



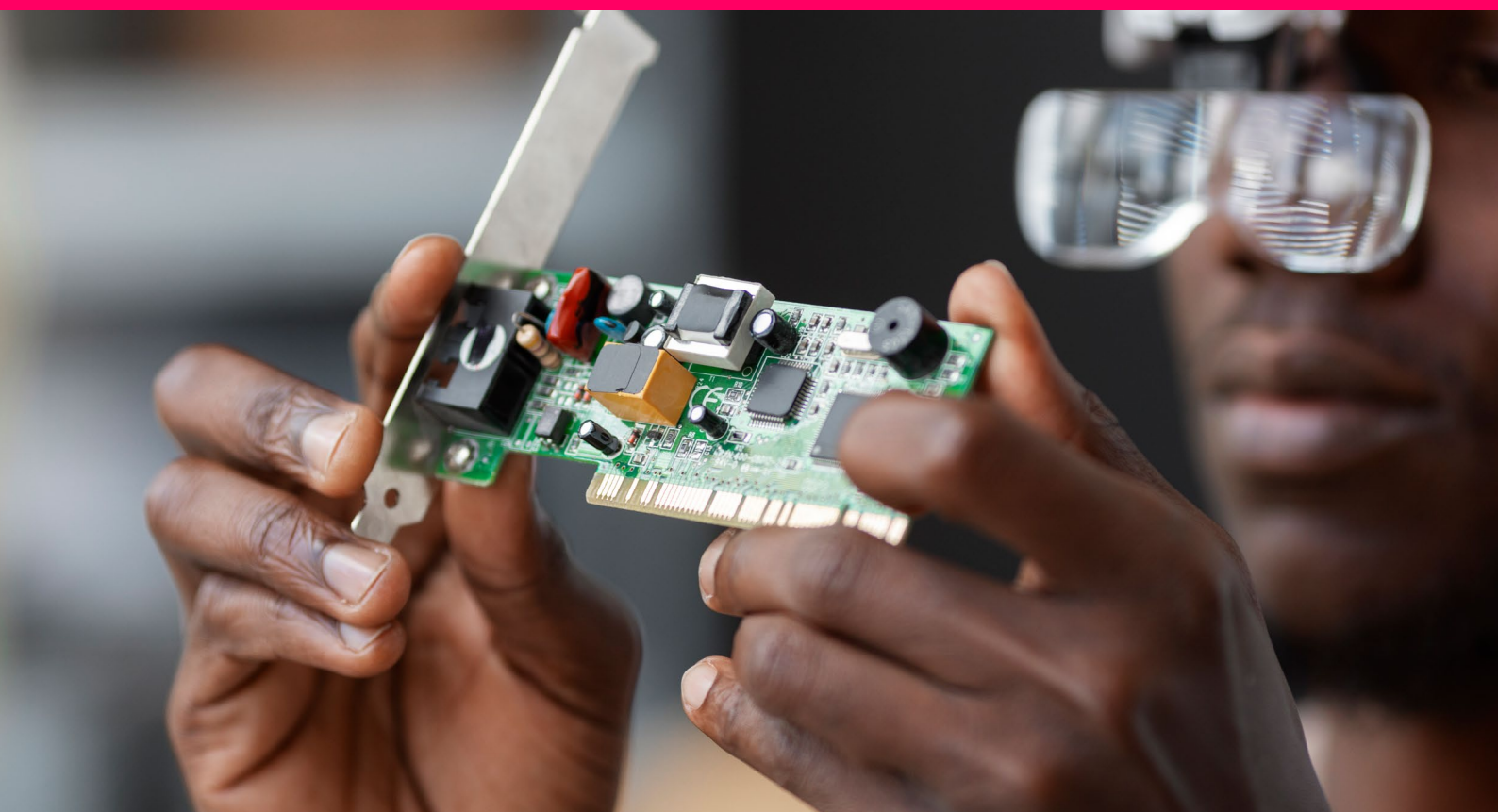
**MINISTRY OF EDUCATION  
GHANA ASSOCIATION OF  
OF SCIENCE TEACHERS**



# **Robotics**

## **for Senior High Schools**

**Year 2**



**Kwame Oteng Gyasi  
Isaac Nzolely**

# MINISTRY OF EDUCATION GHANA ASSOCIATION OF OF SCIENCE TEACHERS

## Robotics for Senior High Schools

Year 2

Kwame Oteng Gyasi  
Isaac Nzoley



Ghana Education  
Service (GES)





© Ministry of Education 2025

This publication is not for sale. All rights reserved. No part of this publication may be reproduced without prior written permission from the Ministry of Education, Ghana.

ISBN:

# Contents

<b>FOREWORD .....</b>	<b>6</b>
<b>SECTION 1 ROBOTS AND SOCIETY 2 .....</b>	<b>7</b>
<b>PRINCIPLES OF ROBOTIC SYSTEMS.....</b>	<b>8</b>
EXPLORING THE BROADER SOCIETAL IMPACT OF ROBOTS.....	8
EXPLORING CAREER OPPORTUNITIES IN ROBOTICS.....	14
<b>SECTION 2 ROBOT CONTROL PRINCIPLES 2.....</b>	<b>27</b>
<b>PRINCIPLES OF ROBOTIC SYSTEMS.....</b>	<b>28</b>
SYSTEMATIC DESIGN METHODS FOR NON-FEEDBACK CONTROL SYSTEMS..	29
SYSTEMATIC DESIGN METHODOLOGY FOR FEEDBACK CONTROL SYSTEMS	34
LOGICAL AND MATHEMATICAL MODELLING OF CONTINUOUS-TIME MA-	
CHINES.....	39
BUILDING LOGIC WITH FINITE STATE MACHINES .....	45
<b>SECTION 3 SENSORS &amp; ACTUATORS 2 .....</b>	<b>58</b>
<b>PRINCIPLES OF ROBOTIC SYSTEMS.....</b>	<b>59</b>
UNDERSTANDING ANALOGUE AND DIGITAL SENSORS .....	59
MATHEMATICAL METHODS FOR SENSOR DATA PROCESSING IN ROBOTICS	
INTRODUCTION.....	70
ROTATION-DISTANCE RELATIONSHIP IN WHEELED ROBOTS.....	77
Graphing and Understanding the Data.....	80
<b>SECTION 4 DIGITAL AND ANALOGUE SYSTEM DESIGN .....</b>	<b>88</b>
<b>ROBOT DESIGN METHODOLOGIES .....</b>	<b>89</b>
INTRODUCTION TO BOOLEAN ALGEBRA .....	89
ADVANCED DIGITAL SYSTEM OPTIMISATION WITH KARNAUGH MAPS .....	110
BUILDING AND TESTING BASIC COMBINATIONAL CIRCUITS ON PRINTED	
CIRCUIT BOARDS .....	130
<b>SECTION 5 TOOLS AND APPS FOR ROBOT DESIGN 2 .....</b>	<b>146</b>
<b>ROBOT DESIGN METHODOLOGIES .....</b>	<b>147</b>
UNDERSTANDING COMPUTER-AIDED DESIGN (CAD) .....	148
EXPLORING 3D PRINTING IN ROBOTICS.....	154
<b>SECTION 6 HIGHER ORDER DESIGN THINKING .....</b>	<b>167</b>
<b>ROBOT CONSTRUCTION &amp; PROGRAMMING .....</b>	<b>168</b>
REVIEWING KEY CONCEPTS.....	168
Introduction to Advanced Control and Feedback Systems.....	170
DEVELOPING ADVANCED ALGORITHMS.....	172
PSEUDOCODE DEVELOPMENT .....	173
CREATING DETAILED FLOWCHARTS.....	175



<b>SECTION 7 ROBOT CONSTRUCTION .....</b>	<b>181</b>
<b>ROBOT CONSTRUCTION &amp; PROGRAMMING .....</b>	<b>182</b>
UNDERSTANDING ROBOT MOVEMENT: CLOSED CHAINS, VELOCITY, AND TRAJECTORY. ....	182
PERFORMING BASIC CALCULATIONS INVOLVING VELOCITY AND TRAJECTORY MOTIONS AND APPLYING TRAJECTORY CALCULATION TO ROBOT NAVIGATIONS .....	199
CREATING ROBOTS USING ROBOTIC KITS AND LOCAL MATERIALS .....	206
MOVING WITHOUT TYRES IN ROBOTICS.....	222
<b>SECTION 8 PROGRAMMING ROBOTS 2 .....</b>	<b>231</b>
<b>ROBOT CONSTRUCTION &amp; PROGRAMMING .....</b>	<b>232</b>
CONTROL SYSTEMS .....	232
A SIMPLE GUIDE TO TROUBLESHOOTING IN ROBOTICS .....	239
FIXING IDENTIFIED PROBLEMS.....	241
REFERENCES .....	248
GLOSSARY .....	250
Acknowledgements.....	253

# FOREWORD

Ghana's new Senior High School Curriculum aims to ensure that all learners achieve their potential by equipping them with 21st Century skills, knowledge, character qualities and shared Ghanaian values. This will prepare learners to live a responsible adult life, progress to further studies and enter the world of work. This is the first time that Ghana has developed a Senior High School Curriculum which focuses on national values, attempting to educate a generation of Ghanaian youth who are proud of our country and can contribute effectively to its development.

The Ministry of Education is proud to have overseen the production of these Learner Materials which can be used in class and for self-study and revision. These materials have been developed through a partnership between the Ghana Education Service, teacher unions (Ghana National Association of Teachers- GNAT, National Association of Graduate Teacher -NAGRAT and the Pre-Tertiary Teachers Association of Ghana- PRETAG) and National Subject Associations. These materials are informative and of high quality because they have been written by teachers for teachers with the expert backing of each subject association.

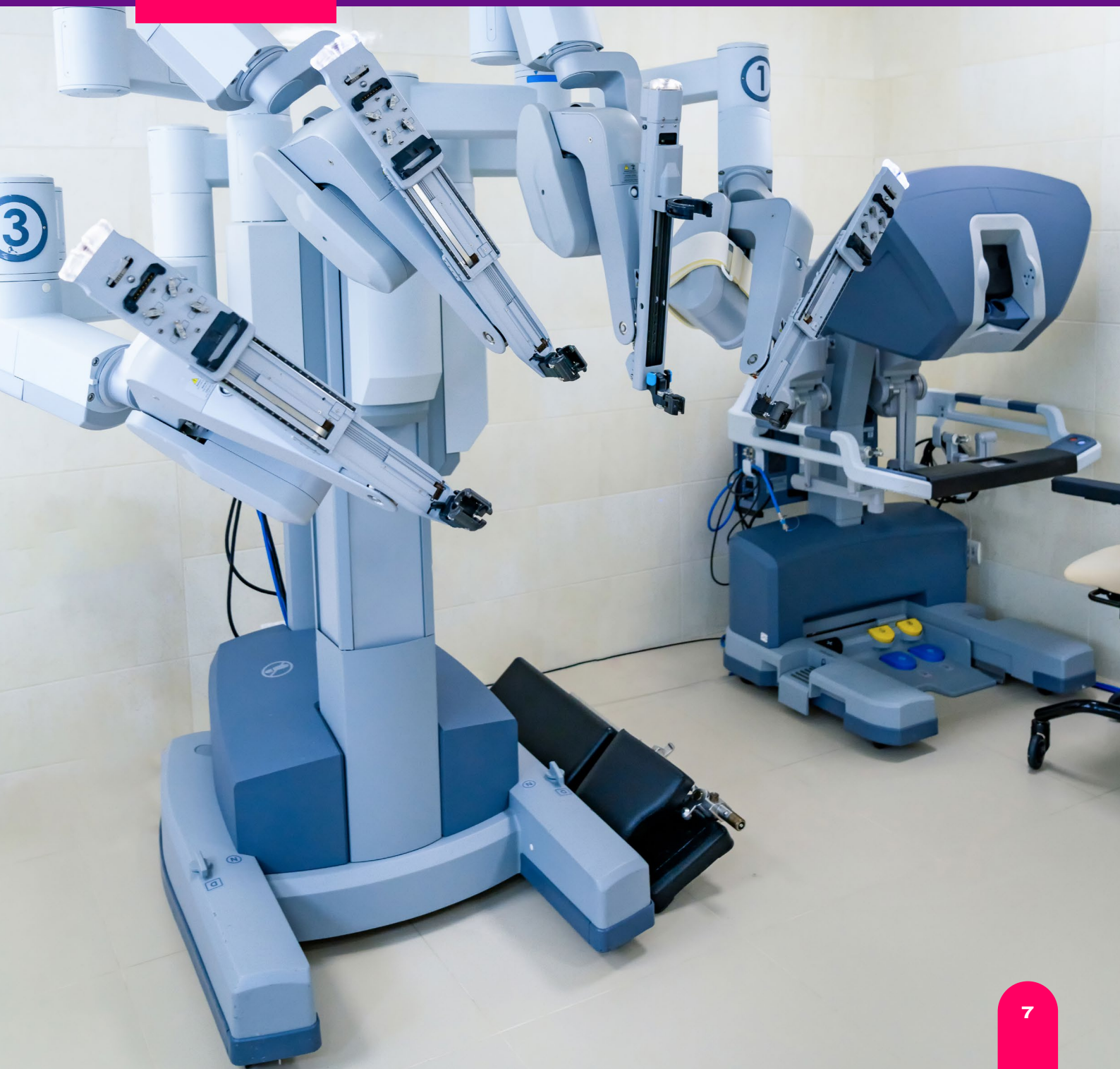
I believe that, if used appropriately, these materials will go a long way to transforming our Senior High Schools and developing Ghana so that we become a proud, prosperous and values-driven nation where our people are our greatest national asset.

**Haruna Iddrisu MP**  
*Minister for Education*

SECTION

1

# ROBOTS AND SOCIETY 2



# PRINCIPLES OF ROBOTIC SYSTEMS

## Robots & Society

### INTRODUCTION

---

In today's world, robots now play a significant part of the daily lives of the human race, from manufacturing to healthcare, agriculture and homes. This section explores how robots impact society, both positively and negatively. You will learn of the importance of ensuring that robots can work safely alongside humans, while considering ethical concerns. Understanding these ideas will help you think critically about the role of robots and how they will shape the future. Additionally, you will practice skills like writing articles on robotics and preparing you for job opportunities related to robotics, an exciting and growing field.

#### KEY IDEAS

- **Robot-Human Coexistence:** For robots to work with humans, there is a need to address issues like safety and have rules for them to follow that they cannot change themselves. This ensures a happy and stable relationship between robots and people in the home, at work or in the community.
- **Roboethics:** This covers the moral and ethical implications of robotics, or put another way dealing with 'right and wrong' related to the design and use of robots. This includes developing the guidelines robots should follow so they do not cause harm to humans, respect their privacy, act openly and operate within the limits of their roles in society.
- **Safety in Robotics:** It is essential to develop and follow safety standards and procedures to prevent accidents or harm when robots interact with humans or other machines.
- **Job Opportunities in Robotics:** Robotics is creating new jobs across industries. Learning how to identify these roles and preparing for them can open doors to exciting careers in this growing field.

### EXPLORING THE BROADER SOCIETAL IMPACT OF ROBOTS

In Year 1 you learned about the amazing things robots can do and how they can help people. You studied the many ways robots improve processes, like doing jobs faster and making tasks safer. You also found out about roboethics, which means applying rules and principles to make sure robots are used in a fair and responsible way. In this section you will study in more detail the impact of robots on society. You will explore both the good and bad things robots can bring to the world. By doing this, you will understand why it is important to use robots wisely and responsibly.

## Analysing the Societal Impact of Robots: A Step-by-Step Guide

In year one you learned how robots can solve problems and contribute positively to society. While problem solving and positive contributions are great you should not forget that it is also important to think about the overall impact on society when robots are used to replace humans. Using robots can also make problems and cause conflict. By considering both the benefits and drawbacks of using robots, designers can make robots that are useful and responsible, minimising harm and maximising their benefits to society.

Here are the steps you can follow to analyse how robots might impact society:

1. **Define the Scenario and Robot's Role:** Start by clearly describing what the robot is supposed to do and where it will be used. For example, is it going to help take care of elderly people, do farming tasks, or provide companionship? Knowing this makes it easier to focus on the analysis.
2. **Identify Potential Impacts:** Next, think about the possible effects the robot could have. You will need to consider how different groups of people might feel about the robot. Ask questions like: Will this robot create new jobs or take jobs away? Will people need new training to use it? Could it improve healthcare, protect the environment, or offer better education? Does it raise any issues like privacy, safety, or fairness? Are there any ethical guidelines or rules that need to be followed when designing the robot?
3. **Minimise Negative Impacts and Maximise Positive Effects:** After identifying possible problems, think about how to reduce them. This could include adding safety features, securing data, or creating rules for using the robot. Also, consider how to increase the robot's positive contributions to society.
4. **Think About the Long-Term:** It is important to consider how the robot might affect society in the future. Will it become outdated quickly and create electronic waste? Can the robot's design be updated to meet future needs?

This approach helps ensure that robots are designed not only to solve problems today but also to have a lasting, positive impact on society.

## Example of Societal Impact Analysis: Robot Companions for Elderly Care

Elli Q is a robot that is designed to provide **companionship** and **help** elderly people who live alone to do tasks with which they might struggle.

### Scenario and Robot's Function

Here is a breakdown of what Elli Q is designed to do:

1. **Better Communication:** Elli Q can have conversations, noticing the user's tone, facial expressions, and preferences. The robot can also help the elderly stay in touch with family and friends through video calls.



2. **Daily Assistance:** Elli Q can remind the elderly to take their medication, go to appointments, and do daily activities. The robot can also help with small tasks like adjusting the thermostat or turning on lights using voice commands.
3. **Health Monitoring:** Elli Q can be equipped with sensors that track things like heart rate and sleep patterns. If something unusual is detected, it can prompt the user to see a doctor (but with a clear warning that it is not a medical device).
4. **Emotional Support:** Elli Q can recognise how the user is feeling and respond accordingly, offering words of encouragement, playing calming music, or suggesting ways to relax.
5. **Cognitive Stimulation:** Elli Q can offer activities to keep the user's mind active, such as memory games, trivia, or learning activities based on their interests.

## Identifying Potential Impacts

Using the list above which defines what Elli Q can do; it is possible to predict the impact it might have on society?

### Positive Impacts

1. It could help elderly people stay independent by assisting with daily tasks.
2. Health might improve because of medication reminders and health monitoring.
3. The mental stimulation features could help keep the mind healthy.
4. Elli Q might help elderly people feel less lonely by improving communication and emotional support.
5. Early health problems could be detected through basic health monitoring.

### Possible Negative Impacts

1. Elderly people might rely too much on Elli Q, leading to less physical activity or social interaction which might lead to health problems.
2. Elderly people need training to use the robot and some might struggle with using the technology, creating challenges for those without tech skills.
3. There could be privacy concerns because Elli Q collects data from conversations, health sensors, and activities.
4. Elli Q's emotional responses might not always be appropriate if they are not programmed carefully.
5. The health and cognitive (thinking and problem solving) activities might not be suitable for the needs and abilities of all individuals.

## Minimising Negative Impacts and Maximising Positive Outcomes

The following points show how potential problems can be solved to make the most of Elli Q's benefits:

1. **Balanced Support:** Elli Q should encourage users to stay active and socialise, not just rely on its assistance.

2. **Digital Accessibility:** Elli Q should be easy to use, with clear instructions and training programs for people who are not familiar with the technology.
3. **Data Transparency and Control:** There should be strong security measures to protect data, and users should have control over what data is collected and how it is used.
4. **Ethical Programming:** Elli Q's emotional responses should focus on supporting well-being, and not try to force elderly people do things.
5. **Inclusive Design:** The health monitoring and cognitive (thinking and problem solving) activities should be designed to include people of all backgrounds and abilities.

## Considering the Long-Term

Looking into the future, there are some long-term impacts to think about:

1. **Obsolescence:** Elli Q's design should allow for software upgrades over time to keep it up to date as technology advances.
2. **E-waste:** To reduce electronic waste the manufacturers of Elli Q should have programs for recycling or refurbishing old models.
3. **Impact on Caregivers:** Elli Q should be used to help, not replace, human caregivers. Caregivers should be part of the setup process to keep a connection between the elderly and their human helpers.

## The Need for Human-Robot Coexistence: A Partnership That Benefits Both Sides

In year one you learned that robots can greatly improve the performance of organisations – manufacturing processes can be speeded up, there are fewer wasteful mistakes and safety records improve. However, it is important to point out that a successful relationship between humans and robots depends on both working together and receiving help from each other's strengths. Even though robots have impressive abilities, the integration of robots with humans brings greater achievements. Here is why:

1. **Creativity and Problem-Solving:**  
Robots are great at following instructions, but they cannot think creatively. Humans can produce new ideas, solve unexpected problems, and adjust to changes. This creativity is essential for making improvements and dealing with challenges that robots cannot handle.
2. **Social Intelligence and Empathy:**  
Robots cannot connect with others on an emotional level. Humans are better at interacting with customers, coworkers, and patients by building trust and understanding. In fields like healthcare, customer service, and education, human skills are needed to guide robots and provide the empathy and emotional intelligence that robots cannot offer.
3. **Ethical Decision-Making:**  
Ethics is becoming a bigger issue as robots play a more significant role in society. While robots can be programmed to follow ethical rules, some situations are too complex for robots to handle. Human judgement is essential to make sure robots act responsibly and in line with society's values.



### Activity 1.1 Analysing the Societal Impact of Robots

1. Your teacher will put you into groups of 3-5 for this activity.
2. Read the scenarios below on AgriBot (the Agricultural Assistant Robot) and ReNurse (the Robotic Nursing Assistant) and choose one for your analysis.

a. **Scenario A: AgriBot - The Agricultural Assistant Robot**

*AgriBot is a versatile robot designed to help farmers in various agricultural tasks. It can be equipped with different attachments to perform functions like:*

- i. **Planting and Seeding:** AgriBot can precisely plant seeds at best depths and distances, reducing waste and maximising yield.
- ii. **Weed Control:** Equipped with cameras and AI-powered weed identification, AgriBot can target and cut weeds with minimal herbicide use.
- iii. **Crop Monitoring and Irrigation:** Sensors on AgriBot can check soil moisture, nutrient levels, and crop health, enabling precise irrigation and targeted fertiliser application.
- iv. **Harvesting:** For specific crops, AgriBot can be adapted for automated harvesting, reducing manual labour, and ensuring efficient collection.

b. **Scenario B: ReNurse - The Robotic Nursing Assistant**

*ReNurse is a mobile robot designed to help nurses in hospitals and care facilities. It can perform various tasks to support patient care, including:*

- i. **Patient Monitoring:** ReNurse can check vital signs, medication schedules, and patient activity levels, providing real-time data to nurses.
- ii. **Basic Care Tasks:** ReNurse can help with tasks like delivering meals, transporting patients, and helping with mobility exercises.
- iii. **Communication and Companionship:** ReNurse can engage with patients through conversation, games, or entertainment, providing social interaction and emotional support.
- iv. **Data Collection and Analysis:** ReNurse can gather patient data and trends, helping nurses in making informed care decisions.

3. Discuss what your chosen robot does and how they might affect society.
4. Share your thoughts with your group. "What excites you about these robots? What concerns do you have?"
5. Thoroughly analyse your chosen scenario by filling out the following analysis worksheet
  - a. **Robot Description:** Write a brief description of the robot you chose and what field it operates in (agriculture or healthcare).

- b. **Positive Impacts:** List at least three potential positive impacts of the robot on society. Consider effects on jobs, the environment, and overall well-being.
  - c. **Negative Impacts:** List at least three potential negative impacts. Think about job displacement, privacy concerns, and the possibility of less human interaction.
  - d. **Ethical Considerations:** Discuss at least two ethical issues related to the robot, such as safety, data security, or biases in its design.
  - e. **Human-Robot Partnership:** Explain how humans will work alongside the robot and why this partnership is important.
6. Be prepared to present your completed worksheet to the whole class as a group. Have a discussion about what you learned. What surprised you? What do you think is the most important aspect of robot integration into society?
  7. Completed Analysis sheet will be presented to your teacher for scoring.

### Activity 1.2 Article Writing

Based on reflections from the previous activity and points from your analysis sheet, write an article of no more than 400 words which outlines the likely impacts of the robot on society. The target audience for your article is a school journal or local newspaper. Written language and approach should be engaging and informative and for readers who have no technical knowledge of robots.

*These guidelines will help you plan your article.*

1. A clear description of the robot's functionalities and its intended field of (agriculture or healthcare) is needed.
2. An analysis of the robot's potential positive and negative societal impacts. This analysis should consider potential effects on jobs, the environment, user privacy, and overall well-being.
3. Ethical considerations surrounding the use of this robot are to be discussed. Safety, data security, and potential biases in design or operation are key areas for exploration.
4. The interaction between humans and robots is critical for readers who need to be persuaded that the robot cannot cause harm, or accidents and there is a role for it in society so make every effort to emphasise it throughout the article.

## EXPLORING CAREER OPPORTUNITIES IN ROBOTICS

In this part of section one, you will dive into the world of robotics careers and learn how to connect your passion with real job opportunities. As you design and build robots, you will also explore how to identify job postings in fields like robotics and technology, whether online, in newspapers, or through other media. Additionally, you will practise preparing sample responses to these job postings, equipping you with the skills to confidently pursue a career in this rapidly growing field. Let this topic ignite your curiosity and inspire you to take the first steps toward becoming a leader in robotics innovation!

The information below is a guide to what can be achieved. Remember you are at the start of your career pathway if you choose robotics. Studying robotics as Senior High School will help you move on to higher qualifications in the same area. One day you might be able to apply for one of the jobs detailed below.

### Common Career Opportunities in Robotics

In year one you got a strong foundation in understanding the capabilities of robots and familiarised yourself with the design and development process. However, the world of robotics extends far beyond the role of designers and programmers. The following table shows the different career paths that can be followed to get you working with others to create and use robots. For each of the ten (10) robotic-related jobs mentioned below, a description of the employment sectors they are linked to, their main job responsibilities, entry qualifications required and essential skills are provided in the second column.

**Table 1.1:** Common Jobs in the field of robotics

Job Title	Description and Requirements
<b>Robotics Engineer</b>	<p><b>Sectors of Operation:</b> Manufacturing, Healthcare, Logistics</p> <p><b>Job Description/Responsibilities:</b> Designs, develops, tests, and implements robotic systems. Responsibilities include defining robot requirements, selecting components, overseeing construction, and integrating robots into existing workflows.</p> <p><b>Educational Requirements:</b> Bachelor's degree in Engineering (Mechanical Engineering, Electrical Engineering, Computer Engineering, Robotics Engineering)</p> <p><b>Required Skills, Technical Competencies and Experiences:</b> Strong analytical, problem-solving, and design skills.</p>

<b>Robot Programmer</b>	<p><b>Sectors of Operation:</b> Manufacturing, Healthcare, Agriculture, Logistics.</p> <p><b>Job Description/Responsibilities:</b> Writes, tests, and maintains robot control software. Responsibilities involve translating engineering designs into robot actions, debugging code, and optimising robot performance.</p> <p><b>Educational Requirements:</b> Bachelor's degree in Computer Science, Robotics, or a related field</p> <p><b>Required Skills, Technical Competencies and Experiences:</b> Strong programming skills (C++, Python, ROS)</p>
<b>Robotics Technician</b>	<p><b>Sectors of Operation:</b> Manufacturing, Mining, Research, Education.</p> <p><b>Job Description/Responsibilities:</b> Operates, maintains, and troubleshoots robots. Responsibilities include performing routine maintenance tasks, diagnosing and repairing malfunctions, and ensuring robots function safely and efficiently.</p> <p><b>Educational Requirements:</b> Diploma or an associate's degree or technical training in robotics.</p> <p><b>Required Skills, Technical Competencies and Experiences:</b> Mechanical aptitude and strong attention to detail.</p>
<b>AI Specialist</b>	<p><b>Sectors of Operation:</b> Various sectors</p> <p><b>Job Description/Responsibilities:</b> Develops and implements artificial intelligence algorithms for robots. Responsibilities involve machine learning, data analysis, and integrating AI functionalities into robotic systems.</p> <p><b>Educational Requirements:</b> Requires a bachelor's degree in computer science, computer engineering, artificial intelligence, or a related field.</p> <p><b>Required Skills, Technical Competencies and Experiences:</b> Strong analytical and problem-solving skills.</p>
<b>Human-Computer Interaction Designer</b>	<p><b>Sectors of Operation:</b> Various sectors</p> <p><b>Job Description/Responsibilities:</b> Designs user interfaces and interaction methods for robots. Responsibilities involve understanding user needs, designing intuitive interfaces, and ensuring seamless interaction between humans and robots.</p> <p><b>Educational Requirements:</b> Requires a bachelor's degree in design, human-computer interaction, computer science, computer engineering, or a related field.</p> <p><b>Required Skills, Technical Competencies and Experiences:</b> Strong user experience (UX) design skills.</p>

<b>Robotics Systems Engineer</b>	<p><b>Sectors of Operation:</b> Manufacturing, Aerospace, Defence, Research.</p> <p><b>Job Description/Responsibilities:</b> Integrates various robotic components into complex systems. Responsibilities involve designing robot systems architecture, managing communication protocols, and ensuring smooth interaction between different robotic subsystems.</p> <p><b>Educational Requirements:</b> Requires a bachelor's degree in engineering (electrical, computer, robotics).</p> <p><b>Required Skills, Technical Competencies and Experiences:</b> Strong systems engineering and project management skills.</p>
<b>Field Service Robotics Engineer</b>	<p><b>Sectors of Operation:</b> Manufacturing, Construction, Agriculture.</p> <p><b>Job Description/Responsibilities:</b> Provides on-site support and maintenance for deployed robots. Responsibilities include troubleshooting technical issues, performing repairs, and optimising robot performance in real-world environments.</p> <p><b>Educational Requirements:</b> Requires a bachelor's degree in engineering or robotics.</p> <p><b>Required Skills, Technical Competencies and Experiences:</b> Strong problem-solving and field service experience.</p>
<b>Biomechanics Engineer</b>	<p><b>Sectors of Operation:</b> Healthcare, Research.</p> <p><b>Job Description/Responsibilities:</b> Applies engineering principles to design robots for medical applications. Responsibilities involve designing robotic prosthetics, surgical robots, or robots for rehabilitation purposes.</p> <p><b>Educational Requirements:</b> Requires a bachelor's degree in mechanical, biomechanical or biomedical engineering</p> <p><b>Required Skills, Technical Competencies and Experiences:</b> Strong understanding of human anatomy and physiology</p>
<b>Robotics Software Architect</b>	<p><b>Sectors of Operation:</b> Manufacturing, Logistics.</p> <p><b>Job Description/Responsibilities:</b> Designs the software architecture for complex robotic systems. Responsibilities involve defining software components, ensuring system scalability, and optimising software performance for real-time applications.</p> <p><b>Educational Requirements:</b> Requires a bachelor's degree in computer science or computer engineering</p> <p><b>Required Skills, Technical Competencies and Experiences:</b> Strong software engineering and architecture design skills</p>

<b>Robotics Patent Attorney</b>	<p><b>Sectors of Operation:</b> Law Firms</p> <p><b>Job Description/Responsibilities:</b> Specialises in intellectual property law related to robotics inventions. Responsibilities involve drafting and securing patents for robotic technologies, advising clients on intellectual property rights, and ensuring legal compliance in the robotics domain.</p> <p><b>Educational Requirements:</b> Requires a Juris Doctor (JD) degree with a specialisation in intellectual property law.</p> <p><b>Required Skills, Technical Competencies and Experiences:</b> Strong understanding of robotics technology.</p>
---------------------------------	---

## After Developing Robotics Skills and Competencies

Once you have acquired the qualifications, skills and knowledge needed for a career in robotics, there are several steps you can take to find a job in this exciting field.

