ЗАДАНИЕ 14

Броуновское движение

Открытие явления • В июне 1827 года британский ботаник Роберт Броун, исследуя водную взвесь пыльцы растения *Clarkia pulchella* под микроскопом, обнаружил[1], что мельчайшие частицы этой пыльцы совершают беспорядочные движения, интенсивность которых не меняется со временем. Броун провел большую серию опытов, пытаясь выяснить природу источника этого движения. Первоначальная гипотеза о том, что таким источником является некая «сила жизни», не подтвердилась после того, как Броун сначала обнаружил такое же движение частиц мертвых растений, а позже — и частиц всех неорганических веществ (например, стекла), которые он исследовал.

[1] R. Brown, *A brief account of microscopical observations made in the months of June, July and August, 1827, on the particles contained in the pollen of plants; and on the general existence of active molecules in organic and inorganic bodies*, Philosophical Magazine, 1828, 4: p. 161–173.

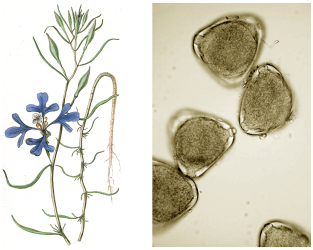

РИС. 14.1 *Clarkia pulchella* и его пыльца

РИС. 14.2 Роберт Броун

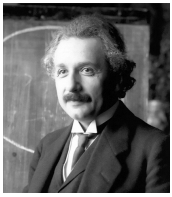

РИС. 14.3 Альберт Эйнштейн

РИС. 14.4 Жан Батист Перрен

Объяснение истинной природы источника броуновского движения, которым является тепловое движение молекул жидкости, было дано только через 80 лет после его открытия несколькими учеными, одним из которых был Альберт Эйнштейн. В 1905 году он публикует статью[2], в которой описывает возможный механизм хаотического движения частиц с точки зрения молекулярно-кинетичес-

[2] A. Einstein, *Uber die von der molekularkinetischen Theorie der Warme geforderte Bewegung von in ruhenden Flussigkeiten suspendierten Teilchen*, Ann. Phys., 1905, 322, p. 549–560.

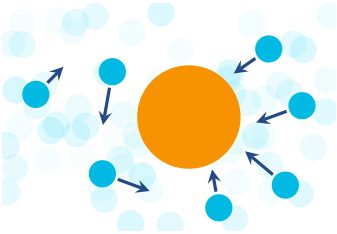

РИС. 14.5 Броуновское движение частицы

кой теории. Причиной броуновского движения по Эйнштейну является тепловое движение молекул воды, приводящее в результате столкновений с частицами вещества к случайным движениям этих частиц (рис. 14.5). Исходя из построенной им модели, Эйнштейн сделал несколько количественных оценок относительно движения взвешенных в жидкости частиц. В частности, он показал, что средний квадрат смещения частицы от ее начальной позиции в проекции на любую прямую пропорционален времени блуждания t:

$$\langle x^2 \rangle = 2Dt, \qquad (14.1)$$

где D — это коэффициент диффузии. Эйнштейн призвал экспериментаторов проверить эти оценки, что позволило бы подтвердить или опровергнуть его теоретические рассуждения.

Таким экспериментатором оказался Жан Перрен. В 1908–1909 годах он со своими студентами провел серию опытов над частицами смолы мастикового дерева и гуммигута (густого млечного сока деревьев рода гарциния), которые полностью подтвердили теорию Эйнштейна и, кроме того, позволили экспериментально, используя найденный коэффициент диффузии D, оценить значения постоянных Больцмана[3] и Авогадро[4]. На рис. 14.6 воспроизведена копия чертежа Перрена из его книги[5].

Случайные блуждания • Броуновское движение по своей сути является процессом *детерминированным*. Кажущаяся хаотичность траекторий броуновских частиц обусловлена (по крайней мере, теоретически) начальным случайным распределением молекул воды (или другого растворителя) и их скоростей. Тем не менее это явление послужило прообразом многих *вероятностных* моделей, известных как *модели случайных блужданий*. Примером одной их таких моделей является задача о лапше Бюффона, рассмотренная нами в упражнениях к главе 1.

Суть любой модели случайных блужданий заключается в моделировании неких условных частиц, совершающих в дискретные моменты времени случайные перемещения в некотором математиче-

[3] Физическая постоянная, определяющая связь между температурой и энергией. Названа в честь австрийского физика Людвига Больцмана, сделавшего большой вклад в статистическую физику, в которой эта постоянная играет ключевую роль.

