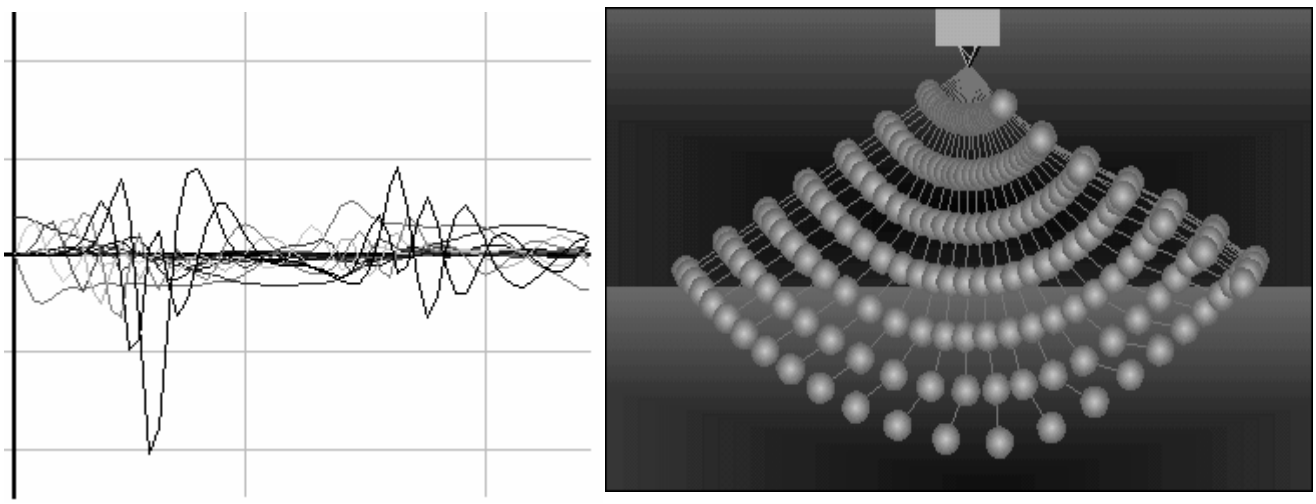
ПРИЛОЖЕНИЕ

П.1. Введение в проблему визуализации научных экспериментов

Для повышения достоверности научных и инженерных результатов важную роль играет визуализация научно-технических расчетов. Пояснить преимущества хорошей системы визуализации помогает простой пример, поясняемый рис П.1. В части рисунка П.1, *а* – изображен результат математических расчетов падения многозвенного манипулятора в виде совокупности графиков, отображающих угловые положения его звеньев. В части рисунка 5.1, *б* – показан тот же расчет, но не в виде графиков, а непосредственно в виде стробоскопированного изображения движения моделируемого манипулятора.



а) *б*)

Рис. П.1. Мониторинг расчета движения многозвенного манипулятора

Оценить адекватность моделирования объекту по левому рисунку весьма проблематично. Эта форма представления информации обладает преимуществом точности в ущерб обозримости получаемых результатов. Если расчет по каким-либо причинам выполнен неверно, то этот факт установить по графикам весьма трудно.

Ошибки, допущенные на начальной стадии проекта, влекут далеко идущие последствия. Правый рисунок представления той же самой информации обладает несомненными преимуществами для оперативной оценки качества моделирования. В этом виде значительно легче отличить правильный результат от результата абсурдного, идущего вразрез со здравым смыслом.

На сегодняшний день на рынке программного обеспечения имеются десятки пакетов, различающихся кругом решаемых задач, представлением входных и выходных данных, используемым математическим аппаратом, требуемыми вычислительными ресурсами и другими характеристиками. К ним, в частности, относятся математические и инженерные пакеты EUREKA, СИАМ, VISSIM, LABVIEW, ENGINE, CIRCUITMAKER, MATHCAD, MATLAB, SCILAB, DERIVE, MAPLE, MATHEMATICA, THEORIST, CLASSIC, MODELVISION и другие, которые могут быть использованы как в учебном процессе, так и в научной работе.

Следует заметить, что каждый из математических и прикладных пакетов – развивающийся, проходящий разные стадии развития. В совокупности они взаимно дополняют друг друга, борются за общую нишу пользователей. В этой среде есть свои лидеры и аутсайдеры, гиганты и карлики, программы с удобным лаконичным интерфейсом, и программы менее быстро осваиваемые, программы наделенные хорошими анимационными способностями и бедные ими.

Нынешний динамично развивающийся рынок научных исследовательских и прикладных программ так или иначе востребует все из них – и в различных сферах появляются свои предпочтения и антипатии, свои пакеты, имеющие те или иные преимущества, и пакеты – просто необходимые для проведения полноценного исследования. Часть из этих пакетов, такие как VISSIM или LABVIEW, специально разработаны для целей моделирования и исследования динамических систем, другие, такие как MATLAB или MAPLE, носят общематематический характер.

