**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра КСУ**

**КУРСОВАЯ РАБОТА**

**по дисциплине «ПРОЕКТИРОВАНИЕ**

**ОПТИМАЛЬНЫХ СИСТЕМ УПРАВЛЕНИЯ»**

Тема: проектирования алгоритма управления динамическим объектом на примере водоизмещающего судна

**Вариант 12**

|  |  |  |
| --- | --- | --- |
| Студент гр. 4491 | Пономарев Д.А. |  |
| Преподаватель | Ветчинкин А.С. |  |

Санкт-Петербург

2018

Задание на курсовую работу

Студент: Пономарев Д.А.

Группа: 4491

Тема работы:

Исходные данные представлены в таблицах 1, 2 и 3.

Таблица 1 - Вариант курсового расчета

|  |  |  |
| --- | --- | --- |
| Вариант задания курсового расчета | Вариант параметров математической модели судна | Косвенный метод решения задачи оптимизации |
| 12 | 1 – 8 | Использование стандартного полинома П2.1 |

Таблица 2 - Вариант математической модели судна

|  |  |  |
| --- | --- | --- |
| Параметр | Обозначение | Вариант судна |
| 4 |
| Скорость хода | , м/сек | 6.17 |
| Длина по ватерлинии | , м | 39 |
| Коэффициенты математической модели |  | -0.69 |
|  | 6.14 |
|  | 1.22 |
|  | -3.12 |
|  | -0.44 |
|  | -3.1 |



Таблица 3 - П2.1 Биноминальные полиномы

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
| 1 | 1 | 1 |  |  |  |  |  |  |  |
| 2 | 1 | 2 | 1 |  |  |  |  |  |  |
| 3 | 1 | 3 | 3 | 1 |  |  |  |  |  |
| 4 | 1 | 4 | 6 | 4 | 1 |  |  |  |  |
| 5 | 1 | 5 | 10 | 10 | 5 | 1 |  |  |  |
| 6 | 1 | 6 | 15 | 20 | 15 | 6 | 1 |  |  |
| 7 | 1 | 7 | 21 | 35 | 35 | 21 | 7 | 1 |  |
| 8 | 1 | 8 | 28 | 56 | 70 | 56 | 28 | 8 | 1 |

Дача сдачи реферата:

Дата защиты реферата:

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Пономарев Д.А.

Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ветчинкин А.С.

Аннотация

Курсовой расчет предназначен для ознакомления студентов с процессом проектирования алгоритма управления динамическим объектом на примере водоизмещающего судна.

Summary

Course work is intended to familiarize students with the process of designing a dynamic object control algorithm using the example of a displacement vessel.

Оглавление

[Задание на курсовую работу 2](#_Toc530520962)

[Аннотация 3](#_Toc530520963)

[Summary 3](#_Toc530520964)

[Введение 5](#_Toc530520965)

[1. Математическое описание объекта управления 6](#_Toc530520966)

[2. Математическая формулировка цели управления 8](#_Toc530520967)

[3. Выбор и реализация метода решения оптимизационной задачи 9](#_Toc530520968)

[4. Анализ чувствительности реализованного метода 11](#_Toc530520969)

[Заключение 13](#_Toc530520970)

[Список литературы 14](#_Toc530520971)

[Приложение 15](#_Toc530520972)

Введение

Курсовой расчет предназначен для ознакомления с процессом проектирования алгоритма управления динамическим объектом на примере водоизмещающего судна.

Проектирование алгоритма управления состоит из следующих этапов:

- математическое описание объекта управления

- математическая формулировка цели управления

- выбор метода решения поставленной оптимизационной задачи

- оценка вариантов решения задачи

1. Математическое описание объекта управления

Динамика судна, как и любого физического тела, подчиняется второму закону Ньютона. Силы и моменты, действующие на судно, в свою очередь, описываются законами гидродинамики. Соотношения между кинематическими параметрами движения ( - угол рыскания,  - угловая скорость рыскания,  - угол дрейфа,  - угол перекладки руля) показаны на рисунке 1.

