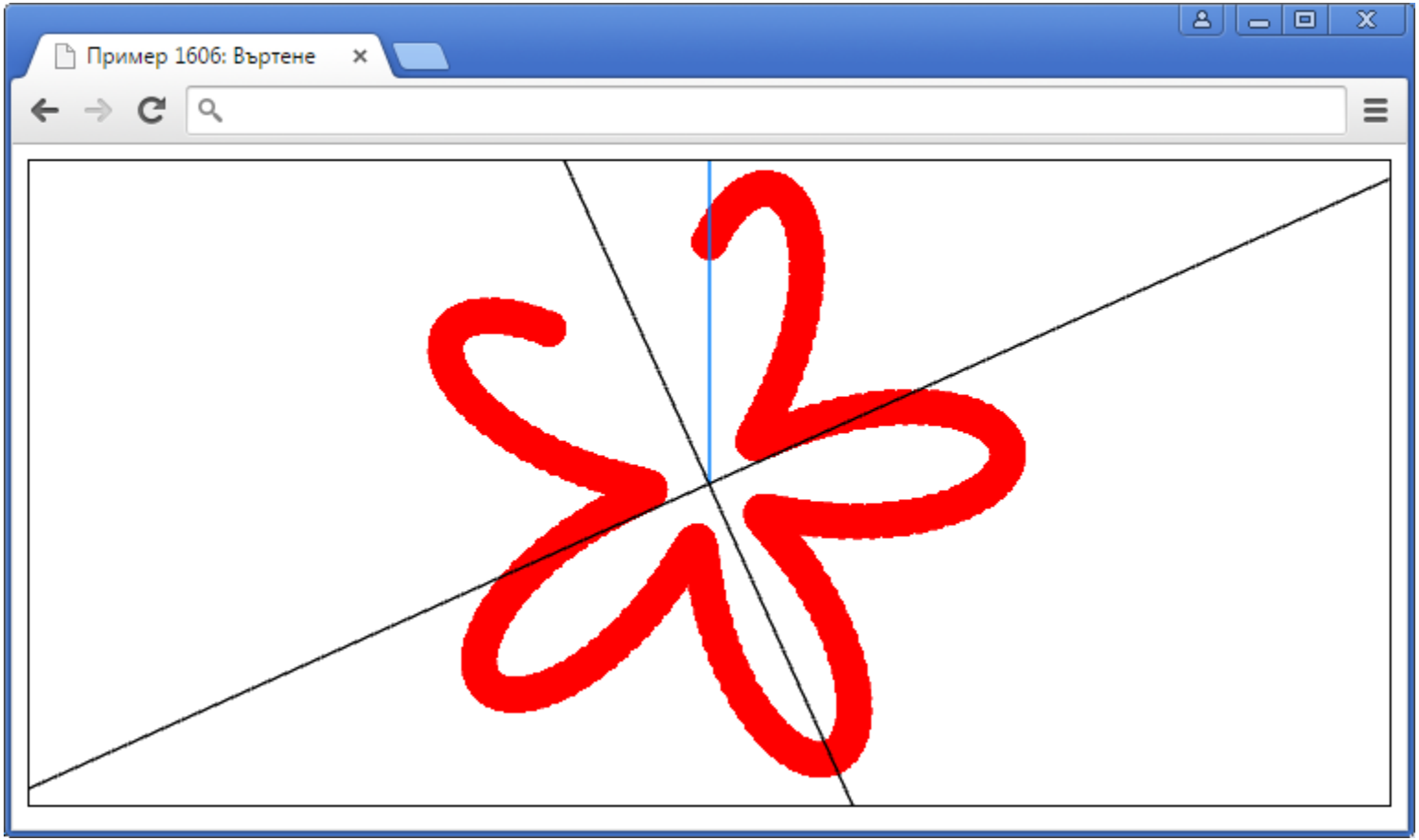
Графика на $r=3+2sin(5x)$

- При рисуване сцената се върти с промяна на посоката нагоре
- Мястото на рисуване е винаги „нагоре“
- Маркирано е с лъча l, чиято втора точка се върти заедно с въртенето на сцената

```
x = i/40;
r = 3+2*Math.sin(5*x);
l.to = [Math.cos(x),Math.sin(x),0];

lookAt([0,0,20],[0,0,0],[Math.cos(x),Math.sin(x),0]);

q[i%n].center = [r*Math.cos(x),r*Math.sin(x),0];
i++;
```