[4] Физическая величина, численно равная количеству молекул (или других структурных единиц) в одном моле вещества.

[5] J. Perrin, *Atoms*, London: Constable & Company, 1916.

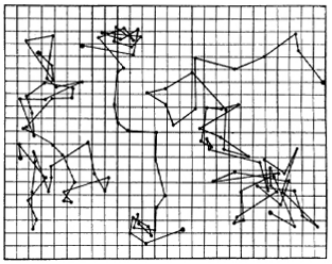

РИС. 14.6 Траектории броуновского движения трех частиц в опытах Перрена

ском пространстве. Пространства, в которых движутся частицы, могут быть самыми разнообразными — непрерывными и дискретными, произвольной размерности и топологии. При этом каждый шаг перемещения частицы является локальным, т.е. частица меняет свое положение незначительно (движется с ограниченной скоростью).

Модели случайных блужданий с успехом применяются практически во всех областях прикладной математики — при моделировании разнообразных систем и процессов в физике, химии, биологии, социологии, информатике и т.д. В последующих главах нами будет рассмотрено несколько моделей случайных блужданий — задача о разорении игрока, метод имитации отжига, алгоритм бактериального поиска, модели диффузии. А пока мы займемся построением простой модели самого броуновского движения.

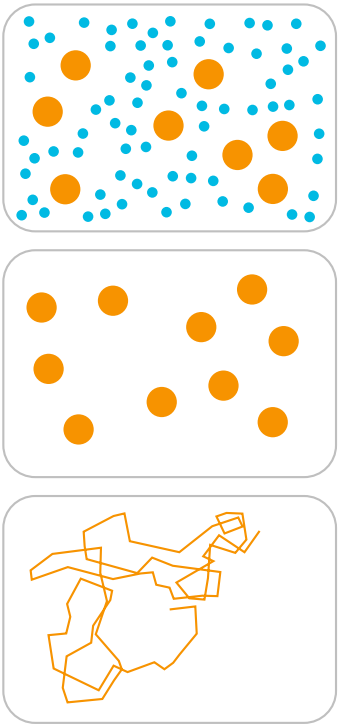

РИС. 14.7 Три режима просмотра модели броуновского движения

Построение модели

Нашей целью является смоделировать процесс плоского движения с учетом столкновений частиц двух типов — больших и тяжелых (броуновские частицы) и мелких и легких (молекулы воды). Модель должна поддерживать три режима просмотра (рис. 14.7):

- *полный* — показывается движение всех типов частиц;
- *броуновский* — показывается движение только броуновских частиц;
- режим просмотра *траекторий* броуновских частиц.

A Создаем новую модель. Устанавливаем режим просмотра on ticks.

B Добавляем к интерфейсу два слайдера, связанных с глобальными переменными:

- `heavy-particles` — количество тяжелых (броуновских) частиц;
- `light-particles` — количество легких частиц (молекул воды).

C В среде NetLogo стандартным видом подвижных агентов являются черепахи. Пользователь, однако, может создавать свои собственные виды агентов с помощью встроенной команды `breed`. Пусть

в нашей модели активными агентами будут частицы. Для этого в начале кода надо указать команду `breed [particles particle]`. Двумя ее аргументами являются названия нового вида черепах во множественном и единственном числе.

D Агенты нового вида «наследуют» все атрибуты черепах — размер, цвет, форму и т.д. Нам потребуется добавить несколько своих атрибутов, специфичных для рассматриваемой модели. Во-первых, это масса `mass` частицы, во-вторых, ее мгновенная скорость `speed` (другими словами, длина вектора скорости). Новые атрибуты добавляются к любому виду агентов с помощью команд типа `turtles-own`, первое слово в имени команды — вид агентов во множественном числе. В нашем случае мы пишем второй строкой кода команду

```
particles-own [mass speed].
```

E Добавляем к интерфейсу модели кнопку `setup` и создаем соответствующую процедуру `setup`, в которой:

- очищаем модель (`clear-all`);
- просим все патчи сменить цвет на белый;
- сбрасываем таймер (`reset-ticks`).

F Создаем процедуру `setup-particle`, отвечающую за настройку атрибутов новой частицы. Аргументами этой процедуры являются размер частицы `s`, ее цвет `c`, масса `m` и скорость `v`. В самой процедуре устанавливаем соответствующие четыре атрибута частицы: размер `size` и т.д. Кроме того, устанавливаем круглую форму частицы и располагаем ее в случайном месте модели.