П.2. Система визуализации научных экспериментов VISUAL MATLAB

В студии VISUAL MATLAB, разработанной автором диссертации, за основу принята стилизованная версия языка MATLAB, изложенная в фундаментальной работе "Матричные Вычисления" авторов Дж. Голуба и Ч. Ван Лоуна. Ниже на рис. П.2 представлен интерфейс пакета.

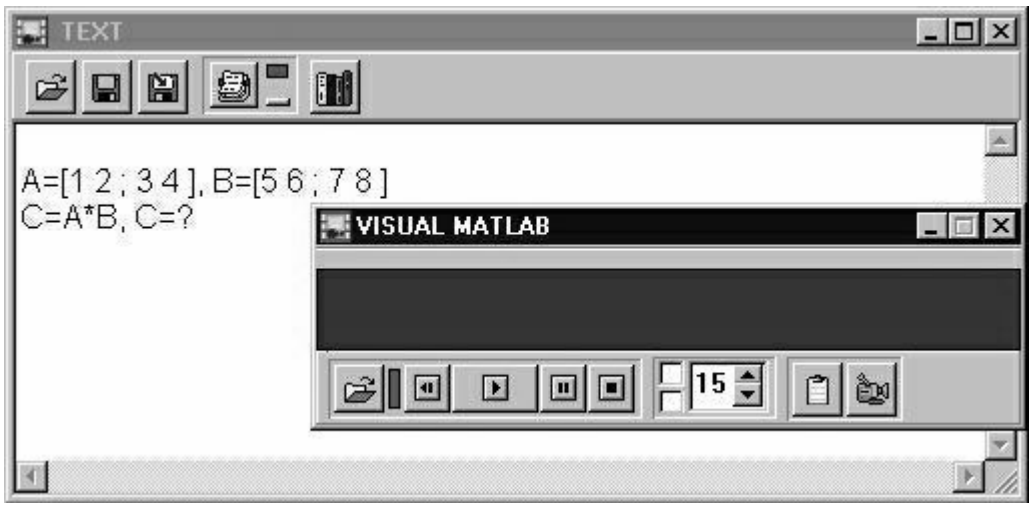


Рис. П.2. Интерфейс пакета VISUAL MATLAB

Перечислим сначала основные отличия от MATLAB. Их три. Во-первых, ось времени. С целью повышения скорости вычислений она генерируется специальной функцией t=time(протяженность), а не t=0:шаг:протяженность. Во-вторых, имена переменных декларируются matrices: имя, имя, имя (имя в одну букву можно не декларировать), также, например, как и в Pascal. В-третьих, точка с запятой используется только после команд вывода матриц и графиков для организации паузы в их рассмотрении. Первое, второе и третье избавляет транслятор от издержек анализа текста, что важно для скорости интерпретации. В остальном синтаксис сохраняется, формулы отделяются друг от друга запятыми, и поскольку MATLAB открытая система, где функции пишутся пользователями, введение новых некоторых функций не противоречит его концепции.

Помимо языка матричных вычислений VISUAL MATLAB поддерживает язык управления картинками, которые могут содержать родительские и дочерние сегменты, подобно тому, как на руке имеются пальцы, а пальцы имеют фаланги. Запятая связывает интерпретируемые предложения, описывающие одновременное движение частей, точка служит цели вывода синтезированного изображения на экран монитора. Синтаксис приближен к синтаксису обычного языка "Simple English", причем есть адаптаторы к прочим европейским языкам, включая русский. Перечень поддерживаемых графических команд содержится в справочном разделе самой программы.

Особое внимание отводилось удобочитаемости интерпретируемого транслятором текста: "кисть манипулятора поворачивается на X градусов влево". Где вычислением переменной X занят, собственно, MATLAB. Сочетание этих двух языков в одном трансляторе (математического и "обычного") создает эффективное средство визуализации математических вычислений VISUAL MATLAB.

Пример. Итак, договоримся считать, что матрицы обозначаются латинскими буквами A, B, ..., x, y, z. В том случае, когда нам нужно привлечь более развитое обозначение, оно обязательно декларируется (уступка скорости трансляции) matrices: Xo, X1, X2, X3, и так далее. Типичные матричные выражения выглядят также, как и в MATLAB, X=[1 1]', Y=X'*X. Штрих обозначает операцию транспонирования.

Вывод информации упрощен y=? или y=?J (в формате JAVASCRIPT), в виде графика y=??, [t y]=?2D (2D график) и т.д. Стиль графика может варьироваться опциями ?- ?~ ?*. Пример вывода графика функции: t=time(100), F=2*t+10*sin(0.5*t), [t F]=?2D_комментарий.

