

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
Белорусский национальный технический университет

Кафедра «Тракторы»

А. С. Поварехо
В. Н. Плищ

САПР МАШИН. ИНЖЕНЕРНЫЙ АНАЛИЗ В СРЕДЕ MATLAB-SIMULINK

Пособие

для обучающихся по специальностям
1-37 01 03 «Тракторостроение», 1-37 01 04 «Многоцелевые
гусеничные и колесные машины (по направлениям)»,
1-37 01 05 «Электрический и автономный транспорт»

*Рекомендовано учебно-методическим объединением по образованию
в области транспорта и транспортной деятельности*

Минск
БНТУ
2022

УДК 629.3:658.512.22:004.9(075.8)

ББК 39.33я7

П42

Р е ц е н з е н т ы:

зав. кафедрой «Тракторы и автомобили» Белорусского государственного аграрного технического университета,
канд. техн. наук, доцент *Г. И. Гедроить*;
заместитель начальника УКЭР-1 по серийному производству и трансмиссиям ОАО «МТЗ», канд. техн. наук *В. Г. Ермаленок*

Поварехо, А. С.

П42

САПР машин. Инженерный анализ в среде MATLAB-Simulink : пособие для обучающихся по специальностям 1-37 01 03 «Тракторостроение», 1-37 01 04 «Многоцелевые гусеничные и колесные машины (по направлениям)», 1-37 01 05 «Электрический и автономный транспорт» / А. С. Поварехо, В. Н. Плищ. – Минск : БНТУ, 2022. – 73 с.

ISBN 978-985-583-702-3.

Учебно-методическое пособие содержит 8 лабораторных работ по курсу «САПР узлов и агрегатов машин», «САПР узлов и агрегатов трактора» для студентов специальностей 1-37 01 04 «Многоцелевые гусеничные и колесные машины», 1-37 01 03 «Тракторостроение», 1-37 01 05 «Электрический и автономный транспорт».

Лабораторные работы предусматривают изучение компьютерных технологий инженерного анализа с помощью симуляционных средств пакета MATLAB Simulink, применяемых при решении ряда научных и инженерных проблем. Применение компьютерных технологий решения задач посредством системы MATLAB иллюстрируется примерами и упражнениями.

Пособие предназначено для студентов и аспирантов высших технических учебных заведений, будет полезна специалистам и научным работникам технического профиля.

УДК 629.3:658.512.22:004.9(075.8)

ББК 39.33я7

ISBN 978-985-583-702-3

© Поварехо А. С., Плищ В. Н., 2022

© Белорусский национальный
технический университет, 2022

ВВЕДЕНИЕ

Целью выполнения лабораторных работ является углубление и закрепление знаний, полученных студентами на лекционных занятиях и в процессе самостоятельной работы по курсу САПР.

В методических указаниях рассмотрены основы работы с некоторыми приложениями пакета MATLAB. Выбор в качестве платформы для функционального проектирования пакета MATLAB обусловлен высоким рейтингом компании MathWorks в области инженерного анализа, а также широким использованием указанного пакета в научно-исследовательских институтах и базовых для специальностей 1-37 01 04 «Многоцелевые гусеничные и колесные машины», 1-37 01 03 «Тракторостроение», 1-37 01 05 «Электрический и автономный транспорт» предприятиях РБ.

Приведены методические рекомендации и описание восьми лабораторных работ согласно учебной программе курса «САПР узлов и агрегатов машин». Работы включают теоретические сведения, касающиеся моделирования (инженерного анализа и синтеза) типовых элементов технических объектов, варианты заданий и требования к выполнению работы и оформлению отчета. Данные работы должны выполняться с использованием пакета MATLAB.

Выполнение приведенных в методических указаниях лабораторных работ предполагает обязательное наличие у студентов базовых знаний таких дисциплин, как «Механика материалов», «Теория механизмов и машин», «Детали машин», «Конструирование и расчет» для указанных выше специальностей.

Методические указания могут быть полезными при выполнении курсового и дипломного проектирования, а также для подготовки магистерских выпускных работ.

Перед началом проведения лабораторных работ студенты проходят инструктаж по технике безопасности и правилам противопожарной техники с отметкой о прохождении инструктажа в специальном журнале лаборатории. За инструктаж ответственен преподаватель, проводящий лабораторные работы.

ТРЕБОВАНИЯ К ВЫПОЛНЕНИЮ РАБОТ

Выполняемые лабораторные работы сводятся к следующему.

1. На основе приведенной в методичке или разработанной математической модели составить программу, структурную или физическую модель в **MATLAB**.

2. Произвести проверку и анализ разработанной модели путем проведения расчета с исходными данными согласно индивидуальному варианту.

3. Построить требуемые графические зависимости, отражающие функционирование моделируемого объекта, используя графические средства **MATLAB** или пакета **Grapher Golden Software, LLC**.

При выполнении работ для обеспечения понятной структуры алгоритма программы должно соблюдаться следующее:

- описание математической модели при программировании содержится в отдельном m-файле;

- передача необходимых для функционирования модели данных осуществляется только через формальные параметры функции (подпрограммы);

- «управляющая программа» формируется согласно поставленной задаче. В большинстве работ это реализация математической модели для построения графических зависимостей;

- ввод исходных данных для расчета и вывод результатов осуществляется только с использованием файлов данных;

- графики строятся с соблюдением требований к оформлению графической документации.

Допускается в отчетах приводить скриншоты программ, схем и графиков. Информация на скриншотах должна быть полностью читаема и оформлена в соответствии с заданием и нормативными документами.

Лабораторная работа № 1

ИНЖЕНЕРНЫЙ АНАЛИЗ С ПОМОЩЬЮ СРЕДСТВ ПРОГРАММИРОВАНИЯ В MATLAB

Цель работы: научиться составлять программы для решения систем обыкновенных дифференциальных уравнений (ОДУ), описывающих динамические процессы в технических объектах с помощью средств программирования пакета **MATLAB**.

Общие сведения

В основном существует два широких класса численных методов решения дифференциальных уравнений.

1. Одноступенчатые методы, в которых используется только информация о самой функции в одной точке и не производятся итерации. При использовании данных методов трудно оценивать допускаемую ошибку.

2. Многоступенчатые методы, в которых для нахождения следующей точки функции с достаточной точностью требуются итерации. Большинство методов этого класса называются методами прогноза и коррекции. Оценку ошибки при этом легко получить в качестве побочного продукта вычислений.

Для решения дифференциальных уравнений и систем в **MATLAB** предусмотрены следующие функции:

ode45(f, interval, X0 [, options]),
ode23(f, interval, X0 [, options]),
ode113(f, interval, X0 [, options]),
ode15s(f, interval, X0 [, options]),
ode23s(f, interval, X0 [, options]),
ode23t (f, interval, X0 [,options]),
ode23tb(f, interval, X0 [, options]).

Входными параметрами этих функций являются: f – вектор-функция для вычисления правой части уравнения в системе; *interval* – массив из двух чисел, определяющий интервал интегрирования дифференциального уравнения или системы; $X0$ – вектор начальных условий системы дифференциальных уравнений; *options* –

параметры управления ходом решения дифференциального уравнения или системы.

Все функции возвращают: массив T – координаты узлов сетки, в которых ищется решение; матрицу X , i -й столбец которой является значением вектор-функции решения в узле T_i .

В решателях **MATLAB** используются следующие алгоритмы:

1. Алгоритмы, предназначенные для решения нежестких систем:

– **ode45** основан на явной формуле Рунге-Кутты (4,5), Дорманда-Принца. Это одношаговый решатель – при вычислении $y(t_n)$ ему нужно только решение в непосредственно предшествующий момент времени $y(t_{n-1})$. В общем, **ode45** – это лучшая функция для применения в качестве «первой попытки» для большинства проблем;

– **ode23** – это реализация явной формулы Рунге-Кутты (2,3), Богацкого и Шампина. Он может быть более эффективным, чем **ode45**, при грубых допусках и при наличии умеренной жесткости. Как и **ode45**, **ode23** – одношаговый решатель;

– **ode113** – это решатель переменного порядка (использует методы Adams-Bashforth-Moulton, PECE). Он может быть более эффективным, чем **ode45**, при строгих допусках и когда функция файла ODE особенно весома для оценки. **ode113** – это многоступенчатый (итерационный) решатель, ему нужны решения в нескольких предыдущих временных точках для вычисления текущего решения.

Если рассмотренные алгоритмы окажутся чрезмерно медленными, следует попробовать один из решателей для жестких систем, приведенных ниже.

2. Алгоритмы, предназначенные для решения жестких систем:

– **ode15s** – это решатель переменного порядка, основанный на формулах численного дифференцирования (NDFs). При необходимости он использует формулы обратного дифференцирования (BDFs, также известные как метод Гира), которые обычно менее эффективны. Как и **ode113**, **ode15s** – многоступенчатый решатель. Целесообразно использовать **ode15s**, когда **ode45** терпит неудачу или очень неэффективен, и есть подозрение, что система жесткая, а также при решении дифференциально-алгебраических уравнений (DAE);

– **ode23s** основан на модифицированной формуле Розенброка второго порядка. Так как это одношаговый решатель, он может быть более эффективным, чем **ode15s** при грубых допусках. Он может решить некоторые проблемные задачи, для которых **ode15s** не эффективен;

– *ode23t* – это реализация трапециевидного правила, которое является численным методом для решения обыкновенных дифференциальных уравнений. Это неявный метод второго порядка, который можно рассматривать и как метод Рунге-Кутты, и как линейный многоступенчатый метод. Решатель целесообразно использовать, если задача только умеренно жесткая и нужно решение без численного затухания. *ode23t* может решить дифференциально-алгебраические уравнения (DAE);

– *ode23tb* – это TR-BDF2 реализация неявной формулы Рунге-Кутты с первым этапом, представляющим собой шаг трапециевидного правила, и вторым этапом, представляющим собой обратную формулу дифференцирования второго порядка. По своей конструкции при оценке обоих этапов используется одна и та же итерационная матрица. Как и *ode23s*, этот решатель может быть более эффективным, чем *ode15s* при грубых допусках.

Решение ОДУ

Необходимо решить дифференциальное уравнение:

$$\frac{dy}{dx} = \cos(x + y) + \frac{3}{2} \cdot (x - y).$$

Начальное условие: $y(0) = 0$. Интервал интегрирования $[0-20]$.

Создаем **m**-файл с именем *primer_odu.m* и пишем:

```
function f = primer_odu (x,y)
f = cos(x+y) + (3/2)*(x-y);
end
```

Затем в командном окне (или создав

Script) вызываем функцию *ode113*:

```
[x, y] = ode113(@primer_odu, [0
20], 0, 0)
plot(x, y)
```

Результатом будет график, приведенный на рис. 1.1.

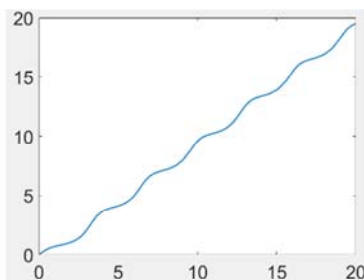


Рис. 1.1. График решения ОДУ

Решение систем ОДУ

Пример 1. Решение нежесткой системы ОДУ.

Примером такой системы является система уравнений, описывающая движение твердого тела без внешних сил.

$$\begin{aligned} \dot{y}_1 &= y_2 \cdot y_3; & y_1(0) &= 0; \\ \dot{y}_2 &= -y_1 \cdot y_3; & y_2(0) &= 1; \\ \dot{y}_3 &= -0.51 \cdot y_1 \cdot y_2; & y_3(0) &= 1. \end{aligned}$$

Чтобы смоделировать эту систему, создаем функцию *non_rigid*, содержащую систему уравнений

```
function dy = non_rigid(t,y)
dy = zeros(3,1); % вектор-столбец
dy(1) = y(2) * y(3);
dy(2) = -y(1) * y(3);
dy(3) = -0.51 * y(1) * y(2);
```

В этом примере изменяем стандартные допуски ошибок с помощью команды *odeset*:

```
options = odeset('RelTol',1e-4,'AbsTol',[1e-4 1e-4 1e-5]);
```

Решаем на временном интервале [0 12]:

```
[t,y] = ode45(@non_rigid,[0 12],[0 1 1],options);
```

Для построения графических зависимостей используем столбцы возвращаемого массива *y* в зависимости от *t* (рис. 1.2):

```
plot(t,y(:,1),'-',t,y(:,2),'-',t,y(:,3),'-')
```

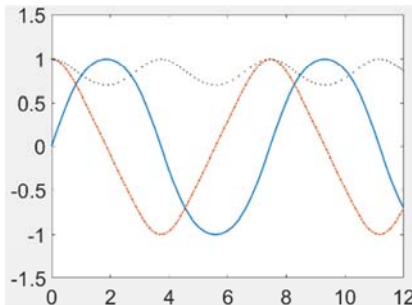


Рис. 1.2. Решение нежесткой системы ОДУ

Пример 2. Решение жесткой системы ОДУ.

Примером жесткой системы являются уравнения Ван-Дер-Поля в релаксационных колебаниях. Цикл колебаний имеет участки, где компоненты решения изменяются медленно, а задача нежесткая, чередующаяся с областями очень резкого изменения, где она довольно жесткая.

$$\begin{aligned} \dot{y}_1 &= y_2; & y_1(0) &= 0; \\ \dot{y}_2 &= 1000 \cdot (1 - y_1^2) \cdot y_2 - y_1; & y_2(0) &= 0. \end{aligned}$$

Для решения создадим функцию **rigid**, содержащую уравнения:

```
function dy = rigid(t,y)
dy = zeros(2,1); % вектор-столбец
dy(1) = y(2);
dy(2) = 1000*(1 - y(1)^2)*y(2) - y(1);
```

Используем относительные и абсолютные допуски по умолчанию и решим систему на временном интервале [0 3000].

```
[t,y] = ode15s(@vdp1000,[0 3000],[2 0]);
```

Для визуализации решения построим графическую зависимость первого столбца возвращаемой матрицы **Y** от **T** (рис. 1.3):

```
plot(t,y(:,1),'-o')
```

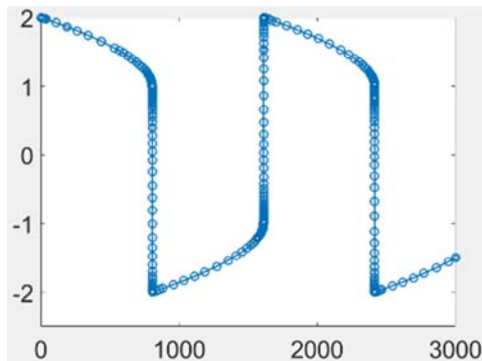


Рис. 1.3. Решение жесткой системы ОДУ

Понижение порядка ОДУ

Приведенные выше функции **MATLAB** осуществляют решение ОДУ первого порядка. Однако большинство задач инженерного анализа механических систем описываются ОДУ второго порядка.

Рассмотрим понижение порядка дифференциальных уравнений на примере расчета вертикальных колебаний упруго подрессоренной массы при безотрывном движении колеса по неровному участку пути (рис. 1.4). Колебательная система имеет следующие параметры: m – масса груза; c – коэффициент жесткости упругой подвески; k – коэффициент демпфирования устройства гашения колебаний (например, амортизатор); v – постоянная горизонтальная скорость системы.

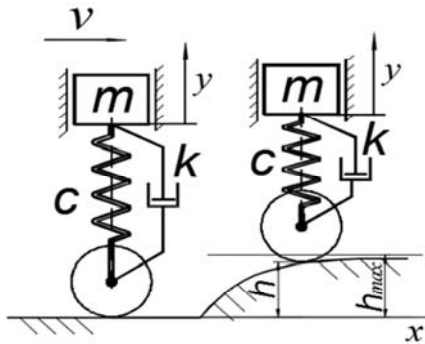


Рис. 1.4. Расчетная схема вертикальных колебаний одномассовой системы

Данная система описывается ОДУ второго порядка:

$$m \cdot y'' = -c \cdot (y - h(t)) - k \cdot (\dot{y} - \dot{h}(t)). \quad (1.1)$$

Принимаем, что профиль опорной поверхности участка задан экспонентой:

$$h(x) = h_{\max} \cdot (1 - e^{-\alpha \cdot x}),$$

где h_{\max} – максимальная высота неровности;

α – параметр, характеризующий кривизну профиля.

Для решения задачи при использовании функций **MATLAB** заменим исходное уравнение эквивалентной системой двух дифференциальных уравнений 1-го порядка. Для этого введем вспомогательные переменные $z_1 = y$; $z_2 = \dot{y}$. Тогда уравнение второго порядка, приведенное к системе ОДУ первого порядка, имеет вид

$$\begin{cases} \dot{z}_1 = z_2, \\ \dot{z}_2 = (-c \cdot (z_1 - h(t)) - k \cdot (z_2 - \dot{h}(t))) / m. \end{cases}$$

Для преобразования уравнения профиля в зависимость от времени учитываем скорость движения массы v : $x = v \cdot t$.

Тогда $h(t) = h_{\max} \cdot (1 - e^{-\alpha \cdot v \cdot t})$.

Численные значения параметров системы приведены в табл. 1.1.

