МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «КИЇВСЬКИЙ АВІАЦІЙНИЙ ІНСТИТУТ»
ФАКУЛЬТЕТ АЕРОНАВІГАЦІЇ, ЕЛЕКТРОНІКИ ТА ТЕЛЕКОМУНІКАЦІЙ
КАФЕДРА АВІОНІКИ ТА СИСТЕМ УПРАВЛІННЯ

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
_____Олена ТАЧИНІНА
«_____» _____2025 р.

КВАЛІФІКАЦІЙНА РОБОТА

(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ
«БАКАЛАВР»

Тема: «Система управління протезом нижньої кінцівки»

Виконавець: студент групи Ба-151-21-2-СУ _____ Кіріл ПРОЦЕНКО

Керівник: к.т.н., доцент _____Антоніна КЛІПА

Нормоконтролер: _____ Микола ДИВНИЧ

Київ 2025

MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE
STATE UNIVERSITY “KYIV AVIATION INSTITUTE”
FACULTY OF AIR NAVIGATION, ELECTRONICS AND TELECOMMUNICATIONS
DEPARTMENT OF AVIONICS AND CONTROL SYSTEMS

APPROVED FOR DEFENCE
Head of the Department
_____ Olena TACHYNINA
“____” _____ 2025

QUALIFICATION PAPER

(EXPLANATORY NOTE)

FOR THE ACADEMIC DEGREE OF BACHELOR

Title: **“**Lower Limb Prosthesis Control System”

Submitted by: student of group Ба-151-21-2-СУ _____Kiril PROTSENKO

Supervisor: PhD, associate professor _____Antonina KLIPA

Standards inspector: _____ Mykola DYVNYCH

Kyiv 2025

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет аеронавігації, електроніки та телекомунікацій

Кафедра авіоніки та систем управління

Спеціальність: 151 «Автоматизація та комп'ютерно-інтегровані технології»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Олена ТАЧИНІНА

«_____»_____2025 р.

ЗАВДАННЯ

на виконання кваліфікаційної роботи

Проценка Кіріла Юрійовича

1. Тема кваліфікаційної роботи «Lower Limb Prosthesis Control System» затверджена наказом ректора від «20» березня 2025 р. № 429/ст.

2. Термін виконання роботи: з 19.05.2025 по 14.06.2025.

3. Вихідні дані роботи:

Параметри користувача:

s	Параметр	Значення	Одиниця вимірювання
1	Висота тіла	1,8	м
2	Вага тіла	70	кг
3	Ввисота бедра	0,4	м
4	Вага бедра	2	кг
5	Висота гомілки	0,4	м
6	Вага гомілки	2	кг

4. Зміст пояснювальної записки:

У розділі 1 досліджуються підходи до створення систем керування для протезних пристроїв. У розділі 2 розглядаються синтез блоків керування та моделювання основних компонентів. У розділі 3 розглядається моделювання компонентів

системи керування з попередньо визначеними параметрами. У розділі 4 розглядається синтезована система керування та аналіз результатів симуляції.

5. Перелік обов’язкового ілюстративного матеріалу:

1) Структурна схема протеза нижньої кінцівки.

2) Схеми розроблених блоків керування.

3) Електричні та механічні схеми компонентів протеза.

6. Календарний план-графік

№ пор.	Завдання	Термін виконання	Відмітка про виконання
1	Формування моделей для симуляції вхідних даних.	19.05.2025-22.05.2025	
2	Моделювання нечіткого логічного контролера	23.05.2025-24.05.2025	
3	Моделювання блоку керування SCADE.	25.05.2025-26.05.2025	
4	Моделювання серводвигуна постійного струму з керованим якорем.	26.05.2025-28.05.2025	
5	Моделювання протеза нижньої частини ноги.	28.05.2025-29.05.2025	
6	Моделювання системи керування протезом та її аналіз.	29.05.2025-31.05.2025	
7	Аналіз результатів	31.05.2025-02.06.2025	

7. Дата видачі завдання: «05» травня 2025 р.

Керівник кваліфікаційної роботи _____ Антоніна КЛІПА
(підпис керівника)

Завдання прийняв до виконання _____ Кіріл ПРОЦЕНКО
(підпис випускника)

NATIONAL AVIATION UNIVERSITY

Faculty of Air Navigation, Electronics and Telecommunications

Department of Avionics and Control Systems

Specialty: 151 “Automation and Computer-integrated Technologies”

APPROVED BY

Head of the Department

_____ Olena TACHYNINA

“_____”_____2025

Qualification Paper Assignment for Graduate Student

Protsenko Kiril Yuriiovych

1. The qualification paper title “Lower Limb Prosthesis Control System” was approved by the Rector’s order of March 20, 2025 № 429/ст.

2. The paper to be completed between: 19.05.2025 and 14.06.2025

3. Initial data for the paper:

Parameters of the user:

№	Parameter	Value	Unit of measurement
1	Height	1,8	m
2	Weight	70	kg
3	Thigh length	0,4	m
4	Thigh mass	2	kg
5	Shank length	0,4	m
6	Shank mass	2	kg

4. The content of the explanatory note:

Chapter 1 explores approaches regarding creating control systems for prosthetic devices. Chapter 2 includes synthesizing of control units and modelling of main components. Chapter 3 contains simulation of control system components with predefined parameters. Chapter 4 includes synthesized control system discussion and its resultant simulation analysis.

5. The list of mandatory illustrations:

1) Structural diagram of lower limb prosthetic device.

2) Designed control units' diagrams.

3) Electrical and mechanical diagrams of prosthetic device components.

6. Timetable

№	Assignment	Dates of completion	Completion mark
1	Forming models for input data simulation.	19.05.2025-22.05.2025	
2	Modelling fuzzy logic controller.	23.05.2025-24.05.2025	
3	Modelling SCADE control unit.	25.05.2025-26.05.2025	
4	Modelling armature-controlled DC servomotor.	26.05.2025-28.05.2025	
5	Modelling lower knee prosthetic leg	28.05.2025-29.05.2025	
6	Simulation of prosthesis control system and its analysis.	29.05.2025-31.05.2025	
7	Results analysis.	31.05.2025-02.06.2025	

7. Assignment issue date: "05" May 2025

Qualification paper supervisor _____ Antonina KLIPA
(the supervisor's signature)

Issued task accepted _____ Kiril PROTSENKO
(the graduate student's signature)

ABSTRACT

Text part of the work: 74 pages, 35 figures, 11 tables, 17 references.

Object of the study - lower limb prosthetics.

Subject of the study - lower limb prosthesis control system.

Purpose of the work - synthesizing lower limb prosthesis control system and its main components.

Research methods - theory of automatic control of moving vehicles, mathematical modelling of mechanical and electrical systems, MATLAB, SIMULINK and SCADE implementation.

The work studies the control system of a lower limb prosthesis for providing walking activity for a user with 170 cm and 80 kg parameters. It explores principle of operation of fuzzy logic controller, implementation of additional control unit developed in SCADE environment according to the standard for medical software, synthesizing armature-controlled DC servomotor replicating knee joint with the aim to achieve fast and precise response and lower knee model of the prosthetic leg to provide comfortable walking activity. The developed approach can be applied during synthesizing lower limb prosthesis control system for providing walking activity for users with different body parameters of height and weight.

CONTROL SYSTEM, SIMULATION, FUZZY LOGIC, PROSTHESIS, LOWER LIMB, PROSTHETIC DEVICE, KNEE JOINT, LOWER KNEE, MATLAB, SCADE.

CONTENTS

LIST OF ABBREVIATIONS

Abbreviation	Meaning
FLC	Fuzzy Logic Controller
MEMS	Micro-Electromechanical Systems
IMU	Inertial Measurement Unit
AC	Alternating Current
DC	Direct Current
MCC	Multiple Decision Coverage
MC/DC	Multiple Condition/Decision Coverage
SCADE	Safety Critical Application Development Environment
FSFIA	Full-State Feedback with Integral Action
IDE	Integrated Development Requirement
LQR	Linear Quadratic Regulator

INTRODUCTION

Prostheses are crucial devices that help people who suffer from limb amputation to regain mobility and basic functions of a lost limb. Therefore, the choice of prosthesis and its design is critically important, since a well-fitted device supports specific requirements concerning the activity of a patient, reduces the risk of falls and injuries caused by it. Prosthetic devices should also be designed with the aim to be suited to the future needs of a patient, allowing easier adaptation to changes in the body mass of a user, mobility, and activity level.

In recent decades, the field of prosthetics has gone through a number of changes, driven by advances in biomedical engineering and other practical fields. With these changes, the role of control systems remained to bridge the gap between human intentions and artificial movement. As prosthetic devices were developed from simple mechanical devices with no feedback to bionic ones, whose feedback is the contraction of muscles, the role of control systems now lies in achieving natural, precise, and responsive motion.

Considering the time in which the qualification paper is written, questions about developing and providing prostheses arise more frequently, since nowadays people suffer from limb losses caused not only by diseases or accidents, but also by war actions. That is why the prosthetics field should be given more attention.

Taking into account current demands for prostheses not only for the military, but for civilians as well, and the progress achieved in prosthesis development, we can conclude that this field is important in providing more affordable prosthetic devices that will help patients return to their everyday life.

The qualification paper explores approaches that are used in control systems to develop lower-limb prosthesis for a patient whose weight is 70 kg and height is 180 cm to provide walking activity, with the assumption that the person has an amputated lower limb with the thigh remaining.

To study the control system of lower limb prosthesis we will use the mathematical modelling for recreating the main parts of the prosthetic leg and its motion, control system approaches for their analysis and synthesis to study its structure and interaction between

components for further developing control system using software based on the modelling and control system analysis results. For the simulation data predefined in the task and technical characteristics of real devices will be used.

CHAPTER 1
PROSTHETIC DEVICES CHARACTERISTICS

1.1. Classification of prosthetic devices

Since there is no universal solution to prosthesis development due to the individual injuries and needs of each patient, prosthetic devices are distinguished by:

1) Amputation level:

a) above-knee or transtibial prostheses, which serve to replace the leg above the knee with a knee joint;

b) below-knee or transfemoral prostheses, which are used in cases of a remaining knee joint;

c) hemipelvectomy prostheses used in cases of amputations at or above hip joints and include prosthetic hip and leg.

2) Functions:

a) Passive prostheses may not have joints that serve to replicate human movement;

b) Body-powered prostheses use a movement of a patient as the source of data;

c) motorized or myoelectric prostheses, which have an external power source such as batteries and use sensors and motors for mimicking the activity of a patient.

3) Activity level:

a) Basic prostheses that are applied for limited mobility or short distances;

b) Dynamic response artificial limbs are used for providing a normal lifestyle for active users;

c) sports-specific prostheses designed considering demands for specific activity, usually made of light-weight, spring-like material;

4) By principle of control:

a) microprocessor knees that regulate movement activity in real-time;

b) bionic legs that use sensors to adapt to its user's mobility. In the last few years, AI has been integrated to analyze the activity of a user and adapt the leg to it and external disturbances.

1.2. Prosthetic limb control systems structure

The control system of an above-knee artificial limb is responsible for interpreting input signals coming from sensors, forming desired control signals, and driving actuators to replicate a natural gait cycle. A general control system consists of the following components:

1) Sensing system. To this group can be included:

a) kinematic sensors, such as inertial measurement units or joint encoders, that provide necessary data about limb position, joint angles, and velocities;

b) Kinetic sensors that detect surface reaction forces and body mass distribution;

c) user interface sensors, such as electromyography sensors, to predict intentions of a user.

2) Control unit. The purpose of this component is to form the control signal based on the data from sensors and provide output to the actuator. The elements used in a prosthetic leg are following:

a) Impedance controllers that regulate stiffness and damping of joints for achieving the desired interaction with ground.

b) Machine Learning-Based controllers improve the performance of a prosthetic leg by predicting the intent of a user based on user-specific information.

c) Finite-state machine controllers that distinguish the walking activity into discrete phases and form control signals depending on a certain phase.

d) Model predictive controllers optimize the behavior of the artificial limb based on the prediction data.

3) Actuation systems are in charge of converting control signals into torque at a knee joint. Among these systems, the next may be highlighted:

a) Electric motors that provide torque at the joint to turn the prosthetic leg.

b) Transmission mechanisms, such as gearboxes, belts, or harmonic drives, that convert motor torque into mechanical movement.

c) Braking elements to decelerate and keep the leg in a stance phase.

4) Communication interface that gives a possibility to tune control parameters, acquire data for gait analysis, and monitor the state of the system remotely.

5) Integration and safety measures. Safety mechanisms such as fail-state shutdown, sensor redundancy are crucial since they prevent the control system from failure, and therefore, the patient from getting injured.

For this qualification work, we will consider the structure of a prosthetic leg shown in fig. 1.2.1.

The structure of the developed prosthetic leg consists of the following components:

1) Sensors act as the input of our system and include information about the angular displacement of the thigh from the IMU sensor and the load acting on the leg from the load cell that is transferred to the microcontroller in the form of crisp values.

2) Microcontroller element includes two systems: Fuzzy logic controller created by means of MATLAB Fuzzy logic toolbox, and SCADE algorithm formed in ANSYS SCADE software. Fuzzy logic controller contains a rule base formed on the information from walking activity of a user, including membership functions describing the motion of upper link of a leg and the weight acting on the sensor. Both of the systems are connected in parallel since the SCADE algorithm was developed as an alternative to Fuzzy logic controller (FLC) and must handle the processing of sensor data in case of failure of FLC. Both of the controllers take as input data from sensors and process it that resulting in crisp output that is given to the DC servomotor.

3) Armature-controlled DC servomotor acts as a knee joint replication with a gearhead connected, which accepts the signal from the microcontroller and rotates the shaft accordingly.

4) The role of the passive ankle-foot model is to provide a stable interaction of a prosthetic leg with a ground for safe and stable movement of a patient.

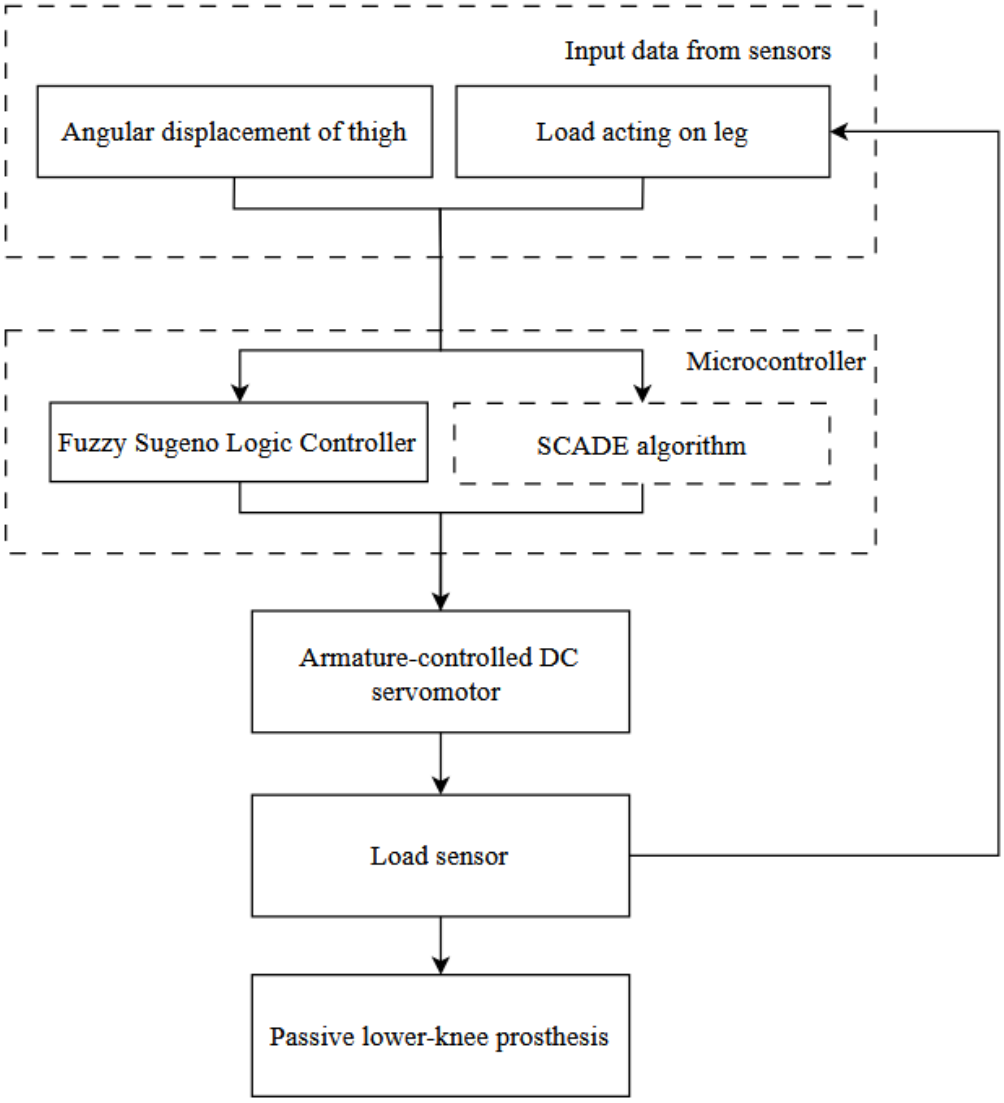

Fig. 1.2.1. Prosthetic leg structural diagram

1.3. Lower limb motion

For real-life applications, inertial measurement units based on Micro-electromechanical Systems (MEMS) components can be used to detect the motion of a thigh and shank. To capture the movement of a prosthesis, Inertial Measurement Unit (IMU) is placed on a thigh or a knee joint levels.

An IMU is a device that captures the dynamic motion of an object, acquiring data about its position in space, angular velocity, and linear acceleration. In general, inertial measurement units consist of the following components:

1) accelerometers that measure linear acceleration along three orthogonal axes;

2) Gyroscopes measure the angular velocity of an artificial limb in space, which can be later integrated to determine the orientation of the leg.

Advantages of using IMUs are their miniature sizes and light weight, allowing them to be integrated into prosthetic systems. Also, their low power consumption is suitable for battery-powered systems. Their low-latency feedback is a crucial advantage for closed-loop systems.

For this qualification paper, consider the diagram of a human leg in space depicted in fig. 1.3.1.

In fig. 1.3.1, θ_1 and θ_2 represent angular displacements of thigh and shank, respectively. The task of the measurement unit is to detect these motions by measuring the angular velocity of the upper and lower links of the artificial limb and integrating them to determine their positions in space.

Since it is theoretically difficult to fully simulate the motion of a human leg, simplified models are used in the form of differential equations or Simulink models. Modelling human leg motion can be difficult due to the complexity of human muscles and joint interactions, and nonlinear dynamics. One of the approaches to approximate lower-limb biomechanics is to use a double pendulum model.

The human leg during gait can be effectively interpreted as a two-link mechanism representing the thigh and shank. Hip and knee joints correspond to the pivot points of a

double pendulum model. It allows to examine a leg movement, avoiding anatomical complexity, offering a suitable approximation.

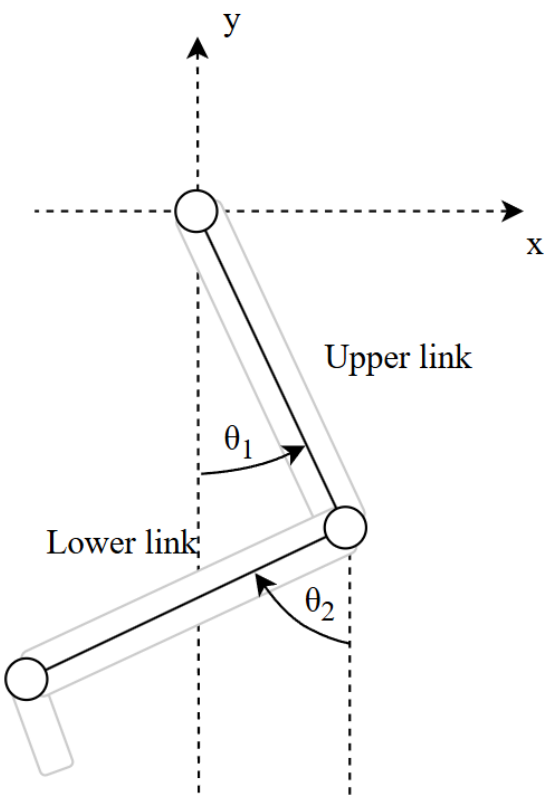

Fig. 1.3.1. Diagram of a human leg

One of the main characteristics of the double pendulum system is its nonlinear behavior, which reflects the complexity of human movement. This property allows to investigate of instability and the influence of disturbances acting during motion.

The double pendulum system can also include effects of inertia and gravity, and momentum between limb segments, which can be used for more accurate replication of natural motion. By assigning parameters of a human leg, such as masses and lengths of thigh and shank, human motion can be modelled for every patient, taking into account their height and weight.

Thus, the pendulum system offers a balance between analytical simplicity and functional realism. Its equations of motion can be derived using Lagrangian mechanics, and the model will remain suitable for real-time simulations.

1.4. Ground forces detection

Force is determined by Newton's second law, which states, that the force acting on a body equal mass of the body multiplied by acceleration. However, devices that use this expression are not commonly used. Instead, most sensors detect load based on deformation or strain in the sensing element caused by applied forces, then convert this change into an electrical signal.