Дополнительной опцией можно указать количество временных отсчетов t=time(T,200), по умолчанию принято генерировать сто точек, начиная с нуля.

Строки матрицы отделяются точкой с запятой A=[1 2 ; 3 4], процедура решения системы линейных уравнений AX=B выглядит как X=A\B. Левая и правая операции деления A\B и A/B отличаются тем, что в первом случае инвертируется матрица A, а во втором инвертируется уже B.

Матрицу можно вводить и формулой, например A=solve([n m],'1/(i+j)'), где n и m - размеры матрицы, в одинарных кавычках размещается функция от индексов элемента, описывающая его численное значение.

Интерпретируемый текст можно брать с любой страницы, в том числе, прямо из интернет, включая картинки, загружаемые в особый буфер, на их основе производится мультипликация. С учетом распространенности интернет-технологий, такое правило добавляет удобств в использовании транслятора. % – это значок комментария.

Плоттер оригинального пакета громоздок, учитывая направленность анимационного пакета, многие его функции излишни. Для контроля вычислений оставлены следующие достаточно широкие возможности. Каждая пара столбцов может рассматриваться как аргумент и функция, в таком случае вывод графика инициируется нотацией A=?@, например [t F]=?@. Опция W используется для вывода черно-белых графиков ?@W, что важно при их печати. Функции нескольких переменных можно выводить как Y=?D, [X Y]=?2D, [X Y Z]=?3D. Реализован вывод матрицы в виде цветных карт A=?CR или поверхности (mesh) A=?MR. Цвет задается опцией из массива {R/G/B/Y/F/T/W/I/3}. Третья координата Z может рассматриваться как интенсивность свечения пикселов [X Y Z]=?3R. Цвет или форма точек задается опцией из массива {R/G/B/+/O}.

Функцию F=F(t) можно рисовать импульсами [t F]=?@-. В ряде случаев на график наносятся маркерные точки [t F X Y]=?@+ или маркеры в виде окружностей [t F X Y]=?@O, их координаты содержатся в векторах X и Y. Ненулевой маркер наносится так [t F x y]=?@: .

Масштабы осей s=[Sx Sy] графика функцию двух переменных можно задавать заранее s=axis(Sx,Sy) или s=axis(Sy).

Та же самая функция возвращает масштаб осей s=axis([t F]), если он неизвестен и вычисляется графикопостроителем автоматически. Опции 1,2, 11, 12, 21, 22, предложенные еще в первой оригинальной версии MATLAB, делят окно вывода на части, их можно указывать так s=axis(Sx,Sy,'12') или непосредственно [X Y]=?@12.

Полюса и нули передаточных функций на комплексной плоскости можно различать между собой формой так: [R I]=?+, [R I]=?O.

Режим удержания графика задается удвоением ?@@, команда close @ заканчивает режим удержания.

FOR-ЦИКЛ имеет вид for i=n:size, ... end (size>=0).

ЛОГИКА (ОПЕРАЦИИ СРАВНЕНИЯ <,>,<>,=,>=,<=,~=,==).

If a>b then ... else ... end .

Функции n=rows(A), m=cols(A) возвращают количество строк n или столбцов m матрицы A соответственно. Функция size(A) возвращает максимальную размерность матрицы. Размерности ее можно вычислять как n=size(A,1), m=size(A,2). A(i,:) это i-тая строчка, A(:,j) это i-тый столбец, A(i:q) это строчки от i до j, A(i:q,j:p) возвращает субблок. Пример: C=zeros(n,m), For i=1:n, For j=1:m, C(i,j)=A(i,:)*B(:,j), End End .

В функциях используются только глобальные переменные. Можно добавлять вызовы из DLL. Функции могут вызываться по формуле. Пример вызова функции по формуле: dX/dt=AX+BU .

function: dX/dt=AX+BU

X=statestep(A*dt,B*U*dt) % тело функции .

Ось времени и целые числа генерируются функциями time(T), time(T,n), line(n), line(a,b). Последовательности из нулей, единиц или нормальный шум задаются zero(n), one(n), noise(n). В случае аргумента в виде матрицы или вектора вместо n используется размер, например, t=time(T), x=noise(t). Второй аргумет функции one(n,m) используется для генерации единичного орта с единственной отличной от нуля координатой m. Инверсную к ней функцию вычисляет zero(n,m).

Пример: l=line(3) – это [1 2 3]', l=line(-3) – это [-1 -2 -3]', l=line(-5,5) – это [-5 -4 -3 -2 -1 0 1 2 3 4 5]', z=zero(2) – это [0 0]', e=one(2) – это [1 1]', e=one(3,2) – это орт [0 1 0]', z=zero(3,2) – это [1 0 1]'.