Таблица 1.1

Параметры одномассовой системы

Масса груза, кг	Коэффициент жесткости, c Н/м	Коэффициент демпфирования, k Н·с/м	Высота неровности, h_{\max} , м	Коэффициент кривизны профиля, α
380	19000	900	0,1	4,6

Для решения создается **m**-файл, в котором вычисляются правые части системы дифференциальных уравнений:

```
function dz=ind_zad6(t,z)
global c k m hmax alfa v
h=hmax*(1-exp(-alfa*v*t));
hp=alfa*v*hmax*exp(-alfa*t);
dz=zeros(2,1);
dz(1)=z(2);
dz(2)=(-c*(z(1)-h)-k*(z(2)-hp))/m;
end
```

Для решения, вызываем функцию **ode45** и по результатам вычислений строим график (рис. 1.5) с помощью функции **plot**:

```
global c k m hmax alfa v
c=19000; m=380; k=900; alfa=4.6; v=1;
```

```
[t,y]=ode45(@ind_zad6,[0 4],[0 0]);
plot(t,y)
```

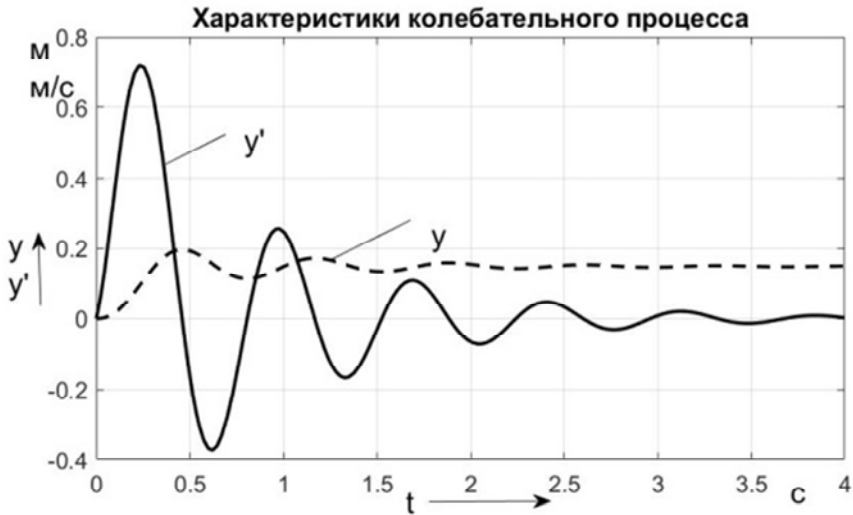


Рис. 1.5. Зависимости скорости и перемещения массы

Задание

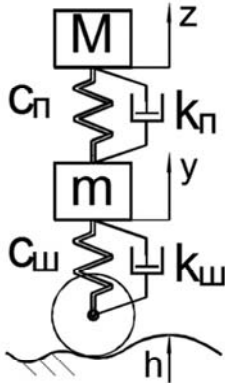


Рис. 1.6. Расчетная схема

1. Составить математическую модель подвески в соответствии с вариантом задания и расчетной схемой (табл. 1.2, рис. 1.6).

2. Составить программу для анализа колебаний поддрессоренной и недрессоренной масс при заданном законе изменения возмущающего воздействия (табл. 1.3, 1.4) в MATLAB.

3. По результатам моделирования построить зависимости перемещений и скоростей поддрессоренной и недрессоренной масс при наезде объекта на заданную неровность.

4. Оформить отчет, содержащий математическую модель, распечатку программы, графики зависимостей согласно п. 3.

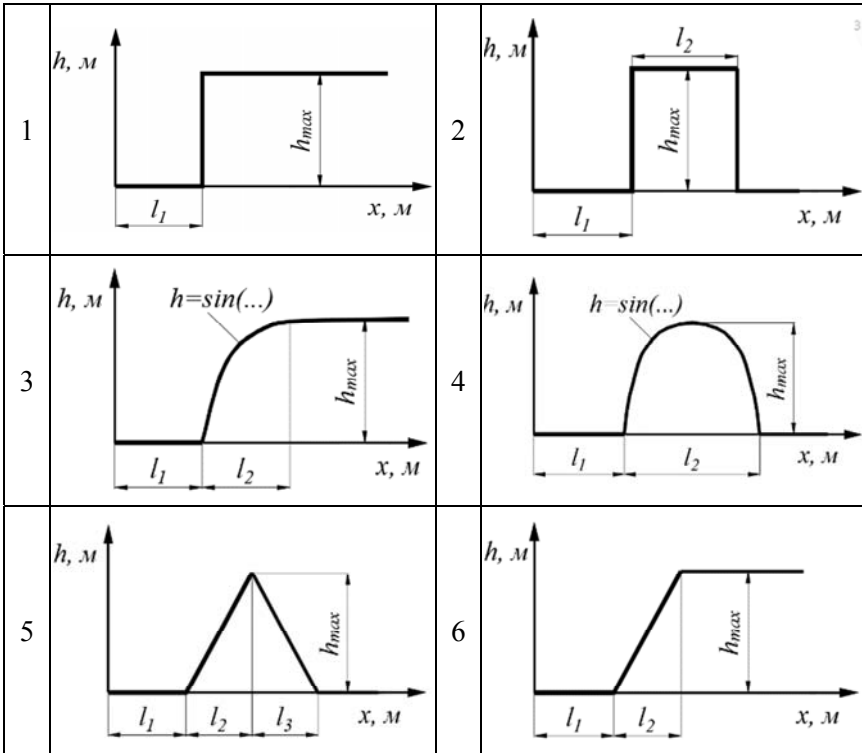
Таблица 1.2

Варианты заданий (выбираются по номеру в списке группы)

Вариант	m , кг	M , кг	$c_{п}$, кН/м	$k_{п}$, Н·с/м	$c_{ш}$, кН/м	$k_{ш}$, Н·с/м
1–12	50	300	20	780	1000	200
13–24	50	400	25	2100	1200	250

Таблица 1.3

Параметры микронеровности
(выбираются согласно списку группы)



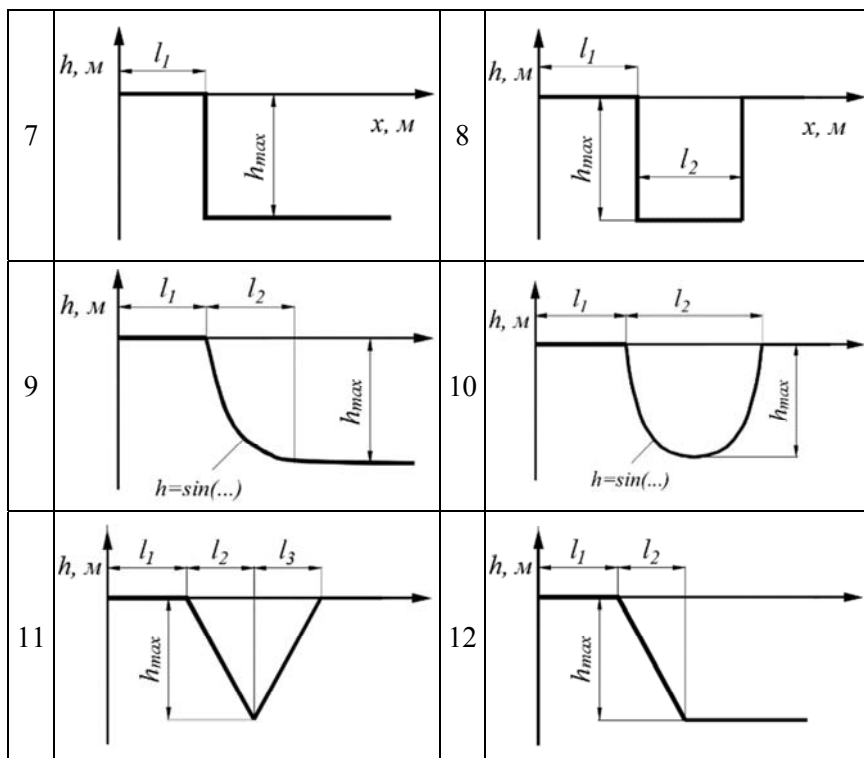


Таблица 1.4

Параметры неровности

Вариант	Неровность	l_1 , м	l_2 , м	l_3 , м	h_{\max} , м
1–12	1–12	0,5	1,2	0,6	0,15
13–24	1–12	0,4	0,9	0,7	0,20

Лабораторная работа № 2

ИНЖЕНЕРНЫЙ АНАЛИЗ С ПОМОЩЬЮ СРЕДСТВ СТРУКТУРНОГО МОДЕЛИРОВАНИЯ В MATLAB SIMULINK

Цель работы: научиться составлять структурные схемы решения систем (ОДУ), описывающих динамические процессы в технических объектах, с помощью средств структурного моделирования в среде **Simulink** пакета **MATLAB**.

Общие сведения

Simulink представляет отдельное приложение и содержит библиотеки различных блоков (рис. 2.1), из которых в рабочем окне строится структурная схема модели. Процесс создания математической модели и ее программирования в явном виде отсутствует. Особенно удобен такой подход для моделирования систем автоматического управления, заданных структурными схемами, дифференциальных уравнений, механических систем и электрических схем.

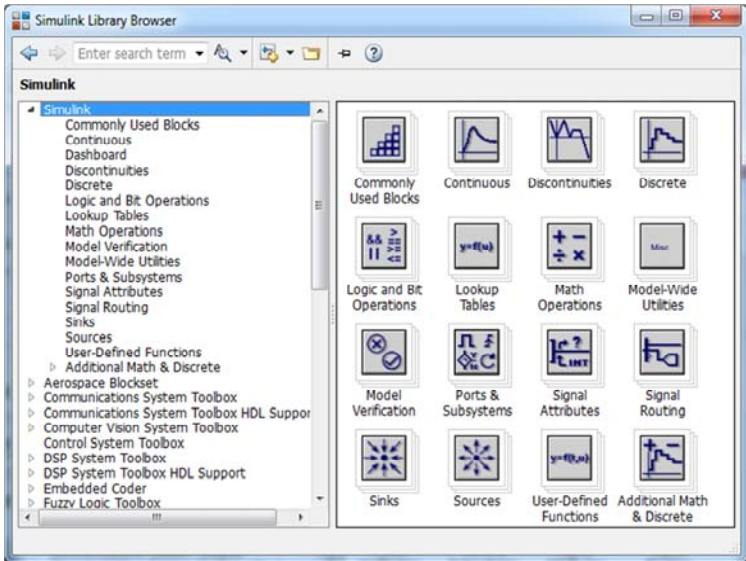


Рис. 2.1. Библиотека блоков базового **Simulink**

Базовые принципы создания моделей в Simulink

Модели состоят из блоков (элементов, звеньев), которые соединяются между собой связями, по которым происходит передача сигналов.

Каждый сигнал представляет собой некоторую величину, зависящую от времени, и может иметь разный физический смысл и размерность, моделируя поведение объектов различной природы.

Сигналы являются аналогами переменных в традиционном программировании. Чаще всего они соответствуют переменным вещественного типа, но могут быть логическими (принимать одно из двух значений: ноль и не ноль) или векторными (передавать одновременно значения несколько величин).

Звенья используются для генерации и преобразования сигналов.

При запуске модель описывается соответствующими уравнениями: автоматически создается программа для их численного решения и вычисляются значения сигналов, описывающие состояние системы в каждый момент времени.

Для создания новой модели выбирается *Simulink Library*, а затем в открывшемся окне – кнопка *New Model* (рис. 2.2). Аналогичный результат получается выбором в меню пути *HOME* → *New* → *Simulink Model*.

В открывшемся окне путем перетаскивания при нажатой левой кнопке мыши необходимых блоков из окна *Simulink Library Browser* создается схема решения уравнения. Если щелкнуть мышью по блоку в модели, то откроется окно параметров блока, вид которого зависит от типа блока и в котором можно задать параметры, касающиеся функциональных особенностей выбранного блока.

Для решения уравнения в *Simulink* используется блок *Integrator* (класс *Continuous*). На его вход подается производная y' , а на выходе получается величина y .

Настроить параметры *Simulink* можно, выполнив команду главного меню *Simulation* → *Model Configuration Parameters*, нажав *Ctrl+E* или выбрав в меню *File* → *Simulink Preferences*.

Процесс настройки заключается в задании параметров решающего модуля (*Solver*):

Simulation time (время моделирования) – временной интервал моделирования в секундах.

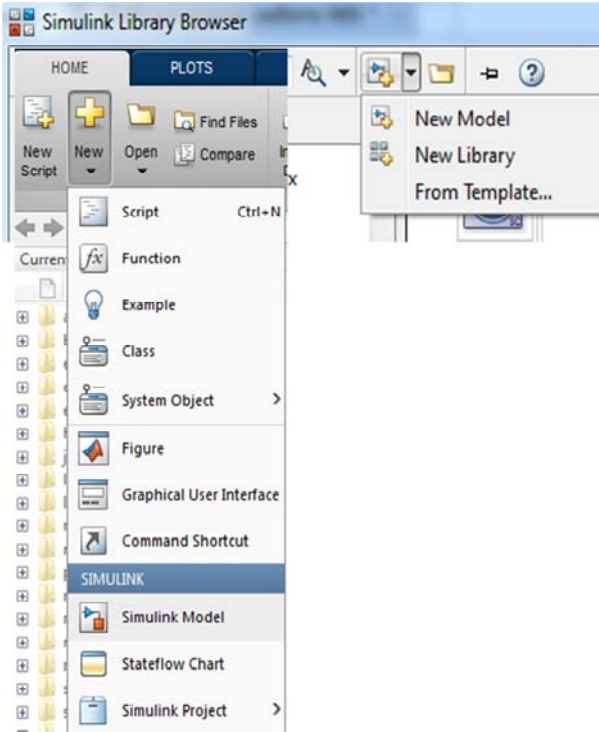


Рис. 2.2. Меню загрузки **Simulink**

Solver Options (параметры решающего модуля) – параметры модуля, реализующего один из методов численного решения интегрирования обыкновенных дифференциальных уравнений.

Структурные модели для решения ОДУ первого порядка

Рассмотрим пример составления структурной модели для дифференциального уравнения с начальными условиями: $t = 0, y_0 = 0$:

$$y'(t) + 2 \cdot y(t) = \sin(t).$$

Структурная схема, соответствующая данному уравнению, представлена на рис. 2.3.

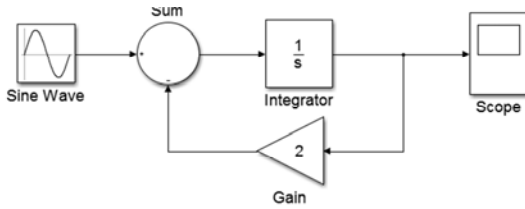


Рис. 2.3. Структурная схема для решения ОДУ

Блоки **Sum** и **Gain** (библиотека **Math Operations**) необходимы для формирования значения y' в соответствии с ОДУ. Для получения сигнала $\sin(t)$ используется блок **Sine Wave** (библиотека **Sources**). Открыв соответствующий блок двойным щелчком мыши или выбрав опцию **Block Parameters** при нажатой правой кнопке мыши, можно задать его параметры. Пример задания параметров для блока **Gain** приведен на рис. 2.4.

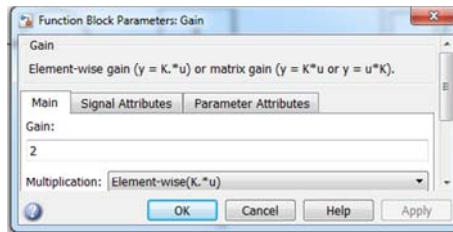


Рис. 2.4. Задание параметров для блока **Gain**

В настройках параметров **Simulink** задается время расчета 2 с.

Полученное в результате решения значение $y(t)$ подается на вход блока **Scope**. При открытии данного блока появляется график решения (рис. 2.5).

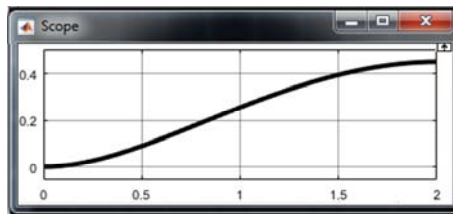


Рис. 2.5. Результаты решения ОДУ в **Simulink**

Структурные модели для решения систем ОДУ первого порядка

Рассмотрим решение системы ОДУ первого порядка на примере приведенной ниже системы обыкновенных дифференциальных уравнений.

Зададим параметры задачи: $\alpha = 0,1$; $\beta = 0,05$; $\gamma = 0,03$; $\delta = 0,2$; $\varepsilon = 0,15$.

$$\begin{cases} \frac{dx}{dt} = x \cdot (\alpha - \beta \cdot y - \gamma \cdot x) \\ \frac{dy}{dt} = -y \cdot (\delta - \varepsilon \cdot x) \end{cases}$$

Блок-схема решения задачи в системе **Simulink** имеет приведенный ниже вид (рис. 2.6).

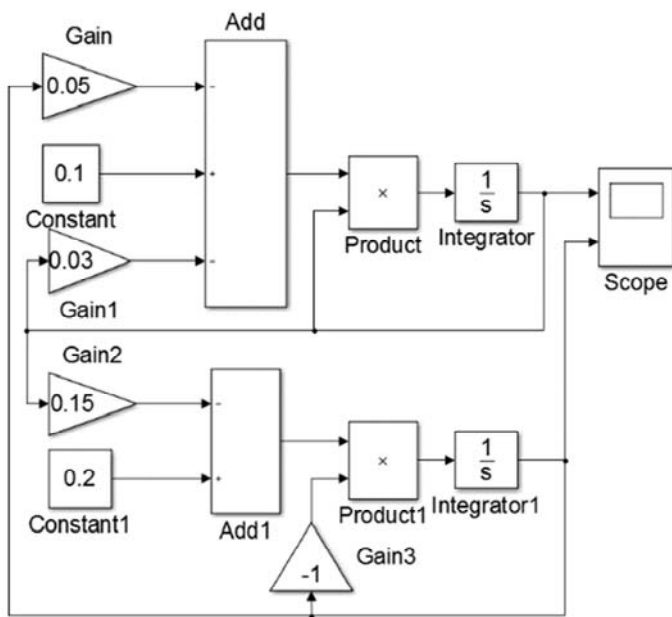


Рис. 2.6. Структурная схема решения системы ОДУ

Для настройки блоков *Integrator(1)* дважды кликаем по ним мышью и в появившемся окне задаем начальные значения: $x_0 = 2$, $y_0 = 0,01$. Аналогичным образом настроим усилители *Gain*, задав в качестве их параметров соответствующие значения коэффициентов в уравнениях. В командной панели *Simulink* в окне *Simulation stop time* задаем время моделирования 200 с.