G В процедуре `setup` перед сбросом таймера создаем `heavy-particles` броуновских частиц, каждую из которых настраиваем с помощью вызова `setup-particle`. Затем аналогичным образом создаем заданное количество легких частиц.

```
create-particles heavy-particles [
  setup-particle 2.4 orange 9 0
]
create-particles light-particles [
  setup-particle 0.8 sky 1 0.1
]
```

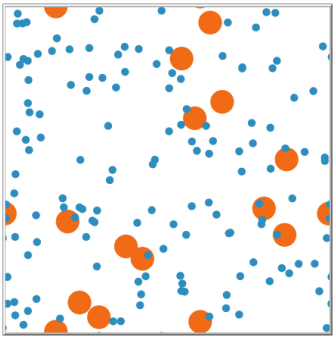

РИС. 14.8 Начальная конфигурация модели, в которой часть частиц накладывается друг на друга

H Тестируем работу кнопки setup (рис. 14.8). Если число частиц достаточно велико, то часть из них будет накладываться друг на друга. Это затруднит нам в будущем корректную обработку столкновений частиц, поэтому требуется более тщательное их начальное расположение, которое будет выполняться процедурой place-particle.

I Сначала определим вспомогательную функцию get-dist [p], которая вычисляет «расстояние» от текущей[6] частицы до частицы p с учетом размеров этих частиц, т.е. расстояние между центрами минус радиусы этих двух частиц[7]. Определенное таким образом расстояние может оказаться *отрицательным*, если частицы хотя бы частично наложены друг на друга[8].

```
1 to-report get-dist [p]
2   report distance p - (size + [size] of p) / 2
3 end
```

Эта же функция будет использоваться нами далее при определении столкновений частиц.

J В процедуре setup после создания всех частиц просим их выполнить процедуру place-particle. Создаем процедуру place-particle, в коде которой *текущая* частица должна проверить, не пересекается ли она с другими частицами. Если пересекается, то переносим ее в случайное место и повторяем проверку и т.д. Для реализации этой схемы нам потребуется цикл while[9]. Код процедуры представлен ниже.

```
1 to place-particle
2   let o-p other particles
3   let p min-one-of o-p [get-dist myself]
4   while [get-dist p < 0.1] [
5     setxy random-xcor random-ycor
6     set p min-one-of o-p [get-dist myself]
7   ]
8 end
```

Во второй строке с помощью функции other мы создаем набор частиц[10] o-p, в который не включена текущая частица. Команда min-one-of в строке 3 принимает два аргумента: первый — набор агентов, второй в квадратных скобках — выражение, которое эти агенты должны вычислить. Функция возвращает агента с минимумом этого выражения. В нашем случае мы просим все частицы вычислить

[6] Везде далее будем называть *текущим* агента, который выполняет заданный код процедуры или функции.

[7] Радиус частицы равен половине ее размера size.

[8] Функция get-dist не может служить расстоянием в строгом смысле этого слова, т.к. для нее не выполняются два обязательных свойства любого расстояния ρ: расстояние от a до a равно нулю ($\rho(a,a) = 0$) и неравенство треугольника $\rho(a,b) \le \rho(a,c) + \rho(c,b)$.

[9] *При использовании в коде циклов* while *крайне рекомендуется перед запуском модели выполнять ее сохранение. Прервать работу модели при ее зацикливании можно, используя только команду меню* Tools → Halt, *однако это не всегда удается сделать. Поэтому вы рискуете потерять последние сделанные вами изменения в модели!*

[10] Agentset в терминологии среды NetLogo.

[11] Команда self обозначает текущего агента (тот, кто выполняет данное действие, синоним местоимения *я*). А команда myself означает: *агент, который попросил меня выполнить данное действие.*

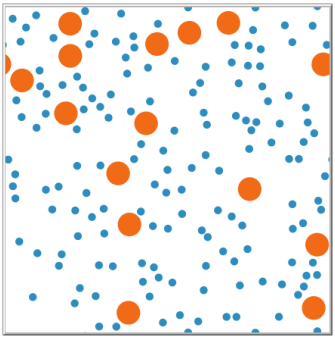

РИС. 14.9 Начальная конфигурация модели, в которой частицы не накладываются друг на друга

расстояние до текущей частицы, которая указывается командой `myself`[11]. Таким образом, частица `p` — ближайшая к текущей частице в смысле определенного нами расстояния. Оператор `while` (строки 6–7) выполняет свой код (во второй паре квадратных скобок) до тех пор, пока логическое условие в первой паре скобок остается истинным. В данном случае мы проверяем, чтобы расстояние от текущей частицы до ближайшей к ней было больше 0.1 (т.е. частицы не должны даже касаться друг друга). Проверяем работу кнопки `setup`, модель теперь должна выглядеть, как на рис. 14.9.

