Содержание

1 Введение

Современные системы автоматического управления широко используют методы модального синтеза, позволяющие размещать полюса дискретной или непрерывной системы в заранее выбранных положениях. Одновременно с регулятором состояния нередко применяется наблюдатель, предоставляющий оценки недоступных напрямую переменных состояния. Качество функционирования замкнутой системы определяется не только выбором матрицы усиления регулятора, но и динамическими свойствами наблюдателя.

В данной лабораторной работе исследуются:

- влияние периода дискретизации T_0 на переходные процессы;
- влияние быстродействия наблюдателя на качество регулирования;
- влияние ошибки начальных условий наблюдателя.

Для варианта 1 задана матричная модель непрерывного объекта в пространстве состояний, по которой выполняется дискретизация и синтез регулятора состояния методом размещения полюсов.

2 Цель работы

Цель лабораторной работы — изучить методы дискретизации непрерывных систем и синтезировать дискретный регулятор состояния и наблюдатель, исследовав влияние динамики наблюдателя и периода дискретизации на поведение замкнутой системы.

3 Постановка задач

В соответствии с заданием необходимо:

1. Выполнить дискретизацию непрерывной системы при нескольких периодах дискретизации T_0.
2. Проверить управляемость и наблюдаемость дискретной модели.
3. Синтезировать регулятор состояния методом размещения полюсов.
4. Найти коэффициент k_0, обеспечивающий нулевую статическую ошибку.
5. Синтезировать наблюдатель с «быстрыми» полюсами, а также наблюдатель с «медленными» полюсами.
6. Исследовать влияние:
   - периода дискретизации;
   - выбора полюсов наблюдателя;
   - ошибки начальных условий $\hat{x}(0) \neq x(0)$.
7. Выполнить моделирование переходных процессов.

4 Структурная схема модели

Моделирование проводилось в среде Simulink. Используемая структура системы показана на рис. 1.

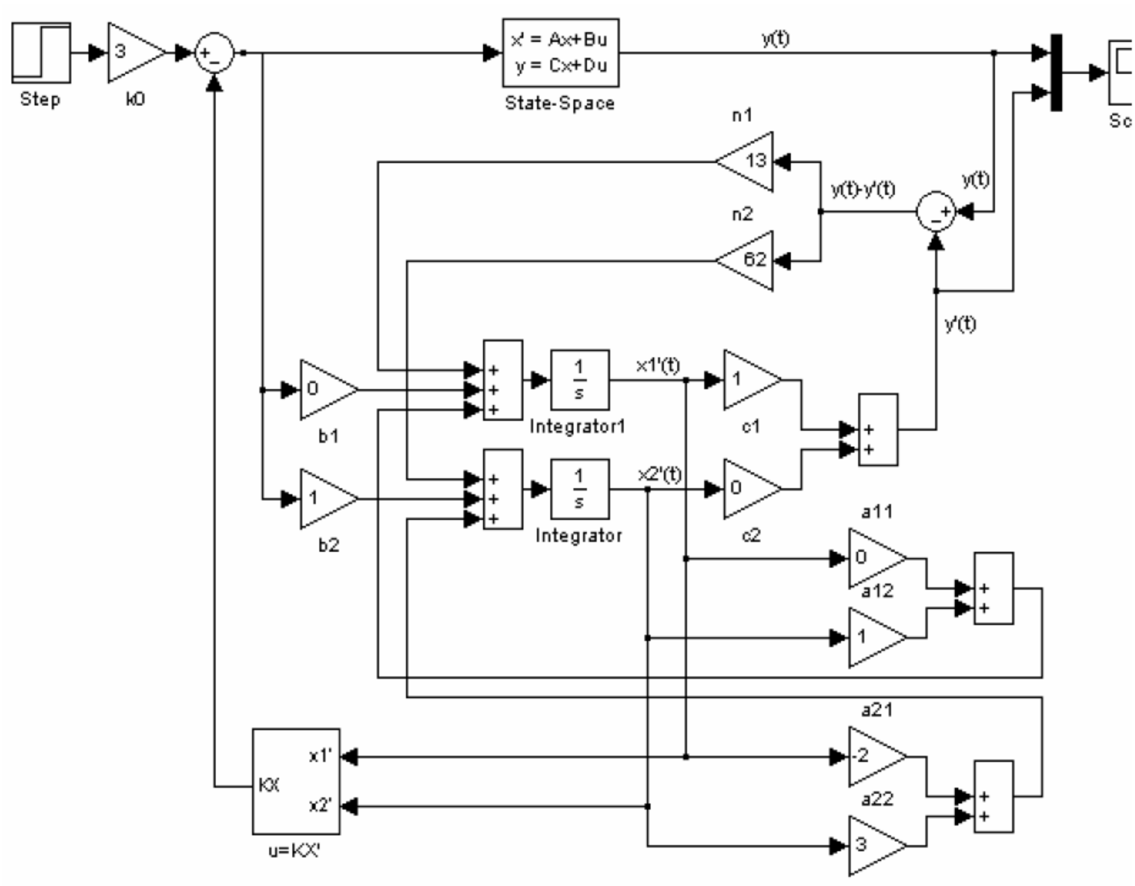

Рис. 1: Схема системы с регулятором и наблюдателем

На схеме видно:

- блок State-Space реализует объект управления;
- регулятор формирует $u(k) = -K\hat{x}(k) + k_0 w(k)$;
- наблюдатель реализован по уравнению:

$$\hat{x}(k+1) = A_d\hat{x}(k) + B_d u(k) + L\big(y(k) - \hat{y}(k)\big);$$

- выход объекта положительной обратной связью подаётся на наблюдатель.

Такая структура полностью соответствует аналитической модели.

5 Переходные характеристики замкнутой системы

Проведено моделирование при трёх значениях периода дискретизации:

$$T_0 \in \{0.1;\ 1;\ 10\} \text{ с.}$$

На рис. 2 представлены переходные характеристики $y(t)$.

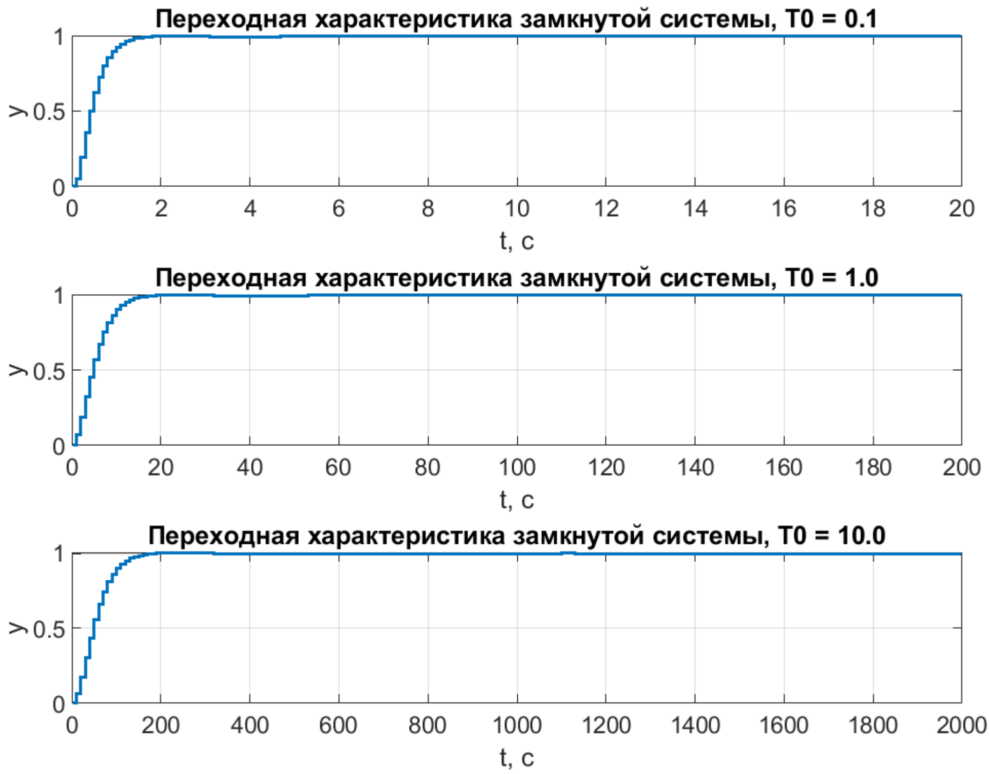

Рис. 2: Переходные характеристики замкнутой системы при различных T_0

Анализ графиков:

- при $T_0 = 0.1$ с система обладает наибольшим быстродействием: установление занимает около 2–3 секунд;
- при $T_0 = 1$ с динамика замедляется примерно в 10 раз, но форма переходного процесса остаётся гладкой;

- при $T_0 = 10$ с переходный процесс растягивается до сотен секунд, однако устойчивость не нарушается.

Таким образом, увеличение периода дискретизации приводит к ухудшению быстродействия, но структура системы остаётся корректно функционирующей.

6 Влияние параметров наблюдателя

На рис. 3 приведены переходные процессы для трёх случаев:

- быстрый наблюдатель, $\hat{x}(0) = 0$;
- медленный наблюдатель, $\hat{x}(0) = 0$;
- быстрый наблюдатель, но $\hat{x}(0) \neq 0$.