A load cell is a type of force sensor designed to measure a weight or load applied to an object. Typically, their principle of operation is based on strain gauge technology. When force is applied, the structure of the sensor deforms. Strain gauges also deform and change resistance. This change is measured using a Wheatstone bridge circuit depicted in fig. 1.4.1. [1].

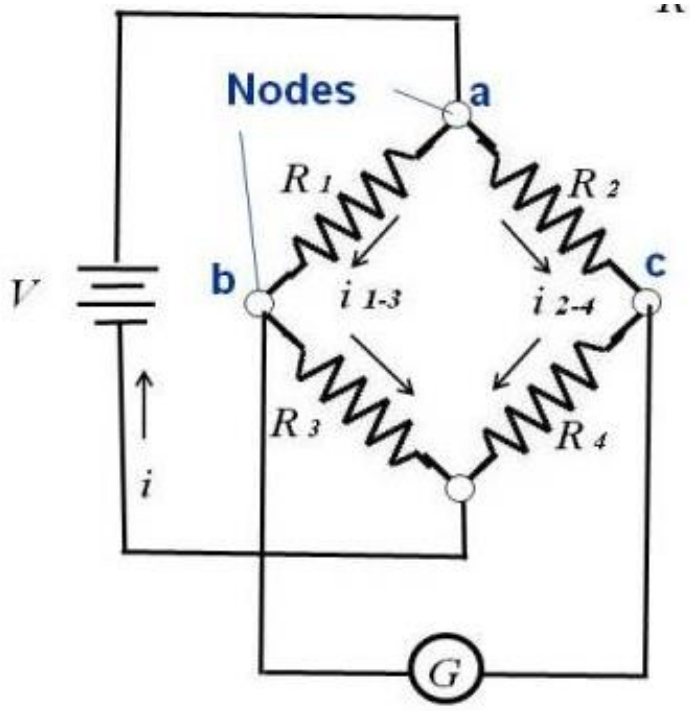

Fig. 1.4.1. Wheatstone bridge circuit

To measure the difference in resistance, the following expression is used:

$$\frac{G}{V} = \frac{(R2 * R3) - (R1 * R4)}{(R1 + R3)(R2 - R4)}.$$

The principle of operation of the bridge circuit lies in two strain gauges connected to the mode, and the output is located inWheatstone bridge as *R1* and *R2*; Equal resistors

are *R3* and *R4*. If the gauge is subjected to an external force, the resistance in *R1* and *R2* changes proportionally, but because the potentiometer measures only the difference between these two resistors, reading remains the same [1].

Load cells are distinguished into:

1) Strain gauge load cell

2) Hydraulic and pneumatic load cells that use fluid and compressed air to measure weight, accordingly. Less accurate compared to strain ones.

4) Capacitive load cells use changes in capacitance to detect applied force.

5) Piezoelectric ones use crystals that generate a charge when subjected to a load.

Force sensors are divided into:

1) Tactile sensors that provide force distribution over a surface

2) Torque sensors that measure torques along all axes.

3) Fiber Bragg Grating and Optical sensors use changes in light wavelength or intensity to determine force, accordingly.

In prosthetic devices, load cells are usually placed in a foot or socket to detect surface reaction force.

1.5. Fuzzy controllers for lower prosthetic limb

The purpose of modern prosthetic devices is not only to restore basic mobility required for daily activity, but also to approximate the adaptability and intuition of natural human motion, which is the significant control challenge since body motion is nonlinear and impacted by great number of uncertain factors as terrain, walking speed, or human intent.

It is hard to deal with uncertainty using classic control system techniques, since they rely on accurate mathematical models. Therefore, fuzzy control was introduced to the world. It acts as an alternative by giving the possibility to approximate the human motion for the control systems, providing the control in the form of terms that are used by humans on a daily basis.

Fuzzy logic control is the system's control through symbolic expressions. Since people don’t operate with strict data on a daily basis, this controller uses a set of expressions that contain strict information about certain parameters of the system. By interpreting the data obtained from sensors, such as acceleration, force, and speed, through the fuzzy logic controller, a smooth, responsive output can be generated that aligns with the intentions of a patient, even if it is difficult to express the intent explicitly.

The schematic diagram of the fuzzy logic controller is given in fig. 1.5.1.

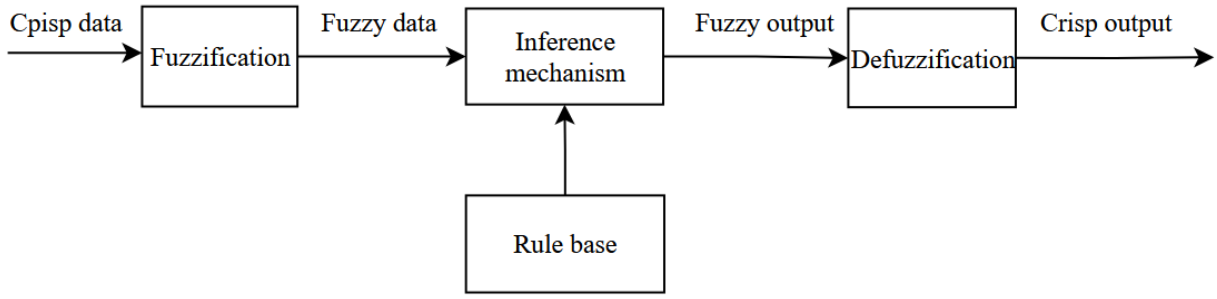

Fig. 1.5.1. Fuzzy logic controller structural diagram

The process of fuzzy control consists of the following stages:

1) Fuzzification stage is used to convert the strict data that comes as input to the controller into fuzzy information.

2) Inference mechanism uses data predefined in the rule base block to outcoming the desired fuzzy data as the output of the controller.

3) The goal of defuzzification is the transform the fuzzy data from the inference mechanism to the strict data for the system to control. There are several widespread methods for defuzzification, such as the center of area method,

4) Rule base contains all the necessary information describing the behavior of the system.

The necessary definitions used in describing the fuzzy control are following:

1) To describe the crisp data that enters into the controller, the definition of universe of discourse U_i is used. In general, they are expressed as an interval or set of real numbers.

2) Linguistic variables are the symbolic expressions that are used to describe the inputs and outputs of the fuzzy system. Variables defining the input and output of the system are denoted as $\tilde{u}$ and $\tilde{y}$, respectively.

3) Linguistic values describe the characteristics of the linguistic variables. Generally, they are named as “positive large”, “positive small”, “negative small”, etc.

4) Linguistic rules are a set of action rules which characterize mapping of inputs to the outputs of the fuzzy system in the “If-Then” form.

5) Membership function $\mu_A(x)$ determines how strongly the input data belongs to the specific fuzzy set. In the sets, each of the elements is mapped to [0,1] by the function. An example of the membership function is the triangular one, which is used in the paper and shown in fig. 1.5.2.

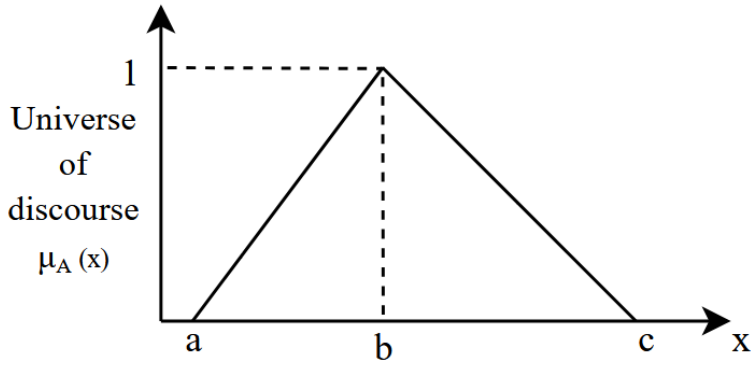

Fig. 1.5.2. Graphical representation of the triangular membership method

The mathematical description of the triangular membership method is given below:

$$\mu_A(x) = \begin{cases} 0 & \text{if } x \le a \\ \dfrac{x-a}{b-a} & \text{if } a \le x \le b \\ \dfrac{c-x}{c-b} & \text{if } b \le x \le c \\ 0 & \text{if } x \ge c \end{cases}.$$

There are several widespread membership functions, such as triangular, Gaussian, and trapezoidal ones, that are typically used.

1.6. Fuzzy Mamdani and Sugeno inference systems

Mamdani inference systems are the most widespread inference systems used. The output result is the membership function. The general form of the linguistic rule is following:

$$\text{if } x \text{ is } A \text{ and } y \text{ is } B \text{ then } z \text{ is } C,$$

where C is the membership function describing the output data.

Mamdani system is applicable to both systems with single and multiple outputs.

Sugeno is similar to the Mamdani, but it carries out the output crisp data in a different way: the output data has the form of linear functions or constant values. The general rule for the fuzzy system has the form:

$$\text{if } x \text{ is } A \text{ and } y \text{ is } B \text{ then } z = f(x, y),$$

where A and B are fuzzy sets,

$z = f(x, y)$ is a linear function.

If z is a first-order polynomial, then the system is called a first-order Sugeno system. For a constant output, Sugeno system is the zero-order one. The zero-order Sugeno system can be also considered as the case of Mamdani singleton output membership function.

The output result of each rule defined by the firing strength ω_i.The output of the fuzzy Sugeno system is the weighted average value of all rule outputs:

$$Output\ result = \frac{\sum_{i=1}^{N} \omega_i z_i}{\sum_{i=1}^{N} \omega_i}.$$

The drawback of Sugeno system is that it can be applied only to multiple-input single-output systems, but it is less complex in terms of processing data since the defuzzification stage is absent due to a mathematical description of the output.

1.7. Certified standards for medical devices and software

Before designing any software, we must ensure that our device corresponds not only to the required specifications given by the customer or supervisor, but also to certified standards. In the aviation field, standards DO-178C and DO-178B are used to develop systems. For automotive one ISO 26262 establishes the necessary requirements. Regarding devices used in medicine, IEC 62304 is widely used. Any software that is designed for these devices or represents the device itself should correspond to the standard specifications. Firstly, for any medical device safety class should be given. According to the standard, there are 3 safety classes by which software must be classified:

1) A class – software cannot lead to a dangerous situation, or it does not result in unacceptable risk.

2) B class – software can lead to a dangerous situation in which non-serious harm to health occurs.

3) C class – software can lead to a dangerous situation that results in serious harm to health or death.

The algorithm used to determine safety classes for the software is shown in fig. 1.7.1. [2].

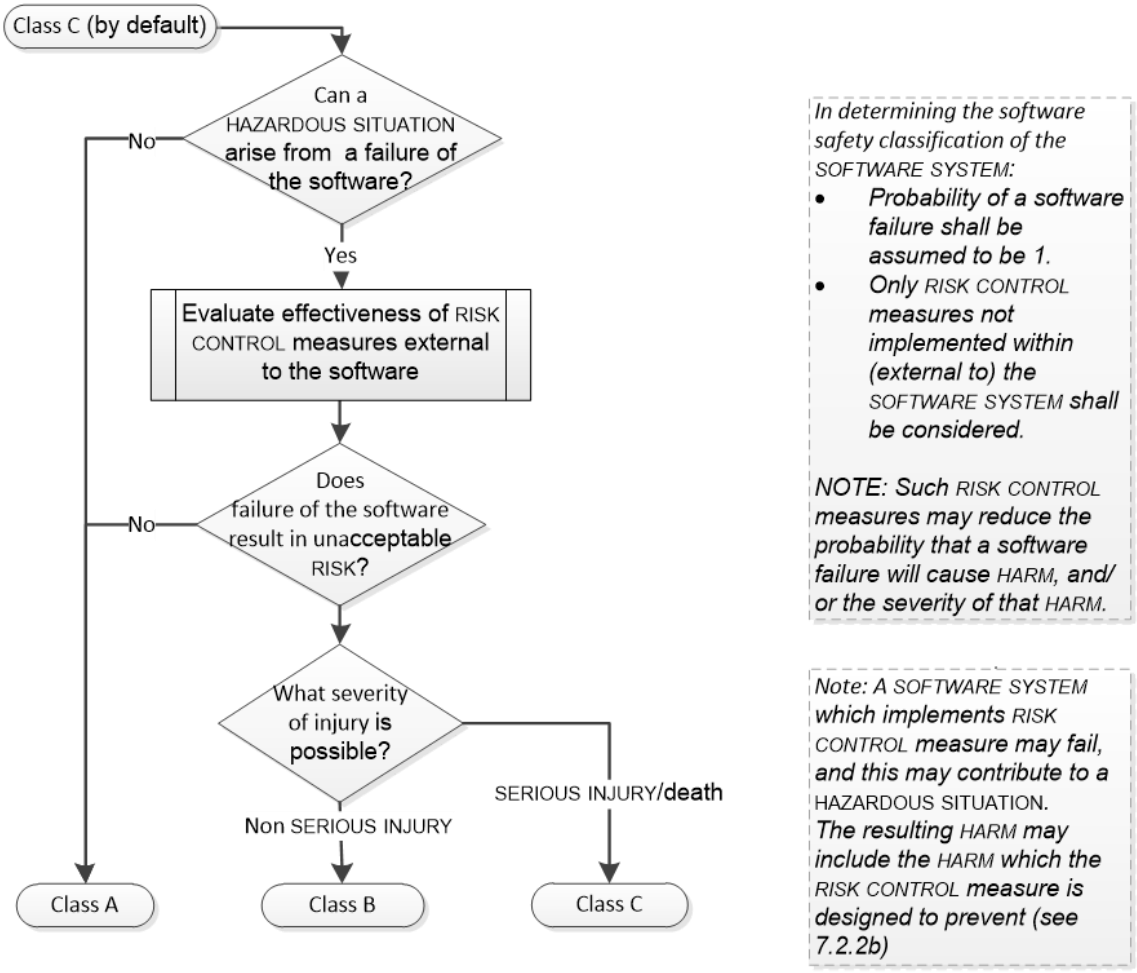

Fig. 1.7.1. Algorithm for classification of medical devices

According to the algorithm, the main criteria for classification are unacceptable risk and severity of injury. To classify the software for artificial prosthetic lower limb, we need to take into account that the leg is designed for a patient with upper-knee amputation, has to support the patient during the stance phase, and give him the possibility to walk.

Any failure of the device will lead to a fall that can result in serious injury or death. Therefore, the safety class for our device is C.

For C and B safety classes, it is necessary to provide additional risk control measures to prevent the failure of the system. To achieve this, we will implement an additional control unit to ensure that the failure of the main controller will not lead to the failure of the whole control system. For the realization of an additional control unit, we will use SCADE environment.

1.8. System design by means of SCADE software

SCADE (Safety Critical Application Development Environment) is the software designed for creating certified control systems with the possibility to form them visually. SCADE is used in aerospace, railways, nuclear energy fields, and other fields that require a significant level of safety and resistance to failures. Its key features are the next:

1) SCADE allows to creation of control systems in the form of block diagrams. It simplifies complex control logic development and makes to verify the correct work of a system during the design process.

2) The programming language of SCADE is Lustre, a synchronous dataflow language specifically developed for real-time systems.

3) SCADE provides tools for testing the dynamics of a system. Using them, we can form and observe different cases, monitor responses, and validate the control logic without running on real-life hardware.

4) With implemented verification instruments, SCADE allows to prove mathematically the correct work of control logic, increasing a system's stability and safety.

5) One of the main features of SCADE is the possibility to automatically generate C code directly from the model. After its generation, this code can be integrated into embedded systems.

6) Integration with other systems. SCADE software can be used to work with external software and hardware using standard interfaces. With this feature, SCADE can be integrated with sensors and actuators of the prosthetic system.

For our qualification work, the model of the control unit will be developed in SCADE. After built-in testing of the system in the environment, it will be integrated into SIMULINK to operate with a fuzzy logic controller in parallel to ensure that in case of failure of one of the systems, another one can provide stable control without failure of the whole system.

1.9. Software testing methods

The important part of each control system development is the testing stage. With the aim of implementing a control system using hardware, we have to be sure first that the software part of the control system operates correctly, or in case of the presence of errors, to eliminate errors. If errors are left undetected, they can probably cause malfunction of the hardware, which will lead to failures of the system. To prevent it, a number of test methods were developed, and the most widespread of which are the following:

1) Multiple Condition coverage (MCC). The essence of this method lies in testing all possible combinations of conditions. It is performed by checking all possible combinations for true and false states. Regarding decisions, each of them is tested separately at least once.

2) Path/branch coverage testing is used to check that every possible path or branch of the control system is executed at least once, respectively. It is useful to determine errors at early stages, but this method becomes less efficient when checking complex systems. Also, these methods may not cover all of the conditions present in the system.

3) Model-Based Testing. This approach uses models to generate test cases, based on which the actual system is tested. The created model represents the full or partial expected dynamics of the system. They can be created using the SIMULINK environment or SCADE Test Suite. This method is convenient for complex systems and those

requirements of which may change during the design process. Another advantage is its automatic generation of test cases.

5) Modified Condition/Decision Coverage MC/DC (MC/DC) is a testing approach widely used in safety-critical systems. Many of the standards require MC/DC to be used for testing critical software, for example, DO-178C and DO-178B software development guidance, automotive standard ISO 26262, and the medical devices standard IEC 62304. This method ensures that:

a) Every decision in the model has been invoked at least once.

b) Every condition in the model has been invoked at least once.

c) Each condition has affected the decision's outcome independently.

Advantages of this test approach include:

a) Detection of hidden logical flows without executing every possible path.

b) It does not require full coverage of all inputs and outputs.

In this qualification paper, we will use MC/DC approach to test the control unit developed in SCADE environment.

1.10. Knee joint replication

The human knee joint is a complex biological mechanism consisting of numerous parts that form together a component that provides stable human motion. This is why the question about replicating the knee joint using mechanical parts arises in the field of control systems. Since it is not sufficient to only mimic human motion, but also provide a fast and accurate response of the knee joint to this signal. Another question that comes to mind is the choice of mechanism that will replicate the knee joint. There 3 widespread solutions to this problem:

1) Servomotors. This type of motor is distinguished from usual motors since they are designed to respond to control signals and adjust their output, making them a universal choice when it comes to providing fast and exact control.

A principle of operation of a servomotor lies in a closed-loop system, which includes a direct current (DC)/alternating current (AC) motor, a position sensor, usually

an encoder or potentiometer, a control circuit, and a gearbox if torque adjustment is required. The control system receives a target position signal, then it is compared to the actual current position, and adjusts the motor's motion accordingly to minimize the error.

The servomotors are divided into:

1) AC Servomotors**.** These motors operate in the same manner as DC ones, but the difference signal of the closed-loop system is processed and used to adjust the frequency, phase, and amplitude of the supplied AC power. For synchronous AC motors, torque is generated when the magnetic field of the stator interacts with the magnetic field of the rotor and their frequency are the same. In asynchronous ones, the rotor lags the stator frequency.

Armature-controlled DC servomotor. The principle of operation is based on the proportionality of torque developed by the motor and armature current, which in turn is impacted by the armature voltage. Increasing armature voltage leads to a greater torque and vice versa.

The advantages of applying a DC servomotor include the next:

a) Since there is no need to regulate the magnetic field of the rotor, the electrical circuit becomes simpler to design.

b) Lower inductance in the armature circuit gives the possibility for adjustments.

2) Field-controlled DC servomotors. Their principle of operation lies in regulating the field current, which leads to modulation of the magnetic flux and the result is the generated torque and back electromotive force.

For armature voltage, a decrease in field flux results in a higher speed and lower torque, while a stronger magnetic field leads to lower speed and higher torque.

3) Linear actuators. While servomotors convert the electrical energy into the angular displacement of the rotor shaft, actuators can transform hydraulic, pneumatic, or electrical energy into the linear motion and are used to push, lift, or slide objects along a straight trajectory. Electric linear actuators consist of the AC or DC electric motor, gear, lead screw or ball screw and a sliding shaft. The motor drives the screw translating rotational motion into linear displacement.

4) Rotary actuators. These types of actuators produce rotational motion by converting electrical, hydraulic, or pneumatic energy. For the electric rotary actuators, DC or stepper motors are used to rotate the shaft of the actuator.

While designing the knee joint of a prosthetic leg, we tend to achieve a balance between fast response, control accuracy, and energy efficiency. Among the options available, armature-controlled DC servomotors present a solution for above-knee prostheses.

DC servomotors provide high-resolution control of torque and angular velocity due to their direct relationship between armature voltage and servomotor output. Quick acceleration and deceleration allow the prosthesis to replicate natural limb dynamics. Torque control ensures a smooth transition between gain phases. Another advantage of such motors is their simplified control design: by keeping the magnetic flux constant and only varying armature voltage, it allows for easier tuning of feedback loops for position and velocity control, and compact microcontroller-based implementation.

Modern DC servomotors available on the market offer torque-to-weight and power-to-weight ratios for customers. It makes them suitable for powering from batteries, extending operating time without sacrificing performance, since energy is consumed only during movement, and low mechanical inertia facilitates energy savings during gait transitions.

One of the requirements for the design of prostheses is the lightweight of components to prevent user fatigue and maintain biomechanical alignment, and DC servomotors are available in miniature but high-torque variants that can satisfy the criterion of choosing a lightweight device.