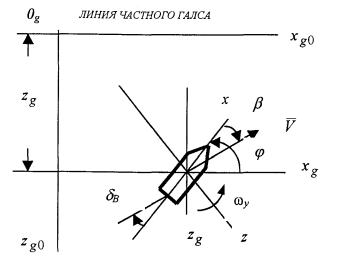


Рисунок 1 – соотношения между кинематическими параметрами

В общем случае, зависимость сил и моментов, действующих на судно от параметров движения носит нелинейный характер. Однако предположение о малых значениях угла дрейфа и угловой скорости рыскания и постоянстве линейной скорости движения судна позволяют линеаризовать эти зависимости и описать динамику в виде системы линейных дифференциальных уравнений относительно углов рыскания, дрейфа, угловой скорости рыскания, угла перекладки руля и одного нелинейного соотношения, отражающего тот факт, что руль не может поворачиваться на произвольный угол при произвольном сигнале управления. Для большинства современных судов максимальный угол перекладки руля равен 35°. Упомянутые соотношения, записанные относительно нормированного времени , имеют вид (1). При записи (1), кроме предположений о малости углов не учитывалось действие на судно ветро-волновых возмущений. т.е. математическая модель (1) соответствует движению судна на тихой воде.

 (1)

где:  - относительная скорость рыскания;  - угол дрейфа;  - угол перекладки руля.

Математическая модель судна в натуральном времени записывается в виде (2).

 (2)

Соотношение между параметрами (1) и (2) имеет вид (3).

 (3)

Значение нормирующей частоты  вычисляется по данным таблиц 1 и 2, содержащим варианты параметров математических моделей судов.

2. Математическая формулировка цели управления

При выполнении настоящей курсовой работы требуется спроектировать алгоритм управления рулем судна, который обеспечивает минимальное время устранения начального значения угла рыскания равного 10°.

3. Выбор и реализация метода решения оптимизационной задачи

Метод решения задачи выбран исходя из исходных данных варианта - метод стандартного полинома - биноминальные полином.

Подобно минимизации интегрального квадратичного функционала задача назначения заданного расположения собственных чисел системы управления (задача модального управления) не эквивалентна задаче максимального быстродействия. Однако и в этом случае возможно приближение к основной задаче за счет поиска соответствующего значения нормирующей частоты выбранного стандартного полинома.

Таким образом, задача модального управления рассматривается как задача поиска такого значения нормирующей частоты выбранного стандартного полинома, которая соответствует минимуму времени переходного процесса по углу рыскания. Эта задача является задачей одномерной оптимизации и может быть решена с помощью MATLAB функции FMINSEARCH.

Для обеспечения решения задачи FMINSEARCH должна ссылаться на функцию (fminsearch\_function), которая вычисляет время переходного процесса, соответствующее текущему значению нормирующей частоты стандартного полинома (с помощью функции calculate\_transition\_time). Для вычисления времени переходного процесса первоначально вычисляются коэффициенты стандартного полинома, соответствующие текущему значению нормирующей частоты (с помощью функции calculate\_system\_parameters). Затем вычисляются корни полученного полинома, которые рассматриваются в дальнейшем как желаемые корни замкнутой системы управления. Операция вычисления корней может быть выполнена с помощью MATLAB функции ROOTS. Затем по известной математической модели объекта управления и желаемым корням характеристического уравнения замкнутой системы вычисляются параметры линейного алгоритма управления. Последняя операция может быть выполнена с помощью MATLAB функции solve. На последнем этапе с помощью ODE45 вычисляется массив значений угла рыскания и путем обработки массива значений угла рыскания определяется время переходного процесса.

Код программы представлен в приложении.

Графики переходных процессов при частоте, соответствующей максимальному быстродействию, представлены на рисунке 2.