### Job Search Resources: Here are some common places to look for job opportunities or vacancies.

1. **Online Job Boards:** Many websites focus on job listings for robotics positions. These sites let you filter jobs based on location, industry, or job title (e.g., Indeed, LinkedIn, Glassdoor).
2. **Company Websites:** Robotics companies often post open positions on their own websites. These sites also provide information about the company and what they are looking for in an employee.
3. **Industry Associations and Networking Events:** Groups related to robotics sometimes hold career fairs or networking events where you can meet professionals in the field and learn about job openings.
4. **Recruitment Agencies:** Some agencies focus on finding candidates for robotics jobs and can help connect you with the right employers.
5. **Social Media Platforms:** Professional sites like LinkedIn are useful for connecting with recruiters and companies in robotics. By creating a strong profile that highlights your skills, you can become more visible to potential employers.
6. **Local Resources:** Depending on where you live, local newspapers, job fairs, and community colleges may offer job postings or career support in robotics.

### What to Look For: When reading job postings, pay attention to these details listed below.

1. **Required Skills and Qualifications:** Do not just look at the job title. Carefully review the skills and qualifications needed for the role. Remember, many jobs in fields like automation and manufacturing use skills that apply to robotics, even if “robotics” is not in the job title.

2. **Job Responsibilities:** Make sure you understand the tasks and expectations of the role.
3. **Company Culture and Values:** Research the company to see if their values and work environment match what you are looking for.

### Application Process: Once you find a job you are interested in, you will usually need to send the following:

1. **Cover Letter:** A customised letter that shows your interest in the job and highlights your skills and experiences that match the job description.
2. **CV or Resume:** A clear, short document that includes your education, work experience, technical skills, and achievements.

### Sample Cover Letter for A Robotics-Related Job

Most job applications require a cover letter. This letter briefly shows your passion for robotics and explains your understanding of the company's work. It should address specific requirements and challenges mentioned in the job description. The letter should also highlight your relevant skills and experience, showing that you are a good fit for the role. Below is a sample.

*[Candidate's Name and Address]*

*[Candidate's Phone Number]*

*[Candidate's Email]*

*[Date]*

*The Head of Human Resource,*

*XYZ Technologies*

*P. O. Box 998877*

*Kumasi, Ghana*

*Dear Sir,*

#### **APPLICATION FOR ROBOTICS ENGINEER POSITION**

*I am writing to express my keen interest in the Robotics Engineer position advertised in the Daily Graphic on March 7, 2024. As a highly motivated and results-oriented Robotics Engineer with [Number] years of experience in designing, developing, and implementing robotic solutions, I am confident that my skills and experience align perfectly with the requirements outlined in the job advertisement.*

*In my previous role at [Previous Company Name], I played a key role in [Briefly describe a relevant project or achievement that showcases your robotics engineering skills. Quantify your impact whenever possible]. My expertise lies in [List 2-3 of your most relevant technical skills, e.g., robot design, automation systems, programming languages (specify which ones)]. I am also proficient in [List any relevant software programs you use, e.g., CAD software, simulation software].*



*I am particularly drawn to XYZ Technologies' innovative approach to utilising robotics in warehouse management. My strong understanding of [Mention specific areas of robotics relevant to the job posting, e.g., warehouse automation, material handling systems, path planning] would allow me to contribute significantly to your team's efforts in optimising efficiency and productivity within your facilities.*

*My resume, attached herewith, provides further details regarding my qualifications and experience. I am eager to learn more about this exciting opportunity and discuss how my skills and experience can benefit XYZ Technologies. Thank you for your time and consideration.*

*Sincerely,*

*[Your Name]*

## Sample Curriculum Vitae (CV) or Resume for A Robotics-Related Job Position

Following the above-written cover letter, the candidate is expected to attach a proofread CV or Resume. The CV must clearly outline the candidate's educational background, including relevant coursework and any robotics-related certifications. It should also include the candidate's work experience, highlighting past projects or achievements related to robotics and showcasing their technical skills and accomplishments. The following is an example of what could have been attached.

*[Candidate's Name and Address]*

*[Candidate's Phone Number]*

*[Candidate's Email]*

### **Summary**

*A highly motivated and results-oriented Robotics Engineer with [Number] years of experience in designing, developing, and implementing robotic solutions for various applications. Proven ability to optimise warehouse operations through automation and proficient in robot design, automation systems, and programming languages like [List languages]. Skilled in utilising [List relevant software programs] to enhance efficiency and productivity.*

### **Skills**

- Robotics Design
- Automation Systems
- Programming Languages (e.g., C++, Python, ROS)
- Warehouse Automation
- Path Planning
- CAD Software (e.g., Solidworks)
- Simulation Software (e.g., Gazebo)
- Communication
- Teamwork
- Adaptability
- Material Handling Systems
- Problem-Solving

### ***Experience***

- Robotics Engineer | [Previous Company Name] | [Start Date] - Present

*Designed and implemented a robotic arm solution for automated product picking in a warehouse, resulting in a [Quantifiable improvement] increase in picking efficiency.*

*Developed and maintained control software for a fleet of warehouse robots, ensuring smooth operation and minimal downtime.*

*Collaborated with cross-functional teams (engineering, operations) to identify automation opportunities and implement effective robotic solutions.*

*Performed regular maintenance and troubleshooting of robotic systems, ensuring optimal performance and reliability.*

*Stayed up to date with the latest advancements in robotics technology and participated in relevant training programs.*

- [Previous Job Title] (if applicable) | [Previous Company Name] | [Start Date] - [End Date]

*[Briefly describe your responsibilities and achievements in this role, highlighting relevant skills]*

### ***Education***

*[Degree Name] in [Field of Study] \ [University Name] \ [Graduation Year]*

*[List any relevant coursework or certifications related to robotics]*

### ***Projects (Optional)***

*[Include any personal projects or contributions to open-source robotics projects that showcase your skills and interests.]*

### **Note**

*Replace the bracketed information with your specific details and tailor the content of the “Experience” and “Projects” sections to best reflect your background and the requirements mentioned in the job advertisement.*

## **Activity 1.3 Research Career Paths in Robotics**

*Your teacher will put you into small groups for this activity.*

1. In your groups, select at least two career options from the list below (or any other robotics career of your choice):
  - a. Robotics Sales Engineer
  - b. Robotics Safety Engineer

- c. Robotics Quality Control Inspector
  - d. Robotics Research Scientist
2. Research the following aspects for both of your chosen career paths:
- a. **Sectors:** Explore which industries or industrial sectors the job could be available in. For example, is the position common in manufacturing, healthcare, education, or technology sectors?
  - b. **Job Description:** What are the typical responsibilities and tasks involved in the career? Write down a detailed list of tasks the employee would perform.
  - c. **Educational Requirements:** What qualifications, degrees, or certifications are necessary for this career? Include both formal education and any additional certifications.
  - d. **Skills and Knowledge:** What technical skills or programming languages are commonly required? Examples could include knowledge of specific robotics platforms, coding using a recognised language like Python, or experience with quality control systems.
3. After completing your research, create a brief report (300-500 words) for each of the two chosen careers, summarising your findings using word processing software. You will share this report with the class to help everyone understand various career paths in robotics.

### Activity 1.4 Finding Relevant Job Postings

*Your teacher will put you into small groups for this activity.*

1. In your groups, search for current or past job postings that are similar to the career paths you selected. You can use resources such as:
  - a. **Online job boards** (e.g., LinkedIn, Indeed, Glassdoor)
  - b. **Company websites** (e.g., major robotics companies like Boston Dynamics, iRobot, ABB)
  - c. **Newspapers or community bulletin boards** (if applicable)

*Focus on the following aspects:*

- a. **Required Skills and Experience:** Look for common technical skills, programming languages, and experiences mentioned in the job posting.
- b. **Responsibilities:** Compare the listed responsibilities to the ones you researched in **Activity 1.3**.
- c. **Educational Background:** Check if the qualifications match your findings from the previous activity.
- d. **Deadline Considerations:** You can include job postings that are past their deadlines but still relevant to your chosen career.

2. Write a summary (200-300 words) for each job posting you find using word processing software. Your summary should highlight the job's main tasks, required skills, qualifications, and the sector in which the job is situated. Focus on how these align with what you learned in Activity 1.3.

### Activity 1.5 Drafting a Sample Cover Letter

Based on the job posting you researched, draft a personalised cover letter using word processing software. Highlight specific skills and qualifications that align with the job posting (e.g., "I have developed proficiency in programming with Python and MATLAB, both of which are listed as required skills in the job posting.").

*Follow these guidelines but make sure you use a fictional name, address and telephone number if you are sharing it with others.*

1. **Introduction:** Mention the job position and where you found the job posting. Express your enthusiasm for the role.
2. **Body:** In 1-2 paragraphs, explain why you are a good fit for the job. Reference the skills, experiences, and qualifications that match the job description. Even if your experience is fictional, it should demonstrate how you are prepared for the job.
3. **Closing:** End by expressing your interest in an interview and thanking the employer for their time.

### Activity 1.6 Drafting a Sample Resume




1. Using word processing software draft a personalised resume that includes relevant educational background, skills, and (fictional) work experiences that directly relate to the job posting you researched.
2. Tailor your resume to the specific career path you are focusing on. Follow the following structure to draft your resume but make sure you use a fictional name, address and telephone number if you are sharing it with others:
  - a. **Contact Information:** Full name, phone number, and email address.
  - b. **Objective Statement:** A brief statement summarising your career goals and interest in the job.
  - c. **Education:** Highlight relevant academic qualifications, including any degrees, certifications, or coursework that align with the job posting.
  - d. **Skills:** List technical and soft skills relevant to the job posting (e.g., "Experience with robotic programming languages such as Python, ROS, or C++").
  - e. **Experience:** Even if your experience is fictional, describe relevant project work, internships, or jobs related to robotics.




- f. **Certifications and Awards:** Include any certifications, even if fictional, to demonstrate your commitment to the field of robotics.

### Activity 1.7 Peer Review and Feedback

1. Pair up with a classmate or work in a small group which your teacher will organise to share your cover letters and resumes. Make sure you are not sharing any personal information with others like your address and telephone number.
2. Review your peer's cover letter and resume, offering constructive feedback on areas such as:
  - a. Clarity of the content
  - b. Relevance of skills and experiences
  - c. Structure and flow of the documents
3. After receiving feedback, revise your cover letter and resume to ensure your documents are well-written and properly formatted.

## EXTENDED READING

Resource	QR Code to Accessing Resource
1. Ethical Dilemma in Robotics (Video Resource) <a href="https://www.youtube.com/watch?v=9uoWaYEgrs">https://www.youtube.com/watch?v=9uoWaYEgrs</a>	
2. IEEE standard review—Ethically aligned design: A vision for prioritising human wellbeing with artificial intelligence and autonomous systems. (Paper Publication) <a href="https://ieeexplore.ieee.org/abstract/document/8058187?casa_token=Nd2IR-VKpGOUAAAAA:YrdzRmtlAHISv8ZsgdCL1HSDGR9VU4rlMN-q6VkeUV5F3yLaouGwrU2Qh_NP4LDprKVr90LcDhTM">https://ieeexplore.ieee.org/abstract/document/8058187?casa_token=Nd2IR-VKpGOUAAAAA:YrdzRmtlAHISv8ZsgdCL1HSDGR9VU4rlMN-q6VkeUV5F3yLaouGwrU2Qh_NP4LDprKVr90LcDhTM</a>	
3. Ethically aligned design for assistive robotics. (Paper Publication) <a href="https://ieeexplore.ieee.org/abstract/document/8535889?casa_token=3b8xIYPSn-RQAAAAA:1PuuoTYiNh3IKUJjNae0gpArcA6hAFAtXsapnUcZNvz4MH-vA56FXRET7zGDtrFYNE1Opz4o10k">https://ieeexplore.ieee.org/abstract/document/8535889?casa_token=3b8xIYPSn-RQAAAAA:1PuuoTYiNh3IKUJjNae0gpArcA6hAFAtXsapnUcZNvz4MH-vA56FXRET7zGDtrFYNE1Opz4o10k</a>	

4. SciTrends - Robotics Careers (Video Resource)	
5. Cool Careers - Episode 4: Robotics and Automation (Video Resource)	
6. How to write a cover letter (Video Resource)	
7. Resume Writing: 4 Tips on How to Write a Standout Resume   Indeed Career Tips (Video Resource)	

# REVIEW QUESTIONS 1.1

1. What does the term “roboethics” refer to?
  - A. The study of how robots are built
  - B. The principles guiding the ethical use of robots
  - C. How robots can replace human workers
  - D. The technical specifications of robot systems
2. Robots are capable of making complex ethical decisions on their own. (True/False)
3. List three general benefits of using robots in society.
4. Describe one way robot designers can minimise the negative societal impacts of robots.
5. Why is it important for humans and robots to work together in fields like healthcare and education?
6. Study the picture below which shows a service robot that works in a busy airport waiting lounge. Visitors catching flights get free soft drinks and snacks from a self-service area which they consume at tables. Their empty cups and plates are collected by the robot, which is programmed to stay in one area, navigate between tables, and return to the kitchen when all its shelves are full. The robot stops at each table and asks whether there are any items that are finished with and visitors place them on the shelves.



**Figure 1.1: Robot in Amsterdam Airport, Netherlands**

- a. List two advantages of using a robot to collect empty cups and plates.
- b. List two disadvantages using a robot to collect empty cups and plates.
- c. Why might staff in the airport lounge be unhappy with their robot colleague?
- d. Name three other locations where a service robot like the one in the picture might be useful.



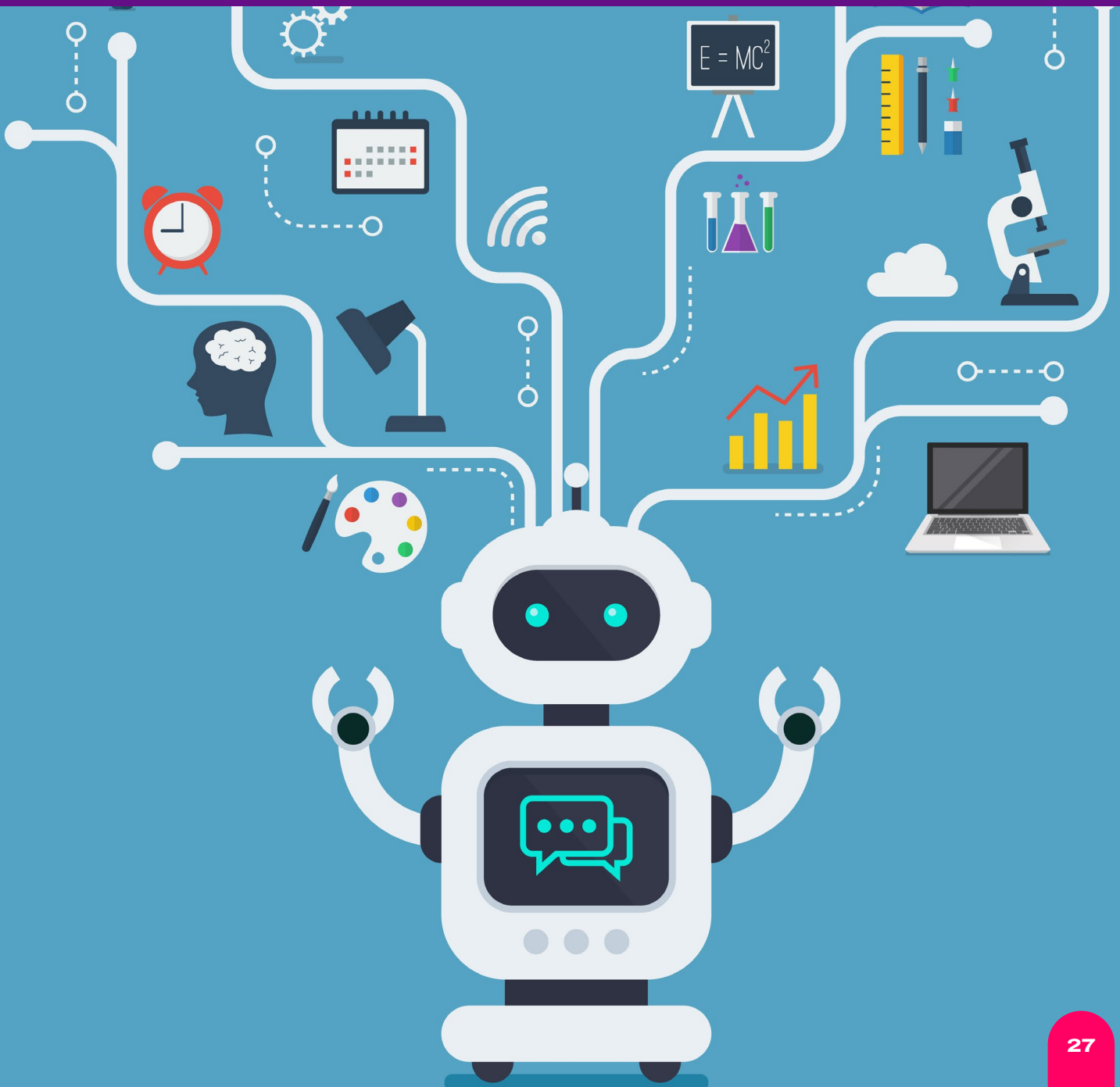
## REVIEW QUESTIONS 1.2

1. What is the primary responsibility of a Robotics Engineer?
  - A. Develop and implement AI algorithms for robots
  - B. Write, test, and maintain robot control software
  - C. Design, develop, test, and implement robotic systems
  - D. Operate, maintain, and troubleshoot robots
2. Fill in the Blank: A \_\_\_\_\_ is required for the position of Robotics Patent Attorney, focusing on intellectual property law related to robotics inventions.
3. True or False: A Robotics Technician typically requires a bachelor's degree to be eligible for employment in the field.
4. List two sectors where a Robot Programmer might work and describe one primary responsibility of this role.
5. Why is it important to research a company's culture and values before applying for a job in robotics? Provide two reasons based on the content provided.

SECTION

# 2

## ROBOT CONTROL PRINCIPLES 2



# PRINCIPLES OF ROBOTIC SYSTEMS

## Robot Control Principles

### INTRODUCTION

In this section, you will learn how to design both non-feedback and feedback control systems using component and system diagrams. These diagrams will serve as vital tools, helping you visualise and implement controls in various scenarios. You will also delve into mathematical modelling, focusing on continuous-time machines and finite-state machines. By analysing different scenarios, you will gain valuable insights into how robots can be programmed to respond effectively to their environments. This section aims to equip you with the knowledge and skills necessary to create sophisticated robotic systems, laying the groundwork for your future innovations in robotics. Read on to embark on this exciting journey into the world of robot control principles!

#### KEY IDEAS

- **Component Diagrams:** Component diagrams are like blueprints for robots. They show all the individual parts (like motors, sensors, and controllers) and how they connect. Learning to create these diagrams helps you understand what each part does in a robot.
- **System Diagrams:** System diagrams are like flowcharts for robots. They illustrate how different parts of a system interact and work together to achieve a goal. By using these diagrams, you can see how information flows through the robot and how it responds to different inputs.
- **Non-Feedback Systems:** In a non-feedback system, the robot makes decisions based on pre-set instructions without checking how well it is doing. Imagine a robot following a simple path: it moves straight ahead without adjusting its course based on what it encounters. Learning about these systems helps you understand basic control methods.
- **Feedback Systems:** Feedback systems are smarter. They constantly check their environment and adjust their actions based on what they observe. For example, if a robot is supposed to pick up an object, it will check if it successfully grabbed it and adjust if it did not. Understanding feedback helps you design robots that can adapt to changes in their surroundings.
- **Mathematical Models:** Mathematical models are like formulas that describe how robots behave. They help you predict what will happen when you make changes to a robot's design or programming. For example, if you know the speed of a robot, you can calculate how far it will travel over time.
- **Finite-State Machines:** A finite-state machine is like a robot's decision-making guide. It helps the robot know what to do based on different situations or "states." For instance, a robot might have states like "moving forward," "turning," or "stopped." Understanding this concept helps you create more complex behaviours in robots.
- **Control Implementation:** This is all about putting your plans into action and making a robot which does something you want it to do. Using the diagrams and models you have studied, you will design and program robots to perform specific tasks. It is like taking all the pieces of a puzzle and putting them together to see the whole picture.

# SYSTEMATIC DESIGN METHODS FOR NON-FEEDBACK CONTROL SYSTEMS

In this section, you will explore how to design non-feedback systems in robotics and automation. Non-feedback systems are important because they control processes where you can directly connect what you input to what you get out, without needing to make changes based on ongoing feedback. For example, if you tell a robot to move forward a certain distance, it will just go that distance without checking if it has gone too far or not. Learning how to design these systems gives you the skills to turn your ideas into real robot designs. You will use diagrams to show the different parts of the system and how they work together. This way, you can see how to make your robots work in the real world!

## Scenarios Using Non-Feedback Systems

Non-feedback systems are used in situations where precise control is important, and you do not need to get feedback from the environment. Here are some examples:

### 1. Simple Timers and Sequencers

- a. **Irrigation systems:** Sprinklers can be set to turn on at specific times, regardless of how wet the soil is.
- b. **Holiday lights:** Decorative lights can be programmed to turn on and off at certain times without checking how bright it is outside.

### 2. Open-Loop Automation Tasks

- a. **Factory assembly lines:** Robots do their tasks in a set order and for a set amount of time without changing their actions based on real-time conditions.
- b. **Car washes:** The car wash follows a fixed process (like washing, rinsing, waxing) without checking how clean the car is during each step.

### 3. Simple Remote-Controlled Devices

- a. **Toy cars:** You control toy cars using a remote, but the cars do not send back information about where they are or how fast they are going.
- b. **Drones:** Basic drones can perform programmed flight paths or respond to your controls without checking their height or direction in real time.

### 4. One-Way Communication Systems

- a. **Fire alarms:** Fire alarms go off when they detect smoke or heat but do not check if the fire is out or if people have left the building.
- b. **Traffic light timers:** Traffic lights follow fixed timing schedules to change colours without checking how many cars are waiting.

### 5. Pre-Programmed Tasks with Limited Environmental Interaction

- a. **Vacuum cleaners:** Many robotic vacuum cleaners have set paths for cleaning rooms and do not change their routes based on obstacles or how dirty the floor is.

- b. **Lawn mowing robots:** Some robotic lawn mowers stay within a set area but may not change their path based on the length of the grass or uneven ground.

The next section concentrates on how a **Fire Alarm Systems** works. This is a non-feedback system because it operates on a simple trigger mechanism without relying on feedback to adjust its function. If there is smoke, then a sensor detects the smoke and sets off the alarm.

## Understanding Key Elements

To design a non-feedback fire alarm system effectively, you need to understand some important parts that help it work. Here is how to do that:

### 1. Identify the Inputs

For our fire alarm system, the main inputs are signals from smoke detectors, heat detectors, and emergency switches.

- a. **Smoke detectors** sense smoke particles in the air.
- b. **Heat detectors** notice when the temperature goes up, which could mean there is a fire.
- c. **Emergency switches** are protected by a glass panel. When someone breaks the glass, it sends a signal to activate the system. These inputs are essential for the fire alarm to work.

### 2. Define the Outputs

Next, you need to know what the system will do when it receives those inputs. The outputs of a fire alarm system include:

- a. Sounding alarm bells and sirens
- b. Flashing lights
- c. Activating sprinklers
- d. Opening security doors

### 3. Determine the Trigger Conditions

Finally, figure out what conditions need to be met to activate the outputs. For the fire alarm system, the alarms will only go off if:

- a. The smoke or heat levels reach a certain point
- b. The emergency switch is activated by breaking the glass

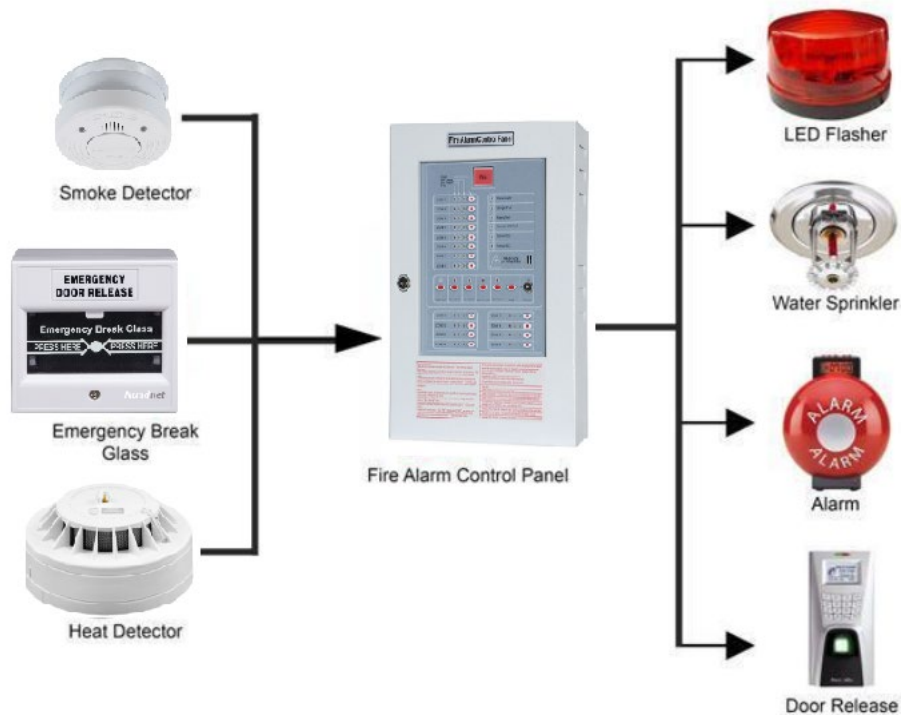
These conditions need to be programmed into the fire alarm control panel so that when the right signals are detected, the alarms and other outputs will activate.

## Steps To Create Diagrams

### Drafting Component Diagrams

- a. List all components: Smoke detectors, heat detectors manually activated switches, control unit, alarm bell, siren, and flashing lights, sprinklers.

- b. Represent each component with standardised symbols.
- c. Connect sensors to the control unit and the control unit to the alarms.



**Figure 2.1: Component Diagram of a Fire Alarm system.**

The component diagram in **Figure 2.1** shows how the alarm works. Signals from the inputs (smoke detector, heat detector and glass covered button) are picked up by the control panel. The control panel processes the signals and sends them to outputs, LED flashers, water sprinklers, bells and door release mechanisms.

No feedback is required in this system, which means once it's activated, it doesn't check if the fire is still there; it just keeps sending signals to the outputs until it is manually deactivated.

## Creating a System Diagrams for the Fire Alarm

- a. **Map out the physical and logical connections**

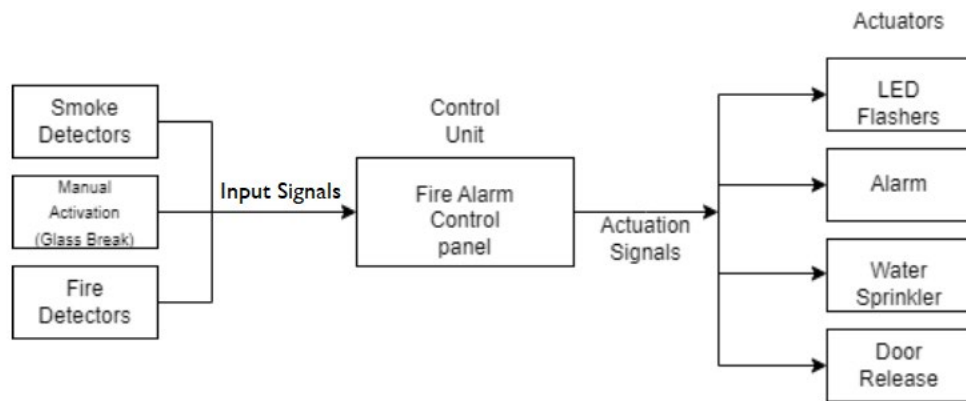
Start by drawing the connections between each part of the system. Use lines to link the smoke detector, heat detector, and emergency break glass to the control panel, and from the control panel to alarms, sprinklers, and other outputs.

- b. **Indicate the signal flow from detection (input) through the control unit to the alarm systems (output)**

Use arrows to show the direction that signals take; inputs go from the sensors to the control panel, outputs go to the sounders and LEDs.

- c. **Ensure clarity in showing how each component interacts within the system**

Show how each part of the system sends and receives signals. Your system diagram should look like **Figure 2.2**, below.



**Figure 2.2: System diagram of a fire alarm system**

A **systems diagram** helps you understand the operation by providing a big picture where all flows, components and connections are represented.

### Activity 2.1 Designing a Non-Feedback Control System

*Your teacher will put you into small groups for this activity.*

Design a non-feedback control system for an electric iron. Follow the guidelines below.

1. **Identify Core Components:** Using keywords like “Working of Electric Iron” on YouTube or “How an electric iron works / Components of an Electric Iron” on the internet, research on how an electric iron works in your groups. While watching the video or doing your research copy and complete the table below to make sure you have all the points you need. Make sure all your group has access to the notes.

**Table 2.1:** Identify Core Components

Guiding Questions	Response
What are the key parts of the device?	
What does each part do in the system?	
What are the inputs that the device needs to start (e.g., power)	
What triggers the device to begin its function (e.g., button press)?	
What does the device produce or do (e.g., heating, spinning, lighting up)?	
How are the outputs connected to the inputs?	