1.11. Lower-knee prosthetic leg model

In terms of control systems, for the lower knee model key component is the spring mass damper system. By means of this simplified system, we can obtain an understanding of how a prosthetic limb interacts with the surface, absorbs shock, stores and returns energy, and mimics the gait cycle of a patient.

The main elements of a spring-mass-damper system are following:

1) Spring represents the elastic properties of leg and prosthetic materials, able to store mechanical energy and release it during extension.

2) Mass represents the weight of prosthetic components and, in some cases weight of a patient.

3) Damper simulates energy dissipation with the help of internal friction and resistance.

The lower-knee prosthesis functions should correspond to:

1) Absorb impact at heel strike, reducing vibrations which travel to the socket and improving comfort while using.

2) Replicate the natural biomechanical dynamics of walking by providing smooth motion transitions and better adaptation to the surface.

3) storing energy contained in a spring and its release during a gait cycle.

The main task regarding the design of the lower-knee model is to determine such coefficients of the suspension system that provide comfortable specific activity.

1.12. Conclusion

In this chapter, we considered main components for lower limb prosthetis control system based on approaches developed in prosthetics and control system fields. Among components, we include the input data in the form of thigh motion and weight acting as external force on the leg. To predict the expected motion of the leg based on input data from sensors, we include two controllers developed in different environments but which take the same input and provide the same output. To replicate natural knee joint of the user we will use armature-controlled DC servomotor that will rotate the shaft with kower knee prosthetic leg attached at certain degree. The lower knee-prosthetic leg provides comfortable walking activity by absorbption and releasing of external force

CHAPTER 2
LOWER LIMB PROSTHESIS MODELLING

2.1. Equations of motion of lower limb

To simulate the motion of the human leg, we have to find the equations of motion of the double pendulum given in fig. 2.1.1.

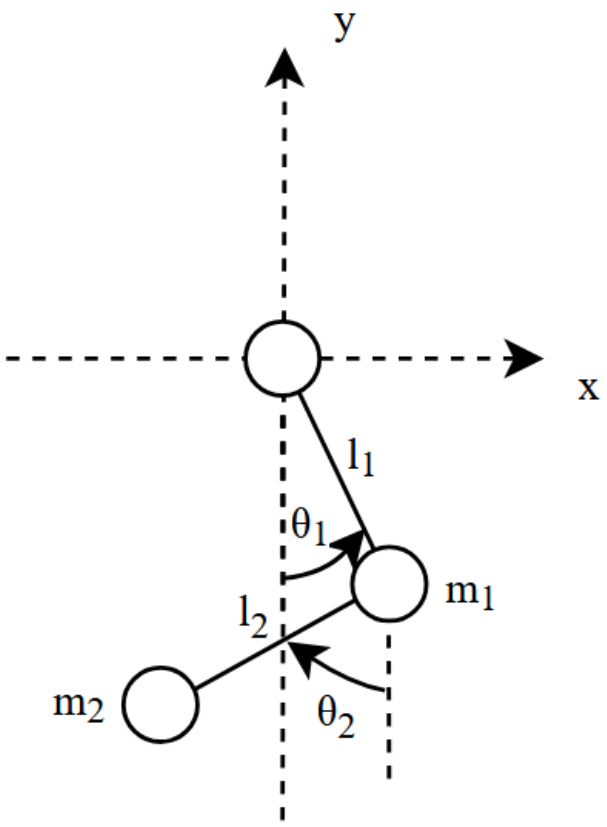

Fig. 2.1.1. Double pendulum diagram

In the following model, m_1 and m_2 are centers of masses of upper and lower links, l_1 and l_2 are their lengths, respectively. θ_1 and θ_2 determine the angular displacements of upper and lower links in space, accordingly.

For determining the equations of motion of the leg, we will derive them by means of Lagrangian dynamics. Unlike Newtonian dynamics, which requires including forces acting on the system, the first one allows us to deal with only potential and kinetic energies combined into a single equation:

$$L = T - V, \tag{2.1.1}$$

where T is the kinetic energy;

V is the potential energy.

The algorithm for deriving equations of motion for the double pendulum consists of the following steps:

1) Determine kinetic and potential energy equations for lower and upper links of the human leg. The kinetic energy is equal to the sum of kinetic ones of both links:

$$T = T_{upper} + T_{lower}, \tag{2.1.2}$$

$$T_{upper} = \frac{1}{2} m_1 \left(\dot{x_1^2} + \dot{y_1^2} \right), \tag{2.1.3}$$

$$T_{lower} = \frac{1}{2} m_2 \left(\dot{x_2^2} + \dot{y_2^2} \right), \tag{2.1.4}$$

where $\dot{x_1^2} + \dot{y_1^2}$ define velocity of center of mass m_1,

$\dot{x_2^2} + \dot{y_2^2}$ define velocity of mass m_2.

Substitute (2.1.3), (2.1.4) into (2.1.2) and obtain the following expression:

$$T = \frac{1}{2} m_1 \left(\dot{x_1^2} + \dot{y_1^2} \right) + \frac{1}{2} m_2 \left(\dot{x_2^2} + \dot{y_2^2} \right). \tag{2.1.5}$$

2) Find potential energy V for both links, which is the of potential energies of upper and lower links:

$$V = V_{upper} + V_{lower}, \tag{2.1.6}$$

$$V_{upper} = m_1 y_1 g, \tag{2.1.7}$$

$$V_{lower} = m_2 y_2 g. \tag{2.1.8}$$

Substitute (2.1.7) and (2.1.8) into (2.1.6):

$$V = -m_1 y_1 g - m_2 y_2 g. \quad (2.1.9)$$

3) Find locations of center for mass m_1 in terms of coordinates x_1 and y_1:

$$x_1 = l_1 \sin\theta_1, \quad (2.1.10)$$

$$y_1 = -l_1 \cos\theta_1. \quad (2.1.11)$$

Location of center of mass m_2 for coordinates x_2 and y_2 is equal to:

$$x_2 = l_1 \sin\theta_1 + l_2 \sin\theta_2, \quad (2.1.12)$$

$$y_2 = -l_1 \cos\theta_1 - l_2 \cos\theta_2, \quad (2.1.13)$$

Find velocities for m_1 by differentiating the equations (2.1.10) and (2.1.11):

$$\dot{x_1} = l_1\dot{\theta_1} \cos\theta_1, \quad (2.1.14)$$

$$\dot{y_1} = l_1\dot{\theta_1} \sin\theta_1. \quad (2.1.15)$$

Differentiate (2.1.12) and (2.1.13) to find velocity for mass m_2:

$$\dot{x_2} = l_1\dot{\theta_1} \cos\theta_1 + l_2\dot{\theta_2} \cos\theta_2, \quad (2.1.16)$$

$$\dot{y_2} = l_1\dot{\theta_1} \sin\theta_1 + l_2\dot{\theta_2} \sin\theta_2. \quad (2.1.17)$$

Now substitute equations (2.1.14) -(2.1.17) into (2.1.5) and (2.1.11), (2.1.13) into (2.1.9) [3]:

$$T = \frac{1}{2}m_1\left(\left(l_1\dot{\theta_1}\cos\theta_1\right)^2 + \left(l_1\dot{\theta_1}\sin\theta_1\right)^2\right) +$$

$$\frac{1}{2}m_2\left(\left(l_1\dot{\theta_1}\cos\theta_1 + l_2\dot{\theta_2}\cos\theta_2\right)^2 + \left(l_1\dot{\theta_1}\sin\theta_1 + l_2\dot{\theta_2}\sin\theta_2\right)^2\right),$$

$$T = \frac{1}{2}m_1 l_1^2\dot{\theta_1^2} + \frac{1}{2}m_2\left(l_1\dot{\theta_1^2} + l_2^2\theta_2^2 + 2l_1 l_2\dot{\theta_1}\,\dot{\theta_2}\cos(\theta_1 - \theta_2)\right), \quad (2.1.18)$$

$$V = -m_1 g l_1 c\cos\theta_1 + m_2 g(l_1\cos\theta_1 + l_2\cos\theta_2). \quad (2.1.19)$$

4) The next stage is to substitute potential (2.1.18) and kinetic (2.1.19) energy expressions onto Lagrangian equation (2.1.1), what is the difference between kinetic and potential energies:

$$L = \frac{1}{2}m_1 l_1^2\dot{\theta_1^2} + \frac{1}{2}m_2\left(l_1\dot{\theta_1^2} + l_2^2\theta_2^2 + 2l_1 l_2\dot{\theta_1}\,\dot{\theta_2}\cos(\theta_1 - \theta_2)\right) +$$
$$m_1 g l_1 c\cos\theta_1 + m_2 g(l_1\cos\theta_1 + l_2\cos\theta_2). \quad (2.1.20)$$

5) Derive the equations of motion using Euler-Lagrange equations:

$$\frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\partial L}{\partial\dot{\theta}_\iota}\right) - \frac{\partial L}{\partial\theta_i}.$$

Equation of motion θ_1 has the form of:

$$0 = \frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\partial L}{\partial \dot{\theta_1}}\right) - \frac{\partial L}{\partial \theta_1},$$

$$0 = (m_1 + m_2)l_1^2\ddot{\theta_1} + m_2 l_1 l_2 \ddot{\theta_2}\cos(\theta_1 - \theta_2) - m_2 l_1 l_2 \dot{\theta_2}^2 \sin(\theta_1 - \theta_2) + (m_1 + m_2)g l_1 \sin\theta_1 . \quad (2.1.21)$$

Equation of motion θ_2 has the form of:

$$0 = \frac{\mathrm{d}}{\mathrm{dt}}\left(\frac{\partial L}{\partial \dot{\theta_2}}\right) - \frac{\partial L}{\partial \theta_2},$$

$$0 = m_2 l_2^2 \ddot{\theta_2} + m_2 l_1 l_2 \ddot{\theta_1}\cos(\theta_1 - \theta_2) + m_2 l_1 l_2 \dot{\theta_1}^2 \sin(\theta_1 - \theta_2) + m_2 g l_1 \sin\theta_1 . \quad (2.1.22)$$

6) With the help of equations (2.1.21), (2.1.22), we may form a set of nonlinear differential equations that describe the double pendulum motion. It is necessary to note, that this system is nonlinear and can be approximated only at a specific condition by means of linearizing. However, we solve these equations to obtain the data of the motion of the upper link. The state variables are:

$$x = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \dot{\theta_1} \\ \dot{\theta_2} \end{bmatrix}.$$

The set of differential equations has the form of:

$$\begin{gathered} \frac{\mathrm{d}}{\mathrm{dt}}\theta_1 = \dot{\theta_1}, \\ \frac{\mathrm{d}}{\mathrm{dt}}\dot{\theta_1} = \frac{-(g(2m_1 m_2)\sin(\theta_1) - m_2 g\sin(\theta_1) - 2\theta_2) - 2Sm_2\left(\dot{\theta_2}^2 l_2 l_1 \dot{\theta_1^2} C\right)}{l_1(2m_1 + m_2 - m_2\cos(2\theta_1 - 2\theta_2))}, \\ \frac{\mathrm{d}}{\mathrm{dt}}\theta_2 = \dot{\theta_2}, \\ \frac{\mathrm{d}}{\mathrm{dt}}\dot{\theta_2} = \frac{2S\left(\dot{\theta_1^2}^2 l_1(M_1) + g(M_1)\cos(\theta_1) + \dot{\theta_2^2} l_2 m_2 C\right)}{l_2(2m_1 + m_2 - m_2\cos(2\theta_1 - 2\theta_2))}, \end{gathered} \quad (2.1.24)$$

where $S = \sin(\theta_1 - \theta_2)$, $C = \cos(\theta_1 - \theta_2)$, $M_1 = m_1 + m_2$.

From set of equations (2.1.24), we derive the motion of upper link of leg, i.e., thigh for its further implementation of input to fuzzy controller. To obtain data regarding motion of the leg, we developed MATLAB code given in Appendix 1.

2.2. Control unit of prosthetic leg

2.2.1. Fuzzy logic controller

Gait cycle, which is used to form the FLC (where the colored leg is the prosthetic one) is shown in fig. 2.2.1.1.

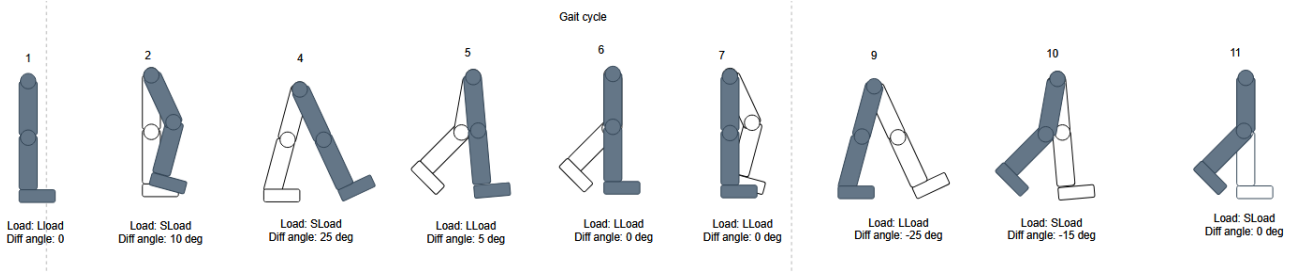

Fig. 2.2.1.1. The gait cycle describes walking activity

On the graph, it can be seen that the knee bends during movement from one furthest position to another. At the furthest position of the leg, the knee angle is straight to provide the stability of movement, while transferring mass from one leg to another. With the following analysis, the following algorithm was created to use it for forming fuzzy control of the prosthetic leg, which is shown in fig. 2.2.1.2.

This algorithm presents an approximate, in terms of the range of values, principle of operation of the control unit. The output control signal depends on the position of the thigh in space. If it is in the range of minus 17,5 to plus 17,5 degrees, then the knee joint is bent on the desired radius for the patient to ensure movement of the leg during walking, preventing the possibility of falling. A load sensor in the control system is additionally used to determine the load that acts on the prosthetic leg. The first criterion that influences the knee angle value is the force load; if it is present, then the knee angle should be equal to zero, otherwise, the angle is determined depending on the position of the thigh.

For the construction of the control object, we chose the fuzzy control system due to its flexibility in terms of adjusting, and possibility to operate human terms for the controlling, while Sugeno model provides exact result in the form of constant values for the servomotor.

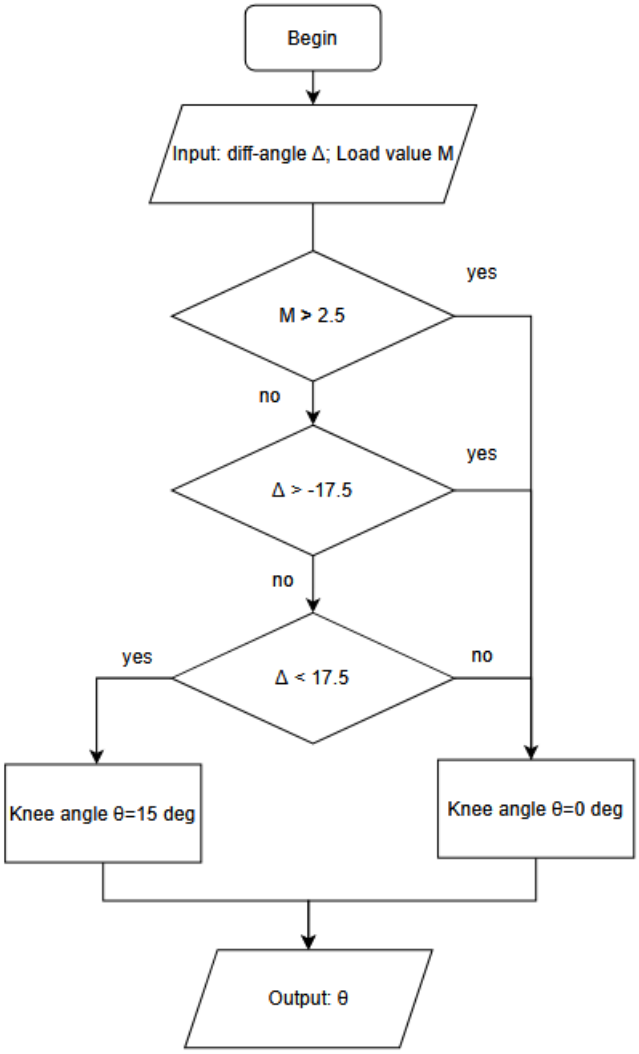

Fig. 2.2.1.2. Knee angle determination algorithm

The developed fuzzy system consists of the two input and one output membership functions, namely “diff-angle”, “Load”, “Knee-angle”, fuzzy sets of which are denoted in tables 2.2.1.1-2.2.1.3, respectively:

Table 2.2.1.1

Membership functions of the input variable “Diff-angle”

№	Membership function variable name:	Value set
1	Nlarge	[- 35 -35 -17,5]
2	NSmall	[-25 -12,5 0]
3	Medium	[-12,5 0 12,5]
4	PSmall	[0 12,5 25]

5	PLarge	[17,5 35 35]

Table 2.2.1.2

Membership functions of the input variable “Load”

№	Membership function variable name:	Value set
1	Sload	[0 2,5 5]
2	MLoad	[0 25 50]
3	Lload	[35 70 70]

Table 2.2.1.3

Membership functions of the output variable “Knee-angle”

№	Membership function variable name:	Value set
1	Zero	0
2	Walking-knee-value	15

The membership functions of input and output variables of the system are also depicted by means of MATLAB fuzzy toolbox and are given below (fig. 2.2.1.3-2.2.1.6).

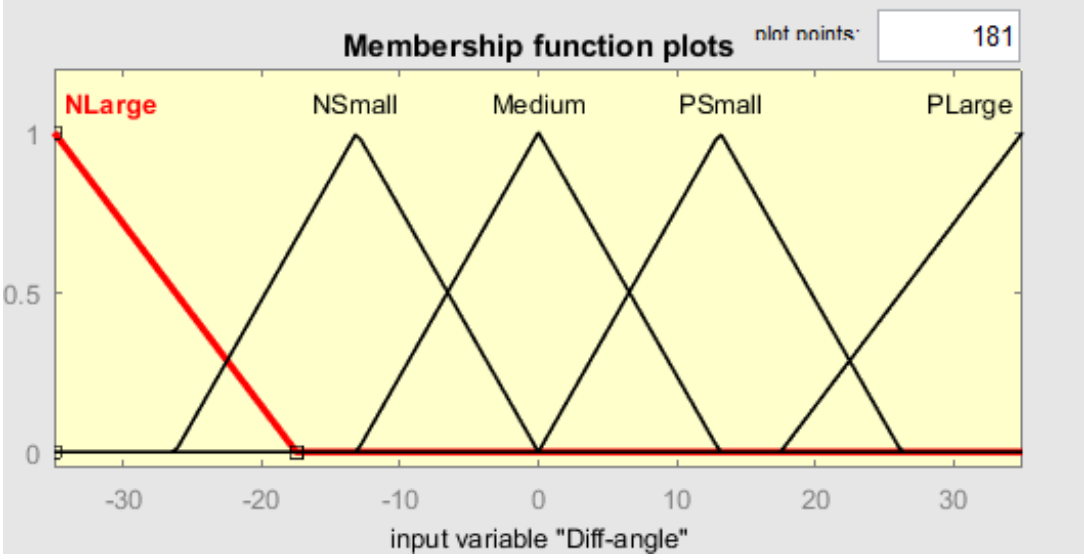

Fig. 2.2.1.3. Membership functions of the input variable “Diff-angle”

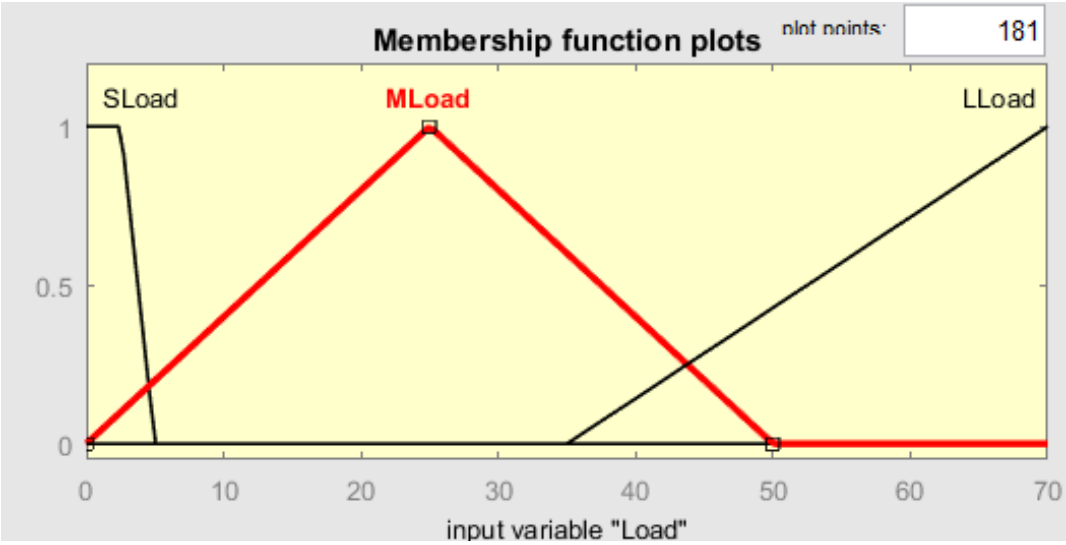

Fig. 2.2.1.4. Membership functions of the input variable “Load”

