ПРОГРАММА, МОДЕЛИРУЮЩАЯ МЕХАНИЗМ И РИСУЮЩАЯ МУЛЬТФИЛЬМ О НЕМ

Н. Н. КОНСТАНТИНОВ, В. В. МИНАХИН, В. Ю. ПОНОМАРЕНКО
(МОСКВА)

Введение

При моделировании механизма полезно иметь возможность получать изображения различных состояний механизма. При этом удобно использовать такой способ задания формы механизма, при котором информация о его положении в данном состоянии и информация о его устройстве были бы разделены. Это требование вполне естественно. В человеческом языке такое разделение имеет место. Скажем, слова «моя кошка» обозначают определенный предмет. При этом ничего не сказано о позе кошки. Можно сказать, что мы стремились к такому способу задания информации о форме предмета, чтобы все, что относится к понятию «кошка вообще», было отделено от информации о положении и позе кошки в данный момент.

Практическим признаком того, что информация обладает указанным свойством, является то, что становится просто и удобно задавать машине законы движения механизма. В случае кошки информация о ее походке и траектории, т. е. сценарий ее движения, задается совершенно независимо от информации о форме. Нужно только, чтобы в описании формы присутствовали переменные, о которых говорится в описании движения. Требования, которые мы предъявляем к информационной системе, можно сформулировать так: это должна быть такая система задания информации о предмете, пользуясь которой легко писать программы, которые выдают мультфильм о предмете по самым разнообразным сценариям.

Наша работа задумана как пробный шаг в направлении создания программ, моделирующих механизмы, и не преследовала никаких целей, кроме опробования некоторой системы задания формы предмета с точки зрения вышеуказанных требований. Мы задаем программе строение тела кошки и законы ее движения и получаем мультфильм (некоторые кадры приводятся на рис. 1), на котором кошка делает несколько шагов, постепенно замедляясь, поворачивает голову и останавливается. Подбирая уравнения, определяющие походку, мы заботились только о внешнем благополучии, а не о том, чтобы описывать истинные физиологические механизмы управления. Но наша программа может быть полезна и для физиологов. Если мы имеем гипотезу о механизме походки, которую мы можем записать в виде дифференциальных уравнений (как правило, второго порядка) относительно переменных, участвующих в описании позы кошки, то мы можем с помощью нашей программы посмотреть, как эта гипотеза работает (о гипотезах см. [5, 6, 7]). Принятая система задания информации удобна для механизмов, являющихся шарнирными системами, состоящими из твердых частей. С известным приближением такой системой является молекула. Программу, подобную нашей, можно применять для визуализации гипотез о строении и работе молекул при химических реакциях (см. [2, 3]).

Интересно провести опыт использования программ, рисующих мультфильм, в качестве вспомогательной техники при создании художественных мультфильмов. Работу над одним монтажным куском длительностью от 10 до 30 *сек* можно представить себе так. Художник-математик должен записать действующих лиц этого куска в виде нашей или подобной информационной системы, а их движения в пределах этого куска — в виде дифференциальных уравнений. Затем машина печатает бумажную ленту — «папирфильм». После этого художники рисуют по папирфильму мультфильм. Таким образом, по-прежнему зритель увидит руку художника. Смысл же всего этого в том, что моделирование движения сделано машиной — это как раз та часть работы, с которой человек справляется плохо (сошлемся на одного из мультипликаторов — [1]; для того чтобы нарисовать танец лягушки, он заснял танец настоящей балерины и, используя этот фильм как модель движения, рисовал по нему фильм о лягушке).

В качестве рисующего устройства использовалась широкая печать — АЦПУ-128. Градация яркости не применялась, и качество изображения сравнительно невысокое. Но большее и не требуется при нашей системе задания формы предметов — рисунки производят неприятное впечатление, если качество передачи формы на бумаге выше, чем качество информации о форме. Когда передача на бумаге приблизительная, воображение зрителя восполняет недостающие детали именно так, как нужно, и впечатление от рисунков получается приличное. Еще лучше выглядит фильм в движении, так как моделирование движения имеет более высокое качество, чем моделирование формы. Зритель получает много «хорошей» информации, и восполнение недостающего облегчается. Интересно, что многие зрители после просмотра фильма не помнили, что изображение было просто теневой проекцией. Изображения, приведенные на рис. 1, прежде чем попасть на страницы этой книги, прошли следующие стадии:

1) на папирфильме, выходящем из АЦПУ-128, проекция кошки остается белой, а весь остальной фон выбит некоторой буквой (бралась буква Ш как самая черная);

2) кадр фотографируется; на негативе — черная кошка и серый фон;

3) серый фон убирается вручную с целью придания этим рисункам более высокого качества как полиграфической продукции.

Работа выполнена на кафедре общих проблем управления механико-математического факультета Московского университета. Отладка программы и ее эксплуатация проводились в Вычислительном центре Московского государственного педагогического института им. В. И. Ленина.

1. Что такое брусок

Кошка будет сделана из «брусков». Форма бруска определяется пятью параметрами: длиной, высотой в начальной части, высотой в конечной части, шириной в начальной части и шириной в конечной части. Определение бруска проведем сначала в специальной системе координат.

Пусть x, y, z — декартова система координат. Бруском называется шестигранник:

$$\left.\begin{aligned} &0 \leqslant x \leqslant 1, \\ &|z| \leqslant k_1 x + \frac{h_н}{2}, \\ &|y| \leqslant k_2 x + \frac{b_н}{2}, \end{aligned}\right\}$$

где $h_н \geqslant 0$, $b_н \geqslant 0$, $\frac{h_к}{2} = k_1 + \frac{h_н}{2} \geqslant 0$, $\frac{b_к}{2} = k_2 + \frac{b_н}{2} \geqslant 0$ (рис. 2).