What conditions cause the device to stop or change state (e.g., timer, movement)?	
How does the device move through stages without external feedback (what prevents the iron overheating)?	

- a. After the research work individually to list the main parts of an electric iron and how each part functions within the system.
  - b. Pair up with a partner from your group to discuss the components you identified.
  - c. Share your ideas with the larger group, focusing on how these components work together.
2. **Create a Component Diagram:** Based on components identified in your groups, draw a component diagram that shows how the parts of the electric iron are connected and what each component does (e.g., the thermostat regulates temperature). You can use sticky notes or a whiteboard to physically place and label each component. You can rearrange them to show their relationships and signal flow.
  3. **Define Inputs and Outputs:**
    - a. List the inputs e.g. power supply and outputs e.g. heating of the soleplate.
    - b. Discuss in your groups to identify **trigger conditions** (e.g., “reaching set temperature,”). Identify when and how the system starts or stops.
  4. **Create a system diagram:** Draw a system diagram using squares for components and arrows in the same way as the fire alarm was done showing how signals flow through the system.
  5. Present your system diagram to the class, explaining the flow of signals and how the system operates without feedback. Be prepared to answer questions.

### Activity 2.2 Open-Ended Non-Feedback System Design

*Your teacher will put you into small groups for this activity.*

Apply your knowledge of non-feedback systems to design a system of your choice. Follow these guidelines:

1. **Choose Your System:** Choose from the following list: automatic coffee maker, traffic light system, vending machine, or car wash or suggest a device of your own. In your groups, research on your chosen system and core components and how the system operates. Use the table in Activity 2.1 to ensure that you have all the points you will need.

2. **Create a Component Diagram:** In your groups, map out a detailed component diagram that shows the system structure. Identify all the main parts and their roles.
3. **Define Inputs, Outputs, and Trigger Conditions:** Define what inputs (e.g., buttons, power) and outputs (e.g., coffee pouring, lights changing) your system will have. Identify when and how the system starts or stops.
4. Present these ideas in a **system diagram**, which shows how signals flow through the system.

## SYSTEMATIC DESIGN METHODOLOGY FOR FEEDBACK CONTROL SYSTEMS

In this section, you will learn how to design feedback control systems by using diagrams that show how the parts of the system work together. Feedback systems use sensors to check values and then, based on their readings the controller make changes to ensure the system operates within the limits that have been defined. This helps the system be more accurate and flexible. By learning how to design these systems, you will improve your ability to create robots and machines that can respond and adjust to changes in different situations.

### Scenarios of Feedback Loop Systems

Feedback loop systems are used in many everyday devices to help them adjust and work more efficiently. Here are some common examples:

1. **Thermostats:** These devices control the temperature in a room by turning the heater or air conditioner on and off based on the temperature it senses. It keeps adjusting itself to maintain the temperature you set.
2. **Speed Control in Motors:** This system helps control the speed of a motor by comparing the actual speed to the desired speed and making changes as needed. This is used in things like electric cars and machines in factories.
3. **Automated Lighting Systems:** These systems adjust the brightness of lights based on how much natural light is already in the room. They make sure the lighting stays consistent by dimming or brightening the lights.
4. **Robotic Arms:** Robotic arms use sensors to adjust their movements based on what they sense. This ensures accuracy for tasks like assembling parts in a factory.

In the next section you will learn how an **Automated Lighting System** takes the form of a feedback loop system.

## Understanding The Key Elements

When designing an automated lighting system that uses feedback, it is important to understand the key parts of the system. These are:

### 1. Identify the Inputs:

The first step is to figure out what information the system needs to operate. For an automated lighting system, the input is the light sensor. This sensor measures how much light is already in the room (ambient light) and sends this information to the system. This is essential because the system needs to know if the room is too bright or too dark. A second input to the system is power, which is by the controller to provide light.

### 2. Define the Outputs:

The next step is to figure out what the system is supposed to do. For our lighting system, the output is the brightness level of the lights. The goal is for the system to keep the room at the right brightness by adjusting the lights.

### 3. Determine the Control Mechanism:

The next step to understand how the system makes decisions to adjust the light. The **controller** (often a microcontroller) compares the actual light level in the room with the desired light level. Based on this comparison, it sends signals to the system to either brighten or dim the lights.

The **light dimmer/driver** is the part of the system that receives these signals and changes the brightness of the lights by adjusting the power going to the lighting system. This ensures the room stays at the right level of brightness.

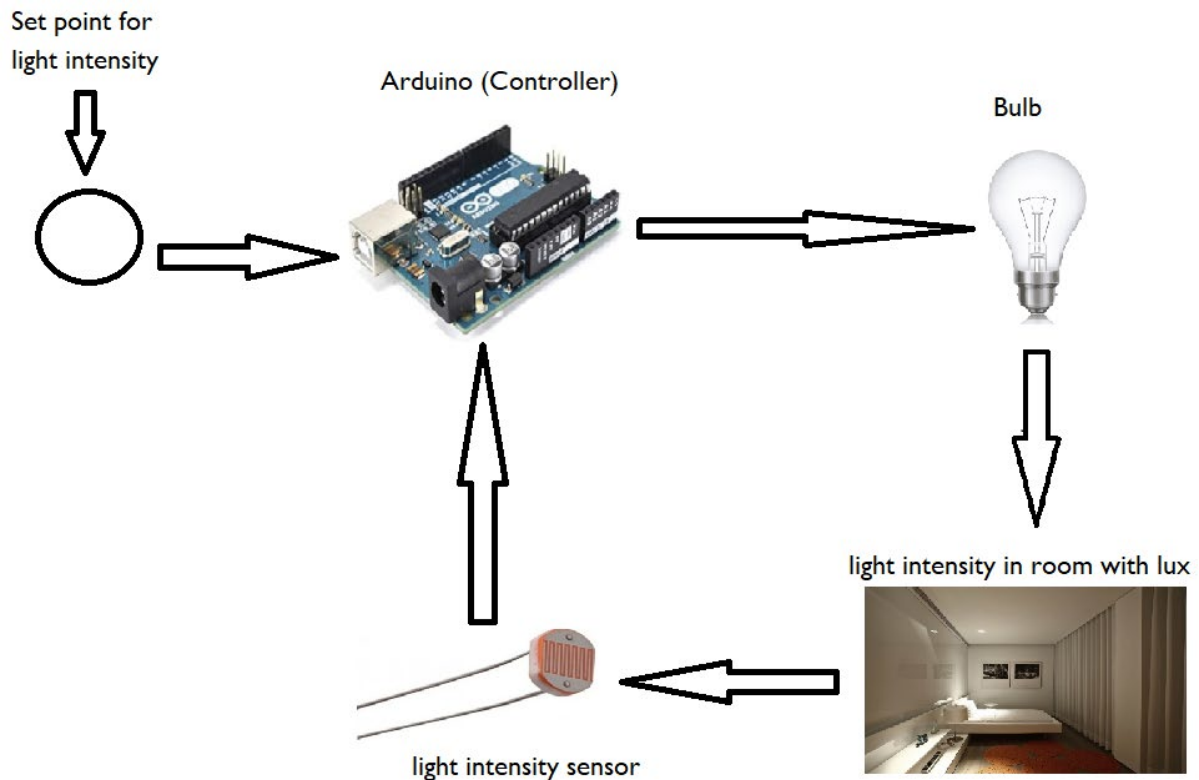
## Steps to Create Component and System Diagrams

To design this system, you can follow these steps to create diagrams that show how everything is connected:

### 1. Drafting Component Diagrams

- a. **List all components:** The main parts are the light sensor, microcontroller, light dimmer/driver, and the lighting system itself.
- b. **Use standard symbols:** Represent each component using the correct symbols so it is easy to understand.
- c. **Show connections:** Draw lines to show how the light sensor connects to the controller, how the controller connects to the dimmer/driver, and how the dimmer/driver controls the lights.

These diagrams help visualise how each part of the feedback system works together to keep the room lit at the perfect brightness.



**Figure 2.3: Component diagram for automated Lighting System**

In an automated lighting system with feedback control, like the one shown in **Figure 2.3**, the **controller** (such as an Arduino) starts by setting up connections for the input (light sensor) and output (AC light dimmer).

Here is how it works step by step:

1. The **light sensor** keeps checking the amount of light in the room (ambient light) and turns this information into an electrical signal. This signal is then sent to the controller.
2. The **controller** processes the data from the light sensor using a control **algorithm**. This is a set of instructions that tells the controller what to do based on the actual light level and the desired light level (the set point for how bright the room should be).
3. After comparing the actual light to the desired level, the controller sends signals to the **AC light dimmer**, which changes the brightness of the lights to match the desired light level.

This process forms a **feedback loop**. The system keeps adjusting the light brightness in real-time, using the information from the light sensor to maintain the perfect amount of light in the room.

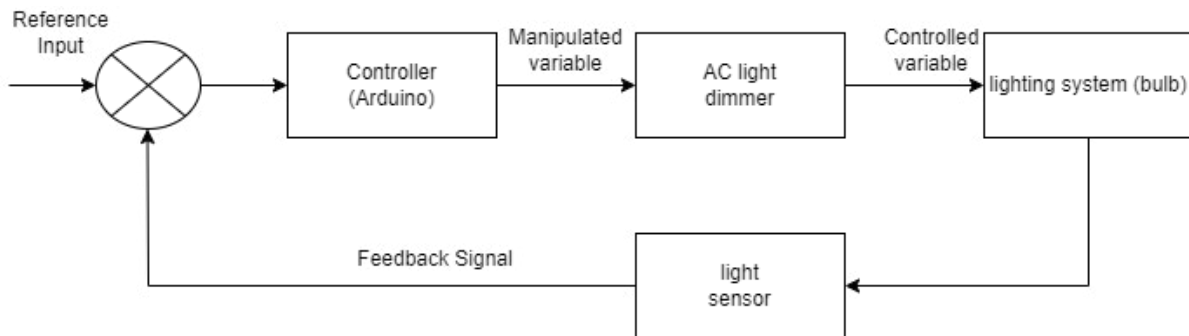
## 2. Creating System Diagrams:

### a. Step 1: Map out the connections

Draw a diagram that shows how all the parts of the system are physically connected to each other and how they logically interact. This will help you understand how each part works with the others.

**b. Step 2: Show the feedback loop**

In your diagram, show how the feedback loop works. Start with the light sensor (input) that detects the light level. Then, follow the path to the controller (Arduino), which processes the information. Next, show how the controller sends signals to the relay (actuator), which controls the lighting system (output). This loop keeps the system working by continuously adjusting the light based on the sensor's readings.

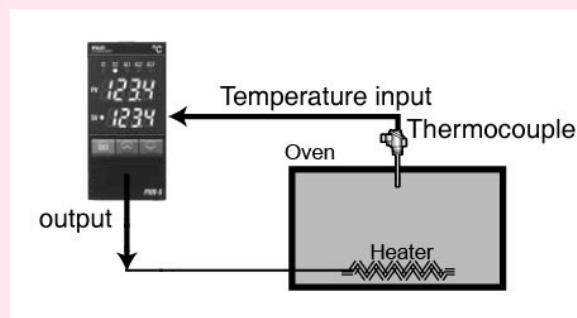


**Figure 2.4: Diagram of an automated Lighting System (feedback loop system)**

### Activity 2.3 Analysing a Feedback System

*Your teacher will put you into small groups for this activity.*

**Scenario:** In your group, analyse the temperature control system in the image below.



**Figure 2.5: How and electric oven maintains a set temperature (<https://www.coulton.com/>)**

**1. Prompts for Analysis:**

- Identify the main components of the system (e.g., sensor, controller, actuator).
- Explain the function of each component and how they interact.
- What is the system's input?
- What is the desired output?
- How does the controller compare these values?
- How does the controller adjust the system's output based on the feedback?

## 2. Task:

- a. Sketch a system diagram of the feedback loop.
- b. Interactive Element: Transfer your sketch to digital software using an online tool (e.g., draw.io, Google Drawings, TinkerCAD).
- c. Working together, write a brief report using a word processor(200-300 words) explaining the system's operation.

3. **Class Presentations:** Present your group's analysis diagram and report to the class. Receive feedback and refine your explanation based on peer input.

## Activity 2.4 Design a Feedback System (Temperature Control)

*Your teacher will put you into small groups for this activity.*

In your groups, design a temperature control system for a room which uses a fan to remove hot air. You may choose a specific environment, such as:

*A **greenhouse** (where maintaining optimal temperature for plant growth is crucial).*

*A **computer server room** (where overheating can lead to system failure).*

*A **laboratory** (where temperature stability affects experiments).*

### 1. Design Components:

- a. What kind of sensor will measure the temperature?
- b. How will the controller (e.g., a microcontroller) decide when to turn the fan on or off?
- c. How fast will the fan spin to cool down the room?

### 2. Feedback Loop Explanation: Write a brief explanation describing how the system works

- a. How does the sensor measure temperature?
- b. What happens when the room gets too warm or too cold?
- c. How does the system adjust the fan speed based on the temperature difference?

### 3. Design Your Diagram:

Use digital tools to create a **system diagram** showing the flow of information from the sensor to the controller, and how it controls the fan.

**Extension:** Add complexity by considering energy efficiency (e.g., turning the system off at night).

### 4. Class Presentation: Present your diagram and explanation to the class. Discuss why it is a feedback system, and how it adapts to different conditions.

### Activity 2.5 Design an Advanced Feedback System (Automated Watering)

*Your teacher will put you into small groups for this activity.*

**Scenario:** Design a system that automatically waters plants based on soil moisture levels. Before designing, have a **group discussion** using these prompts:

- What kind of sensor will you use to measure soil moisture?
- How will the controller know when to turn the water pump on or off?
- What device will deliver water to the plants?
- What role does the actuator play in the system?

#### 1. Component Diagram

- a. Create a **detailed component diagram** showing how the system's components are connected and interact.
- b. Diagram Requirements:
  - i. Show how the soil moisture sensor sends data to the controller.
  - ii. Indicate how the controller decides when to activate the water pump.
  - iii. Use arrows to represent the flow of information between components.
  - iv. Present the diagram to the class, emphasising how it works as a feedback loop.

#### 2. System Diagram (Advanced)

- a. Draw a **system diagram** that highlights the feedback loop:
  - i. **Input:** Soil moisture level.
  - ii. **Controller's Decision:** Whether to water the plants.
  - iii. **Output:** Amount of water delivered.
- b. **Consider External Factors:** Include how external factors like weather or soil type influence decision-making.

3. **Class Discussion:** Compare different group designs. Discuss the strengths and weaknesses of each system design.

## LOGICAL AND MATHEMATICAL MODELLING OF CONTINUOUS-TIME MACHINES

In the following section you will focus on analysing situations and creating mathematical models for machines that work continuously. Continuous-time machines, unlike those that work in steps (discrete-time machines), operate with constant changes. Examples include the smooth motion of a robotic arm or how temperature remains constant in a flow control



system. To understand how these machines work, you will learn how to use a powerful tool: mathematical modelling.

Here is why mathematical modelling is important:

1. **Design and Control:** Mathematical models act like blueprints for designing and controlling continuous-time machines. They show how the system behaves, helping you predict its actions and create good control strategies.
2. **Optimisation:** With models, you can simulate different situations and adjust settings to get the best performance. This lets you test control methods on the model before using them in the real world, which saves time and resources.
3. **Analysis and Understanding:** Models help you better understand the principles that guide how the machine works. you can check its stability, how fast it responds, and spot any potential problems.

## Key Mathematical Tools

### Differential Equations

Differential equations are equations that show how different variables relate to each other and how those relationships change over time. They are crucial for understanding the continuous behaviour of machines.

**Example:** When you model the motion of a robotic arm, you can use a differential equation that connects the arm's position, velocity, and acceleration to the torque applied to it.

### Laplace Transforms

Laplace transforms are a mathematical method that changes differential equations from the time domain (the variable  $t$ ) to the frequency domain (the variable  $s$ ). This transformation makes it easier to analyse how the system reacts to different input signals (or frequencies).

#### *Benefits of using Laplace transforms*

- a. **Simplified Analysis:** It makes it easier to study how the system behaves when looking at different frequencies instead of just time.
- b. **Control System Design:** It helps simplify the design of control systems by focusing on how they respond to various frequencies.
- c. **Manipulating Transfer Functions:** It allows you to work with transfer functions, which are mathematical expressions that come from Laplace transforms, making it easier to create control strategies.

## Common Real-Life Scenarios of Continuous-Time Machines

Continuous-time machines are all around us, constantly running in various systems. Here are some real-world examples:

## Robotics

1. **Robot arms in automation:** These arms need to move very precisely. This requires continuous control of joint angles and speeds, which is done through motor control systems that are modelled using differential equations.
2. **Mobile robots:** To move and navigate, mobile robots continuously adjust their speed and direction based on sensor data and environmental conditions.

## Process Control

1. **Temperature control systems:** In buildings or factories, these systems keep temperatures within a set range by constantly monitoring and adjusting based on how much heat is gained or lost. Differential equations help model these processes.
2. **Chemical reaction control:** Chemical processes, such as those in factories, need continuous adjustments of temperature, pressure, and flow rates to ensure quality. This is done by continuously monitoring these factors and using models to make adjustments.

## Case Study: Temperature Control System in a Chick Breeding Coop

### Scenario: Chick Breeding Coop Temperature Control

*In a chick breeding coop, keeping the temperature just right is very important for the chicks' survival and growth. The temperature needs to stay within a certain range to keep the chicks healthy.*

## Steps in Developing Logical and Mathematical Models

### 1. Determine the Objective

The main goal is to maintain the coop's temperature within a specific range around the ideal temperature, ensuring the chicks grow in healthy conditions.

### 2. Identify Key Variables

- a. **Current temperature ( $T_{\text{current}}$ ):** The temperature inside the coop at any given moment.
- b. **Optimal temperature ( $T$ ):** The best temperature for the chicks to grow.
- c. **Permissible temperature range ( $\pm t$ ):** The acceptable difference between the current and optimal temperatures.
- d. **Heating/Cooling system:** Adjusts the coop temperature by either heating or cooling it.

### 3. Developing Logical Models

#### Threshold Management

- a. If the temperature goes below the ideal range ( $T - t$ ), the heating system turns on.

- b. If the temperature rises above the range  $(T + t)$ , the cooling system activates.
- c. If the temperature is within the range, no action is needed.

#### 4. Developing Logical Models for the System

The system uses a logical model to manage how the heating and cooling work.

- a. If the temperature drops below  $T - t$ , the system turns on the heating.
- b. If the temperature rises above  $T + t$ , the system activates cooling.
- c. If the temperature is within the range  $T - t \leq T_{\text{current}} \leq T + t$ , no action is needed because the conditions are just right.

*If  $T_{\text{current}} < T - t$ , increase heating.*

*If  $T_{\text{current}} > T + t$ , activate cooling.*

*If  $T - t \leq T_{\text{current}} \leq T + t$ , no action required*

## Designing a Mathematical Model for Heating the Coop

The heating system operates based on the principle that the number of kilowatts of power applied directly influences the temperature increase inside the coop. In developing a model to determine the amount of power (in kilowatts) needed to be applied to regulate the heating in different situations, the following assumptions are made so the mathematical model is derived from it:

Assumptions:

- a.  $T_{\text{current}}$ : Current temperature inside the coop.
- b.  $T$ : Optimal temperature that you want to achieve (the target temperature)
- c.  $C$ : Thermal capacity of the coop (a constant based on its physical properties i.e. how much heat it can hold).
- d.  $P$ : Power in kilowatts (kW) applied to heat the coop.
- e.  $t$ : is the time interval (in seconds) for which the power is applied.

$$p = \frac{(T - T_{\text{current}}) * C}{\Delta t}$$

The equation calculates the power ( $P$ ) needed to achieve the desired temperature increase over a specific time interval. This ensures that the heating process is controlled and efficient, preventing overheating and ensuring the well-being of the chicks.

In the following example some actual figures have been used to show how to **calculate the power needed** to raise the temperature inside the chick breeding coop to the optimal value.

Consider you want to raise the temperature in the coop from **20°C** to **35°C**. The thermal capacity of the coop is **1000 J/°C** (a hypothetical value). This means that for every degree Celsius the temperature needs to increase, it requires 1000 Joules of energy. The time interval taken to do this needs to be **1 hour (3600 seconds)**.

**Step-by-Step:**

Calculate the temperature difference:

$$35^{\circ}\text{C} - 20^{\circ}\text{C} = 15^{\circ}\text{C}$$

- a. Use the formula from above and substitute  $15^{\circ}\text{C}$  and the values in the example:

$$P = \frac{(150^{\circ}\text{C}) * 1000 \frac{\text{J}}{^{\circ}\text{C}}}{3600\text{s}}$$

$$P = \frac{1500 \text{ J}}{3600\text{s}}$$

$$P = 4.17 \frac{\text{J}}{\text{s}},$$

Since 1 watt (W) = 1 joule/second (J/s), it follows that:  $P = 4.17 \text{ W}$

- b. Convert watts to kilowatts ( $1 \text{ kW} = 1000 \text{ W}$ ):

$$P = \frac{4.17}{1000}$$

$$P \approx 0.00417 \text{ kW}$$

**Conclusion:** To raise the temperature from  $20^{\circ}\text{C}$  to  $35^{\circ}\text{C}$  in one hour, you would need about **0.00417 kilowatts** of power.

## 5. Testing and Refining Models

Once you develop these models, you need to test them in the real world and refine them based on the results. This process helps to minimise errors and make sure the system works as expected.

By learning how to create and use logical and mathematical models, you gain the skills to design and control complex systems, like robots and automated machines, which need to operate continuously and adapt to changes in the real world.

### Activity 2.6 Designing a Temperature Control System

You have been asked to design a temperature control system for a small office room. Your goal is to keep the temperature inside the room comfortable, even if the temperature outside changes. You will use a heater to control the temperature, and your task is to develop a mathematical model for this system.

#### 1. Identify Key Components

- Write down a list of all the key parts involved in the system.
- Think about things like the **heater**, the **temperature sensor**, and the **controller** (the part that tells the heater when to turn on and off).
- Write a short description of what each part does in the system.

## 2. State the Main Goal

Write a clear sentence that explains the **main goal** of the system, for example:  
*“The goal is to keep the room temperature at 22°C, no matter what the temperature outside is.”*

## 3. Formulate the Solution

- a. Think about **how** the system will achieve this goal.
- b. Write down how the system will work, starting from how the temperature sensor measures the room’s temperature, how the controller decides when to turn on the heater, and how the heater works to adjust the temperature.

## 4. Develop Logical Models

- a. Create a **simple diagram** (draw it on paper or digitally) that shows how all the parts of the system work together.
- b. Include arrows showing how information flows from the sensor to the controller and to the heater.
- c. Use a **decision rule** for the heater, like:
  - i. “If the room temperature is below 22°C, turn the heater on.”
  - ii. “If the room temperature is above 22°C, turn the heater off.”
- d. Develop logical models for the decision rules.

## 5. Create a Mathematical Model for Heat Control

Write down the **formula** that shows how much power the heater needs to add to raise the room’s temperature to 22°C.

### Activity 2.7 Chemical Reaction Control Mathematical Model Design

In this scenario, you are working with a chemical reaction that requires a steady flow of a liquid reactant. You have a sensor to measure the current flow rate and a control valve to adjust the flow. Your task is to ensure that the flow rate stays constant.

*Follow the instructions below to guide your work.*

#### 1. Analyse the System

- a. Write a list of the **key parts** of the flow rate control system:
  - i. The **sensor** that measures the flow rate.
  - ii. The control valve that adjusts the flow rate.
- b. Describe the **desired flow rate** you want to maintain (this could be given in the problem, like “5 litres per minute”).
- c. Think about potential **error scenarios** where things could go wrong, like:
  - i. “What if the sensor stops working?”
  - ii. “What if the pressure suddenly changes?”

## 2. Write Error Detection and Correction Equations

- a. Write down a simple formula that checks whether the flow rate is correct:

$$\text{Error} = \text{Desired Flow Rate} - \text{Actual Flow Rate}$$

If the **error** is too big (e.g., more than 10%), you know something is wrong.

- b. Write what the system should do to correct the error. For example:
  - i. “If the error is positive (the actual flow rate is too low), the control valve should open more.”

*After writing it down, try to represent these in a mathematical model.*

## 3. Design Error Avoidance Strategies

- a. Think of ways to avoid errors before they happen.
- b. For example, you could install a **backup sensor** that takes over if the first sensor fails for example.

## 4. Present Your Solution

- a. Write out a description of your system and how it works to detect and correct errors.
- b. Include any diagrams that help show how your error correction works.
- c. Explain why your error prevention strategies make the system more reliable.
- d. Think about safety—how does your system make sure the chemical reaction happens safely, even if something goes wrong?

# BUILDING LOGIC WITH FINITE STATE MACHINES

Finite-State Machines (FSMs) are a useful tool for controlling systems that can only be in one specific state at a time. In this part of section 2 you will build on what you already know about FSMs—like their key parts: **states**, **transitions**, and **inputs/outputs**. You will learn how FSMs are used to design control systems in **automation and robotics**. You will see how FSMs can model the way these systems should behave by using **state diagrams** that show how the system moves from one state to another, based on signals from **sensors**.

You will learn how to set **control outputs** for each state and use simple **maths operations** in FSM models to make the system smarter and better at handling more complex tasks. By the end of this part of section 2, you will know how to use FSMs to design control systems for real-world projects in robotics and automation.

## What is a Finite-State Machine?

A **Finite-State Machine** is a way to model systems that change based on different events or inputs. Although a machine can have many possible states it can only be in one state at

any given moment. The word finite refers to the actual number of states that are possible for that machine. For example, imagine a vending machine—it can either be waiting for you to select an item, dispensing the item, or out of stock. These are its “states.”

FSMs are useful because they help **model and control** how a system should behave at different times (in different states) depending on various inputs and events. FSMs are used in robots, traffic lights, and even elevators!

## Key Concepts of FSMs

1. **States:** These are the different conditions a system can be in.  
Example: In a traffic light system, the states could be “Green,” “Yellow,” and “Red.”
2. **Transitions:** These are the rules for when the system should move from one state to another.  
Example: In a traffic light, the transition from “Green” to “Yellow” might be triggered by a timer.
3. **Inputs:** These are signals or conditions that trigger transitions between states.  
Example: In a vending machine, the input might be when a button is pressed to select an item.
4. **Outputs:** Each state can have its own action or output.  
Example: When the traffic light is in the “Green” state, it lights up the green light to allow cars to move.

## Translating System Requirements into FSMs

When designing FSMs, it is essential that you follow these steps:

1. **Identify States:** First, you need to figure out what the possible states of your system are. For example, in a vending machine, the states could be “Idle,” “Selecting Item,” “Dispensing Item,” and “Out of Stock.”
2. **Define Transitions:** Next, decide how the system will move from one state to another. In the vending machine, moving from “Idle” to “Selecting Item” could happen when a customer presses a button.
3. **Create a State Diagram:** A state diagram is a drawing that shows all the states and how the system moves between them. This makes it easier to see how the system works

## State Transitions Triggered by Sensor Inputs

In a system controlled by a **Finite-State Machine (FSM)**, **sensor inputs** are very important because they decide when the system should move from one state to another.

### Example: Traffic Light System

In a traffic light system, the states “Green,” “Yellow,” and “Red” are controlled by both timers and sensors:



1. **Timers:** These are used to control how long each light stays on.
2. **Sensors:** These detect when cars are waiting and help decide when the light should change.

For example, the system might switch from “Green” to “Yellow” when a timer runs out, and from “Yellow” to “Red” when a sensor detects no more cars.

## Control Outputs Defined for Each State

Each state in an FSM has specific **outputs**, which are actions that control how the system behaves. These outputs are commands that tell the system what to do.

### Example: Vending Machine

1. **Idle State:** The machine shows a message like “Select an Item” on the screen.
2. **Selecting Item State:** The machine activates the buttons so you can choose what you want.
3. **Dispensing Item State:** The machine gives you the item you picked and updates its inventory.
4. **Out of Stock State:** The machine shows “Out of Stock” and turns off the buttons so no one can select that item.

### How It Works:

Each state tells the system exactly what actions to take, like showing messages on the screen, turning on buttons, or physically giving out the item.

## Examples of Real-World Control Systems

### Traffic Light System

The following information shows you how a **traffic light system** can be designed using an FSM. Here are the states and transitions:

1. **States**
  - a. **Green:** Cars are allowed to pass.
  - b. **Yellow:** Cars are warned that the light is about to turn red.
  - c. **Red:** Cars must stop.
2. **Transitions**
  - a. **Green to Yellow:** This transition happens after a set amount of time (using a timer).
  - b. **Yellow to Red:** This happens after a shorter time, also based on a timer.
  - c. **Red to Green:** This is triggered when sensors detect that there are no more cars waiting.

Each state has a specific **output**. For example, in the “Green” state, the traffic light turns green, and cars are allowed to move.

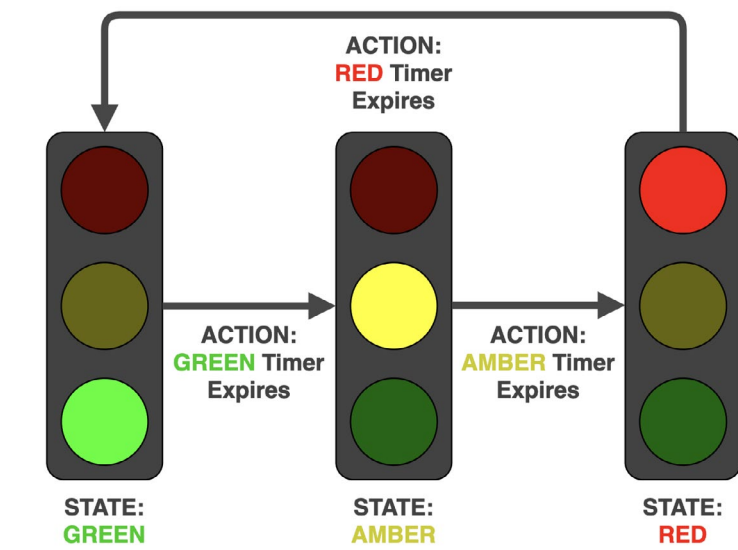


Figure 2.6: Finite state diagram of a traffic light controller

## Vending Machine

An FSM can also be used to design a **vending machine**

### 1. States

- Idle:** The machine is waiting for a user to select an item.
- Selecting Item:** The user is choosing what they want.
- Dispensing Item:** The machine gives the user their item.
- Out of Stock:** The machine has no more items to sell.