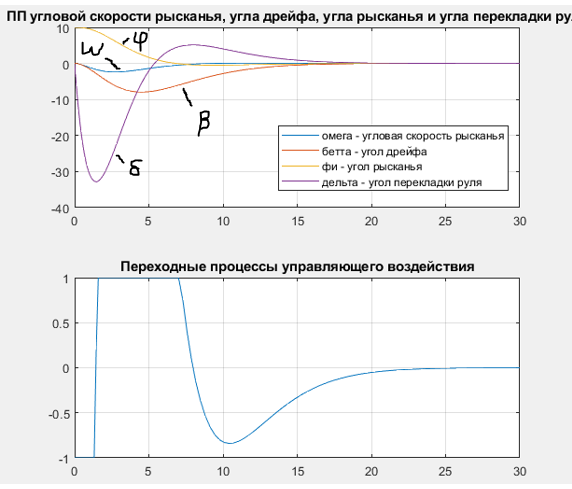


Рисунок 2 – Графики переходных процессов

Результаты выполнения программы, представлены на рисунке 3.

|  |
| --- |
| w = 0.5844  time = 6.0685 |

Рисунок 3 – Результаты выполнения работы программы

Таким образом, найдено значение нормирующей частоты выбранного стандартного полинома

w = 0.5844,

которое соответствует минимуму времени переходного процесса по углу рыскания

t = 6.0685

4. Анализ чувствительности реализованного метода

Поскольку задача проектирования решается относительно приближенной математической модели объекта управления, представляется целесообразным проанализировать чувствительность полученных различными методами алгоритмов управления к изменению параметров объекта управления. Эту операцию выполним методом моделирования.

Проанализируем чувствительность реализованного метода управления к изменению скорости хода объекта управления. Для этого проведем серию экспериментов при различных значениях скорости хода. Результаты опытов представлены в таблице 4 и на рисунке 4.

Таблица 4. Зависимость времени переходного процесса от скорости хода

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № опыта | 1 | 2 | 3 | 4 | 5 |
| Значение скорости хода относительно исходного | 0.6 | 0.8 | 1 | 1.2 | 1.4 |
| Время переходного процесса | 10.1141 | 7.5856 | 6.0685 | 5.0572 | 4.3346 |

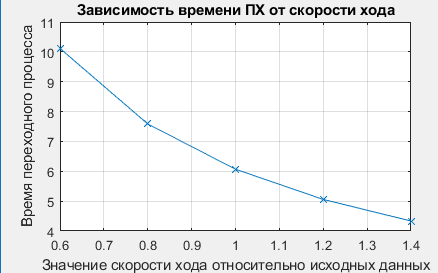


Рисунок 4 – Зависимость времени переходного процесса от скорости хода

Таким образом, время переходного процесса при небольших изменениях скорости хода обратно пропорционально скорости хода.

Проанализируем чувствительность реализованного метода управления к изменению длины по ватерлинии объекта управления. Для этого проведем серию экспериментов при различных значениях длины по ватерлинии. Результаты опытов представлены в таблице 5 и на рисунке 5.

Таблица 5. Зависимость времени переходного процесса от скорости хода

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № опыта | 1 | 2 | 3 | 4 | 5 |
| Значение длины по ватерлинии относительно исходного | 0.6 | 0.8 | 1 | 1.2 | 1.4 |
| Время переходного процесса, с | 3.6411 | 4.8548 | 6.0685 | 7.2822 | 8.4959 |

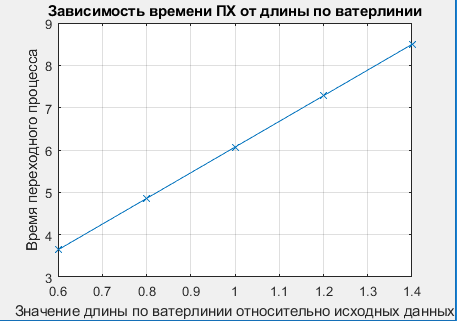


Рисунок 4 – Зависимость времени переходного процесса от длины по ватерлинии

Таким образом, время переходного процесса при небольших относительных изменениях длины по ватерлинии пропорционально длине по ватерлинии.