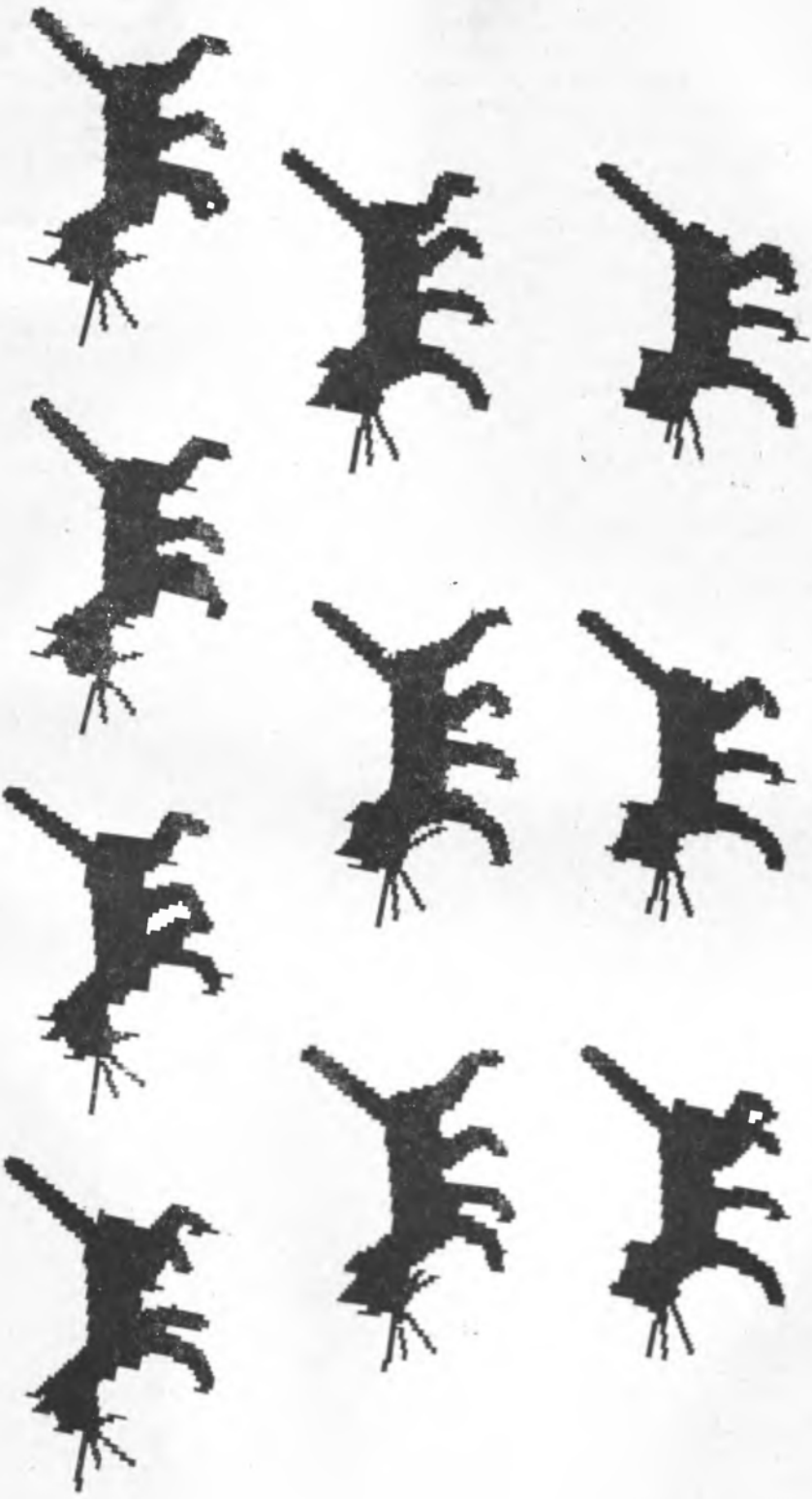


Рис. 1.

Если некоторые из чисел, характеризующих эту область, равны нулю, то число граней может быть меньше шести, в частности, может получиться четырехугольная пирамида или тетраэдр.

Бруском называется также образ вышеопределенного многогранника при подобном преобразовании пространства, сохраняющем ориентацию. Репером бруска называется образ исходной системы координат при том же преобразовании.

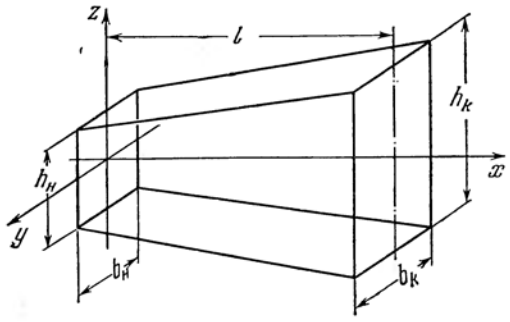


Рис 2. Брусок.

Легко видеть, что для полного определения бруска нужно знать 11 чисел. Мы выберем эти числа так, что 5 из них будут определять размеры и форму бруска (это $h_н$, $h_к$, $b_н$, $b_к$ и коэффициент подобия преобразования l, определяющий длину бруска), а 6 чисел — его положение (три сдвига начала координат и три угла поворота системы координат). Таким образом, каждому бруску ставится в соответствие список из 11 чисел и каждому списку из 11 чисел, при условии, что эти числа удовлетворяют естественным неравенствам, — брусок. Положение репера бруска определяется следующими характеристиками.

Пусть $R = \{x, y, z\}$ — абсолютная система координат, а $R_1 = \{x_1, y_1, z_1\}$ — репер бруска (рис. 3). Будем задавать положение R_1 в R

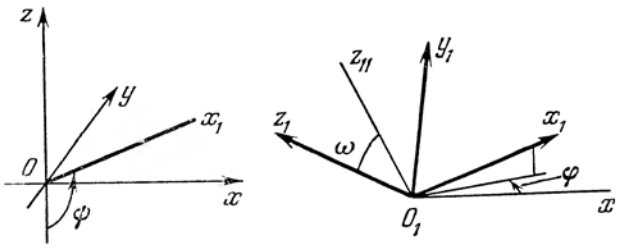


Рис. 3. Углы ψ, φ и ω в хорошем случае.

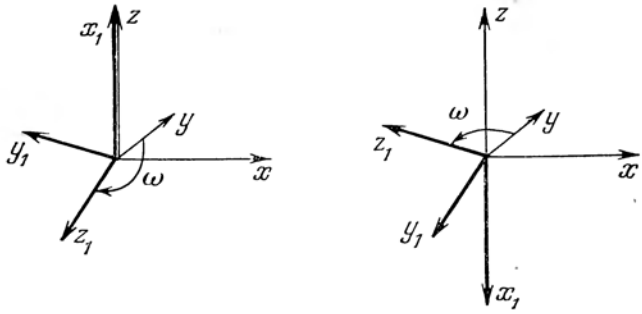


Рис. 4. Угол ω в плохом случае.

следующими числами: координатами точки O_1 в репере R x_{01}, y_{01}, z_{01} и тремя углами ψ, φ и ω, где

ψ — угол между отрицательным направлением оси z и положительным направлением оси x_1,

φ — угол между положительным направлением оси x и положительным направлением проекции x_1 на плоскость xOy. Если проекция x_1 вырождается в точку ($\psi = 0$ или π), то значение φ безразлично.

ω — угол, определяемый по-разному в зависимости от того, выполнено условие $0 < \psi < \pi$ или нет.

Если $0 < \psi < \pi$ (будем называть этот случай хорошим), то ω — это угол между z_{11} и z_1, где z_{11} — луч, перпендикулярный к x_1, такой, что его проекция на xOy лежит на той же прямой, что и проекция x_1, и он направлен вверх от xOy (в сторону z), а угол отсчитывается от z_{11} к z_1 в сторону, положительную с точки зрения x_1 (т. е. если смотреть с конца x_1, то против часовой стрелки; см. рис. 3).