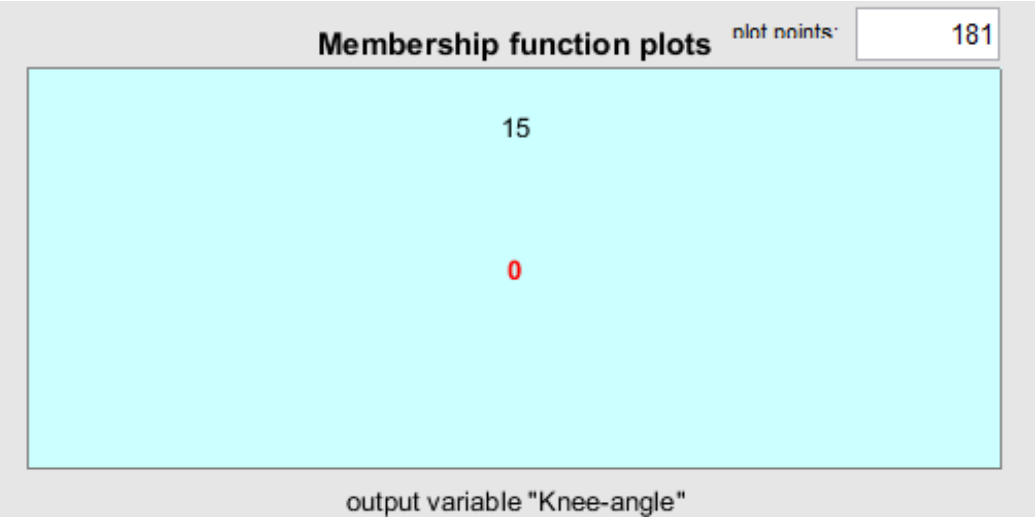

Fig 2.2.1.5. Membership functions of the output variable “Knee-angle”

The View of the system in MATLAB fuzzy editor is given below in fig. 2.2.1.6:

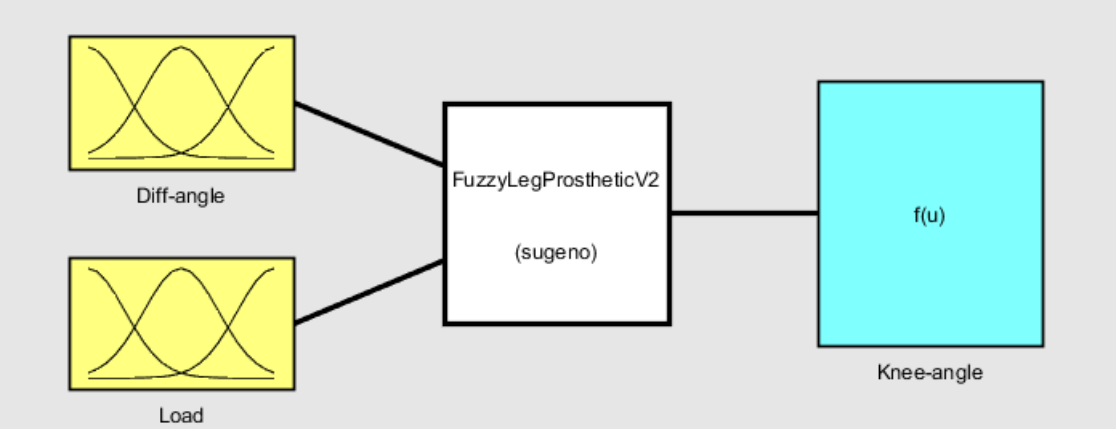

Fig. 2.2.1.6. Fuzzy logic control unit in fuzzy editor

The rule base of the Sugeno system contains the following rules:

1. If (Diff-angle is NLarge) and (Load is SLoad) then (Knee-angle is Zero) (1)
2. If (Diff-angle is NLarge) and (Load is MLoad) then (Knee-angle is Zero) (1)
3. If (Diff-angle is NLarge) and (Load is LLoad) then (Knee-angle is Zero) (1)
4. If (Diff-angle is NSmall) and (Load is SLoad) then (Knee-angle is Walking-knee-value) (1)
5. If (Diff-angle is NSmall) and (Load is MLoad) then (Knee-angle is Zero) (1)
6. If (Diff-angle is NSmall) and (Load is LLoad) then (Knee-angle is Zero) (1)
7. If (Diff-angle is Medium) and (Load is SLoad) then (Knee-angle is Walking-knee-value) (1)
8. If (Diff-angle is Medium) and (Load is MLoad) then (Knee-angle is Zero) (1)
9. If (Diff-angle is Medium) and (Load is LLoad) then (Knee-angle is Zero) (1)
10. If (Diff-angle is PLarge) and (Load is SLoad) then (Knee-angle is Zero) (1)
11. If (Diff-angle is PLarge) and (Load is MLoad) then (Knee-angle is Zero) (1)
12. If (Diff-angle is PLarge) and (Load is LLoad) then (Knee-angle is Zero) (1)
13. If (Diff-angle is PSmall) and (Load is SLoad) then (Knee-angle is Walking-knee-value) (1)
14. If (Diff-angle is PSmall) and (Load is MLoad) then (Knee-angle is Zero) (1)
15. If (Diff-angle is PSmall) and (Load is LLoad) then (Knee-angle is Zero) (1)

Total rule table 2.2.1.4 by means of which we can predict the fuzzy output is given below:

Table 2.2.1.4

Rule table of fuzzy logic controller

Load/Diff-angle	NLarge	NSmall	Medium	PSMall	PLarge
SLoad	Zero	Walking-knee-walue	Walking-knee-walue	Walking-knee-walue	Zero
MLoad	Zero	Zero	Zero	Zero	Zero
LLoad	Zero	Zero	Zero	Zero	Zero

The rule base in fuzzy editor is given (fig. 2.2.1.7).

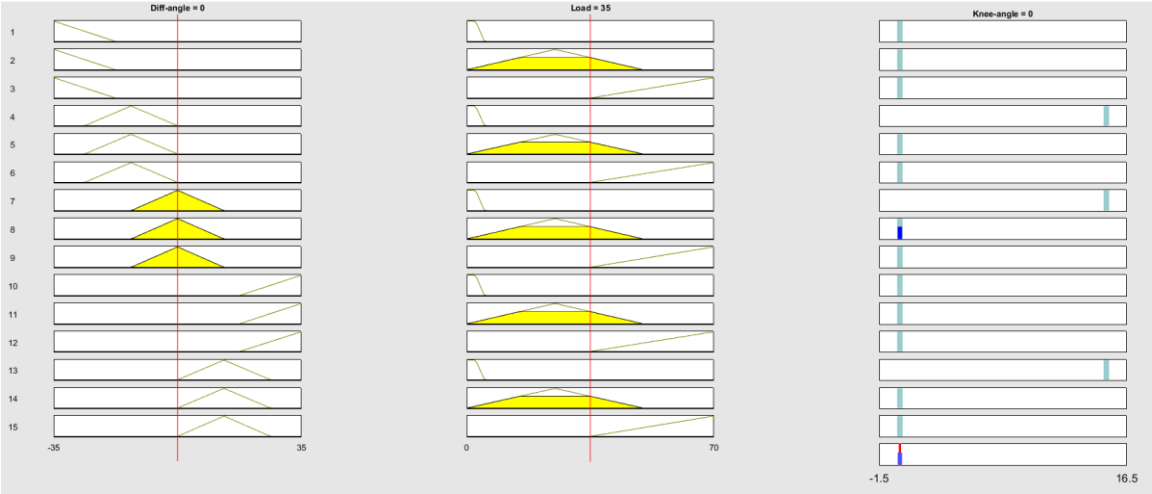

Fig. 2.2.1.7. Rule base formed by means of MATLAB fuzzy toolbox

2.2.2. SCADE control unit

Taking into account that the failure of the control system will lead to a fall of the patient and further injuries of the possible death, we are more likely to provide the secondary control system, which will provide stable walking. For these purposes, we chose SCADE software, in which we developed the model that can be used as an alternative. Though it does not provide the flexibility and possibility to operate using linguistic values offered by fuzzy systems, it can guarantee that the patient will be able to continue motion without falling.

SCADE model, which represents the algorithm is shown in fig. 2.2.2.1.

We can see in fig. 2.2.2.1, that the SCADE model consists of a pair of if-blocks, and each of them checks whether the criterion of the block is satisfied. The condition of the outer if-block is weight presence. If it is not satisfied, the pair of if-blocks check the criterion of the presence of leg in the swing phase in boundaries from minus 17,5 to plus 17,5 degrees. For each block, according outcome is in the form of a constant value.

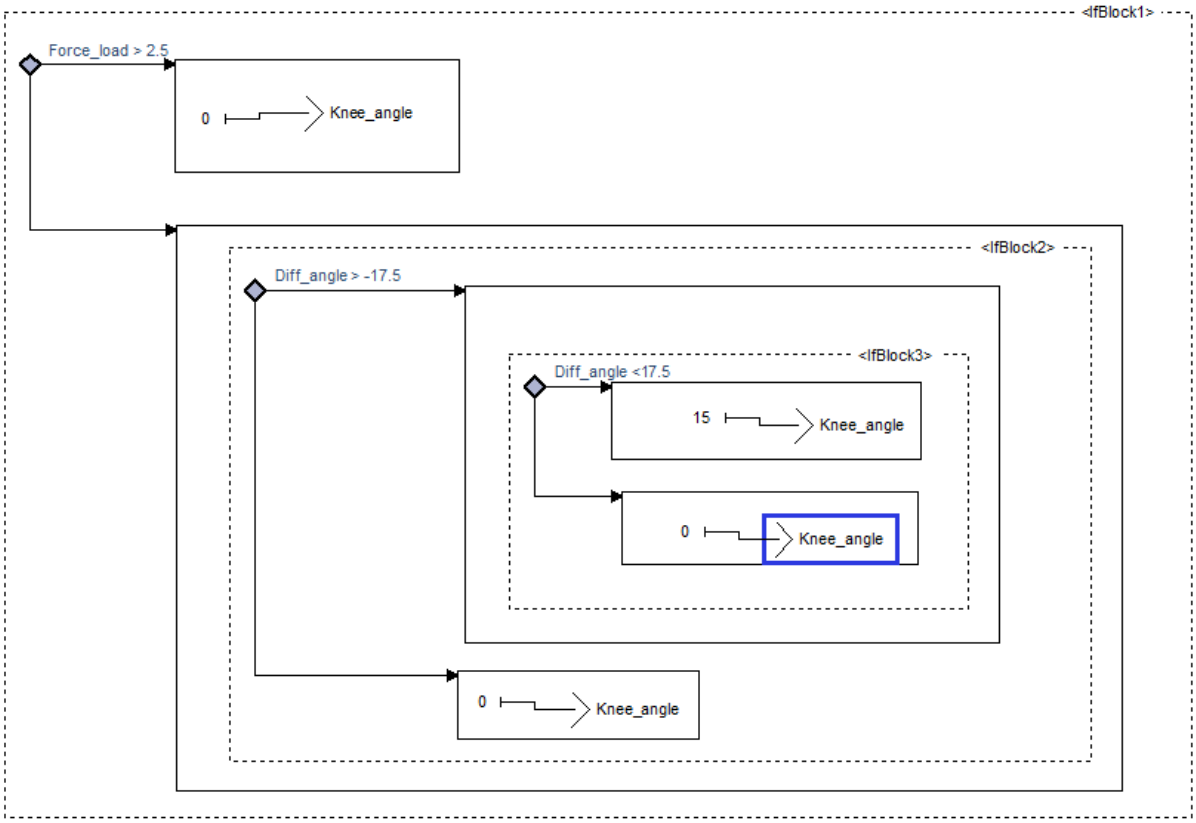

Fig. 2.2.2.1. Control unit designed in SCADE

Both Simulink and SCADE models correspond to multi-version heterogeneous software. It is approach of system architecting, in which two or more components are created in the way that they must provide execution of the same function with avoiding of appearing common errors of both components.

The methods of creating multi-version software:

1) Source code is formed using two or more programming languages.

2) Object code is generated using two or more different compilers;

3) Each object code is executed on the different processors or on the same one but using instruments that allow to distinguish projects.

4) Executed object code is changed and loaded by means of two or more link editors.

5) Code or software requirements are composed by two or more different developers' groups which coordinate with each other.

7) Code or software requirements are developed by means of two or more integrated development requirements (IDE) and each version is verificated using separated test environment.

8) Requirements are developed according to two or more certified standards.

Both Simulink and SCADE models of represent control unit component, both of them perform the same function. The only difference that there were used two different development environments: Simulink and SCADE suite. Both of models were tested separately using different approaches such as simulation and MC/DC. Because of that, we can conclude, that points 2 and 3 are performed.

To implement SCADE control unit software into SIMULINK environment, we converted C code from SCADE to SIMULINK. The resulting code of MATLAB function is given in Appendix 4.

2.3. Armature-controlled DC servomotor

2.3.1. Mathematical modelling of DC servomotor

The diagram of the armature-controlled DC servomotor system is given in fig. 2.3.1.1.

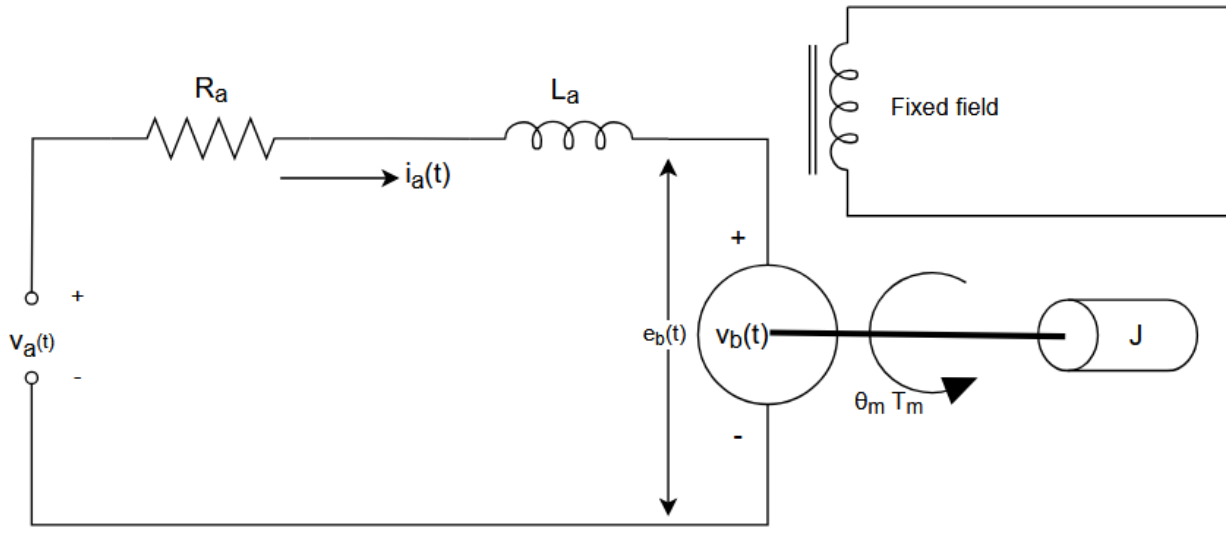

Fig. 2.3.1.1. Armature-controlled DC servomotor schematic diagram

The following system consists of electric circuit and mechanical system representing with angular movement. The derivation algorithm of the servomotor equations is divided into the determining the equations for both parts.

1) In armature-controlled DC servomotors, the applied voltage e_a controls the shaft of the motor with the constant field current i_a. Air-gap flux is proportional to the field current:

$$\phi \propto i_f \rightarrow \phi = K_f i_f,$$

where K_f is the field constant.

The torque produced by the servomotor is proportional to the air-gap flux and the armature current:

$$T_m \propto K_f i_f i_a \rightarrow T_m = K_t i_a, T_m = K_t I_a(s),$$

where K_t is the motor torque constant.

Back EMF is proportional to the speed and therefore equals to:

$$e_b = K_b \left(\frac{\mathrm{d}}{\mathrm{dt}}\theta\right)$$

For the electrical circuit equation derivation, Kirchhoff's laws for current and voltage are applied, which state:

1) Directed sum of all voltages in the circuit must be equal to zero:

$$\sum_{i=1}^{n} V_i = 0,$$

where n is the number of all voltages in the loop

2) Algebraic sum of currents meeting at a node must be zero:

$$\sum_{i=1}^{n} I_i = 0,$$

where n is the number of branches with currents meeting at a node.

Using Kirchhoff's voltage law, we determine the differential equation describing the armature circuit:

$$e_a = L_a \left(\frac{\mathrm{d}}{\mathrm{dt}} i_a\right) + R_a i_a + e_b. \qquad (2.3.1.1)$$

The torque equation is equal to the sum of inertia moment of load J, viscous friction b_0:

$$J\frac{\mathrm{d}^2}{\mathrm{dt}^2}\theta + b_0\left(\frac{\mathrm{d}}{\mathrm{dt}}\theta\right) = T_m = K_t i_a \qquad (2.3.1.2)$$

To determine the transfer function, we convert the equations (2.3.1.1) and (2.3.1.2) from time domain to the Laplace domain:

$$e_a = L_a\left(\frac{\mathrm{d}}{\mathrm{dt}}i_a\right) + R_a i_a + e_b \rightarrow E_a(s) = L_a s I_a(s) + R_a I_a(s) + E_b(s),$$

$$J\frac{\mathrm{d}^2}{\mathrm{dt}^2}\theta + b_0\left(\frac{\mathrm{d}}{\mathrm{dt}}\theta\right) = T_m = K_t i_a \rightarrow Js^2\theta(\mathrm{s}) + b_0 s\theta(\mathrm{s}) = T_m = K_t i_a,$$

$$(Js^2 + b_0 s)\theta(s) = K_t I_a(s) \rightarrow \theta(s) = \frac{K_t I_a(s)}{Js^2 + b_0 s}.$$

The resulting transfer function is determined as the ratio of the angular displacement of the rotor shaft $\theta(s)$ and the armature current $E_a(s)$:

$$G(\mathrm{s}) = \frac{\theta(\mathrm{s})}{E_a(\mathrm{s})} = \frac{K_t}{s[(sL_a + R_a)(sJ + b_0) + K_t K_b]}.$$

State-space of the Armature-controlled DC servomotor is represented by state and output equations of the form:

$$\dot{x} = Ax(t) + Bu(t),$$

$$y = Cx(t) + Du(t)$$

From the equations (2.3.1.1) and (2.3.1.2) derived, we may obtain expressions for angular acceleration and armature current determination:

$$J\frac{\mathrm{d}^2}{\mathrm{dt}^2}\theta + b_0\left(\frac{\mathrm{d}}{\mathrm{dt}}\theta\right) = K_t i_a \rightarrow J\ddot{\theta} = K_t I_a - b_0\dot{\theta} \rightarrow \ddot{\theta} = \frac{K_t}{J}i_a - \frac{b_0}{J}\dot{\theta},$$

$$L_a\left(\frac{\mathrm{d}}{\mathrm{dt}}i_a\right) + R_a i_a + K_b\left(\frac{\mathrm{d}}{\mathrm{dt}}\theta\right) = e_a \rightarrow L_a\ddot{i_a} +$$

$$R_a i_a + K_b\dot{\theta} = e_a \rightarrow \ddot{i_a} = \frac{e_a}{L_a} - \frac{R_a}{L_a}i_a - \frac{K_b}{L_a}\dot{\theta}.$$

As state-space variables, we choose angular displacement, angular velocity, and armature current:

$$\mathrm{x}_1 = \theta, \mathrm{x}_2 = \dot{\theta}, \mathrm{x}_3 = i_a.$$

State-space matrices A, B, C, D that describe the system are given below:

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -\frac{b_0}{J} & \frac{K_t}{J} \\ 0 & -\frac{K_b}{L_a} & -\frac{R_a}{L_a} \end{bmatrix}, B = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{L_a} \end{bmatrix}, C = [1 \quad 0 \quad 0], D = 0.$$

2.3.2. Full-state feedback controller with integral action design

Pole placement through Ackermann's formula is used in cases when the performance of the system is slow or the system is uncontrollable. It is used only for single-input single output systems, with the necessary condition for its application to be controllable, i.a, we need to ensure that there always exists such input $u(t)$, that any state x_0 can be transferred to any other system state x_0. To check the controllability of the system, we use the following expression:

$$M_c = [B \quad AB \quad \cdots \quad A^{n-1}B]. \qquad (2.3.2.1)$$

Rank of the matrix M_c obtained from (2.3.2.1) Must be equal to the number of states of the system:

$$n = \text{rank}(M_c),$$

where n is the number of states.

Ackermann's formula determines the feedback gains needed to move poles to the desired location of the following form:

$$K = [k_1 \quad k_2 \quad \ldots \quad k_n]$$

for such input, that:

$$u(t) = r(t) - Kx(t).$$

Resulting Ackermann's formula has the form:

$$[0 \quad 0 \quad \cdots \quad 1][B \quad AB \quad \cdots \quad A^{n-1}B]^{-1}\phi(A),$$

where $\phi(A)$ is the characteristic polynomial evaluated at matrix A and has the form:

$$\phi(A) = \alpha_n I + \alpha_{n-1}A + \cdots + \alpha_1 A^{n-1} + A^n.$$

The principle of operation of a state-feedback controller is to locate the poles of the system to desired coordinates by means of the feedback gain matrix K_g. The control law for a full-state feedback system is given below:

$$u = r - K_g x,$$

where K_g is the feedback gain matrix.