K Добавляем к интерфейсу кнопку `go` и создаем соответствующую процедуру, в которой просим все частицы переместиться на расстояние, определяемое их скоростью, и обновляем таймер.

```
1 to go
2   ask particles [fd speed]
3   tick
4 end
```

Проверяем работу новой кнопки, легкие частицы начнут двигаться с одинаковой скоростью, тяжелые — останутся неподвижными. При этом частицы, не сталкиваясь, свободно проходят друг сквозь друга.

L Добавляем к интерфейсу модели выпадающий список с именем `mode` со следующими тремя опциями: `"full"`, `"brownian"` и `"traces"`. В процедуру `go` перед обновлением таймера вставим код, устанавливающий текущий режим просмотра. Если выбран режим `full`, то командой `clear-drawing` удаляем все траектории, которые, возможно, были нарисованы до этого, и просим все частицы: 1) сделаться видимыми с помощью команды

```
set hidden? false;
```

2) поднять перо командой `penup` (или `pu`). Если включен режим просмотра `brownian`, то также удаляем траектории, просим частицы: 1) установить атрибут `hidden?` равным `color = sky` (такой цвет в нашей модели имеют легкие частицы, которые и будут в результате скрыты); 2) поднять перо. Если же выбран режим `traces`, то просим все частицы сделаться скрытыми, после чего просим частицы

цвета `orange` опустить перо (команда `pendown` или `pd`). Последнее достигается командой

```
ask particles with [color = orange] [pd].
```

Команда `with` оставляет в заданном наборе агентов только тех из них, для которых выполнено условие в скобках. Проверяем переключение режимов просмотра[12].

M Приступаем к обработке столкновений частиц. При каждом вызове процедуры `go` будем обрабатывать только одно возможное столкновение каждой частицы. Для этого добавим к атрибутам частиц логическую переменную `collided?`. В начале процедуры `go` просим все частицы установить этот атрибут равным `false` (т.е. частица еще не участвовала в столкновении). Следующей командой просим каждую частицу выполнить процедуру проверки столкновения `check-collide`.

N Создаем процедуру `check-collide`. В ее коде сначала проверяем, если частица уже поучаствовала в столкновении с другой частицей (`collided? = true`), то останавливаем процедуру (команда `stop`). После этого находим ближайшую частицу, как это было сделано в процедуре `place-particle`. Если расстояние до этой частицы меньше нуля, то, значит, мы имеем столкновение, которое будем обрабатывать командой `collide-with p`.

```
1 to check-collide
2   if collided? [stop]
3   let o-p other particles
4   let p min-one-of o-p [get-dist myself]
5   if get-dist p < 0 [collide-with p]
6 end
```

O Пишем код процедуры `collide-with`. Обозначим для краткости массу и скорость частицы, с которой сталкивается текущая частица, `mp` и `sp`. Изменения скоростей частиц после их столкновения можно рассчитать из законов сохранения импульса и энергии, предполагая, что соударение является абсолютно упругим[13]. Важным направлением при столкновении частиц является прямая, соединяющая центры частиц (рис. 14.10). Угол ее наклона θ к оси Oy[14] (в градусах) можно получить командой `towards`. Атрибут `heading` задает текущее направление скорости частицы (т.е. угол наклона векто-

[12] Чтобы увидеть траектории тяжелых частиц, можно временно сделать ненулевой их скорость в процедуре `setup`.

[13] Д. В. Сивухин, *Общий курс физики. Механика*, М., 1979, §28.

[14] В среде NetLogo все углы отсчитываются от направления на север (вверх), т.е. от оси Oy. Положительными при этом считаются углы, направленные *по часовой стрелке*.