Если хороший случай не имеет места, то ω — это угол от положительного направления оси y к положительному направлению оси z_1 (она лежит в плоскости xOy).

2. Кошка как дерево или формула

Вся информация о форме предмета будет представлена в форме иерархии частей этого предмета. Сам предмет является частью нулевого порядка; он состоит из частей первого порядка; они в свою очередь — из частей второго порядка и т. д., но каждая часть может быть тупиком в схеме, т. е. не состоять из частей. Каждая часть, на каком бы уровне она ни находилась, задается как пара списков: первый список — это список 11 чисел, задающих брусок, а второй список — это список названий подчиненных частей. Тем самым формально схема описана полностью, не сказано только, каков ее содержательный смысл.

Пусть предмет не состоит из частей. Тогда он представлен только первым списком, а второй список пустой. Первый список соответствует некоторому бруску. Если теперь попросить программу нарисовать предмет, она должна нарисовать теневую проекцию этого бруска на плоскость yOz, выделив ту часть рисунка, которая не выходит за рамки кадра. Это и есть содержательный смысл схемы в случае, если она состоит из одного элемента.

Пусть теперь в схеме больше одного элемента и тем самым больше одного этажа. Рассмотрим сначала случай, когда этажей два. Пусть A_0 — весь предмет, A_1 — подчиненная ему часть, которая в свою очередь не состоит из частей. Если теперь попросить программу нарисовать этот предмет, она должна взять брусок, соответствующий первому списку части A_1, но считать, что он задан в системе координат, связанной с бруском, записанным на нулевом этаже. Эта система координат есть не что иное, как репер бруска нулевого этажа. Длина бруска нулевого этажа определяет масштаб изображения бруска первого этажа, а ориентация в пространстве бруска нулевого этажа определяет, как нужно повернуть брусок, заданный списком A_1. После того, как произведен этот поворот, брусок проектируется на плоскость yOz. Если есть еще части, подчиненные A_0, с ними делается то же самое. На рисунке получится объединение проекций всех частей, подчиненных A_0. Сам брусок части A_0 не изображается, он служит только системой отсчета для подчиненных частей. При этом, конечно, некоторые из 11 чисел соответствующего ему списка оказываются несущественными.

Теперь дадим формальное индуктивное определение того, как по схеме собирается предмет. Если схема не состоит из частей, то предмет есть брусок, определяемый единственным списком этой схемы. Если для частей $A_1, A_2, \ldots, A_n$, подчиненных данной части A, определены предметы, им соответствующие, то предмет, соответствующий части A, строится так. Пусть каждой части A_i в абсолютной системе координат соответствует некоторое множество точек M_i. Рассмотрим множество N_i точек, координаты которых в репере бруска A совпадают с координатами точек M_i

в абсолютной системе координат. Поставим в соответствие части A фигуру, полученную объединением всех N_i. Рисунком, соответствующим этой фигуре, является ее проекция на плоскость yOz.

Описанную схему можно изображать на бумаге как дерево или как формулу. Эти записи формально эквивалентны и могут быть удобны для различных случаев.

На сх. 1 приведена схема кошки в виде дерева. Элементарных списков в виде наборов 11 чисел мы не приводим, чтобы не загромождать схему.

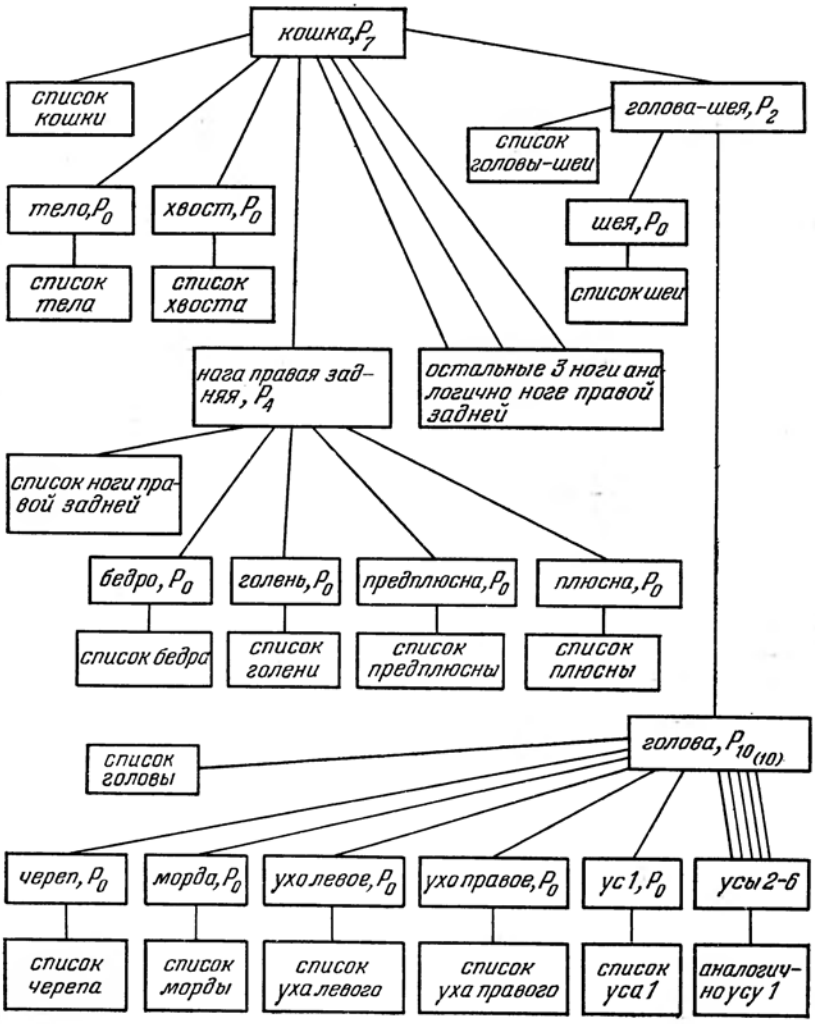


Схема 1.

Они просто заменены названиями соответствующих списков. В каждой вершине этого рисунка стоит название соответствующей части тела кошки. Индекс при букве P показывает, из скольких частей состоит эта часть.

Формула, соответствующая данному дереву, есть «польская» запись этого дерева. Формальное определение следующее. Формулы составляются из знаков двух категорий: исходного алфавита и связок. Элементом алфавита является элементарный список; так называется список из 11 чисел, являющихся описанием бруска. Связкой служит символ P_i, где i — натуральное число или 0. Индуктивное определение:

1) XP_0, где X — элемент алфавита, есть формула;

2) $XP_nA_1A_2 \ldots A_n$, где X — элемент алфавита, а A_1, A_2, $\ldots$, A_n — формулы, есть формула.

Эти определения соответствуют обычному стилю математической логики (см., например, [4], стр. 70).