Для вывода информации об итоговых решениях системы ОДУ используем блок *Scope*, который позволяют получить графики изменения функций уравнений (рис. 2.7).

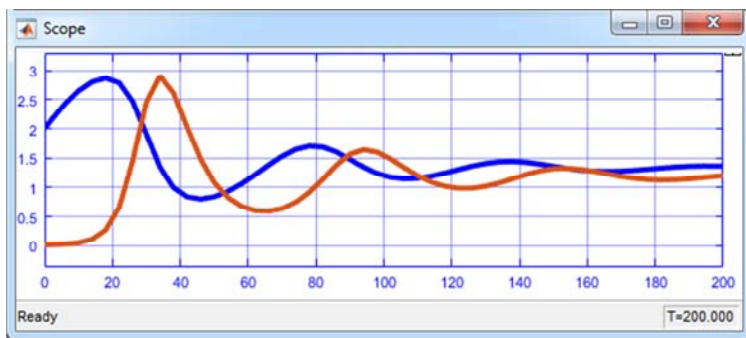


Рис. 2.7. Результаты решения системы ОДУ структурным методом

Для верификации полученных решений проинтегрируем систему ОДУ средствами программирования **MATLAB**, для чего создадим в **MATLAB** m-файл для задания правой части системы ОДУ:

```
function dy=vlm(t,y)
dy=zeros(2,1);
dy(1)=y(1)*(0.1-0.05*y(2)-0.03*y(1));
dy(2)=-y(2)*(0.2-0.15*y(1));
end
```

Систему решаем, используя встроенную функцию *ode45*. Для построения графика решений используем функцию *plot*.

```
[T, Y] = ode45('vlm',[0 200],[2 0.01]);
plot(T, Y(:,1), T, Y(:,2));
```

В качестве параметров функции *ode45* задаем имя m-файла, диапазон изменения независимой переменной и начальные условия. Результаты представлены на рис. 2.8.

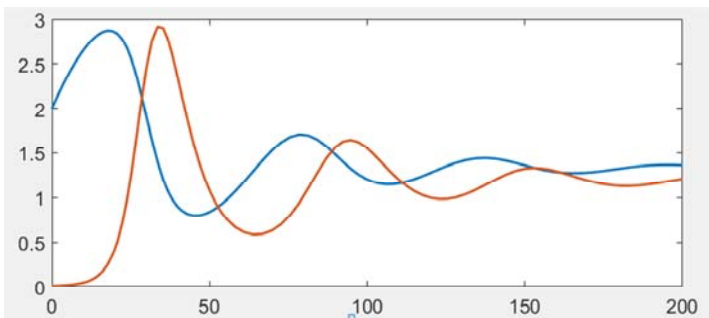


Рис. 2.8. Результаты решения системы ОДУ программированием

Модель вынужденных колебаний одномассовой системы

Рассмотрим решение задачи исследования вертикальных колебаний упруго поддрессоренной массы при безотрывном движении колеса по неровному участку пути (рис. 1.4) с помощью средств структурного моделирования **MATLAB**:

Структурная схема ОДУ приведена на рис. 2.9.

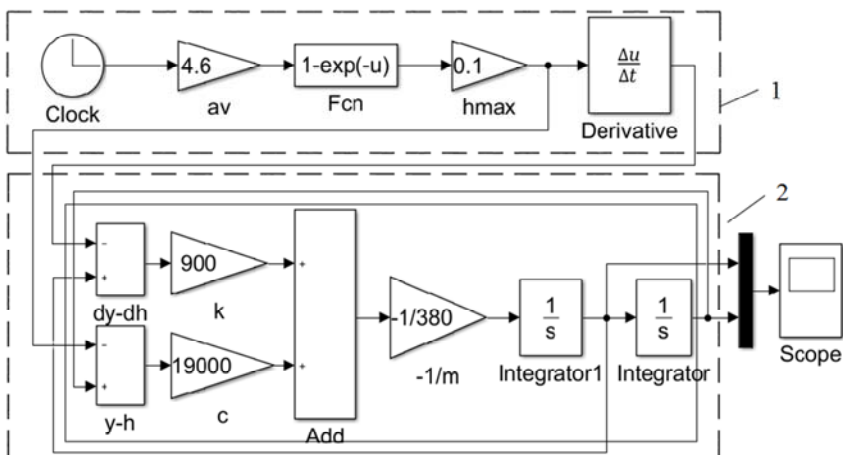


Рис. 2.9. Структурная схема моделирования вынужденных колебаний одномассовой системы:
1 – задание неровности; 2 – задание ОДУ

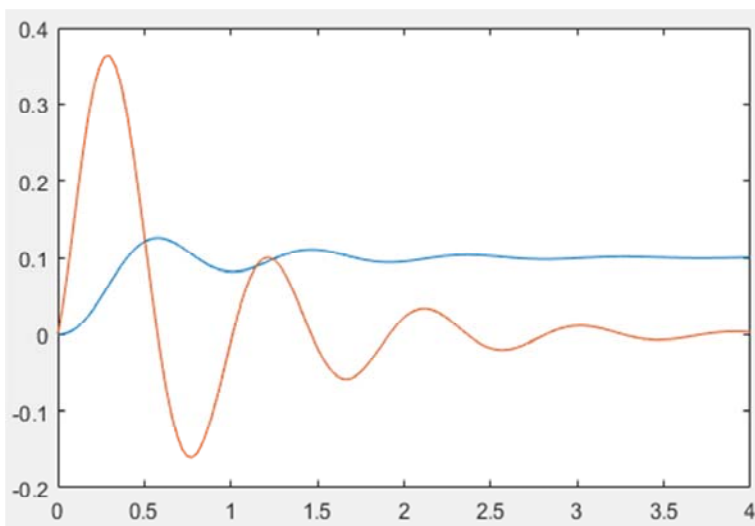


Рис. 2.10. Результаты моделирования одномассовой системы

Задание

1. Составить математическую модель подвески в соответствии с вариантом задания и расчетной схемой (рис. 1.6, табл. 1.2).
2. Построить структурную схему для анализа колебаний двухмассовой системы в **Simulink** при заданном законе изменения возмущающего воздействия (табл. 1.3) в **MATLAB**.
3. По результатам моделирования построить зависимости перемещений и скоростей подрессоренной и непрорессоренной масс при наезде объекта на заданную неровность.
4. Сравнить с решением задачи в **MATLAB** с помощью функции *ode45*.
5. Оформить отчет, содержащий математическую модель, структурную схему, графики зависимостей, согласно п. 3.

Лабораторная работа № 3

МОДЕЛИРОВАНИЕ МЕХАНИЧЕСКИХ СИСТЕМ В СРЕДЕ SIMULINK SIMSCAPE

Цель работы: научиться составлять физические модели на основе расчетных схем технических объектов с помощью блоков базовой библиотеки *Simulink Simscape* пакета **MATLAB**.

Общие сведения

Физическое мультидоменное моделирование основано на использовании библиотеки моделей элементов физических устройств, из которых можно составлять физические принципиальные схемы. В **MATLAB** указанные возможности реализуются в модуле *Simscape*.

Simscape предоставляет блоки для построения систем из различных областей знаний, при помощи которых создаются модели физических компонентов аналогично процессу сборки реальной физической системы. На основе созданной модели *Simscape* автоматически генерирует дифференциальные уравнения, характеризующие поведение системы. В физической модели доступны библиотеки базового **Simulink**, которые присоединяются к физической модели с помощью блоков-конвертеров *PS-Simulink Converter* и *Simulink-PS Converter*.

Каждая схема *Simscape* должна содержать блок конфигурации решателя из библиотеки утилит *Simscape* (рис. 3.1).

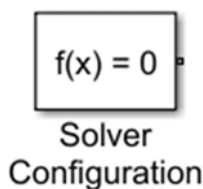


Рис. 3.1. Блок конфигурации решателя

Библиотека компонентов *Simscape* по структуре похожа на библиотеку стандартных элементов **Simulink**.

Компоненты библиотеки сгруппированы по нескольким разделам (рис. 3.2).

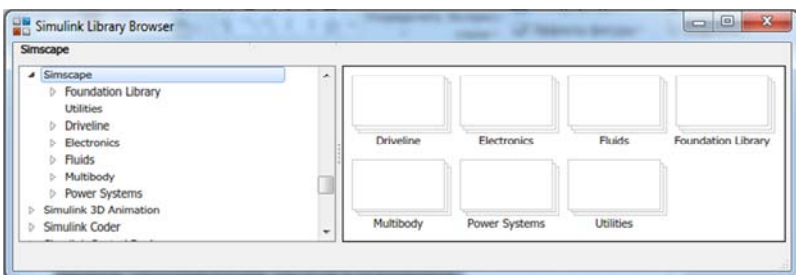


Рис. 3.2. Библиотека компонентов *Simscape*

– **Foundation Library** – основные гидравлические, пневматические, электрические и тепловые элементы.

– **Driveline** – содержит основные элементы силовых передач и трансмиссии.

– **Electronics** – элементы моделирования электронных схем, полупроводников, интегральных схем и электродвигателей.

– **Fluids** – элементы моделирования гидравлических систем, гидrocиллиндров, гидродвигателей, насосов и резервуаров.

– **Multibody** – элементы моделирования пространственных трехмерных механических систем с генерацией 3D-анимации.

– **PowerSystems** – элементы моделирования систем генерации и передачи энергии.

Рассмотрим составление модели колебаний одномассовой системы (рис. 1.4) с использованием блоков библиотеки базовых компонентов **Foundation Library** → **Mechanical** (рис. 3.3).

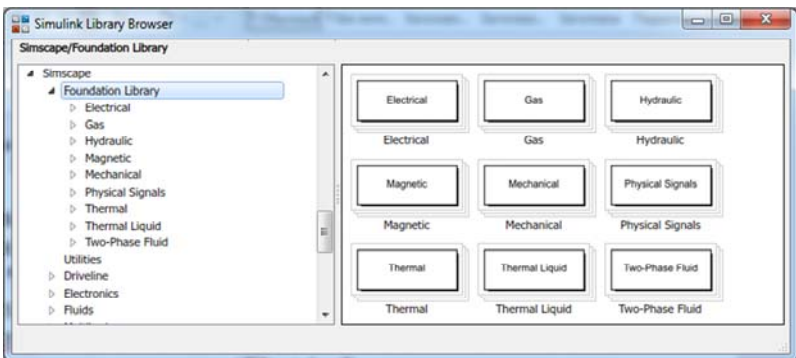


Рис. 3.3. Библиотека компонентов *Foundation library*

Внутри раздела **Foundation library** компоненты сгруппированы по физическому типу моделируемых систем и сигналов в них:

- Электрические элементы (**Electrical**) – моделирование электронных и электрических схем;
- Пневматические элементы (**Gas**) – моделирование основных пневматических эффектов на основе законов идеального газа;
- Гидравлические элементы (**Hydraulic**) – моделирование основных гидравлических эффектов;
- Электромагнитные и магнитные элементы (**Magnetic**) – моделирование магнитных эффектов;
- Механические элементы (**Mechanical**) – моделирование простых механических систем;
- Элементы работы с физическими сигналами (**Physical Signals**) – математические операции над физическими сигналами в моделях;
- Термические элементы (**Thermal**) – моделирование явлений теплопереноса;
- Термические жидкости (**Thermal Liquid**) – моделирование явлений теплопереноса в жидкости;
- Двухфазные жидкости (**Two Phase Fluid**) – моделирование жидкостей с двухфазными состояниями.

Библиотека **Mechanical** включает 5 разделов:

1. **Mechanical Sources** (Механические источники) с компонентами (рис. 3.4).

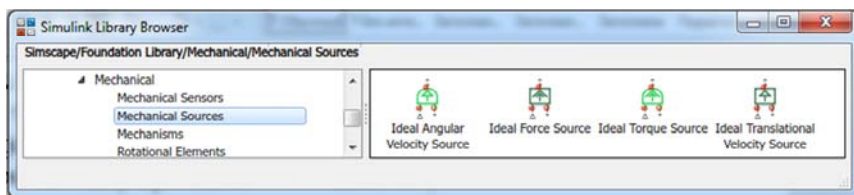


Рис. 3.4. Библиотека компонентов **Mechanical Sources**



Ideal Angular Velocity Source – идеальный источник угловой скорости, который генерирует разность скоростей на своих портах, пропорциональную входному физическому сигналу. Обеспечивает

заданную угловую скорость независимо от крутящего момента, действующего на систему.

Порт **S** – это физический сигнальный порт, через который подается управляющий сигнал. Относительная скорость прямо пропорциональна сигналу на управляющем порту **S**. Угловая скорость равна $\omega = \omega_R - \omega_C$, где ω_R, ω_C – абсолютные угловые скорости в портах **R** и **C** соответственно.

Порт **R** – механический вращательный консервативный порт; **C** – механический вращательный консервативный порт, связанный с исходной опорной точкой (корпусом).



Ideal Force Source – идеальный источник, генерирующий силу, пропорциональную входному физическому сигналу, подаваемому к порту **S**. Сила положительна, если она действует в направлении от **C** до **R**, т.е. относительная скорость $v = v_C - v_R > 0$, где v_R, v_C – абсолютные скорости в портах **R** и **C**.



Ideal Torque Source – идеальный источник, генерирующий крутящий момент, пропорциональный входному сигналу, подаваемому к порту **S**. Крутящий момент положителен, если он действует в направлении от **C** до **R**, т.е. относительная скорость $\omega = \omega_R - \omega_C < 0$, где ω_R, ω_C – абсолютные угловые скорости в портах **R** и **C** соответственно.



Ideal Translational Velocity Source – идеальный источник скорости в механических поступательных системах, который генерирует разность скоростей на своих портах, пропорциональную входному физическому сигналу, подаваемому к порту **S**. Скорость измеряется как $v = v_R - v_C$, где v_R, v_C – абсолютные скорости в портах **R** и **C** соответственно.

2. **Mechanical Sensors** (Механические датчики) (рис. 3.5).

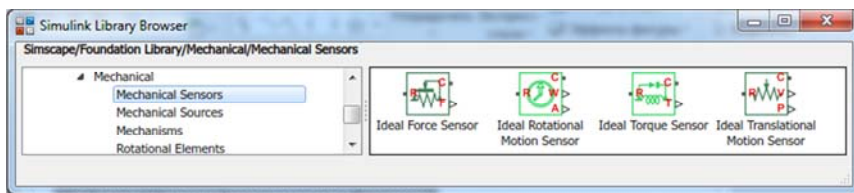


Рис. 3.5. Библиотека компонентов *Mechanical Sensors*

Ideal Force Sensor – датчик силы в механических поступательных системах. **F** – это физический порт, который выводит результат измерения. Порты **R** и **C** представляют собой механические поступательные порты, которые соединяют блок с элементом, где контролируется сила. Положительное направление датчика – от порта **R** к порту **C**.

Ideal Rotational Motion Sensor – датчик вращательного движения, которое преобразует величину, измеренную между двумя механическими узлами вращения, в сигнал, пропорциональный угловой скорости или углу. Порты **R** и **C** представляют собой механические порты, которые соединяют блок с узлами, движение которых контролируется. Порты **W** и **A** являются выходными портами физического сигнала для скорости и углового перемещения соответственно. Положительное направление блока – от порта **R** до порта **C**. Это означает, что скорость измеряется как $\omega = \omega_R - \omega_C$, где ω_R , ω_C – абсолютные угловые скорости в портах **R** и **C** соответственно.

Ideal Torque Sensor – датчик крутящего момента в механических системах вращения. Соединения **R** и **C** являются механическими портами вращения, которые связаны с элементами, между которыми измеряется крутящий момент. **T** – это физический порт, который выводит результат измерения. Положительное направление блока – от порта **R** до порта **C**.

Ideal Translational Motion Sensor – датчик поступательного движения, измеряющий разницу скоростей между двумя механическими поступательными узлами. Порты **R** и **C** соединяют блок с узлами, движение которых контролируется. Соединения **V** и **P** являются выходными портами физического сигнала для скорости и положения соответственно. Положительное направление блока – от порта **R** до порта **C**, т. е. скорость измеряется как $v = v_C - v_R$, где v_R , v_C – абсолютные скорости в портах **R** и **C** соответственно.

3. *Translational Elements* (Элементы поступательного движения) (рис. 3.6).

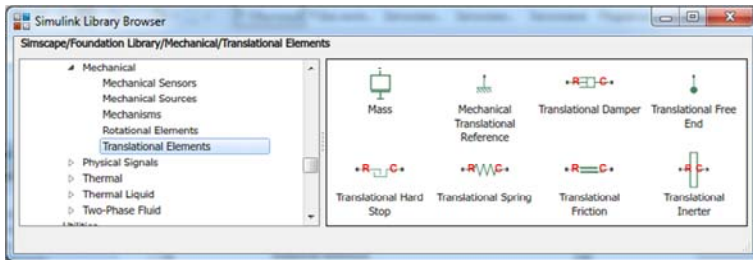


Рис. 3.6. Библиотека компонентов *Translational Elements*

Mass – идеальная механическая поступательная масса, движение которой описывается следующим уравнением:

$$F = m \cdot \frac{dv}{dt},$$

где F – сила;

m – масса;

v – скорость;

t – время.

Блок имеет один механический поступательный консервативный порт, связанный с подключением массы к системе. Положительное направление блока – от его порта до опорной точки, т. е. сила инерции положительна, если масса ускоряется в положительном направлении.