ПРОБА

Анимация в 3D

Въртене на сцена

Двойственост

- Различни движения с един и същ външен вид
- Понякога едното е много по-леко от другото

Пример

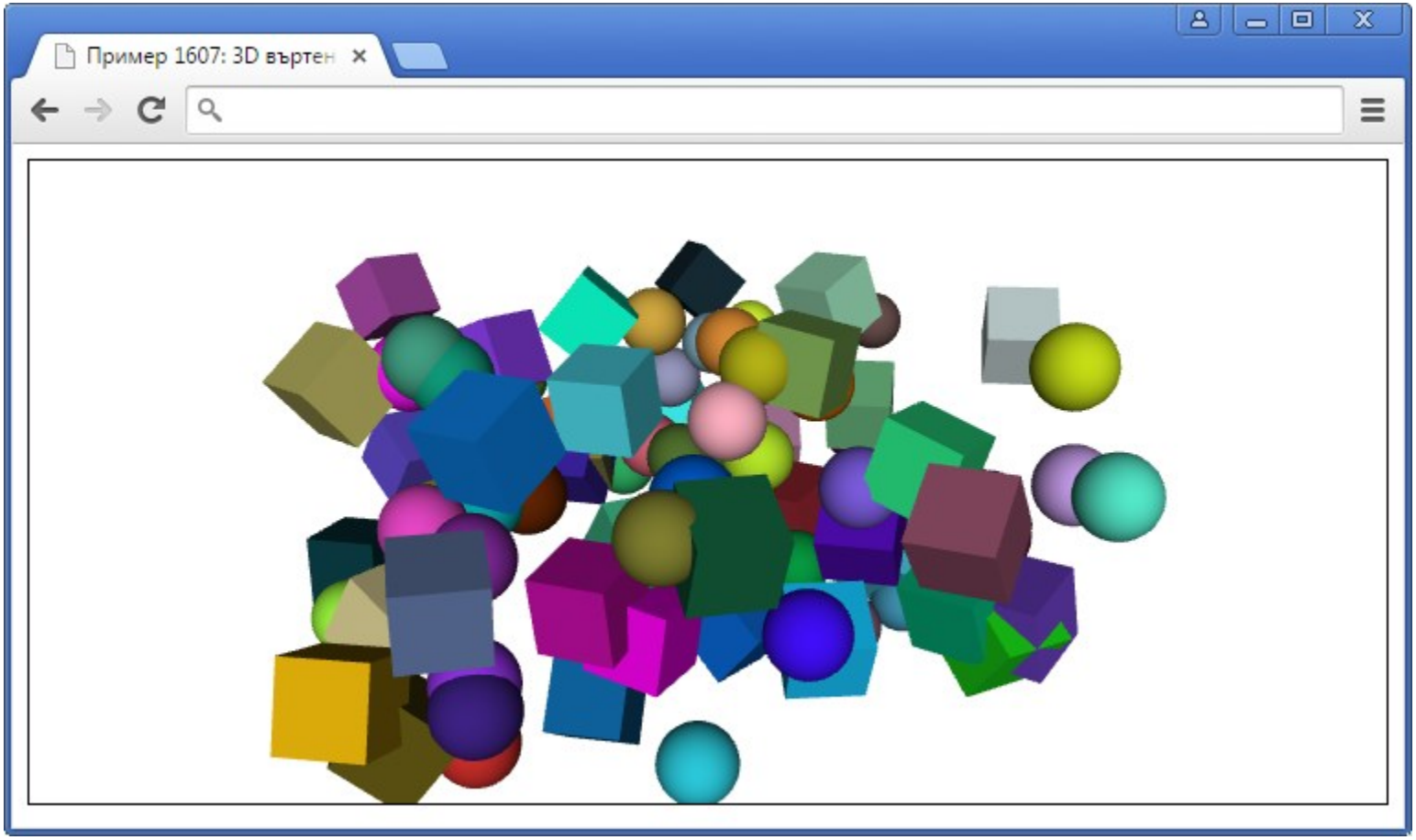
- Сцена от много обекти, всички се въртят около оста Z
- Неподвижна сцена, гледната точка се върти около оста Z