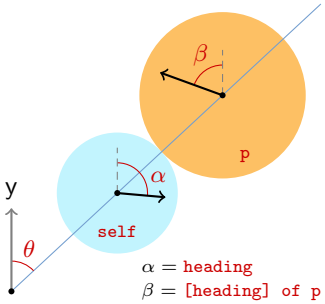

РИС. 14.10 Система углов в процедуре `collide-with`

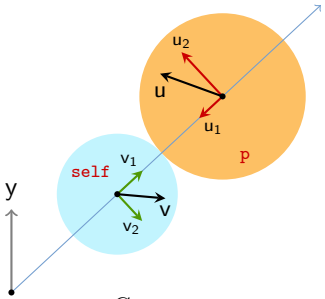

РИС. 14.11 Схема разложения скоростей частиц в процедуре `collide-with`

ра скорости к оси Oy). Разложим скорость каждой частицы (v и u) на две компоненты — вдоль выделенного направления (v_1 и u_1) и перпендикулярно ему (v_2 и u_2), как показано на рис. 14.11. При упругом столкновении меняются только проекции скоростей вдоль прямой, соединяющей центры частиц. Чтобы их найти, сначала определяем проекцию скорости v_c центра масс системы на эту прямую:

$$v_c = \frac{m_1 v_1 + m_2 u_1}{m_1 + m_2}, \qquad (14.2)$$

а затем обновляем v_1 и u_1 согласно формулам

$$\tilde{v}_1 = 2v_c - v_1, \quad \tilde{u}_1 = 2v_c - u_1. \qquad (14.3)$$

Теперь вычисляем новые направление и величину скорости каждой частицы (с помощью процедуры `update-particle`, которую мы рассмотрим ниже). Код процедуры `collide-with`, реализующей описанную схему, выглядит следующим образом.

```
to collide-with [p]
  let mp [mass] of p
  let sp [speed] of p
  let theta towards p
  let v1 speed * cos (theta - heading)
  let v2 speed * sin (theta - heading)
  let u1 sp * cos (theta - [heading] of p)
  let u2 sp * sin (theta - [heading] of p)
  let vc (mass * v1 + mp * u1) / (mass + mp)
  set v1 2 * vc - v1
  set u1 2 * vc - u1
  update-particle v1 v2 theta
  ask p [update-particle u1 u2 theta]
end
```

P В процедуре `update-particle`, которая принимает первыми двумя аргументами компоненты скорости `v1` и `v2` относительно направления, заданного углом `theta` (третий аргумент), сначала вычисляем величину скорости. После этого, если скорость ненулевая[15], определяем новое направление, используя встроенную функцию `atan`[16]. Последней командой запоминаем, что текущая частица имела столкновение на данном такте работы модели, чтобы исключить ее столкновения с другими частицами на этом же такте.

```
to update-particle [v1 v2 theta]
  set speed sqrt (v1 ^ 2 + v2 ^ 2)
  if speed > 0 [
```

[15] Для нулевой скорости ее направление не имеет смысла.

[16] В NetLogo функция `atan x y` возвращает угол (в градусах, в диапазоне от 0° до 360°) наклона вектора с координатами `(x,y)` к горизонтальной оси.

РИС. 14.12 Окончательный ин-

```
    set heading theta - (atan v2 v1)
  ]
  set collided? True
end
```

Q Проверяем работу модели. На рис. 14.12 показан окончательный вид модели в режиме просмотра траекторий броуновских частиц для случая одной броуновской частицы.

УПРАЖНЕНИЯ

1 Если внимательно проследить за динамикой системы в полном режиме просмотра, то можно увидеть, что через некоторое время некоторые частицы в модели сцепляются в пары (рис. 14.13), которые будем называть *спиннерами*, потому что они обладают ярко выраженным вращательным движением. Эффект этот интересен тем, что он совершенно не закладывался нами в модель. Возникает этот эффект, когда частицы сразу после столкновения не успевают разойтись на положительное расстояние (в нашем смысле этого слова) и на следующем шаге опять «сталкиваются», что приводит к смене направления и очередному столкновению на следующем шаге. С точки зрения моделирования броуновского движения этот эффект все-таки является нежелательным. Простейший способ снизить частоту появления спиннеров заключается в уменьшении шага времени, что приведет к уменьшению расстояния, на которое перемещаются частицы за один такт, и соответствующему снижению некорректно обрабатываемых столкновений. Добавьте к интерфейсу слайдер с именем `dt` и диапазоном изменения от 0.1[17] до 1.0. Команду движения `fd speed` замените на `fd speed * dt` (расстояние — это скорость, умноженная на время). Наконец, вместо команды `tick`, которая увеличивает таймер на 1, используйте команду `tick-advance dt`, которая увеличивает таймер на значение `dt`. Перерисовка модели при этом происходит только в целочисленные моменты времени.

