МИНОБРНАУКИ РОССИИ

Санкт-Петербургский государственный
электротехнический университет «ЛЭТИ»

МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ СИСТЕМ УПРАВЛЕНИЯ

Методические указания
к курсовому проектированию

Санкт-Петербург
Издательство СПбГЭТУ «ЛЭТИ»
2014

УДК 621.372

Математическое моделирование системы управления: методич. указания к курсовому проектированию / сост.: О. Ю. Лукомская, А. Г. Шпекторов. СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2014. 40 с.

Содержат указания по базовым процедурам исследования и моделирования движения морских подвижных объектов на примере катера и корабля на подводных крыльях. Приводятся теоретические основы анализа движения подвижных объектов, а также практические рекомендации по созданию программной модели и исследованию движения морских подвижных объектов в среде MATLAB, по разработке интерфейса программы моделирования кораблей разных типов с использованием элементов MATLAB GUI.

Предназначены для магистрантов, обучающихся по направлению «Автоматизация и управление», а также могут быть полезны инженерно-техническим работникам этой области знаний.

Утверждено
редакционно-издательским советом университета
в качестве методических указаний

Целью курсовой работы является выработка у студентов умения и практических навыков составления и написания программных моделей для исследования в среде MATLAB, создания интерфейсов для вывода результатов моделирования с использованием элементов MATLAB GUI.

Курсовая работа включает в себя следующие основные этапы:

1. Постановка задачи и формирование ее математической модели.
2. Разработка блок-схемы алгоритма решения задачи.
3. Написание и отладка программной модели движения.
4. Представление и анализ результатов работы.

СПИСОК ОБОЗНАЧЕНИЙ

GUI – Graphic User Interface;

ГЭУ – главная энергетическая установка;

КПК – корабль на подводных крыльях;

НВК – надводный водоизмещающий корабль;

ПК – персональный компьютер;

СИ – система измерений;

ТТД – тактико-технические данные.

1. ФОРМИРОВАНИЕ МАТЕМАТИЧЕСКИХ МОДЕЛЕЙ ОБЪЕКТОВ УПРАВЛЕНИЯ

1.1. Математическая модель движения надводного водоизмещающего корабля

В наиболее общем виде поступательное движение любого объекта, в данном случае надводного водоизмещающего корабля, можно описать с помощью второго закона Ньютона:

$$Ma = F - R, \tag{1.1}$$

где M, a – масса корабля и его ускорение.

Правая часть представляет собой алгебраическую (с учетом направления, знака) сумму сил, действующих на корабль (рис. 1.1). Для НВК будем рассматривать следующие силы: F – движущая сила, или сила тяги винта (движителя), Н; R – сила сопротивления движению, Н.

Уравнение (1.1) называют также *балансным уравнением движения* корабля. При балансе или равенстве сил F и R ускорение корабля равно нулю,

корабль движется равномерно (с постоянной скоростью) и прямолинейно. Если $F > R$ – корабль разгоняется и наоборот.

Сила сопротивления направлена в сторону, противоположную силе тяги, и зависит от многих параметров движения, геометрии корпуса НВК и внешних возмущений [1]. Однако в первом приближении ее можно считать пропорциональной квадрату скорости корабля: $R = Av^2$, или для возможности учета направления

$$R = Av|v|, \tag{1.2}$$

где A – коэффициент пропорциональности.

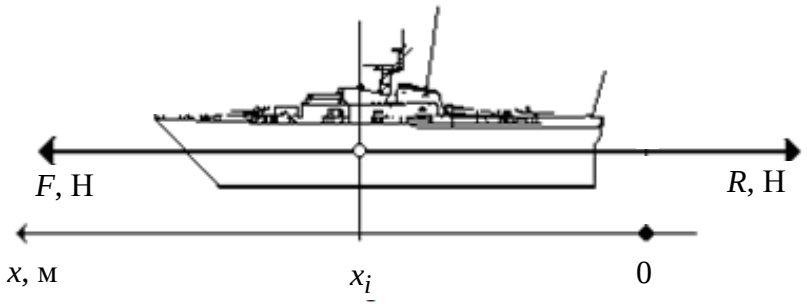

Рис. 1.1. Схема сил, действующих на надводный водоизмещающий корабль при прямолинейном движении

Такой подход является приближенным и годится лишь для построения упрощенных моделей. С учетом формулы (1.2) балансное уравнение движения (1.1) теперь может быть записано следующим образом:

$$Ma = F - Av|v|. \tag{1.3}$$

Входящая в уравнение (1.3) скорость в некоторый момент времени t_i (мгновенная скорость) в направлении координаты пройденного расстояния x (рис. 1.1) в общем случае определяется как $v = dx/dt$.

Коэффициент пропорциональности силы сопротивления движению A может быть определен по граничным условиям, известным для каждого моделируемого объекта, максимальному значению скорости $v_{\max}$ при максимальной силе тяги $F_{\max}$. Поскольку скорость $v_{\max}$ ограничена силой сопротивления среды, т. е. $F_{\max} = R_{\max}$, то можно записать $F_{\max} = Av_{\max}^2$, откуда:

$$A = \frac{F_{\max}}{v_{\max}^2}. \tag{1.4}$$

Ограничения разработанной модели:

1. Предполагается, что корабль движется на постоянном курсе.

2. Не учитывается волнение моря, гидродинамические особенности корпуса, переменное воздействие ветра и т. п.

Сила тяги, определяемая оборотами винтов судна, не может изменяться мгновенно вследствие инерционности вращающих винт двигателей. Полагая зависимость силы тяги от времени линейной, ее изменение можно также описать при помощи дифференциального уравнения с нелинейностью типа «ограничение» («насыщение»):

$$\frac{dF}{dt} = k_F; \quad |F| \le F^*, \tag{1.5}$$

где k_F – коэффициент скорости изменения тяги, Н/с; F^*, $F^* \le F_{\max}$ – заданное значение силы тяги. Коэффициент k_F – постоянный по модулю, но может иметь разные знаки (для моделирования разгона и торможения НВК).

При известном начальном значении силы тяги можно решить дифференциальное уравнение (1.5) и задавать в модели силу тяги как функцию времени.

Для придания модели универсальности целесообразно, по возможности, использовать не абсолютные величины, а их относительные значения. В данном случае предпочтительно принять относительные значения силы тяги в процентах от максимальной:

$$F^* = \frac{P^* F_{\max}}{100}, \tag{1.6}$$

где P^* – относительное значение силы тяги в процентах от максимальной.

Соответственно, имеем модель НВК:

$$\begin{aligned} &\frac{dx}{dt} = v; \\ &\frac{dv}{dt} = \frac{1}{M}\left[F(t) - A|v|v\right]; \\ &A = \frac{F_{\max}}{v_{\max}^2}; \\ &F(t) = \frac{P(t) F_{\max}}{100}. \end{aligned} \tag{1.7}$$

Полученная *математическая модель* (1.7) *позволяет* для любого момента времени определить расстояние, пройденное кораблем (координата

$x(t)$), и скорость движения $v(t)$ по задаваемым значениям силы тяги P^*. Переход к относительным величинам позволяет вводить в качестве исходных данных значения силы тяги (мощности двигателя) в процентах от максимальной, а не в ньютонах, как и принято в современной практике судовождения.

Математическая модель (1.7) может быть использована для изучения динамических характеристик корабля при проектировании, эксплуатации, создании тренажеров и т. п.

1.2. Математическая модель движения корабля на подводных крыльях

Скорость движения кораблей на подводных крыльях в два-три раза выше скорости водоизмещающих кораблей и может достигать 65 узлов (≈120 км/ч). Это предельная скорость для КПК из-за кавитации, возникающей на крыльях и лопастях винта.

Особенностью движения корабля на подводных крыльях, в отличие от водоизмещающего надводного корабля, является наличие трех режимов движения:

– *движение на корпусе (водоизмещающий режим)*. Осуществляется до определенной скорости v_k, после чего переходит в режим глиссирования;

– *глиссирование* – режим движения, когда корабль, набрав скорость v_k, начинает выходить из воды на крылья, достигая при этом скорости $v_{1\max}$, после которой переходит в следующий режим;

– *движение на крыльях (крыльевой режим)*; при достижении скорости $v_{1\max}$ на крыльях возникает и поддерживается соответствующая подъемная сила, сопротивление среды резко уменьшается, а максимально возможная скорость $v_{2\max}$ увеличивается.

Указанная особенность режима движения на крыльях должна быть учтена при разработке математической модели. При моделировании движения водоизмещающего корабля было показано, что сила сопротивления движению пропорциональна квадрату скорости (1.2). Для корабля на подводных крыльях существует два коэффициента пропорциональности:

A_1 – при $v \le v_k$ (водоизмещающий режим);

A_2 – при $v \geq v_k$ (режим глиссирования и движения на крыльях), что обеспечивает возможность достижения максимальных скоростей $v_{1\max}$ и $v_{2\max}$.

Таким образом, математическая модель КПК имеет вид

$$\begin{aligned}
&\frac{dx}{dt} = v;\\
&\frac{dv}{dt} = \frac{1}{M}\left[F(t) - A|v|v\right];\\
&A = \begin{cases} \dfrac{F_{\max}}{v_{1\max}^2}, & v < v_k;\\ \dfrac{F_{\max}}{v_{2\max}^2}, & v \geq v_k; \end{cases}\\
&F(t) = \frac{P(t)F_{\max}}{100}.
\end{aligned} \tag{1.8}$$

1.3. Уточнение математических моделей

Одним из методов построения математической и реализующей ее на компьютере программной моделей является метод *«быстрого прототипа»*. Основное содержание его заключается в том, что вначале строится максимально упрощенная математическая модель с целью получить первые результаты исследования и оценить их правдоподобность в максимально короткие сроки, т. е. проверяется перспективность выбранного направления исследований. В дальнейшем ММ постепенно усложняется и уточняется с целью все большего приближения выходных данных к реальным.

В математической модели (1.8), например, показано, что коэффициент силы сопротивления движению A принимает значения A_2 при $v \geq v_k$, т. е. подчиняется релейному закону (рис. 2.3, *а*). В первом приближении это соответствует действительности. Однако в реальных условиях сила не может измениться скачком, вследствие чего корабль выходит в крыльевой режим не мгновенно, а в течение некоторого времени. Естественно, что сила сопротивления, выражаемая коэффициентом A, уменьшается постепенно, по мере выхода корпуса из воды на крылья. При скорости $v_{1\max}$ значение коэффициента A действительно становится равным A_2.