Чтобы читателю легче было оценить удобство этой системы, покажем на нескольких примерах, как ею пользоваться.

Чего я хочу добиться?	Что для этого надо сделать?
1. Изменить масштаб изображения	Изменить число l в списке кошки
2. Изменить точку зрения	Изменить числа ψ, φ, ω в списке кошки
3. Повернуть голову вместе со всеми ее деталями	Изменить числа ψ, φ, ω в списке головы
4. Подобно увеличить голову по отношению к телу	Изменить число l в списке головы
5. Изменить форму головы	Изменить соответствующие числа в списках черепа и головы
6. Сделать хвост гибким	Заменить формулу хвоста на более сложную
7. Изменить взаимное расположение частей ноги	Изменить соответствующие числа в списках этих частей

Заметим, что все числа списков, присутствующих в формуле кошки, делятся на две группы: числа, определяющие позу кошки в данный момент, и числа, характеризующие ее строение и форму вообще, независимо от выбора момента времени. К первой группе относятся числа, описывающие поворот головы, положение частей ног, положение всей кошки как целого. Во вторую группу входят числа, определяющие размеры головы, размеры туловища, ног и вообще все те размеры и углы, изменение которых при движении не предусмотрено.

В соответствии с этим работающая часть программы естественным образом распадается на две, по сути дела, независимые части: рисующая часть читает формулу и печатает ее изображение, а вторая — двигающая часть — решает дифференциальные уравнения, задающие походку, и записывает в формулу кошки числа первой группы. Описание двигающей программы читатель найдет в п. 5, а о чтении дерева см. в п. 4.

3. Немного о некоторых технических деталях

В предыдущих пунктах мы рассказали о системе, которая позволяет представлять относительно широкий класс объектов в виде дерева простых элементов — системе довольно гибкой и удобной. Теперь нам хотелось бы остановиться немного на том, как эта система в применении к нашей «Кошечке» была вписана в память машины «БЭСМ-4». Мы расскажем о том, как записывали информацию об одном бруске, и о том, как записывалась в памяти информация об устройстве всего дерева. Напомним, что память машины, которой мы пользовались, состоит из трехадресных ячеек длиной 45 разрядов (в адресе 12 разрядов).

Прежде всего об одном бруске. Все характеризующие его числа записывались в с ж а т о м в и д е. Сжатое число — это 15-разрядное двоичное число a, $bcd0$, где a, b, c, d — восьмеричные цифры. Сжатый вид числа x — это такое сжатое число y, что $x + 4 = y$ с точностью до 8^{-3}. В таком виде записывались ω, φ, ψ, l, $h_н$ и т. д. (причем углы измерялись в количестве единиц π) в массиве из четырех ячеек, называвшемся «координаты бруска». Как он устроен, см. рис. 5.

Здесь l — длина бруска, а $h_н$, $b_н$, $h_к$, $b_к$ выражены в единицах l (в долях l); «1» в 4-м разряде 1-го гнезда 1-й ячейки означает, что ω измеряется в соответствии с плохим определением.

Но в соответствии с нашей системой кошка записывалась не в виде бруска, а в виде формулы. Формула же определяется своим бруском,

17_8 разрядов

00010 или 00000	ω	x_{01}
y_{01}	z_{01}	φ
ψ	l	$h_н$
$b_н$	$h_к$	$b_к$

Рис. 5. Массив «координаты бруска».

012	0	Адрес 1	Адрес 2
	Адрес 3	Адрес 4	Адрес 5
	Адрес 6	Адрес 7	Адрес 10
	Адрес 11	Адрес 12	

Рис. 6. «Массив «расшифровка бруска».

во-первых, и подчиненными формулами,— во-вторых. Поэтому наряду с массивом «координаты бруска» каждой формуле соответствует еще и массив «расшифровка формулы». Он устроен согласно рис. 6.

В коде первой ячейки указано, сколько формул подчинено данной формуле, а дальше перечислены адреса «заголовков» подчиненных формул.

		AK	AP

Рис. 7. «Заголовок».

В «заголовке» указан адрес начала массива «координаты бруска» и адрес начала «расшифровки формулы» (см. рис. 7).

Это уже — информация о подчиненных формулах. У них тоже есть подчиненные и т. д. AP есть адрес расшифровки или нуль. Рисующая программа ползает по всему дереву, и если $AP = 0$, то брусок, соответствующий данной формуле, рисуется.

Таким образом, каждой формуле в памяти машины соответствуют «заголовок» (одна ячейка), массив «координаты бруска» и массив «расшифровка формулы», причем последний в том случае, когда AP в «заголовке» не равен нулю. В таком случае координаты бруска самостоятельной ценности не имеют и используются только как система отсчета для нижеследующих этажей.

Итак, вся информация представляет собой в памяти дерево вида, изображенного на схеме 2. Стоит сравнить это со схемой 1 для того, чтобы понять, что разница между ними не больше, чем между теорией и практикой.

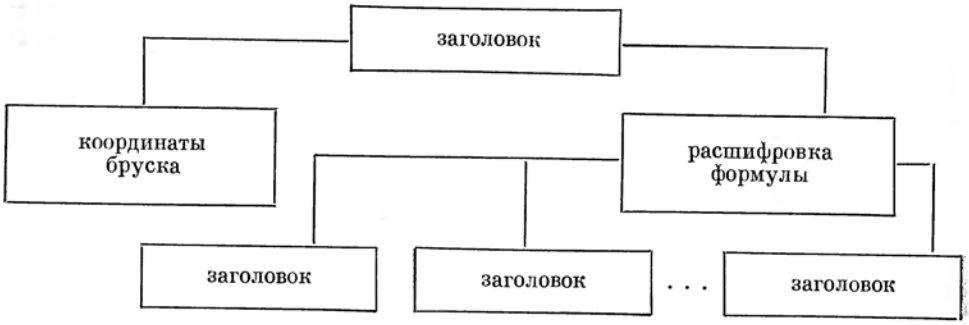


Схема 2.

4. Чтение дерева

Чтением дерева занимается программа «анализ». Она должна, начиная с вершины, каким-то упорядоченным образом проползти по нему, выявить все концевые ветви, где $AP = 0$, и нарисовать соответствующие им бруски. Это будут (см. схему 1) тело, хвост, череп, морда, ухо левое, ухо правое, ус 1-й, усы 2-й, 3-й, . . ., 6-й, шея, четыре бедра, четыре голени и т. д. На картинке они все соберутся в кошку. У программы «анализ» есть ячейка — приемник аргумента. В нее нужно заслать заголовок дерева, которое нужно изучить, после этого отдать управление программе «анализ». Программа «анализ» смотрит, равно ли AP в этом заголовке нулю. Если да, то отдает управление (с возвратом) подпрограмме, которая рисует брусок в памяти машины, заполняя программный буфер печати. Если $AP \neq 0$, то «анализ» перебирает заголовки всех формул, подчиненных исходной, и каждый раз, засылая в аргументную ячейку заголовок очередной формулы, передает управление программе «анализ», т. е. сам на себя. Чтобы такая программа не запуталась с возвратами и не портила рабочих ячеек, она организована как «совершенно стандартная». Если обращение к ней происходит на формуле, которая в дереве лежит на глубине n от вершины (глубина вершины — 0), то регистр адреса при этом обращении равен n. Этим определяется положение рабочих ячеек и анкеты возврата в соответствующих стеках. Таким образом, находясь в какой-то точке дерева, мы помним информацию, относящуюся к пути, который в дереве соединяет эту точку с вершиной, и забываем уже рассмотренные ответвления. Еще мы должны помнить, какую по счету формулу из подчиненных данной мы просматриваем.