### 2. Transitions

- Idle to Selecting Item:** The user presses a button.
- Selecting Item to Dispensing Item:** The user confirms their selection, and the payment is verified.
- Dispensing Item to Idle:** After the item is given, the machine goes back to waiting for the next user.
- Any State to Out of Stock:** This happens if the inventory reaches zero.

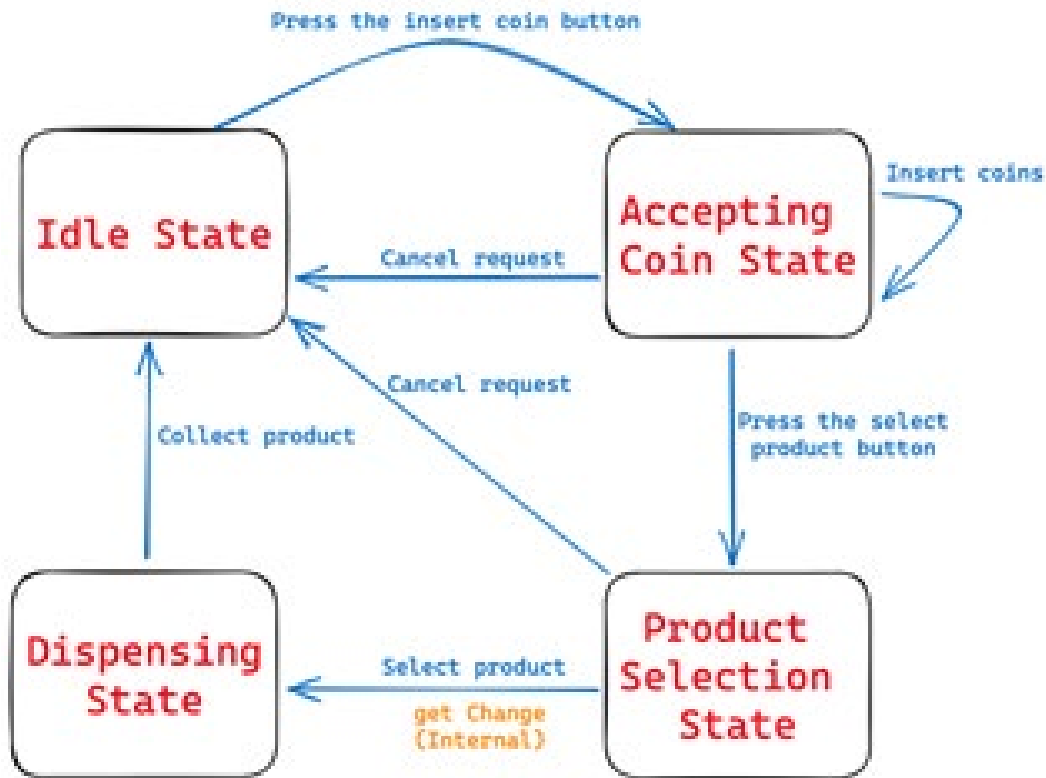


Figure 2.7: Finite state machine diagram of a vending machine

## Using Arithmetic Expressions for Better Control in FSMs

Finite-State Machines (FSMs) are great at controlling systems, but using only logical conditions (like “on” or “off”) can have some limits. By adding **arithmetic expressions** (simple maths), you can make FSMs smarter and more precise in how they control systems.

### Limitations of Using Only Logical FSMs

1. **Fixed Thresholds:** In a logical FSM, the system might move between states based on simple rules, like if a sensor reading is “high” or “low.” This can be too basic when you need more precise control.
2. **Limited Outputs:** The outputs are often just “on” or “off” actions, which does not work well when more detailed control is needed, like adjusting speed or distance.

### Benefits of Adding Arithmetic to FSMs

1. **Dynamic Control:** When arithmetic is used, you can control state transitions based on actual sensor data. This allows the system make decisions based on calculations, not just fixed rules.
2. **Improved Outputs:** Arithmetic allows you to calculate precise actions, like how fast a motor should go or how far a robot should move, based on real-time data from sensors.

## How to Use Arithmetic in FSMs

Basic maths operators like +, -, \*, and / are used in FSMs to calculate what the system should do next. These operators are used in expressions (equations) which can include sensor readings or even values from previous calculations.

### Example 1: Temperature Control System

In a basic FSM, the system might turn off the heater when the temperature is “high.” But with arithmetic, you can define a specific condition like:

If temperature > 35°C, turn off the heater.

This can be expressed mathematically:

$$\text{if } T > 35^{\circ}\text{C, then } H = 0$$

T represents the temperature in degrees Celsius.

H represents the heater’s state (0 = off, 1 = on).

This makes the system more precise and allows it to adjust based on the actual temperature.

### Example 2: Robot Arm

Imagine a robot arm that needs to pick up an object. In a simple FSM, the command might be “move arm” when the sensor detects an object. But with arithmetic, you can calculate exactly how far the arm needs to move using the formula:

$$\text{Distance to object} = \text{Object position} - \text{Current arm position}$$

This calculation helps the robot arm move the exact distance it needs to reach the object, making it much more accurate.

## Real-World Example: Robot Arm with Calculations

In this next section you will revisit the example of a robot arm picking up an object studied in an earlier part of this section. When controlling the movement of robot arms, the principle of **Euclidean distance** is used to calculate how far the arm needs to move. Euclidean distance is the straight-line distance between two points in space and applying this idea helps to ensure the robot arm move accurately to the object. In this case you will learn how this principle is applied to three dimensional space.

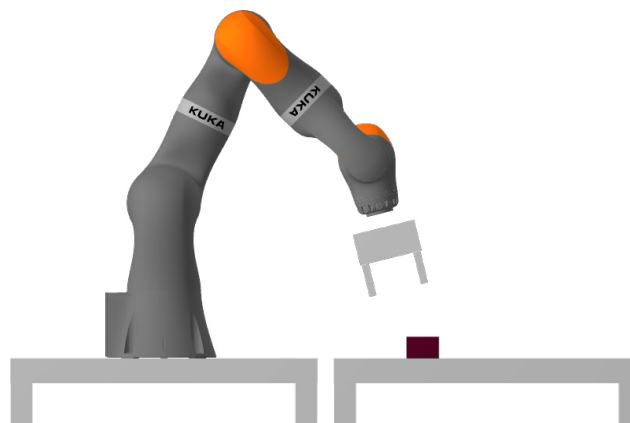


Figure 2.8: Robotic arm picking an object

## States of the Robot Arm

1. **Idle:** The robot arm is not doing anything and is waiting for instructions.
2. **Move to Object:** The robot arm calculates how far it needs to move and starts moving.
3. **Grip Object:** The robot arm activates its gripper to grab the object.
4. **Return to Idle:** After grabbing the object, the arm moves back to its starting position and waits for the next task.

## Transitions

**Idle to Move to Object:** The arm starts moving when the sensor detects an object.

**Calculation:** To work out how far the arm needs to move, the **Euclidean distance formula** is used. For 3D space, the distance between the robot's current position and the object's position this formula is:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$

## Example Calculation

1. The current position of the robot arm ( is
2. The object's position (x2,y2,z2) is (5,7,4)

Now, substitute these values into the formula above:

$$\begin{aligned} d &= \sqrt{(5 - 2)^2 + (7 - 3)^2 + (4 - 1)^2} \\ d &= \sqrt{(3)^2 + (4)^2 + (3)^2} \\ d &= \sqrt{(3)^2 + (4)^2 + (3)^2} \\ d &= \sqrt{34} \\ d &= 5.83 \end{aligned}$$

So, the robot arm needs to move **5.83 units** to reach the object.

3. **Move to Object to Grip Object:** This transition happens when the robot arm reaches the object. At this point, the arm stops moving and activates the gripper to grab the object.
4. **Grip Object to Return to Idle:** After successfully gripping the object, the arm needs to move back to its starting position (Idle).

**Calculation:** The distance the arm needs to move back is calculated the same way but in reverse. Since the arm is returning to its starting point, the distance will be the same, **5.83 units**. If the idle position changes, the new coordinates should be substituted into the formula to calculate the new distance.

By using **arithmetic expressions** like the Euclidean distance formula, the robot arm can accurately calculate how far to move to pick up the object. This makes the FSM model smarter and more precise. Instead of just moving a fixed amount, the robot knows exactly how far to go based on real-time data from its sensors. This ensures that the arm can always reach and retrieve objects with accuracy.

## Activity 2.8 Designing an Automated Watering System (FSM Model)

*Your teacher will put you into groups for this activity.*

In your group, design an FSM model for an automated watering system for a potted plant. The system uses a moisture sensor to determine the soil moisture level. Based on the sensor readings, the system activates a water pump to maintain optimal moisture conditions. Use the following prompts to complete the task:

1. **Define the States:** Use the following questions as a guide during discussions to identify the states of an automated watering system for a potted plant.

*What are the possible states the system can be in? Think about what the system is doing when it is not watering the plant and when it is actively watering.*

### Hint

You might have an “Idle” state (when the system is just checking the soil) and a “Watering” state (when the pump is on). What would a state where the moisture is perfect look like?

2. **Create Transitions Using Sensor Readings**

How does the system know when to move from one state to another? Think about what the **moisture sensor** is measuring.

### Hint

What should the system do if the moisture level drops too low? What if the soil reaches the right moisture level?

3. **Use Arithmetic Expressions**

- a. How would you decide when to start and stop watering? Create an arithmetic expression that checks the moisture level.

- b. Write an expression like:

If soil moisture < threshold, start watering

- c. How do you decide what the threshold should be?

Share your ideas and discuss different approaches with your class. Compare your FSM models to others and think about how your model could be improved.

4. **Define Control Outputs for Each State**

What happens in each state? For example, what should the system do in the “Watering” state? What should it do when the soil is wet enough?

### Hint

Think about when the water pump should be turned on or off. How will the system signal this?

5. **Incorporate Calculations into Control Outputs**

- a. The system might not always water for the same amount of time. How will you calculate how long to run the pump based on how dry the soil is?

*Prompt: Write an expression for the watering time based on how much the moisture level needs to increase to reach the desired level.*

- b. How do you find the **Moisture deficit**? How does the **Pump rate** affect the watering?
6. Present your **state diagrams** and **tables** to your teacher or peers and ask for feedback on whether your transitions and control outputs make sense.



## REVIEW QUESTIONS 2.1

1. Describe the role of the emergency break glass in a fire alarm system.
2. Explain why a fire alarm system is classified as a non-feedback system.
3. Discuss why non-feedback control systems, like the fire alarm system, might be used in situations where real-time feedback is unnecessary. Give at least two real-world examples where this approach is beneficial.
4. A conveyor belt is an example of a non-feedback system used in manufacturing. There are three components, a switch, an electric motor and a belt.
  - a. Identify the inputs, outputs and controller in this non-feedback system.
  - b. Explain why the conveyor belt is a non-feedback system.
  - c. Create a systems diagram for the conveyor belt.

## REVIEW QUESTIONS 2.2

1. Describe how a feedback loop works in a thermostat system.
2. How are feedback control systems used to control the speed of electric cars and the temperature in home heating systems? Briefly explain the purpose of the feedback in each system.
3. Describe the difference between the input and output of a feedback loop in an automated lighting system.
4. What would happen if a light sensor in an automated lighting system failed to detect the correct ambient light level? Provide a possible outcome and how it would affect the system.
5. Discuss the advantages of using feedback loops in control systems like thermostats or automated lighting systems. Consider the impact using a feedback loop has on energy use, changing environmental conditions, maintaining set control values, automation.

## REVIEW QUESTIONS 2.3

1. What is a continuous-time machine?
2. Give one example of the application of continuous time machines in robotics and explain the effect of using it.
3. What is the main environmental goal of a temperature control system in a room?
4. Explain why a temperature control system can be regarded as a continuous time machine.
5. Predict how the temperature control system would behave if the temperature sensor malfunctioned and always read a lower temperature than actual.
6. Evaluate the importance of feedback in a temperature control system.
7. A school wants to raise the temperature in classrooms from  $15^{\circ}\text{C}$  to  $22^{\circ}\text{C}$ . The thermal capacity of a typical classroom is  $200,000 \text{ J}/^{\circ}\text{C}$ . This means that for every degree Celsius the temperature needs to increase, it requires 200,000 Joules of energy. It takes one (1) hour (3600 seconds) to make the change. Work out the power required in kilowatts to raise the temperature in one hour from  $15^{\circ}\text{C}$  to  $22^{\circ}\text{C}$ .

## REVIEW QUESTIONS 2.4

1. Define a Finite-State Machine (FSM)
2. What are the main components of an FSM?
3. Identify and describe the states and transitions in a simple traffic light controller FSM.
4. Given a vending machine FSM with states “Idle,” “Selecting Item,” “Dispensing Item,” and “Out of Stock,” outline the transitions and control outputs for each state.
5. State the limitations of a purely logical FSM in a temperature control system and explain how incorporating arithmetic expressions can improve its functionality.
6. Evaluate the effectiveness of an FSM model for an automated watering system with states “Idle,” “Watering,” and “Moisture Sufficient.” Include the role of sensor inputs and transitions in your evaluation.
7. Create an FSM model for a robotic arm that includes arithmetic expressions for precise control. The states should include “Idle,” “Move to Object,” “Grip Object,” and “Return to Idle.” Define the transitions and control outputs.



SECTION

# 3

## SENSORS & ACTUATORS 2

# PRINCIPLES OF ROBOTIC SYSTEMS

## Sensors & Actuators

### INTRODUCTION

In this section, you will learn how to classify sensors based on how they work and the type of signal they produce. Sensors can be analogue or digital, depending on whether they give smooth, continuous signals or fixed, step-by-step signals. They can also be active or passive, based on whether they need power to send out signals or simply measure signals around them.

You will also learn how to change analogue signals, like temperature or light, into digital data using mathematics. Additionally, you will explore how to create graphs that show how far a robot moves based on how much its wheels rotate. Understanding these ideas will give you important skills for working with advanced robotics systems.

#### KEY IDEAS

- **Analogue to Digital Conversion:** Analogue to digital conversion is the process of changing real-world continuous signals, like light or temperature, into digital data that computers or electronics (microcontrollers) can understand and process.
- **Rotation-Distance graph:** It is a graph that shows how far a wheeled robot moves based on how often its wheel completes a full rotation. The relationship between the number of wheel rotations and the distance travelled by the robot is used in the control and navigation of a robot.
- **Sensor Classification:** Sensors can be categorised based on two key characteristics: their mode of operation (active vs. passive) and the type of signal they produce (analogue vs. digital).

## UNDERSTANDING ANALOGUE AND DIGITAL SENSORS

### Analogue vs. Digital Signals

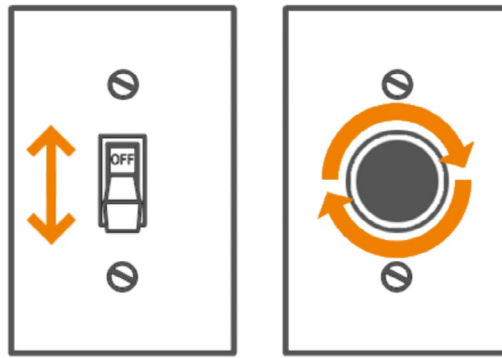
There are two types of signal: **analogue signals** and **digital signals**. You might have heard these words before when talking about things like radios, TVs, or clocks.

1. **Analogue signals** are smooth and continuous, meaning they can have any value at any time. Imagine a dimmer switch for a light as shown on the right in Figure 3.1. As you

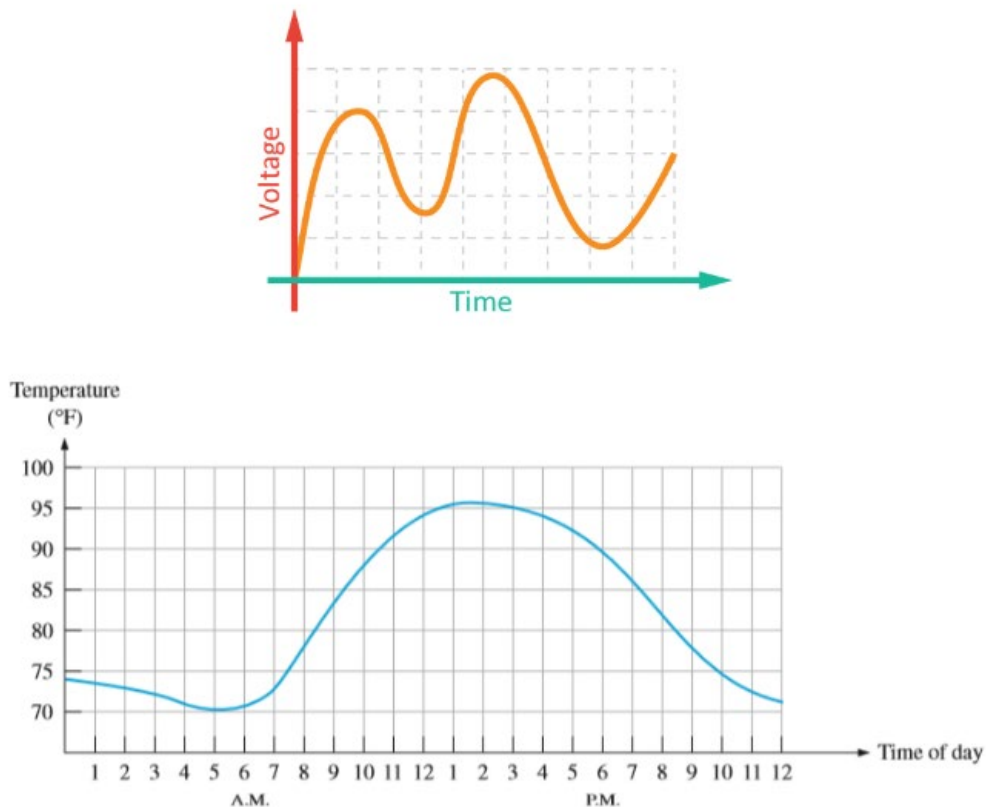


slowly turn it, the light gradually gets brighter or dimmer. The change is smooth, with no jumps. This is how an analogue signal works, smoothly changing over time.

2. **Digital signals**, on the other hand, are made up of clear, fixed or discrete steps. Think about a light switch that only has two positions—on or off as shown on the left in **Figure 3.1**. There is no in-between; the light is either fully on or completely off. This is how a digital signal behaves. It jumps from one value to another, like stepping between numbers.



**Figure 3.1: Digital and Analogue Switches**



**Figure 3.2: Examples of Analogue Signals represented on a graph.**

In **Figure 3.2**, the graphs show that the dependent variable (voltage or temperature) is on the y-axis, while time, the independent variable (time), is on the x-axis. Between the minimum and maximum **time** values on each graph, there are an uncountable (infinite) number of



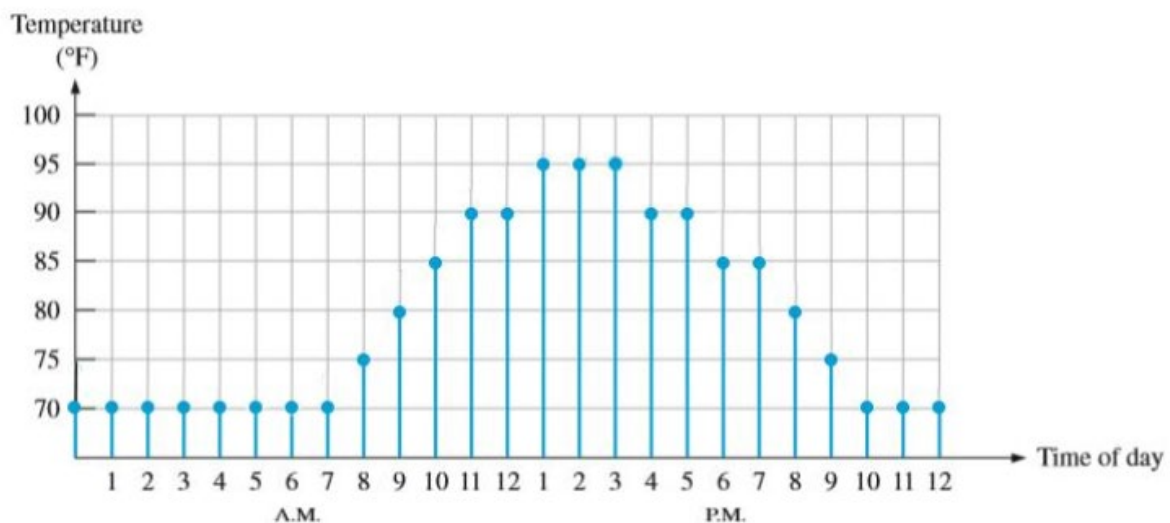
time points. You can picture this by thinking about the number of real numbers between 1 and 2 (1.1, 1.11, 1.2, 1.234, ...). For each of these points, there is a corresponding voltage or temperature reading on the y-axis.

All signals start off as analogue, which is why it is often said that people live in an analogue world. Things like temperature, sound, colours, humidity, and light are all analogue because they can vary smoothly from one value to another, there are no jumps. For example, the number of colours you can see, the sounds you can hear, and the smells you can detect are limitless, meaning there is an infinite range of possibilities. Given that most signals occur naturally, a question arises: how do digital signals emerge?

## Digital signals

Digital signals are different to analogue signals because they do not use continuous values but instead use discrete values. In robotics, the word “discrete” means specific or definite. A digital signal is rather like counting in a number sequence, each number represents a discrete value: 2, 4, 6, 8. So, a digital signal shows changes where the value only takes on specific finite points and is recorded at set intervals.

For the temperature graph in **Figure 3.2**, if you collected data only at certain times then this can be plotted to represent a digital signal.



**Figure 3.3: Digital Signal represented on a graph.**

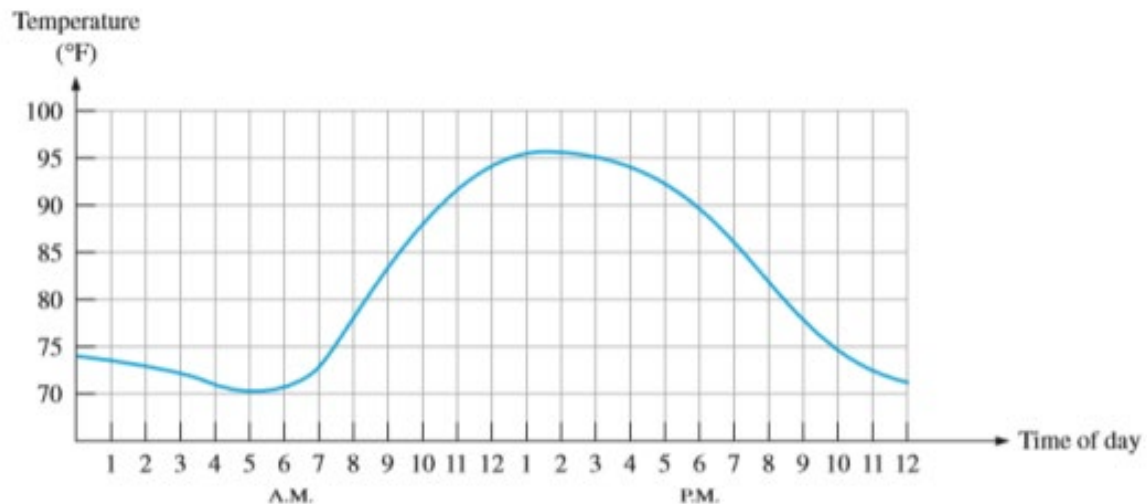
In **Figure 3.3**, you can see that there are no temperature records between the hours, thereby reducing the number of possible points on the graph from an infinite range of values to just 24 at hourly intervals. Also, the temperature readings are precise values like 70°C, 75°C, and 80°C.

This is similar to how the digital switch in **Figure 3.1** works. The switch is either on or off, the dimmer switch has a range of values between on and off.

## Analogue Sensors

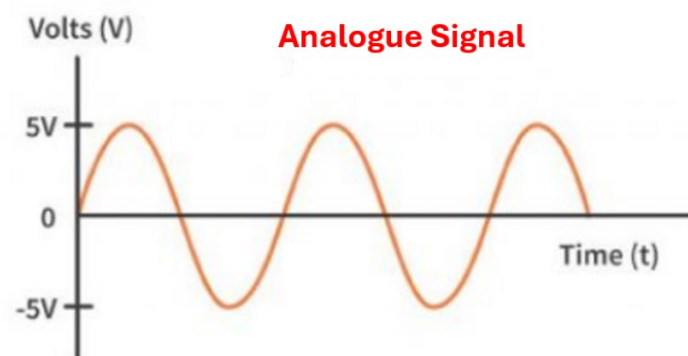
Here are the main features of analogue sensors.

1. **Continuous Output:** Analogue sensors give an output signal that can take any value in a given range, allowing changes to take place smoothly as the measured condition changes. For example, an analogue light sensor will gradually adjust its signal as the brightness of the light in its environment increases or decreases over time. Similarly, an analogue temperature sensor will slowly change its output as the temperature goes up or down over time as shown in **Figure 3.4**.



**Figure 3.4: The output of an analogue temperature sensor**

2. **Voltage (or Current) Output:** Analogue sensors typically produce output signals in the form of voltage or current signals. For example, in a temperature sensor, if the temperature rises, the amplitude of the voltage signal will increase in a way that matches the change in temperature. Note that amplitude is the height of the wave from its central axis to its peak (or trough). This way, the signal gives a clear visual representation of the measured condition. In Figure 3.5, the amplitude of the signal is 5V. If a temperature sensor measures a lower temperature than it currently is (indicating a decrease in temperature), this leads to a reduction in the amplitude of the signal.



**Figure 3.5: The output signal of an analogue sensor measured as a voltage**

3. **Precision Limitations:** Analogue sensors are less precise and accurate than digital sensors because they are easily affected by environmental factors like temperature changes and electrical noise. For instance, an analogue colour sensor may struggle to detect specific shades of red, especially under varying lighting conditions, as many reds can appear similar to oranges.
4. **Direct Connection:** Analogue sensors are directly connected to measurement devices. To process their signals with a computer, an ADC converts the analogue signal into digital data. For example, a heart rate sensor produces an analogue signal reflecting your heartbeat, which can be displayed as voltage fluctuations on an ECG/EKG or converted to digital form to show specific heart rates.
5. **Applications:** Analog sensors are commonly used in applications where continuous, real-time monitoring of physical quantities is important. Examples include:
  - a. **Thermocouples** for measuring temperature
  - b. **Heart Rate Sensor** for measuring heart rates/pulse
  - c. **Accelerometer** for measuring acceleration

## Digital Sensors

Here are the key features of digital sensors:

1. **Discrete Output:** Digital sensors give specific, fixed output values that can be easily measured. These values are usually displayed in binary code (0s and 1s), which makes them perfect for working with digital systems like computers and robots.
2. **Accuracy and Precision:** Digital sensors provide highly accurate and precise measurements, essential for reliable data in scientific research and medical testing.
3. **Digital Signal Processing Power:** Many digital sensors perform digital signal processing tasks to enhance their data. They can calibrate for accuracy, filter out noise, and compress data without losing important details. These capabilities make them versatile for applications like health monitoring, environmental sensing, and smart technology, where accurate and clear data is crucial.
4. **Communication Versatility and Microcontroller Compatibility:** The ease by which digital sensors can connect to networks and their compatibility with microcontrollers makes them valuable. Digital sensors feature communication interfaces and protocols that enable them to easily connect and share data with other devices like computers, smartphones, or microcontrollers. This allows for seamless integration in projects such as smart home systems, enhancing effectiveness in tasks like monitoring, controlling, and data analysis.
5. **Enhanced Features:** Many digital sensors can log data over time, have built-in clocks for timestamped measurements, and connect to visual displays for easy data viewing.

## Analogue vs. Digital Sensors

For some physical measurements, both analogue and digital sensors are available. For example, quantities like acceleration can be measured using either an analogue or a digital accelerometer, and temperature can be measured with an analogue thermocouple or a

digital temperature sensor. The choice between an analogue and digital sensor depends on the specific application requirements. The table below provides a comparison to help guide your decision.

**Table 3.1:** Comparison between Analogue and Digital Sensors based on application requirements

Requirement	Analogue	Digital
Need for Precision and Accuracy	May offer lower precision and accuracy.	Usually provide higher precision and accuracy.
Data Processing Capabilities	Require external processing (e.g., ADCs).	Often have built-in processing capabilities.
Nature of Data Required	Suitable for continuous, real-time data.	Suitable for discrete data points.
High-Fidelity, Continuous Data Acquisition	Excel in capturing detailed, real-time data (e.g., audio, vibration analysis).	May not capture as detailed real-time data.
Simplicity and Cost-Effectiveness	Often simpler and more cost-effective.	Tend to be more complex and expensive.
Ease of Integration	Require additional components (e.g., ADCs) to interface with digital systems.	Easily integrated with digital systems.

Passive vs. Active Sensors

Understanding the difference between passive and active sensors is important when selecting the right sensor for a task. Both types can be effective, however, power considerations in robotic designs may influence the choice. The table below shows some differences between passive and active sensors

**Table 3.2:** Differences between Passive and Active Sensors

Characteristics	Passive	Active
Power requirement	No external power needed. They operate by responding to ambient energy.	Require an external power source to generate the outgoing signal.
Energy Emission	Do not emit energy; they only detect and respond to existing energy in the environment.	Emit energy (e.g., sound waves, light, or radio waves) into the environment.
Detection Method	Respond to natural stimuli such as light, heat, magnetism, or vibrations.	Emit a signal and measure the reflected or returned energy to gather information.

Energy Efficiency	Energy-efficient due to lack of need for external power.	Less energy-efficient as they require power to emit signals.
Examples	Light sensors, photodiodes (light detectors), infrared sensors (temperature detection), microphones (sound detection).	Ultrasonic sensors (use sound for distance measurement), LiDAR sensors (use light waves for 3D mapping), radar sensors (use radio waves for object detection).

## Choosing the Right Sensor

When deciding between an active and passive sensor, it is important to know the specific task that the robot is to undertake. Here are some factors to consider when making this choice.

1. **Task requirements:** The robot's task dictates the information it needs. For line following, passive colour sensors detect the line's contrast. For obstacle avoidance, active ultrasonic sensors measure distances by sending out sound waves and timing their return. Thus, different tasks require different sensors to gather the right data.
2. **Environmental conditions:** Passive sensors can be affected by external factors like light or noise, causing incorrect readings. In contrast, active sensors generate their own signals, providing precise measurements even in challenging environments, making them more reliable in low light or noisy conditions.
3. **Power availability:** If saving power is important, passive sensors are better as they use less energy by detecting changes rather than generating signals. This makes them ideal for long-term, battery-powered use. Active sensors, though more precise, require more power due to their need to generate a signal.