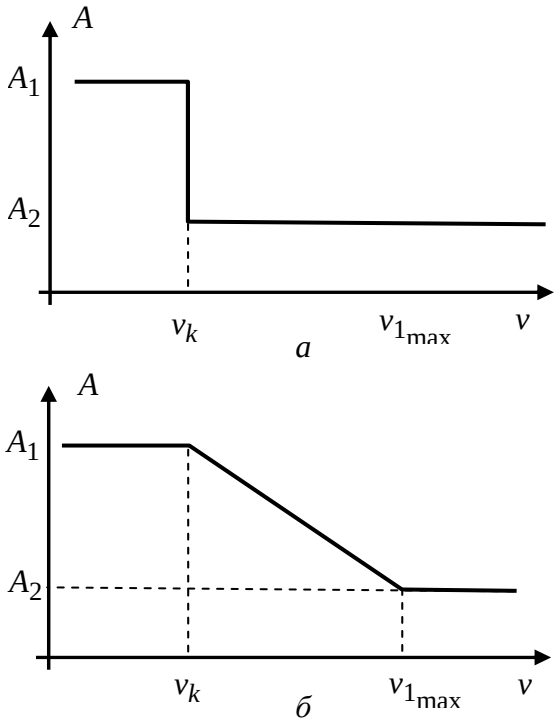

Рис. 1.2. Законы изменения коэффициента *A*: *а* – релейный; *б* – линейный

Очевидно, что принятый релейный закон смены коэффициентов недостаточно точен. Для устранения недостатка можно принять, например линейный закон изменения коэффициента *A* в зависимости от скорости корабля (рис. 1.2, *б*). Это приблизит модель к реальному объекту. Вероятно, этот закон не является строго линейным и для дальнейшего уточнения модели требуются дополнительные аналитические или натурные исследования. Процесс уточнения математической модели носит последовательный итерационный характер.

Для принятого линейного закона коэффициент *A* будет определяться из следующей системы уравнений:

$$
\begin{aligned}
&A = A_1, \quad v \le v_k;\\
&A = A_1 - \frac{(v - v_k)(A_1 - A_2)}{v_{1\max} - v_k}, \quad v_k \le v \le v_{1\max};\\
&A = A_2, \quad v \ge v_{1\max};\\
&A_1 = \frac{F_{\max}}{v_{1\max}^2}, v \le v_k;\\
&A_2 = \frac{F_{\max}}{v_{2\max}^2}, v \ge v_k.
\end{aligned}
\qquad (1.9)
$$

2. ФОРМИРОВАНИЕ ПРОГРАММНЫХ МОДЕЛЕЙ

Компьютерная программа, реализующая полученную математическую модель, обычно носит название *программной модели*. Создание программной модели является следующим этапом моделирования на ПК.

Современные программные средства позволяют моделировать динамические системы в непрерывном времени. Программная среда MATLAB име-

ет достаточно мощный аппарат для численного интегрирования и аналитического решения дифференциальных уравнений. Одной из многих функций численного интегрирования является функция **ode45**, имеющая следующий базовый формат записи:

[TOUT,YOUT] = ode45(ODEFUN,TSPAN,Y0).

Здесь **ODEFUN** – имя функции, вычисляющей правую часть дифференциального уравнения; **TSPAN** – вектор, определяющий начальное и конечное значения временного интервала, на котором осуществляется интегрирование; **Y0** – вектор начальных условий; **TOUT** – вектор моментов времени, для которого получены матрицы частного решения **YOUT**.

Альтернативным способом численного моделирования нелинейных динамических систем является переход к разностным уравнениям. При этом программные модели могут быть сформированы без применения специализированных функций на любом высокоуровневом языке программирования. Уравнения решают последовательно для дискретных моделей времени t_i, t_{i+1}, ... При этом $t_{i+1} = t_i + \Delta t$, где $\Delta t = \text{const}$ – шаг приращения времени (интегрирования). Таким образом, при выборе достаточно малого шага Δt мгновенная скорость может быть определена по конечной разности расстояний:

$$v_i = \frac{\Delta x_i}{\Delta t} = \frac{\left(x_i - x_{i-1}\right)}{\Delta t}, \qquad (2.1)$$

а ускорение в момент времени $t_{i+1} = t_i + \Delta t$ – по формуле

$$a_{i+1} = \left(v_{i+1} - v_i\right)/\Delta t = \left(\Delta x_{i+1}/\Delta t - \Delta x_i/\Delta t\right)/\Delta t = \left(\Delta x_{i+1} - \Delta x_i\right)/\Delta t_2 . \qquad (2.2)$$

Подставив формулы (2.1) и (2.2) в уравнение движения (1.3), получим:

$$M\left(x_{i+1} - x_i - \Delta x_i\right)\big/\Delta t^2 = F_i - A\Delta x_i\left|\Delta x_i\right|\big/\Delta t^2 , \qquad (2.3)$$

откуда легко получить расчетную рекуррентную формулу для вычисления очередного значения координаты пройденного расстояния:

$$x_{i+1} = x_i + \Delta x_i + \left(F_i\Delta t - A\Delta x_i\left|\Delta x_i\right|\right)/M . \qquad (2.4)$$

Из уравнений (2.3) и (2.4) видно, что для вычисления скорости движения НВК необходимо иметь значения двух координат (x_{i-1} и x_i), в то время как для определения ускорения – три, дополнительно x_{i+1}.

В окончательном виде математическая модель движения надводного водоизмещающего корабля (1.7) может быть представлена системой уравнений:

$$x_{i+1} = x_i + \Delta x_i + \frac{\left(P_i \; F_{\max} \; \Delta t^2 \big/ 100 - A \, \Delta x_i \; |\Delta x_i|\right)}{M};$$

$$v_{i+1} = \frac{\Delta x_{i+1}}{\Delta t_i} = \frac{\left(x_{i+1} - x_i\right)}{\Delta t};$$

$$\Delta x_i = x_i - x_{i-1}; \qquad (2.5)$$

$$A = \frac{F_{\max}}{v_{\max}^2},$$

а математическая модель движения КПК (1.8)

$$x_{i+1} = x_i + \Delta x_i + \frac{\left(P_i \; F_{\max} \; \Delta t^2 \big/ 100 - A \, \Delta x_i \; |\Delta x_i|\right)}{M};$$

$$v_{i+1} = \frac{\Delta x_{i+1}}{\Delta t_i} = \frac{\left(x_{i+1} - x_i\right)}{\Delta t};$$

$$\Delta x_i = x_i - x_{i-1}; \qquad (2.6)$$

$$A = \frac{F_{\max}}{v_{1\max}^2}; \quad v_{i+1} \le v_k;$$

$$A = \frac{F_{\max}}{v_{2\max}^2}; \quad v_{i+1} \ge v_k,$$

где P_i – относительное значения силы тяги в процентах от максимальной.

Математическая модель (2.6) также может быть уточнена в соответствии с (1.9).

Достоинством использования функции **ode45** является встроенный выбор шага интегрирования. В расширенном формате записи функции **ode45** можно задавать другие параметры моделирования, в том числе и диапазон изменения шага интегрирования.

2.1. Выбор шага интегрирования

Так как при моделировании на ПК уравнения решают последовательно для определенных (дискретных) значений времени, то для обеспечения требуемой точности достаточно важен выбор шага приращения времени Δt.

Известно, что при выборе неоправданно большого шага численные методы интегрирования могут «разойтись», т. е. привести к неограниченному росту вычисляемых переменных. Слишком малый шаг интегрирования при-

водит к значительному увеличению времени расчетов, создает дополнительную нагрузку на память компьютера [2].

При построении разностной модели шаг целесообразно выбирать по двум критериям.

Во-первых, шаг Δt выбирают как максимальный отрезок времени, в течение которого ускорение практически не меняется или меняется на величину ε, сколь угодно малую и наперед заданную, т. е. $\Delta a \le \varepsilon$. Так как изменение ускорения $a = (F - R)/M$ при прочих равных условиях обратно пропорционально массе объекта, то при моделировании объектов с большой массой можно выбирать и большие значения шага Δt. Например, для катеров водоизмещением несколько тонн целесообразно принять $\Delta t = 1\,\text{с}$, для кораблей и судов водоизмещением в сотни и тысячи тонн Δt принимают до 1 мин.

Во-вторых, выбор шага связан с допустимыми изменениями силы тяги (мощности ГЭУ) за время Δt. При малых скоростях хода сила сопротивления R пренебрежимо мала, а приращение ускорения $\Delta a = a_i - a_{i-1} = (F_i - F_{i-1})/M$. Следовательно, выбор Δt по условию $\Delta a \le \varepsilon$ соответствует предельным изменениям силы тяги на каждом шаге, т. е. $\Delta F \le M\varepsilon$.

Здесь необходимо подчеркнуть, что под ΔF подразумевается предельно возможная скорость изменения силы тяги движителя (мощности ГЭУ), которая зависит и от динамических свойств двигателя. Например, при резком, практически мгновенном переводе рукоятки задания мощности на максимальное значение двигатель (турбина, дизель, реактор) набирает заданную мощность спустя некоторое время, с запаздыванием, определяемым величиной ΔF [3].

2.2. Примеры моделирования движения подвижных объектов

Моделирование катера (малого НВК)

Исходные данные:

1. Масса катера: $M = 1\,\text{т} = 1000\,\text{кг}$.
2. Максимальная сила тяги движителя: $F_{\max} = 1000\,\text{Н}$.
3. Допустимое изменение силы тяги: $\Delta F \le 200\,\text{Н/с}$.
4. Максимальная скорость: $v_{\max} = 72$ км/ч $= 20$ м/с.

Разработка разностной модели движения:

1. Шаг приращения времени (принимается): $\Delta t = 1\,\text{с}$.

2. Относительное (в процентах от максимальной) допустимое изменение силы тяги за время $(\Delta t = 1\,\text{с})$:

$$\Delta P_i \le \Delta t(\Delta F_{\max} / F_{\max}) \cdot 100 \le 1 \cdot (200/1000) \cdot 100 \le 20\,\%.$$

3. Коэффициент пропорциональности силы сопротивления движению:

$$A = \frac{F_{\max}}{v_{\max}^2} = \frac{1000}{20^2} = 2{,}5.$$

4. Координата пройденного расстояния:

$$x_{i+1} = x_i + (x_i - x_{i-1}) + \left(P_i \cdot 1000 \cdot 1^2 / 100 - 2{,}5 \cdot (x_i - x_{i-1}) \cdot |x_i - x_{i-1}|\right) / 1000.$$

5. Текущая скорость:

$$v_{i+1} = \Delta x_{i+1} / \Delta t_i = (x_{i+1} - x_i) / 1 = x_{i+1} - x_i.$$

Программная модель движения катера может быть реализована на любом языке высокого уровня. Входной величиной для программной модели является относительное значение силы тяги P_i, выходными – координата пройденного расстояния x_{i+1} и скорости v_{i+1} в моменты времени с шагом приращения Δt.