Нулевая матрица создается функциями zeros(n), zeros(n,m), zeros(A). Матрица из единиц генерируется сходно: ones(n), ones(n,m), ones(A). В последнем случае от A берутся ее размеры. Единичную матрицу возвращает функция eye(n). Кроме того, есть функции генерации "шахматной доски" chess(n) и случайного шума rand(n). Функция rand(A) возвращает нормальный шум с амплитудами из A.

Пример: A=zeros(2) – это [0 0 ; 0 0], B=ones(2) – это [1 1 ; 1 1], C=eye(2) – это [1 0 ; 0 1], chess(8) – "шахматная доска" – [0 1 0 1.. ; 1 0 1 0..], вывод A=?CR.

Поточечные сложение, вычитание, умножение, деление и инверсия элементов матриц задаются функциями R=A±B, R=mulp(A,B), R=divp(A,B), R=invp(A). В состав элементарных функции включены функции вычисления абсолютных значений элементов A=abs(B), возведения в степень A=B^n, факториал A=fact(n), корень квадратный A=sqrt(B), максимальный элемент матрицы M=max(A), минимальный элемент матрицы m=min(A), две функции округления A=round(B) и A=floor(B), наращивания и вычитания единицы A=inc(B) и A=dec(B), суммирования элементов столбцов X=sum(A), булева функция отрицания A=not(B). Есть знаковая функции A=sign(B) и ее положительная A=signp(B) и отрицательная A=signn(B) разновидности, а также функция "выпрямления" A=diode(B). Тригонометрические функции: экспонента A=exp(B), функции логарифмов A=ln(B) и A=lg(B), и прочие функции A=sin(B), A=arcsin(B), A=cos(B), A=arccos(B), A=tg(B), A=arctg(Y[,X]), A=ctg(B), A=th(B), A=arth(B), A=sh(B), A=arsh(B), A=ch(B), A=arch(B). Радианы в градусы и градусы в радианы переводятся D=deg(R), R=rad(D), Pi(n) это вектор из чисел Пи, одиночное число Пи возвращается и при нуле P=Pi(0), функции pi2(x) и pi2pi(x) нормализуют значение аргумента.

Матричные функции. Корень квадратный A=sqrtm(B), матричная экспонента A=expm(B) и матричный логарифм B=lnm(A).

D=eig(A) возвращает вещественнозначную Жорданову матрицу (каждое комплексное число a+jb представлено блоком [a b; –b a], матрица собственных векторов содержит в этом случае в соседствующих столбцах вещественную и мнимую компоненты комплексного собственного вектора), [V,D]=eig(A) или [V,_]=eig(A) возвращает компоненты разложения A=VD/V .

S=svd(A) возвращает сингулярные значения A, [U,S,V]=svd(A) возращает компоненты сингулярного разложения A=USV' .

A=inv(B) это псевдоинверся (совпадающая с обратной матрицей в невырожденном случае), кроме того вычисляются детерминант d=det(A), число обусловленности c=cond(A), ранк r=rank(A) и фробениусова норма n=norm(A).

Пусть N=[c1 c2 ...]' вектор коэффициентов полинома.

Фробениусова матрица рассчитывается A=compan(N), корни полинома как R=roots(N) или R=roots(A). Коэффициенты можно рассчитать по корням N=polyc(R), операция свертки N=conv(N1,N2) возвращает коэффициенты произведения двух полиномов. Значение полинома в точке v=polyv(N,x), x=[a b] это комплексный аргумент a+jb. Нормальный шум генерирует X=noise(t). Флип инверсию, а также одномерное и двумерное сглаживания возвращают функции Y=flip(X), Y=smooth(X), B=smooth2D(A). Функция может быть прорежена Y=sample(X,n) с шагом n.

Скалярное произведение двух векторов Y'*X возвращает S=muls(Y,X). Квадратичная форма x'Ax вычисляется для каждого значения аргумента x' из колонки аргументов X как F=qform(A,X). Корелляционную и автокорреляционную функции вычисляют K=xcor(Y,X), R=xcor(X). Дискретное Фурье преобразование F=fft(Y). Есть функция насыщения Y=sat(X,m) с уровнем m. Дистанция между трэком на плоскости и точкой вычисляется функцией D=dist([Y X],[y x]). Протяженность процесса l=length(X).

Экстремальные задачи. Пусть заданы интервал от a до b, а также в случае необходимости дополнительно количество точек n и точность решения tolerance.

Тогда F=solve([a b],'f(x)=x*x+x-1') калькулирует значения аргумента и функции F=[x f]. С целью ускорения вычислений запрещено использовать вложенные скобки f(q(x)), вместо этого записывается цепочечная система уравнений 'y=q(x)_z=p(x,y)_..f(x)=f(x,y,z,...)'.

Аргумент точки экстремума X=solve([a b],'x*x+x-1->extr,x'), график [F X 0]=?@:_Extremum .