The picture below shows some types of sensors typically used for Arduino projects.

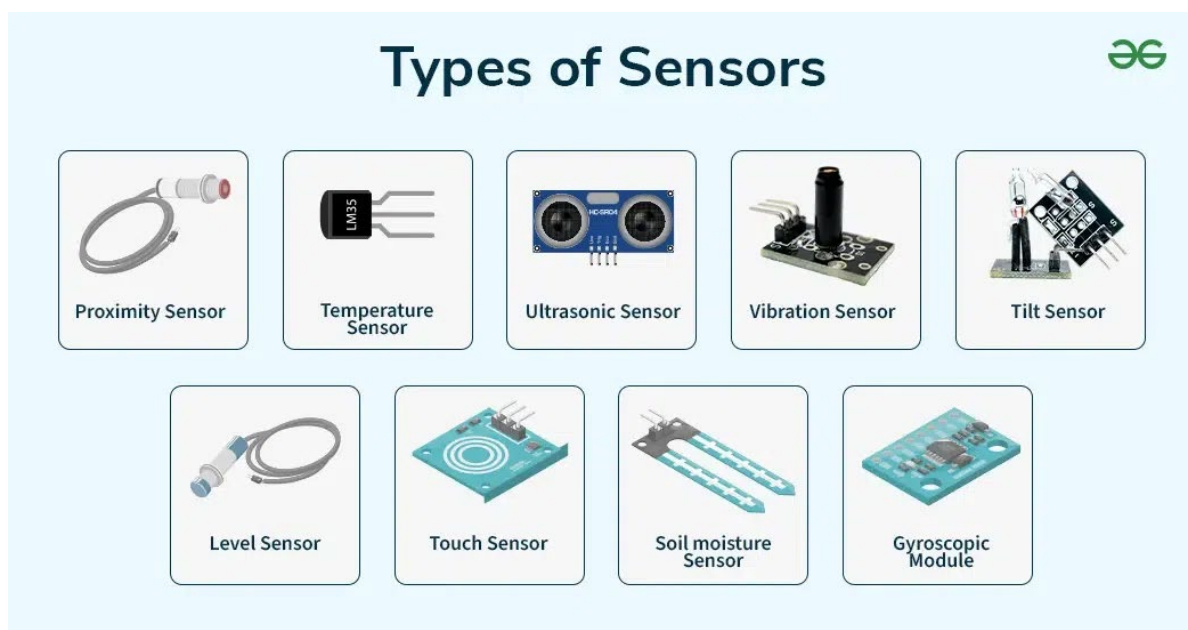


Figure 3.7: Examples of Sensors [<https://www.geeksforgeeks.org/types-of-sensors/>]

## Classifying Sensors

The information above helps to classify sensors based on **how they operate** and the **type of signal** they produce into

1. Active analogue
2. Passive analogue
3. Active digital
4. Passive digital

Each sensor will have its own datasheet that gives detailed information about a specific sensor model. It provides the sensor's;

1. **Specifications:** This refers to important details about the sensor, such as the voltage it needs to operate, the range it can measure, how sensitive it is to changes, how accurate its readings are, and how much power it uses.
2. **Functionality:** This explains how the sensor works by turning a physical property, like temperature, pressure, or light, into an electrical signal that a computer or device can understand and use.
3. **Pinout Diagram:** This is a visual guide that shows the connection points (or pins) of the sensor and explains what each pin does, helping you connect it correctly to other devices.
4. **Applications:** These are examples of how the sensor can be used in different projects, such as in robots, weather stations, or smart home devices, showing its practical uses in various fields.

## Measuring the Output Signals of Sensors

Multimeter and oscilloscopes are two devices that can measure the output signal of a sensor.

1. **Multimeter:** this tool measures voltage, current, and resistance and can be used to verify the output signal of an analogue sensor and determining its range.
  - a. **Analogue Multimeter:** This type features a needle that moves across a scale to display the measured value. (Figure 3.7, right)
  - b. **Digital Multimeter:** This version uses a numerical display to show the exact measured value, making it easier to read and interpret results. (Figure 3.7, left)

Both types usually have two probes: a red one (positive) and a black one (negative/ground). Connect the probes to the multimeter and the sensor. Set the multimeter to measure voltage. If the voltage changes smoothly, the sensor is analogue. If the voltage jumps between fixed values, the sensor is digital. For instance, an analogue temperature sensor's voltage varies gradually with temperature, while a digital sensor jumps between set values.





Figure 3.7: An Analogue and Digital Multimeter

2. **Oscilloscope:** This instrument is used to show electrical signals as graphs. It allows you to see how an analogue signal changes over time or how the voltage levels in a digital signal vary. It helps in understanding the behaviour of the signals by displaying them visually. Like multimetre, oscilloscopes come in two types - analogue and digital.
  - a. **Analogue Oscilloscope:** This type captures and displays the voltage waveform in its original form, allowing you to see the signal as it is.
  - b. **Digital Oscilloscope:** This version uses an analogue-to-digital converter (ADC) to capture and store information digitally. Because of their advanced features and capabilities, most engineers prefer digital oscilloscopes for debugging and design tasks.

It is important to note that both types of oscilloscopes can graphically represent both digital and analogue signals.

To check if a sensor's output is analogue using an oscilloscope, connect the probes to the sensor's output and ground. If the waveform is smooth and continuous, it is analogue. If it shows sharp transitions like square waves, it is digital. Analogue signals, such as from a microphone, show gradual changes, while digital signals have distinct high and low levels.

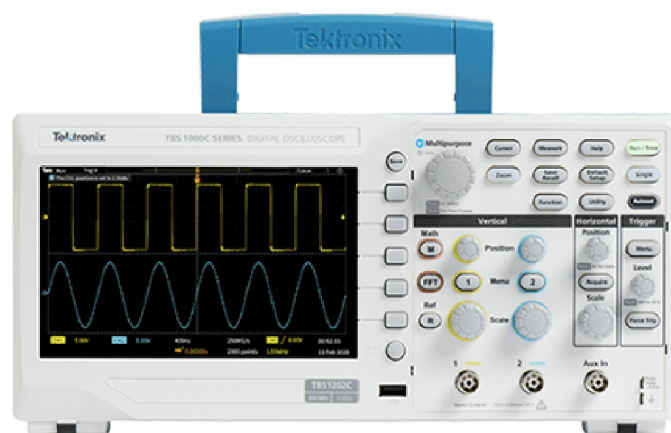


Figure 3.8: A Digital Oscilloscope





**Figure 3.9: An Analogue Oscilloscope**

*Follow the links below to read more about multimeters and oscilloscopes.*

#### Reference Link

<https://learn.sparkfun.com/tutorials/how-to-use-a-multimeter>

#### QR Code



<https://www.tek.com/en/blog/what-is-an-oscilloscope?bpv=2>



### Activity 3.1

Your teacher will put you into small groups for this activity and provide a number of sensors for you to handle from the Lego Spike Prime robotics kit.

1. Your first task is to study the Lego Spike Prime sensors state whether they are either passive or active sensors (use the table above to help you along with information from the data sheet if you have it)
2. Your second task is to study some sensor datasheets and review the information provide for users. In your groups search the internet and find some video clips or read books to find out more about different types of sensors and download their

datasheets. Collect your information together in a digital folder or write detailed written notes. Answer the following questions based on your research.

- a. Name three pieces of information you saw on the first page of each datasheet you analysed.
  - b. What information does the part number on a datasheet provide?
  - c. List some of the technical specifications of the sensors you analysed?
  - d. What is the relevance of 2D images of components in datasheets?
  - e. Do all datasheets have graphs?
  - f. Identify specific values (e.g., voltage range, output signal type) and explain how this information helps determine the sensor category (active/passive, analogue/digital) from the sensor datasheet you analysed.
3. In the same groups as for question one (1) search the internet and find some video clips or read books to watch videos to find out more about multimeters and oscilloscopes are, and how to use them. Answer the following questions based on your research.
- a. Describe the key features of an analogue and digital oscilloscope you have researched.
  - b. What does the horizontal and vertical axis of an oscilloscope represent?
  - c. What does the trigger function in an oscilloscope do?
  - d. Describe the key features of an analogue and digital multimeter you have researched.
  - e. To what voltage range should you set your multimeter if you want to measure the voltage of a 1.5V DC battery?
  - f. Describe how you can use a multimeter to decide whether a sensor is analogue or digital.
  - g. Under your teacher's supervision, measure the voltage of an AA battery and the resistance of a piece of wood using a multimeter. Discuss your results and share them with the class.

# MATHEMATICAL METHODS FOR SENSOR DATA PROCESSING IN ROBOTICS INTRODUCTION

Robots use sensors to understand their environment. Analogue sensors provide continuous data, while the microcontrollers used in robots can only use discrete digital signals. To bridge this, mathematical concepts like ratios and gradients are used to convert analogue data into digital form.

## Analogue Sensors and the Need for Analogue-To-Digital Converters (ADCs)

Most sensors in robotics are analogue, producing continuous signals. Microcontrollers, however, understand digital signals (0s and 1s). An Analogue-to-Digital Converter (ADC) is used to convert analogue signals into digital form for the microcontroller to process.

### What is an ADC and How Does it Work?

An Analogue-to-Digital Converter (ADC) converts continuous analogue signals, like sound or temperature, into digital form.

This involves two steps: sampling and quantisation.

**Sampling** involves the ADC taking voltage measurements at regular intervals.

**Quantisation** involves the ADC converting these measurements into binary codes that digital devices can process.

This is summarised in **Figure 3.10** below.

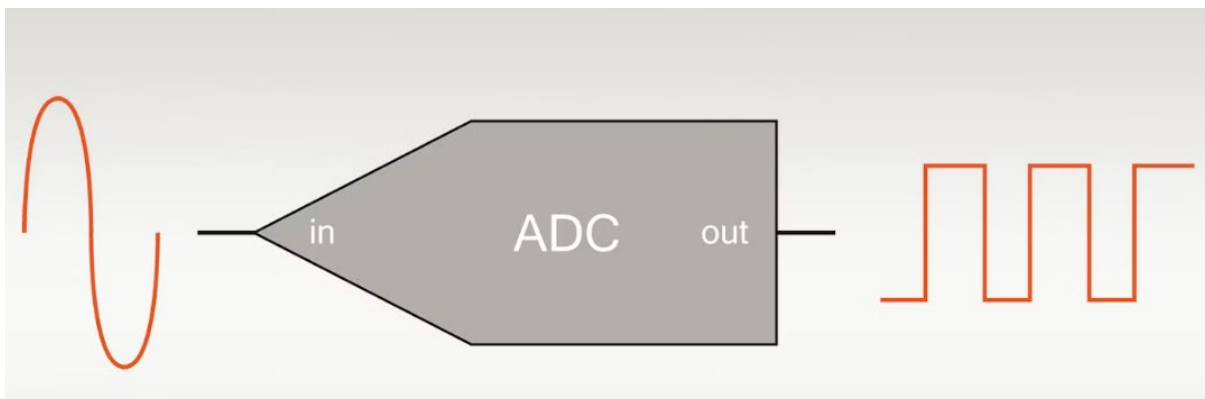


Figure 3.10: Analogue to Digital Conversion [Source: <https://dewesoft.com/blog/what-is-adc-converter>]

## Example Step-by-Step Approach of Converting Analogue Signal to Digital Signal

This example will use the water levels in a tank which is measured continuously over time – the analogue signal.

To change the water level data from analogue to digital Sampling and Quantisation are undertaken. The analogue data is represented by the continuous time graph in **Figure 3.11**.

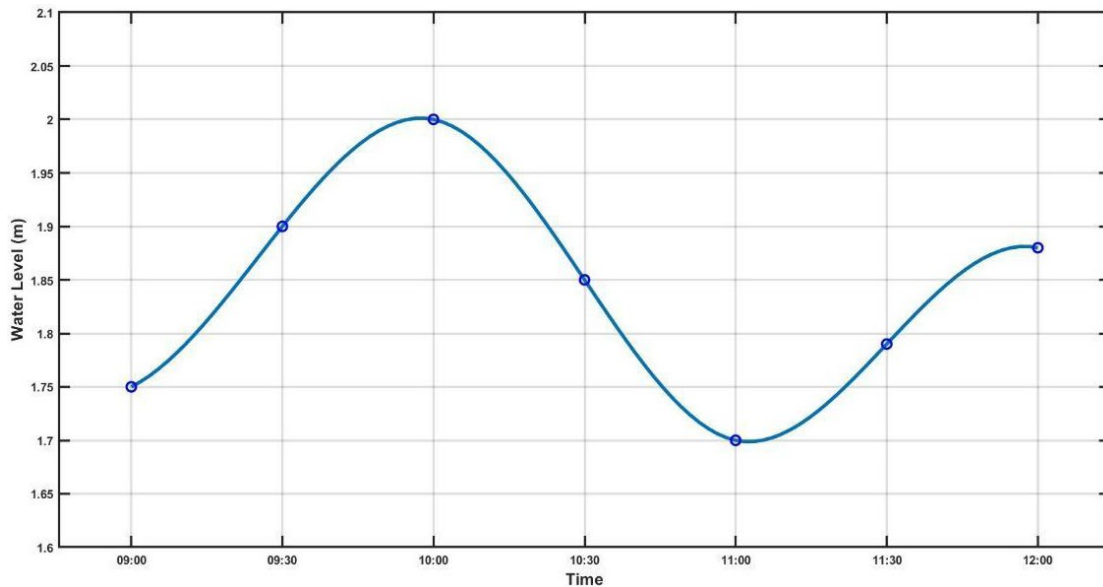


Figure 3.11: A Continuous-time Graph showing Water Levels Readings

### Step 1: Sampling (Discretising the Time Axis)

By sampling the water levels at fixed 30-minute intervals, a discrete-time signal is created as shown in **Figure 3.12**.

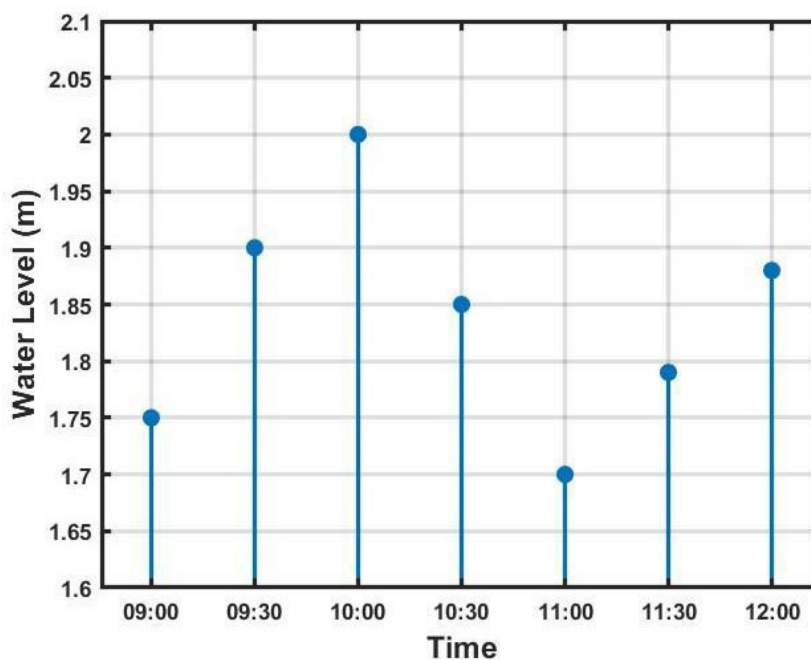


Figure 3.12: A Discrete-time Graph showing Water level Readings

**Table 3.3:** Discrete-Time Water Level Readings

Time	Water Level (m)
9:00	1.75
9:30	1.90
10:00	2.00
10:30	1.85
11:00	1.70
11:30	1.79
12:00	1.88

## Mathematical Representation of Sampling

Using the sampling data from **Table 3.3** the following formula is applied:

$$x[n]=x_a(nT)$$

Where:

- $x[n]$  is the sampled data which forms the discrete digital signal.
- $x_a(t)$  is the analogue signal at time  $t$ .
- $T$  is the sampling interval (how often you take the measurements).
- $n$  is an integer representing the sample number.
- $x_a(nT)$  is the value of from the continuous-time graph at time  $t = n * T$
- $*$  represents multiplication

The continuous-time graph (the graph is  $x_a$ ) shown in **Figure 3.11**, represents the analogue signal.

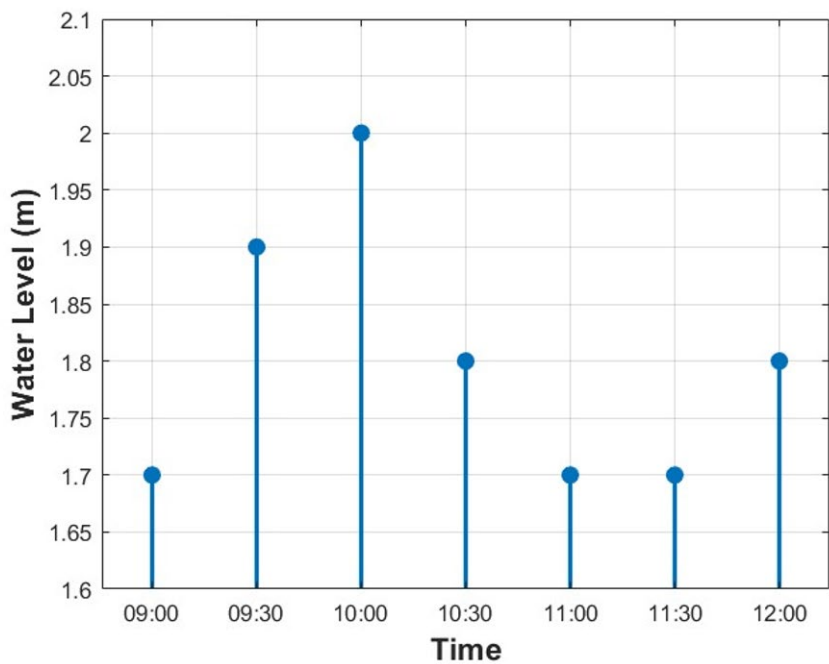
With the sampling interval  $T$  set to 30 minutes and the values of  $n$  ranging from 1 to 7 (representing the number of data points), the values of  $x_a(t)$  at times like  $x_a(1 * 30\text{mins})$ ,  $x_a(2 * 30\text{mins})$ ,  $x_a(3 * 30\text{mins})$  can be worked out.

This will give a discrete range of  $x[1] - x[7]$  for the sampled graph as shown in **Figure 3.12**, with  $x[1]$  having a value of 1.75m and  $x[2]$  having a value of 1.90m as shown in **Table 3.3**.

This means the value of the analogue signal is being measured every 30 minutes starting from time 9:00 which is the first sample time ( $n = 1$ ), just like taking water level readings at specific times of the day.

## Step 2: Quantisation (Assigning Discrete Values)

Digital signals differ from analogue signals because they only represent a fixed set of discrete values. While sampling only discretises the time axis ( $x$ -axis), quantisation discretises both the time and signal (value) axes (i.e. both  $x$ - and  $y$ -axis).



**Figure 3.13: Digital Signal**

Using the water level example in **Figure 3.13**, the measurements are to be quantised to the nearest 0.1 metres. This means that a recorded value of 1.85 metres quantised to the nearest 0.1 would be 1.8metres if rounded down or 1.9 metres if it were to be rounded up, depending on the systems rounding rules.

In the quantised version of this water level scenario, a rule can be applied which only allows readings like 1.7 m, 1.8 m, 1.9 m, and so on. The rule applied in this case is round down all values. For example, the actual reading at 11:30 which was 1.79 metres is rounded down to 1.7 metres which is the closest allowed value. Similarly, the reading at 12:00 which was originally 1.88 metres, is rounded to 1.8 metres. This is shown in **Table 3.4**.

**Table 3.4: Quantised Water Level Readings**

Time	Original Water Level (m)	Quantised Water Level (m)
9:00	1.75	1.7
9:30	1.90	1.9
10:00	2.00	2.0
10:30	1.85	1.8
11:00	1.70	1.7
11:30	1.79	1.7
12:00	1.88	1.8

## Mathematical Representation of Quantisation

This process can be described mathematically as:

- Define the Quantisation Level: This is the step size between discrete values (e.g., 0.1 m).
- Determine the Quantisation Range: This includes the minimum and maximum values (e.g., 1.7m to 2m).
- Round Each Sampled Value: Round each sampled value to the nearest quantisation level (e.g., 1.88 to 1.8).

Mathematically, this can be represented as:

$$xq[n] = Q[x[n]]$$

Where:

- $xq[n]$  is the quantised sample.
- $x[n]$  is the sampled value.
- $Q$  is the quantisation function.

In this case, the quantisation function which can be thought of as a floor function or method will simply round down the value to the nearest accepted value. In **Figure 3.13**, where temperature values are rounded down, the quantisation function does exactly this. It can be further expressed mathematically as:

$$xq[t] = QL * \text{floor}(x[t] / QL)$$

Where:

- $xq[t]$  is the quantised value at time  $t$ .
- $x[t]$  is the sampled value at time  $t$ .
- $QL$  is the quantisation level.
- The  $\text{floor}()$  function rounds down the value to the nearest whole number.

**Example:** Apply the above equations to the water level reading at 12:00 from **Figure 3.12** assuming the quantisation level is 0.1:

**Solution**

$$QL = \text{Quantisation level} = 0.1$$

$$x[t] = x[12:00] = 1.88\text{m (from Table 3.4)}$$

$$xq[12:00] = 0.1 * \text{floor}(1.88\text{m}/0.1)$$

$$= 0.1 * \text{floor}(18.8)$$

$$= 0.1 * 18\text{m}$$

$$= 1.8\text{m}$$

In this example, the level reading of 1.88m at 12:00 is rounded down to 1.8m, which is the nearest quantisation level based on a 0.1m step size.



## Percentage Error in Digital Signals

In digital signals, both the x and y axes are discretised, which means that only a specific set of values is allowed. When a value does not fall within this accepted set, it must be rounded up or down, leading to some error because the exact measured values are not always represented perfectly.

For example, rounding down a water level of 1.88m at 12:00 to 1.8m introduces a percentage error of 4.26%. This error can be calculated using the formula:

Percentage Error =  $(| \text{Approximate Value} - \text{Actual Value} | / | \text{Actual Value} |) * 100\%$

i.e. In the 6PM example above, Percentage Error =  $(|1.8 - 1.88| / 1.88) * 100\% = 4.26\%$

(The “| |” symbols mean absolute value, so even though  $1.8 - 1.88 = \text{negative } 0.08$ , positive 0.08 is used).

Note that percentage error is always positive because it represents the absolute difference between measured value and the true value, divided by the true value.

This error does not mean that digital signals are inaccurate or undesirable. In fact, there are several methods to manage these errors, and digital signal errors are often easier to handle than those in analogue signals. The benefits of digital signals will become clearer when comparing the analogue with digital signals

One simple way to reduce the 4.26% error in this example is to increase the number of intervals (quantisation levels) on the y-axis. Instead of only accepting values like 1.7m, 1.8m, 1.9m and 2m (with 0.1m intervals), you could reduce the interval to 0.05m. The set of accepted values would then include 1.7m, 1.75m, 1.8m, 1.85m and so on. This means that the value 1.88m at 12:00 would be rounded down to 1.85m instead of 1.8m, reducing the percentage error from 4.26% to 1.60%.

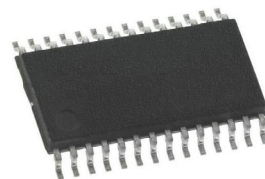
## ADC in Microcontrollers

Analogue-to-digital conversion (ADC) can be demonstrated more easily on some microcontroller boards than on others. For instance, most Arduino-based microcontrollers have analogue input pins that can receive signals from analogue sensors and convert these signals into digital formats. These boards often feature dedicated ADC pins.

In contrast, Raspberry Pi boards are designed primarily for digital input and output. When connecting an analogue sensor to a Raspberry Pi, you need to use an external ADC chip. This chip converts the analogue signal from the sensor into a digital format that the Raspberry Pi can understand. This setup requires additional hardware and programming.



ADC0804: An 8-bit ADC





MAX1131: A 12-bit, high-speed ADC

**Figure 3.14: Examples of commonly used ADC chips**

In educational platforms like the LEGO Spike Prime, the sensors, such as the colour sensor and distance sensor, primarily collect analogue data (e.g., light or pressure) but transmit this data digitally. Inside these sensors, an inbuilt ADC chip converts the analogue data before sending it to the main controller. This setup simplifies the process for beginners, allowing them to focus on programming and problem-solving with digital data.

While platforms like Arduino offer more flexibility for working directly with analogue signals, using LEGO Spike Prime helps beginners understand how digital systems work without dealing with the complexities of ADC programming. So far in the robotics course you have focused on LEGO Spike Prime, but future lessons will introduce Arduino-based boards, which will allow you to explore how ADC is handled programmatically.

Refer to the links below to learn more about Analog to Digital Conversion.

Reference Link	QR Code
LEGO Spike - Using Force Sensor <a href="https://www.youtube.com/watch?v=YtWkRYzpUSg">https://www.youtube.com/watch?v=YtWkRYzpUSg</a>	
Analogue to Digital Conversion <a href="https://www.youtube.com/watch?v=zucfv7IU0Ws">https://www.youtube.com/watch?v=zucfv7IU0Ws</a>	

### Activity 3.2

Your teacher will put you into groups of 3-5 and provide you with a continuous time graph.

- In this activity, you will learn how to convert continuous-time analogue signals (smooth curves) into both discrete-time graphs and digital signals by using the methods of sampling and quantisation. As a group study the continuous-time graph provided by your teacher and convert it into a discrete-time graph and digital signal graph, respectively. Plot your two new graphs and present your graphs to your classmates. Your presentation should answer the questions below.
  - What were the steps you took to achieve the two new plots?
  - What was your sampling interval?
  - What was your quantisation level?
  - What was the reasoning behind the values in b and c above?
- In this activity, you will calculate how much error was introduced when you converted the continuous-time signals to digital signals in the previous activity (Activity 3.1) and find ways to reduce these errors. In your groups from Activity

3.1, go online and watch a video or read an article on real-life scenarios that need high accuracy in their applications. Also, use your digital signal graphs to:

- a. Calculate the percentage error for each value you plotted.
- b. Calculate the average percentage error for the whole graph

*Present your results to your classmates. Your presentation should address the following issues;*

- i. How can you reduce the errors you calculated?
- ii. Use a real-life scenario to explain why it is important to reduce the errors.

## ROTATION-DISTANCE RELATIONSHIP IN WHEELED ROBOTS

Robotic systems often depend on wheeled movement, where understanding the connection between wheel rotation and the distance travelled is critical. This knowledge enables the accurate control of robotic motion, which is a key skill in both basic and advanced robotics applications. In this next part of section three you will explore the mathematical principles governing the movement of wheeled robots, focusing on the relationship between wheel rotation and distance travelled.

### The Mechanics of Wheeled Robots

The way wheeled vehicles move is very important for learning about robotics. From bicycles ridden every day to advanced machines like industrial robots, wheels are used everywhere. To truly understand how robots move, you must understand the relationship between the distance covered when a wheel turns and how far the robot travels.

### The Concept of Circumference

Just like bicycles, cars, and trucks, most robotic wheels are circular in shape. distance around the edge of the wheel is known as the circumference. To calculate the circumference of a circle, you use the formula:

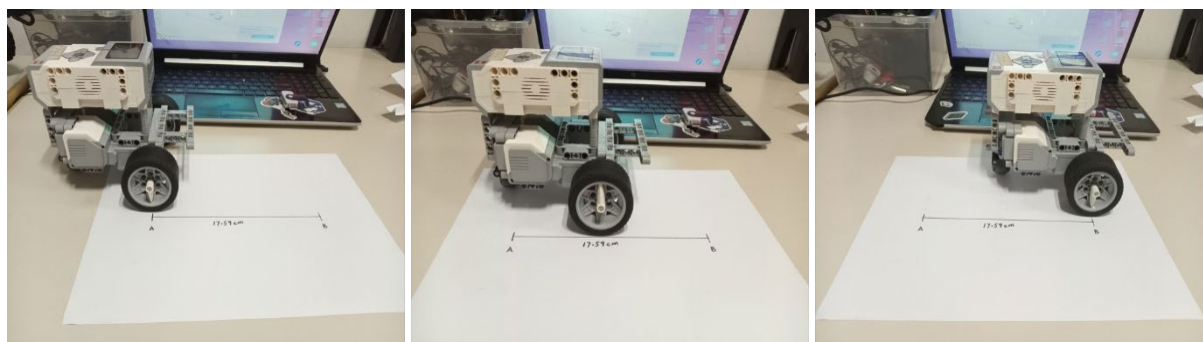
$$C = \pi d$$

In this formula, C is the circumference, d is the diameter (the distance across the circle), and  $\pi$  is a constant value approximately equal to 3.142. Since the diameter is twice the radius of the circle, the formula can also be written as:

$$C = 2\pi r$$

where r is the radius of the wheel.


In one single rotation a circular wheel covers the distance equal to the circumference of the wheel. This is illustrated in **Figure 3.14** where the wheel is seen to undergo one full rotation as seen from the position of the white pointer (referred to as a Bionicle tooth) in the three frames (the down, up, down motion).



**Figure 3.15: Rotational movement of a wheel**

In **Figure 3.15** the two-wheel robot has wheels with a radius of 2.8 cm, giving a circumference of about 17.59 cm. So, one full wheel rotation moves the robot 17.59 cm. The angle of wheel rotation relates to the distance travelled: 180 degrees (half rotation) covers half the circumference, and 90 degrees (quarter rotation) covers one-quarter. For example, a 90-degree rotation moves the robot  $0.25 * 17.59$  cm. Understanding this relationship between angular turns and distance helps predict and control the robot's movement.

What if the wheel completes five full rotations, what distance will the robot travel? Follow the video link below to watch a tutorial on the concepts covered in this lesson.

Resource Link	QR Code
<p>How to calculate rotations for distance with Lego Mindstorms EV3 robots</p> <p><a href="https://www.youtube.com/watch?v=2VEaebQuamc">https://www.youtube.com/watch?v=2VEaebQuamc</a></p>	

## Mathematical Modelling

Based on the knowledge that the circumference of a wheel is equal to the distance travelled in one rotation then you can see that in more than one rotation the robot will travel more than the circumference.