Непрерывная модель движения имеет вид

$$\frac{dx}{dt} = v;$$

$$\frac{dv}{dt} = \frac{1}{1000}\left[10P^* - 2{,}5\,|v|\,v\right].$$

Относительная тяга изменяется во времени со скоростью 20 %/с.

В табл. 2.1 приведены результаты моделирования разностной программной модели при максимальной тяге. Здесь и далее в таблицах величины имеют единицы измерения: *t*, с; *P*, %P_{max}; *x*, м; *v*, м/с.

По данным табл. 2.1 строятся графики основных зависимостей движения катера $P = f(t)$, $x = f(t)$, $V = f(t)$, по которым определяются основные динамические параметры движения (рис. 2.1).

Основные динамические параметры движения катера:

- время набора максимальной скорости: 20 м/с – 70 с; при этом катер проходит расстояние 1100 м;

- время торможения: 30 с на расстоянии 213,7 м;
- общее время движения катера составило 60 + 30 = 90 с;
- общее пройденное расстояние: 1100 + 213,7 = 1313,7 м.

Таблица 2.1

Результаты моделирования разностной модели НВК

Режим	T	P	x_{i+1}	v_{i+1}	Режим	T	P	x_{i+1}	v_{i+1}
Разгон	0	0	0	0	Торможение	65	100	1000,2	19,9
	1	20	0,2	0,2		70	100	1100,0	20,0
	2	40	0,8	0,6		71	80	1119,8	19,8
	3	60	2,0	1,2		72	60	1139,2	19,4
	4	80	4,0	2,0		73	40	1158,1	18,9
	5	100	7,0	3,0		74	20	1176,3	18,2
	6	100	10,9	4,0		75	0	1293,6	17,4
	7	100	15,9	4,9		76	–20	1210,0	16,4
	8	100	21,7	5,9		77	–40	1225,4	15,3
	9	100	28,5	6,8		78	–60	1239,5	14,1
	10	100	36,2	7,7		79	–80	1252,4	17,8
	11	100	44,7	8,5		80	–100	1263,8	11,4
	15	100	86,6	11,6		81	–100	1273,9	10,4
	20	100	153,6	14,5		82	–100	1282,8	8,9
	25	100	232,5	16,5		83	–100	1290,4	7,7
	30	100	319,5	17,9		84	–100	1296,9	6,5
	35	100	411,5	18,7		85	–100	1302,3	5,4
	40	100	506,8	19,2		86	–100	1306,7	4,3
	45	100	603,9	19,5		87	–100	1309,9	3,3
	50	100	702,2	19,7		88	–100	1317,2	2,3
	55	100	801,2	19,8		89	–100	1313,4	1,2
	60	100	900,6	19,9		90	–100	1313,7	0,2

Моделирование в среде MATLAB по непрерывной модели движения требует создания вектор-функции расчета правой части дифференциального уравнения.

Моделирование корабля на подводных крыльях

Исходные данные:

1. Масса корабля: $M = 100$ т $= 10^5$ кг.

2. Максимальная сила тяги движителя: $F_{\max} = 5 \cdot 10^4$ Н.

3. Допустимое изменение сила тяги: $\Delta F \le 1000$ Н/с.

4. Скорость начала глиссирования: $v_k = 25$ км/ч $= 6{,}9444$ м/с.

5. Скорость выхода на крылья: $v_{1\max} = 30$ км/ч $= 8{,}3333$ м/с.

6. Максимальная скорость движения на крыльях: $v_{2\max} = 30$ км/ч $= 20$ м/с.

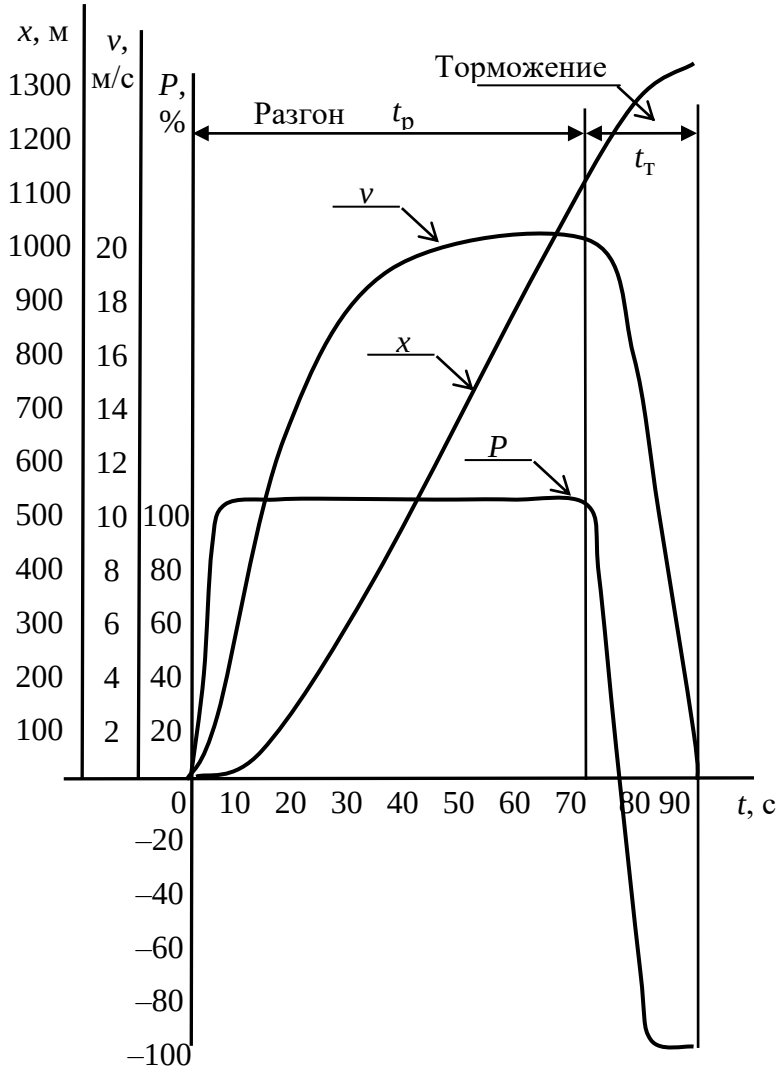

Рис. 2.1. Основные зависимости движения катера: $P = f(t)$, $x = f(t)$, $v = f(t)$.

Разработка математической модели движения:

1. Шаг приращения времени (принимается): $\Delta t = 10$ с.

2. Относительное (в процентах от максимальной) допустимое изменение силы тяги за время $\Delta t = 10$ с:

$$\Delta P_i \le \Delta t \left(\Delta F_{\max} / F_{\max}\right) \cdot 100 \le 10 \cdot \left(1000 / 5 \cdot 10^4\right) \cdot 100 \le 20\,\%.$$

3. Коэффициент пропорциональности силы сопротивления движению:

– при $v < v_k$: $A_1 = \dfrac{F_{\max}}{v_{1\max}^2} = \dfrac{5 \cdot 10^4}{8{,}3333^2} = 720$;

– при $v \ge v_k$: $A_2 = \dfrac{F_{\max}}{v_{2\max}^2} = \dfrac{5 \cdot 10^4}{20^2} = 125$.

4. Координата пройденного расстояния:

1) при $v < v_k$

$$x_{i+1} = x_i + (x_i - x_{i-1}) + \left(P_i \cdot 5 \cdot 10^4 - 720 \cdot (x_i - x_{i-1}) \cdot |x_i - x_{i-1}|\right) \Big/ 10^5;$$

2) при $v \ge v_k$

$$x_{i+1} = x_i + (x_i - x_{i-1}) + \left(P_i \cdot 5 \cdot 10^4 - 125 \cdot (x_i - x_{i-1}) \cdot |x_i - x_{i-1}|\right) \Big/ 10^5.$$

5. Текущая скорость:

$$v_{i+1} = \Delta x_{i+1} / \Delta t_i = (x_{i+1} - x_i) / 10.$$

Непрерывная модель корабля на подводных крыльях имеет вид:

$$\frac{dx}{dt} = v;$$

$$\frac{dv}{dt} = \frac{1}{10^5}\left[500P(t) - A|v|v\right];$$

$$A = \begin{cases} 720, & v < v_k; \\ 125, & v \ge v_k. \end{cases}$$

Поскольку разность силы тяги составляет 1000 Н/с, скорость изменения $P(t)$ составляет 2 %/с.

В табл. 2.2 представлены результаты исследования движения КПК с помощью программной модели, так же, как и для водоизмещающего корабля.

По данным таблицы строится график зависимостей движения КПК $P = f(t)$, $x = f(t)$, $V = f(t)$, по которым определяются основные динамические параметры (рис. 2.2, 2.3).

Основные динамические параметры движения корабля на подводных крыльях:

- время набора максимальной скорости 20 м/с – 120 с из них:
  – в водоизмещающем режиме – 37 с,
  – в режиме глиссирования – 6 с,
  – на крыльях – 77 с;

- до выхода на крылья КПК проходит расстояние 430 м;
- дистанция глиссирования – 90 м;
- время торможения – 73 с, при этом КПК проходит расстояние 870 м.