5. Моделирование движения кошки

Для того чтобы получить набор картинок-кадров, из которых можно составить мультипликационный фильм, необходимо было задать в памяти машины не только форму и размеры, но и движение объекта.

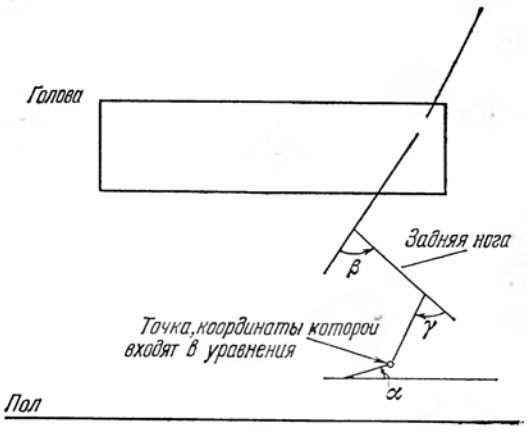


Рис. 8.

Для моделирования движения прежде всего задавался закон (в виде функции $v(t)$), по которому перемещалось туловище кошки, т. е. в некотором смысле вся кошка. Она с некоторой начальной скоростью появлялась из-за кадра и, замедляясь до полной остановки, шла через экран со скоростью $v(t)$. Как только этот основной закон был задан, оставалось только подогнать под него движение различных частей — в первую очередь лап. Этим и занимается блок программы, называющийся «уравнением». Необходимо было понять, по какому закону двигаются лапы у настоящей кошки, и так, чтобы получилось похоже, воспроизвести эти закономерности в виде системы дифференциальных уравнений. Программа решает эти уравнения шагами (методом, сходным с методом Рунге — Кутта, но можно было взять в точности метод Рунге — Кутта) и на каждом шагу выдает ответ в виде картинки. В уравнения входят координаты концов третьих звеньев лап относительно плеч (рис. 8).

Избыточные степени свободы нужно ликвидировать с помощью добавочных условий. Этих условий три:

1) движения ног происходят в вертикальной плоскости, в которой движется кошка;

2) $\alpha = dy$, где y — расстояние от конца третьего звена ноги до пола, а d — постоянный коэффициент;

3) $\beta = c\gamma$, где $c = -1{,}2$ для задней ноги и 0,8 для передней.

Определением формы по двум точкам занимается подпрограмма «нога». Таким образом, оставалось лишь описать движение концов третьих звеньев всех четырех лап.

Про каждую в отдельности было понятно, что в системе координат, связанной с плечом, она должна описывать что-то, похожее на плоский снизу эллипс (рис. 9).

Но был и другой, более содержательный, вопрос: порядок перестановки лап. Вот этот порядок: правая задняя лапа, со сдвигом фаз на четверть периода правая передняя, еще на четверть периода — левая задняя и еще на столько же — левая передняя ([5, 6,7]) (рис. 10).

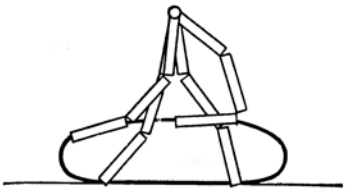


Рис. 9. Траектория конца третьего звена.

Опишем теперь движение конца одной лапы. Так сказать, формализуем кривую рис. 9. Выберем систему координат так: ось x идет вдоль земли в сторону, противоположную движению кошки, ось y проходит через плечо P (рис. 11).

Разумеется, скорость движения точки A по кривой зависит от скорости движения всей кошки $v(t)$ (скорость не постоянна). «Уравнение» занимается вычислением $x''(t)$, $y''(t)$ по заданным t, $x(t)$, $y(t)$, $x'(t)$, $y'(t)$, $v(t)$, $v'(t)$. Программа вычисляет значения правых частей уравнений:

$$\begin{aligned} x''(t) &= \ldots, \\ y''(t) &= \ldots \end{aligned} \qquad \text{(A)}$$

Вычисления ведутся через промежутки времени Δt. Выяснилось, что для работы программы необходимо запоминать значения координат,

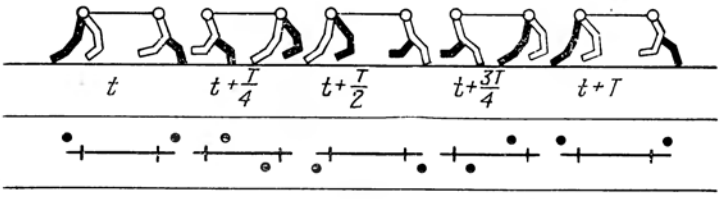


Рис. 10. Выбранный тип походки.

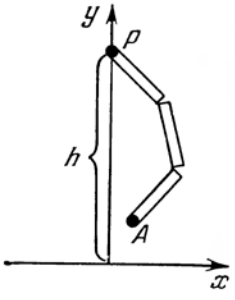


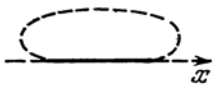
Рис. 11.

их производных и скорости кошки в предыдущий момент $t - \Delta t$. Дальше будет ясно, зачем.

Итак, что же надо писать в правых частях (A)? Если лапа стоит на земле, т. е. если $y(t) = 0$, то, очевидно, имеем рис. 12.

Но не всегда. Если нога слишком уйдет назад (т. е. в положительном направлении оси x), ее пора будет поднимать. Это регулируется кон-

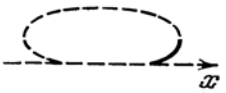
стантой $Ш_{зад}$. Если $|\dot{x}| < Ш_{зад}$, то действительно выражение ⟨1⟩. Если же нет и, кроме того, $x > 0$, то лапу пора поднимать. Но не всегда. Если скорость кошки равна нулю, то лапу стоит оставить на месте (для



$$\begin{aligned} x''(t) &= v'(t), \\ y''(t) &= 0. \end{aligned} \qquad \langle 1 \rangle$$

Рис. 12.

этого тоже достаточно выполнить ⟨1⟩. Мы считаем, что если кошка встала, то $v'(t) = 0$). Если же $v(t) \neq 0$, то решено было (весьма произвольно) положить согласно рис. 13: l здесь — суммарная длина трех верхних суставов лапы, а $\rho(t)$ — расстояние от плеча до конца третьего сустава.