Интервал P=[a b], на котором функция F=[x f] меняет знаки, уточняет P=fzero(F), это позволяет искать ее корень методом половинного деления R=solve(P,'x*x+x-1=0,x'), [F R 0]=?@:_Root.

Пусть T - шаг по времени, X=X(0) это некоторый вектор начального состояния. Тогда X=solve(T,'dX/dt=A*X') возвращает следующую точку решения линейного или нелинейного дифференциального уравнения. Процедуру можно итерационно повторять. Последняя функция раскрывает широкие возможности для моделирования динамических систем.

Функции для организации численных экспериментов. Флип вертикальный A=flipud(B), горизонтальный A=fliplr(B), сдвиг по вертикали A=pushud(B[,n]), по горизонтали A=pushlr(B[,n]).

Вектор диагонали D=diag(A), обратно A=diag(D), нижний треугольник L=tril(A), верхний треугольник R=triu(A), вектор из элементов нижнего треугольника X=trilx(L) и обратно L=trilr(X), построение симметричной матрицы A=trils(L), теплицева матрица A=toeplitz(X), вектор столбцов X=colsx(A), вектор строк X=rowsx(A).

Удаление каймы справа и внизу A=cut(B), n-каймы A=cut(B,n), срез сверху A=cutu(B[,n]), снизу A=cutd(B[,n]), слева A=cutl(B[,n]) и справа A=cutr(B[,n]). Сохранение только крайних элементов B=border(A), замещение центральных B=border(A,C).

Расширенные функции поиска максимума и минимума с указанием номера элемента [M,i]=max(X), [m,i]=min(Y). Отсортировать каждую колонку или строку A=sortud(B), A=sortlr(B). P=permud(B), P=permlr(B) это векторы перестановок, иначе [A,P]=sortud(B), [A,P]=sortlr(B).

Сортировка с помощью вектора (или строки) перестановок p=[3 2 1...]' всех строк A=sortud(B,p), всех столбцов A=sortlr(B,p), восстановить B=restud(A,p), B=restlr(A,p). Рассчитать вектор перестановок по наклонам сигналов (резкости возрастания) p=sharp(F) или частоте осцилляций p=sharpos(F), последующая сортировка столбцов A=sortlr(F,p). A=multiply(B,n) это n-раз размножение [B; B; ...] .

Композиция A=[B C D .. ; E F G ..], декомпозиция [B C D ..]=A или [B; C; D; ..]=A, A(i,:) это i-я строка, A(:,j) это j-й столбец, A(q:i) это строки от q до i, A(q:i,p:j) это субматрица. Пусть J=[k j] тогда A(J) возвращает строки A([k j]) от k до j. Пусть I=[m i] тогда A(J,I) возвращает блок B=A([k j],[m i]). K=[J; I], B=block(K,A). В композициях ограничен анализ вложенных функций f(g(..)) глубиной не более четырех.

Комплексная арифметика тоже представлена.

Пусть A=[R I] это комплексная матрица R+jI. Комплексные функции возвращают вещественный и мнимый ее аргументы re(A), im(A), модули mag(A), фазы angle(A), комплексно сопряженную матрицу conj(A), транспонированную матрицу tran(A). Vi=reim(A,i) возвращает i-й комплексный столбец, обратный процесс A=reim([V1 V2 ...]), комплексная диагональ D=diagc(A), диагональная матрица A=diagc(D).

Сложить A+B, вычесть A-B, умножить mulc(A,B), разделить divc(A,B), псевдообратить C=invc(A), поточечные умножения и деления: mulpc(A,B), divpc(A,B), умножение на вещественную матрицу mulc(real matrix,A), но mulc(A,real vector) вычисляет только умножение на вещественный вектор.

J=eigc(R) это комплексная Жорданова матрица. Компоненты [V,J]=eigc(R) или [V,J,S]=eigc(R), J=reim(roots(R)), J=?+ .

Разложения матриц играют важную роль в компьютерном анализе.

Пусть S будет симметричная положительно определенная матрица S>0. Тогда матричное разложение Холецкого S=L*L' это L=ll(S). Пусть Q будет ортогональной матрицей, L и R это левая и правая треугольная матрицы. Разложение A=L*R возвращает функция L=lr(A), причем [L,R]=lr(A). Разложение A=L*Q возвращает функция Q=lq(A), причем [L,Q]=lq(A). Разложение A=Q*R возвращает функция Q=qr(A), причем [Q,R]=qr(A).

Пусть Ax=B, A неотрицательно определенная симметричная матрица A=A'. B-зависимая триангуляция A=L*T это [L,T,p,b]=lt(A,B) T это правая трапецевидная матрица, p это вектор перестановки, преобразованная правая часть b состоит из L\sortud(B,p).