Таблица 2.2

Результаты моделирования разностной модели КПК

Режим	T	P	x_{i+1}	v_{i+1}	Режим	T	P	x_{i+1}	v_{i+1}
Разгон	0	0	0	0	Торможение				
	10	10	10,0	1,0		110	100	1373	19,8
	20	40	39,3	2,9		120	100	1572	19,9
	30	60	92,4	5,3		130	80	1761	18,9
	40	80	165,2	7,3		140	60	1936	17,5
	50	100	281,4	11,6		150	40	1092	15,6
	60	100	430,7	14,9		160	20	2228	13,6
	70	100	602,0	17,1		170	0	2341	11,3
	80	100	787,0	18,5		180	–20	2428	8,7
	90	100	979,0	19,2		190	–40	2440	1,2
	100	100	1175,0	19,6		200	–60	2421	–1,8

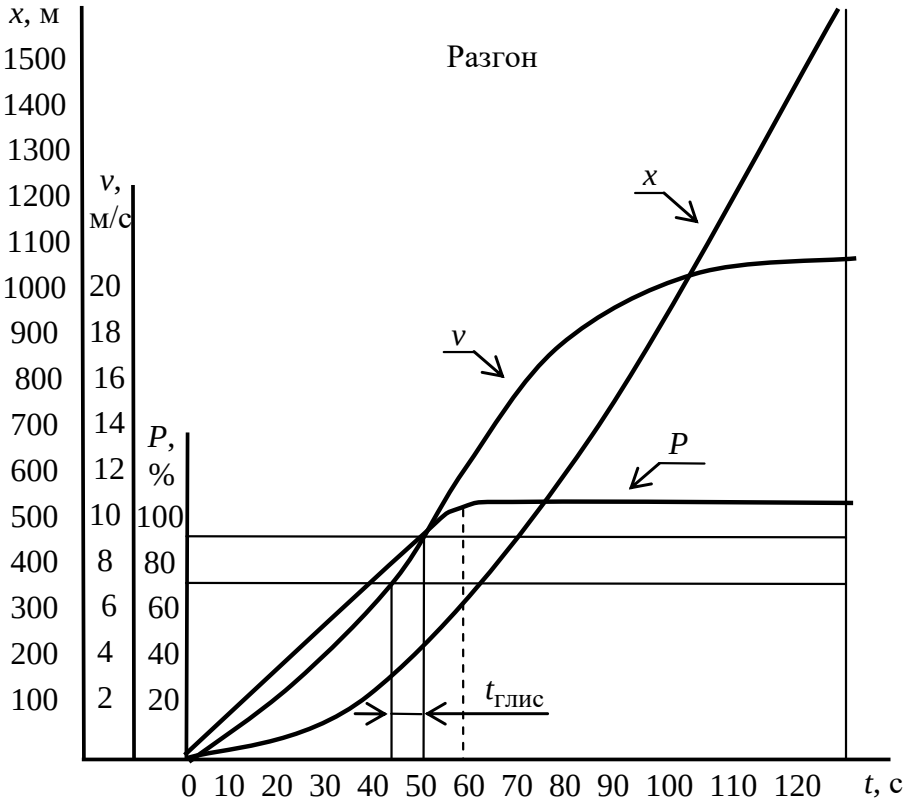

Рис. 2.2. Основные зависимости движения КПК в режиме разгона

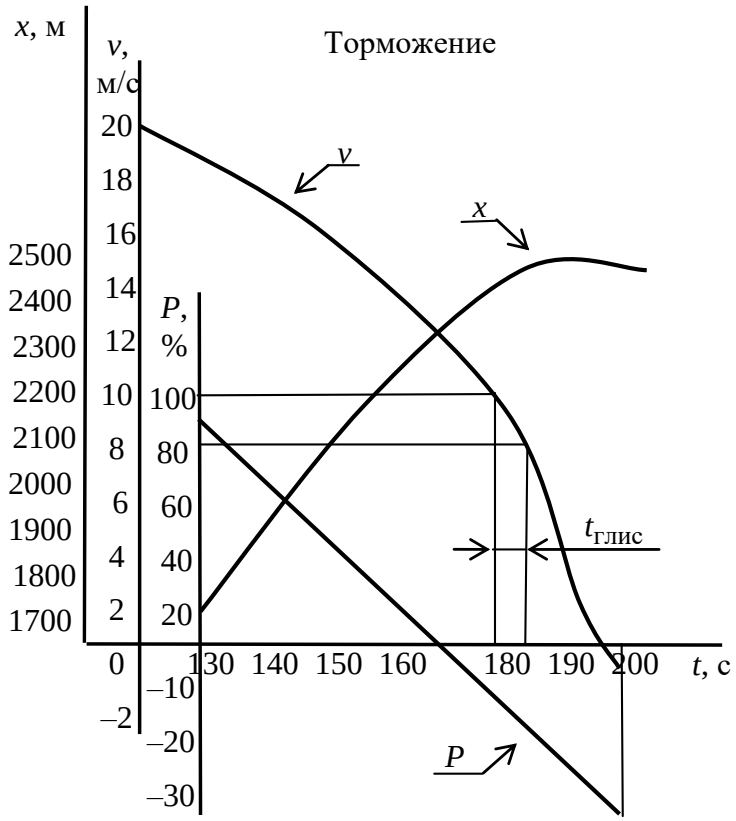

Рис. 2.3. Основные зависимости движения КПК в режиме торможения

В рассмотренных примерах движение в режимах разгона и торможения осуществлялось на максимальной мощности двигателей. Однако с практической точки зрения интересно также моделирование режимов движение на неполной мощности, что следует учесть при разработке программной модели в курсовом проекте.

3. РАЗРАБОТКА ПРИЛОЖЕНИЯ С ГРАФИЧЕСКИМ ИНТЕРФЕЙСОМ

При разработке программных моделей особое внимание следует уделять организации взаимодействия пользователя с программой, удобству ввода данных, простоте представления результатов и надежности. В настоящее время эти задачи удобно решать с использованием элементов графического интерфейса пользователя (GUI). Прежде чем ознакомиться с набором и принципами использования средств GUI в среде MATLAB, необходимо сформулировать основные требования к разрабатываемым приложениям.

1. Набор исходных данных должен обеспечивать гибкость решения поставленной задачи. Важно добиться того, чтобы изменения в процедуре исследования (например, выбор другого шага интегрирования или временного диапа-

зона) не приводили к изменению текста программы или программного кода.

2. Исходные данные и конечные результаты должны быть представлены в удобном для чтения и редактирования виде, с указанием размерности и физических величин измерения.

3. Должны быть учтены ограничения на ввод некорректных исходных данных, который может привести к сбою и ошибкам работы программной модели.

Последнее требование представляет особую сложность, так как область допустимых значений исходных данных не всегда можно найти на стадии получения программной модели. Поэтому процесс разработки приложения носит итерационный характер: недопустимые значения определяются при исследовании программных моделей, а затем осуществляется корректировка приложения.

3.1. Знакомство со средствами графического интерфейса в среде MATLAB

Средства GUI представляют собой набор графических объектов, которые носят общее название *элементов управления*. К ним относятся различные кнопки, списки, поля для ввода текста и др. Разработка приложений с графическим интерфейсом состоит в том, чтобы создать набор таких элементов и запрограммировать события, которые происходят при работе с ними.

В состав среды MATLAB интегрирован пакет GUIDE, позволяющий удобно и быстро создавать окна с элементами управления. Есть возможности для создания приложений GUI вручную [4].

Далее приводится инструкция по работе с элементами управления в среде MATLAB. Следует отметить, что приложения GUI, созданные в разных версиях MATLAB, могут оказаться несовместимы. Инструкция предполагает использование MATLAB 6.5.

Создание, сохранение, открытие окон со средствами GUI

Для создания окна с элементами следует GUI выбрать в главном меню **File -> New -> GUI**. При этом откроется приложение GUIDE. По умолчанию предлагаются 4 шаблона:

1) пустой шаблон (blank);
2) шаблон с элементами управления (GUI with UIControls);
3) шаблон с графикой и меню (GUI with Axes and Menu);
4) диалоговый шаблон (Modal question dialog).

В рассматриваемом примере выбран пустой шаблон, куда могут быть включены любые элементы управления, графики и меню. Для краткости будем называть окна со средствами GUI пользовательскими приложениями, или просто приложениями.

Для сохранения приложения GUI выбрать в меню **File -> Save**. Имя файла соответствует общим требованиям языка MATLAB. При сохранении формируются два файла: filename.fig (окно приложения графического интерфейса) и filename.m – обработчик событий.

Чтобы открыть для редактирования существующее приложение, следует выбрать в меню **File -> New -> GUI**, переключиться на вкладку **Open Existing GUI** и выбрать имя файла.

Примечание: для обычного просмотра приложения можно выбрать в меню главного окна **MATLAB File -> Open -> filename.fig**. Однако его редактирование при этом будет недоступно.

Создание внешнего вида приложения с элементами управления и графики. Общий вид окна пустого шаблона приведен на рис. 3.1.

Элементы управления расположены с левой стороны от общего окна:

– кнопка (Push button);

– кнопка-флаг (Toggle button);

– радиокнопка (Radio button);

– флажок (Checkbox);

– поле для ввода текста (Edit text);

– текст (Static text);

– элемент плавной регулировки, или ползунок (Slider);

– рамка (Frame)

– список (Listbox);

– всплывающее меню (Popup menu);

– графические оси (Axes).

Для создания внешнего вида приложения следует выбрать нажатием мыши требуемый элемент управления, затем щелкнуть им на экране окна в желаемом месте. Стандартными средствами редактирования можно растягивать, копировать или дублировать элементы (для дублирования щелкнуть на элементе правой кнопкой мыши, выбрать Duplicate). Элементы можно размещать поверх друг друга, в этом случае их последовательность меняется выбором переднего или заднего плана (Bring to front, Send to back).

В меню **Tools -> Align Objects** есть разные варианты выравнивания объектов, установки равных интервалов между объектами.

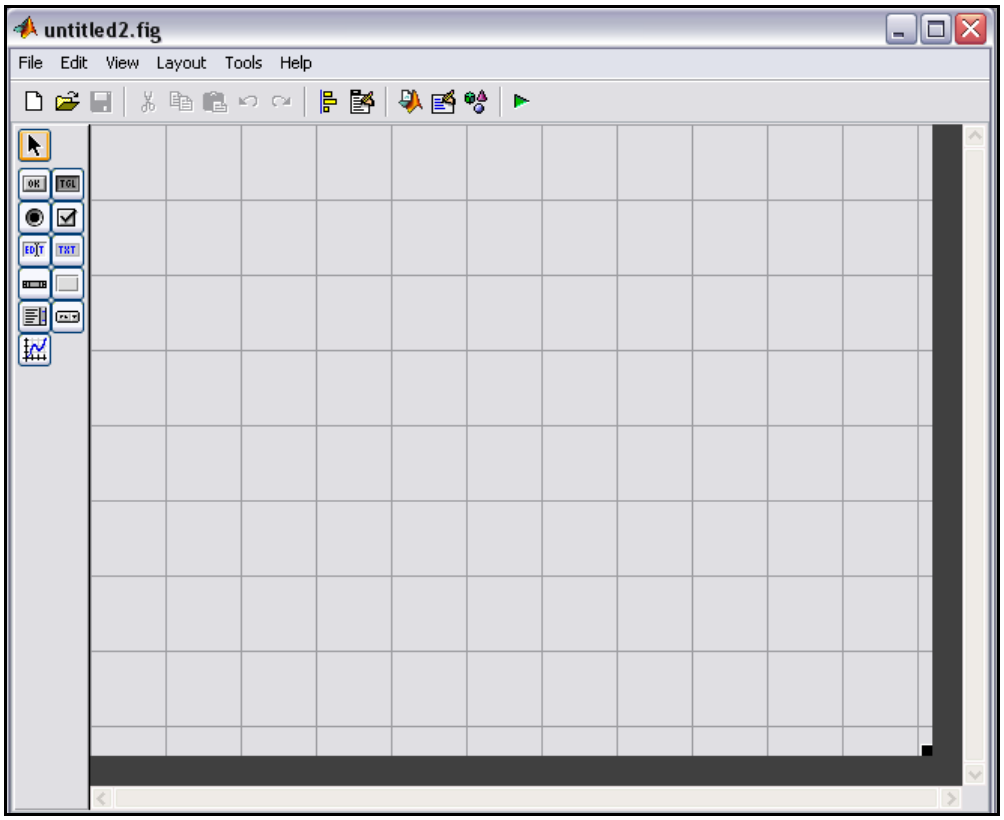

Рис. 3.1. Вид окна пустого шаблона

В порядке выполнения примера рекомендуется создать окно с элементами управления следующего вида (рис. 3.2).