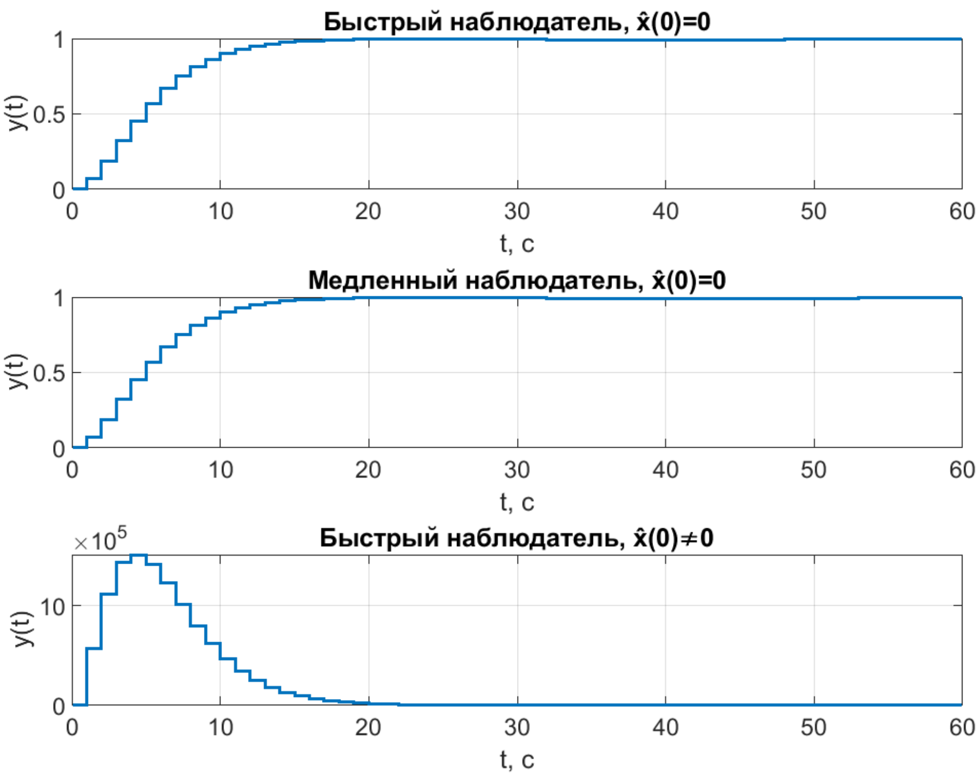

Рис. 3: Влияние параметров наблюдателя на переходный процесс

Быстрый наблюдатель, $\hat{x}(0) = 0$

Полюса наблюдателя заданы значительно быстрее, чем полюса системы. Наблюдение:

- выход $y(t)$ практически совпадает с эталонным переходным процессом;
- наблюдатель быстро подавляет ошибку оценивания;
- перерегулирование отсутствует.

Медленный наблюдатель, $\hat{x}(0) = 0$

Полюса близки к единичной окружности:

$$p_{\text{obs}} = \{0.9,\ 0.92,\ 0.95\}.$$

Наблюдение:

- переходный процесс становится заметно медленнее;
- нарастание выходного сигнала более плавное;
- наблюдатель «тормозит» замкнутую систему.

Быстрый наблюдатель, $\hat{x}(0) \neq 0$

Начальная оценка задана как:

$$\hat{x}(0) = [5\ 5\ 5]^T.$$

Наблюдение:

- возникает огромный бросок выхода ($\sim 10^5$);
- быстрая динамика наблюдателя усиливает ошибку начальных условий;
- система после кратковременного выброса остаётся устойчивой и затухает.

Этот случай показывает: **быстрые наблюдатели требуют точной инициализации**, иначе возможны большие переходные перенапряжения.

7 Общие выводы

По результатам работы можно сделать следующие выводы:

1. Увеличение периода дискретизации T_0 приводит к замедлению переходного процесса, однако структура системы остаётся устойчивой.

2. Быстрый наблюдатель обеспечивает высокое качество переходного процесса и быстрое затухание ошибки оценивания.

3. Медленный наблюдатель ухудшает динамику системы и снижает быстродействие.

4. Ошибка начальных условий наблюдателя при его высокой скорости может вызывать огромные кратковременные выбросы выходной величины.

5. Для реальных систем важно правильно выбирать компромисс между быстродействием наблюдателя и чувствительностью к ошибкам инициализации.