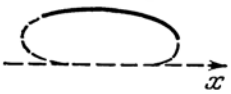


$$\begin{aligned} x''(t) &= M_1 \frac{-x(t)}{|l - \rho(t)|}, \\ y''(t) &= M_2 \left(\frac{h}{8} - y(t) \right) \end{aligned} \qquad \langle 2 \rangle$$

Рис. 13.

Так что чем больше лапа вытянута, тем «бóльшая сила» прикладывается, чтобы ее остановить. А второе уравнение имитирует подъем — тем быстрее, чем ближе к земле лапа находится.

Передвижением по верхней дуге управляют последующие выражения. Они начинают действовать в момент t_0 такой, что $x'(t_0) \leqslant 0$, а $x'(t_0 - \Delta t) > 0$ (вот зачем нужно запоминать характеристики предыдущего момента). Тогда см. рис. 14.

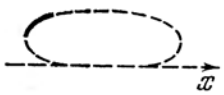


$$\begin{aligned} x''(t) &= -\frac{\pi}{K_2} v(t) \left(1 - \frac{1}{2\pi} \Delta\varphi \right) x(t), \\ y''(t) &= K_1 x'(t). \end{aligned} \qquad \langle 3 \rangle$$

Рис. 14.

Смысл множителя $\left(1 - \frac{1}{2\pi}\Delta\varphi\right)$ будет пояснен ниже, в п. 6.

K_2 полагается равным $|x(t_0)|$ (т. е. программа вычисляет его всякий раз, когда $x'(t)$ меняет знак с плюса на минус), и первое уравнение — просто уравнение синусоидального колебания, параметры которого управляются величиной $|v(t)|$ и величиной начального отклонения $|x(t_0)| = K_2$. Вторая формула обеспечивает подъем лапы при отрицательной горизонтальной скорости и переход к опусканию ее после прохождения «положения равновесия». K_1 вычисляется всякий раз при переходе к выражениям ⟨3⟩ по формуле



$$\begin{aligned} x''(t) &= M_1 \frac{-x(t)}{|l - \rho(t)|}, \\ y''(t) &= K_1 x'(t). \end{aligned} \qquad \langle 5 \rangle$$

Рис. 15.

$$K_1 = \frac{y'(t_0)}{x(t_0)}.$$

Выражения ⟨3⟩ действуют, если $x'(t) \leqslant 0$, $y \neq 0$ и лапа еще не ушла слишком далеко вперед. Выясняется это сравнением $|x(t)|$ с константой $Ш_{пер}$. Если при отрицательном $x'(t)$ $|x|$ достигнет значения, не меньшего $Ш_{пер}$, то действовать начинают выражения ⟨5⟩ (рис. 15).

K_1 здесь остается прежним, так что эти формулы обеспечивают возвращение лапы к оси y и опускание ее вниз. Действуют они до тех пор, пока $x'(t)$ не изменит знак. Пусть в момент t_0 это произошло. Теперь лапу необходимо поставить на землю. Хотелось бы, конечно, чтобы касание произошло, во-первых, с нулевой вертикальной скоростью, а во-вторых, с горизонтальной скоростью, равной $v(t_к)$, где $t_к$ — момент касания ноги земли. Кроме того, следует обеспечить, чтобы лапа коснулась земли достаточно быстро еще при отрицательном x. Последнего можно добиться, если задать заранее момент времени $t_к$. Его решили положить равным

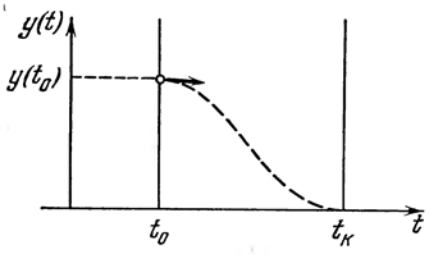


Рис. 16.

$$t_к = \frac{|x(t_0)|}{5M} + t_0,$$

где M — число, заранее ограничивающее $v(t)$. Таким образом, кошка передвинется не более чем на $1/5\,|x(t_0)|$, а лапа уже будет поставлена.

Рассмотрим теперь вопрос о нулевой вертикальной скорости. Задача сводится к тому, чтобы найти функцию $y(t)$, проходящую (рис. 16) через точки $(t_0, y(t_0))$, $(t_к, 0)$ и имеющую в точке $(t_0, y(t_0))$ производную $y'(t_0)$, а в точке $(t_к, 0)$ — нулевую производную. Этого можно добиться, например, положив

$$y(t) = a(t - t_0)^3 + b(t - t_0)^2 + c(t - t_0) + d, \qquad (В)$$

где $d = y(t_0)$, $c = y'(t_0)$, $b = -\frac{2y'(t_0)(t_к - t_0) + 3y(t_0)}{(t_к - t_0)^2}$, $a = \frac{y'(t_0)(t_к - t_0) + 2y(t_0)}{(t_к - t_0)^3}$.

(Значения a, b, c, d находятся из четырех уравнений — два для y и два для y' в точках t_0 и $t_к$.)

Из (В) получаем, обозначая a через K_6, b через K_7 и t_0 через K_5:

$$y''(t) = 6K_6(t - K_5) + 2K_7.$$

Конечно, весьма неестественно, что мы простое вычисление по формуле (В) подменяем решением дифференциального уравнения с начальными условиями $y(t_0)$, $y'(t_0)$ — этим решением будет именно выражение (В).

$$\begin{aligned} x''(t) &= K_4 y'(t) + v'(t), \\ y''(t) &= 6K_6(t - K_5) + 2K_7. \end{aligned} \qquad (6)$$

Рис. 17.

Но дело в том, что «Уравнение» — отдельный блок программы, а внешняя ее часть работает именно с выражением для вторых производных.

Теперь перейдем к горизонтальной координате. Для того чтобы добиться совпадения $x'(t)$ с $v(t)$ в момент касания, можно, например, положить $x'(t) = K_4 y(t) + v(t)$, где $K_4 = -\frac{v(t_0)}{y(t_0)}$. (Напомним, что t_0 — момент перемены знака для $x'(t)$.) Очевидно, что при $y(t) = 0$ получаем, как и хотели, $x'(t) = v(t)$. Вновь превратим формулу в уравнение для определения второй производной и перепишем оба выражения для x и для y (рис. 17).