Mechanical Translational Reference – опорная точка или система координат (СК) для всех механических поступательных портов. Все поступательные порты, жестко связанные с неподвижной СК, должны быть подключены к механическому поступательному опорному блоку.

Translational Damper – демпфирующий элемент вязкого трения в механических поступательных системах, описываемый следующим уравнением:

$$F = k \cdot (v_R - v_C),$$

где F – сила, создаваемая демпфером;

k – коэффициент демпфирования (вязкого трения), по умолчанию $k = 100 \text{ Н/(м/с)}$;

v_R, v_C – абсолютные скорости портов **R** (связан со штоком демпфера), **C** (связан с корпусом демпфера).

Translational Free End – представляет собой механический поступательный порт, перемещающийся свободно, без усилия. Блок-схемы физической сети не допускают несвязанных портов. Этот блок служит для завершения механических портов на других блоках, которые являются несвязанными. Также может использоваться для установки начальной поступательной скорости в узле.

Translational Friction – задает трение в контакте между движущимися телами. Сила трения моделируется как функция относительной скорости и принимается как сумма трения, согласно Штрибек-эффекту, кулоновской и вязкой составляющих.

Кривая Штрибека представляет отрицательно наклонные характеристики, имеющие место при низких скоростях. Кулоновское трение создает постоянную силу при любой скорости. Вязкое трение, создает сопротивление движению с силой, прямо пропорциональной относительной скорости.

Положительное направление блока – от порта **R** к порту **C**. Это означает, что если скорость порта **R** больше скорости порта **C**, то блок передает силу от **R** к **C**.

Для блока в качестве параметров задается:

– *Breakaway friction force* (сила отрывного трения), которая является суммой кулоновского и статического трений (должна быть \geq величине силы кулоновского трения, по умолчанию равна 25 Н);

– *Coulomb friction force* (кулоновская сила трения) – значение по умолчанию – 20 Н;

– *Viscous friction coefficient* – коэффициент вязкого трения, значение по умолчанию – 100 Н/(м/с).

– *Linear region velocity threshold* – скорость начала относительного движения фрикционной пары, в этот момент сумма сил трения Штрибека и Кулона является силой отрывного трения. Значение по умолчанию – 10^{-4} м/с ;

– *Transition approximation coefficient* – коэффициент используется для аппроксимации перехода между отрывным и кулоновским трениями. Значение по умолчанию – 10 с/м.

Translational Hard Stop – блок представляет собой двусторонний механический поступательный жесткий упор, ограничивающий движение тела между его границами. Предполагается, что ударное взаимодействие между телом и упорами является упругим с силой, линейно пропорциональной перемещению тела после контакта с упором (рис. 3.7). Для учета рассеивания энергии и неупругих эффектов в качестве параметра блока вводится демпфирование.

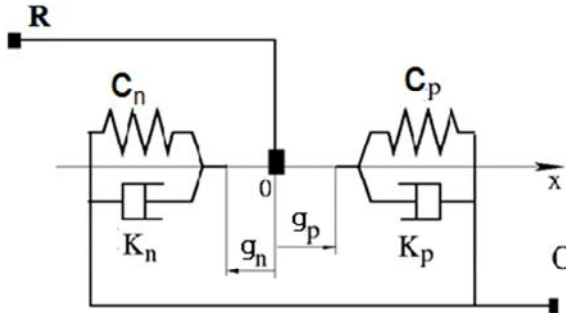


Рис. 3.7. Идеализация механического поступательного жесткого упора

Жесткий упор описывается следующими уравнениями:

$$\begin{cases} c_p \cdot \delta + k_p \cdot (v_R - v_C), & \text{если } \delta \geq g_p; \\ 0, & \text{если } g_n < \delta < g_p; \\ c_n \cdot \delta + k_n \cdot (v_R - v_C), & \text{если } \delta \leq g_n; \end{cases}$$

$$\delta = x_R - x_C; \quad v_R = \frac{dx_R}{dt}; \quad v_C = \frac{dx_C}{dt},$$

где δ – смещение тела;

g_n, g_p – зазоры в положительном и отрицательном направлениях соответственно;

v_R, v_C – абсолютные скорости на портах **R** и **C**;

x_R, x_C – абсолютные перемещения на портах **R** и **C**;

c_n, c_p – коэффициенты жесткости упоров в положительном и отрицательном направлении движения тела соответственно;

k_n, k_p – коэффициенты демпфирования упоров в положительном и отрицательном направлении.

Уравнения приведены относительно локальной системы координат, ось которой направлена от порта **R** к порту **C**. Зазор в отрицательном направлении должен быть указан с отрицательным значением. Блок передает усилие от порта **R** к порту **C**, когда устраняется зазор в положительном.

Translational Inerter – поступательный инерционный блок, представляющий собой устройство, которое создает силу, пропорциональную скорости изменения относительной скорости через порты.

Translational Spring – пружинный блок, представляющий собой идеальную механическую линейную пружину, описываемую следующими уравнениями:

$$F = c \cdot x; \quad x = x_0 + x_R - x_C; \quad v = \frac{dx}{dt},$$

где c – жесткость пружины;

x – деформация пружины;

x_0 – предварительная деформация (при сжатии $x_0 > 0$, при растяжении $x_0 < 0$);

v – относительная скорость;

t – время.

Положительное направление блока – от порта **R** до порта **C** (сила положительна, если она действует в направлении от **R** до **C**).

4. **Rotational Elements** (Элементы вращательного движения) (рис. 3.8).

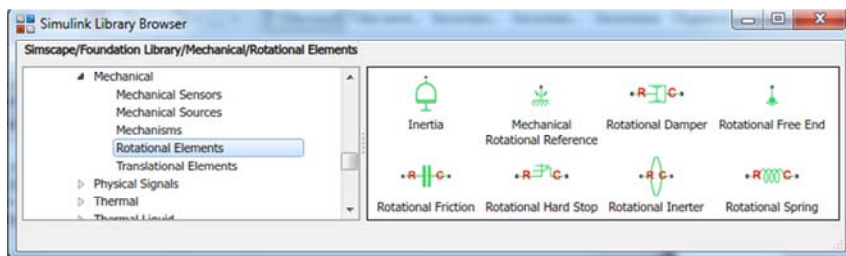


Рис. 3.8. Библиотека компонентов **Rotational Elements**

Функциональное назначение и параметры элементов данной библиотеки аналогичны блокам библиотеки поступательно движущимся

щихся элементов, с той разницей, что вместо кинематических параметров поступательного движения рассматриваются углы поворота и угловая скорость, а вместо силовых параметров – крутящий и фрикционный моменты.

5. *Mechanisms* (Механизмы) (рис. 3.9).



Рис. 3.9. Библиотека компонентов *Mechanisms*

Gear Box – блок, представляющий собой идеальную, непланетарную коробку передач с фиксированным передаточным отношением, которое определяется как отношение угловой скорости входного вала к угловой скорости выходного вала.

Коробка передач описывается следующими уравнениями:

$$\omega_1 = u \cdot \omega_2; \quad M_2 = u \cdot M_1; \quad P_1 = \omega_1 \cdot M_1; \quad P_2 = \omega_2 \cdot M_2,$$

где ω_i , M_i , P_i – угловая скорость, момент и мощность на валах коробки передач (1 – входной вал, порт **S**; 2 – выходной вал, порт **O**); u – передаточное отношение.

Положительное направление блока – от порта **S** к порту **O**.

Lever – представляет собой механический рычаг, варианты исполнения которого приведены на рис. 3.10.

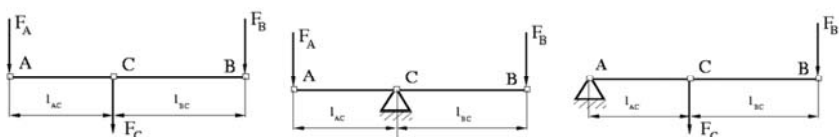


Рис. 3.10. Варианты рычажного механизма

Уравнения, описывающие силовые соотношения, получаются из условия равновесия.

Абсолютные перемещения узлов рычага являются положительными, если они соответствуют глобальной системе координат.

Wheel and Axle – механизм является идеальным преобразователем вращательного движения (порт **A**) в механическое поступательное движение (периферия колеса – порт **P**) (рис. 3.11).

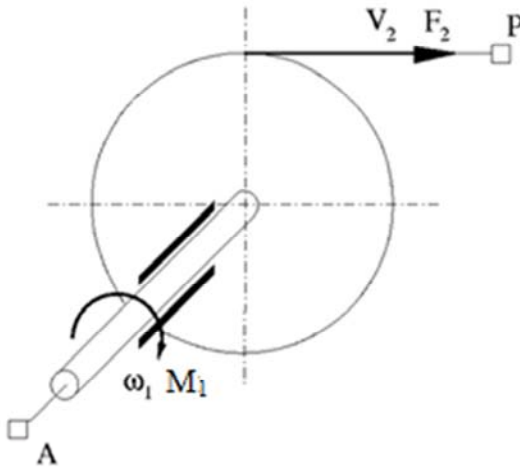


Рис. 3.11. Схема механизма **Wheel and Axle**

Описывается следующими уравнениями:

$$M = r \cdot F \cdot \text{ог},$$

где M – крутящий момент на оси;

r – радиус колеса;

F – сила на периферии колеса;

ог – принимает значение $+1$, если вращение оси в глобально заданном положительном направлении преобразуется в поступательное движение в положительном направлении, и -1 , если положительное вращение приводит к поступательному движению в отрицательном направлении.

Блок может быть использован при моделировании реечных передач, рулевых колес, подъемных устройств, лебедок и так далее.

Положительное направление блока – от порта **A** к порту **P**.

Модель вынужденных колебаний одномассовой системы

Рассмотрим решение задачи исследования вертикальных колебаний упруго поддрессоренного груза при безотрывном движении колеса по неровному участку пути (рис. 1.4) с помощью средств физического моделирования **MATLAB**:

Схема модели приведена на рис. 3.12.

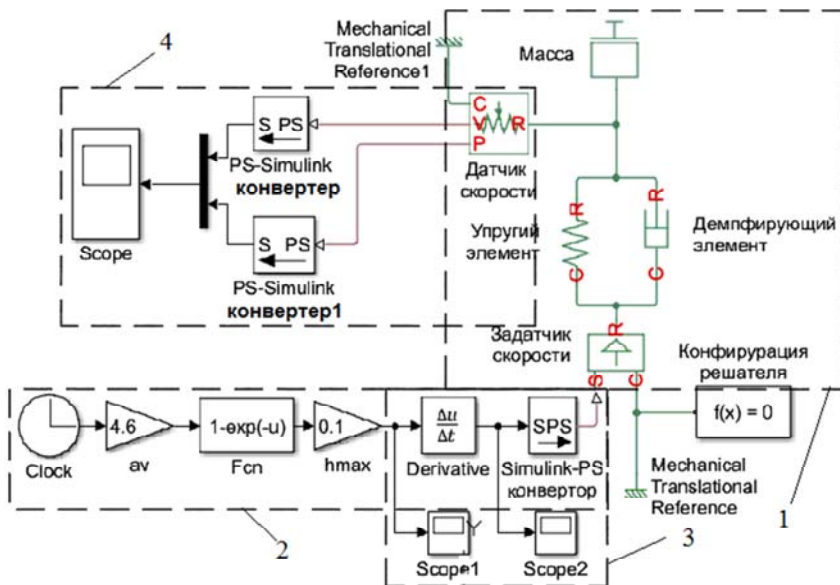


Рис. 3.12. Схема моделирования вынужденных колебаний одномассовой системы с использованием библиотеки *Simscape*:

1 – физическая модель системы; 2 – задание неровности;

3 – вычисление и вывод $h(t)$ и $h'(t)$;

4 – визуализация результатов моделирования

На рис. 3.13 приведены результаты моделирования.

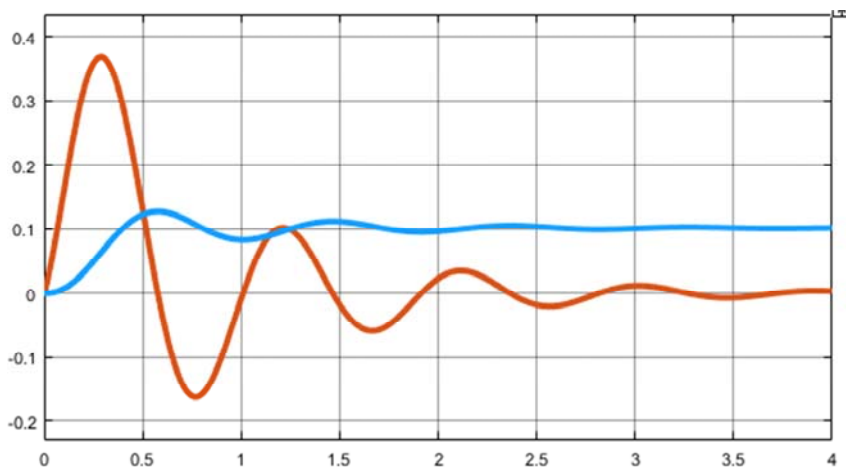


Рис. 3.13. Результаты моделирования одномассовой системы

Задание

1. Построить физическую модель для анализа колебаний двух-массовой системы в **Simulink** в соответствии с вариантом задания и расчетной схемой (рисунок 1.6, таблица 1.2) при заданном законе изменения возмущающего воздействия (таблица 1.3) в **MATLAB**.

2. По результатам моделирования построить зависимости перемещений и скоростей подрессоренной и непрорессоренной масс при наезде объекта на заданную неровность.

3. Сравнить с решением задачи в **MATLAB** с помощью функции **ode45** (лабораторная работа 1) и структурным моделированием (лабораторная работа 2).

4. Оформить отчет, содержащий физическую схему модели, графики зависимостей, согласно п. 3.

Лабораторная работа № 4

МОДЕЛИРОВАНИЕ ГИДРОМЕХАНИЧЕСКИХ СИСТЕМ В СРЕДЕ SIMULINK SIMSCAPE

Цель работы: научиться составлять физические модели на основе расчетных схем технических объектов с помощью библиотек модуля **Simulink Simscape** пакета **MATLAB**.

Общие сведения

Рассмотрим составление модели трогания и разгона машины, оборудованной трансмиссией с гидрорегулируемым сцеплением.

В модели будут совместно использоваться блоки **Simscape Driveline** (Трансмиссия) для моделирования двигателя и сцепления, а также **Simscape Foundation Library** (Базовая библиотека) для моделирования гидравлического привода сцепления. Кроме того, будут применяться базовые блоки **Simulink**.

Рассмотрим основные блоки библиотеки **Simscape Foundation Library hydraulic**, которые будут задействованы при моделировании.



Constant Volume Hydraulic Chamber – моделирует емкость фиксированного объема с жесткими или гибкими стенками и позволяет учесть сжимаемость жидкости с помощью параметров блока.



Hydraulic Pressure Sensor – идеальный датчик гидравлического давления, Порты **A** и **B** используются для подключения датчика к гидравлической линии. Порт **P** – это физический порт сигнала, который выводит значение давления. Перепад давления определяется как $p = p_A - p_B$.



Hydraulic Pressure Source – идеальный источник гидравлической энергии, обеспечивающий заданное давление на выходе независимо от потребления системы. Порты **T** и **P** являются входным и выходным соответственно, а **S** – управляющий порт. Перепад давления на источнике – $p = p_P - p_T$.

Блоки *Simscape Driveline Clutches* и *Engines*, используемые в модели, имеют следующее назначение и параметры.



Disk Friction Clutch – модель дисковой фрикционной муфты, которая обеспечивает передачу крутящего момента между ведущим и ведомым валами. Сцепление начинает включаться, когда управляющее давление на физическом сигнальном порту **P**, превышает пороговое давление включения. Для блокировки сцепления относительная скорость между портами **B** и **F** должна быть меньше допустимой скорости сцепления, а передаваемый крутящий момент должен быть меньше предела статического трения. Заблокированная муфта остается заблокированной до тех пор, пока крутящийся момент, передаваемый через муфту, не превысит предел статического трения. Порты **B** и **F** являются механическими портами угловой скорости. Блок определяет скорость скольжения как разницу $\omega = \omega_F - \omega_B$, где ω – скорость скольжения; ω_F – это угловая скорость ведомого вала; ω_B – угловая скорость основного ведущего вала.

Давление на входе (порт **P**) должно иметь значение ≥ 0 и измеряться в паскалях (Па).

Generic Engine – модель системного уровня двигателей с искровым зажиганием и дизельных двигателей.



Модель двигателя определяется функцией потребляемой мощности двигателя $g(\Omega)$, которая задает максимальную мощность для данной угловой скорости двигателя Ω . Нормализованный входной сигнал дроссельной заслонки T ($0 \leq T \leq 1$) определяет фактическую мощность двигателя. Мощность задается в виде доли максимальной мощности, возможной в стабильном состоянии при фиксированной частоте вращения двигателя. Модулируется фактическая мощность двигателя P , как $P(\Omega, T) = Tg(\Omega)$. Крутящий момент двигателя $\tau = P/\Omega$. Если частота вращения двигателя падает ниже минимально устойчивой, крутящий момент двигателя принимается равным нулю. Если частота вращения двигателя превышает максимальную скорость, выдается сообщение об ошибке.

Типичная функция потребляемой мощности двигателем представлена на рис. 4.1.