If 1 rotation = X metres which is also equal to the circumference of the wheel, then Y rotations will be equal to  $(Y/1) * X$  metres [i.e. if more, less divide] which can be written in words as

$$\text{Distance travelled} = \text{Number of rotations (Y)} * \text{Circumference (X)} \quad \text{eqn. 1}$$

For example, the distance travelled by the robot in Figure 3.15 after four full rotations of both wheels can be calculated as:

Distance = Number of rotations \* Circumference  
 Distance = 4 \* 17.59 cm = 70.36 cm

In similar fashion, the number of rotations required for the robot to move a specific distance can be calculated using the formula below which can be derived by simply making the Number of rotations the subject from eqn. 1, i.e.

Number of Rotations (Y) = Distance / Circumference (X)

For example: The number of rotations required by the robot in Fig 3.14 to move a distance of 1 metre can be calculated as:

Number of Rotations = Distance / Circumference  
 Number of Rotations = 1m / 17.59cm, (NB: 1m = 100cm),  
 Number of Rotations = 100cm / 17.59cm or 1 m / 0.1759m  
 Number of Rotations = 5.69 rotations

The distance moved can also be calculated using angles. One full rotation is equivalent to 360 degrees. As such, in the example above, the angle through which the wheels rotate to move the robot a distance of 1 metre can be calculated as:

Number of Rotations (degrees) = (Distance/Circumference) \* 360 degrees    OR  
 Number of Rotations (degrees) = Rotations \* 360 degrees  
 Number of Rotations (degrees) = (1m/0.1759m) \* 360 degrees = 5.69 \* 360 degrees  
 Number of Rotations (degrees) = 2048.4 degrees

### Activity 3.3 Checking How Far a Wheel Moves in One Turn

This activity will help you to verify that the distance a wheel moves is directly proportional to its circumference.

#### Materials You will Need

- Wheels from a robot. (It could be one from your available Lego kits, one you make from cardboard or a simulated one)
- Measuring tape or a ruler
- A piece of string or thread

#### Steps to Follow

##### 1. Measure the Wheel

- a. Use the string and a ruler to measure the wheel's diameter (This may be indicated on the wheel depending on the type you are using).
- b. Work out circumference of the wheel using the  $\pi d$  formula, where  $\pi = 3.142$  and  $d$  is the diameter. The result of the calculation is how far the wheel will move in one full rotation.

##### 2. Test Your Calculation

- a. Take the string and wrap it around the wheel to measure its circumference.

- b. Compare this string length with the number you calculated.
- c. Mark a point on the wheel or attach a Bionicle tooth to it.
- d. Move the wheel, so it completes exactly one full turn and measure how far it travelled.
- e. See if the distance the wheel moved matches the circumference you calculated earlier.

### 3. Do More Tests

- a. Try this experiment with wheels of different sizes to see how the size of the wheel affects the distance travelled in one full rotation.
- b. Look at your results to check if the relationship between wheel size and the distance it moves stays the same. i.e. how is the distance travelled related to the wheel size?

## Graphing and Understanding the Data

To help you see how wheel rotations affect the distance a robot travels, you can draw a graph using the data that you collected.

- a. On the x-axis (abscissa/horizontal axis) is the number of wheel rotations. This is the independent variable—it is what you choose to change.
- b. On the y-axis (ordinate/vertical axis) is the distance travelled by the robot. This is the dependent variable—it depends on how many times the wheel rotates.

Each point you plot on the graph represents one set of measurements, showing how far the robot has travelled after a certain number of wheel rotations.

For example, the wheel from the robot in **Figure 3.15** was seen to have a circumference of 17.59cm. If the wheel undergoes one full rotation, the robot should travel 17.59cm. If it turns twice, it should travel 35.18cm, and so on.

This relationship between rotations and distance is shown in the graph in **Figure 3.16**, and the actual numbers you used for plotting the points can be found in **Table 3.5**. These tools help us understand the connection between wheel rotations and how far the robot moves.

**B** Data points for Robot wheel with a circumference of 17.59cm

Number of rotations	1	2	3	4	5	6	7	8	9
Distance (cm)	17.59	35.18	52.77	70.36	87.95	105.54	123.13	140.72	158.31

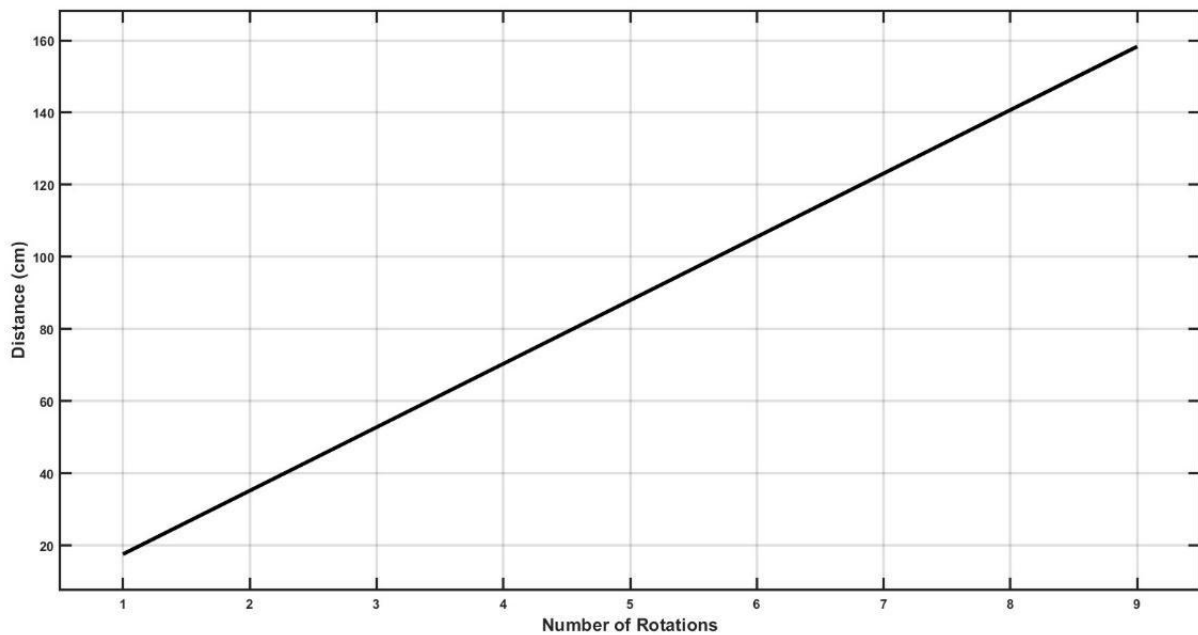


Figure 3.16: Rotation Distance graph for a robot wheel with circumference 17.59cm

### What Happens After You Plot the Data

Once you have put all your data points on the graph, you will start to see a pattern. If everything went well, the points should line up in a straight line as seen in Figure 3.16. This shows that the more the times the wheel rotates, the farther the robot moves. In other words, the number of rotations of the wheel and the distance moved are directly proportional, meaning that for every extra rotation, the distance increases by the same amount.

The steepness of the line on the graph is called the slope or gradient. In this case, the gradient tells you how far the robot moves with each wheel rotation. The steeper the line, the farther the robot travels for each turn of the wheel.

To figure out the gradient, you can use this formula:

$$\text{Gradient} = \text{change in } y / \text{change in } x = \frac{y_2 - y_1}{x_2 - x_1}$$

This means that for the graph in Figure 3.16, you can state that:

$$\text{Gradient} = \text{change in distance} / \text{change in rotations.}$$

For example, for 2 and 3 rotations, their corresponding distances are 35.18cm and 52.77cm, respectively. Using the formula above gives a gradient of:

$$\text{Gradient} = (52.77 - 35.18) / (3 - 2) = 17.59\text{cm}$$

Why is this important? The gradient actually tells you the circumference of the wheel as seen in this example. So, by calculating the gradient, you will know how far the robot moves for each wheel rotation—basically, the wheel's size!



## Programming Robots to Move

Now that you understand how wheel rotations relate to the distance a robot travels, the next step is to use this knowledge in robot programming. By controlling how many times the robot's wheels turn, you can make it move forward or backward over exact distances.

This is the basis for making the robot do more complicated things. You tell the robot's motors how many turns the wheels should make but getting it to move the right distance means you also need to think about things like wheel size, motor speed, and the type of surface the robot is on.

To make the robot move in a straight line, you will need to use more advanced programming. The robot might need to check its direction regularly using sensors and make small adjustments to stay on track. These corrections stop the robot from drifting off course. Watch the videos below to learn more about programming robots to move in a straight line.

Resource	QR Code
SPIKE Robot Programming (Part 1): Forward/Backward and Left/Right <a href="https://www.youtube.com/watch?v=Fs1MsJ0kwEo">https://www.youtube.com/watch?v=Fs1MsJ0kwEo</a>	
Perfect Straight Movement on the SPIKE Prime Robot!!! <a href="https://www.youtube.com/watch?v=WT-aUj57rpI">https://www.youtube.com/watch?v=WT-aUj57rpI</a>	




### Activity 3.4 Understanding How Wheel Rotations Affect Distance Travelled

Your teacher will put you into groups for this activity and give you a number of wheels of different diameters and a tape measure

1. In this activity, you will explore how the number of times a wheel rotates is related to the distance a robot travels. You will also see how different wheel sizes change the distance travelled with each rotation.
  - a. Use the measuring tape provided by your teacher to measure the diameter of your wheel (the distance across the wheel from one side to the other). Write down this measurement.
  - b. Next, use the string provided to wrap around the wheel to measure the distance around it.
  - c. Then, check your result by calculating the circumference using the formula for circumference and the diameter you measured in a.

- d. Now, predict how far your group's wheel will move in one full rotation by using the circumference you calculated. Mark a starting point on the floor and a mark on your wheel, then roll the wheel forward for one full turn and see if it matches your prediction.
  - e. Repeat points a to d for wheels of different sizes.
  - f. Compare your results with other groups. Discuss why larger wheels move farther than smaller ones with each rotation.
2. In this activity, you will use your knowledge of wheel rotations to program a robot to move specific distances. You will learn how to control its movement by calculating the number of wheel turns needed. In the same groups as the previous activity search the internet, for a video clip on the basic commands needed to make a robot move forward or backward (watch one related to the robotic kit you have available like LEGO). Now it is time for a challenge! You will program your available robot to move exactly 1 metre.
  - a. Use what you learned in the first activity to calculate how many wheel rotations are needed for the robot to travel this distance. Then, program the robot to move that number of turns.
  - b. Run your program and measure the actual distance your robot travelled. Did it go the full metre? If not, adjust your program and try again until you get the distance right.
  - c. After testing, discuss what happened with your classmates.
    - i. Did different groups have different results? Think about how changes in wheel size or the type of surface might have affected the robot's accuracy.
    - ii. If you want a bigger challenge, you can also experiment with programming the robot to correct its path using sensors, making sure it moves in a straight line.

## EXTENDED READING

Resource	QR Code
<p>How to Convert Analog Signals to Digital Signals    Electronic Terminology Course Preview</p> <p><a href="https://www.youtube.com/watch?v=OBcCZkCf-OI">https://www.youtube.com/watch?v=OBcCZkCf-OI</a></p>	
<p>Sampling and Quantisation of Analog Signal [HD]</p> <p><a href="https://www.youtube.com/watch?v=W5q-Ac0JVdk">https://www.youtube.com/watch?v=W5q-Ac0JVdk</a></p>	
<p>Get Your SPIKE Prime Robot Moving Straight Forward</p> <p><a href="https://www.youtube.com/watch?v=gAnNf3ji-Ig">https://www.youtube.com/watch?v=gAnNf3ji-Ig</a></p>	

## REVIEW QUESTIONS 3.1

1. Define the term “continuous value” in the context of analogue signals
2. State 3 differences between passive and active sensors.
3. What type of sensor (analogue or digital) will be more suitable for a scenario where the sensor is to detect whether a button has been pressed to switch light on or off? Explain your reasoning.
4. State and explain any 3 properties of a sensor that can be found on a sensor datasheet.
5. Why might you choose an active sensor over a passive one for certain tasks?

## REVIEW QUESTIONS 3.2

1. What is the difference between an analogue signal and a digital signal?
2. What are the two main steps involved in converting an analogue signal into a digital signal?
3. Calculate the percentage error for each data point in the table below.

Time	Sampled values (cm)	Quantised values (cm)
7:00	74.78	70
8:00	83.50	80
9:00	92.00	90
10:00	88.97	85

4. If you were working with a device that required a high level of accuracy in signal conversion, what adjustments could you make to minimise errors?

## REVIEW QUESTIONS 3.3

1. What is the circumference of a robot wheel with a diameter of 10 cm?
2. If a robot wheel has a diameter of 5 cm and completes 3 rotations, how far will the robot travel?
3. A robot needs to move forward 2 metres. If its wheels have a diameter of 8 cm, how many rotations are needed?
4. Why does a robot with larger wheels travel further with the same number of rotations compared to a robot with smaller wheels?





**SECTION**

# 4

## **DIGITAL AND ANALOGUE SYSTEM DESIGN**



# ROBOT DESIGN METHODOLOGIES

## Digital and Analogue System Design

### INTRODUCTION

Imagine you are building a robot. You need to make sure it can make decisions like “Should I move forward or turn left?” Boolean Algebra is like a magical tool that helps you give robots the brainpower to make these decisions. In this section, you will learn how to use Boolean Algebra to design the “brain” of a robot. You will use simple building blocks called logic gates to create complex thinking patterns. It is like building with LEGO bricks, but instead of bricks, you will be using logic gates to build a robot’s mind.

#### KEY IDEAS

- **Boolean Algebra:** This is a foundational concept in digital system design. It allows you to define the logic behind digital circuits using variables (representing logical values) and operators (for example AND, OR, NOT) to express relationships between these values. By applying Boolean algebra rules and theorems, you can simplify complex expressions, leading to more efficient and cost-effective digital circuits.
- **Karnaugh Map (K-Map):** This is a visual tool used to simplify Boolean expressions further. They represent truth tables in a grid system, allowing for easier identification of patterns and redundancies within the logic. This is necessary for optimising digital systems, which make all robots work correctly.
- **Soldering Techniques:** Assembling robots requires not only an understanding of system schematics but also practical skills like soldering and PCB (printed circuit board) assembly to ensure the electronic circuits work correctly.

### INTRODUCTION TO BOOLEAN ALGEBRA

Imagine trying to communicate with a friend who only understands “YES” and “NO” — this is exactly how computers work, using a language called binary code. A **binary system** is a way of representing information using only **two possible values**. In the language of computers YES is given the **value one** (1) and NO the **value zero** (0). That is the world of binary code, the language of computers. To make sense of this binary world, robotics uses a special kind of maths called Boolean Algebra. A long time ago, a clever mathematician named George Boole created this maths. He used **symbols and operators** to represent **YES or TRUE** (1) and **NO or FALSE** (0), and he figured out rules to combine these symbols and operators to solve logic problems. This is a lot like **solving logic puzzles** with numbers, but instead of numbers, digital engineers use **TRUE and FALSE**.

Boolean Algebra is like the building blocks of computer science and robotics. It helps engineers design the circuits that make computers and robots work. By understanding Boolean Algebra, you can model complex systems, from simple calculators to powerful robots. It is like learning a secret language that only computers understand.

## Importance of Boolean Algebra in Digital Systems

All digital devices, such as computers, smartphones, smart TVs, digital cameras, gaming consoles, and robots, make use of the binary number system to encode all their signals. In the binary number system, information is represented using just two states: 0 (which represents False) and 1 (which represents True). Boolean algebra uses this binary system to achieve the following in these systems.

1. It helps understand how the circuits in these digital systems work.
2. It helps describe the connections between different parts of a circuit.
3. By understanding and describing these circuits, you can make these circuits simpler and faster (optimisation).

## Key Concepts and Rules of Boolean Algebra

Boolean algebra uses several basic concepts and rules. It is only by becoming familiar with these rules that one can better apply them in understanding digital systems. The following are some important concepts to take note:

1. **Variables:** Imagine you have a light switch. It can be either ON or OFF. By using a letter, like “A”, to represent the switch. When the switch is ON, then  $A = 1$ . When it is OFF, then  $A = 0$ . The letter you use to represent the state of the switch is known as a “Variable”. These variables in Boolean algebra represent logical values, and are represented by letters like A, B, C. They can only have one value, either True (1) or False (0).
2. **Logical Operators:** Imagine you have a robot that needs to make decisions. It needs to know when to move forward, turn, or stop. You can use **logic gates**, which are like small digital circuit based **decision-makers**, to help the robot. A logic gate in an electronic circuit receives a signal, processes it, and sends it to the next step, much like a gate opens to let someone pass through. These are used to perform basic logical functions or operations on binary values (0s and 1s). These operations/operators implemented by logic gates are fundamental in building digital circuits. Here are some common logic gates.
  - a. **AND Gate:** This gate only lets the signal through (or outputs 1) when both of its inputs are 1. It is like saying, “If this and that is true, then do this.” This operation is usually represented using a simple dot (.) or simply by placing the variables next to each other. For example, A AND B can be written as  $A \cdot B$  or simply AB. It can also be represented diagrammatically, as shown in **Figure 4.1**.

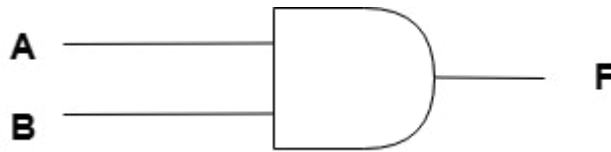


Figure 4.1: Two-input AND Gate

The AND gate represented in **Figure 4.1** has two inputs (A and B) and one output (F). Below is a simple **truth table** for this two-input AND gate. You can see that from all the possible signal (or input) combinations there is only ONE which has two inputs that are 1.

A	B	$F = A.B$
0	0	0
0	1	0
1	0	0
1	1	1

(An important learning point for you at this stage is that the term “truth table” comes from the table’s function of displaying the “truth” (or output value) for every possible combination of input variables, making it a fundamental tool in Boolean algebra and digital logic design)

- b. **OR Gate:** This lets the signal through if at least one of its inputs is 1. It is like saying, “If this or that is true, then do this.” This is often represented by a plus sign (+). For example, A OR B is written as  $A+B$ . It can also be represented diagrammatically, as shown in **Figure 4.2**.



Figure 4.2: Two-input OR Gate

The OR gate represented in **Figure 4.2** has two inputs (A and B) and one output (F). Below is a simple truth table for this two-input AND gate.

A	B	$F = A+B$
0	0	0
0	1	1
1	0	1
1	1	1

- c. **NOT Gate:** This gate inverts (reverses) the signal input. If the input is 1, the output is 0, and vice versa. It is like saying, “If this is true, then make it false.” NOT is represented by a bar over the variable ( $\bar{}$ ). It can also be represented diagrammatically, as shown in Figure 4.3. So, the inverse of 1 is 0, in Boolean Algebra this is written as the inverse on the variable A is

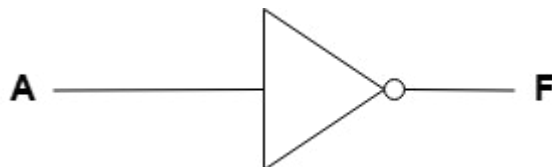


Figure 4.3: NOT Gate

The NOT gate represented in **Figure 4.3** has one input (A) and one output (F). Below is the truth table for this NOT gate.

A	F =
0	1
1	0

- d. **NAND Gate:** The NOT Gate can be combined with the AND Gate to produce a NAND Gate (NOT-AND). It implies an AND gate with a complemented or inverted output. It can also be represented diagrammatically, as shown in Figure 4.4. The circular shape at the output represents the inversion (change to opposite).

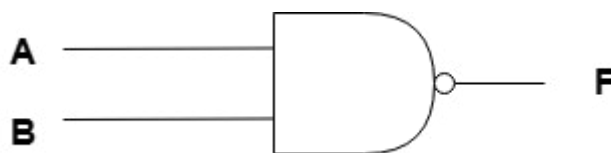


Figure 4.4: Two-input NAND Gate

The NAND gate represented in **Figure 4.4** has two inputs (A and B) and one output (F). Below is a simple table for this two-input NAND gate.

A	B	AB	F = ()
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

(An important learning point for you at this stage is that the term “complemented” in Boolean Algebra means “reversed” or “inverted. The complement of a Boolean

variable is the opposite value of the variable. If the variable is 1 (TRUE), its complement is 0 (FALSE), and vice versa)

- e. **NOR Gate:** The NOT Gate can be combined with the OR Gate to produce a NOR Gate (NOT-OR). It implies an OR gate with a complemented (inverted) output. It can also be represented diagrammatically, as shown in Figure 4.5. The circular shape at the output represents the inversion (change to opposite).

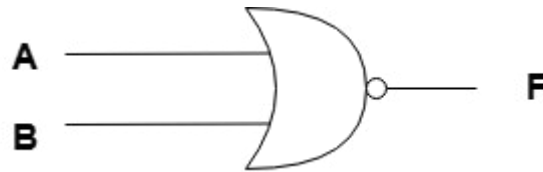


Figure 4.5: Two-input NOR Gate

The NOR gate represented in **Figure 4.5** has two inputs (A and B) and one output (F). Below is a simple table for this two-input NOR gate.

A	B	A+B	$F = (A+B)'$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

Some other logic gates or operations, which may not be common but are sometimes used, are as follows:

- a. **XOR (Exclusive OR):** An XOR gate only activates if one of its inputs is on, but not both. It outputs 1 when the inputs are different and 0 when they are the same. It is like saying, “one or the other, but not both”. It is represented by a circle with a plus sign inside ( $\oplus$ ). Example:  $1 \oplus 0 = 1$ , but  $1 \oplus 1 = 0$ .

It can also be represented diagrammatically, as shown in **Figure 4.6**.

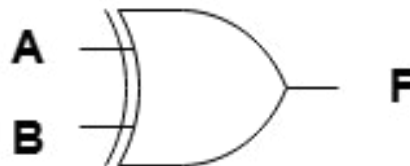


Figure 4.6: Two-input XOR Gate

Below is a simple truth table of an XOR operator.

A	B	$F = A \oplus B$
0	0	0

0	1	1
1	0	1
1	1	0

- b. **XNOR (Exclusive NOR):** An XNOR gate is the opposite of an XOR gate. It turns on only when both switches are in the same state: either both on or both off. It outputs 1 if both inputs are the same (either both 0 or both 1). It is like saying, “either both or neither.” (i.e., Outputs 1 if the inputs are the same, and 0 if they are different). It is represented by a circle with a dot ( $\odot$ ). Example:  $1 \text{ XNOR } 1 = 1$ , and  $0 \text{ XNOR } 0 = 1$ , but  $1 \text{ XNOR } 0 = 0$ . It can also be represented diagrammatically, as shown in Figure 4.7.

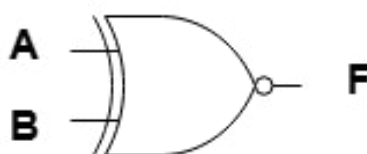


Figure 4.7: Two-input XNOR Gate

(note that the circle at the output of the XNOR gate represents an inversion or negation of the XOR gate output)

Below is a simple truth table of an XNOR operator.

A	B	F = AB
0	0	1
0	1	0
1	0	0
1	1	1

3. **Boolean identities:** These are equations or expressions that are always true regardless of the values of the variables involved. Examples include:
- $A + A = A$  (if you have a condition (A), and you combine it with itself using the OR operation, it does not change the outcome. The result is still (A). In other words, ORing a variable with itself does not change its value. So, in this case  $1 + 1 = 1$ )
  - $A \cdot A = A$  (if you have a condition (A), and you combine it with itself using the AND operation, it does not change the outcome. The result is still (A). In other words, ANDing a variable with itself does not change its value. So, in this case  $1 \cdot 1 = 1$ )
  - $A + \bar{A} = 1$  (This principle is known as the Law of Complementation in Boolean algebra. It signifies that any Boolean variable ORed with its complement will

always result in TRUE, indicating the system's ability to account for both possible states of *A*. So, in this case  $1 + 0 = 1$ .

By understanding these concepts and applying the rules of Boolean algebra, you can create expressions that represent the logic behind digital circuits. These expressions can then be translated into actual circuit designs using logic gates (for example AND, OR, NOT gates).

## Important Note on Truth Tables

Just as demonstrated in the logic gates demonstrated above, **truth tables** are used to show **all the possible combinations of inputs and their corresponding outputs** for a given logic gate or circuit. It is important to note that the number of inputs determines the number of outputs. Since this is binary data, a **single variable or input** will have **2** () possible outputs (0 and 1).

The combination of **two inputs** (*A* and *B*) will have **4** () outputs or outcomes determined from the four possible combinations (00, 01, 10, 11), which can be matched to the **four rows** in the table below.

A	B	F
0	0	
0	1	
1	0	
1	1	

Now for **3 inputs** (*A*, *B* and *C*), there will be **8** () possible combinations (000, 001, 010, 011, 100, 101, 110, 111) which can be matched to the **eight rows** in the table below.

A	B	C	F
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	



So, the rule is for number of inputs, there will be possible outputs. Also, note the logical treatment of the all the possible combinations of 1's and 0's. They are always arranged in increasing order, counting in binary from 0 to - 1, just as demonstrated in the tables above.

## Application of Boolean Algebra in Digital Systems

Boolean algebra acts as a good tool for defining the control logic behind various automation solutions. For example:

1. You can use Boolean variables to represent the different control conditions or states of sensors and actuators in an automated system.

For example, a variable “LightSensor” could be True (1) if light is detected and False (0) if light is not detected.

2. You can use Boolean operators (AND, OR, NOT) to express the desired relationships between these conditions.

For example, an automatic light can be controlled by the following expression:  $\text{LightOn} = \text{LightSensor} + \text{MotionSensor}$ . This can be expressed in Boolean terms as LightOn will be TRUE if either LightSensor **or** MotionSensor\ is TRUE.

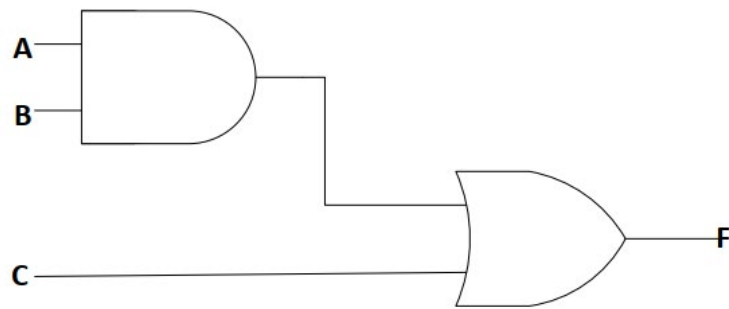
That is, either the light sensor detects darkness (true) and triggers the turning on of the light **OR** when the Motion Sensor detects motion (true) or both (true, true). Just like the OR Gate functions. Check this with the section on the OR gate above which explains that it lets the signal through **if at least one of its inputs is 1**.

3. By representing digital circuitry using Boolean algebra, you can analyse them and simplify complex expressions, leading to more efficient and cost-effective automation designs. Techniques like De Morgan's Laws and Boolean identities can help minimise the number of logic gates needed to implement the control logic.

## Combinational Logic Functions

The three basic logic functions AND, OR, and NOT, together with the other logic gates, can be combined in various forms to make more complex logic functions known as **Combination Logic Circuits**. These circuits may be used to implement functions such as comparison, arithmetic, code conversion, encoding, decoding, data selection, counting and storage. A digital system can, therefore, be seen as composed of an arrangement of these individual logic functions connected in such a way as to perform a specified operation or produce a defined output.

The output function of a combinational logic circuit can be determined simply by following the basic mode of operation of its comprising logic gates, as explained in the earlier sections. They may seem complex at first sight, however, with patience and adherence to the rules of operation of these logic gates, you will soon find the outputs for the various input combinations. Consider the combinational logic circuit in Figure4.8.



**Figure 4.8: An Example of a combinational logic circuit**

To derive the Boolean expression and truth table for a given combinational logic circuit, first generate all the possible input combinations that exist for the logic circuit. Remember from earlier in the section that for a  $x$  input logic circuit, it would have  $2^x$  possible combinations. Begin at the left-most inputs and work toward the final output, writing the expression for each gate. For the example circuit in **Figure 1.21**, the Boolean expression is determined in the following three steps:

1. The combinational circuit has 3 inputs (A, B and C).
2. The expression for the left-most AND gate with inputs A and B is  $AB$ .
3. The output of the left-most AND gate ( $AB$ ) is one of the inputs to the OR gate, and C is the other input.
4. Therefore, ORing  $AB$  with C the Boolean expression for the combinational circuit is  $AB + C$ , which is the final output expression for the entire circuit, written as  **$F = AB + C$** .

So, the truth table for a combinational circuit with three inputs, A and B and C, where A and B are ANDed, the output becomes an input of an OR gate along with the input C can be generated as follows.

A	B	C	AB	F=AB+C
0	0	0	0	0
0	0	1	0	1
0	1	0	0	0
0	1	1	0	1
1	0	0	0	1
1	0	1	0	1
1	1	0	1	1
1	1	1	1	1

## Explanation of truth table

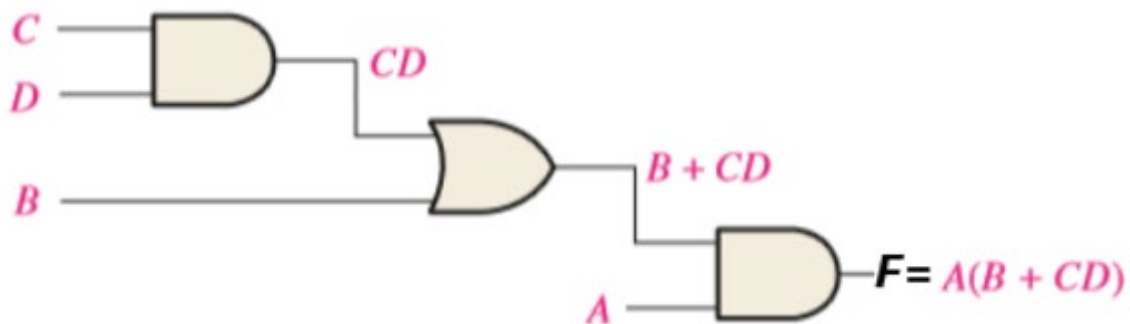
**Inputs:** The table lists all possible combinations of the inputs A, B, and C.

**AND Gate Output (AB) is column 3:** Shows the result of the AND operation between A and B.

**Final Output ((AB + C):** Shows the result of the OR operation between the output of the AND gate and the input C.

Note that it may not be necessary to show all the intermediary steps, **such as AB**, in the truth table; however, including them sometimes helps resolve the output function (F) faster.