Итак, лапа поставлена, цикл завершен и можно снова переходить к формулам (1). Однако все не так просто. Нумеруя наши формулы, мы

пропустили цифру ⟨4⟩. И вот почему. Посмотрим, что произойдет, если лапа находится где-то в точке A (рис. 18), и в этот момент скорость кошки стала равной нулю и больше не менялась. Если $x'(t_A) < 0$, то действовали формулы ⟨3⟩. Они по-прежнему будут работать и вычислять

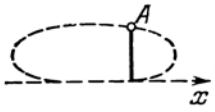
Рис. 18.

$$x''(t) = 0,$$
$$y''(t) = K_1 x'(t_A),$$

т. е. лапа будет продолжать двигаться вперед со скоростью $x'(t_A)$, дойдет до $x = -Ш_{зад}$, затем формулы ⟨5⟩ ее затормозят, и скорость станет равной нулю. Могло быть и с самого начала $x'(t_A) = 0$. Если будут действовать формулы ⟨3⟩, то они выдадут нам

$$x'' = 0,$$
$$y'' = 0,$$

и в зависимости от знака $y'(t_A)$ лапа будет двигаться вверх или вниз. Оба эти варианта нас не устраивают. Особенно первый: кошка стоит, а лапа



$$x''(t) = 0,$$
$$y''(t) = -K_3. \qquad \langle 4 \rangle$$

Рис. 19.

идет вверх, вверх и вверх. В этом случае необходимо перейти к чему-нибудь более разумному. Можно было бы, например, заставить кошку застыть с поднятой лапой, но решено было, что лучше лапу опустить. Это и делают формулы ⟨4⟩. См. рис. 19, где $K_3 = \frac{(y'(t_A))^2}{2y(t_A)}$. Легко видеть, что такое постоянное y'' обеспечивает постановку ноги на землю с нулевой скоростью.

Было решено также, что формулы ⟨4⟩ будут вступать в действие всегда, когда $x'(t) = 0$ и $v(t) = 0$, т. е. не обязательно на участке, обслуживаемом формулами ⟨3⟩ (см. рис. 14).

* * *

Итак, теперь выписаны все формулы. Они сведены в табл. 1, где имеются также выражения для вычисления коэффициентов $K_1, \ldots, K_7$ и указано, когда же эти коэффициенты вычисляются.

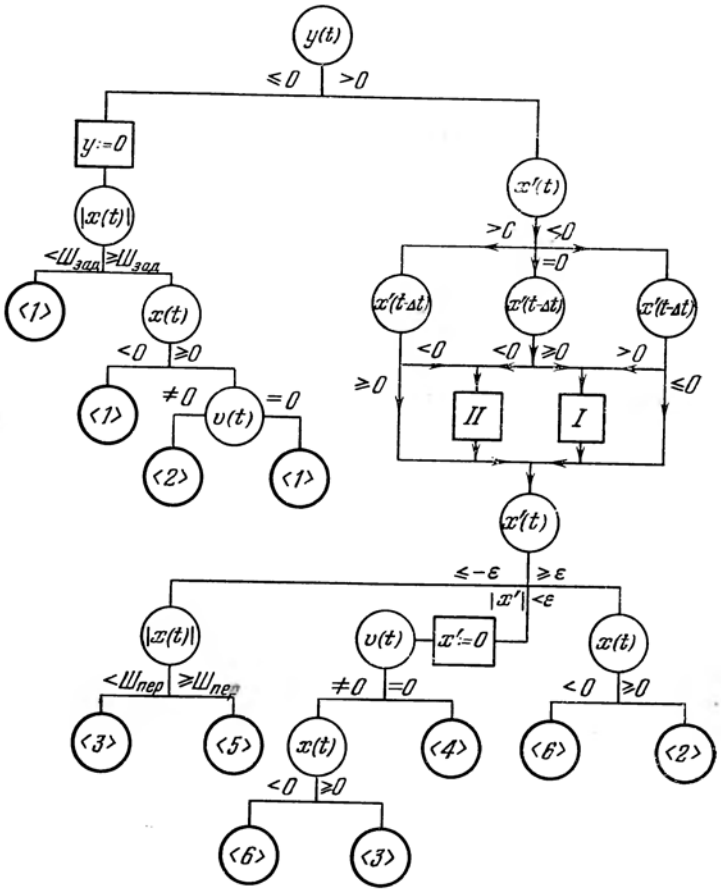
Таблица 1

⟨1⟩	$x''(t) = v'(t)$	$y''(t) = 0$
⟨2⟩	$x''(t) = M_1 \frac{-x(t)}{\lvert l - \rho(t) \rvert}$	$y''(t) = M_2 \left(\frac{h}{8} - y(t) \right)$
I	Если $x'(t) \leqslant 0$ и $x'(t - \Delta t) > 0$, то	$K_1 = \frac{y'(t)}{x(t)}$ $K_2 = \lvert x(t) \rvert$ $K_3 = \frac{(y'(t))^2}{2y(t)}$
⟨3⟩	$x''(t) = -\frac{\pi}{K_2} v(t) \left(1 - \frac{1}{2\pi} \Delta\varphi\right) x(t)$	$y''(t) = K_1 x'(t)$

Продолжение табл. 1

⟨4⟩	$x''(t)=0$	$y''(t)=-K_3$
⟨5⟩	$x''(t)=M_1\dfrac{-x(t)}{\lvert l-\rho(t)\rvert}$	$y''(t)=K_1x'(t)$
II	Если $x'(t)\geqslant 0$ и $x'(t-\Delta t)<0$, то	$K_3=\dfrac{(y'(t))^2}{2y(t)}$ $K_4=-\dfrac{v(t)}{y(t)}$ $K_5=t$ $K_6=\dfrac{y'(t)\left(\dfrac{\lvert x(t)\rvert}{5M}\right)+2y(t)}{\left(\dfrac{\lvert x(t)\rvert}{5M}\right)^3}$ $K_7=-\dfrac{2y'(t)\left(\dfrac{\lvert x(t)\rvert}{5M}\right)+3y(t)}{\left(\dfrac{\lvert x(t)\rvert}{5M}\right)^2}$
⟨6⟩	$x''(t)=K_4y'(t)+v'(t)$	$y''(t)=6K_6(t-K_5)+2K_7$

Таблица 2



В каких случаях какая из формул работает, показано в табл. 2. По сути дела, это блок-схема программы «Уравнение». Кружки обозначают проверку некоего условия на указанные в них переменные, квадратики описывают некоторые действия, в кружках и в угловых скобках стоят номера формул, вступающих в работу, а римские цифры в квадратиках — номера, показывающие, какая из групп коэффициентов табл. 1 здесь вычисляется.

Не следует удивляться тому, что y может быть отрицательным. Это связано с дискретностью вычислений, с тем, что счет ведется с некоторым

Таблица 3

Лапа	пр. з.	пр. п.	лев. з.	лев. п.
$Ш_{пер}$	0,3	0,4	0,3	0,4
$Ш_{зад}$	0,4	0,3	0,4	0,3
$5M$	2,0			
ε	0,001			
KM_1	1,0			
KM_2	15,0			
h	0,75			

шагом Δt. Именно поэтому в схему включено действие присвоения $y := 0$. Так же объясняется и почему $x'(t)$ сравнивается не непосредственно с нулем, а с некоторой константой ε. Кстати, значения используемых в программе констант приведены в табл. 3.