В данном примере рассматривается моделирование динамической системы с передаточной функцией $H(s)=\dfrac{1}{a_1 s^2 + a_2 s + a_3}$, замыкаемой на выбор либо регулятором в виде апериодического звена $H_p(s)=\dfrac{k}{s+b_0}$, либо в виде пропорционального звена $H_p(s)=k$.

Необходимо определить и построить на графике переходную или импульсную характеристику замкнутой динамической системы, по выбору.

В данном окне присутствуют:

– для ввода текста (коэффициентов): 4 поля, и 1 поле для вывода текущего значения *k*;

– для переменного коэффициента: 1 элемент плавной регулировки (пол-

зунок);

– выбор вида регулятора: 2 радиокнопки;

– для запуска процедуры моделирования и завершения работы: 2 обычные кнопки;

– для выбора типа графика: 1 список;

– оси для вывода графика;

– текстовая поясняющая информация (названия коэффициентов, диапазон изменения *k* и т. д.).

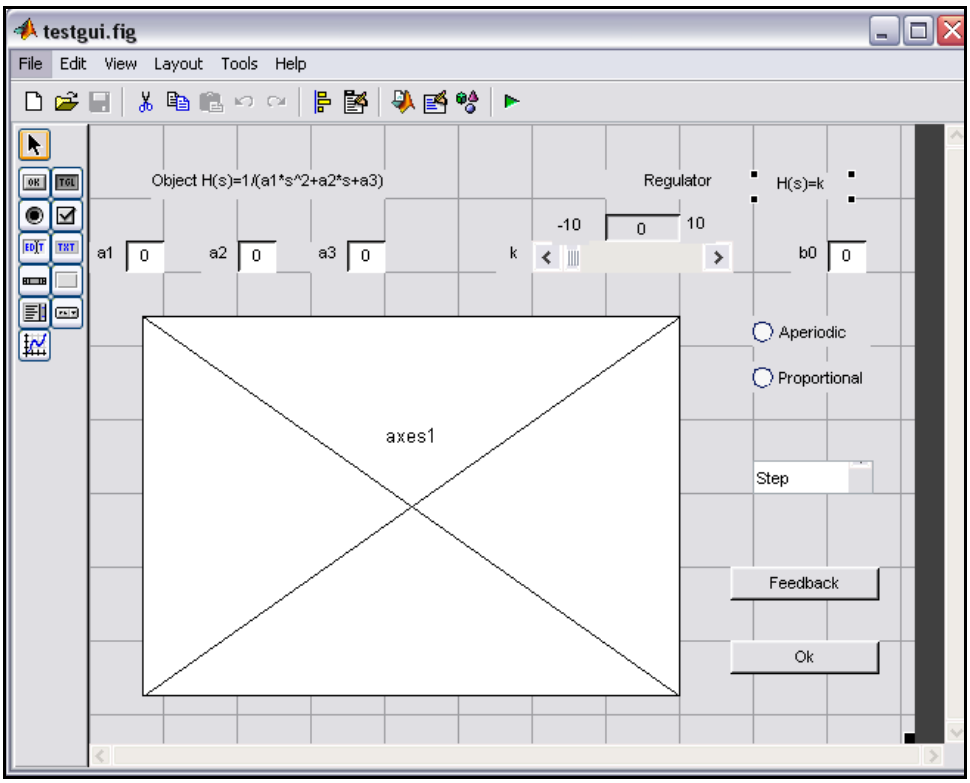

Рис. 3.2. Рабочий экран среды GUIDE: создание окна с элементами управления

Примечание 1: поскольку вид регулятора выбирается пользователем, поверх поля статического текста «**H(s) = k**» создайте поле такого же размера с текстом «**H(s) = k / (s+b0)**». Затем поместите его на задний план.

Примечание 2: для ровного расположения группы элементов их следует выделить. Для этого выбрать в панели элементов значок курсора в виде стрелки. Выбрать в меню **Tools** пункт **Align objects** либо нажать соответствующую ему кнопку на панели инструментов. В открывшемся окне параметров выравнивания можно указать тип (горизонтальное, вертикальное), расстояние между элементами (в пикселах) и т. д.

Примечание 3: инструкции, поясняющие как изменить подписи элемен-

тов управления, приведены в следующем разделе.

Свойства элементов управления. Прежде чем переходить к процедуре обработки событий, необходимо ознакомиться со свойствами элементов управления.

Список всех элементов можно просмотреть через меню **View -> Object Browser** (откроется в отдельном окне). Окно свойств элементов открывается через меню **View -> Property Inspector**. Двойным щелчком по элементу управления в окне редактирования или в списке можно перейти к свойствам конкретного элемента (рис. 3.3).

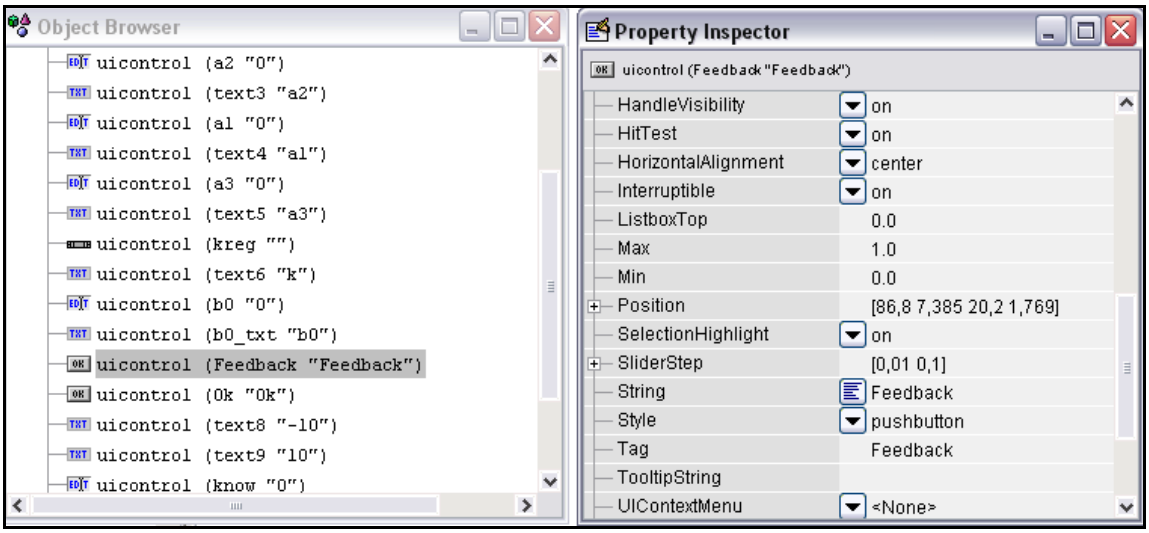

Рис. 3.3. Вид окна «Свойства элемента»

Свойства полностью определяют внешний вид и возможности реакций на события элементов управления. Выделим основные свойства:

Tag – метка, уникальное имя элемента, по которому осуществляется доступ к любым свойствам элемента из программы обработчика.

Примечание 1: при выборе тегов для элементов рекомендуется выбирать интуитивно понятные имена, чтобы облегчить задачу описания обработки событий. К примеру, в указанном примере список имеет тег «plottype», поля для ввода коэффициентов называются a1, a2 и т. д. (верхний и нижний регистры в тегах также различаются).

String – строка текста, в которой задается видимое имя элемента управления. В полях для ввода текста через это поле осуществляется доступ к вводимым данным. В элементе типа «список» в данном свойстве указываются все возможные значения списка. Для этого нужно нажать иконку в свойстве. В рассматриваемом примере для списка с тегом plottype введите две строчки:

plot, impulse.

Примечание 2: в поле ввода текста данные имеют тип **string** – строковый. Чтобы использовать данные в виде чисел, их нужно преобразовывать из строкового типа в числовой функцией **str2double**().

Visible – свойство видимости элемента (принимает значения «On», «Off»). В рассматриваемом примере установите параметр видимости текста «**H(s) = k»** в «Off».

Value – значение поля. Для флажков, радиокнопок и т. д. данное свойство принимает значение 1 или 0 (установлен/не установлен).

Для элементов типа **slider** можно установить значения полей max и min – границы изменяемого диапазона. Изменить тип элемента управления можно при помощи свойства *Style*.

3.2. Алгоритмизация приложений пользовательского интерфейса

Математические модели в виде (1.7)–(1.9) достаточно легко описываются алгоритмами в виде стандартных блок-схем. В то же время для приложений GUI такой алгоритмический подход не слишком удобен, так как использование приложения предполагает постоянное взаимодействие с оператором. Для создания алгоритма нужно описать все возможные события, которые поддерживаются элементами управления, из которых состоит приложение, и некоторую последовательность действий при обработке этих событий. Такой алгоритм удобно назвать сценарием.

Сценарий можно представить в виде таблицы либо текста. В качестве примера приводится процедура составления сценария для приложения, представленного на рис. 3.2.

Положим, при запуске приложения все коэффициенты математической модели объекта равны единице, и по умолчанию предлагается обратную связь считать апериодическим звеном. Построение графика обеспечивается нажатием кнопки Feedback. Коэффициент регулятора задается ползунком. В статическом поле над ползунком должно выводиться текущее значение коэффициента. Пользователь может ввести значения коэффициентов передаточной функции, выбрать тип регулятора (пропорциональный, апериодическое звено) и вид выходного графика (переходная характеристика, импульсная характеристика).