The following example in **Figure 4.9** is a bit more complicated but you can see the Boolean outputs of each logic gate in red. To resolve this combinational logic circuit, follow the steps in the previous example.



**Figure 4.9: An Example of a combinational logic circuit**

1. The combinational circuit has 4 inputs (A, B, C and D); therefore, it would have 16 possible input combinations.
2. The expression for the left-most AND gate with inputs C and D is  $CD$ .
3. The output of the left-most AND gate ( $CD$ ) is one of the inputs to the OR gate along with another input B is the other input. Therefore, the expression for the OR gate is  $CD + B$ .
4. The output of the OR gate is one of the inputs to the right-most AND gate along with another input A. Therefore, the expression for this AND gate is  $A(B + CD)$ , which is the final output expression for the entire circuit, written as  $F = A(B + CD)$ .

Based on the mode of operation of the AND gates and OR gate, the truth table can be generated as follows.

A	B	C	D	CD	B + CD	F=A(B+CD)
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	0	1	0	0	0	0
0	0	1	1	1	1	0
0	1	0	0	0	1	0

0	1	0	1	0	1	0
0	1	1	0	0	1	0
0	1	1	1	1	1	0
1	0	0	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	0	0	0
1	0	1	1	1	1	1
1	1	0	0	0	1	1
1	1	0	1	0	1	1
1	1	1	0	0	1	1
1	1	1	1	1	1	1

## Examples of Boolean Expressions in Digital Circuits

### 1. Automatic Door Lock

**Scenario:** A door should only unlock if both a key is present (*KeySensor*), and a correct code is entered on the keypad (*CodeCorrect*).

#### Solution

**Boolean Expression:**  $\text{DoorUnlock} = \text{KeySensor} \cdot \text{CodeCorrect}$

This expression uses the AND operator, ensuring both conditions (key and code) are met for unlocking.

### 2. Security System Activation

**Scenario:** An alarm should sound (*AlarmOn*) if either a window sensor (*WindowSensor*) detects a break-in or a motion sensor (*MotionSensor*) detects movement inside the house when the system is armed (*SystemArmed*).

#### Solution

**Boolean Expression:**  $\text{AlarmOn} = \text{SystemArmed} \cdot (\text{WindowSensor} + \text{MotionSensor})$

The OR operator in the control box electronics is used to trigger the alarm if either sensor detects an issue while the system is armed.

### 3. Security System Activation

**Scenario:** An alarm should sound (*AlarmOn*) if either a window sensor (*WindowSensor*) detects a break-in or a motion sensor (*MotionSensor*) detects movement inside the house when the system is armed (*SystemArmed*).

#### Solution

**Boolean Expression:**  $\text{AlarmOn} = \text{SystemArmed} \cdot (\text{WindowSensor} + \text{MotionSensor})$

The OR operator in the control box electronics is used to trigger the alarm if either sensor detects an issue while the system is armed.

## Boolean Algebra Rules and Theorems

By simplifying Boolean expressions, you are making the digital circuitry, such as the robot’s brain, more efficient. It is like taking a long, complicated sentence and turning it into a short, easy-to-understand phrase. By doing this, robotics engineers can build robots that are faster, cheaper, and more reliable. To achieve this, it is important to understand some basic Boolean algebra rules and theorems. These theorems and properties of Boolean algebra help simplify complex Boolean expressions and design digital circuits efficiently. The sections that follow explain these theorems.

### Theorem 1: The Complement of a Complement

You have already learned that the NOT gate acts as an inverter in a binary system. Therefore, the NOT (1), or the complement of 1, yields 0. Conversely, the NOT (0), or the complement of 0, results in 1, as shown in the truth table below for a NOT Gate.

A	$\overline{A}$
0	1
1	0

From this table, Boolean Algebra establishes that the complement of a complemented value is the same as the original value uncomplemented. In short . This is demonstrated in the truth table below.

A	$\overline{A}$	$\overline{\overline{A}}$
0	1	0
1	0	1

## Theorem 2: Theorems Relating to the OR Gate

By observing the truth table of a 2 input OR gate, as shown in the table below, a number of theorems can be derived.

	A	B	$F=A+B$
row 1	0	0	0
row 2	0	1	1
row 3	1	0	1
row 4	1	1	1

From this OR Gate the following deductions can be made

$A + 0 = A$	As seen in row 1 and 3
$A + 1 = 1$	As seen in row 2 and 4
$A + A = A$	As seen in row 1 and 4
$A + \bar{A} = 1$	As seen in row 2 and 3

## Theorem 3: Theorems Relating to the AND Gate

By observing the truth table of a 2 input AND gate, as shown in the table below, a number of theorems can be derived.

row 2	0	1	0
row 3	1	0	0
row 4	1	1	1

	A	B	$F=A \cdot B$
row 1	0	0	0

From this AND Gate, the following deductions can be made

$A \cdot 0 = 0$	<i>As seen in row 1 and 3</i>
$A \cdot 1 = A$	<i>As seen in row 2 and 4</i>
$A \cdot A = A$	<i>As seen in row 1 and 4</i>
$A \cdot \bar{A} = 0$	<i>As seen in row 2 and 3</i>

## Theorem 4: Commutative Theorem

The commutative law of addition for two variables is written as:

$$A + B = B + A$$

This law states that the order in which the variables are **ORed** makes no difference. Remember, in Boolean algebra, just as applied to logic circuits, addition and OR operations are the same.

Also, the commutative law of multiplication for two variables is written as:

$$A \bullet B = B \bullet A$$

$$\text{Or } AB = BA$$

This law states that the order in which the variables are **ANDed** makes no difference. Remember, in Boolean algebra as applied to logic circuits, the dot ( $\bullet$ ) or the absence of an operator means the same as an AND operation.

## Theorem 5: Associative Theorem

The associative law of addition is written as follows for three variables:

$$A + (B + C) = (A + B) + C$$

This law states that when **ORing** more than two variables, the result is the same regardless of the grouping of the variables.

The associative law of multiplication is written as follows for three variables:

$$A(BC) = (AB)C$$

This law states that it makes no difference in what order the variables are grouped when **ANDing** more than two variables.

## Theorem 6: Distributive Theorem

The distributive law is written for three variables as follows:

$$A(B + C) = (AB) + (AC)$$



This law states that ORing two or more variables and then ANDing the result with a single variable is equivalent to ANDing the single variable with each of the two or more variables and then ORing the products. The distributive law also expresses the process of factoring in which the common variable  $A$  is factored out of the product terms, for example:

$$AB + AC = A(B + C)$$

The distributive law can also be written as follows:

$$A + (BC) = (A + B)(A + C)$$

This other expression states that ANDing two or more variables and then ORing the result with a single variable is equivalent to ORing the single variable with each of the two or more variables and then ANDing their additions. The distributive law also expresses the process of factoring in which the common variable  $A$  is factored out of the addition terms, for example:

$$(A+B)(A+C) = A+(BC)$$

Based on this distributive law, you can say that  $A + (\bar{A}B) = A + B$ . This can be proven below as:

$$\bar{A} + (AB) = (\bar{A} + A) \cdot (\bar{A} + B)$$

Remember that from the theorems relating to the OR Gate  $(A +) = 1$ ; therefore, the expression above can be resolved as:

$$= 1 \cdot (A + B)$$

Also, remember that from the theorems relating to the AND Gate  $(A \cdot 1) = A$ , therefore the expression above can be resolved as:

$$= A + B$$

Based on this distributive law, you can say that  $\bar{A} + (AB) = \bar{A} + B$ . This can be proven below as:

$$\bar{A} + (AB) = (\bar{A} + A) \cdot (\bar{A} + B)$$

Remember that from the theorems relating to the OR Gate  $(A + \bar{A}) = 1$ ; therefore, the expression above can be resolved as:

$$= 1 \cdot (\bar{A} + B)$$

Also, remember that from the theorems relating to the AND Gate  $(A \cdot 1) = A$ , therefore the expression above can be resolved as:

$$= \bar{A} + B$$

Finally, based on this theorem and the principle of factoring, consider the following logical expression:

$$A + AB$$

By factoring out the common factor  $A$ , you have the expression now as

$$A(1 + B)$$

From the theorems relating to the OR Gate, when any variable is ORed with 1, the result is 1. Therefore, the expression above can be simplified as:

$$A(1)$$

Also, from the theorems relating to the AND Gate, when any variable is ANDed with 1, the result is that variable. Therefore, the above expression can simply be stated as:

$$A$$

In other words,  $A + AB = A$ , as simple as that.

So, in short, the important equations to remember under the Distributive Law are summarised and presented in the table below.

1	$A(B + C) = (AB) + (AC)$
2	$A + (BC) = (A + B)(A + C)$
3	$A + (\bar{A}B) = A + B$
4	$\bar{A} + (AB) = \bar{A} + B$
5	$A + AB = A$

## Theorem 7: DeMorgan's Theorem

DeMorgan, a mathematician who knew Boole, proposed two theorems that are an important part of Boolean algebra. In practical terms, DeMorgan's theorems provide mathematical verification of the equivalency of the NAND and NOR gates.

DeMorgan's first theorem is stated as follows:

"The complement of a product of variables is equal to the sum of the complements of the variables."

Stated another way, "The complement of two or more ANDed variables is equivalent to the OR of the complements of the individual variables."

The formula for expressing this theorem for two variables is:

$$(\bar{A}\bar{B}) = \bar{A} + \bar{B}$$

DeMorgan's second theorem is like the first and is stated as follows:

"The complement of a sum of variables is equal to the product of the complements of the variables."

Stated another way: "The complement of two or more ORed variables is equivalent to the AND of the complements of the individual variables."

The formula for expressing this theorem for two variables is

$$(\overline{A + B}) = \overline{A} \cdot \overline{B}$$

**Figure 4.7** shows the gate equivalencies and truth tables for the two DeMorgan’s Theorems stated above.

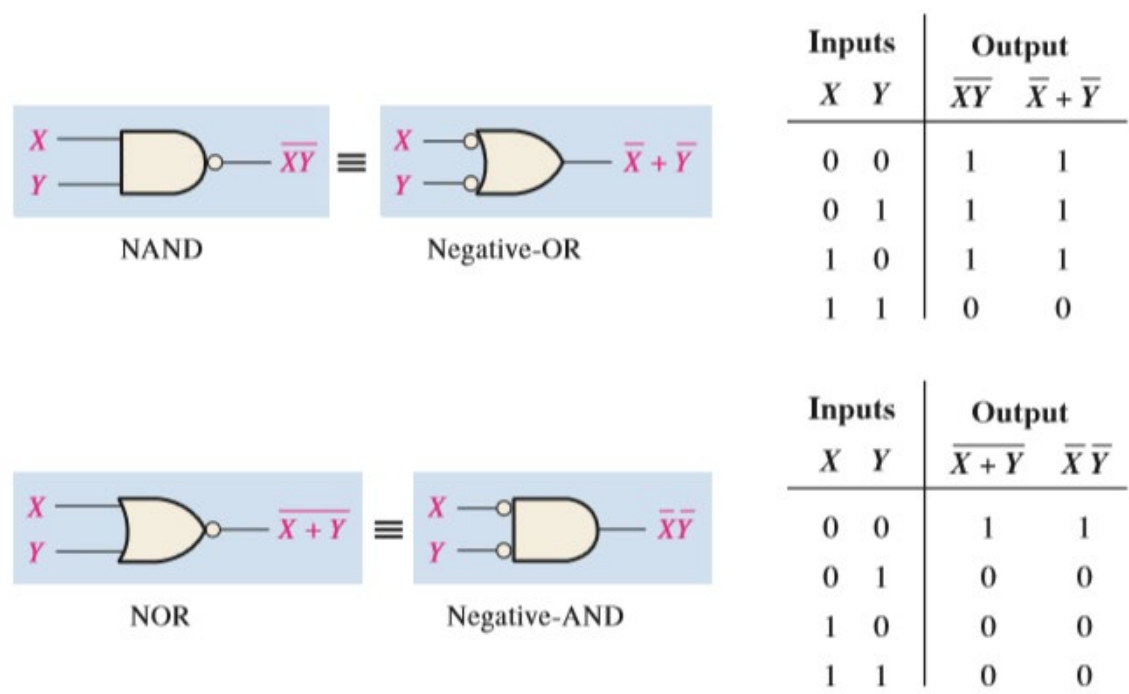


Figure 4.7: Gate equivalencies and the corresponding truth tables of DeMorgan

As stated earlier, DeMorgan’s theorems also apply to expressions in which there are more than two variables. The table below illustrates the application of DeMorgan’s theorems to 3-variable and 4-variable expressions.

	3-4 Variables' Equation	Equation's Equivalent
1	$F = \overline{ABC}$	$F = \overline{A} + \overline{B} + \overline{C}$
2	$F = \overline{A + B + C}$	$F = \overline{A} \cdot \overline{B} \cdot \overline{C}$
3	$F = \overline{ABCD}$	$F = \overline{A} + \overline{B} + \overline{C} + \overline{D}$
4	$F = \overline{A + B + C + D}$	$F = \overline{A} \cdot \overline{B} \cdot \overline{C} \cdot \overline{D}$
5	$F = \overline{\overline{A} + \overline{B} + \overline{C}}$	$F = \overline{\overline{A} \cdot \overline{B} \cdot \overline{C}} = A \cdot B \cdot C$
6	$F = \overline{\overline{ABC}}$	$F = \overline{\overline{A} + \overline{B} + \overline{C}} = A + B + C$

When working on complex logical expressions with more than two variables that require the application of DeMorgan’s Theorems, it is important to simplify the expression first. In simplifying, you can represent one set of complemented variables as a single variable, such

as X and the others as Y or any other variable. To demonstrate work through this complex examples:

This expression can be first simplified if the term is represented by X and by Y, thus making the expression simply as: which, according to DeMorgan, is equivalent to

So, by replacing the X and Y with what they stand for, you would get the following:

Notice that in the preceding result, you have two terms to each of which you can again apply DeMorgan's theorem individually as follows:

Notice that you still have two terms in the expression to which DeMorgan's theorem can again be applied. These terms are . A final application of DeMorgan's theorem gives the following result:

Although this result can be simplified further by the use of some of the Boolean Algebra rules you examined earlier, DeMorgan's theorems cannot be used any further.

## Summarisation of Boolean Algebra Theorems

All the laws/theorems you have examined up to this point are summarised in the Table below. It is important to keep them at your fingertips because their main purpose is to minimise Boolean expressions.

	Theorem	Equations
1	The Complement of a Complement	$\overline{\overline{A}} = A$
2	Theorems Relating to the OR Gate	$A + 0 = A$
		$A + 1 = 1$
		$A + A = A$
		$A + \overline{A} = 1$
3	Theorems Relating to the AND Gate	$A \cdot 0 = 0$
		$A \cdot 1 = A$
		$A \cdot A = A$
		$A \cdot \overline{A} = 0$
4	Commutative Theorem	$A + B = B + A$
		$A \cdot B = B \cdot A$
5	Associative Theorem	$A + (B + C) = (A + B) + C$
		$A(BC) = (AB)C$
6	Distributive Theorem	$A(B + C) = (AB) + (AC)$
		$A + (BC) = (A + B)(A + C)$
		$A + (\overline{A}B) = A + B$

7	DeMorgan's Theorems	$\overline{A} + (AB) = \overline{A} + B$
		$A + AB = A$
		$\overline{AB} = \overline{A} + \overline{B}$
		$\overline{A + B} = \overline{A} \cdot \overline{B}$

Now that you have been introduced to the basics of Boolean algebra, your challenge is to try to simplify some complex expressions. But remember, when you simplify, you follow a specific order:

1. **NOT operations** come first. This includes things like De Morgan's Law.
2. **AND operations** are next.
3. **OR operations** are the last.

By following this order, you can make sure your simplifications are correct. Look at some examples below to see how the simplification process works.

## Step-by-Step Simplification Process

To demonstrate the simplification process of complex Boolean expressions, examine the following example  $F = A \cdot B + A \cdot \overline{B} + \overline{A} \cdot B$ . This uses the laws in the table above.

**Example 1:**  $F = A \cdot B + A \cdot \overline{B} + \overline{A} \cdot B$ . The following laws are applied to simplify.

1. **Apply Distributive Law:** Group common terms to apply the distributive law:  

$$F = A \cdot (B + \overline{B}) + \overline{A} \cdot B$$
2. **Simplify Using theorems relating to the OR Gate:** According to the theorems relating to the OR Gate,  $B + \overline{B} = 1$ , hence substitute into the expression to further simplify it:  

$$F = A \cdot 1 + \overline{A} \cdot B$$
3. **Apply theorems relating to the AND Gate:** according to the identity law,  $A \cdot 1 = A$ , hence substitute into the expression to further simplify it:  

$$F = A + \overline{A} \cdot B$$
4. **Apply Distributive and Complement:** to further simplify the expression, you must expand it using the Distributive law.  
Hence  $A + \overline{A} \cdot B$  becomes  $(A + \overline{A}) \cdot (A + B)$   
Since,  $A + \overline{A} = 1$  according to the theorems relating to the OR Gate, you can substitute into the expression to further simplify it:  

$$F = 1 \cdot (A + B)$$
5. **Apply Identity law again:** by applying identity law to the expression again,  $1 \cdot (A + B) = A + B$ .  
Thus, the simplified form of  $F = A \cdot B + A \cdot \overline{B} + \overline{A} \cdot B$  is  $F = (A + B)$

**Example 2:**  $AB + A(B + C) + B(B + C)$

1. This expression has no complement (NOT) of terms; therefore, you must start with the application of the distributive law to the second and third terms in the expression, which involve AND

$$AB + AB + AC + BB + BC$$

2. Apply theorems relating to AND Gate ( $BB = B$ ) to the fourth term, as follows:

$$AB + AB + AC + B + BC$$

3. Apply theorems relating to OR Gate ( $AB + AB = AB$ ) to the first two terms as follows:

$$AB + AC + B + BC$$

4. Apply Distributive Law ( $B + BC = B$ ) to the last two terms as follows:

$$AB + AC + B$$

5. Apply the Associative Law to restate the expression as follows:

$$B + AB + AC$$

6. Apply Distributive Law ( $B + AB = B$ ) to the first and second terms as follows

$$B + AC$$

At this point, the expression is simplified as much as possible. Once you gain experience in applying Boolean algebra, you can often combine many individual steps without stating them explicitly like this example did.

**Example 3:**  $AB + A\bar{B}$

**Solution:**

$$AB + A\bar{B}$$

apply the distributive law to the terms in the expression to factor out A, as follows

$$A(B + \bar{B})$$

apply the theorems relating to the OR Gate ( $B + \bar{B} = 1$ ) as follows

$$A(1)$$

apply the theorems relating to the AND Gate  $A \cdot 1 = A$  as follows

$$A$$

At this point, the expression is simplified as much as possible. You can demonstrate that these two expressions are equivalent using the truth table below.

A	B	F = $AB + A\bar{B}$	A
0	0	0	0
0	1	0	0
1	0	1	1
1	1	1	1

↑ equal ↑

**Example 4:**  $AB + A\bar{B}C + A\bar{B}\bar{C}$

**Solution:**

$$AB + A\bar{B}C + A\bar{B}\bar{C}$$

apply the distributive law by factoring out the common factor of A as follows:

$$A(B + \bar{B}C + \bar{B}\bar{C})$$

apply the distributive law on the terms  $(B + \bar{B}C) = B + C$  as follows:

$$A(B + C + \bar{B}\bar{C})$$

apply the distributive law on the terms  $(B + \bar{B}\bar{C}) = B + \bar{C}$  as follows:

$$A(B + C + \bar{C})$$

apply the theorems relating to the OR Gate  $(C + \bar{C}) = 1$  as follows:

$$A(B + 1)$$

apply the theorems relating to the OR Gate  $(B + 1) = 1$  as follows:

$$A(1)$$

apply the theorems relating to the AND Gate  $(A \cdot 1) = A$  as follows:

$$A$$

### Activity 4.1 Simplifying Complex Boolean Expressions

*Organise yourselves into groups of 3-5 for this activity.*

Sit and work in your groups to analyse at least one of the following scenarios and develop Boolean logic systems for automated decision-making. Follow the steps below to complete your design, formulation, and simplification for each system scenario provided.

#### Scenario 1: Automated Home Control System

In your Design a Boolean logic system to control a home's alarm and lighting functions based on specific conditions:

##### 1. Alarm Control

The alarm should activate if any of the doors (D1, D2) or windows (W1, W2) are open when the system is armed (A).

##### 2. Lighting Control

The lights should turn on if it is dark outside (DarkSensor) and motion is detected inside the house (MotionSensor).

#### Task Requirements:

##### 1. Boolean Expression Formulation



- a. Write the Boolean expression for the alarm system that meets the above conditions.
- b. Write the Boolean expression for the lighting system.

## 2. Truth Table Creation

Create a truth table for each Boolean expression. Include all possible combinations of inputs and their corresponding outputs.

## 3. Simplification of Boolean Expressions

Using Boolean algebra rules and theorems, simplify each Boolean expression as much as possible.

### Scenario 2: Digital Voting System for Robotics Competition

Design a Boolean logic system to determine the winner in a robotics competition based on votes from three judges (J1, J2, J3).

The system should declare a candidate the winner if at least two out of three judges vote in favour of the candidate.

#### Task Requirements:

### 1. Boolean Expression Formulation

Write the Boolean expression that determines if a candidate wins based on the votes from the three judges.

### 2. Truth Table Creation

Create a truth table for the Boolean expression. Include all possible combinations of votes and the output that indicates a win or loss.

### 3. Simplification of Boolean Expressions

Simplify the Boolean expression using Boolean algebra rules and theorems.

#### Documentation

For each scenario, create a detailed report that includes:

1. The initial Boolean expression
2. The truth table for all input combinations
3. The final simplified Boolean expression

#### Group Presentation

After completing your selected scenario(s), present your findings and explain each step of the Boolean logic process to the class. Discuss any challenges you encountered and explain how you resolved them.

## ADVANCED DIGITAL SYSTEM OPTIMISATION WITH KARNAUGH MAPS

In this lesson, you will begin by strengthening your understanding of two key forms for expressing Boolean expressions: The Sum-of-Products (SOP) and the Product-of-Sums

(POS). Knowing these standard forms is essential because they serve as a foundation for simplifying Boolean expressions systematically, especially when using more advanced tools. SOP and POS forms allow for clear organisation of logic terms, making it easier to analyse, reduce, and eventually optimise digital circuits.

With this foundation, you will then apply Karnaugh Maps (K-Maps), a visual method to simplify Boolean expressions even further. K-Maps are valuable for identifying and eliminating redundant steps in complex logic, which helps make digital circuits more efficient and cost-effective by minimising unnecessary components. Through hands-on practice, you will learn to group adjacent ones or zeros on the K-Map, reducing the number of logic gates required. This lesson will improve your problem-solving skills and prepare them to optimise digital systems effectively, building on your knowledge of Boolean Algebra and logic gates.

## Standard Forms of Boolean Expressions

All Boolean expressions, regardless of their form, can be converted into either of two standard (or canonical) forms:

1. the sum-of-products (SOP) form or
2. the product-of-sums (POS) form

Standardisation makes the evaluation, simplification, and implementation of Boolean expressions much more systematic and easier.

### The Sum-of-Products (SOP) Form

In presenting a Logical expression in the SOP form, you must **consider the various input combinations** which have a **functional output value of 1 or TRUE**. The individual input variables are then ANDed (or multiplied), and finally, all the ANDed Combinations are ORed (or Summed), hence the name Sum-of-Products.

Consider the truth table below having 3 inputs, A, B and C and an output function F

	A	B	C	F
Row0	0	0	0	0
Row1	0	0	1	0
Row2	0	1	0	0
Row3	0	1	1	1
Row4	1	0	0	0
Row5	1	0	1	1
Row6	1	1	0	0
Row7	1	1	1	1

From the truth table, we realise that the output function  $F$  is True for the input combinations at Row3, Row5 and Row7. In generating the Standard SOP form, you must express the input variable  $A$ :

1. as when  $A$  is TRUE or 1 and
2. as when  $A$  is FALSE or 0

We would then find the product of the input variables where the output Function is True or 1. So, you would have the following ANDed terms to consider:

1. Row3:  $ABC$
2. Row5:  $ABC$  and
3. Row7:  $ABC$

These terms would then be ORed (summed) together to give you the Standard/Canonical SOP form, as seen in the expression below:

$$F = ABC + ABC + ABC$$

This expression can simply be represented or implemented using a combinational circuit using simple AND and OR gates known as an AND-OR logic.

Let us take another example. Consider the truth table below having 3 inputs,  $A$ ,  $B$  and  $C$  and an output function  $F$ .

	<b>A</b>	<b>B</b>	<b>C</b>	<b>F</b>
<b>Row0</b>	0	0	0	0
<b>Row1</b>	0	0	1	0
<b>Row2</b>	0	1	0	1
<b>Row3</b>	0	1	1	1
<b>Row4</b>	1	0	0	0
<b>Row5</b>	1	0	1	1
<b>Row6</b>	1	1	0	1
<b>Row7</b>	1	1	1	1

From the truth table, we realise that the output function  $F$  is True for the input combinations at Row2, Row3, Row5, Row6 and Row7. Now we would have the following ANDed terms to produce the following terms to consider:

1. Row2:  $ABC$
2. Row3:  $ABC$
3. Row5:  $ABC$
4. Row6:  $ABC$  and
5. Row7:  $ABC$

These terms would then be ORed (summed) together to give us the Standard/Canonical SOP form, as seen in the expression below:

$$F = ABC + ABC + ABC + ABC + ABC$$

### The Product-of-Sums (POS) Form

In presenting a Logical expression in the POS form, you should consider only we are interested in the various input combinations which have a functional **output value of 0 or FALSE**. The individual input variables are then ORed (or summed), and finally, all the ORed Combinations are ANDed (or multiplied/product), hence the name Product-of-Sums. So, this is like the reverse of the standard SOP form.

Consider the truth table below having 3 inputs, A, B and C and an output function F.

	A	B	C	F
Row0	0	0	0	0
Row1	0	0	1	0
Row2	0	1	0	1
Row3	0	1	1	1
Row4	1	0	0	0
Row5	1	0	1	1
Row6	1	1	0	1
Row7	1	1	1	1

From the truth table, you can see we realise that the output function F is False for the input combinations at Row0, Row1 and Row4. In generating the Standard POS form, you we express the input variable A:

1. as when A is FALSE or 0 and
2. as when A is TRUE or 1

You must We now would then OR (or find the sum of) the input variables where the output Function is False or 0. So, you we would have the following ORed terms to consider:

1. Row0: A+B+C
2. Row1: A+B+C and
3. Row4: A+B+C

These terms would then be ANDed (multiplied) together to give youus the Standard/Canonical POS form, as seen in the expression below.

$$F = (A+B+C)(A+B+C)(A+B+C)$$

Implementing a POS expression using a combinational logic circuit simply requires ANDing the outputs of two or more OR gates. A summed term is produced by an OR operation, and the product (multiplication) of two or more sums terms is produced by an AND operation. Therefore, a POS expression can be implemented by OR-AND logic

Here is Let us take another example to find the Product of Sums. Consider the truth table below having 3 inputs, A, B and C and an output function F.

	A	B	C	F
Row0	0	0	0	0
Row1	0	0	1	0
Row2	0	1	0	0
Row3	0	1	1	1
Row4	1	0	0	0
Row5	1	0	1	1
Row6	1	1	0	0
Row7	1	1	1	1

From the truth table, you can see we realise that the output function F is False for the input combinations at Row0, Row1, Row2, Row4 and Row6. Now We would then OR (or find the sum of) the input variables where the output Function is False or 0. This gives 0. So, we would have the following ORed terms to consider.

1. Row0:  $A+B+C$
2. Row1:  $A+B+C$
3. Row2:  $A+B+C$
4. Row4:  $A+B+C$  and
5. Row6:  $A+B+C$

These terms would then be ANDed (multiplied) together to give youus the Standard/Canonical POS form, as seen in the expression below:

$$F = (A+B+C)(A+B+C)(A+B+C)(A+B+C)(A+B+C)$$

Note that the Standard/Canonical SOP can be used to generate the Standard/Canonical POS form by simply complementing it and applying DeMorgan's Theorem.

Having understood these standard forms, you we can progress to apply this knowledge in minimising Boolean expressions using K-Maps.

## The Karnaugh Map (K-Map)

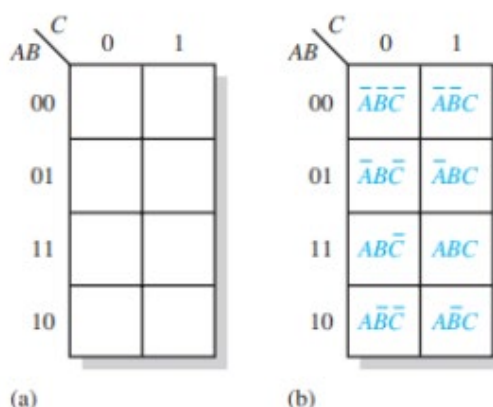
A Karnaugh map offers a structured **approach for simplifying Boolean expressions**, serving as a visual tool that, when used correctly, yields the simplest Sum-of-Products (SOP) or Product-of-Sums (POS) expression—known as the minimum expression. Previously, you learned we saw that simplifying Boolean expressions relied heavily on understanding and accurately applying Boolean theorems. In contrast, the Karnaugh map acts as a “cookbook” method, allowing for straightforward simplification without needing to manually apply each theorem.

**A Karnaugh map is similar to a truth table** in that it shows all possible values of input variables and the resulting output for each. However, instead of using columns and rows like a truth table, a Karnaugh map is organised into an array of cells, where each cell represents a binary combination of input variables. The cells are arranged so that simplifying an expression becomes a straightforward process of grouping adjacent cells. Although Karnaugh maps can handle expressions with two, three, four, or even five variables, you will **focus on 3-variable and 4-variable cases** to illustrate the key principles.

**The number of cells in a Karnaugh map, as well as the number of rows in a truth table, is equal to the total number of possible input variable combinations (output).** So, for three variables, the number of cells is  $2^3 = 8$ . For four variables, the number of cells is  $2^4 = 16$ .