Псевдоинверсия матрицы A: P=inv(A) колоночным методом Гревилля [P,s]=pinv(A) возвращает ранг и синусы углов базиса векторов столбцов s=[rank(A), sin(углы колонок)]'. Пусть нормальная система уравнения имеет вид AX=B, A=A', A>0. Псевдоинверсия X=Xo+W(AW)°(B-AXo), ||AX-B||->min, ||inv(W)(X-Xo)||->min, b=B or b=[B Xo diag(W)] и X=lsm(A,b), [X,r]=lsm(A,b), r это rank(A).

Функция h=tol(tolerance) возвращает прежнее и устанавливает новое значение точности, но только для следующей за ней операции триангуляции, метода наименьших квадратов, псевдоинверсии.

Моделирование динамических систем.

Пусть модель пространства состояний системы dX/dt=AX+Bu, y=CX+du задана матрицей S=[A B;C d].

Рассмотрим также передаточную функцию y=Qu, Q=N(p)/D(p), тогда S=tf(N,D) от коэффициентов полиномов.

Ступенчатая и импульсная весовые функции вычисляются как обычно.

Пусть t=time(T), тогда h=step(S,t), q=impulse(S,t).

Преобразования моделей также достаточно традиционны.

Операции S=tf2ss(N,D), [A,B,C,d]=tf2ss(N,D), [N,D]=ss2tf(rcform(S)), также [A,B,C]=tf2ss(N,D), [A,B]=tf2ss(N,D), A=cut(S), B=cut(cutl(S,rows(A))), C=cut(cutu(S,rows(A))).

Для вычисления Q(p)=N(p)/D(p)=N1(p)/D2(p)*N2(p)/D2(p) используется операция конволюции N=conv(N1,N2), D=conv(D1,D2). A=compan(D) – это матрица Фробениуса.

Канонические формы динамических систем.

Сбалансированная каноническая форма F=bcform(S). Столбцовая и строчная реализации F=ccform(S), F=rcform(S). [F,T,H]=bcform(S) или [F,T]=ccform(S), [F,T]=rcform(S) возвращают матрицу преобразования A=T\AT, B=T\B, C=CT, d=d и ганкелевы сингулярные числа H.

Временная область.

Пусть dt будет шаг по времени (matrices: dt), X это вектор начального состояния, U это вектор отсчетов скалярного входного сигнала.

Выходной сигнал: Y=lsim(S,[U;dt]), причем Y=lsim([S [X;0]],[U;dt]) учитывает начальное состояние X, X=lsim([S [X;1]],[U;dt]) вычисляет финальное состояние.

Пусть A=A*dt, B=B*dt , тогда X=statestep(A) это следующий шаг системы dX/dt=AX, функция двойного аргумента X=statestep(A,B*u) возвращает следующий шаг системы dX/dt=AX+Bu.

Интегратор: Y=int(U,dt) или Y=int(U,t).

Дискретные системы. Выходной сигнал: Y=dsim(S,U), причем Y=dsim([S [X;0]],U) соответствует начальному состоянию X, X=dsim([S [X;1]],U) вычисляет финальное состояние. Дискретный интегратор: Y=int(U).

Частотные характеристики.

Передаточная функция Q(p)=N(p)/D(p) задана коэффициентами полиномов. F=rcform(S) возвращает строчную каноническую форму F=[A B;C d], также справедливо F=tf(N,D) или F=tf2ss(N,D), причем [N,D]=ss2tf(F).

Диаграммы Боде m=bode(F,w), m=?D и Найквиста G=nyquist(F,w), G=?2D для w=[0 W 2W ...]' с шагом W. Можно установить и комплекснозначную выборку в виде двойной w=[re im]-колонки.

Матричные алгебраические уравнения.

Алгебраическое уравнение Риккати:

P*A+A'*P-P*B/R*B'*P=-Q, P=are([A -B/R*B'; -Q -A'])

Уравнение Ляпунова:

P*A+A'*P=-Q, P=are([A zeros(A); -Q -A'])

Решение оптимальных задач:

dX/dt=AX+Bu, Y=CX+dU, функционал f=0.5 § (X'QX+U'RU) dt -> min

Пусть H=[A -B/R*B'; -Q -A'], P=are(H), K=B'/R*P

Линейный регулятор: U=-KX.

Операторы линейных динамических систем.

Пусть задана импульсная весовая функция q=impulse(S,t) с шагом T, тогда матрица оператора свертки будет M=T*toeplitz(q).

В общем случае M=lom(S,t,'Option'), Y=lsim(S,[U;dt],'Option'). Для дискретных систем: M=dom(S,t,'option'), Y=dsim(S,U,'Option').