State-space equations for the SISO system gain the form:

$$\dot{x} = \left(A - BK_p\right)x + Br,$$

$$y = Cx.$$

The states of the following system can be directly measured by means of sensors, therefore, it is fully observable and there is no need for the observer design.

Integral action is introduced to the system due to the necessity to eliminate the steady-state error, which is the difference between an input of the system and its output. To include integral action into the system, additional state variable w is added, which is the integral of the error.

The integral state variable is described by:

$$w = \int (y - r)\mathrm{dt},$$

where r is the reference input;

y is the output of the system;

$e = r - y$ is the error signal.

The control input u of the system takes the form:

$$u = -K_g x - K_i w.$$

State space representation of the full-state feedback with integral action (FSFIA) is given by the following equations:

$$\dot{x} = A_a x_a + B_a u + B_r r,$$

$$y = C_a x_a,$$

where A_a, B_a, C_a are augmented state-space matrices, B_r is the vector multiplying the reference input, x_a is vector of the augmented state variables. The resulting state-space matrices with integral and feedback controllers have the following form [4]:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{b_0}{J} & \frac{K_t}{J} & 0 \\ 0 & -\frac{K_b}{L_a} & -\frac{R_a}{L_a} & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}, B_r = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}, C = [1 \quad 0 \quad 0 \quad 0], D = 0$$

The schematic diagram of the full-state feedback with integral action is given (fig. 2.3.2.1).

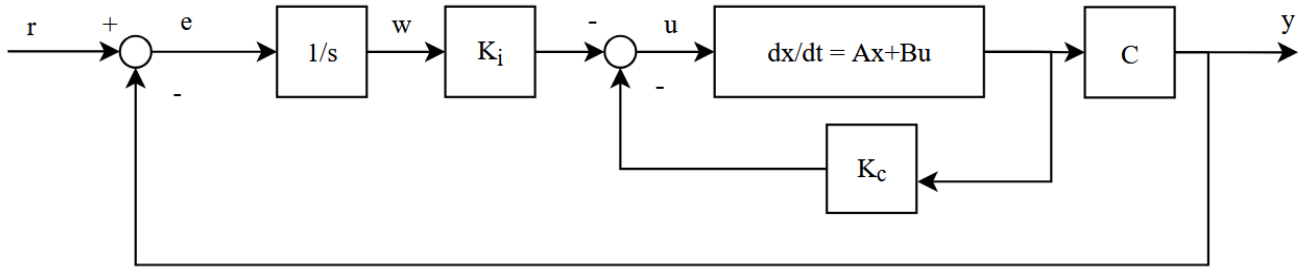

Fig. 2.3.2.1. Diagram of a system with FSFIA

2.4. Location of desired poles

Location of poles on the complex plane directly affects the system stability and performance. If system poles are located in the right half of the complex plane, then the system is unstable. If poles lie on an imaginary axis, then the system is on the boundary of stability. Poles on the left half plane describe not only the stability of the system, but also its transient parameters, such as rise time, overshoot, settling time, and steady-state time. The general recommendations about choosing desirable pole locations include:

1) There exists direct correlation between fast response of the system and its accuracy. Since response of our system is directly proportional to the location of poles, i.a. the further the poles of system from imaginary axis of complex plane, the faster the system, but at the same it is more subjected to noise input. Thus, it is necessary to balance between the fast response and resistance to input disturbance.

2) The further poles from the real axis, the more oscillatory the response of the system will be obtained.

3) The further poles from the imaginary axis, the less oscillatory the response system will be.

Fig. 2.4.1. presenting the graphical representation of the correlation between the location of poles and the system response is given [5].

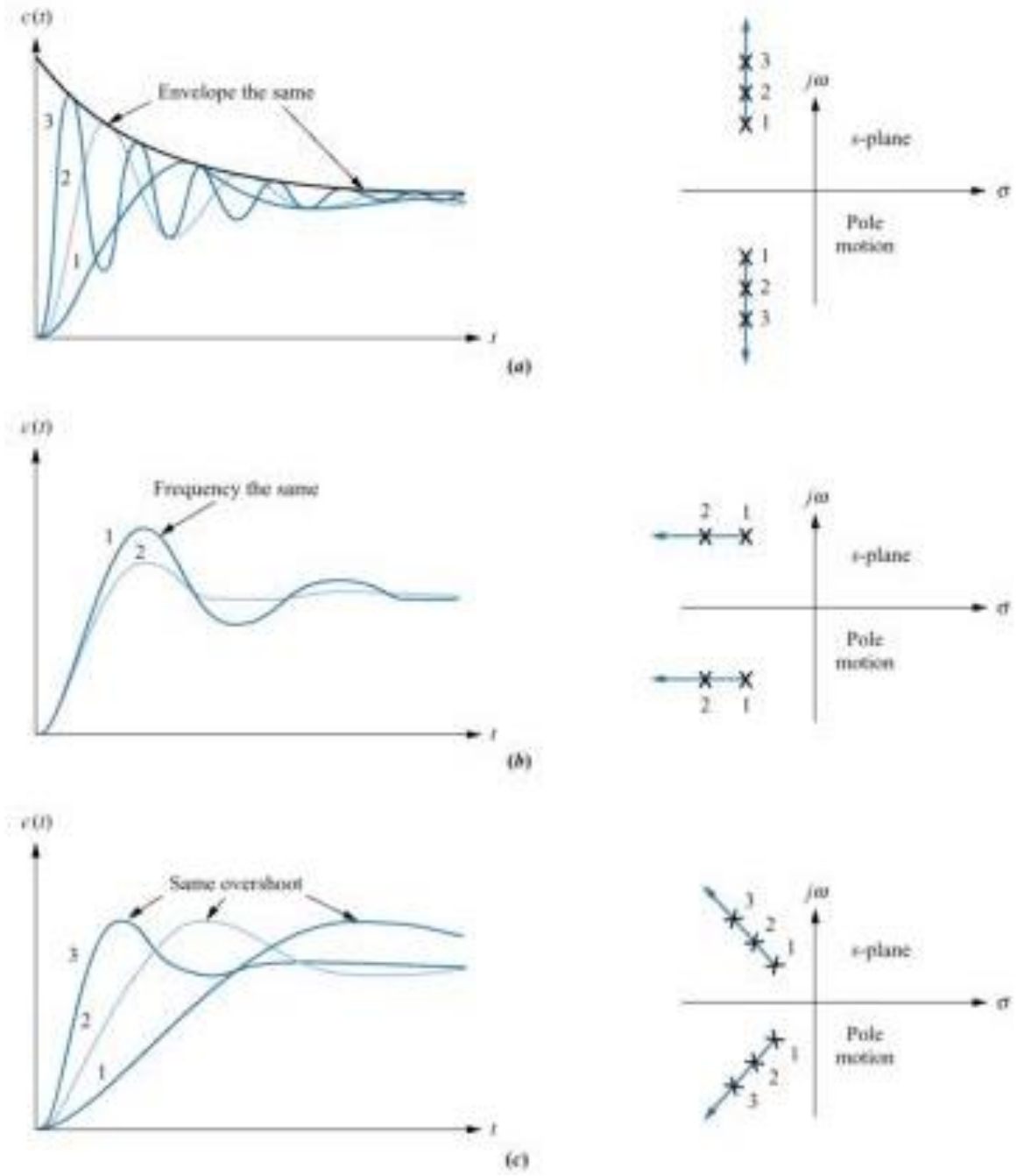

Fig. 2.4.1. Correlation between location of poles and system response

2.5. Lower-knee prosthetic leg mathematical model

A mass spring damper system used as a lower-knee prosthesis is shown in fig. 2.5.1.

The principle of operation of the mass-spring damper system is based on Newton's and Hooke's laws:

1) Newton's second law states, that the net force F is equals to the product of the mass of the object and the acceleration a:

$$F = m \times a.$$

2) Hooke's law describes that the force is proportional to the extension of the elastic body:

$$F = -kx,$$

where the minus sign means that the directions of force and displacement x are opposite to each other.

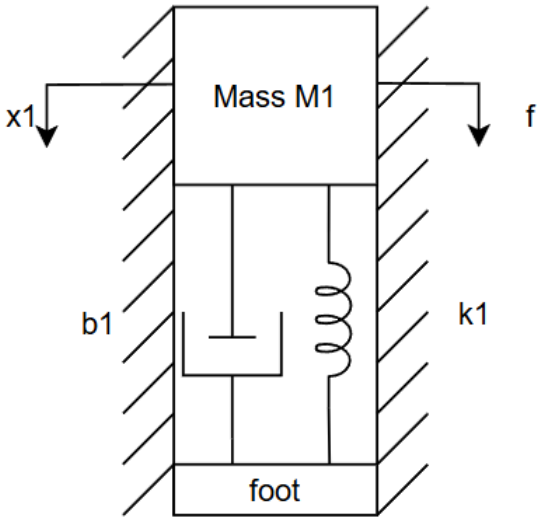

Fig. 2.5.1. Mechanical model of lower-knee prosthesis

The mass spring damper system is described by the following differential equations:

$$M_1\ddot{x_1} = f - k_1x_1 - b_1\dot{x_1}, \quad (2.5.1)$$

Now determine the transfer function of the mass-spring-damper system. Apply the Laplace transform to (2.5.1) for converting the system from a time domain to the Laplace one:

$$M_1\ddot{x_1} = f - k_1x_1 - b_1\dot{x_1} \rightarrow M_1s^2x_1(s) = f - k_1x_1(s) - b_1sx_1(s),$$

$$M_1 s^2 = F - k_1 - b_1 s.$$

Transfer function is the ratio of Laplace transform output to Laplace transform input. For our system, displacement of mass m is the output, applied force to the mass is the input of the system.

$$M_1 s^2 = f - k_1 - b_1 s \rightarrow M_1 s^2 + k_1 + b_1 s = F,$$

$$G(s) = \frac{X_1(s)}{F(s)} = \frac{F}{M_1 s^2 + k_1 + b_1 s}.$$

Secondly, determine the necessary parameters of the system. Applied force to the system is equal to [6]:

$$F = \left(a_{sf} * m\right) * g,$$

where a_{sf} is the scaling factor used to adjust the magnitude of the applied force;

g is the gravity force.

The values of scaling factors depend on the activity of human [6]:

1) For walking on the flat surface: $a_{sf} = 1.4$;

2) For running: $a_{sf} = 2{,}5$;

3) For jumping: $a_{sf} = 5{,}5$.

The value for spring constant is determined by [6]:

$$k = (2\pi f_{human})^2 * m,$$

where f_{human} is the human natural frequency influenced by the following activities:

1) For walking activity: $f_{human} = 1{,}8$ Hz;

2) For running activity: $f_{human} = 2{,}5$ Hz;

2) For jumping activity: $f_{human} = 5{,}5$ Hz.

The expression for damping coefficient is equal to [6]:

$$c = 2\sqrt{k * m}.$$

After coefficients of spring and damper were obtained, form linear-quadratic regulator to decrease the displacement of body mass m. To do it, firstly set state space model of spring-mass-damper system. State variables of our system are displacement of mass x and its velocity $\dot{x}$:

$$x = \begin{bmatrix} x \\ \dot{x} \end{bmatrix}, A = \begin{bmatrix} 0 & 1 \\ -\frac{b}{m} & -\frac{k}{m} \end{bmatrix}, B = \begin{bmatrix} 0 \\ \frac{f}{m} \end{bmatrix}, C = [1 \quad 0], D = [0].$$

Linear quadratic regulator is the control technique used to regulate the system performance. Its goal is to determine the input $u(t)$ that minimizes quadratic cost function of the form:

$$J = \int_0^\infty (x^T Q x + u^T R u)\mathrm{dt} + x^T P x,$$

where Q and R are symmetric positive weight matrices, by regulating factors q_i of which desired input is determined. Weight matrices have the form of:

$$Q = \begin{bmatrix} q_1 & 0 \\ 0 & q_2 \end{bmatrix}, \qquad R = [\, q_3].$$

To minimize the cost function, we need to solve algebraic Riccati equation:

$$C^T R_3 C - PBR^{-1}B^T P + A^T P + PA = 0,$$

where R_3 is positively defined symmetric matrix;

P is the solution to Riccati equation.

Control law $u(t)$ is described as follows:

$$u = -Kx$$

from which set of coefficients K at which cost function is equal to minimum is calculated.

The model examined before takes uses step function as input with an external force included. However, for the real-life applications external force acts on the system continuously; therefore, we will additionally check the forced response of the lower-knee model. For this purpose, the differential equation (2.5.1) will have the following form:

$$f\sin(\omega t) - k_1 x_1 - b_1 \dot{x_1} = M_1 \ddot{x_1}. \qquad (2.5.2)$$

For Simulink modelling, the equation (2.5.2) is transformed into:

$$\ddot{x_1} = \frac{f}{m}\sin(\omega t) + b_1 \dot{x_1} + k_1 x_1,$$

where ω is the natural frequency of the prosthetic leg. It determines the oscillations at which the prosthetic leg vibrates during impact from external force f and is determined by the expression [6]:

$$\omega = \sqrt{\frac{k}{m}}.$$

To determine response of lower knee prosthetic leg model, we used MATLAB code shown in Appendix 3.

2.6. Conclusion

In this chapter, we derived mathematical models for components of lower limb prosthesis. To recreate walking activity of the user, we will use set of equations describing motion of thigh in space (2.1.24) and weight of the user acting on the leg from (2.5.2). For armature-controlled DC servomotor, we derived state-space model. In order to improve transient characteristics, we developed full-state feedback with integral action system. The lower knee prosthetic leg state-space model is based on the spring-mass damper system with implementation of LQR controller. The following models will be implemented in MATLAB and SIMULINK software for analyzing their response parameters.

CHAPTER 3
SIMULATION RESULTS

3.1. Simulation data of human leg

For the simulation, we will use the parameters of a patient with 180 cm and 70 kg. Based on this information, table 3.1.1 with formed leg parameters is given below:

Table 3.1.1

Constant parameters

№	Constant parameter	Value	Unit of measurement
1	Upper link mass	2	kg
2	Upper link length	0,4	m
3	Lower link mass	2	kg
4	Lower link length	0,4	m

To begin the motion of the lower-knee model, we assign initial parameters, including the position of links and their velocities. Table 3.1.2 with these parameters is given.

Table 3.1.2

Initial parameters of the leg

№	Initial parameter	Value	Unit of measurement
1	Upper link position	0,4	rad
2	Upper link velocity	0	rad/s

3	Lower link position	0,2	rad
4	Lower link velocity	0	rad/s

Simulation was done by means of MATLAB software. To solve the set of ordinary differential equations, we used the Runge-Kutta method, which is implemented in MATLAB in the form of the operator “ode45”, which has the following syntax:

[t,y] = ode45(odefun, tspan, y0),

Where the inputs of the operator are:

1) time interval “tspan”;

2) set of ordinary differential equations “odefun”;

3) y0 are initial conditions y0.

As the output, we obtain time t and the solution of equations y in the form of matrices.

The graph containing the results of the simulation is given below (fig. 3.1.1).

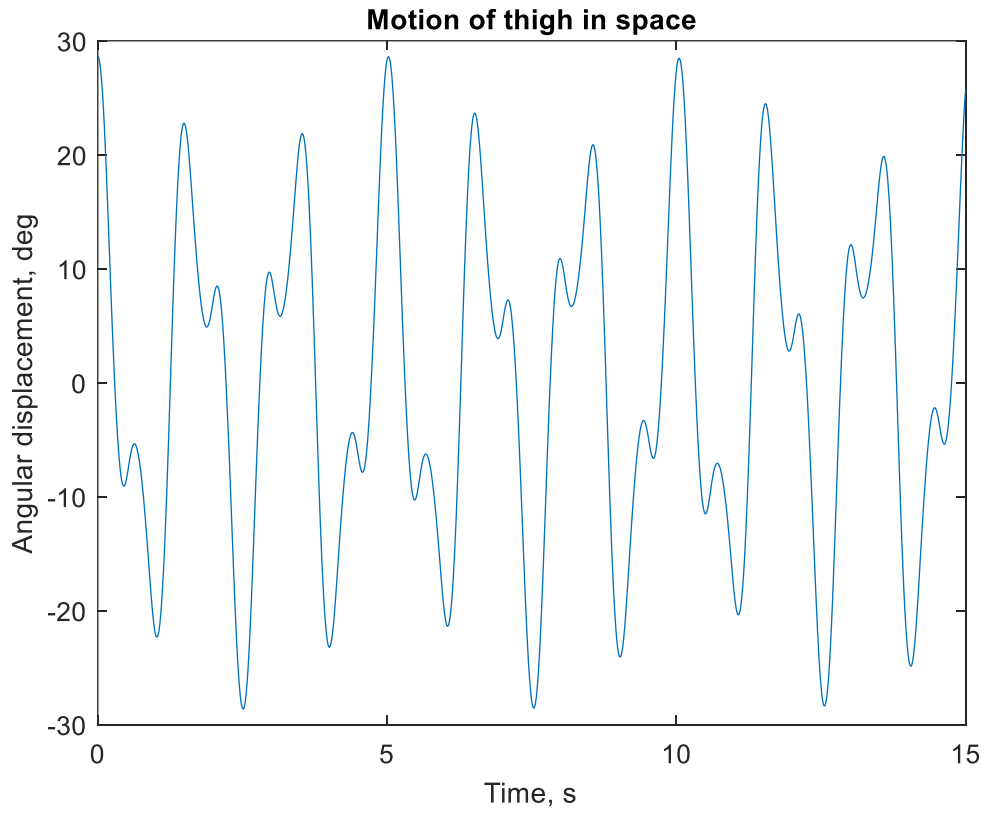

Fig. 3.1.1. Angular displacement of the thigh in space

As can be seen from the graph, the upper link's motion lies within boundaries from minus 30 to plus 30 degrees to represent maximum angle at which thigh of patient is displaced during walking activity.

To simulate the load acting on the leg during walking activity, we will use the SIMULINK toolbox with the sine function. The result of the simulation is given in fig. 3.1.2.

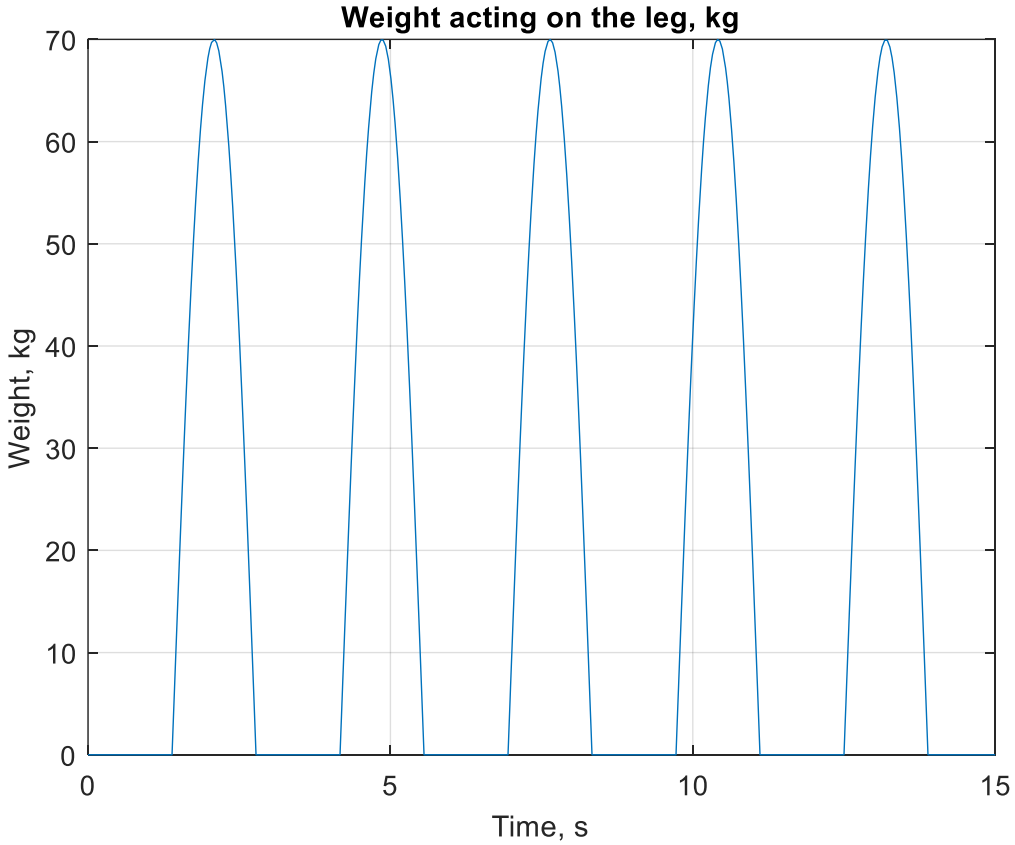

Fig. 3.1.2. The external force acting on the leg

The curve shown in fig. 3.1.2 represents weight input data obtained from load cell. Amplitude of weight takes values from 0 to 70 kg.