Окончание работы с приложением производится по нажатию кнопки OK, окно приложение при этом должно быть закрыто.

Сценарий для приложения приведен в табл. 3.1.

Таблица 3.1

Сценарий приложения testgui с графическим интерфейсом

Событие	Идентификатор элемента управления	Действие	Примечание
Открытие окна приложения testgui (начальная установка)	Testgui	Установить значения a1, a2, a3, b0 равными 1	Свойство **String** объектов a1, a2, a3, b0
		Включить радиокнопку Aperiodic	Свойство **Value** объекта aper
		Включить видимость текста H(s)=k/(s+b0)	Свойство **Visible** объекта aper_txt
		Выключить видимость текста H(s)=k	Свойство **Visible** объекта prop_txt
		Включить видимость объекта b0	Свойство **Visible** объекта b0
Изменение положение ползунка регулятора k	Kreg	Считать положение ползунка регулятора k	Свойство **Value** объекта kreg
		Записать значение положение ползунка в текстовом поле регулятора	Свойство **String** объекта know
Включение радиокнопки Aperiodic	Aper	Включить радиокнопку Aperiodic	Свойство **Value** объекта aper
		Выключить радиокнопку Proportional	Свойство **Value** объекта prop
		Включить видимость текста **H(s)=k/(s+b0)**	Свойство **Visible** объекта aper_txt
		Выключить видимость текста **H(s)=k**	Свойство **Visible** объекта prop_txt
		Включить видимость текстового поля b	Свойство **Visible** объекта b0
		Включить видимость текста b0	Свойство **Visible** объекта b0_txt
Включение радиокнопки Proportional	Prop	Включить радиокнопку Proportional	Свойство **Value** объекта prop
		Выключить радиокнопку Aperiodic	Свойство **Value** объекта aper
		Выключить видимость текста **H(s)=k/(s+b0)**	Свойство **Visible** объекта aper_txt
		Включить видимость текста **H(s)=k**	Свойство **Visible** объекта prop_txt
		Выключить видимость полей b и b0	Свойство **Visible** объектов b0 и b0_txt

Событие	Идентификатор элемента управления	Действие	Примечание
Нажатие кнопки Feedback	Feedback	Считать значения коэффициентов a1, a2, a3, b0	Свойство **String** объектов a1, a2, a3, b0
		Считать значение коэффициента регулятора k	Свойство **Value** объекта kreg
		Считать заданный тип регулятора	Свойство **Value** объекта aper, свойство **Value** объекта prop
		Считать заданный вид графика	Свойство **Value** объекта plottype
		Запустить процедуру расчета программной модели	–
Нажатие кнопки OK	OK	Закрыть окно приложения testgui	–

Отличием сценариев от стандартных алгоритмов является нестрогий порядок выполнения действий. Это касается как программирования событий в целом, так и последовательности изменения свойств при программировании каждого события.

3.3. Описание обработки событий

Процедура обработки событий состоит в программном описании изменений свойств элементов GUI. Как уже отмечалось, при создании окна автоматически создается файл filename.m. Перейти к этому файлу из окна редактирования GUI можно, используя меню **View –> M-file Editor**, либо через соответствующую ему иконку на панели инструментов.

Рассмотрим структуру файла. Это файл-функция, совпадающая по имени с файлом окна с элементами GUI, содержащая встроенные функции, существующие только для данного приложения, описывающие действия с элементами управления. Функции, в основном, бывают двух типов:

CreateFcn() – набор инструкций, выполняемых при создании элемента управления (так называемые конструкторы);

Callback() – набор инструкций, выполняемый в случае действия с элементом управления (ввод текста в поле, нажатие кнопки, выбор элемента списка и т. д.). Это так называемые функции обратного вызова.

Следует обратить внимание на то, что все имена функций имеют общую структуру: тег_тип(). Практически все функции имеют следующие аргументы:

hObject – объект, тег которого указан в имени функции;

eventdata – зарезервированный параметр для будущих версий MATLAB (не используется);

handles – «родительский» объект окна GUI, содержащий все прочие существующие в окне объекты. Используя этот параметр, можно изменять свойства любого элемента управления окна.

Для изменения свойств объекта используется функция **set**(). Для доступа к свойствам объекта используется функция **get**().

В соответствии с табл. 3.1 необходимо задать некоторые начальные свойства приложения testgui. Следует отметить, что начальный выбор ненулевых параметров модели предотвращает ошибки работы программы (иначе по умолчанию объект являлся бы некорректным). Для этого отыщем функцию **testgui_OpeningFcn**() (конструктор testgui) и введем кроме имеющихся операторов следующие:

```
set(handles.aper,'Value',1); % установка радиокнопки Aperiodic
set(handles.a1,'String',1); % начальное значение коэффициента a1 и т. д.
set(handles.a2,'String',1);
set(handles.a3,'String',1);
set(handles.b0,'String',1);
```

Примечание 1: выражение типа handles.a1 означает – объект с тегом a1, принадлежащий родительскому объекту handles.

При плавной регулировке коэффициента желательно просматривать его текущее значение. Для этого в окне создано поле с тегом know – оно расположено непосредственно над ползунком. Его значение должно изменяться при регулировке, следовательно, находим функцию обратного вызова ползунка (в примере она называется **kreg_Callback**()) и вводим в ней следующие операторы:

```
l = get(hObject,'Value'); % извлечение значения текущего положения
% слайдера;
set(handles.know,'String',l); % вывод текущего положения в объекте
% с тегом know.
```

В случае нажатия кнопки OK необходимо закрыть окно приложения. Соответственно, в функции **OK_Callback**() вводим команду **close**.

Хотя полностью приложение еще не сформировано, можно посмотреть на проделанные действия. Вызов приложения можно сделать стандартными

способами MATLAB из главного окна или окна редактора M-файла. Кроме того, запустить приложение можно через окно редактора GUI, выбрав в меню **Tools** подпункт **Run** либо нажав соответствующую ему кнопку на панели инструментов.

Кроме имеющихся функций обработки событий можно также создавать собственные функции или использовать стандартные. К примеру, анализируя табл. 3.1, можно заметить, что процедура выбора типа регуляторов через радиокнопки содержит практически одинаковые действия. Очевидно, что сразу обе радиокнопки не могут быть нажаты, а значит, после включения одной надо выключать вторую. Аналогично – после выключения. Кроме того, у нас предусмотрено информационное окно с выводом вида регулятора, а коэффициент b0 нужен только для апериодического звена. Чтобы не писать инструкцию для каждой радиокнопки два раза, создадим собственную функцию переключения, включающую все описанные инструкции:

```
function turnto(handles,choose)
if choose == 'aper'
  set(handles.prop,'Value',0);
  set(handles.aper,'Value',1); % переключения радиокнопок
  set(handles.b0,'Visible','On');
  set(handles.b0_txt,'Visible','On'); % управление видимостью b0
  set(handles.aper_txt,'Visible','On');
set(handles.prop_txt,'Visible','Off'); % управление видимостью текстовой
% информации
elseif choose == 'prop'
  set(handles.prop,'Value',1);
  set(handles.aper,'Value',0);
  set(handles.b0,'Visible','Off');
  set(handles.b0_txt,'Visible','Off');
  set(handles.aper_txt,'Visible','Off');
  set(handles.prop_txt,'Visible','On');
end
```

Теперь в процедурах обратного вызова радиокнопок добавим условные операторы. Для функции **aper_Callback**():

```
if get(hObject,'Value')==1
  turnto(handles,'aper');
```

```
else
  turnto(handles,'prop');
end
```

Для **prop_Callback**(), соответственно, наоборот:

```
if get(hObject,'Value')==1
  turnto(handles,'prop');
else
  turnto(handles,'aper');
end
```

Фактически, осталось описать одну инструкцию, соответствующую нажатию кнопки Feedback. Она осуществляет запуск программной модели. В данном случае программную модель удобно реализовать средствами библиотеки CONTROL TOOLBOX.

Таким образом, нажав кнопку Feedback, нужно считывать текущие коэффициенты объекта и установленного регулятора, сформировать из них lti-объекты, замкнуть объект регулятором и построить временные характеристики в зависимости от выбранного в списке вида графика.

В функции Feedback_Callback() пишем инструкции:

```
a1_h = handles.a1;  % выбор объекта a1 из родительского объекта
a1 = str2double(get(a1_h,'String')); % извлечение численного значения из
% объекта a1
a2_h = handles.a2;
a2 = str2double(get(a2_h,'String'));
a3_h = handles.a3;
a3 = str2double(get(a3_h,'String'));
b0_h = handles.b0;
b0 = str2double(get(b0_h,'String'));
k_h = handles.kreg;
k = get(k_h,'Value');

tf_obj = tf(1,[a1 a2 a3]);
if get(handles.aper,'Value')==1  % если регулятор – апериодическое звено
  tf_reg = tf(k,[1 b0]);
elseif get(handles.prop,'Value')==1 % если регулятор – пропорциональное
% звено
```

```
  tf_reg = tf(k);
else
  error('Wrong regulator definition'); % сообщение об ошибке «неверно
  % задан регулятор»
end
tf_z = feedback(tf_obj,tf_reg);
axes(handles.axes1); % размещение осей будущего рисунка в объекте
% axes1 нашего окна
cla;                 % очистка осей перед выводом (стандартная процедура)
if (get(handles.plottype,'Value')==1) % первое значение в списке - Step
  [y,t] = step(tf_z);
  plot(t,y)
elseif (get(handles.plottype,'Value')==2) % второе значение в списке Im-
pulse
  [y,t] = impulse(tf_z);
  plot(t,y)
end
```

Теперь приложение с графическим интерфейсом сформировано. Различные способы запуска приложений уже были описаны. Внешний вид приложения testgui приведен на рис. 3.4.