Въртене на сцена

Въртене на сцена чрез гледна точка

- Симулация на функцията demo
- Сцената е от неподвижни, разпръснати обекти
- Позицията на гледната точка се върти, целта е фиксирана

```
function rotateViewpoint()
{
   t = Suica.time;
   lookAt( [100*Math.cos(t),100*Math.sin(t),30],
                                    [0,0,0], [0,0,1] );
}
```

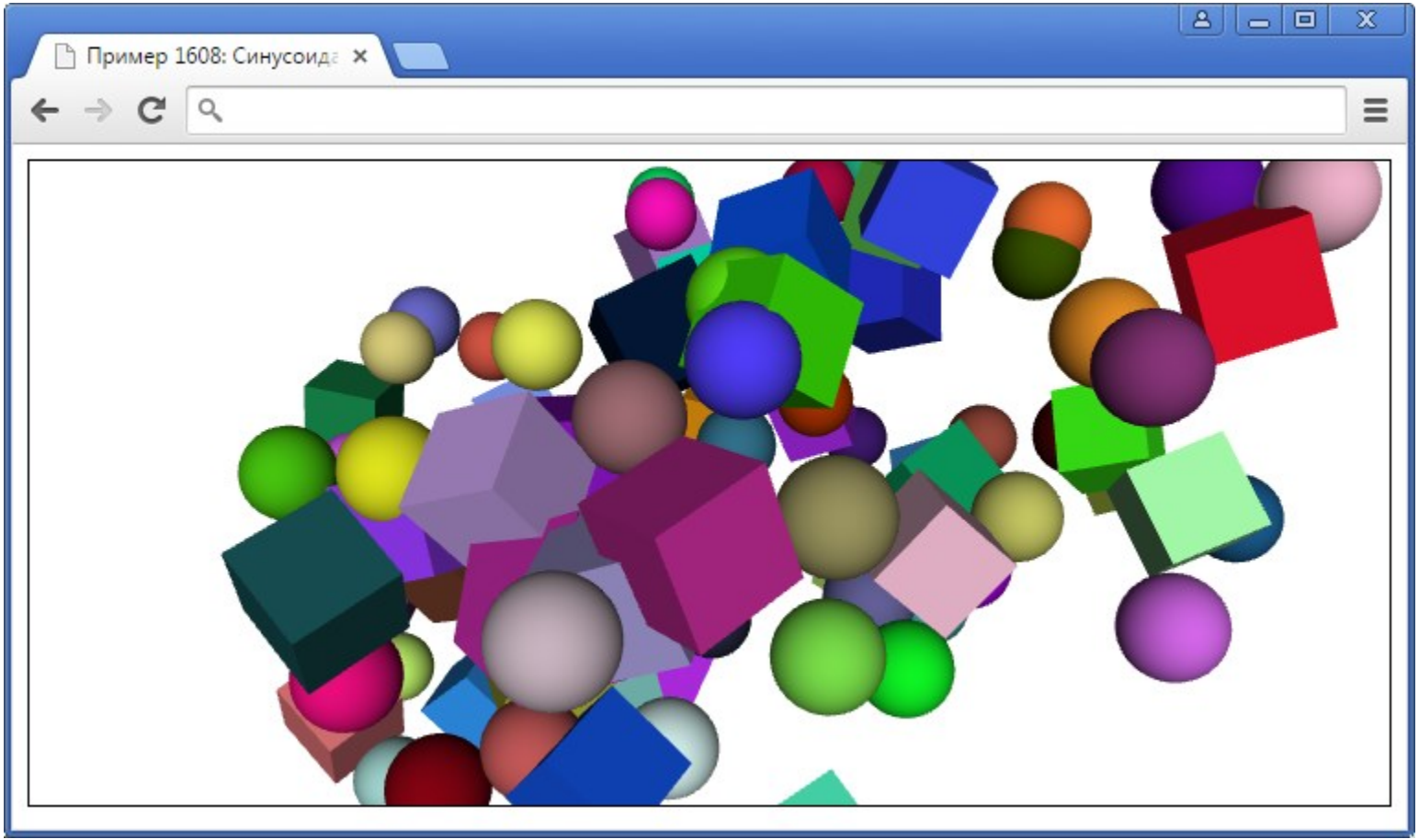


ПРОБА

Синусоидално въртене

- Гледната точка се върти около сцената
- Синусоидално се приближава и отдалечава от нея
- Има и леко поклащане чрез промяна на посоката нагоре (посоката е променлива, но се запазва почти вертикална)

```
function rotateViewpoint()
{
   t = Suica.time;
   d = 100+40*sin(2*t);
   lookAt( [d*cos(t),d*sin(t),30],
           [0,0,0],
           [sin(2*t),cos(3*t),2] );
}
```



ПРОБА

Примери

Преследване на обект

Сцена

- Квадратна мрежа от „постройки“
- Обект се движи по случаен начин в нея
- Гледната точка „преследва“ обекта

Идея

- Гледната точка ще има за цел обекта
- Позицията и целта няма да се променят рязко, а с линейна комбинация

Мрежа от постройки

- Кубоиди във възли с нечетни координати
- Всички са със случайна височина