Вывод: значение параметров объекта управления очень сильно влияют на время переходного процесса. Так время переходного процесса при небольших изменениях скорости хода обратно пропорционально скорости хода, а также пропорционально длине по ватерлинии.

Заключение

В ходе работы был спроектирован алгоритм управления динамическим объектом на примере водоизмещающего судна.

Был выбран и реализован метод стандартного полинома (биноминального), который обеспечил минимальное время устранения начального значения угла рыскания равного 10°.

Минимальное время переходного процесса при исходных данных составило 6.0685 сек.

Также была проанализирована чувствительность основного показателя качества (переходного процесса) к изменению параметров математической модели объекта управления. Было определено, что время переходного процесса обратно пропорционально скорости хода и пропорционально длине по ватерлинии.

Сложность задач, решаемых на этапе проектирования, и реализации алгоритма управления не большая.

Список литературы

1. Методические указания к курсовому расчету «ПРОЕКТИРОВАНИЕ ОПТИМАЛЬНЫХ СИСТЕМ УПРАВЛЕНИЯ», Санкт–Петербург, 2013

2. Методические указания к лабораторным работам «ПРОЕКТИРОВАНИЕ ОПТИМАЛЬНЫХ СИСТЕМ УПРАВЛЕНИЯ», В. А. Зуев А. С. Ветчинкин, Санкт–Петербург, 2013

3. Конспект лекций по дисциплине «Проектирование оптимальных систем управления»

Приложение

Файл main.m

|  |
| --- |
| clc; clear; close all;  w0 = 0.3;  [w, time] = fminsearch('fminearch\_function', w0) |

Файл fminsearch\_function.m

|  |
| --- |
| function transitionTime = fminearch\_function(w0)  %@brief This function is used by fminsearch()  %@param w0 - base frequency  %@return transition time    % Step 1. Define global variable and some constans:  global K  x0 = [0 0 10 0];  maxTransitionTime = 30;    % Step 2. Calculate data for system and graph:  calculate\_system\_parameters(w0);  [t, x] = ode45('odefun', [0 maxTransitionTime], x0);  transitionTime = calculate\_transition\_time(t, x);    % Step 3. Show data:  fprintf("w0 = %f, t = %f, k = [%f, %f, %f, %f]\n", w0, transitionTime, K);    % Step 4. Show graphs:  subplot(2, 1, 1); plot(t, x(:, 1:4));  legend('омега - угловая скорость рысканья', ...  'бетта - угол дрейфа', ...  'фи - угол рысканья', ...  'дельта - угол перекладки руля')  subplot(2, 1, 2); plot(t, control\_impact(x'))  pause(0.1)  end |