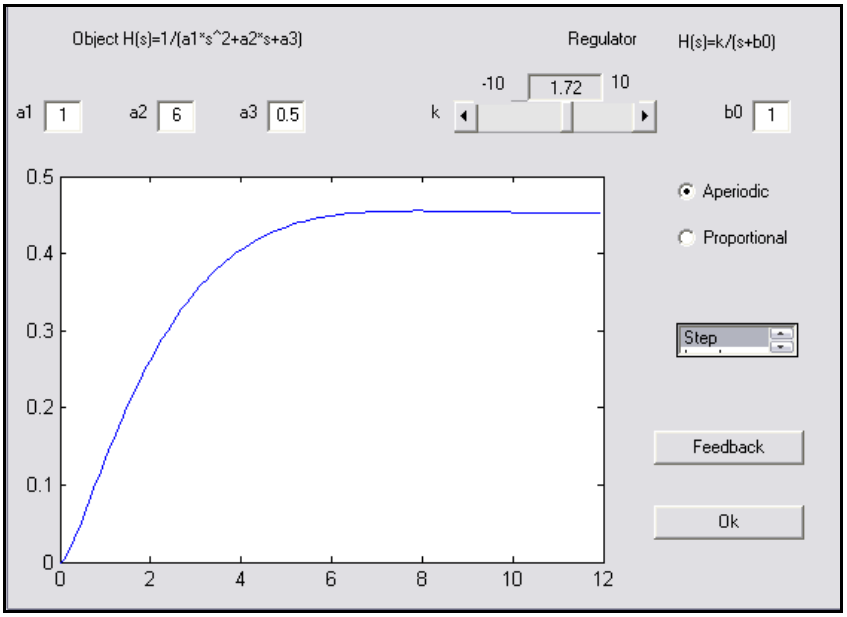

Рис. 3.4. Вид окна приложения

Приложение скомпоновано следующим образом: в верхней части задаются числовые параметры (слева – модель, справа – регулятор), большая

часть рабочей области отведена под график – главный результат работы приложения. В правой части по вертикали собраны дополнительные средства управления приложением. Для большего удобства рабочую область приложения можно поделить на сегменты, используя средство GUI рамку (Frame). Рамка редко используется при обработке событий, но при этом очень полезна для пользователя.

Примечание 2: в шаблоне M-файла с описанием обработок для каждого элемента управления в качестве помощи присутствуют комментарии, в которых указаны возможные действия с этим элементом управления.

3.4 Создание строк и команд меню в приложениях

Инструментарий среды GUIDE позволяет также создавать меню для пользовательских приложений. Фактически работа с элементами меню ничем не отличается от работы с другими элементами управления. Для создания меню приложения следует выбрать в среде GUIDE меню **Tools -> Menu editor**. Откроется окно редактора меню, в котором практически все действия выполняются с панели инструментов: создание и удаление разделов меню, создание и сортировка подпунктов меню. При выделении мышью раздела или пункта в правой части окна редактора задаются дополнительные параметры: тег и обозначение, разделитель и т. д.

Для примера создадим в нашем приложении два раздела меню. Один пусть дублирует выбор типа регулятора, второй выводит краткую информацию о назначении приложения. При помощи панели инструментов редактора меню создадим элементы согласно рис. 3.5. Процедуры выбора типа регулятора уже описаны при помощи функции turnto(). Соответственно, добавляем в функцию обратного вызова **aper_m_Callback**() строчку turnto(handles,'aper'). Аналогично, для второго подпункта вызываем функцию turnto со вторым параметром 'prop'.

Для описания второго раздела меню необходимо осуществить вывод текстовой информации. Проще всего это реализуется с помощью функции **helpdlg**(). Тогда в процедуре **info_m_Callback**() указываем следующий оператор:

```
helpdlg({'Замыкание линейного объекта регулятором';
  'Выбирайте тип регулятора и вид графика';})
```

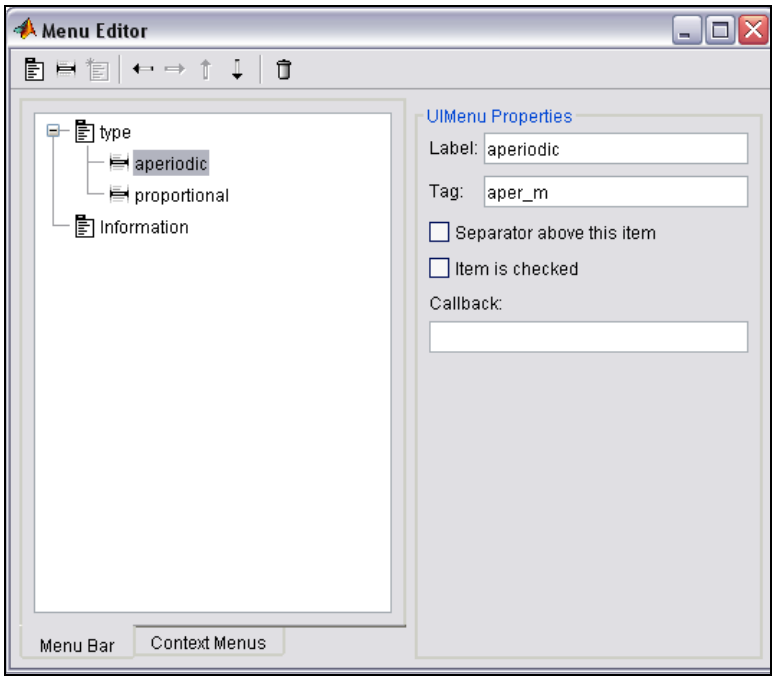

Рис. 3.5. Вид окна «создание элементов»

Итоговый вид приложения с меню показан на рис. 3.6 и 3.7.

Примечание 1: Функция **helpdlg**() – не единственная для вывода текстовой информации. Есть функции **msgbox**(), **dialog**() и др. Описание этих функций можно найти через help.

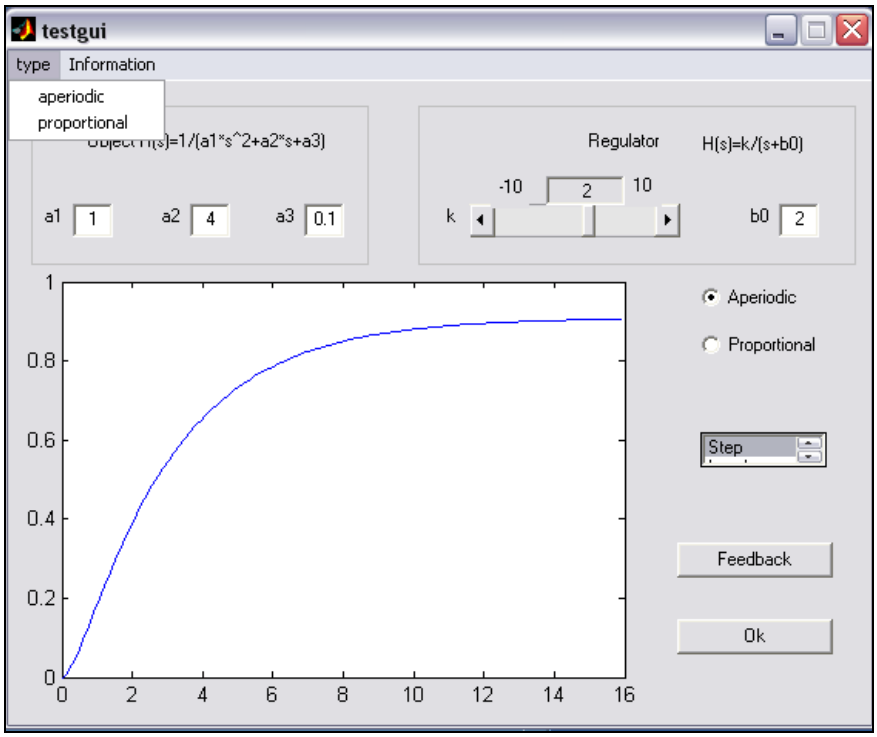

Рис. 3.6. Рабочий экран приложения с меню

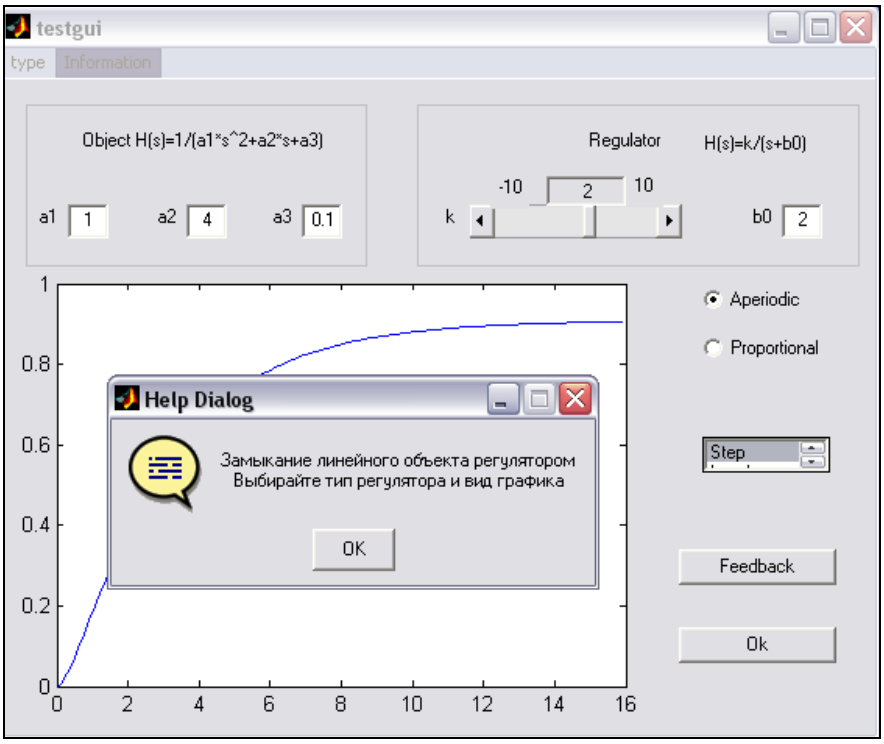

Рис. 3.7. Рабочий экран приложения с диалоговым окном помощи

Примечание 2: Более поздние версии пакета MATLAB имеют несколько отличный от рассматриваемого в п. 3 интерфейс, содержат другие элементы управления и стандартные функции обратного вызова. Их подробное описание и инструкцию по использованию приведено в [4].

4. СОДЕРЖАНИЕ КУРСОВОЙ РАБОТЫ

1. Разработать математическую модель движения корабля по индивидуальным исходным данным (п. 5).

2. Составить и отладить напрерывную и разностную программную модель движения корабля на языке высокого уровня MATLAB.

3. Разработать интерфейс программы моделирования кораблей разных типов с использованием средств MATLAB GUI, позволяющий осуществлять в рабочем окне ввод исходных данных и вывод результатов.

4. Исследовать динамические свойства моделируемого объекта на ПК, используя примеры или по указанию преподавателя. Результаты занести в таблицу.

5. По данным таблицы построить графики функций $P = f(t)$; $x = f(t)$; $V = f(t)$.

6. По построенным графикам определить основные параметры движения корабля, указанные в соответствующих примерах.