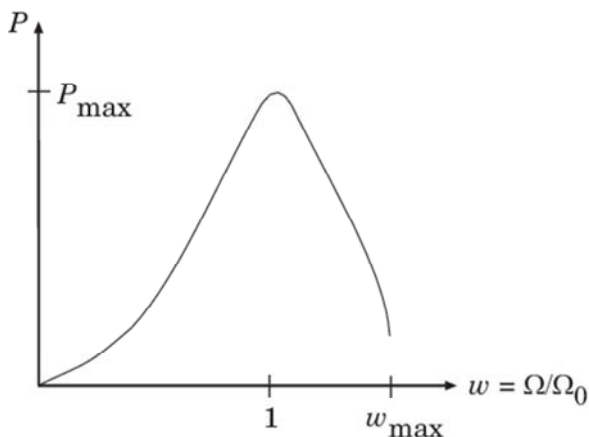


Рис. 4.1. Зависимость мощности двигателя от относительной скорости

Мощность двигателя отлична от нуля, когда угловая скорость находится внутри рабочего диапазона, $\Omega_{\min} \leq \Omega \leq \Omega_{\max}$. Абсолютная максимальная мощность двигателя P_{\max} определяется угловой скоростью Ω_0 так, что $P_{\max} = g(\Omega_0)$.

Порты **F** и **B** связаны с коленчатым валом двигателя и блоком двигателя соответственно. Порты **P** и **FC** являются выходными портами физического сигнала, через которые выводятся мощность двигателя и расход топлива.

Закладка «*Engine Torque*» в окне параметров блока позволяет выбрать методику задания крутящего момента двигателя.

1. *Мощность двигателя согласно полиному.* По умолчанию полином 3-го порядка, соответствующий пиковой мощности. Задаются: тип двигателя, максимальная мощность, угловая скорость при максимальной мощности, максимальная угловая скорость, минимальная устойчивая угловая скорость.

2. *Табулированные данные крутящего момента.* Задаются: вектор скорости (первое и последнее значения которого интерпретируются как скорость холостого хода и максимальная скорость соответственно), вектор крутящего момента, метод интерполяции табулированных данных.

3. *Табулированные данные о мощности.* Задаются: вектор скорости, вектор мощности, метод интерполяции табулированных данных.

Закладка «*Dynamics*» позволяет выбрать способ моделирования инерции коленчатого вала: без учета инерции или с указанием момента инерции и начальной скорости.

Закладка «*Limits*» задает диапазон изменения угловой скорости, в котором крутящий момент двигателя снижается до нуля по мере приближения угловой скорости к скорости сваливания.

Закладка «*Fuel Consumption*» позволяет выбрать модель для определения расхода топлива двигателя.

1. *Constant per revolution*. Задается постоянный объем топлива, потребляемого за один оборот коленчатого вала.

2. *Fuel consumption by speed and torque*. Задается вектор угловой скорости; вектор крутящего момента двигателя (соответствует размеру вектора скорости); матрица расхода топлива (г/с), соответствующая векторам частоты вращения и крутящего момента двигателя (количество строк должно равняться количеству элементов в векторе скорости, а количество столбцов – количеству элементов в векторе крутящего момента); метод интерполяции.

3. *Brake specific fuel consumption by speed and torque (BSFC)* – удельный расход топлива (на 1 кВт·ч) в зависимости от угловой скорости и момента при испытании двигателя на тормозном стенде. Задаваемые параметры аналогичны предыдущему методу, только вместо часового расхода задается удельный расход (г/кВт·ч).

4. *Brake specific fuel consumption by speed and brake mean effective pressure (BSFC)* – удельный расход топлива при испытании двигателя на тормозном стенде как функция угловой скорости и среднего эффективного давления. Задается вектор угловой скорости; вектор среднего эффективного давления в цилиндрах двигателя (**BMEP**) в кПа; матрица **BSFC**, соответствующего векторам частоты вращения и среднего эффективного давления; метод интерполяции.

Закладка «*Speed Control*» – позволяет выбрать модель управления угловой скоростью холостого хода.

1. *No idle speed controller* – регулятор отсутствует и регулирование скоростью осуществляется непосредственно дроссельной заслонкой.

2. *Enable idle speed controller* – включается регулятор, обеспечивающий поддержание угловой скорости в заданных пределах.

В этом случае задаются:

– *Idle speed reference* (эталонная скорость холостого хода) – значение контрольной скорости, ниже которой скорость увеличивается, а выше которой скорость уменьшается;

– *Controller time constant* (постоянная времени регулятора) – задается время запаздывания регулятора (>0);

– *Controller threshold speed* (пороговая угловая скорость регулятора). Параметр используется для сглаживания регулируемого положения дроссельной заслонки когда частота вращения двигателя пересекает эталонную частоту холостого хода (>0).



Signal Builder – позволяет создавать взаимозаменяемые группы кусочно-линейных сигналов и использовать их в модели. Окно **Signal Builder** представлено на рис. 4.2.

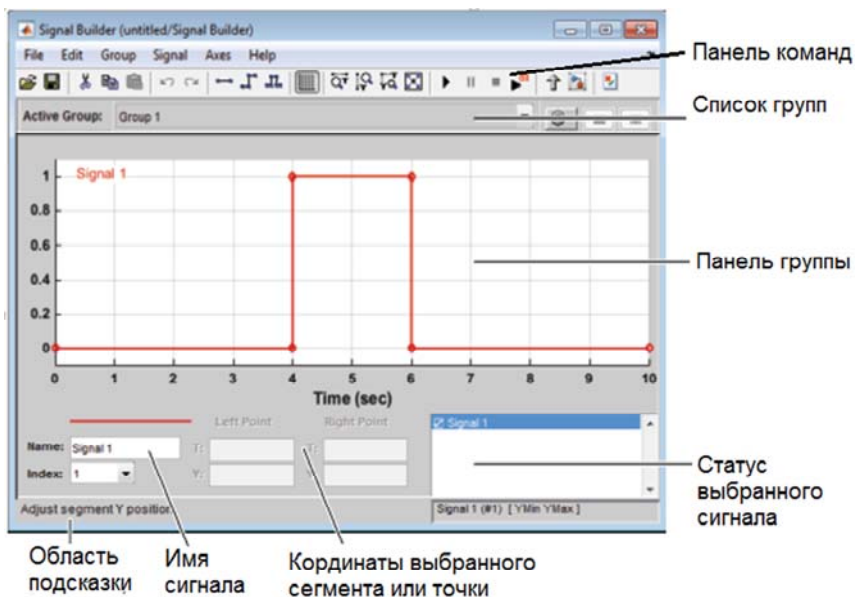


Рис. 4.2. Окно **Signal Builder**

Схема модели трогания машины с фрикционным гидрорегулируемым сцеплением представлена на рис. 4.3.

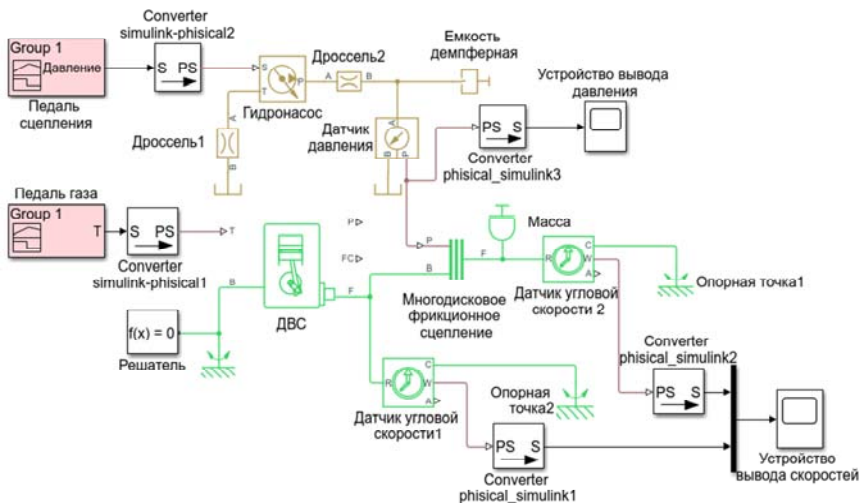


Рис. 4.3. Схема модели трогания машины с фрикционным гидроуправляемым сцеплением

Для преобразования сигналов **Simulink** в физические сигналы и наоборот используются блоки **Simulink-PS** и **PS-Simulink** соответственно. Блоки **Mux** и **Scope** позволяют визуализировать выходные сигналы, полученные в результате моделирования.

На рис. 4.4 представлены сигналы настройки блоков управления давлением и дроссельной заслонкой двигателя.

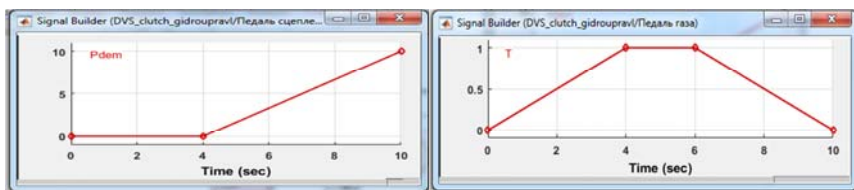


Рис. 4.4. Сигналы блоков управления давлением и дроссельной заслонкой двигателя

На рис. 4.5 представлены настройки параметров конвертеров преобразования сигналов **Simulink** в физические сигналы.

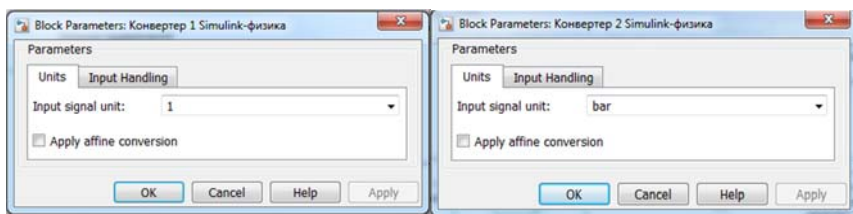


Рис. 4.5. Настройки параметров конвертеров

На рис. 4.6 представлены результаты моделирования давления и угловой скорости на входном и выходном валах сцепления.

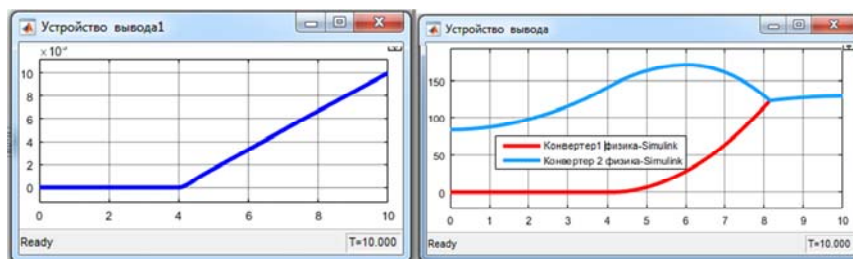


Рис. 4.6. Результаты моделирования давления и угловой скорости на входном и выходном валах сцепления

Задание

1. Построить физическую модель для анализа характеристик процесса трогания и разгона колесной машины в *Simulink Simscape* в соответствии с вариантом задания (табл. 4.1).

2. По результатам моделирования построить зависимости угловых скоростей двигателя и ведущего колеса от времени, перемещения и скорости машины.

3. По графикам определить время буксования сцепления и работу буксования.

4. Оформить отчет, содержащий физическую схему модели, графики зависимостей, согласно п. 2.

Варианты задания принимаются согласно номеру n студента в списке группы: если $n \leq 10$, то номер варианта принимается равным n , а коэффициент запаса сцепления $\beta = 0,25$; если $10 < n \leq 20$, то номер варианта принимается равным $n - 10$, а $\beta = 0,3$.

Таблица 4.1

Характеристики двигателя

№ вар.	$N_{\text{ном}}$, кВт	ω_0 , с ⁻¹	ω_M , с ⁻¹	ω_H , с ⁻¹	ω_{max} , с ⁻¹	$M_{\text{дв.0}}$, Н м	M_{max} , Н м	M_H , Н м	$I_{\text{дв}}$, кг·м ²
1	55	94,2	146,6	230,4	240,8	243	274	239	1,54
2	84	83,7	130,9	183,2	199,0	480	529	461	2,46
3	123	62,8	89,0	130,9	141,4	1000	1029	938	3,8
4	76,5	104,7	146,6	199,0	214,7	422	470	384	2,3
5	128	73,3	162,3	219,9	235,6	612	637	584	3,85
6	23,5	62,8	157,1	230,4	248,7	113	123	102	1,5
7	268	62,8	133,5	193,7	209,4	1410	1529	1382	7,3
8	195	68,1	151,8	219,9	243,5	931	960	885	5,2
9	90	94,2	157,9	251,2	238,1	348	622	453	2,6
10	114	83,7	146,6	219,8	230,2	824	961	816	4,5

Примечание: ω_0 , ω_M , ω_H , ω_{max} – угловая скорость холостого хода, при максимальном моменте, при максимальной мощности, максимальная скорость соответственно; $M_{\text{дв.0}}$, M_{max} , M_H – моменты при ω_0 , ω_M , ω_H соответственно; $I_{\text{дв}}$ – момент инерции двигателя; $N_{\text{ном}}$ – номинальная мощность.

Таблица 4.2

Параметры трансмиссии

№ вар.	D_{max} , мм	p_{max} , МПа	$t_{\text{сц}}$, с	$U_{\text{тр}}$	$r_{\text{к}}$, м	m , т
1	250	0,1	2	33,73	0,85	3,60
2	300	0,15	2,5	32,35	0,70	7,20
3	400	0,2	3	24,10	0,30	12,50
4	300	0,15	2,2	31,74	0,70	6,29
5	400	0,12	3,5	27,70	0,40	7,32
6	200	0,16	2,8	25,10	0,65	2,50
7	450	0,2	3,2	67,55	0,35	42,00
8	320	0,1	3,6	30,10	0,75	6,20
9	300	0,22	4	32,77	0,90	5,30
10	330	0,25	3,8	35,24	1,00	6,50

Примечание: D_{max} – максимальный диаметр сцепления; p_{max} – максимальное давление в приводе; $t_{\text{сц}}$ – темп включения сцепления; $U_{\text{тр}}$ – передаточное отношение трансмиссии; $r_{\text{к}}$ – радиус ведущего колеса; m – масса машины.

Лабораторная работа № 5

МОДЕЛИРОВАНИЕ ПРОСТЕЙШИХ МЕХАНИЧЕСКИХ СИСТЕМ В MATLAB SIMSCAPE MULTIBODY

Цель работы: смоделировать свободное падение тела, имеющего некоторую начальную скорость, в **MATLAB Simscape Multibody**, получить график зависимости координаты и скорости тела от времени.

Создание модели

Математическая модель рассматриваемой задачи имеет вид:

$$z = -g \cdot t^2 / 2,$$

где z – координата тела;

g – ускорение свободного падения;

t – время.

Модель представлена на рис. 5.1.

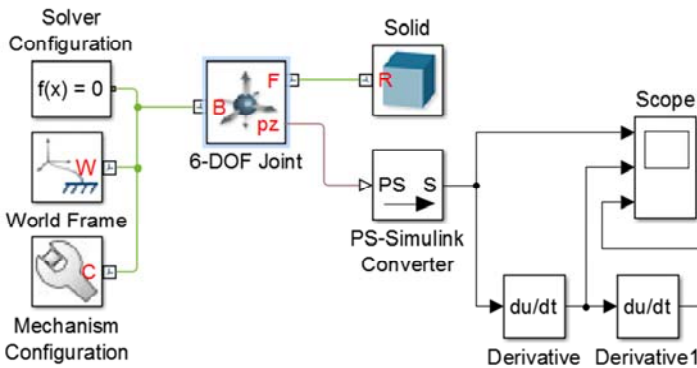


Рис. 5.1. Схема модели свободного падения тела в *Simscape Multibody*

Стандартны и необходимы для любой модели три блока: **Solver Configuration**, **World Frame** и **Mechanism Configuration**. Они зада-

ют тип решателя, МСК (точку отсчета), а в блоке конфигурации механизма указывается направление действия силы тяжести. Для автоматического размещения этих блоков в окно модели можно воспользоваться командой «*smnew*».

Роль тела выполняет блок **Solid**. Для моделирования свободно падающего тела задействован блок **6-DOF Joint**, дающий 6 степеней свободы в системе координат (СК), порт **F (Follower)** которого соединен с СК **R** твердого тела **Solid**, а порт **B (Base)** – с МСК (**World Frame**).

В настройках блока **Mechanism Configuration** необходимо задать направление действия силы тяжести (в данном случае – ось **Z**).

Выходной порт **pz** для замера перемещения тела в направлении оси **Z** появляется при выборе измеряемого параметра в окне свойств блока шарнира (рис. 5.2) **Z Prismatic Primitive** → **Sensing** → **Position**.

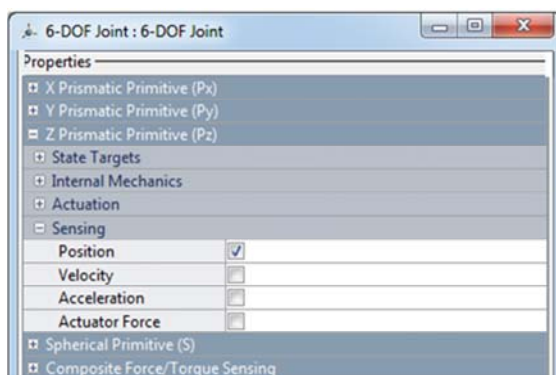


Рис. 5.2. Окно определения свойств соединительного блока **6-DOF Joint**

Блоки **PS-Simulink Converter** и **Scope** используются для вывода графических зависимостей. Для визуализации графика нужно открыть окно блока **Scope** двойным щелчком на самом блоке. На графике показаны зависимости перемещения, скорости и ускорения центра тяжести тела от времени (рис. 5.3).

Точность вычислений зависит от параметров решателя, которые можно настроить, выбрав в меню **Simulation** → **Model Configuration Parameters** → **Solver**, и в соответствующем поле задать шаг вычислений, точность и пр.