РИС. 14.13 Спиннеры

2 Второй способ снизить частоту появления спиннеров состоит в запоминании каждой частицей в отдельном атрибуте (например, `lp`) последней столкнувшейся с ней частицы, и при детекции столкновения запомненную частицу исключать из списка кандидатов на столкновение:

```
let o-p other particles with [lp != myself].
```

[17] Меньшие значения не стоит использовать, т.к. система не будет успевать пересчитывать модель с заданной частотой, которая по умолчанию равна 30 кадрам в секунду.

При обработке столкновения требуется аккуратное обновление этого атрибута как самой частицы, так и частицы, запомненной в `lp`.

3 В библиотеке моделей NetLogo имеется семейство моделей GasLab, посвященных моделированию различных процессов в газах[18]. В основе этих моделей лежит как раз динамика столкновений частиц. Исследуйте поведение и устройство модели GasLab Circlular Particles, в которой производятся корректная детекция и обработка столкновений частиц разных размеров, основанная на вычислении времени ближайшего столкновения.

[18] Меню File → Models Library.

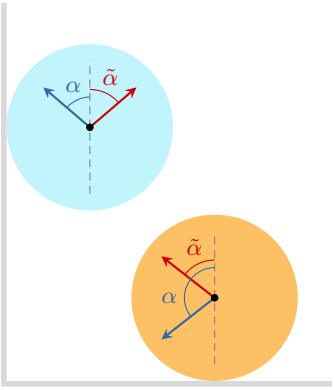

РИС. 14.14 Изменение направления движения при столкновении частицы со стенками: α — угол падения, $\tilde{\alpha}$ — угол отражения.

4 Добавьте в модель столкновения со стенками, которые располагаются на границах модели. В процедуре `go` после обработки столкновений между частицами попросите все частицы выполнить проверку на столкновение со стенками. Проверку и саму обработку столкновения, если оно произошло, выполните в процедуре `collide-with-wall`. Правило изменения скорости в данном случае оказывается простым: угол отражения должен быть равен углу падения, т.е. при столкновении с вертикальной стенкой меняет знак горизонтальная компонента скорости, при столкновении с горизонтальной стенкой — вертикальная компонента скорости. Если `heading` — это текущее направление движения, то после столкновения с правой или левой стенками это направление нужно установить равным `(- heading)`, а после столкновения с верхней или нижней стенками — `(180 - heading)` (рис. 14.14).

5 В нашей модели изменение скоростей при столкновении частиц производится с учетом закона сохранения кинетической энергии. Таким образом, теоретически кинетическая энергия системы частиц должна оставаться неизменной. Добавьте к интерфейсу модели монитор, который будет показывать полную энергию системы. Чтобы вычислить сумму некоторой величины для всех агентов, можно использовать команду `sum`. В нашем случае полная кинетическая энергия определяется выражением:

```
sum [0.5 * mass * speed ^ 2] of particles.
```

Убедитесь, что энергия в целом остается постоянной, но все-таки претерпевает небольшие флуктуации, которые обусловлены погрешностью округления при выполнении арифметических действий над действительными числами.

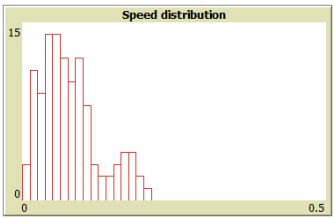

РИС. 14.15 Гистограмма распределения скоростей частиц

6 Добавьте к интерфейсу модели график, показывающий распределение скоростей частиц (рис. 14.15). Для

этого в свойствах графика надо указать в поле Pen update commands команду рисования

```
histogram [speed] of particles,
```

а в дополнительных настройках установить тип кривой (Mode) — Bar. Кроме того, в поле Plot setup commands надо указать две команды. Первая команда

```
set-plot-x-range 0 0.5
```

задает диапазон изменения величины, распределение которой мы строим (в нашем случае от 0 до 0.5). Вторая команда

```
set-histogram-num-bars 40
```

определяет число столбиков гистограммы.