Файл calculate\_system\_parameters.m

|  |
| --- |
| function calculate\_system\_parameters(w0)  %@brief This function calculate main global system parameters (A, B, C)  %@param w0 - base frequency    %Step 1. Define global variables  global A B K    %Step 2. Init data:  % variant 12.  % vessel veriant - 4  % Solution method - standard polynomial p.2.1  V0 = 6.17; L = 39; r21 = -0.69; r31 = 6.14; q21 = 1.22;  q31 = -3.12; s21 = -0.44; s31 = -3.1; W = V0/L; a11 = -r31\*W;  a12 = -q31\*W^2; a21 = -r21; a22 = -q21\*W; b11 = -s31\*W^2; b21 = -s21\*W;    %Step 3. Calculate K, then A and B:  syms k1 k2 k3 k4  characteristicPolynomial = ...  [1; ... %s^4  k4 - a22 - a11; ... %s^3  a11\*a22 - a12\*a21 - a11\*k4 - a22\*k4 + b11\*k1 + b21\*k2; ... %s^2  b11\*k3 - a11\*b21\*k2 + a12\*b21\*k1 + a21\*b11\*k2 - a22\*b11\*k1 + a11\*a22\*k4 - a12\*a21\*k4; ... %s^1  a12\*b21\*k3 - a22\*b11\*k3];    binomialPolynomial = ...  [1\*w0^0; ... %s^4  4\*w0^1; ... %s^3  6\*w0^2; ... %s^2  4\*w0^3; ... %s^1  1\*w0^4]; %s^0    [sk1, sk2, sk3, sk4] = solve(...  characteristicPolynomial(1) == binomialPolynomial(1), ...  characteristicPolynomial(2) == binomialPolynomial(2), ...  characteristicPolynomial(3) == binomialPolynomial(3), ...  characteristicPolynomial(4) == binomialPolynomial(4), ...  characteristicPolynomial(5) == binomialPolynomial(5));    K = [double(sk1), double(sk2), double(sk3), double(sk4)];  A = [a11 a12 0 b11; % угловая скорость рысканья омега\_у  a21 a22 0 b21; % угол дрейфа бетта  1 0 0 0; % угол рысканья фи  -K(1) -K(2) -K(3) -K(4)]; % угол перекладки руля дельта\_в (не больше 35 градусов)  B = [0 0 0 1]; % коэффицинты перед управляющим воздействием    % Step 4. Check roots. All roots should be equal to w0. It's not obligatory  %roots(poly(A))  end |

Файл calculate\_transition\_time.m

|  |
| --- |
| function transitionTime = calculate\_transition\_time(t, x)  %@brief Calculate transition time  %@param t - time vector with size (pointsAmount, 1)  %@param x - state variables matrix with size(pointsAmount, stateVariabelsAmount)  %@return transition time scalar    pointsAmount = length(t);  transitionTime = 0;  initialDeviation = 10;  for i = pointsAmount : -1 : 1  if abs(x(i, 3)) > 0.05\*initialDeviation  transitionTime = t(i);  break;  end  end  end |

Файл control\_impact.m

|  |
| --- |
| function u = control\_impact(x)  %@brief Calculate control impact u = -K1\*x1 - K2\*x2 - K3\*x3 - K4\*x4  %@param x - state variables with size (pointsAmount, stateVariablesAmount)  %@return control impact with size(pointsAmount, 1)  %@note u <= |Umax|  global K  Umax = 1;  u = -K(1).\*x(1, :) - K(2).\*x(2, :) - K(3).\*x(3, :) - K(4).\*x(4, :);  for i = 1:length(u)  if u(i) > Umax  u(i) = Umax;  elseif u(i) < -Umax  u(i) = -Umax;  end  end  end |

Файл odefun.m

|  |
| --- |
| function dxdt = odefun(t, x)  %@brief This function is used by ode45()  %@param t - time vector with size (pointsAmount, 1)  %@param x - state variables matrix with size(pointsAmount, stateVariabelsAmount)  %@return left parts of equations  global A  rudderAngleMax = 35;  if x(4) > rudderAngleMax  x(4) = rudderAngleMax;  elseif x(4) < -rudderAngleMax  x(4) = -rudderAngleMax;  end  dxdt = [A(1,1)\*x(1) + A(1,2)\*x(2) + A(1,3)\*x(3) + A(1,4)\*x(4); ...  A(2,1)\*x(1) + A(2,2)\*x(2) + A(2,3)\*x(3) + A(2,4)\*x(4); ...  A(3,1)\*x(1) + A(3,2)\*x(2) + A(3,3)\*x(3) + A(3,4)\*x(4); ...  A(4,1)\*x(1) + A(4,2)\*x(2) + A(4,3)\*x(3) + A(4,4)\*x(4)];  end |