Содержание пояснительной записки:

1) исходные данные;
2) математическая модель движения корабля;
3) программная модель;
4) сценарий программы с графическим интерфейсом;
5) рабочие экраны интерфейса программы моделирования;
6) результаты исследования программной модели (таблица(ы), графики);
7) основные параметры движения корабля при разгоне и торможении.

Требования к приложению с графическим интерфейсом:

1. Возможность ввода с клавиатуры исходных данных по объектам управления. Для упрощения допускается указать исходные данные индивидуального варианта в качестве начальных значений текстовых полей приложения.

2. Возможность выбора способа интегрирования – непрерывной и разностной программных моделей.

3. Возможность ввода с клавиатуры основных параметров интегрирования: начальных значений $x(0)=x_0$, $v(0)=v_0$, диапазона интегрирования для функции ODE45, и т. д.

4. Наличие ступенчатой регулировки относительной тяги. Шаг рекомендуется выбрать равным 10 %.

5. Приложение должно обеспечить возможность раздельного моделирования режимов разгона и торможения.

5. ИСХОДНЫЕ ДАННЫЕ ДЛЯ КУРСОВОГО РАСЧЕТА

В табл. 5.1–5.3 содержатся сведения, необходимые для разработки математических моделей движения кораблей. При их использовании необходимо учесть следующее.

1. Максимальная скорость $v_{\max}$ должна быть переведена в СИ с учетом, что 1 узел = 1 миля/ч = 1852/3600 м/с = 0,51 м/с.

2. Максимальная сила тяги $F_{\max}$ может быть приближенно определена из следующих соотношений: 1 л. с. = 735,5 Вт; 1 Вт = 1 Н·м/с,

$$F_{\max} = N_{\max} / v_{\max} = (\text{Н} \cdot \text{м/с}) / (\text{м/с}) = \text{Н},$$

где $N_{\max}$ – максимальная мощность двигателей, Вт = Н·м/с.

3. Максимально допустимую скорость изменения силы тяги ΔF принять:

– для кораблей с водоизмещением до 10 000 т – 0,2 $F_{\max}$;

– для кораблей с большим водоизмещением – 0,1 $F_{\max}$.

Примечание: в таблицах приняты следующие обозначения:

W – водоизмещение, т;

N – мощность ГЭУ, л. с.;

v – скорость, узлы.

Таблица 5.1

Основные ТТД надводных кораблей [5]

№ НВК	Наименование НВК	W	N	v
1	«Марат»	26 170	61 000	23,0
2	«Октябрьская Революция»	25 464	60 600	23,0
3	«Парижская Коммуна»	30 395	61 000	21,5
4	«Архангельск»	33 500	40 000	20,5
5	«Киров»	9 550	122 500	34,0
6	«М.Горький»	9 728	129 750	36,1
7	«Красный Кавказ»	9 030	55 000	29,0
8	«Червона Украина»	7 999	46 300	22,0
9	«Мурманск»	10 400	90 000	30,0
10	«Аврора»	7 271	11 610	20,0
11	«Ташкент»	3 200	110 000	42,0
12	«Ленинград»	2 693	66 000	43,0
13	«Минск»	2 597	66 900	40,0
14	«Огневой»	2 767	60 000	37,0
15	«Сторожевой»	2 529	54 000	38,0
16	«Гневный»	2 402	56 500	38,6
17	«Опытный»	2 016	70 000	42,0
18	«Новик»	1 700	30 000	27,0
19	«Жаркий»	1 552	26 000	27,0
20	«Легкий»	1 745	45 000	21,0
21	«Хасан»	1 900	200	15,4
22	«Ленин»	1 082	2 970	11,3
23	«Ударный»	253	400	9,0
24	«Активный»	314	480	8,5
25	«Бобруйск»	130	200	9,0
26	«Смоленск»	750	1 800	13,0
27	«Красное Знамя»	1 823	2 200	14,5
28	«Бакинский рабочий»	760	6 200	19,0
29	«Красная Абхазия	1 400	750	8,0
30	«Ленин»	750	2 200	15,0

Таблица 5.2

Основные ТТД катеров [6]

№ НВК	Тип/наименование катера	W	N	v
1	Большой разъездной катер	32,0	150	10,5
2	Большой разъездной катер	8,7	150	14,5
3	Малый разъездной катер	3,2	55	13,5
4	Большой рабочий катер	11,0	100	9,5
5	Малый рабочий катер	4,6	50	9,0
6	«Тритон»	2,3	150	29,0
7	«Аллигатор»	0,8	36	19,0
8	«Новинка»	1,2	70	20,8
9	«Юбилейный»	1,7	90	28,0
10	«Аист»	4,6	235	19,5
11	«Невка»	6,0	235	30,0
12	«Ракета»	25,5	1 000	30,0

Таблица 5.3

Основные ТТД кораблей на подводных крыльях [5]

№ КПК	W	N	Скорость, узлы		
			v_k	$v_{1_{\max}}$	$v_{2_{\max}}$
1	1,5	60	11,8	14,2	34
2	2,0	80	12,2	15,0	36
3	3,0	120	12,1	14,6	35
4	4,5	180	11,0	14,0	34
5	5,0	200	9,7	11,0	30
6	6,7	300	12,9	16,0	38
7	7,0	274	10,0	11,4	30
8	8,0	313	10,5	12,0	29
9	9,4	400	11,0	13,0	32
10	10,0	500	13,0	17,0	40
11	11,0	430	10,2	11,8	30
12	12,8	500	10,0	11,7	30
13	13,0	650	14,0	17,5	42
14	14,0	550	10,7	12,0	30
15	14,0	1 100	14,6	17,5	42
16	16,0	627	10,0	12,5	30
17	17,8	700	11,0	13,0	32
18	18,2	1 000	12,5	15,0	36
19	18,2	713	11,0	13,0	32
20	25,0	1 000	10,4	12,5	30
21	21,0	800	9,5	11,7	28
22	22,5	900	9,8	11,8	30
23	23,0	900	10,4	12,5	30
24	23,0	800	9,0	10,8	26
25	25,0	1 200	12,1	15,6	42
26	26,5	1 000	14,6	17,5	42

Окончание табл 5.3

№ КПК	W	N	Скорость, узлы		
			v_k	$v_{1_{\max}}$	$v_{2_{\max}}$
27	27,7	1 100	13,9	16,7	40
28	40,0	2 000	13,2	15,8	38
29	50,0	3 500	16,0	19,2	46
30	60,0	3 500	13,2	15,8	38
31	30,0	1 200	10,0	12,8	30
32	35,0	1 400	11,0	13,0	32
33	24,0	1 000	14,0	17,0	40
34	15,0	605	12,5	15,0	36
35	17,0	800	13,2	16,0	38
36	45,0	2 000	14,5	17,5	42
37	48,0	2 000	14,0	17,0	40

Каждый студент по указанию преподавателя получает индивидуальное задание в виде трех чисел, соответствующих строкам табл. 5.1–5.3.

Примечание: обработка исходных данных (перевод значений в единицы измерения системы СИ и т. д.) должна осуществляться приложением. Исходные данные, приведенные в табл. 5.1–5.3 должны быть заданы в приложении без изменений.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. К какому классу математических моделей относятся модели НВК и КПК?
2. Поясните форму записи силы гидродниамического сопротивления.
3. Что такое глиссирование? Чем характеризуется этот режим движения?
4. При каких ограничениях можно применять математические модели кораблей?
5. Дайте сравнительный анализ интегрирования непрерывной и разностной математических моделей.
6. Можно ли использовать LTI-модели (см. пример из п. 4) для моделирования моделей НВК и КПК?
7. Поясните процедуру уточнения математической модели КПК.
8. Сравните время разгона и торможения судов на полной тяге. Поясните полученные результаты.

9. Поясните смысл скоростей $v_{1\max}$ и $v_{2\max}$ в уравнениях движения КПК.

10. На каком основании силу тяги можно задавать, не используя дифференциальное уравнение (1.6)?

11. Как изменятся переходные процессы движения при неполной тяге двигателя? Поясните результаты.

12. Каким образом определяется коэффициент сопротивления *A*?

13. Какие преимущества есть у разностных математических моделей перед непрерывными?

14. Каким образом программируются события в приложениях GUI среды MATLAB?

15. Перечислите основные элементы управления графического интерфейса.

16. Каким образом формируется сценарий для приложения? Какое у него назначение?

17. Что такое функция-конструктор и функция начального вызова?

18. Как определить исходные данные приложения GUI?

19. Что такое «родительский» объект в приложениях GUI?

20. Поясните смысл параметров функций обратного вызова.

СПИСОК ЛИТЕРАТУРЫ

1. Джильмер Т. С. Проектирование современного корабля. Л.: Судостроение, 1974.

2. Мирошников А. Н., Румянцев С. Н. Моделирование систем управления технических средств транспорта: учеб. изд-е. СПб.: Элмор, 1999.

3. Бубнов Е. А. Системы управления судов с АЭУ. СПб., 1998.

4. Ануфриев И. Е. Приложения с GUI и дескрипторная графика. URL: http://matlab.exponenta.ru/gui/

5. Бережной С. С. Корабли и суда ВМФ СССР. 1928–1945: Справочник. М.: Воениздат, 1988.

6. Иванов Л. Н., Сафонов А. И., Бурзун А. Е. Катер (устройство и управление). М.: Воениздат, 1974.

Содержание

Список сокращений 3
1. Формирование математических моделей объектов управления 3
1.1. Математическая модель движения надводного водоизмещающего корабля 3
1.2. Математическая модель движения корабля на подводных крыльях 6
1.3. Уточнение математических моделей 7
2. Формирование программных моделей 9
2.1. Выбор шага интегрирования 11
2.2. Примеры моделирования движения подвижных объектов 12
3. Разработка приложения с графическим интерфейсом 17
3.1. Знакомство со средствами графического интерфейса в среде MATLAB 18
3.2. Алгоритмизация приложений пользовательского интерфейса 23
3.3. Описание обработки событий 25
3.4. Создание строк и команд меню в приложениях 30
4. Содержание курсовой работы 32
5. Исходные данные для курсового расчета 33
Контрольные вопросы 37
Список литературы 38

Редактор Н. В. Лукина

Подписано в печать 00.00.14. Формат 60 × 84 1/16.
Бумага офсетная. Печать цифровая. Печ. л. 2,5.
Тираж 42 экз. Заказ .

Издательство СПбГЭТУ «ЛЭТИ»
197376, С.-Петербург, ул. Проф. Попова, 5