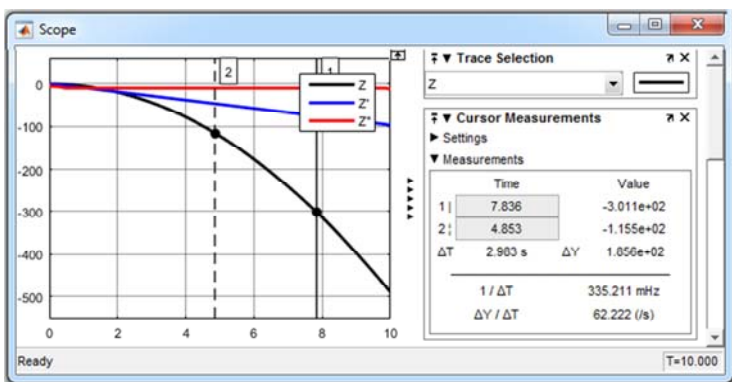


Рис. 5.3. Графическое представление результатов моделирования

Задание

1. Определить траекторию движения брошенного под заданным углом к горизонту тела определенной формы. Движение рассматривать в плоскости XZ и построить соответствующий график (n – номер варианта).

2. По результатам моделирования построить зависимости перемещения, скорости и ускорения тела от времени.

3. По графикам определить значение вычисляемых параметров через 5 с и 10 с движения.

4. Оформить отчет, содержащий физическую схему модели, графики зависимостей, согласно п. 2.

I. Варианты 1–10.

Материал тела: алюминий.

Форма тела: шар.

Диаметр шара: $5 + 0,1 \cdot n$, см.

Угол броска относительно оси X МСК: $n \cdot 16$, град.

Сила броска: $(n + 10) \cdot 10$, Н.

II. Варианты 11–20.

Материал тела: сталь 45.

Форма тела: куб.

Размеры стороны куба: $5 + 0,1 \cdot (n - 11)$, см.

Угол броска относительно оси X МСК: $(n - 11) \cdot 16$, град.

Сила броска: $(n + 1) \cdot 10$, Н.

III. Варианты 21–30.

Материал тела: медь.

Форма тела: диск (основание перпендикулярно направлению движения).

Диаметр диска: $5 + 0,1 \cdot (n - 21)$, см.

Толщина диска: $3 + 0,1 \cdot (n - 21)$, см.

Угол броска относительно оси ХМСК: $(n - 21) \cdot 16$, град.

Сила броска: $(n - 10) \cdot 10$, Н.

Примечание. Сила сопротивления воздуха пропорциональна квадрату скорости движения тела v и площади проекции тела на плоскость, перпендикулярную к направлению движения S :

$$F_{\text{сопр}} = C_x \cdot \left(\frac{\rho \cdot v^2}{2}\right) \cdot S,$$

где ρ – плотность среды;

C_x – коэффициент лобового сопротивления (для шара $C_x = 0,5$; для куба $C_x = 1,28$; для диска, перпендикулярного направлению движения $C_x = 1,18$).

Лабораторная работа № 6

МОДЕЛИРОВАНИЕ КОЛЕБАНИЙ МАТЕМАТИЧЕСКОГО МАЯТНИКА В MATLAB SIMSCAPE MULTIBODY

Цель работы: смоделировать свободные колебания математического маятника и получить графические зависимости его угловых колебаний (угловое перемещение, скорость и ускорение колебаний).

Создание модели

Моделируемая система содержит три тела – стержень, фиксированный центр, соединенные шарниром, и груз, жестко закрепленный на нижнем конце стержня. Для простоты принимается, что стержень имеет форму параллелепипеда.

Математический маятник является примером механизма с одной связью. Для обеспечения соединения стержня с фиксированным центром и грузом необходимо создать две СК на концах стержня с помощью блоков *Rigid Transforms*, а для задания тел будут использованы блоки *Solid*, которые задают их геометрию, инерционные характеристики и цвет. Блок *Revolute Joint* будет использован для обеспечения вращательной степени свободы между стержнем и МСК (фиксированным центром).

Последовательность создания модели приведена ниже.

1. В командной строке **MATLAB** вводим «*smnew*». Откроется библиотека блоков *Simscape Multibody* и шаблон модели с часто используемыми блоками.

2. Добавляем в модель блок *Solid1* и *Rigid Transform1*. Блоки преобразования позволяют создавать новые СК, с которыми можно связывать блоки *Joints* при сборке составных тел.

3. Соединяем блоки, как показано на рис. 6.1. При этом порты базовых СК (**B**) блоков преобразования должны быть обращены к порту **R** блока *Solid1*.

Блок *Solid* представляет собой опорное тело, а *Solid1* – рычаг маятника.

Поскольку каждый блок *Rigid Transform* применяет пространственное преобразование относительно своей опорной СК, измене-

ние подключений портов обычно изменяет пространственные отношения между двумя СК.

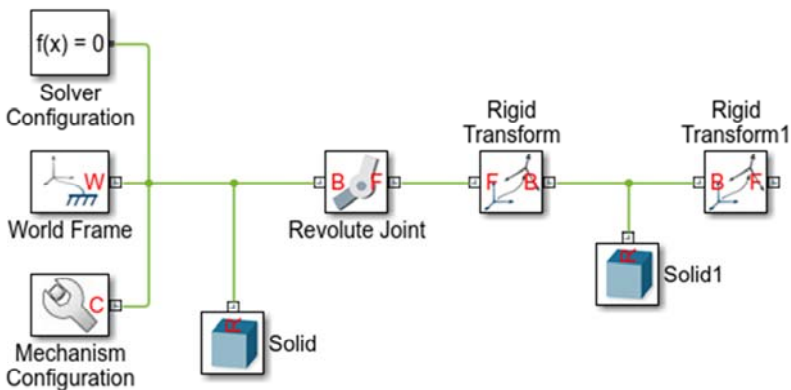


Рис. 6.1. Схема модели однозвенного маятника

4. Для блока **Solid** задаем: *Shape – Cylinder; Radius – 2 см; Length – 1 см; Color – [0.8 0.45 0]*. Так как опорное тело неподвижно, то параметры инерции оставляем по умолчанию.

5. Для блока **Solid1** задаем параметры рычага маятника (рис. 6.2): размеры сечения по осям **Z** и **Y** зададим по 1 см, а длину вдоль оси **X** – 20 см. За материал принимаем алюминий (*Density – 2700 кг/м³*). Цвет зададим составляющими *RGB [0.25 0.4 0.7]*.

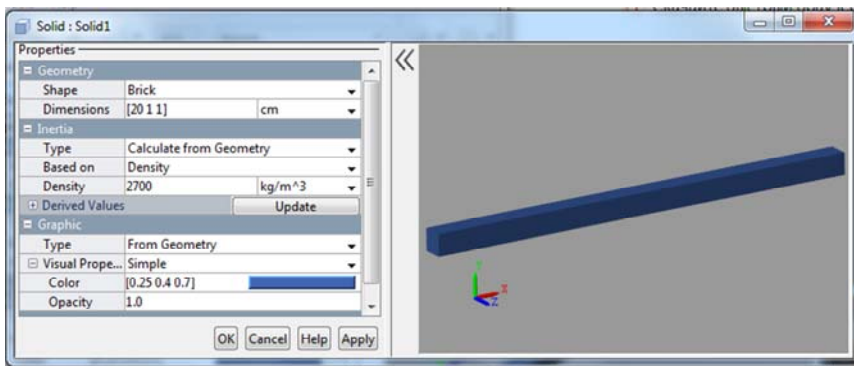


Рис. 6.2. Задание свойств рычага маятника

6. Для блоков **Rigid Transform** и **Rigid Transform1**, зададим параметры смещения между основной СК рычага и СК портов блоков **Rigid Transform**, расположенных в конечных точках рычага (рис. 6.3).

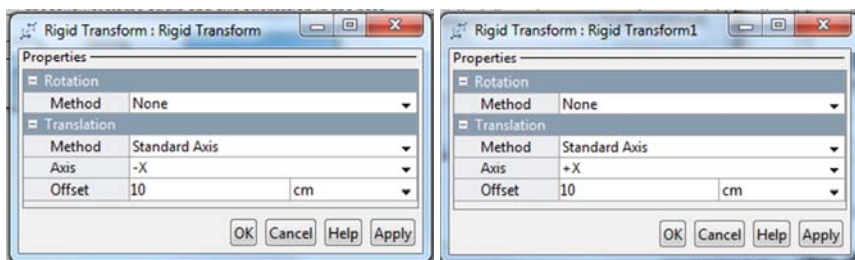


Рис. 6.3. Задание свойств блоков преобразований СК

Задание силы тяжести и стартовой позиции маятника

Блок **Revolute Joint** использует общую ось **Z** МСК и СК как ось вращения. Для колебаний маятника под действием силы тяжести изменяем вектор силы тяжести, чтобы он совпал с осью **Y**. В диалоговом окне блока **Mechanism Configuration** задаем значение *Uniform Gravity* → *Gravity* [0 -9.81 0]. Знак минус задает направление силы тяжести противоположно положительному направлению оси **Y**.

Требуемый угол начального положения маятника можно задать с помощью меню **State Targets** в диалоговом окне блока **Revolute Joint**. Для этого выбираем **State Targets** → **Position** и вводим нужное значение угла. Зададим угол, равный нулю, что соответствует горизонтальной стартовой позиции маятника (рис. 6.4).

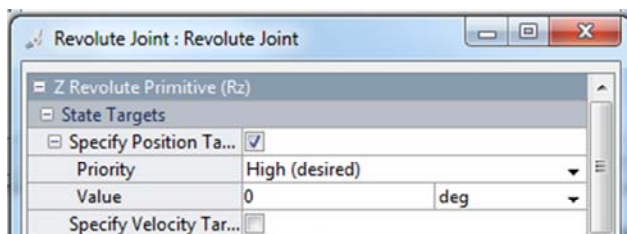


Рис. 6.4. Задание начальной позиции маятника

Задание конфигурации решателя

Открываем окно *Model Configuration Parameters* и во вкладке *Solver* устанавливаем параметр *Solver* на *ode15s (stiff/NDF)*. Этот решатель рекомендуется для физических моделей.

Устанавливаем максимальный шаг расчета *Max step size* на *0.01*. Следует иметь в виду, что маленький шаг увеличивает точность симуляции и обеспечивает более сглаженную анимацию в *Mechanics Explorer*, однако увеличивает время расчета.

Выполнение моделирования

Для визуализации модели во вкладке *Simulation* нажимаем *Update Diagram*. При этом открывается окно *Mechanics Explorer*, где отображается 3D-модель в ее начальной настройке (рис. 6.5).

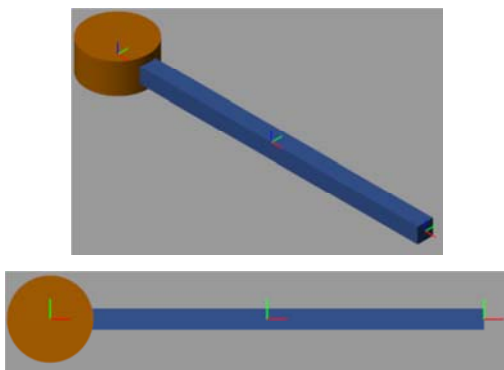



Рис. 6.5. Визуальное представление маятника в *Mechanics Explorer*

В панели инструментов *Mechanics Explorer* устанавливаем параметр *View convention* на *Y up (XY Front)*. Это обеспечивает ориентацию силы тяжести вертикально на экране. Нажимаем кнопку представления *Front View*, чтобы обновить отображение *Mechanics Explorer* и получить вид спереди. Сохраним настройки визуализации путем нажатия на *Save* (кнопка ).

Запускаем симуляцию. В окне *Mechanics Explorer* представляется анимация колебаний маятника.

Задание

1. Построить график свободных колебаний математического маятника для приведенных ниже параметров (n – номер варианта).

I. Варианты 1–10.

Материал маятника: сталь 45.

Размеры маятника: $[20 + n / 2, 1, 1 + n / 10]$, см.

Угол начального положения маятника: $n \cdot 16$, град.

II. Варианты 11–20.

Материал маятника: алюминий.

Размеры маятника: $[20 + (n - 10) / 2, 1 + (n - 10) / 10, 1]$, см.

Угол начального положения маятника: $(n - 10) \cdot 14$, град.

III. Варианты 21–30.

Материал маятника: медь.

Размеры маятника: $[20 + (n - 20) / 2, 1,5, 1 + (n - 20) / 10]$, см.

Угол начального положения маятника: $(n - 20) \cdot 12$, град.

2. Оформить отчет, содержащий физическую схему модели, графики зависимостей, согласно п. 1.

Лабораторная работа № 7

ИССЛЕДОВАНИЕ КОЛЕБАНИЙ МАТЕМАТИЧЕСКОГО МАЯТНИКА В MATLAB SIMSCAPE MULTIBODY

Цель работы: с помощью блоков измерения движения проанализировать влияние на колебания математического маятника внешних сил и крутящих моментов, построить графические зависимости угловых колебаний маятника.

Основные положения

Отправной точкой является модель математического маятника, полученная в лабораторной работе № 6. Добавив силы и крутящие моменты, можно изменить колебания маятника от незатухающих и свободных к затухающим и управляемым. Применяемые при анализе колебаний силы и крутящие моменты приведены на рис. 7.1.

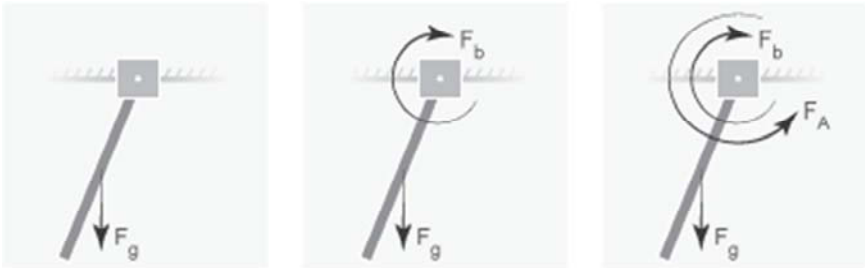


Рис. 7.1. Силы и моменты, действующие на маятник

1. Гравитационная сила (F_g) – глобальная сила, действующая на каждое тело, прямо пропорциональная его массе и действующая в соответствии с вектором ускорения свободного падения g . Этот вектор задается с помощью блока *Mechanism Configuration*.

2. Демпфирование соединения (F_b) – внутренний крутящий момент между маятником и осью качания. Определяется путем задания линейного коэффициента затухания с помощью блока *Revolute Joint*, который соединяет маятник с опорным телом.

3. Активный крутящий момент (F_A) – крутящий момент между маятником и осью качания, который задается непосредственно как физический сигнал *Simscape* с помощью блока *Revolute Joint*, соединяющего маятник с опорным телом.

Определение движения маятника

1. Открываем созданную в ЛР № 6 модель колебаний маятника.
2. В меню *Sensing* диалогового окна блока *Revolute Joint* (рис. 7.2) выбираем следующие переменные: *Position* и *Velocity*.

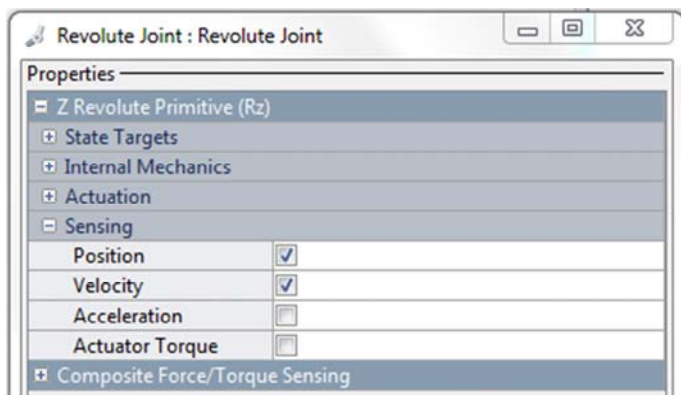


Рис. 7.2. Диалоговое окно блока *Revolute Joint*

В результате на блоке создается два дополнительных порта физического сигнала, маркированных q и w , которые выводят угловое положение и угловую скорость маятника относительно МСК.

3. Добавляем в модель блоки (рис. 7.3), которые позволяют вывести положение и скорость маятника в базовый **MATLAB**:

Simscape → *Utilities* → *PS-Simulink Converter*;

Simulink → *Sinks* → *To Workspace*.

4. Изменяем параметры *Variable name* в диалоговых окнах блока *To Workspace* на q – перемещение и w – скорость маятника, что позволяет вывести эти параметры через *To Workspace* из портов блока *Revolute Joint* в рабочее пространство **MATLAB**.

5. Сохраняем модель для последующего исследования.

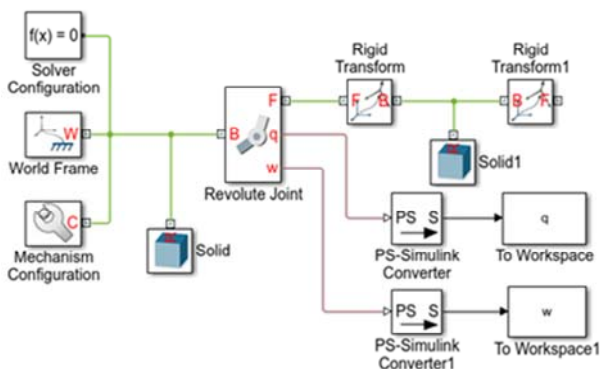


Рис. 7.3. Схема модели для анализа колебаний однозвенного маятника

Анализ свободных незатухающих колебаний маятника

1. Открываем окно *Model Configuration Parameters* и устанавливаем параметры *Solver* на *ode15s (stiff/NDF)* и *Max step size* на *0.01*.

2. Запускаем симуляцию. В окне *Mechanics Explorer* открывается 3D-модель математического маятника.