6. Координация движения различных лап

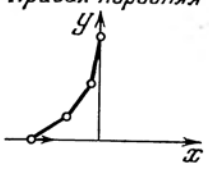
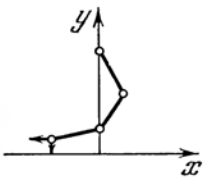
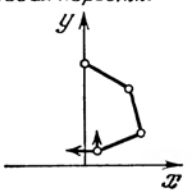
Объясним теперь, зачем в формулу ⟨3⟩ включен множитель $\left(1 - \frac{1}{2\pi}\Delta\varphi\right)$. Дело вот в чем. Движение каждой лапы определяется, во-первых, «Уравнением», а во-вторых, начальными условиями. Эти начальные условия были положены равными значениям, приведенным в табл. 4.

Таблица 4

Начальные условия для ног

Правая задняя 	$y\ (0) = 0$ $x\ (0) = 0{,}3$ $y'\ (0) = 0$ $x'\ (0) = v\ (0) = 0{,}554$	$y\ (-\Delta t) = 0$ $x\ (-\Delta t) = 0{,}289$ $y'\ (-\Delta t) = 0$ $x'\ (-\Delta t) = v\ (0) = 0{,}554$

Продолжение табл. 4

Правая передняя 	$y\ (0)=0$ $x\ (0)=-0{,}2$ $y'\ (0)=0$ $x'\ (0)=v\,(0)=0{,}554$	$y\ (-\Delta t)=0$ $x\ (-\Delta t)=-0{,}211$ $y'\ (-\Delta t)=0$ $x'\ (-\Delta t)=v\,(0)=0{,}554$
Левая задняя 	$y\ (0)=0{,}22$ $x\ (0)=-0{,}2$ $y'\ (0)=-0{,}1$ $x'\ (0)=-0{,}45$	$y\ (-\Delta t)=0{,}222$ $x\ (-\Delta t)=-0{,}199$ $y'\ (-\Delta t)=-0{,}099$ $x'\ (-\Delta t)=-0{,}451$
Левая передняя 	$y\ (0)=0{,}22$ $x\ (0)=0{,}3$ $y'\ (0)=+0{,}1$ $x'\ (0)=-0{,}45$	$y\ (-\Delta t)=0{,}222$ $x\ (-\Delta t)=0{,}301$ $y'\ (-\Delta t)=0{,}101$ $x'\ (-\Delta t)=-0{,}449$

Были заданы и начальные значения коэффициентов $K_1, \ldots, K_7$. Они, естественно, свои для каждой лапы (табл. 5).

Так как $v\,(t)$ вовсе не постоянна, то задание таких начальных условий для четырех лап еще отнюдь не обеспечивает правильного сдвига

Таблица 5

Начальные значения коэффициентов

Правая задняя	$K_1,\ K_2,\ \ldots,\ K_7=0$
Правая передняя	$K_1,\ K_2,\ \ldots,\ K_7=0$
Левая задняя	$K_1=1/3,\ K_2=0{,}6,\ K_3,\ \ldots,\ K_7=0$
Левая передняя	$K_1=1/3,\ K_2=0{,}6,\ K_3,\ \ldots,\ K_7=0$

фаз на четверть периода во все время движения кошки. Для того чтобы подправлять расхождения, и вводится множитель $\left(1-\frac{1}{2\pi}\Delta\varphi\right)$. Делается это так.

Положение каждой лапы проектируется на окружность с помощью так называемого «фазового отображения» (см. рис. 20) по формулам:

$$\varphi=\begin{cases}\dfrac{\pi}{2}+\dfrac{\pi}{2\sqrt{l^2-h^2}}x, & \text{если} \quad x'\geqslant 0,\\[2ex] \dfrac{3\pi}{2}-\dfrac{\pi}{2\sqrt{l^2-h^2}}x, & \text{если} \quad x'<0.\end{cases}$$

Получаются четыре числа φ_1, φ_2, φ_3, φ_4, по одному на каждую лапу. Затем ищется такой крест (два перпендикулярных диаметра), чтобы сумма квадратов расстояний от его концов до точек φ_1, φ_2, φ_3, φ_4 была минимальной (рис. 21). После этого в уравнение ⟨3⟩ для каждой ноги вносится

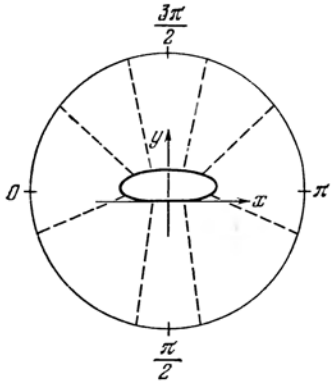


Рис. 20. «Фазовое отображение».

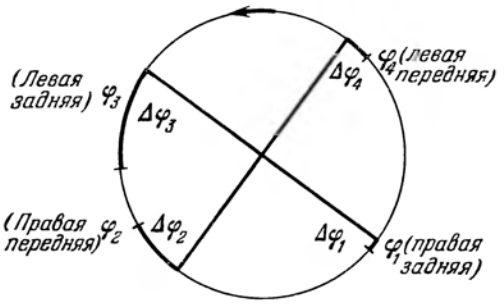


Рис. 21. Фазовый круг. Крест. Поправки.

соответствующий ей множитель $\left(1 - \frac{1}{2\pi}\Delta\varphi\right)$, т. е. если лапа, например, отстает ($\Delta\varphi$ отрицательно), $|x''|$ становится больше и лапа будет догонять свое место. Обратим внимание, что такое управление осуществляется, только когда действуют формулы ⟨3⟩. Когда нога стоит на земле, оно, естественно, бессмысленно. А остальными участками решили пренебречь.

ЛИТЕРАТУРА

1. Арнольди Э. М., Жизнь и сказка Уолта Диснея, Л., «Искусство», 1968.
2. Журкин В. Б., Лысов Ю. П., Опыт моделирования на ЭВМ движений в молекуле ДНК МОИП, Доклады, Общая биология, Изд. МГУ, 1971.
3. Левинталь С., Построение молекулярных моделей с помощью вычислительной машины, Сб. «Молекулы и клетки», вып. 3, М., «Мир», 1968, 29.
4. Новиков П. С., Элементы математической логики, М., Физматгиз, 1959.
5. Орловский Г. Н., Северин Ф. В., Шик М. Л., Влияние скорости и нагрузки на координацию движения при беге собаки, Биофизика **11**, 2, 1966, 364—366.
6. Суханов В. Б., Общая система симметричной локомации наземных позвоночных и особенности передвижения низших тетрапод, Л., «Наука», 1968.
7. Шик М. Л., Орловский Г. Н., Координация конечностей при беге собаки, Биофизика **10**, 6, 1965, 1037—1047.

Поступило в редакцию 25 XII 1972.