- Цветовете им са случайно сини

```
for (var x=-12; x<11; x+=2)
for (var y=-12; y<11; y+=2)
{
   var c = random(0.5,1);
   cuboid([x+1,y+1,0],[0.8,0.8,random(0.4,2)]).custom({
      origin: [0,0,-0.5],
      color:  [0,c/2,c]
   });
}
```

Движение на обект

- Обектът ball е сфера
- Посоката на движение dir е ъгъл
- Параметърът k определя броя стъпки за изминаване на пътя от кръстовище до кръстовище (2 единици разстояние)

```
dir=0;
k=20;
function chase()
{
   ball.center[0] += 2/k*Math.cos(dir);
   ball.center[1] += 2/k*Math.sin(dir);
}
```

Завиване на обект

- Завиване се прави на всеки k стъпки (броят кадри е в променливата frame)
- При завой към посоката на движение добавяме -π/2, 0 или π/2 – така се завива наляво, или надясно, или се продължава напред

```
if (frame%k==0)
{
   dir += Math.PI/2*(Math.round(random(-1.5,1.5)));
   ball.center[0] = Math.round(ball.center[0]);
   ball.center[1] = Math.round(ball.center[1]);
}
```

Ограничение на движението

- Не искаме обектът да излиза извън рамките на „града“
- „Постройките“ са така разположени, че границите са ±10
- Зануляваме frame, за да се опита веднага нова посока