3. Строим совмещенный график зависимости положения и скорости маятника от времени, например, путем ввода следующего кода в командном окне **MATLAB**:

```
figure; % Открытие нового графического окна
hold on; % Разрешить размещение на графике нескольких кривых
plot(q); % Вывод зависимости угла от времени
plot(w); % Вывод зависимости угловой скорости от времени
```

Полученные зависимости представлены на рис. 7.4.

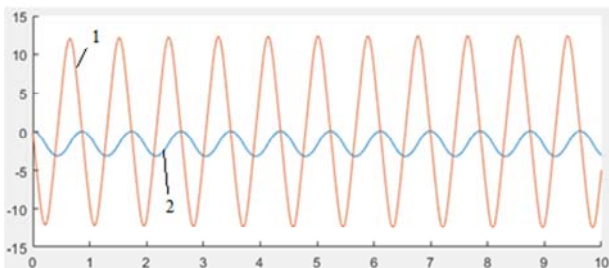


Рис. 7.4. Изменение положения (1) и скорости (2) колебаний маятника

4. Построим зависимость угловой скорости от углового положения путем ввода следующего кода в командном окне **MATLAB**:

```
figure;  
plot(q.data, w.data);
```

Результат представлен на рис. 7.5.

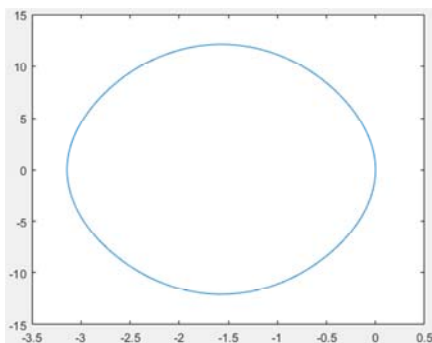


Рис. 7.5. Зависимость угловой скорости от углового положения однозвенного маятника

5. Смоделируем колебания маятника с помощью различных начальных углов. Их задаем в меню **State Targets** → **Position** диалогового окна блока **Revolute Joint**. В качестве начальных углов используем -80 , -40 , 0 , 40 , и 80 градусов.

Результаты моделирования приведены на рис. 7.6.

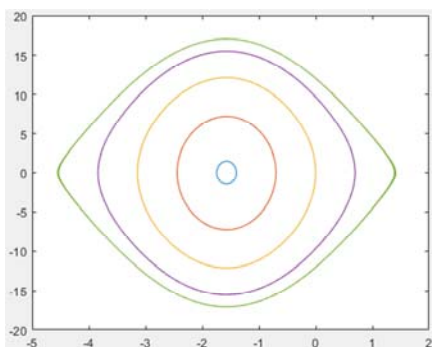


Рис. 7.6. Зависимости угловой скорости от углового положения однозвенного маятника для различных начальных углов

Анализ затухающих колебаний маятника

1. В диалоговом окне блока *Revolute Joint*, закладка *Internal Mechanics* → *Damping* служит для задания коэффициента демпфирования, который, определяет затухание колебаний маятника. Коэффициент затухания задаем $8e-5$ ($N*m)/(deg/s)$.

2. Задаем начальный угол маятника 0 deg (градусов).

3. Запускаем симуляцию и строим зависимости положения и скорости маятника от времени (рис. 7.7):

```
figure;  
hold on;  
plot(q);  
plot(w);
```

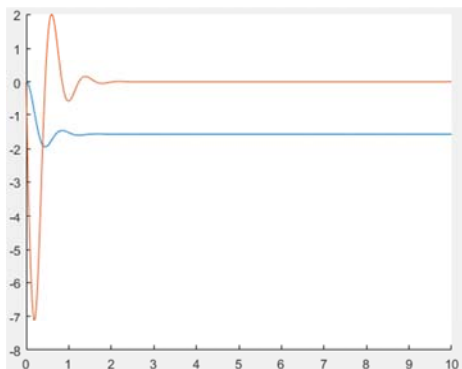


Рис. 7.7. График затухающих колебаний однозвенного маятника

4. Анализ полученного графика показывает, что колебания маятника затухают со временем из-за демпфирования.

5. Построим фазовый график (рис. 7.8), введя в командном окне MATLAB:

```
figure;  
plot(q.data, w.data);
```

Смоделируем колебания при различных начальных углах положения маятника (рис. 7.9), задавая его в меню *State Targets* → *Position* диалогового окна блока *Revolute Joint*: -240 , -180 , -120 , -60 , 0 , и 60 градусов.

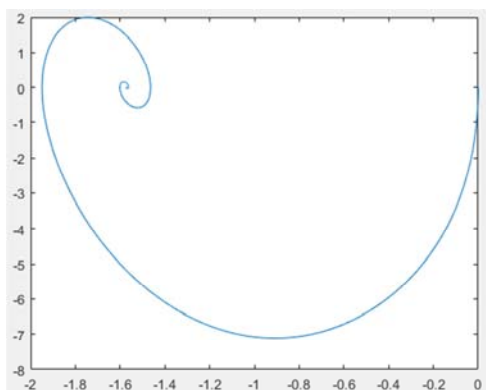


Рис. 7.8. Фазовый график затухающих колебаний однозвенного маятника

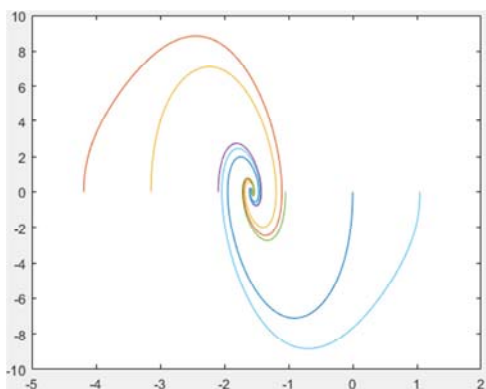


Рис. 7.9. Фазовый график затухающих колебаний однозвенного маятника для различных начальных углов

Анализ колебаний маятника при силовом воздействии с учетом демпфирования

1. Для задания силового воздействия (крутящего момента) служит закладка *Actuation* → *Torque* диалогового окна блока *Revolute Joint*.

В закладке выбираем *Provided by Input* (задается входным сигналом). На блоке появляется входной порт *t* физического сигнала для задания крутящего момента, действующего в шарнире.

2. Зададим крутящий момент в виде синусоидального закона, для чего добавляем в модель (рис. 7.10) следующие блоки:

Simscape → ***Utilities*** → ***Simulink-PS Converter***;
Simulink → ***Sources*** → ***Sine Wave***.

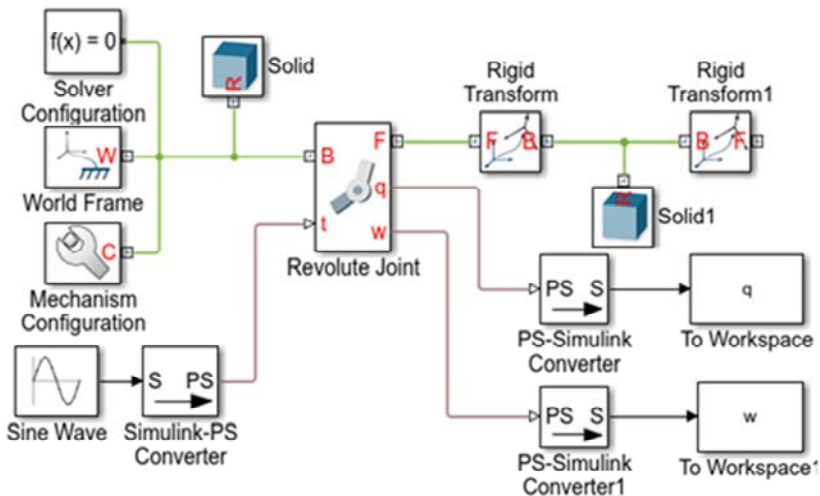


Рис. 7.10. Схема модели для анализа колебаний однозвенного маятника при задании силового воздействия

3. В диалоговом окне блока ***Sine Wave***, закладка ***Amplitude*** вводим амплитуду 0.06 , что соответствует крутящему моменту, изменяющемуся в диапазоне от $-0,06$ Н·м до $0,06$ Н·м. Частота изменения крутящего момента ***Frequency*** = 1 rad/s.

4. В диалоговом окне блока ***Revolute Joint*** задаем ***State Targets*** → ***Position*** → ***Value*** = 0 градусов.

5. Выполняем симуляцию и строим зависимость положения и скорости маятника во времени.

```
figure;
hold on;
plot(q);
plot(w);
```

На рис. 7.11 показан полученный график.

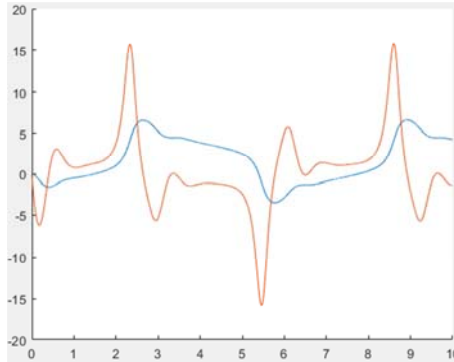


Рис. 7.11. График затухающих колебаний однозвенного маятника при силовом воздействии

Задание

1. Построить график свободных затухающих колебаний и фазовый график для приведенных ниже значений коэффициентов демпфирования для маятника (n – номер варианта). Параметры маятника взять из ЛР № 6.

Варианты 1–10: $(n + 10) / 2 \cdot 10^{-5}$, (Н·м)/(град/с)

Варианты 11–20: $n / 2 \cdot 10^{-5}$, (Н·м)/(град/с)

Варианты 21–30: $(n - 10) / 2 \cdot 10^{-5}$, (Н·м)/(град/с)

2. Построить график колебаний и фазовый график для заданных законов изменения крутящего момента (n – номер варианта). Параметры маятника взять из ЛР № 6, а коэффициент демпфирования – из ЛР №7. Параметры силового воздействия приведены в табл. 7.1.

Таблица 7.1

Параметры для задания крутящего момента

№ варианта	1–10	11–20	21–30
Тип воздействия	синусоидальный (<i>Sine Wave</i>)	Прямоугольный импульс (<i>Pulse Generator</i>)	Пилообразный сигнал (<i>Repeating Sequence</i>)
Амплитуда, Н·м	$0,05 + 0,004 \cdot n$	$0,05 + 0,003 \cdot (n - 10)$	$0,05 + 0,003 \cdot (n - 10)$

Окончание табл. 7.1

№ варианта	1–10	11–20	21–30
Частота, рад/с	$1 + 0,1 \cdot n$	–	–
Период, с	–	$2 + (n - 10) / 10$	$2 + (n - 20) / 10$
Ширина импульса, %	–	$(n + 2) / 2$	–

4. Оформить отчет, содержащий физическую схему модели, графики зависимостей согласно п. 1, 2.

Лабораторная работа № 8

МОДЕЛИРОВАНИЕ МНОГОЗВЕННЫХ РЫЧАЖНЫХ МЕХАНИЧЕСКИХ СИСТЕМ В SIMSCAPE MULTIBODY

Цель работы: создать физическую модель кривошипно-шатунного механизма (КШМ), включающего кривошип, шатун и поршень в *Simscape Multibody* и исследовать закон движения поршня, для чего получить графики зависимостей его перемещения, скорости и ускорения от времени.

Постановка задачи

Расчетная схема КШМ приведена на рис. 8.1, а его массо-инерционные характеристики – в табл. 8.1.

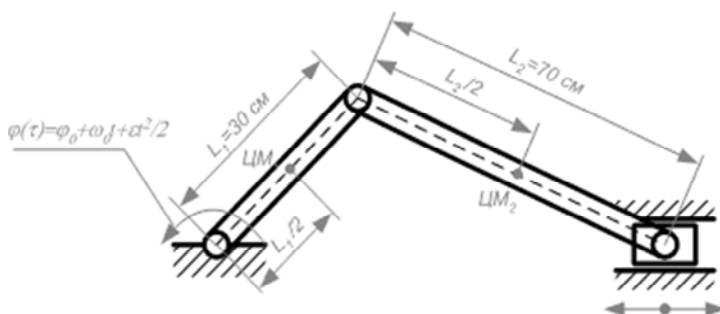


Рис. 8.1. Расчетная схема кривошипно-шатунного механизма

Таблица 8.1

Массо-инерционные характеристики КШМ

Параметр	Кривошип	Шатун
Масса, кг	1	1,5
Тензор инерции, г/см ²	$\begin{bmatrix} 50 & 0 & 0 \\ 0 & 50 & 0 \\ 0 & 0 & 300 \end{bmatrix}$	$\begin{bmatrix} 80 & 0 & 0 \\ 0 & 80 & 0 \\ 0 & 0 & 500 \end{bmatrix}$

Создание модели простого рычага

Для упрощения расчетных физических схем целесообразно создать ряд дополнительных подсистем на базе стандартных блоков **Simulink**, обеспечив возможность задания характеристик, входящих в них элементов.

С учетом этого для построения модели КШМ создадим подсистему, включающую звено в форме параллелепипеда, в середине которого расположена его базовая система координат (БСК). С помощью блоков **Rigid Transform** определим две СК, жестко связанные с БСК тела и определяющие положение узлов звена, с помощью которых оно будет шарнирно соединяться с другими элементами при сборке многозвенных механизмов. Блок **Solid** задает геометрию, инерционные характеристики и отображение данного звена.

Указанные блоки сгруппируем в подсистему, для которой создадим маску, позволяющую в процессе моделирования задавать характеристики звена с помощью окна параметров.

Построение модели

1. Создаем новую **Simulink**-модель, введя в командной строке **MATLAB** команду *smnew*. В результате появится окно модели, созданное на базе шаблона **Simulink Multibody**.

2. Удаляем неиспользуемые в модели блоки.

3. Соединяем оставшиеся блоки, как показано на рис. 8.2, чтобы порты базовой СК (**B**) блоков преобразования были обращены к порту **R** блока **Solid**, т. к. каждый блок **Rigid Transform** применяет преобразование координат относительно своей опорной СК (**B**).

4. В диалоговом окне блока **Solid** задаем его параметры в общем виде (рис. 8.3). Это позволит позже задать переменные **MATLAB**, определенные с помощью блока **Subsystem**, который содержит блоки **Solid** и **Rigid Transform**.

5. В диалоговых окнах **Rigid Transform** и **Rigid Transform1** задаем параметры, определяющие смещение между основной СК тела и СК портов указанных блоков, расположенных в конечных точках рычага (рис. 8.4).

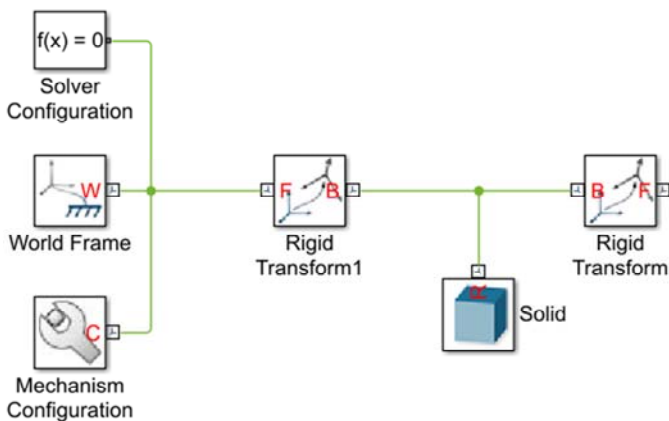


Рис. 8.2. Схема модели рычага

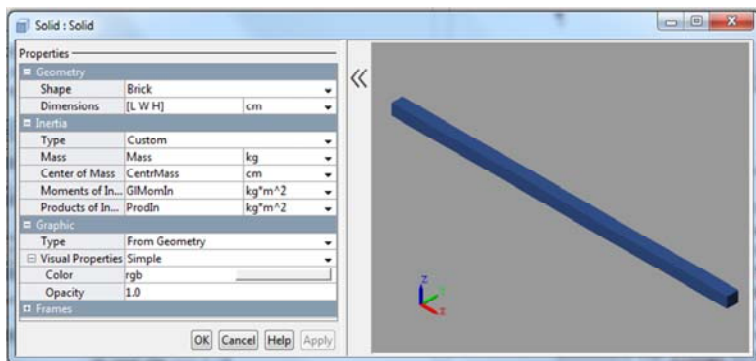


Рис. 8.3. Окно задания свойств блока *Solid*

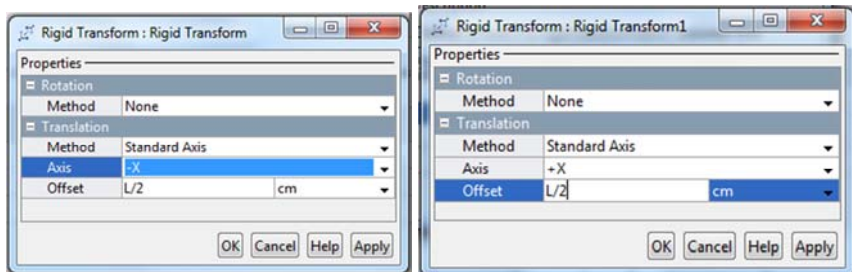


Рис. 8.4. Окна задания свойств блоков *Rigid Transform* и *Rigid Transform1*

Генерация подсистемы

1. Выбираем блок **Solid** и два блока **Rigid Transform**.
2. Щелкаем правой кнопкой по подсвеченной области и выбираем **Create Subsystem from Selection**. Simulink формирует новый блок **Subsystem**, содержащий **Solid** и два блока **Rigid Transform**. Созданному пользователскому блоку присвоим имя **Prostoy Rychag** (рис. 8.5).