3.2. Control units' responses

3.2.1. Fuzzy logic controller simulation

To simulate the fuzzy logic controller operation, we use the data obtained in the previous chapter. Simulation is taken in the SIMULINK software package, with testing two input parameters at once to ensure that both of them impact the output independently from each other. The results of the simulation with the motion of the thigh as an input are given in fig. 3.2.1.1.

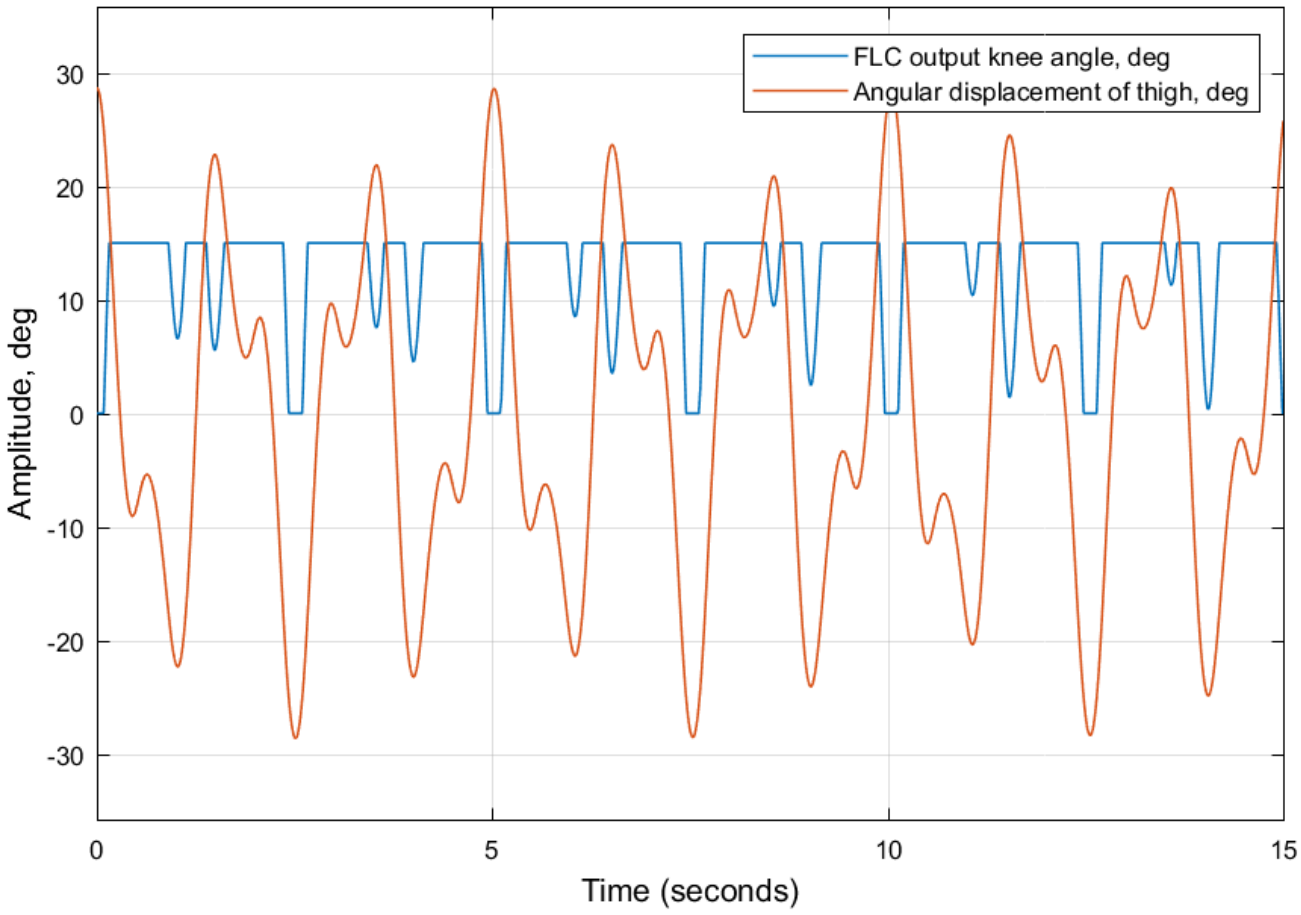

Fig. 3.2.1.1. Output of the Fuzzy logic controller with the motion of the thigh included

From the graph in fig. 3.2.1.1, we see, that curve representing fuzzy controller output tends from 0 to 15 degrees based on the input simulation data of thigh movement. When angular displacement of upper link is in the range from minus 17,5 to plus 17,5

degrees, representing swing phase, signal of controller tends to 15 degrees. In other cases, the signal is equal to 0 degrees.

For the simulation with only the weight of the patient subjected as input, consider the natural frequency of the prosthetic leg equal to ω=11,3097. The result is shown in fig. 3.2.1.2.

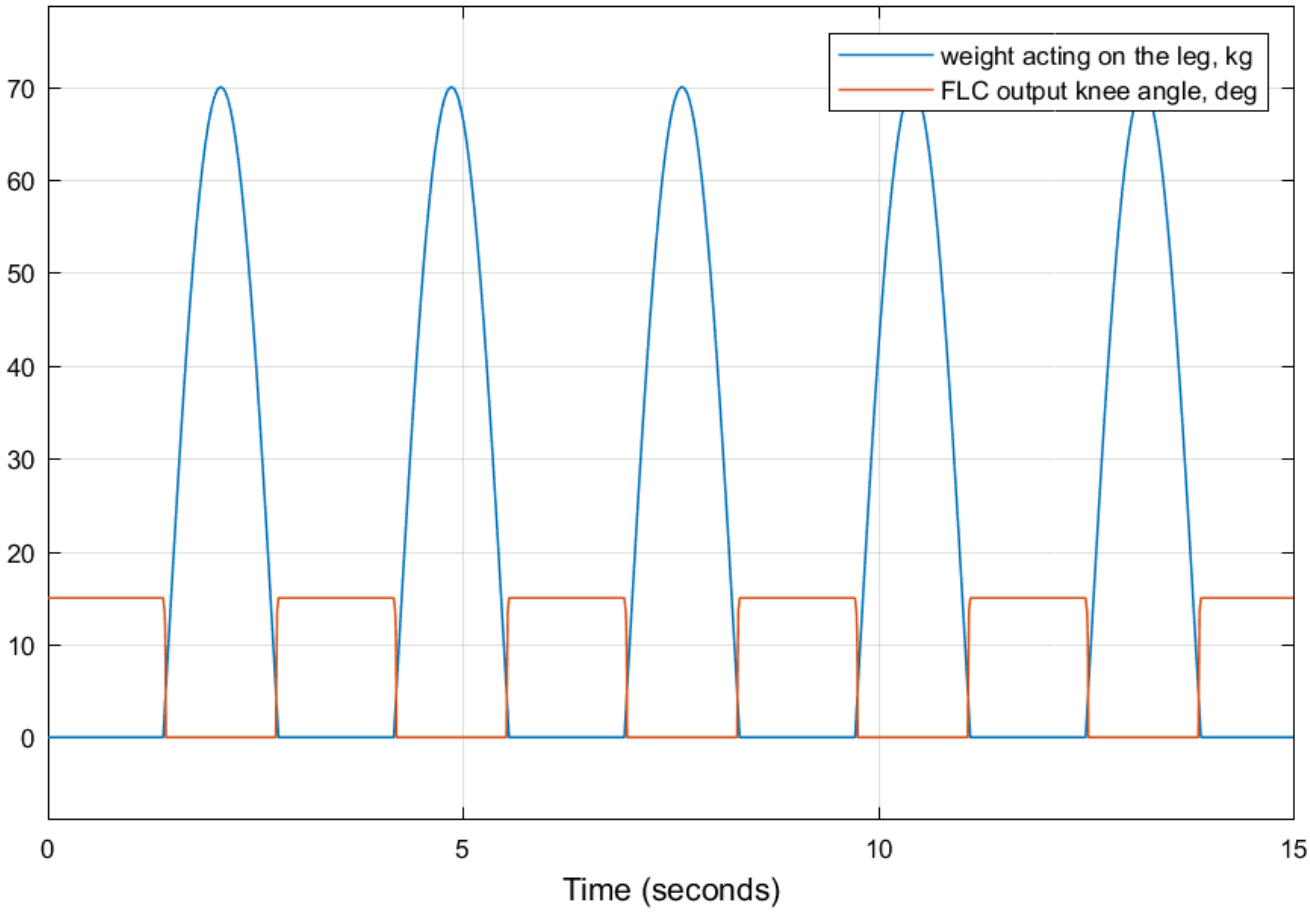

Fig. 3.2.1.2. Output of the Fuzzy logic controller with weight data included

It can be seen in fig. 3.2.1.2 that the output signal of the control unit depends on the load subjected to the prosthetic leg. During detection of weight from 2,5 to 70 kg, knee angle value tends to zero to allows patient to lean on the prosthetic leg to transfer body mass from one leg to another during gait cycle.

In conclusion, we can say, that control signal from controller depends from both input signals in equal manner.

3.2.2. SCADE control unit testing

To determine whether the model works as it was intended, we additionally developed test table 3.2.2.1, in which wrote results of checking the model by means of MC/DC criterion. For the verification of model, we entered variable values into the input of the algorithm that may impact on the resulting outcome to compare the actual result with the desired one described in commentary section.

Table 3.2.2.1.

Test table of SCADE control unit

	Input signals		Output signal	
Step	Difference angle Δ	Force load	Knee angle	Commentaries
1	-40	0	0	Force load is equal to zero, delta angle is out of the permissible limits, therefore knee angle is equal to zero.
2	-5	2	15	Force load is less than the limit value, delta angle is in range of the permissible boundaries, therefore knee angle is equal to 15
3	0	2	15	Force load is less than the limit value, delta angle is in range of the permissible boundaries, therefore knee angle is equal to 15
4	5	0	15	Force load is less than the limit value, delta angle is in range of the permissible boundaries, therefore knee angle is equal to 15

5	35	0	0	Force load is less than the limit value, delta angle is out of range of the permissible boundaries, therefore knee angle is equal to 0
6	-35	35	0	Force load is greater than the limit value, delta angle is out of range of the permissible boundaries, therefore knee angle is equal to 0
7	5	70	0	Force load is greater than the limit value, delta angle is in range of the permissible boundaries, therefore knee angle is equal to 0

3.3. Armature-controlled DC servomotor response

To determine the technical parameters of the DC servomotor based on the real appropriate device, we may determine the load torque that will act on the system. Taking into account that the DC servomotor will replace the human knee joint in the prosthetic leg. From the table 3.1.1, lower-knee parameters are following:

$$mass\ m_{lower} = 2 \text{ kg},$$

$$length\ l_{lower} = 0{,}4 \text{ m}.$$

With these specifications, determine the torque required:

$$t = l_{lower} * m_{lower} * g = 0{,}4 * 2 * 9{,}81 = 7{,}848 \text{ Nm}.$$

Assume, that form of lower-knee model has the form of thin solid cylinder. Load moment of inertia of our lower link can be determined by the following formula:

$$J_L = \frac{1}{3}mL^2 = \frac{1}{3}2 * 0{,}4^2 = 0{,}1667 \text{ kg} * \text{m}^2. \quad (3.2.1)$$

For this work, we will use the following technical characteristics of the “RE 40 ∅40 mm, graphite brushes, 150-watt DC servomotor with 24V supply”, specifications of which are given in the table 3.3.1.

Table 3.3.1

Technical specifications of the DC servomotor

№	DC servmotor parameter	Value	Unit of measurement
1	Armature resistance	0,299	Ω
2	Armature inductance	0,0000824	H
3	Torque Constant	2,4462	Nm/A
4	Constant Velocity	0,41	v/rad/s
5	Coef. Bearing friction	0,00304	Nm/(rad/sec)
6	Motor moment of inertia of load	0,0000142	kgm^2
7	Nominal current	6	A

Since the chosen motor does not have the required torque to carry the lower link part of the prosthetic leg, we may add gearbox to decrease velocity of DC servomotor and increase load torque simultaneously. Gear reductor used in the simulation is “planetary gearhead GP 42 A 2.25-11.3 Nm”. Its technical specifications are denoted in table 3.3.2.

Table 3.3.2

Technical specifications of the gearhead

№	Technical parameter	Value	Unit of measurement
1	Gear ratio	81:1	-
2	Maximum continuous torque	11,3	Nm
3	Maximum intermittent torque	16,9	Nm
4	Moment of inertia	15	$cm*g^2$

With the chosen gearhead, the parameters of the DC servomotor are recalculated in the following manner:

Torque Constant K_t equals to:

$$K_t = 0{,}0302 * 81 = 2{,}4462 \frac{\text{Nm}}{\text{A}}.$$

Velocity constant K_b is equal to:

$$K_b = \frac{33{,}196}{81} = 0{,}41 \frac{\text{v}}{\frac{\text{rad}}{\text{s}}}.$$

With nominal current of 6 A, available torque is given:

$$t_{available} = 2{,}4462 * 6 = 14{,}677 \text{ Nm}.$$

Taking into account the 91% efficiency of the gearhead, we will determine the actual load of inertia that our system is capable of handling:

$$T_{usable} = 14{,}677 * 0{,}91 = 13{,}36 \text{ Nm}$$

As a result, we determined that the usable torque provides a 33% margin required to hold the lower-knee model of the prosthetic leg.

Since the load torques generated by the servomotor shaft and gearhead are small enough, we may neglect them, taking into account only the load of inertia of the lower-knee prosthesis from (3.2.1). With new information obtained, we may analyze the transfer function of the servomotor. MATLAB code used to form mathematical models of the DC servomotor is given in Appendix 2.

Poles of the closed-loop of the DC servomotor system are following:

$$\begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} \begin{bmatrix} -3608{,}3 \\ -2{,}8 \\ -17{,}6 \end{bmatrix}.$$

The step response of the armature-controlled DC servomotor is shown in fig. 3.3.1.

Since the crucial parameter for our model is steady-state error, we may eliminate it by means of adding integral control to our system, that was developed in chapter 2.

To increase the performance of our system, we can use the pole-placement technique using Ackermann's formula. The syntax in MATLAB looks as follows:

K=acker(A,B,pl)

Where A and B are state-space matrices of the systems, and pl is the matrix of poles we need to obtain. As a result, we will get a set of gains that move the pole to specified locations.

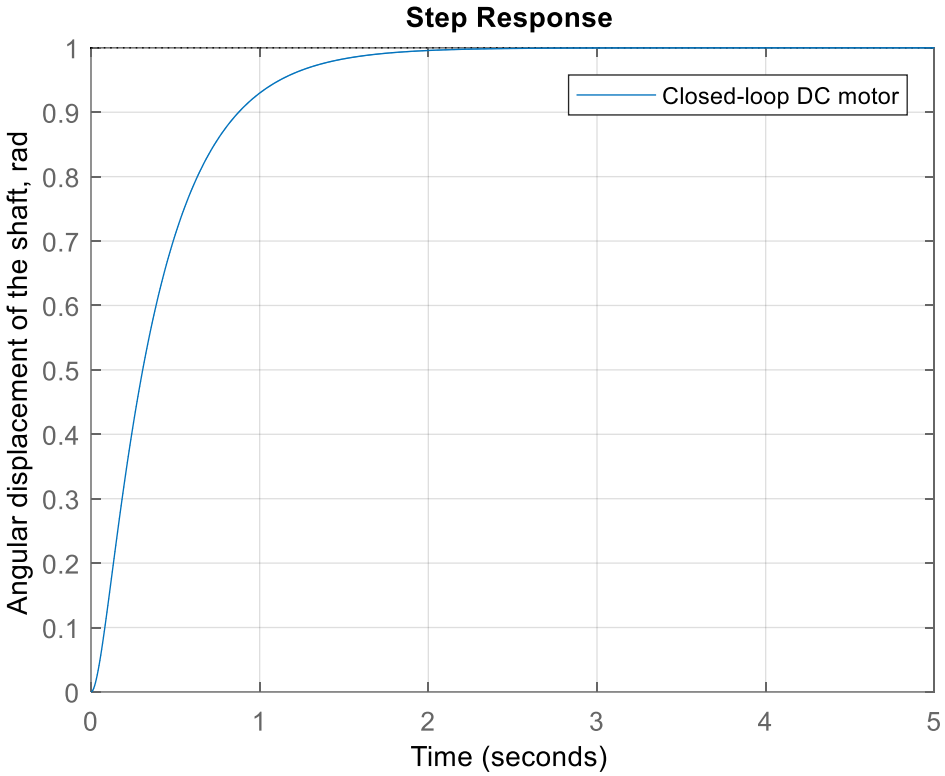

Fig. 3.3.1. Step response of closed-loop armature-controlled DC servomotor system

The transient parameters we tend to achieve are following:

1) Rise time $t_r < 0.1$ s;

2) Settling time $t_s < 0.05$ s;

3) Overshoot $M_p < 0.1$ %;

4) Steady-state error $e_{ss} = 0$.

The poles that provide the desirable performance are given below:

$$\begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix} \begin{bmatrix} -65{,}0000 + 20{,}0000i \\ -65{,}0000 - 20{,}0000i \\ -55{,}0000 + 65{,}0000i \\ -55{,}0000 - 65{,}0000i \end{bmatrix}.$$

The gain values for the feedback controller are:

$$\begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} \begin{bmatrix} 8{,}0879 \\ -0{,}2641 \\ -0{,}2792 \end{bmatrix}.$$

The gain values for integral controller:

$$[q_4][-186{,}8722].$$

The step response of the DC servomotor with full state feedback and integral controllers is following (fig. 3.3.2).

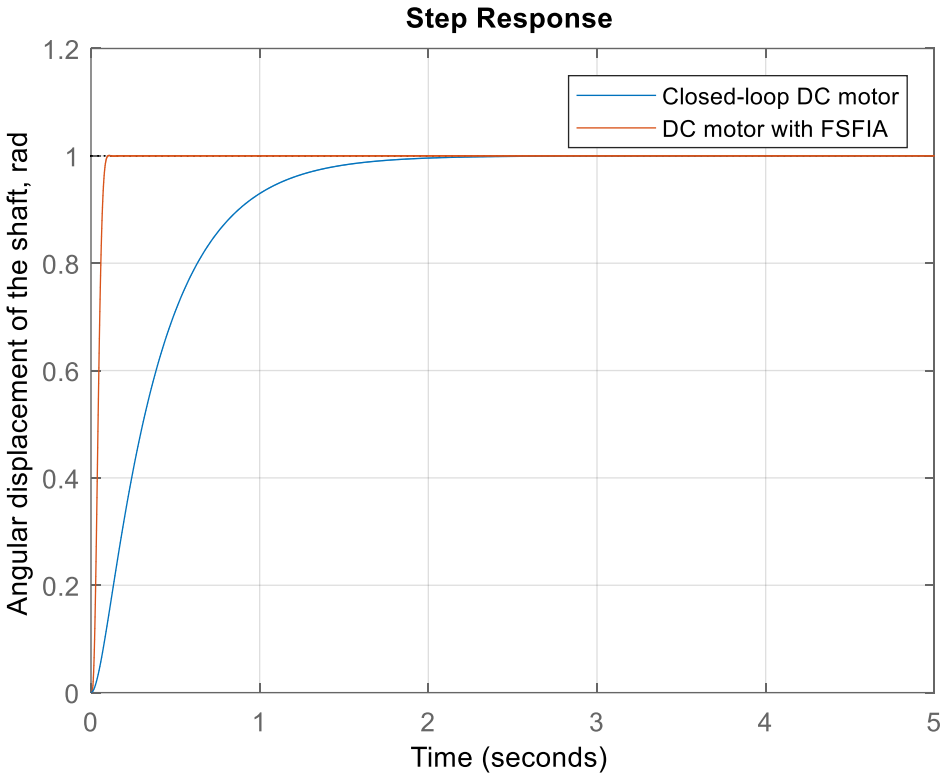

Fig. 3.3.2. Step responses of closed-loop DC servomotor and full state feedback motor with integral action

Comparison of transient parameters of both systems is given in table 3.3.3.

Table 3.3.3

Comparison of transient parameters

№	Performance characteristic	System in closed-loop	System with FSFIA
1	Steady-state time, s	1	1
2	Settling time, s	3,14	0,083
3	Rise time, s	1,76	0,0462
4	Overshoot, %	0	0,0702

The performance indices of the closed-loop system with FSFIA are following:

$$H_2\ norm = 3{,}9932,$$

$$H_\infty\ norm = 1.$$

A Bode diagram with gain and phase margins is given (fig. 3.3.3).