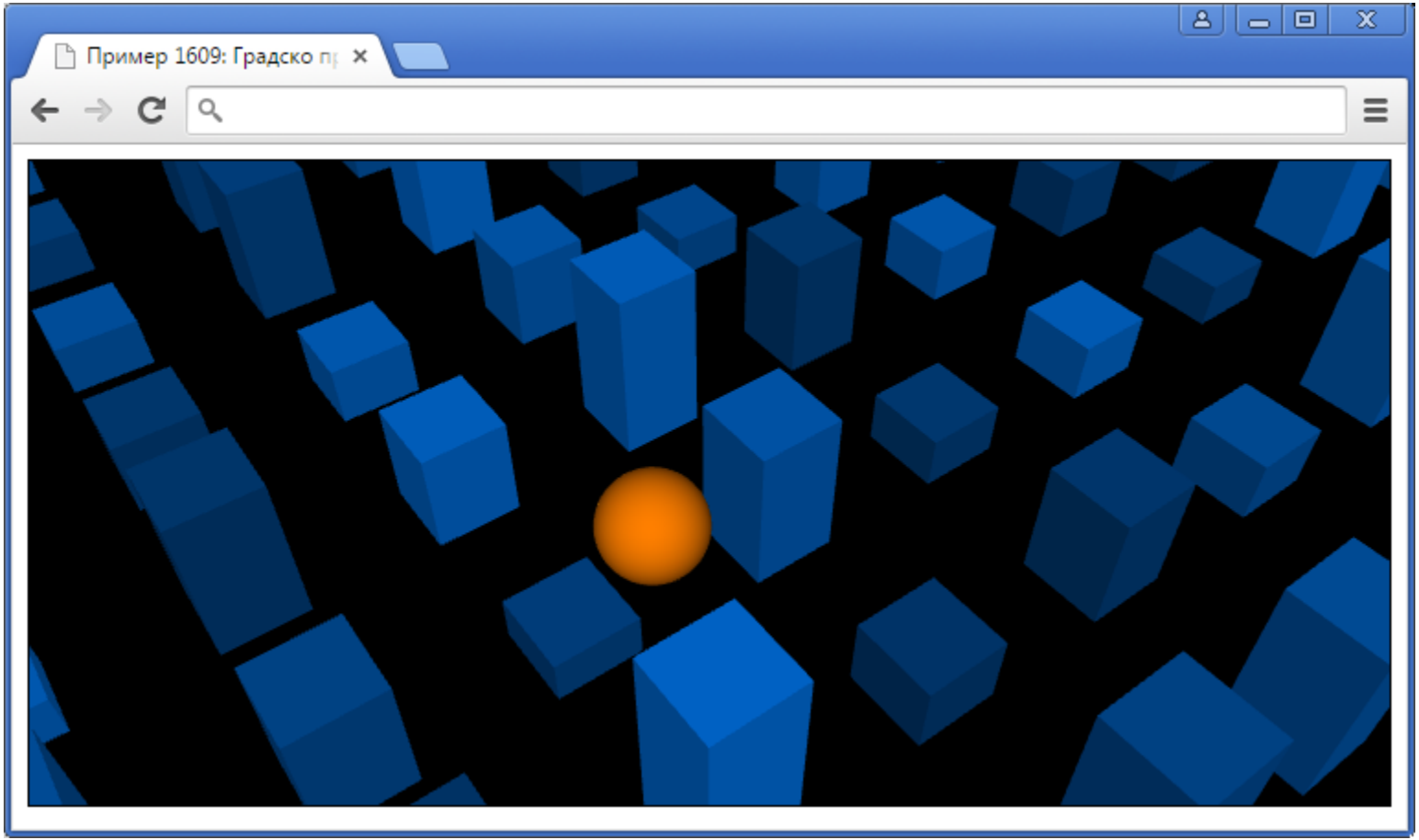
```
if (Math.abs(ball.center[0])>10 ||
    Math.abs(ball.center[1])>10 )
{
   ball.center[0] = Math.max(ball.center[0],-10);
   ball.center[0] = Math.min(ball.center[0],+10);
   ball.center[1] = Math.max(ball.center[1],-10);
   ball.center[1] = Math.min(ball.center[1],+10);
   frame = 0;
}
```

Движение на гледната точка

- Позицията и целта се променят плавно в посока към движещия се обект
- За да не са прекалено близки, позицията е отместена
- Коефициентите в линейните комбинации определят колко плавно ще се движи гледната точка

```
for (var i=0; i<3; i++)
{
   target[i] = target[i]*0.96+0.04*ball.center[i];
   pos[i] = pos[i]*0.98+0.02*ball.center[i]+0.3/(4-i);
}

lookAt(pos,target,up);
```



ПРОБА

Движение из сцена

Сцена

- Същата мрежа от „постройки“
- Гледната точка е спрямо обекта

Реализация

- Позицията ще съвпада с координатите на обекта (дори няма да е нужно да има обект)
- Посоката на движение и позицията определят целта