Приложение. MATLAB-код

```
clc; clear; close all;

A = [-39   -71   -23;
      23    40    13;
     -6.6 -10.7  -3.3];

B = [0.228; -0.127; 0.038];
C = [28000 84000 112000];
D = 0;

T0_list = [0.1 1 10];
p_cl    = [0.5 0.6 0.7];
```

```
x0      = [0;0;0];
xhat0   = [0;0;0];
Nsteps  = 200;

results = struct([]);

for k = 1:numel(T0_list)

    T0 = T0_list(k);
    fprintf('\n================================\n');
    fprintf('PERIOD T0 = %.1f s\n', T0);
    fprintf('================================\n');

    sysd = c2d(ss(A,B,C,D), T0, 'zoh');
    [Ad,Bd,Cd,Dd] = ssdata(sysd);

    rWc = rank(ctrb(Ad,Bd));
    rWo = rank(obsv(Ad,Cd));
    fprintf('rank(Wc)=%d, rank(Wo)=%d\n', rWc, rWo);

    K = place(Ad, Bd, p_cl);
    fprintf('K =\n'); disp(K);

    sys_cl = ss(Ad - Bd*K, Bd, Cd, 0, T0);
    dc = dcgain(sys_cl);
    k0 = 1/dc;
    fprintf('k0 = %.6g\n', k0);

    p_obs = p_cl.^2;
    L = place(Ad', Cd', p_obs)';
    fprintf('L =\n'); disp(L);

    Ablock = Ad - Bd*K - L*Cd;
    Bblock = [Bd*k0, L];
    Cblock = -K;
    Dblock = [k0, 0];

    results(k).T0 = T0;
```

```
    results(k).Ad = Ad;  results(k).Bd = Bd;  results(k).Cd = Cd;
    results(k).K  = K;   results(k).k0 = k0;
    results(k).L  = L;
    results(k).Ablock = Ablock;
    results(k).Bblock = Bblock;
    results(k).Cblock = Cblock;
    results(k).Dblock = Dblock;
end

%% Step response of closed-loop system
figure('Name','Step responses of closed-loop system');

for k = 1:numel(T0_list)

    T0 = results(k).T0;

    Ablock = results(k).Ablock;
    Bblock = results(k).Bblock;
    Cblock = results(k).Cblock;
    Dblock = results(k).Dblock;

    sys_ctrl = ss(Ablock, Bblock, Cblock, Dblock, T0);

    Ad = results(k).Ad; Bd = results(k).Bd; Cd = results(k).Cd;
    K  = results(k).K;  k0 = results(k).k0;

    sys_closed = ss(Ad - Bd*K, Bd*k0, Cd, 0, T0);

    t = (0:Nsteps)*T0;
    w = ones(size(t));
    y = lsim(sys_closed, w, t);

    subplot(3,1,k);
    stairs(t,y,'LineWidth',1.4);
    grid on;
    xlabel('t, s'); ylabel('y');
    title(sprintf('Closed-loop step response, T0 = %.1f',T0));
end
```

```

%% Observer influence (fast/slow, different xhat0)
T0 = 1;
idx = find([results.T0] == T0);

Ad = results(idx).Ad;
Bd = results(idx).Bd;
Cd = results(idx).Cd;

K  = results(idx).K;
k0 = results(idx).k0;

L_fast = results(idx).L;

p_obs_slow = [0.9 0.92 0.95];
L_slow = place(Ad', Cd', p_obs_slow).';

figure('Name','Observer influence on step response');

cases = { ...
    struct('L', L_fast, 'xhat0', [0;0;0], 'title', 'Fast observer, xhat(0)=0'), ...
    struct('L', L_slow, 'xhat0', [0;0;0], 'title', 'Slow observer, xhat(0)=0'), ...
    struct('L', L_fast, 'xhat0', [5;5;5], 'title', 'Fast observer, xhat(0)!=0') ...
};

x0 = [0;0;0];
t  = 0:T0:60;
w  = ones(size(t));

for i = 1:numel(cases)
    L = cases{i}.L;
    xhat0 = cases{i}.xhat0;

    Acl = [Ad          -Bd*K;
           L*Cd  Ad - Bd*K - L*Cd];

    Bcl = [Bd*k0;
           Bd*k0];
```

```
    Ccl = [Cd  zeros(size(Cd))];
    Dcl = 0;

    sys_obs = ss(Acl, Bcl, Ccl, Dcl, T0);

    z0 = [x0; xhat0];
    [y, t_out] = lsim(sys_obs, w, t, z0);

    subplot(3,1,i);
    stairs(t_out, y, 'LineWidth', 1.4);
    grid on;
    xlabel('t, s'); ylabel('y(t)');
    title(cases{i}.title);
    fprintf('y(end) = %.6f\n', y(end));
end
```

Листинг 1: Основной скрипт лабораторной работы