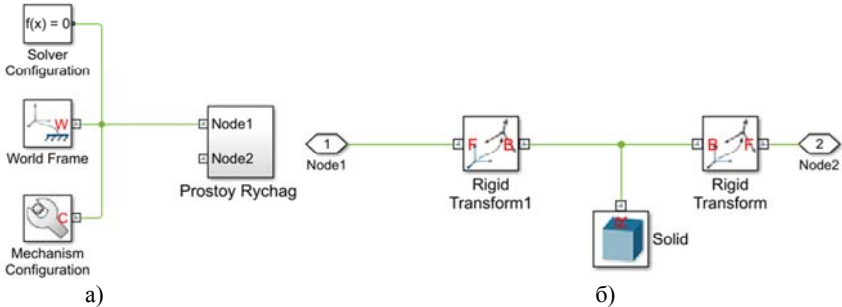


Рис. 8.5. Модель рычага (а) и подсистемы **Prostoy Rychag** (б)

3. Щелкаем правой кнопкой по блоку **Prostoy Rychag** и выбираем **Mask** → **Create Mask**. Открывается редактор (рис. 8.6), который позволяет задать имена переменных, ранее введенных в диалоговые окна блоков **Solid**, **Rigid Transform** и **Rigid Transform1**.

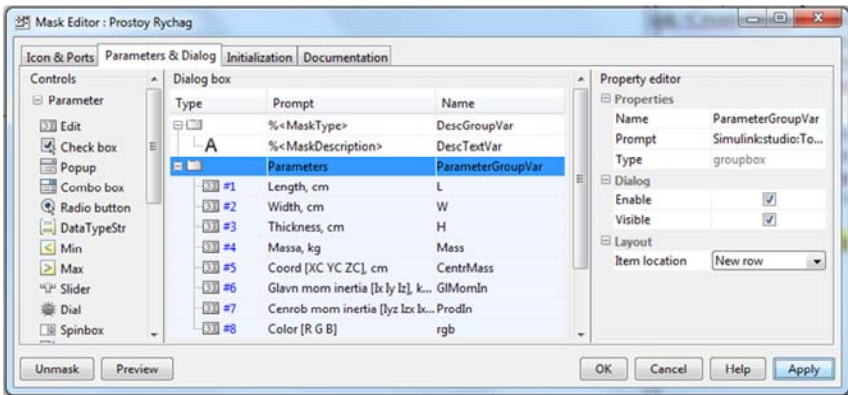



Рис. 8.6. Окно **Mask Editor** для задания параметров подсистемы **Prostoy Rychag**

4. Во вкладке *Parameters & Dialog* окна *Mask Editor* добавляем восемь полей редактирования  *Edit* в колонку *Parameters* панели *Dialog box*. В колонке *Prompt* вводится текст, который отражается для каждого параметра в диалоговом окне блока *Prostoy Rychag*. В полях колонки *Name* задаем идентификаторы параметров тела.

5. Дважды кликаем по блоку *Prostoy Rychag* и в появившемся окне вводим численные значения параметров (рис. 8.7).

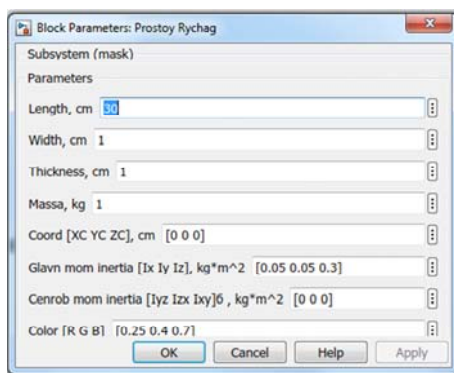



Рис. 8.7. Задание параметров подсистемы *Prostoy Rychag*

Для визуализации модели используется окно *Mechanics Explorer*. В панели инструментов окна *Mechanics Explorer* нажимаем изометрическую кнопку представления , чтобы получить 3D-представление, показанное ниже. Для просмотра СК, определенных в модели, выбираем *View* → *Show Frames* в панели меню *Mechanics Explorer* (рис. 8.8).

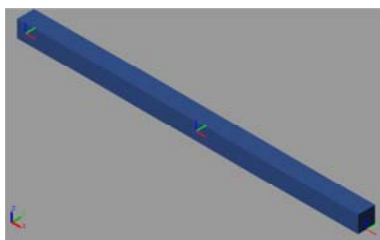


Рис. 8.8. Визуальное представление подсистемы *Prostoy Rychag*

Создание библиотеки пользовательских блоков

Для хранения созданных пользователем подсистем (блоков) целесообразно создать соответствующую библиотеку.

1. Для этого выбрать **File** → **New** → **Library** и в появившееся окно библиотеки перетащить соответствующую подсистему. Аналогично можно поместить в данную библиотеку и другие пользовательские блоки.

2. Сохранить библиотеку, присвоив ей имя, например, **subsystem_library1**. При этом создается файл библиотеки с расширением **.slx**, из которого впоследствии можно переместить в создаваемую модель интересные пользовательские блоки.

Замечание. После внесения изменений в блоки, находящиеся в библиотеке, они модифицируются во всех моделях, куда были помещены ранее.

Создание модели КШМ

1. Создаем новую **Simulink**-модель.

2. Создание модели механизма начинаем с неподвижного звена **Solid**, которое связано с МСК модели. Для блока **Solid** задаем: **Shape** – **Cylinder**; **Radius** – 2 см; **Length** – 1 см; **Color** – [0.8 0.45 0]. Так как опорное тело неподвижно, то параметры инерции оставляем по умолчанию.

3. Кривошип может совершать вращение только вокруг оси **Z**, поэтому в качестве шарнира будем использовать блок **Revolute Joint** из вкладки **Joints**. Добавляем кривошип в модель механизма, используя ранее созданный блок подсистемы **Prostoy Rychag**.

4. Аналогичным образом добавляем в систему шатун **Prostoy Rychag1** и связываем его с кривошипом с помощью блока **Revolute Joint1**. Получаем схему, представленную на рис. 8.9.

5. Настраиваем параметры блоков, представляющих кривошип и шатун, задавая их параметры в диалоговом окне аналогично рис. 8.7: кривошип – **Prostoy Rychag**, $L = 30$ см; шатун – **Prostoy Rychag1**, $L = 70$ см.

6. Задаем для блока **Mechanism Configuration** направление силы гравитации вдоль оси **Y** и запускаем симуляцию. В окне **Mechanics Explorer** наблюдаем свободные колебания полученного двухзвенного механизма (рис. 8.10).

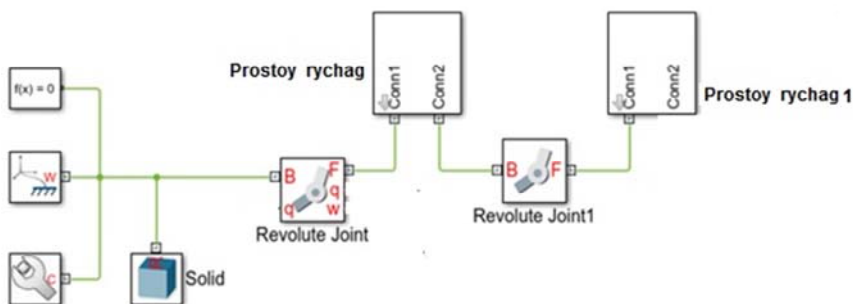


Рис. 8.9. Модель, включающая кривошип и шатун КШМ

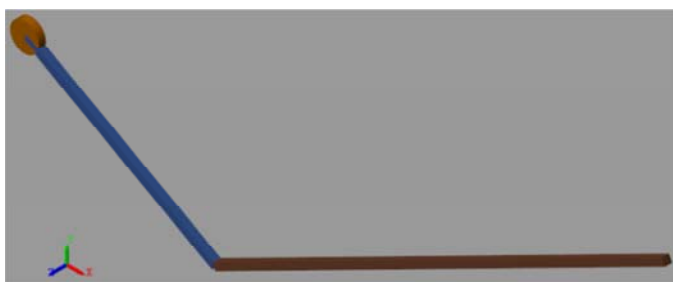


Рис. 8.10. Визуализация построенной модели

7. Добавляем в модель поршень, для чего используем блок **Solid**. Зададим форму поршня в виде цилиндра диаметром 7 см и длиной 15 см (аналогично п. 2). Будем считать, что ось соединения поршня с шатуном проходит через его центр. В качестве материала выберем алюминий, плотность которого 2700 кг/м^3 .

8. Для связи поршня с шатуном воспользуемся блоком **Revolute Joint2**. Однако следует учесть, что при задании цилиндрической формы для блока **Solid1** ось цилиндра располагается вдоль оси **Z**. Поэтому необходимо с помощью блока преобразования **Rigid Transform1** добавить в блок **Solid** новую СК, начало которой совпадает с началом БСК блока, а ось **X** – с осью **Z** БСК (рис. 8.11).

9. Поршень соединяем с МСК призматическим узлом скольжения **Prismatic Joint**, который осуществляет скольжение в направлении оси **Z** (рис. 8.12).

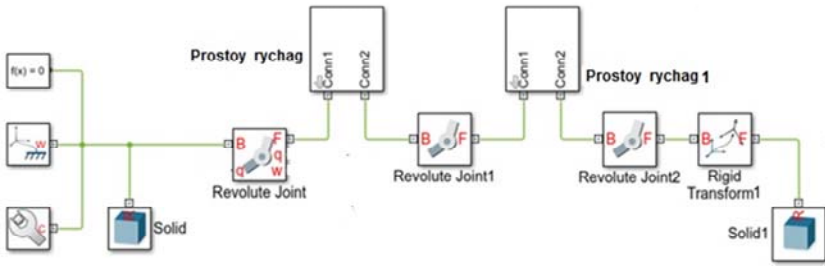


Рис. 8.11. Вид модели после вставки *Solid1*

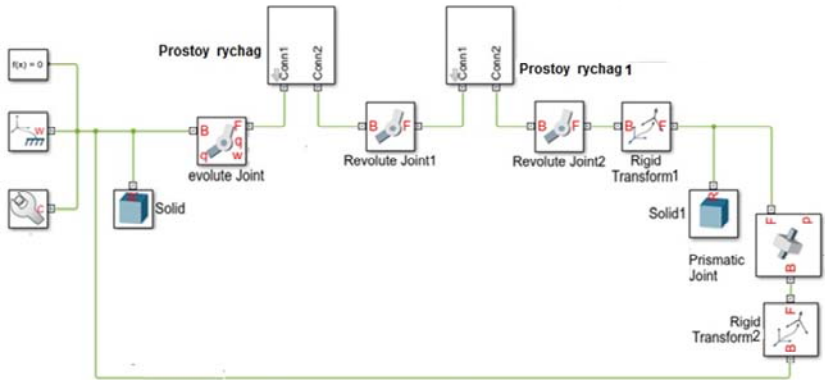


Рис. 8.12 Вид модели после добавления *Prismatic Joints*

Чтобы согласовать системы координат блоков, с обеих сторон *Prismatic Joint* установлены преобразователи СК – *Rigid Transform1* и *Rigid Transform2*, настроенные так, чтобы новая СК была направлена осью *Z* в направлении оси *X* МСК и оси *X* поршня.

Настройка блоков *Rigid Transform* представлена на рис. 8.13.

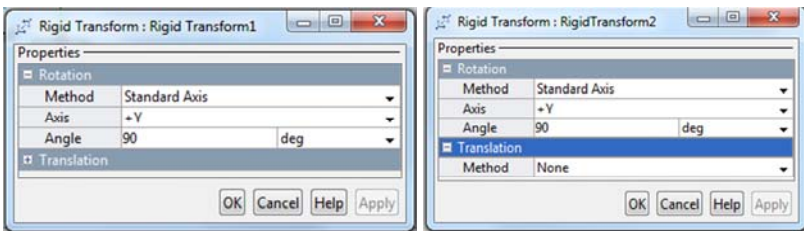


Рис. 8.13. Вид диалоговых окон блоков *Rigid Transform1* и *Rigid Transform2*

10. Для вывода результатов моделирования добавляем в модель блоки *PS-Simulink Converter* и *Scope* (рис. 8.14). Соответственно, в диалоговом окне блока *Revolute Joints* в разделе *Sensing* добавляем порты вывода сигналов *Position* (угол поворота) q и *Velocity* (угловая скорость) w , а для блока *Prismatic Joints* аналогично – порт *Position* (перемещение) p для вывода перемещения поршня.

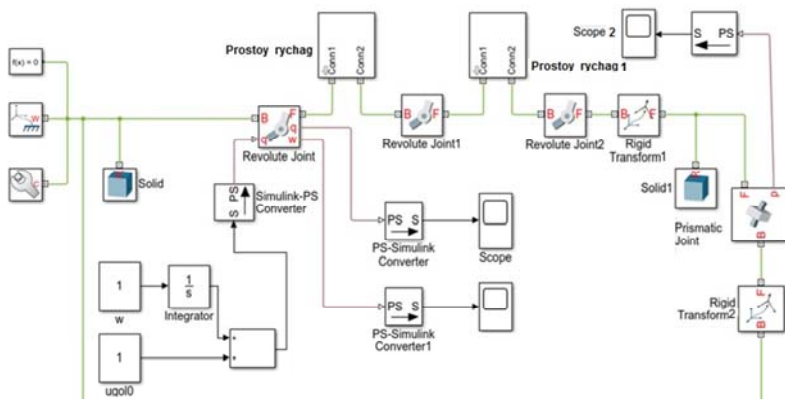


Рис. 8.14. Физическая схема КШМ

Для задания внешнего воздействия на систему воспользуемся блоками *Constant*, *Integrator*, *Add* библиотек *Simulink* и *Simulink-PS Converter*. Для приема сигнала в блоке *Revolute Joints* откроем входной порт, выбрав в его диалоговом окне *Motion* → *Provided by Input*. В качестве начальных условий задается начальное положение кривошипа (блок *ugol0*) и угловая скорость (блок *w*).

Итоговая модель КШМ представлена на рис. 8.14, а ее визуализация с помощью *Mechanics Explorer* – на рис. 8.15.

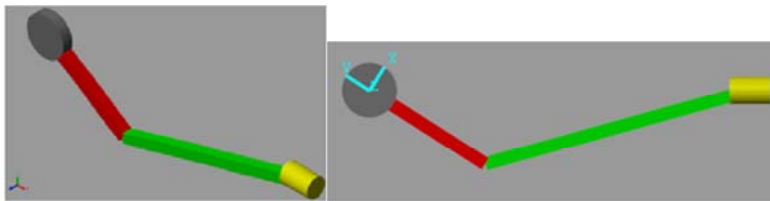


Рис. 8.15. Визуализация модели КШМ

Задание

1. Провести кинематический анализ предложенного механизма.
2. Построить модель механизма, используя блоки *SimScape MultiBody* и *Simulink*.
3. Подключить кинематический привод к начальному звену или к другому элементу (по требованию преподавателя).
4. Исследовать движение механизма (подключить датчики, силы и др.), организовать вывод моделируемых параметров (ускорения, скорости, перемещения).
5. Оформить отчет, содержащий физическую модель механизма, графики зависимостей, согласно п. 4.

ЛИТЕРАТУРА

1. Артоболевский, И. И. Теория механизмов и машин: учебник для вузов / И. И. Артоболевский. – 6-е изд., стер., перепеч. с изд. 1988 г. – М.: Альянс, 2011. – 640 с.
2. Дьяконов, В. П. Simulink 5/6/7. Самоучитель / В. П. Дьяконов.– М.: ДМК-Пресс, 2008. – 784 с.
3. Мусалимов, В. М. Моделирование мехатронных систем в среде MATLAB (Simulink / SimMechanics): учебное пособие для высших учебных заведений / В. М. Мусалимов [и др.]. – СПб.: НИУ ИТМО, 2013. – 114 с.
4. Справочная система MATLAB.
5. Черных, И. В. Simulink. Среда создания инженерных приложений / И. В. Черных. – М.: Диалог-МИФИ, 2004. – 491 с.

СОДЕРЖАНИЕ

Введение.....	3
Требования к выполнению работ.....	4
Лабораторная работа № 1. Инженерный анализ с помощью средств программирования в MATLAB.....	5
Лабораторная работа № 2. Инженерный анализ с помощью средств структурного моделирования в MATLAB Simulink.....	15
Лабораторная работа № 3. Моделирование механических систем в среде Simulink Simscape	23
Лабораторная работа № 4. Моделирование гидромеханических систем в среде Simulink Simscape.....	36
Лабораторная работа № 5. Моделирование простейших механических систем в MATLAB Simscape Multibody.....	44
Лабораторная работа № 6. Моделирование колебаний математического маятника в MATLAB Simscape Multibody	48
Лабораторная работа № 7. Исследование колебаний математического маятника в MATLAB Simscape Multibody	53
Лабораторная работа № 8. Моделирование многозвенных рычажных механических систем в Simscape Multibody	62
Литература	72

Учебное издание

ПОВАРЕХО Александр Сергеевич
ПЛИЩ Владимир Николаевич

**САПР МАШИН. ИНЖЕНЕРНЫЙ
АНАЛИЗ В СРЕДЕ
MATLAB-SIMULINK**

Пособие

для обучающихся по специальностям
1-37 01 03 «Тракторостроение», 1-37 01 04 «Многоцелевые
гусеничные и колесные машины (по направлениям)»,
1-37 01 05 «Электрический и автономный транспорт»

Редактор *Н. А. Костешева*
Компьютерная верстка *Е. А. Бесланской*

Подписано в печать 04.02.2022. Формат 60×84 ¹/₁₆. Бумага офсетная. Ризография.
Усл. печ. л. 4,30. Уч.-изд. л. 3,36. Тираж 100. Заказ 582.

Издатель и полиграфическое исполнение: Белорусский национальный технический университет.
Свидетельство о государственной регистрации издателя, изготовителя, распространителя
печатных изданий № 1/173 от 12.02.2014. Пр. Независимости, 65. 220013, г. Минск.