Вычисление главной сингулярной функции и сингулярного числа:

f=eigf(S,t,'Option'), [f,d]=eigf(S,t,'Option')

'Option' для свертки это S, для сопряженного FSF оператора это A (adjoint), для прочих составных операторов FS, SF, FSF, S+A, S-A, SA, S+FS, S-FS, S+SF, S-SF, FS+SF, FS-SF, для теплицева оператора это T, ганкелева FT оператора это H, для прочих составных операторов FT, TF, FTF, T+A, T-A, T+FT, T-FT, T+TF, T-TF, FT+TF, FT-TF, все операторы 0.5-нормализованы.

Нелинейные системы.

Маятник: dX/dt=F(X),

F=chain([Masses;Lengths],X) возвращает производные F(X), X=[A;dA/dt] где A это относительные углы элементов цепочки.

X=solve(dt,'dX/dt=f(X)') это шаг дифференциального уравнения.

Y=qsim(a,Yo*line(size)) это процесс Yk+1=a Yk (1 - Yk) .

Множество Мандельброта: Zk+1=C+Zk*Zk, Z=[Re Im], Zo=[0 0], M=qsim(Mo,size), для C из Mo=[Re1 Re2 Im1 Im2].

Множество Джулиа: Zk+1=C+Zk*Zk, Z=[Re Im], для Zo из Mo, M=qsim(Mo,[size max(| Z |) C] для некоторого C=[Re Im].

M содержит финальные нормы | Z | для раскраски M=?CR.

Генетические алгоритмы:

G=crossing(G) возвращает [G; crossing+mutation]. G=crossing(max,G) или G=crossing([min;max],G) с насыщением

П.3. ВИЗУАЛИЗАЦИЯ В VISUAL MATLAB

Помимо языка матричных вычислений VISUAL MATLAB поддерживает язык управления картинками. Синтаксис приближен к синтаксису обычного языка, причем, как отмечалось, есть адаптаторы к прочим европейским языкам, включая русский (до семи языков).

Рис. П.3. Редактор сцены VISUAL MATLAB

Персонажи и части векторных рисунков именуются редактором сцены, см. П.3. Векторные рисунки могут содержать родительские и дочерние сегменты, подобно тому, как на руке имеются пальцы, а пальцы имеют фаланги. Сложное подлежащее имеет вид: cat head (голова кота) и т.д.

СЦЕНАРИЙ

Сценарий, это действие, заключенное обычно в скобки цикла. Запятая связывает интерпретируемые предложения, описывающие одновременное движение частей, точка служит цели вывода синтезированного изображения на экран монитора.

ЦИКЛЫ REPEAT, DO (СЧЕТЧИК ТАКТА T)

Repeat ... show end

Do [размер цикла N] ... end

For T Do, ... end

For T=1:N Do, ... end

Do while T<N, ... end

Размер цикла указан в квадратных скобках, он может отсутствовать.

В добавляемом пользователем к сценарию словаре новых слов (глаголов) вместо прямого используется безличное обращение к подлежащему subject (объект) или местоимения he (он), she (она), it (оно), his (его), her (ее), this (этот, эта), that (тот,та).

ПРАВИЛА ФОРМИРОВАНИЯ НОВЫХ СЛОВ

Глагол (фактически, это подпрограмма) описывается в императиве (go), в форме третьего лица (goes), может быть добавлена также инговая форма (going), через запятую:

word: go, goes, going

<описание глагола>

Разрешается ветвить действие глагола анализом наречий (adverb), подлежащего (name), отсутствия рисунка (absent).

IF (ВЕТВЛЕНИЯ ПО НАРЕЧИЯМ И Т.П.)

If 'adverb' then ... else ... end

If subject is 'name' then ... else ... end

If subject is absent then ... else ... end

If a>b then ... else ... end

Объект имеет имя 'name' (конкретное имя или наречие употребляется без кавычек) и состояние state (ячейка памяти, ее можно, например, увеличивать state=state+1). Эта переменная позволяет описывать протяженное действие (continuous verbs).

Логические знаки в операции сравнения обычные <,>,<>,=,>=,<=,~=,==

БАЗОВЫЕ ГЛАГОЛЫ

Перечислим базовый набор глаголов, их всего несколько. Переменные, указанные в квадратных скобках, должны быть заменены на конкретные (без скобок) или пропущены (тогда берется их предыдущее значение). Все, что указано ниже в кавычках '..' заменяется конкретными именами файлов (name, без кавычек) или персонажей (subject, без кавычек). Если персонаж считан из файла, с именем name, то имя этого файла становится именем персонажа.

OPEN (ОТКРЫТЬ ФАЙЛ)

Open subject 'name.bmp/jpg' % открыть персонаж (bmp-фон прозрачен)

Open background 'name.bmp/jpg/wmf' [X:Y] % открыть фон размера X:Y

Open background 'name.bmp/jpg/wmf' [:] % открыть фон

Open background 'name01.jpg' [:] % открыть нумерованный файл

Open background >. % следующий кадр

Open background <. % предыдущий

Open [foreground] 'name[.box]' % открыть векторный рисунок

Open scene 'name' % открыть сцену

SHOW (ПОКАЗАТЬ)

Show обновляет экран, точка играет ту же роль!