7 Разработайте и реализуйте на базе рассмотренной модели эксперимент по проверке формулы Эйнштейна (14.1) и определению коэффициента диффузии D. Например, на рис. 14.16 показано распределение позиций одной броуновской частицы в пять последовательных моментов времени (яркость точки обратно пропорциональна времени), а на рис. 14.17 для той же серии расчетов построен график зависимости усредненного квадрата горизонтального смещения частицы от времени, находящийся в согласии с формулой Эйнштейна.

8 Исследуйте экспериментально, как влияют разные параметры модели (ее размеры, количество частиц того или иного типа, начальная скорость частиц, их размер и т.д.) на коэффициент диффузии.

9 Реализуйте на основе рассмотренной модели броуновского движения простую модель диффузии. В этой модели все частицы имеют одинаковые массу и размер, а отличаются только цветом. Расположите оранжевые частицы в центре модели в виде регулярной прямоугольной решетки и задайте им начальную нулевую скорость. Голубые частицы расположите вокруг оранжевых в случайном порядке, задайте им начальную ненулевую скорость. Запустите модель и посмотрите, как оранжевый кластер в результате диффузии постепенно теряет свою первоначальную квадратную форму и приобретает форму размытого круга (рис. 14.18).

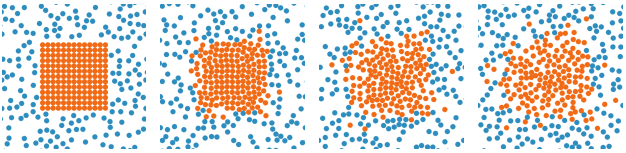

РИС. 14.18 Модель диффузии

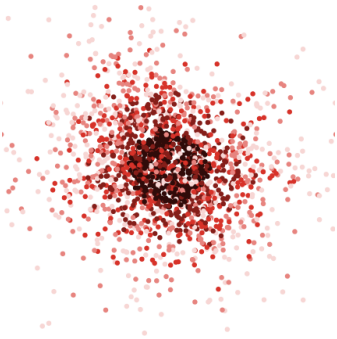

РИС. 14.16 Распределение положения броуновской частицы в пять последовательных моментов времени

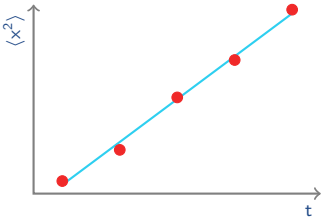

РИС. 14.17 Зависимость квадрата горизонтального смещения броуновской частицы от времени

10 На основе модели из предыдущего упражнения и с учетом столкновения частиц со стенками (упражнение 4) постройте модель маятника Ньютона (или колыбели Ньютона)[19]. Все частицы в этой модели должны располагаться в ряд, как это показано на рис. 14.19. В начальный момент оранжевые частицы должны быть неподвижны, а скорость единственной голубой частицы должна быть направлена строго горизонтально. Исследуйте поведение модели.

[19] F. Herrmann, P. Schmalzle, *Simple explanation of a well-known collision experiment*, American Journal of Physics, 1981, 49:8, p. 761–764.

РИС. 14.19 Схема расположения частиц в модели маятника Ньютона

11 На основе рассмотренных выше моделей разработайте и реализуйте модель известного мысленного эксперимента с демоном Максвелла[20]. Пространство такой модели должно быть разделено на две части *полунепроницаемой* стенкой, которая и будет служить демоном Максвелла (рис. 14.20).

[20] Р. Фейнман, *Характер физических законов*, Библиотечка «Квант», Выпуск 62, М.: Наука, 1987, лекция 5.

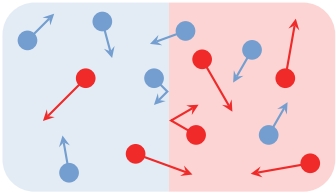

РИС. 14.20 Модель демона Максвелла

Эта стенка пропускает слева направо только те частицы, которые имеют скорость выше средней скорости среди всех частиц модели. Для частиц со скоростями ниже средней, движущихся слева направо, стенка является непроницаемой. При движении справа налево все происходит наоборот — медленные частицы проходят стенку, быстрые частицы от нее отражаются. Для визуализации процесса изменяйте цвет частицы в зависимости от ее скорости, например быстрые (со скоростью большей средней) раскрашивайте в красный цвет, медленные — в синий. Добавьте к интерфейсу модели два монитора (или один график), которые должны показывать среднюю кинетическую энергию частиц (т.е. фактически температуру) в левой и правой частях модели.