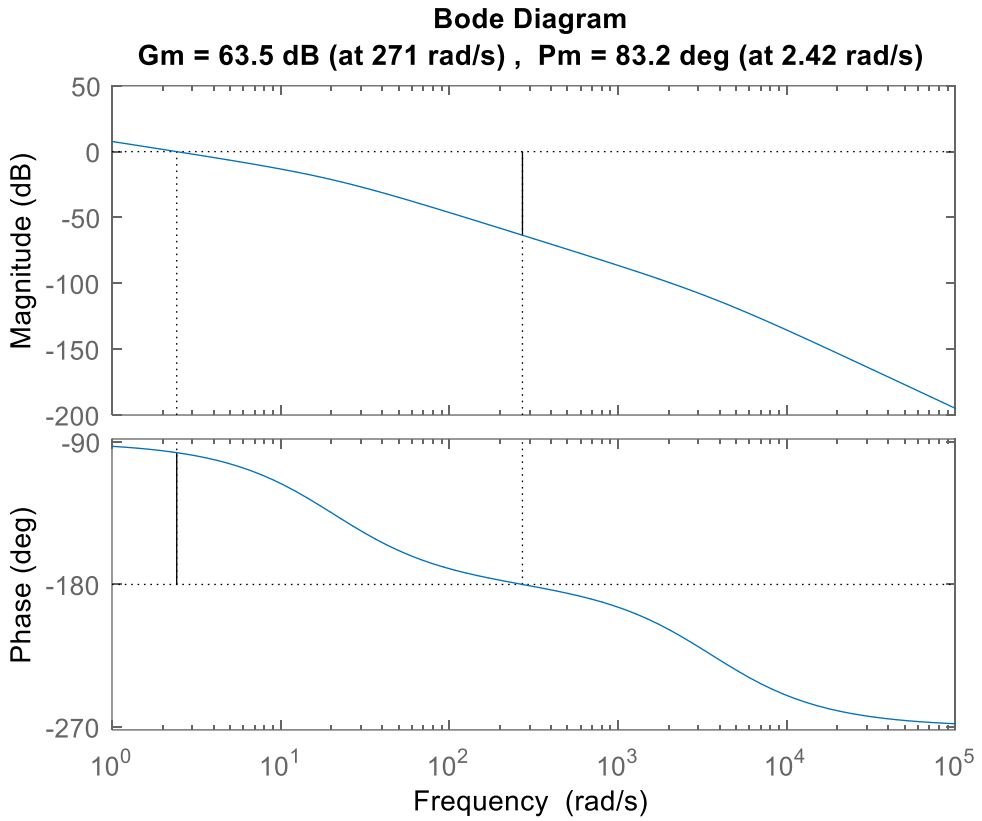

Fig. 3.3.3. Bode diagram of the DC servomotor with FSFIA

Since phase margin of the system is negative, we may additionally check the Nyquist criterion to ensure that the system is stable. Nyquist diagram of DC servomotor is given in fig. 3.3.4.

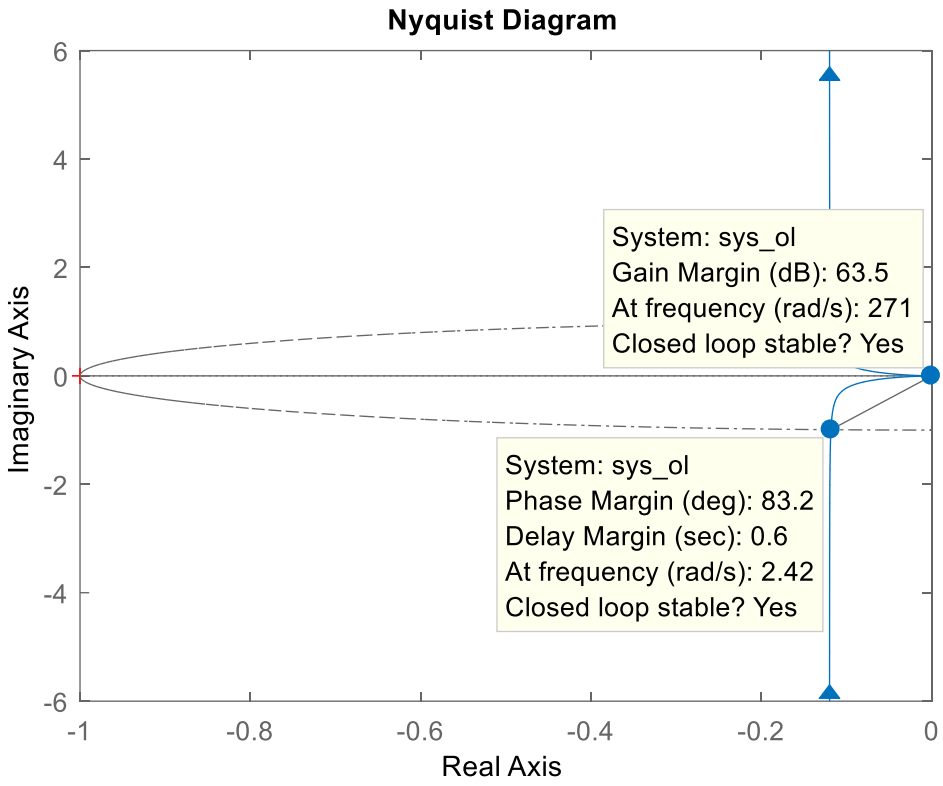

Fig. 3.3.4. Response of the FSFIA system to disturbance input

3.4. Lower-knee prosthetic leg model response

For determining the coefficients of system components, we chose human natural frequency $f_{human} = 1{,}8$ Hz and scaling factor of $a_{sf} = 1{,}4$ [6]. Table 3.4.1 of values obtained in the result of calculations is given.

Table 3.4.1

Calculated coefficients of the lower-knee model

№	Variable	Value	Unit of measurement
1	Spring coefficient	8953,7	N/m
2	Damper coefficient	1583,36	Ns/m
3	Body mass	70	kg
4	Applied force	961	N

In order to create a linear quadratic regulator for the system, we will use the ‘lqr’ operator, which has the following syntax:

lqr: [F, P, E] =lqr (A, B, Q, R), as a result of execution, we will receive:

- Matrix F is the matrix of optimal gains of the controller.

- Matrix P, which is the solution of the Riccati equation;

- Vector E that contains eigenvalues of the state matrix of the closed-loop system A-BF.

The transient parameters we tend to achieve for the spring mass damper system are following:

1) Rise time $t_r < 0{,}03$ s;

2) Settling time $t_s < 0{,}05$ s;

3) Overshoot $M_p < 0{,}05$ %;

4) Steady-state error $e_{ss} = 0$;

5) Steady-state value $v_{ss} < 0{,}1$.

These requirements were taken since we need to provide the similar by shape to weight input data for providing comfortable walking activity.

Weight matrices Q and R for linear quadratic regulator have the following values:

$$Q = \begin{bmatrix} 1000 & 0 \\ 0 & 11 \end{bmatrix}, R = [1].$$

Controller F contains the following gains:

$$F = [23{,}6523 \quad 2{,}4951].$$

Eigenvalues of system are given below:

$$\begin{bmatrix} -9{,}5680 \\ -47{,}3194 \end{bmatrix}.$$

H_2-norm and H_∞-norm of the system are:

$$H_2\ norm = 1{,}2890,$$

$$H_\infty\ norm = 0{,}2417.$$

Step response of the system has the form (fig. 3.4.1).

Table of comparison of the transient response characteristics of the lower-knee model is as follows (Table 3.4.2).

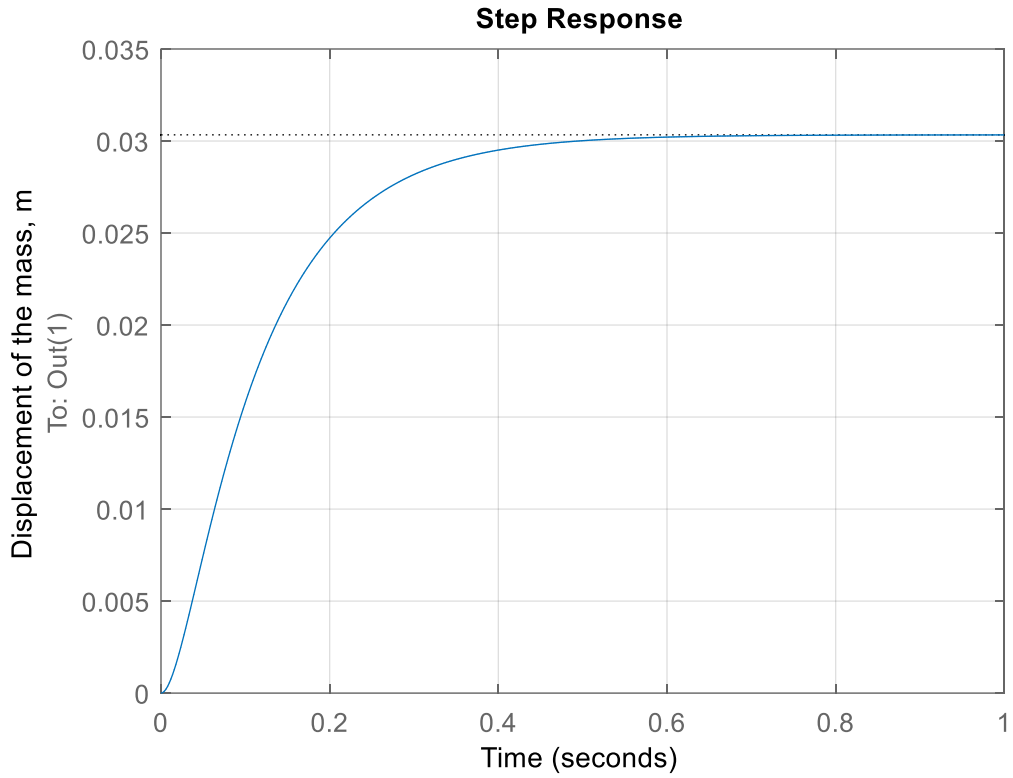

Fig. 3.4.1. Step response of the lower-knee model under the applied force

Table 3.4.2

Comparison of transient parameters of the lower-knee model

№	Performance characteristic	Response characteristics of the system without the LQR controller	Response characteristics of the system with the LQR controller
1	Steady-state value	0,107	0,0303
2	Settling time, s	0,516	0,432
3	Rise time, s	0,297	0,238
4	Overshoot, %	0	0

Response of the lower-knee model to continuous forced response is given in fig. 3.4.2.

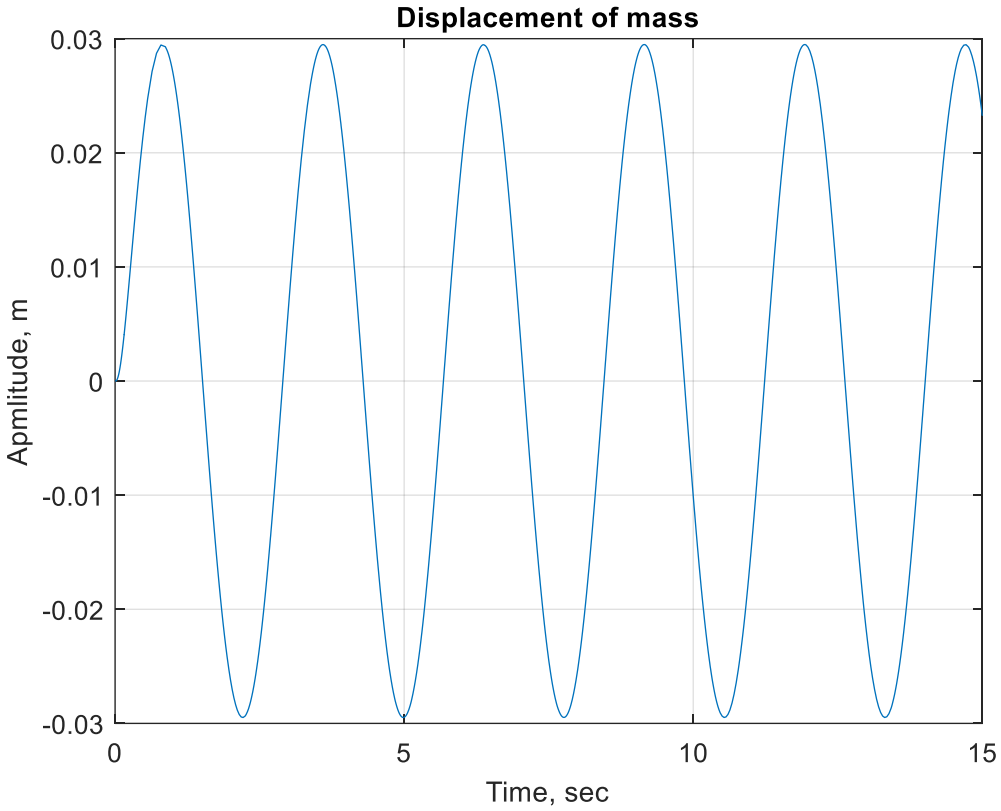

Fig. 3.4.2. Response of the lower-knee model on forced response

3.5. Conclusion

In this chapter, we simulated derived models from the chapter 2 in MATLAB and SIMULINK environments. For input data simulation we used parameters of the user from tables 3.1.1, 3.1.2 and 3.4.1. Using input data, examined influence of input signals on the output of the FLC. According to fig. 3.2.1.1 and 3.2.1.2, output system of the FLC is formed according to the rule base to fit the gait cycle of the user. For the DC servomotor parameters, we considered characteristics of real application taking into account the load mounted on the shaft in the form of solid cylinder that replicates the lower knee prosthetic leg, chose poles of the system that fit to requirements for DC servomotor response. For lower knee prosthetic leg model, we determined spring and damper coefficients with LQR controller implemented into the system.

CHAPTER 4
PROSTHESIS CONTROL SYSTEM SYNTHESIS

4.1. SIMULINK model of lower-limb prosthesis

Consider the designed control system, which results from this qualification paper. The system was formed using SIMULINK software, and its diagram is given (fig. 4.1.1).

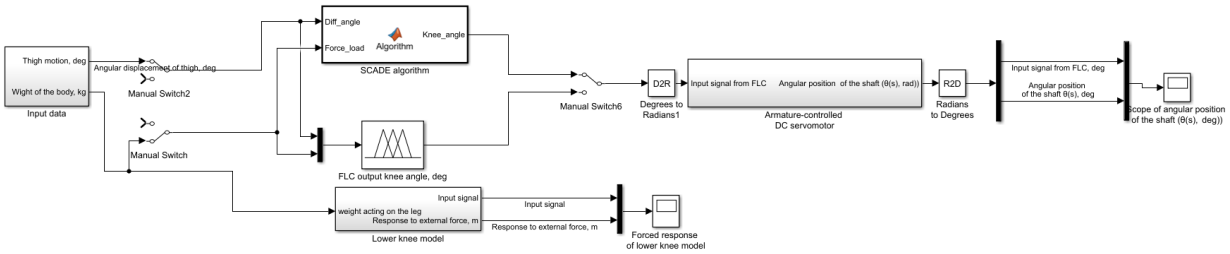

Fig. 4.1.1. Lower limb prosthesis control system

The following control system consists of:

1) Input data block containing thigh motion and data about the weight acting on the prosthetic leg (fig. 4.1.2).

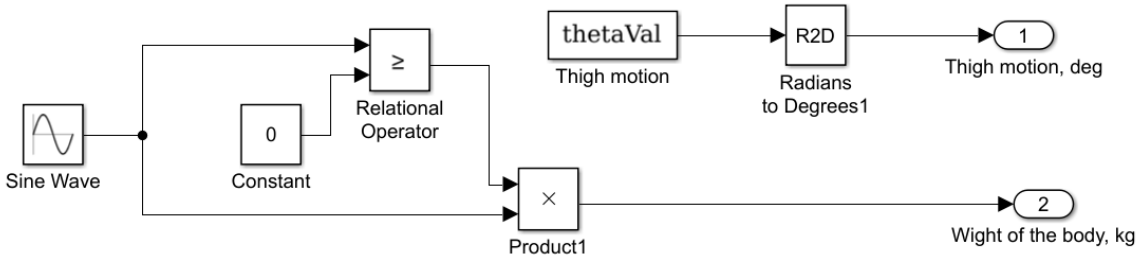

Fig. 4.1.2. Input data SIMULINK block

2) Fuzzy logic controller connected in parallel with the SCADE control unit.

3) Armature-controlled DC servomotor system. Its structural diagram is shown in fig. 4.1.3.

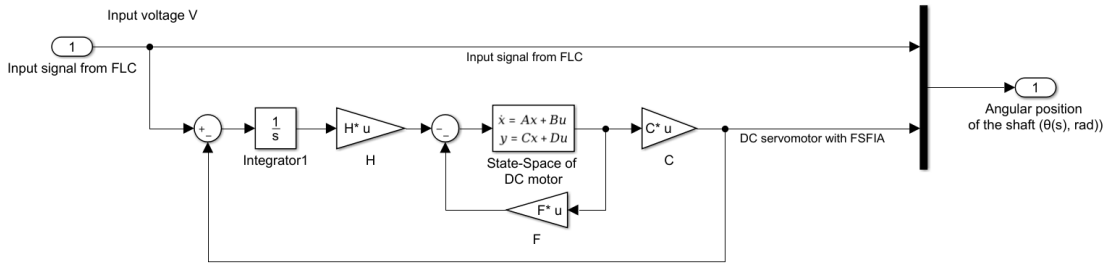

Fig. 4.1.3. Armature-controlled DC servomotor Simulink diagram

4) Lower knee system, diagram of what is given (fig. 4.1.4).

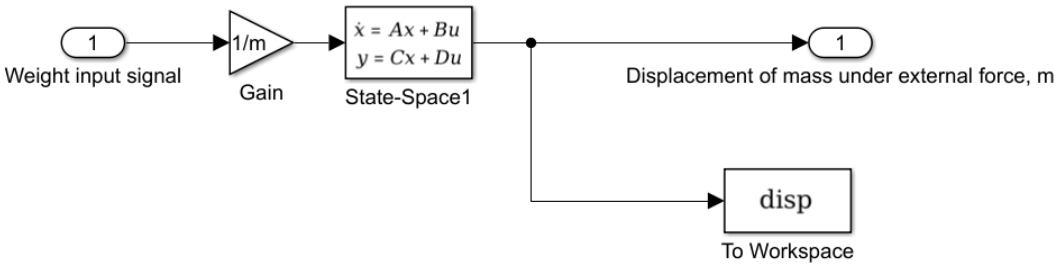

Fig. 4.1.4. Lower knee system SIMULINK diagram

4.2. Control system simulation analysis

As can be seen from fig. 4.1.1, the input data of our system are the angular displacement of the thigh and weight acting as an external force on the lower knee model. Response of DC servomotor to input data is given in fig. 4.2.1.

We see from fig. 4.2.1, the input data in the form of the weight of the patient and the motion of its thigh impact on the final output of the logic controller in a way, that displacement of the rotor shaft can be achieved only during the swing phase and with weight that acts on the prosthetic leg. When weight progresses from 0 to 70 kg, the shaft rotor does not move, and the leg is kept straight. If weight is absent or small enough, the shaft of the rotor is moved accordingly to the position of the thigh. According to fuzzy

logic controller, the replicated knee joint in the form of a servomotor is displaced from 0 to 15 degrees.

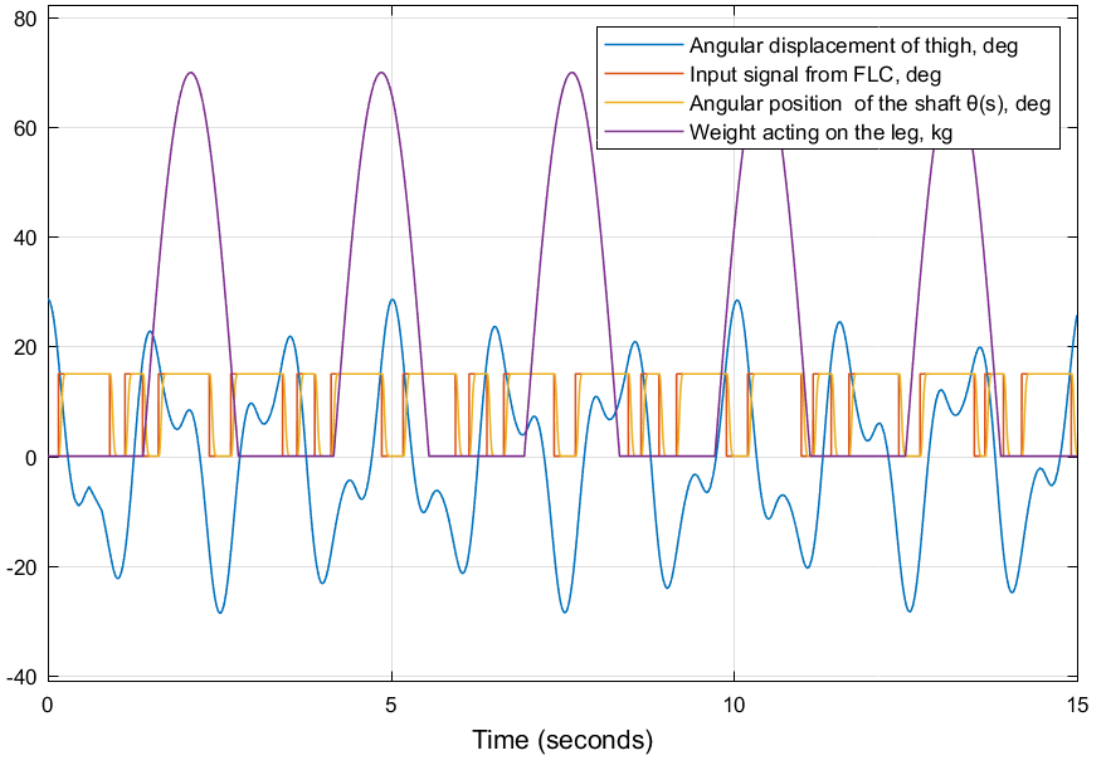

Fig. 4.2.1. Response of DC servomotor to input data

Comparison of fuzzy logic and SCADE controllers is given in fig. 4.2.2.