MOVE (ПЕРЕМЕЩАЕТСЯ)

'Subject' moves [distance] up/down/left/right % дистанция в пикселах

'Subject' moves at [level] vertical/horizontal line % в пикселах

'Subject' moves at [x;y[;rad]] position % в пикселах и радианах

'Subject' moves to front/back

TURN (ПОВОРАЧИВАЕТСЯ, НЕ ДЛЯ ФОТО)

'Subject' turns [angle] [forward/backward] % относительный угол в радианах

'Subject' turns at [absolute angle] [forward/backward] % абсолютное значение

FLIP (ИНВЕРТИРУЕТСЯ)

'Subject' flips up-down/left-right/up/down/left/right % сверху-вниз или слева-направо

EXPAND-DIMINISH (УВЕЛИЧИВАЕТСЯ-УМЕНЬШАЕТСЯ)

'Subject' expands [number] [vertically/horizontally] % в количество раз

'Subject' expands [number] [up/down/left/right] % искажение вверх, вниз, влево, вправо

'Subject' expands [number] [up-right/up-left] % перекосы направо и налево

'Subject' diminishes [number] [vertically/horizontally] % в количество раз

'Subject' diminishes [number] [up/down/left/right] % искажение вверх, вниз, влево, вправо

'Subject' diminishes [number] [up-right/up-left] % перекосы направо и налево

LET X BE ... COORDINATE[S] P=[X Y]', S=[X Y Z]'

[Let] P are 'subject' position coordinates % вернуть позиционные координаты, пикселы

[Let] S are 'subject' coordinates % вернуть все координаты [x,y,rad]'

[Let] X is 'subject' horizontal coordinate % горизонтальная координата

[Let] Y is 'subject' vertical coordinate % вертикальная координата

[Let] Z is 'subject' angular coordinate % deg, в градусах

APPEAR-DISAPPEAR (ПОЯВЛЯЕТСЯ-ИСЧЕЗАЕТ)

'Subject' appears

'Subject' disappears

COPY (присоединить персонажи к фоновому изображению)

Copy foreground to background

Copy 'subject' to background

PLAY (ИГРАТЬ, ГОЛОС & МУЗЫКА)

Play name.wav % голос

Play name.mid/rmi % музыка

Replay, stop play, continue the play (перезапуск, остановка, продолжить)

PAUSE (ПАУЗА)

Pause N % в секундах

Рассмотрим процесс создания сцены при помощи системы визуализации научных расчетов VISUAL MATLAB. Предположим, что изображение визуализируемого объекта хранится в файле с типом BMP. Для того, чтобы на экране отсутствовал фон, проецировался именно объект, а не квадрат с объектом, этот фон должен быть монотонным. Тогда по первой же угловой точке изображения система может делать фон прозрачным.

Разрешается импортировать рисунок непосредственно из текста сценария, однако для большего удобства начального размещения объектов они могут быть открыты при помощи меню Tools (строчка Designer).

На панели дизайнера имеется кнопка браузера, для захвата рисунков объектов и кнопка с ножницами (ее нажатие делает фон под рисунком прозрачным). Там же расположено окно для присвоения уникального имени рисунку.

Объект полезно оцентровать (меню Edit/Autocentering). Там же, в меню Tools, имеется позиция вызова панели управления объектом Control Handle.

На панели имеются кнопки перемещения и сжатия изображения на экране. Designer содержит генераторы многоугольников, а Control Handle позволяет их деформировать, сжимать, разжимать, дорисовывая части рисунка, или создавая векторный рисунок (без BMP). Векторные рисунки значительно более мобильны, в ряде случаев можно обойтись только при помощи их, не привлекая пиксельные изображения.

Объект на экране разрешается размещать просто мышкой, кликом левой кнопки. Кликом правой кнопки мыши разрешается менять объект манипуляции (из имеющихся на экране выбирается тот, который ближе всего к курсору мышки). Designer содержит также кнопку Map, открывающую карту, иллюстрирующую разложение рисунка на сегменты. С ее помощью можно добавить сегменты, каждый сегмент можно разложить на субсегменты и так далее (те есть, на кисти могут быть отдельно управляемые пальцы). Всего имеется пять уровней вложений, минимальный содержит точки, из которых состоят векторные многоугольники. Тем самым, этими точками тоже можно управлять, подбирая их форму.

Созданная сцена называется "передний план" (foreground) и записывается в файл с расширением box (коробка). В качестве заднего плана (background) разрешается использовать любой BMP или JPG рисунок. После создания сцены с визуализируемыми персонажами наступает черед написания сценария визуализации, представляющего собой обычный текст.