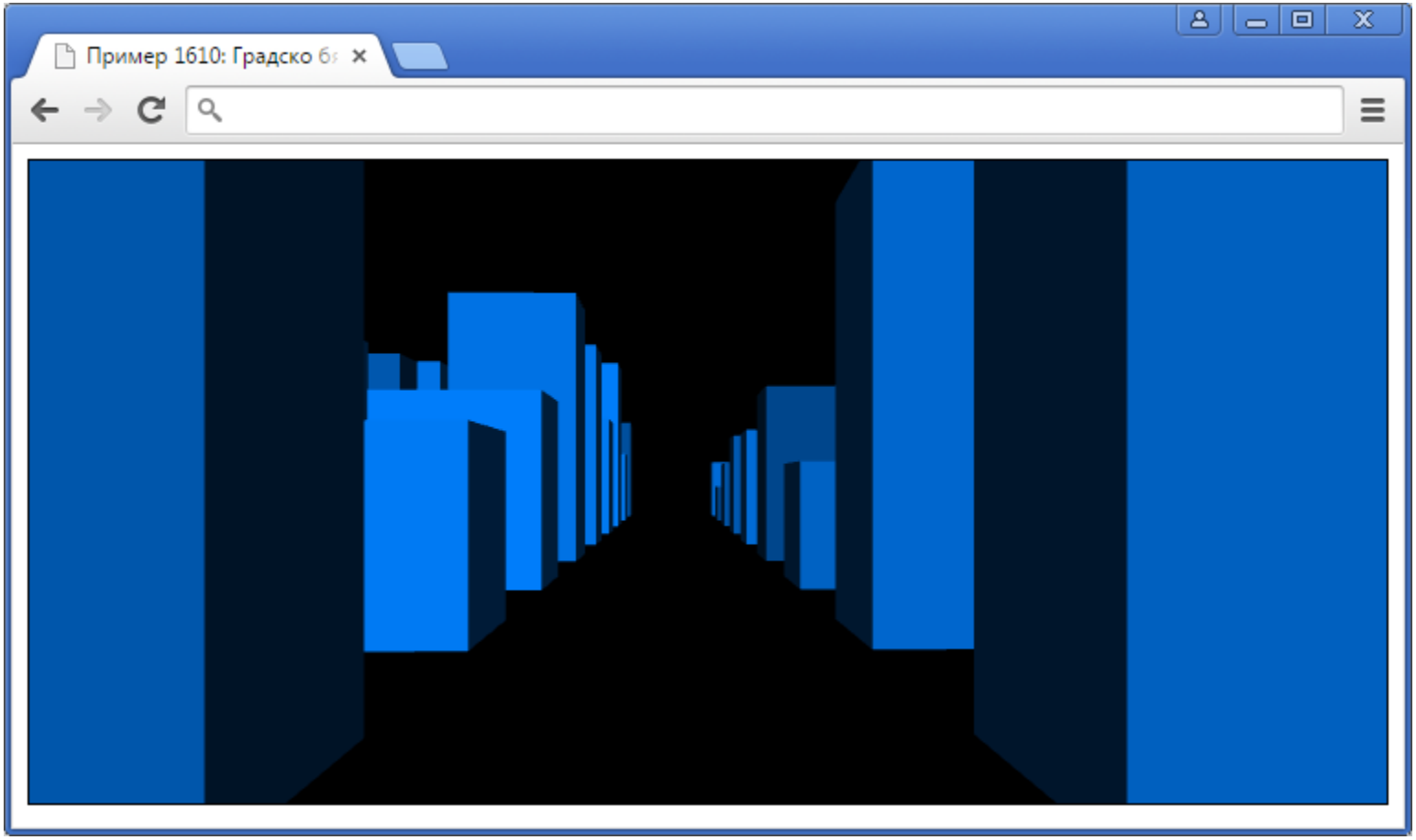
Заглаждане на движението

- Имаме втора позиция pos2 и посока dir2, които с линейна комбинация следват pos и dir
- Целта target2 е точка при кръгово въртене на ъгъл dir2 около център pos2

```
for (var i=0; i<3; i++)
   pos2[i] = pos2[i]*0.95+0.05*pos[i];

dir2 = dir2*0.95+0.05*dir;
target2 = [ pos2[0]+Math.cos(dir2),
            pos2[1]+Math.sin(dir2),0.5];

lookAt(pos2,target2,up);
```



ПРОБА

Обобщение

Гледна точка

Гледна точка

- Задава се с функцията lookAt

Параметри

- Позиция – тримерна точка на мястото, откъдето гледаме
- Цел – тримерна точка към която гледаме, тя се проектира в средата на графичното поле
- Посока нагоре – вектор, който за да се вижда вертикален, образът се завърта по подходящ начин

Движение на гледна точка

Постигани ефекти

- Илюзия за движение на цялата сцена с всичките ѝ обекти
- Реализация на оглеждане, което не е възможно с demo
- Движение вътре в сцена

Плавност

- Движението на гледна точка е по-чувствително към резки промени

ИКТ в НОС

Край

Коментари, въпроси