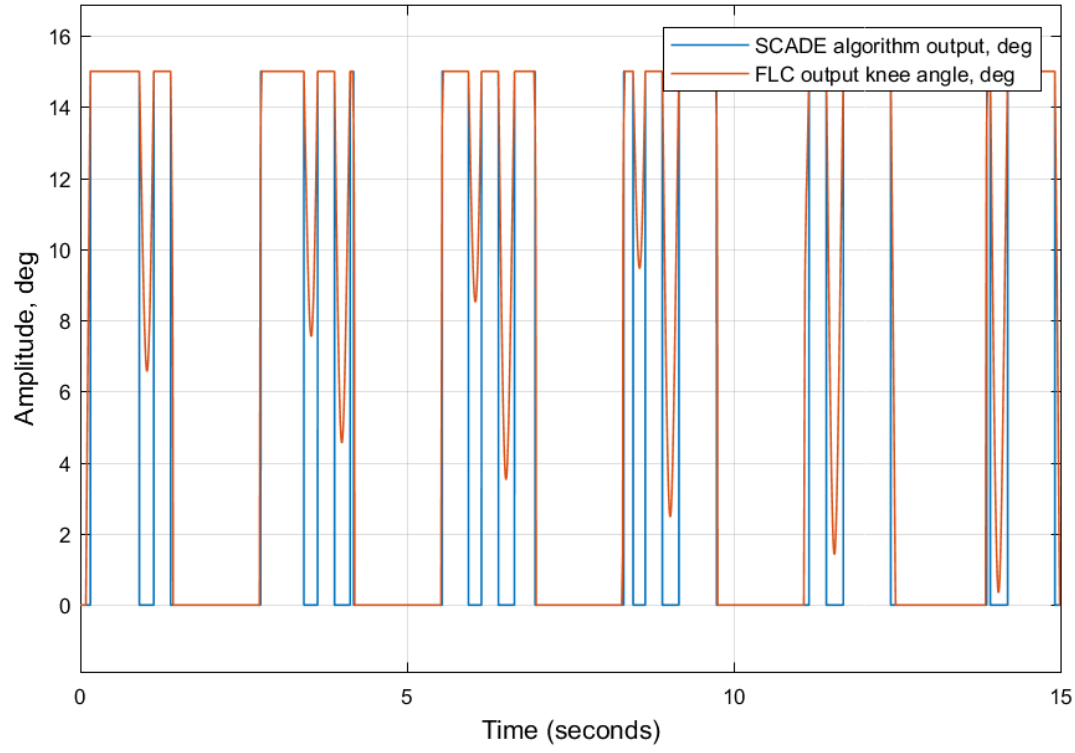

Fig. 4.2.2. Comparison of output signals of controllers

From comparison we see, that both control units handle the input signal coming from the control unit, but fuzzy controller can provide more exact signal based on its more detailed description of system.

For the lower-knee model, the output result is displayed in fig. 4.2.3.

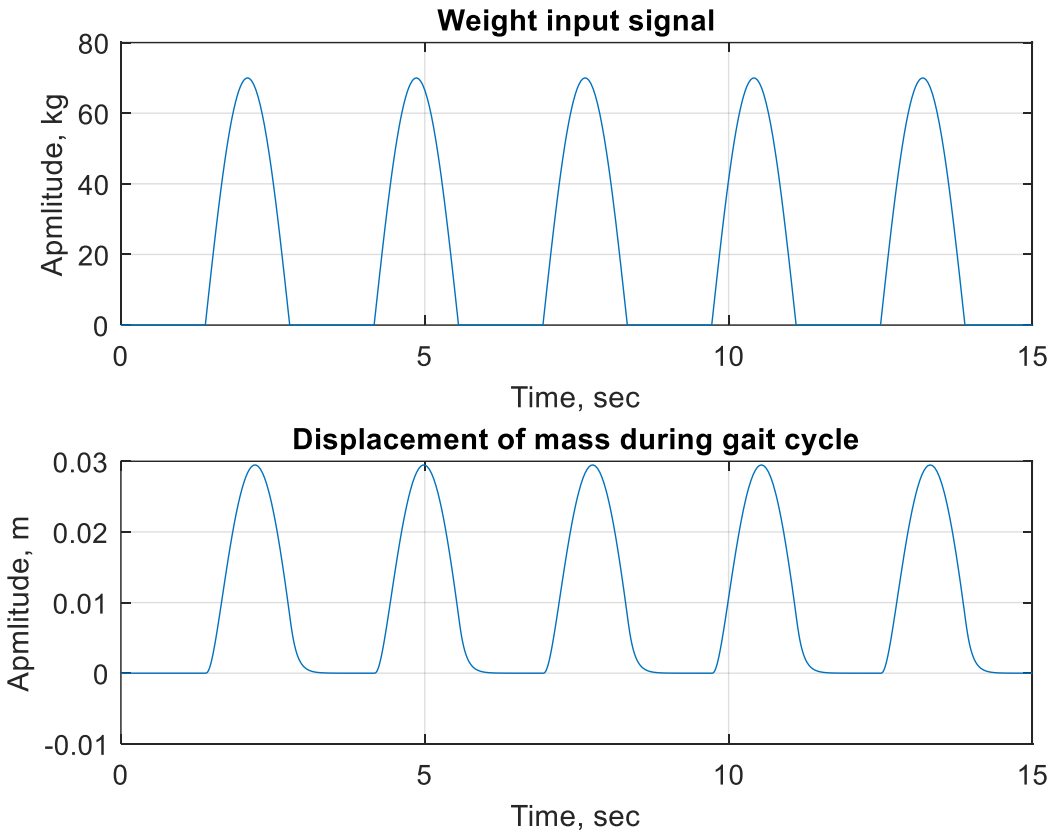

Fig. 4.2.3. Response of the lower-knee model to the weight of the patient

Comparing curves representing load cell data being subjected to weight and displacement of mass of patient during walking activity, we may see that form and peak value of both curves coincide in time for synchronizing of both curves in time to ensure that gradual suppression of spring-damper system under impact of body weight.

4.3. Conclusion

In the chapter, we developed lower limb prosthesis using components developed and simulated in chapters 2 and 3 in SIMULINK and MATLAB environments. In fig. 4.1.1 the control system is given consisting of input data block, FLC and SCADE models

connected in parallel, armature-controller DC servomotor and lower knee model. In order to examine interaction of SCADE algorithm with other components, we converted C code from SCADE Suite into the MATLAB function. Technical characteristics and parameters for each part were taken from the results of simulation in chapter 3. Also, we observed output signals of each block which can be shown in fig. 4.2.1-4.2.3.

CONCLUSIONS

In the qualification paper with the topic “Control system of prosthetic leg”, we:

1) Got acquainted with the typical designs of prostheses, their types, classification, and main components.

2) Developed the structure of the control system of the prosthetic leg and its main elements, such as:

a) input simulation data of thigh movement and weight that impact the leg,

b) fuzzy logic controller.

c) SCADE control unit.

d) DC servomotor mimicking knee joint operation.

e) Lower-knee model that provides comfortable interaction with the surface.

2) Derived set of ordinary differential equations describing thigh movement based on the double pendulum model.

3) Defined input body mass data as the simulation of the force cell during walking activity.

4) Developed Fuzzy logic controller that produces output in the form of knee angle, which is used to rotate the motor shaft, taking as input the upper link angular displacement and body mass impacting the leg.

5) Synthesized SCADE controller as reserve control system in order to use it in case of failure of the fuzzy one. From the graphs obtained, we determined that both control units take into account both input variables to form an appropriate output.

6) Derived mathematical model of armature-controlled DC servomotor, for obtaining gaster response of the system to the reference signal, developed a full state feedback controller with integral one. Simulated control system based on the parameters of real-life applications. Chose poles based on the transient parameters’ requirements. The transient response parameters obtained as a result of calculations are as follows:

a) Rise time $t_r = 0{,}0462$ s;

b) Settling time $t_s = 0{,}083$ s;

c) Overshoot M_p= 0,0702 %;

d) Steady-state error e_{ss} = 0.

H_2-norm of the obtained system is equal to 3,9932, and H_∞-norm is equal to 1.

7) Developed a mathematical model of the lower-knee model with calculated coefficients of the damper and spring to provide a comfortable walking activity for the patient. Formed for this model LQR controller increases the performance of the system with the performance indices. Transient response parameters gain the following values:

a) Rise time t_r =0,238 s;

b) Settling time t_s = 0,432 s;

c) Overshoot M_p= 0 %;

d) Steady-state value v_{ss} = 0,0303 m.

H_2-norm is equal to 1,2890, and H_∞-norm takes the value of 0,2417.

8) Synthesized the control system of a prosthetic leg for patient with weight of 70 kg and height of 180 cm to provide comfortable walking activity by means of MATLAB, SIMULINK and SCADE software, depicted in fig. 4.1.1. Although the system can be improved, for the current moment the developed system be used as basis for creating real-time applications since it includes main elements that are used in lower limb prosthetic devices.

LIST OF REFERENCES

1. Wheatstone Bridge Circuit [Electronic resource] // Access mode: https://www.grc.nasa.gov/www/k-12/airplane/tunwheat.html

2. International Standard - Medical device software – Software life cycle processes. // IEC 62304:2022.

3. Gabriela Gonz´alez. Single and Double plane pendulum [Electronic resource] / Gabriela Gonz´alez // Access mode: https://www.phys.lsu.edu/faculty/gonzalez/Teaching/Phys7221/DoublePendulum.pdf

4. Akash, Raihan Khan. Comparative Analysis of Control Strategies for Position Regulation in DC Servo Motors [Electronic resource] / Akash, Raihan Khan //ArXiv – 2025 - abs/2501.11820. // Access mode: https://www.semanticscholar.org/paper/Comparative-Analysis-of-Control-Strategies-for-in-Akash/1e6fb11e4cc6afb752ea293ceebdc9f1f4835ad8

5. How to choose the desired poles for this system [MIMO closed system] [Electronic resource] // Access mode: https://technicalsource9.medium.com/how-to-choose-the-desired-poles-for-this-system-mimo-closed-system-6c0d9b9ed53b

6. Fauzan Dharmawan. Modeling Ideal k and c Value for Lower Knee Prosthetic Leg Using Spring-Damper Model with MATLAB [Electronic resource] / Fauzan Dharmawan, Putri Wulandari and Ahmad Chirzun // Access mode: https://easychair.org/publications/preprint/zv6G/open

7. Tian, Shuo. Research on the mechanical structure and control system of prostheses based on intelligent solutions [Electronic resource] / Tian, Shuo // Theoretical and Natural Science – 2023 - 19. 102-110. 10.54254/2753-8818/19/20230508. // Access mode: https://www.researchgate.net/publication/376368494_Research_on_the_mechanical_structure_and_control_system_of_prostheses_based_on_intelligent_solutions

8. Ngo, Hoang-Trung. Design and control of an active prosthetic leg / Hoang-Trung & Ngo, Thien & Nguyen, Danh Ngoc & Le, Hoai-Nam // 2023 – access mode:

https://www.researchgate.net/publication/323956385_Design_and_control_of_an_active_prosthetic_leg

9. S R Adarsh. Model identification and control of prosthetic leg [Electronic resource] / S R Adarsh, S Selvakumar // J. Phys.: Conf. – 2020 - Ser. 1706 012109 // Access mode: https://iopscience.iop.org/article/10.1088/1742-6596/1706/1/012109

10. Kurniawan. Electric Bionic Legs Used Gyroscope And Accelerometer With Fuzzy Method. [Electronic resource] / Kurniawan, Rifky & Setiyoko, Annas & Jamiin, Mohammad & Herijono, Budi & Ramadhan, Insan & Astuti, Wijiani & Rizaldi, Febri & Aji, Mahdi & Syai'in, Mat & Adianto, Adianto & Indarti, Rini & Rinanto, Noorman.// 2019 - 1-5. 10.1109/ISESD.2019.8909541. // Access mode: https://www.researchgate.net/publication/337535706_Electric_Bionic_Legs_Used_Gyroscope_And_Accelerometer_With_Fuzzy_Method

11. Naufal Rahmat Setiawan. DC Motor Controller Using Full State Feedback [Electronic resource] / Setiawan, Naufal Rahmat & Ma'arif, Alfian & Widodo, Nuryono // Control Systems and Optimization Letters – 2023 - Vol. 1, No. 1, 2023. ISSN: 2985-6116 // Access mode:

https://www.researchgate.net/publication/369534681_DC_Motor_Controller_Using_Full_State_Feedback

12. Elsayed, Shereen. THEORETICAL AND EXPERIMENTAL INVESTIGATION OF DC SERVO-MOTOR [Electronic resource] / Dessouky, Sobhy & Mostafa, Hussam // Port-Said Engineering Research Journal – 2014 - 18. 25-31. 10.21608/pserj.2014.45262. Access mode: https://www.researchgate.net/publication/335359647_THEORETICAL_AND_EXPERIMENTAL_INVESTIGATION_OF_DC_SERVO-MOTOR

13. Olatomiwa, Lanre. Design and Comparative Assessment of State Feedback Controllers for Position Control of 8692 DC Servomotor [Electronic resource] / Olatomiwa, Lanre & Sanusi, Kamilu & Abdulhakeem, Mohammad // International Journal of Intelligent Systems and Applications – 2015 - 7. 28-33. 10.5815/ijisa.2015.09.04. // Access mode:

https://www.researchgate.net/publication/280599178_Design_and_Comparative_Assessment_of_State_Feedback_Controllers_for_Position_Control_of_8692_DC_Servomotor

14. Iswanto. Pole Placement Based State Feedback for DC Motor Position Control [Electronic resource] / Iswanto, Nia Maharani Raharja, Alfian Ma'arif, Yogi Ramadhan and Phisca Aditya Rosyady // J. Phys.: Conf. – 2020 - Ser. 1783 012057. // Access mode: https://iopscience.iop.org/article/10.1088/1742-6596/1783/1/012057

15. Okubanjo, Ayodeji. Performance Evaluation of PD and LQR Controller for Coupled Mass Spring Damper System [Electronic resource] / Okubanjo, Ayodeji & Oluwadamilola, Oyetola & Ade-Ikuesan, O.O. & Olaluwoye, Olawale // 2018 - 4. 199-210. // Access mode: https://www.researchgate.net/publication/327060361_Performance_Evaluation_of_PD_and_LQR_Controller_for_Coupled_Mass_Spring_Damper_System

16. Amiot, David. Development of a Passive and Slope Adaptable Prosthetic Foot [Electronic resource] / Amiot, David & Schmidt, Rachel & Law, Angwei & Meinig, Erich & Yu, Lynn & Olesnavage, Kathryn & Prost, Victor & Winter, Amos // 2017 - V05AT08A066. 10.1115/DETC2017-67947. // Access mode: https://www.researchgate.net/publication/320851533_Development_of_a_Passive_and_Slope_Adaptable_Prosthetic_Foot

17. Okubanjo, Ayodeji. Performance Evaluation of PD and LQR Controller for Coupled Mass Spring Damper System [Electronic resource] / Okubanjo, Ayodeji & Oluwadamilola, Oyetola & Ade-Ikuesan, O.O. & Olaluwoye, Olawale // 2018 - 4. 199-210. // Access mode: https://www.researchgate.net/publication/327060361_Performance_Evaluation_of_PD_and_LQR_Controller_for_Coupled_Mass_Spring_Damper_System

APPENDIX 1

MATLAB script "DiffEqOfHumanLeg"

```
%% Initial data
tspan = [0 15];
dt = 0.01;

g = 9.81;
L1 = 0.4;  L2 = 0.4;
m1 = 2.0;  m2 = 2.0;

% Initial Conditions
th1_0 = 0.5;  th2_0 = 0.2;
th1dt_0 = 0;  th2dt_0 = 0;

% Use ode45
options = odeset('RelTol', 1e-6, 'AbsTol', 1e-6);
[t,x] =
ode45(@(t,y)doublePendulumODE(t,y,g,L1,L2,m1,m2),tspan,[th1
_0,th1dt_0,th2_0,th2dt_0],options);

% Extract and Convert Data
th1 = x(:,1);
th2 = x(:,3);

x1 = L1/2 * sin(th1);
y1 = -L1/2 * cos(th1);
x2 = x1 + L2/2 * sin(th2);
y2 = y1 - L2/2 * cos(th2);

%extract data for simulation the input data to fuzzy
controller
thetaVal = [t th1]
plot(t,th1*180/pi)
title('Motion of thigh in space')
xlabel('Time, s')
ylabel('Angular displacement, deg')

function dydt = doublePendulumODE(t, x, g, L1, L2, m1, m2)
    % Unpack the state variables
    th1 = x(1); th1dt = x(2);
    th2 = x(3); th2dt = x(4);

    % Equations for the describing motion of the leg
```

```
    den1 = (L1*(2*m1+m2-m2*cos(2*th1-2*th2)))
    den2 = (L2*(2*m1+m2-m2*cos(2*th1-2*th2)))
    dydt = zeros(4, 1);
    dydt(1) = th1dt;
    dydt(2) = (-g*(2*m1+m2)*sin(th1)-m2*g*sin(th1-2*th2)-
2*sin(th1-th2)*m2*(th2dt^2*L2+th1dt^2*L1*cos(th1-
th2)))/den1;
    dydt(3) = th2dt;
    dydt(4) = (2*sin(th1-
th2)*(th1dt^2*L1*(m1+m2)+g*(m1+m2)*cos(th1)+th2dt^2*L2*m2*c
os(th1-th2)))/den2;
end
```

APPENDIX 2
MATLAB script "DCmotorWithFSFIA"

```
%% Initial data:
Ra = 0.299;                 % Ohms
La = 8.2400e-05;            % H
Kt = 2.4662;                % Nm/A
Kb = 0.41;                 % v/rad/s
b = 0.00304;                % Nms
J = 0.1668;              % kg*m^2;

%% state-space model of the DC servomotor
A = [0 1 0
     0 -b/J Kt/J
     0 -Kb/La -Ra/La];
B = [0; 0; 1/La];
C = [1 0 0];
D = 0;

sys_ol = ss(A,B,C,D);
sys_cl_wi = feedback(sys_ol, 1);
figure; step(sys_cl_wi)
ctrbm = ctrb(sys_cl_wi)
%% system with integral action:
Ai = [A zeros(3,1); -C 0];
Bi = [B;0];
Ci = [C 0];

%% Pole placement by means of the Ackermann formula:
% desired poles are following:
pcl = [-55-65i, -55+65i, -65-20i, -65+20i]
kx = acker(Ai,Bi, pcl);
F = kx(1:3); H = kx(4);
Acl = Ai-Bi*kx;
Br = [0;0;0;1];

%% closed-loop system:
sys_cl = ss(Acl, Br, Ci, D)
hold on
step(sys_cl, 5)
legend('Closed-loop DC motor', 'DC motor with FSFIA')
hold off

%% from ss to tf:
[num, den] = ss2tf(Ai, Bi, Ci, D)
```

```
systf = tf(num,den)
figure
rlocus(num,den)
%% 3. comparison of poles
disp('Poles of the closed-loop DC servomotor system are:')
pole(sys_cl_wi)
disp('Poles of the DC servomotor system with full state
feedback with integral action are:')
pole(sys_cl)
figure(3)
nyquist(sys_cl)
%%
disp('H2-norm of the closed-loop DC servomotor system
are:')
normh2(sys_cl_wi)
disp('H2-norm of the DC servomotor system with full state
feedback with integral action are:')
normh2(sys_cl)

%%
disp('Hinf-norm of the closed-loop DC servomotor system
are:')
hinfnorm(sys_cl_wi)
disp('Hinf-norm of the DC servomotor system with full state
feedback with integral action are:')
hinfnorm(sys_cl)
```

APPENDIX 3
MATLAB script "LowerKneeModel"

```
%% initial data
%% transfer function of spring mass damper system of the
lower link of the prosthetic leg:
m=70 % kg
fh = 1.8 % Hz for walking activity
fsf = 1.4 % scaling factor walking activity for adjusting
the force
f = (fsf*m)*9.81 % external force acting on the system
k = ((2*pi*fh)^2)*m
b = 2*sqrt(k*m)
omega=(k/m)^(1/2)
s = tf('s');

Gs = tf([f], [m b k]);
step(Gs), grid
%% state space of model:
Aspd = [0 1; -k/m -b/m]
Bspd = [0; f/m]
Cspd = [1 0]
Dspd = [0]
sys = ss(Aspd,Bspd,Cspd,Dspd)
step(sys)

%% lqr regulator development:
C2=[Cspd; zeros(1,1),1];
D2=zeros(2,1);
sys_ext=ss(Aspd,Bspd,C2, D2)

W = [1000 11];
R1=diag (W);
R2=1;
[Fc, Pc, Ec] =lqr (Aspd, Bspd, R1, R2);

lqr_sys = feedback(sys_ext, Fc);
step(lqr_sys)
```

APPENDIX 4
SCADE function implemented in SIMULINK

```
function Knee_angle = Algorithm(Diff_angle,Force_load)
    if Force_load > 2.5
        Knee_angle = 0.0;
    else
        if Diff_angle > -17.5
            if Diff_angle < 17.5
               Knee_angle = 15.0;
            else
                Knee_angle = 0.0;
            end
        else
            Knee_angle = 0.0;
        end
    end
end
```