## The 3-Variable Karnaugh Map

The 3-variable Karnaugh map is an array of eight cells, as shown in **Figure 4.8(a)**. In this case, A, B, and C are used for the variables although other letters could be used. Binary values of A and B are down along the left side (notice the sequence) and the values of C are across the top. The value of a given cell is the binary values of A and B at the left in the same row combined with the value of C at the top in the same column. For example, the cell in the upper left corner has a binary value of 000 and the cell in the lower right corner has a binary value of 101. **Figure 4.8(b)** shows the standard product terms that are represented by each cell in the Karnaugh map



**Figure 4.8:** A 3-variable Karnaugh map showing Boolean product terms for each cell

## The 4-Variable Karnaugh Map

The 4-variable Karnaugh map is an array of sixteen cells, as shown in **Figure 4.9(a)**. Binary values of A and B are downalong the left side, and the values of C and D are across the top. The value of a given cell is the binary values of A and B at the left in the same row combined with the binary values of C and D at the top in the same column. For example, the cell in the **upper right corner has a binary value of 001,0** and the cell in the **lower right corner has a binary value of 1010**. **Figure 4.9(b)** shows the standard product terms that are represented by each cell in the 4-variable Karnaugh map.

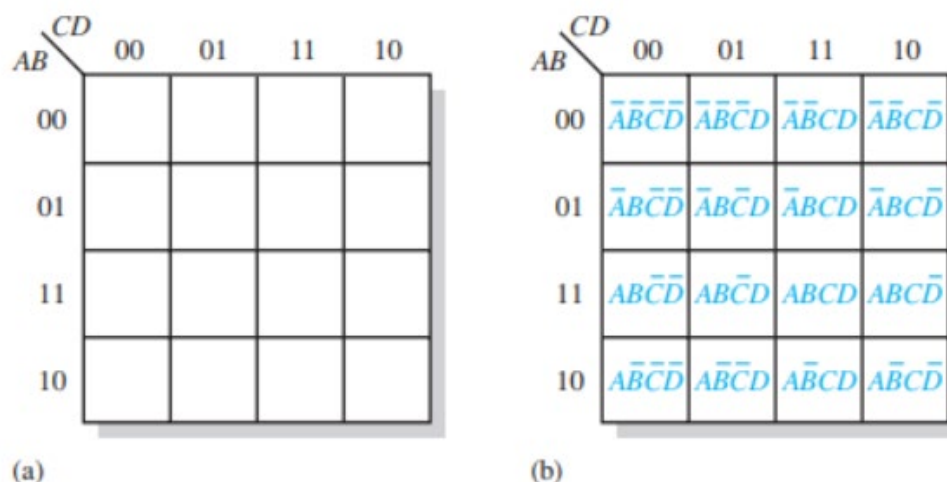


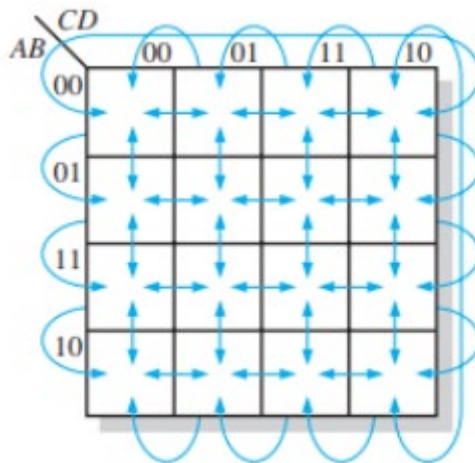
Figure. 4.9: A 4-variable Karnaugh map showing Boolean product terms for each cell

## Cell Adjacency

The cells in a Karnaugh map are arranged so that there is only a single-variable change between adjacent cells. In a 3-variable map, the 010 cell is adjacent to the 000 cell, the 011 cell, and the 110 cell. The 010 cell is not adjacent to the 001 cell, the 111 cell, the 100 cell, or the 101 cell. The cells in a Karnaugh map are arranged so that there is only a **single-variable change between adjacent cells**. Adjacency is defined by a single-variable change. In the 3-variable map the 010 cell is adjacent to the 000 cell, the 011 cell, and the 110 cell. The 010 cell is not adjacent to the 001 cell, the 111 cell, the 100 cell, or the 101 cell.

Physically, each cell is adjacent to the cells that are immediately next to it on any of its four sides. A cell is **not adjacent** to the cells that **diagonally** touch any of its corners. Also, the cells in the top row are adjacent to the corresponding cells in the bottom row and the cells in the outer left column are adjacent to the corresponding cells in the outer right column. This is called “**wrap-around**” adjacency because you can think of the map as wrapping around from top to bottom to form a cylinder or from left to right to form a cylinder. **Figure 4.10** illustrates the cell adjacencies with a 4-variable map, although the same rules for adjacency apply to Karnaugh maps with any number of cells.





**Figure 4.10:** Adjacent cells on a Karnaugh map (arrows point between adjacent cells)

## Karnaugh Map SOP Minimisation

As stated in the last section, the Karnaugh map is used for simplifying Boolean expressions to their minimum form. A minimised SOP expression contains the fewest possible terms with the fewest possible variables per term. Generally, a minimum SOP expression can be implemented with fewer logic gates than a standard expression. In this section, Karnaugh maps with up to four variables are covered.

## Mapping a Standard SOP Expression

*Consider three input variables A, B and C.*

These produce an SOP expression:

$$ABC + ABC + ABC + ABC$$

*Now determine the binary, or equivalent value, of each product term in the expression. For example, ABC would be 000, ABC, 001, and so on.*

*Construct a three variable Karnaugh map.*

For an SOP expression in standard form, a 1 is placed on the Karnaugh map for each product term in the expression. Each 1 is placed in a cell corresponding to the **BINARY** value of a product term. In thisFor example, for the product term ABC, a 1 goes in the 11001 cell on **Figure 4.11**, a 3-variable map.

When an SOP expression is completely mapped, there will be a number of 1s on the Karnaugh map equal to the number of product terms in the standard SOP expression. No need to worry about theThe cells that do not have a 1 because they are the cells for which the value of the product term expression is 0. Usually, when working with SOP expressions, the 0s are left off the map.

The following steps and the illustration in **Figure 4.11** show the mapping process for the all the product terms in the SOP expression.

**Step 1:** Determine the **binary** value of each product term in the standard SOP expression. After some practice, you can usually do the evaluation of terms mentally.

**Step 2:** As each product term is evaluated, place a 1 on the Karnaugh map in the cell having the same value as the product term.

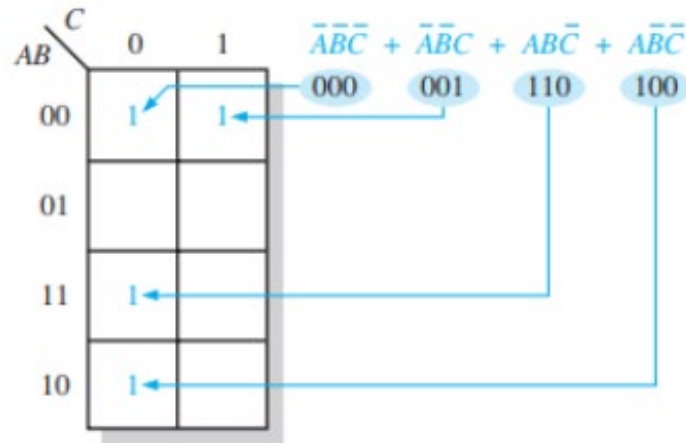


Figure 4.11: Example of mapping a standard SOP expression on a K-Map

Here is Let's consider another example

**Example:** Map the following standard SOP expression on a Karnaugh map:

$$ABC + \bar{A}BC + A\bar{B}C + \bar{A}\bar{B}C$$

**Solution:**

Evaluate the expression as shown below. Place a 1 on the 3-variable Karnaugh map as illustrated in Figure 10.5 for each standard product term in the expression.

$$ABC + \bar{A}BC + A\bar{B}C + \bar{A}\bar{B}C$$

$$001 \quad 010 \quad 110 \quad 111$$

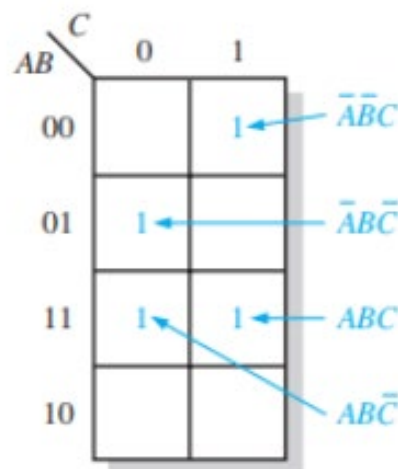


Figure 4.12: Example of mapping a standard SOP expression on a K-Map

**Example:** Map the following standard SOP expression on a four variable Karnaugh map

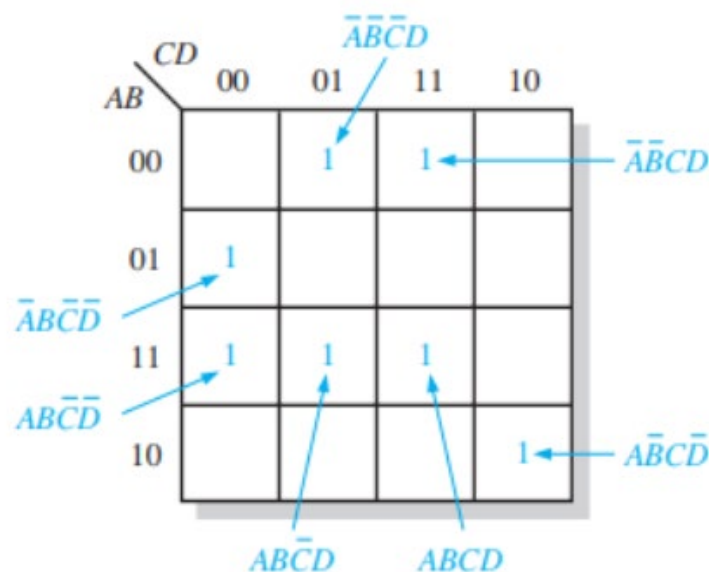
$$\bar{A}\bar{B}CD + \bar{A}B\bar{C}\bar{D} + AB\bar{C}D + ABCD + AB\bar{C}\bar{D} + \bar{A}\bar{B}\bar{C}D + \bar{A}\bar{B}C\bar{D}$$

**Solution:**

Evaluate the expression as shown below. Place a 1 on the 4-variable Karnaugh map as illustrated in Figure 10.6 for each standard product term in the expression.

$$\bar{A}\bar{B}CD + \bar{A}B\bar{C}\bar{D} + AB\bar{C}D + ABCD + AB\bar{C}\bar{D} + \bar{A}\bar{B}\bar{C}D + \bar{A}\bar{B}C\bar{D}$$

0 0 1 1    0 1 0 0    1 1 0 1    1 1 1 1    1 1 0 0    0 0 0 1    1 0 1 0



**Figure 4.13:** Example of mapping a standard SOP expression on a K-Map

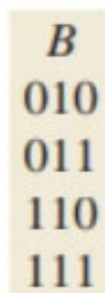
A Boolean expression must first be in standard form before you use a Karnaugh map. If an expression is not in standard form, then it must be converted to standard form by the procedure used to convert a minimal SOP expression into a standard SOP form or by numerical expansion. Since an expression should be evaluated before mapping anyway, numerical expansion is probably the most efficient approach.

Recall that a nonstandard product term has one or more missing variables. For example, assume that one of the product terms in a certain 3-variable SOP expression is  $AB$ . This term can be expanded numerically to standard form as follows.

1. First, write the binary value of the two variables and attach a 0 for the missing variable C: 100.
2. Next, write the binary value of the two variables and attach a 1 for the missing variable C: 101.
3. The two resulting binary numbers are the values of the standard SOP terms  $ABC$  and  $AB\bar{C}$ .

As another example, assume that one of the product terms in a 3-variable expression is B (remember that a single variable counts as a product term in an SOP expression). This term can be expanded numerically to standard form as follows:

- a. Write the binary value of the variable; then attach all possible values for the missing variables A and C as shown in Figure 4.14



<i>B</i>
010
011
110
111

**Figure 4.14: Numerical Expansion of B in a 3-variable expression**

- b. The four resulting binary numbers are the values of the standard SOP terms, , , and

## Karnaugh Map Simplification of SOP Expressions (minterms)

The process that results in an expression containing the fewest possible terms with the fewest possible variables is called **minimization**. After an SOP expression has been mapped, a minimum/minimal SOP expression is obtained by grouping the 1s and determining the minimum SOP expression from the map.

### Grouping the 1s

You can group 1s on the Karnaugh map according to the following rules by enclosing those adjacent cells containing 1s. The goal is to maximise the size of the groups and to minimise the number of groups.

1. A group must contain either 1, 2, 4, 8, or 16 cells, which are all powers of two. In the case of a 3-variable map,  $2^3 = 8$  cells is the maximum group.
2. Each cell in a group must be adjacent to one or more cells in that same group, but all cells in the group do not have to be adjacent to each other.
3. Always include the largest possible number of 1s in a group in accordance with rule 1.
4. Each 1 on the map must be included in at least one group. The 1s already in a group can be included in another group as long as the overlapping groups include noncommon 1s.

For example, by following the rules above, the 1s in each of the Karnaugh maps in **Figure 4.15** below are grouped as illustrated in **Figure 4.16**.

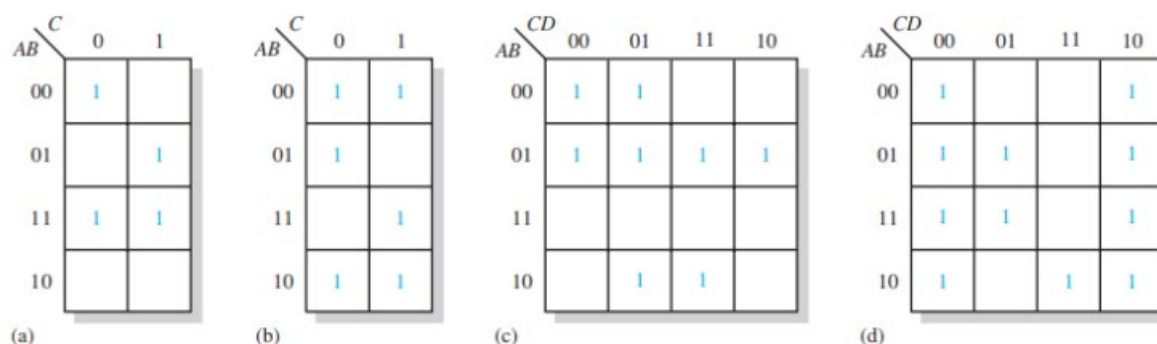


Figure 4.15: 1s in various K-Maps

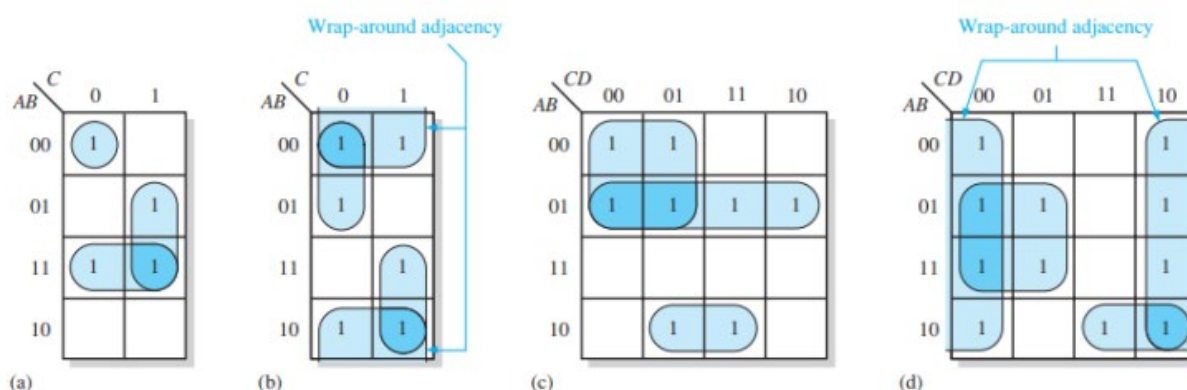


Figure 4.16: Grouped 1s in various K-Maps showing Adjacency

## Determining the Minimum SOP Expression from the K-Map

When all the 1s representing the standard product terms in an expression are properly mapped and grouped, the process of determining the resulting minimum SOP expression begins. The following rules are applied to find the minimum product terms and the minimum SOP expression:

1. Group the cells that have 1s. Each group of cells containing 1s creates one product term composed of all variables that occur in only one form (either uncomplemented or complemented) within the group. Variables that occur both uncomplemented and complemented within the group are eliminated. These are called contradictory variables.
2. Determine the minimum product term for each group.
  - a. For a 3-variable map:
    - A 1-cell group yields a 3-variable product term
    - A 2-cell group yields a 2-variable product term
    - A 4-cell group yields a 1-variable term
    - An 8-cell group yields a value of 1 for the expression

b. For a 4-variable map:

A 1-cell group yields a 4-variable product term

A 2-cell group yields a 3-variable product term

A 4-cell group yields a 2-variable product term

An 8-cell group yields a 1-variable term

A 16-cell group yields a value of 1 for the expression

3. When all the minimum product terms are derived from the Karnaugh map, they are summed to form the minimum SOP expression.

For example, for the K-Map in **Figure 4.17**, by following the rules above, the resulting minimum SOP expression is  $B + AC + ACD$

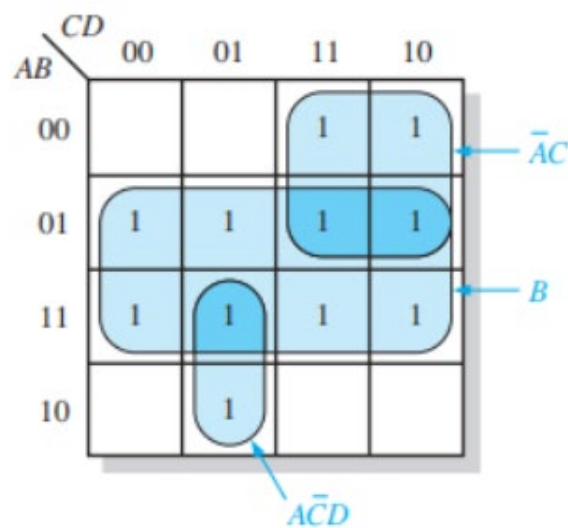


Figure 4.17: Grouped 1s in a K-Map showing Adjacency and product terms

Consider the K-Maps depicted in **Figure 4.18**:

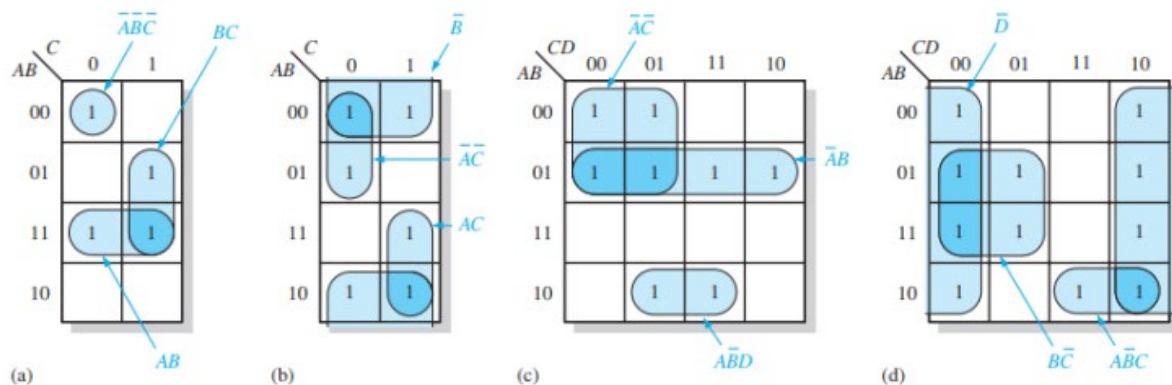


Figure 4.18: Grouped 1s in various K-Maps showing Adjacency and product terms

The resulting minimum SOP expressions are

$$(a) AB + BC + \bar{A}\bar{B}\bar{C}$$

$$(b) \bar{B} + \bar{A}\bar{C} + AC$$

$$(c) \bar{A}B + \bar{A}\bar{C} + A\bar{B}D$$

$$(d) \bar{D} + A\bar{B}C + B\bar{C}$$

## Mapping Directly from a Truth Table

Remember that earlier youon we represented **each combinational logic circuit function** with a **truth table**. After generating the truth table from a logic diagram, you can determine the minimal SOP form by drawing a Karnaugh map and placing 11's in the map for input combination that resulted in a 1.

An example of a Boolean expression and its truth table representation is shown in **Figure 4.19**. Notice in the truth table that the output X is 1 for four different input variable combinations. The 1Is in the output column of the truth table are mapped directly onto a Karnaugh map into the cells corresponding to the values of the associated input variable combinations, as shown in **Figure 4.19**. In the figure you can see that the Boolean expression, the truth table, and the Karnaugh map are simply different ways to represent a logic function.

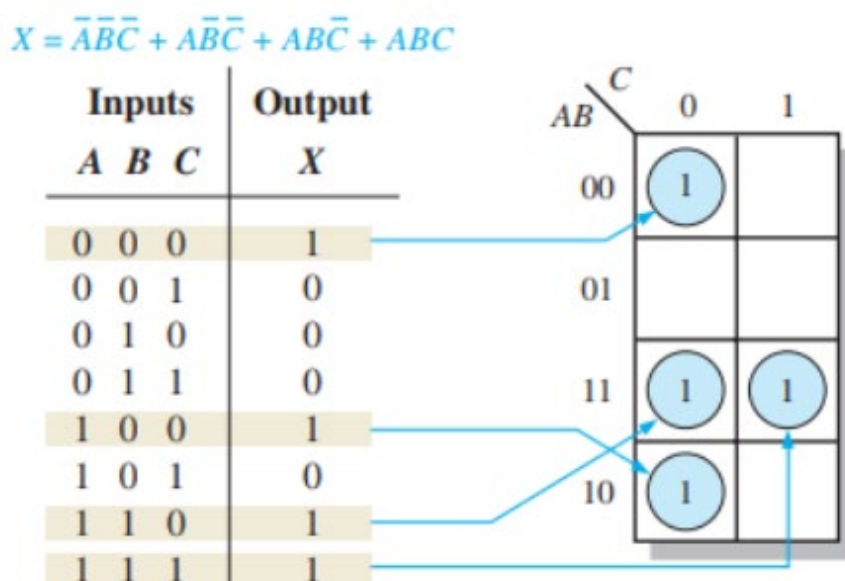


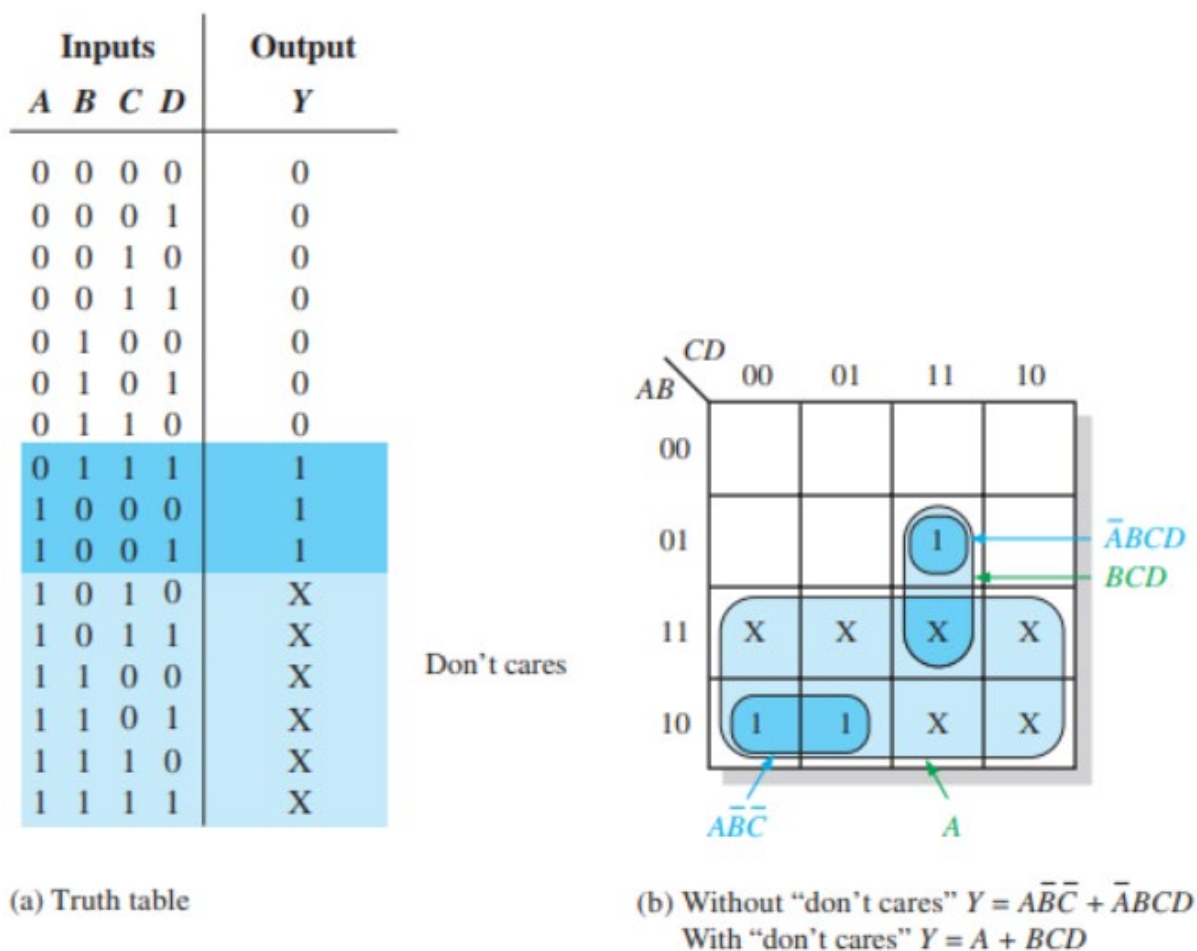
Figure. 4.19: Example of mapping directly from a truth table to a Karnaugh map.

## “Don’t Care” Conditions

Sometimes a situation arises in which some input variable combinations are not allowed; in order words youwe **don’t care** what these input variable combinations would result in. That is, for these “don’t care” terms either a 1 or a 0 may be assigned to the output; it really does not matter since they will never occur.



The “don’t care” terms can be used to advantage on the Karnaugh map. **Figure 4.20** shows that for each “don’t care” term, an X is placed in the cell. When grouping the 1Is, the Xs can be treated as 1Is to make a larger grouping or as 0s if they cannot be used to advantage. The larger a group, the simpler the resulting term will be.



**Figure 4.20:** Example of the use of “don’t care” conditions to simplify an expression.

If the “don’t cares” are used as 1Is, the resulting expression for the function is  $A + BCD$ , as indicated in part (b). If the “don’t cares” are not used as 1s, the resulting expression is  $\bar{A}\bar{B}\bar{C} + \bar{A}BCD$ ; so, you can see the advantage of using “don’t care” terms to get the simplest expression.

## Karnaugh Map POS Minimisation

In the last section, you studied the minimisation of an SOP expression using a Karnaugh map. In this section, we focus on POS expressions. The approaches are much the same except that with POS expressions, 0s representing the standard sum terms are placed on the Karnaugh map instead of 1s. The process is similar to that for SOP expressions, so the section below only covers a result of the similarity, we would only highlight the essential points.

## Mapping a Standard POS Expression

For a POS expression in standard form, a 0 is placed on the Karnaugh map for each sum term in the expression. Each 0 is placed in a cell corresponding to the value of a sum term. For example, for the sum term  $A+B+C$ , a 0 goes in the 010 cell on a 3-variable map.

When a POS expression is completely mapped, there will be a number of 0s on the Karnaugh map equal to the number of sum terms in the standard POS expression. The cells that do not have a 0 are the cells for which the expression is 1. Usually, when working with POS expressions, the 1s are left off. The following steps and the illustration in **Figure 4.21** show the mapping process.

**Step 1:** Determine the binary value of each sum term in the standard POS expression. This is the binary value that makes the term equal to 0.

**Step 2:** As each sum term is evaluated, place a 0 on the Karnaugh map in the corresponding cell

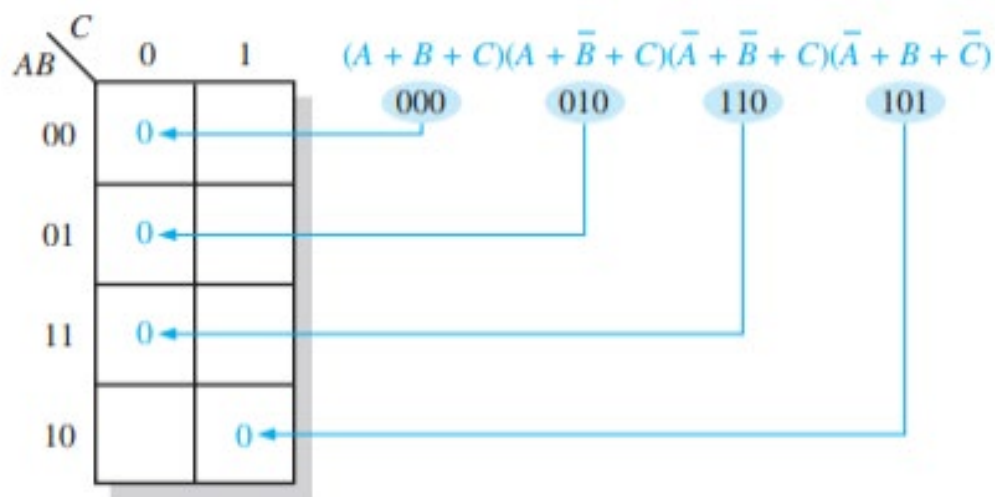


Figure 4.21: Example of mapping a standard POS expression.

## Karnaugh Map Simplification of POS Expressions

The process for minimising a POS expression is basically the same as for an SOP expression except that you group 0s to produce minimum sum terms instead of grouping 1s to produce minimum product terms. The rules for grouping the 0s are the same as those for grouping the 1s that you learned earlier.

**Example:** Use a Karnaugh map to minimise the following POS expression

$$(B + C + D)(A + B + \bar{C} + D)(\bar{A} + B + C + \bar{D})(A + \bar{B} + C + D)(\bar{A} + \bar{B} + C + D)$$

**Solution:** The first term must be expanded into  $A+B+C+D$  and  $A+B+C+D$  to get a standard POS expression, which is then mapped; and the cells are grouped as shown in **Figure 4.22**. The sum term for each group is shown and the resulting minimum POS expression is

$$(C + D)(A + B + D)(\bar{A} + B + C)$$

Keep in mind that this minimum POS expression is equivalent to the original standard POS expression.

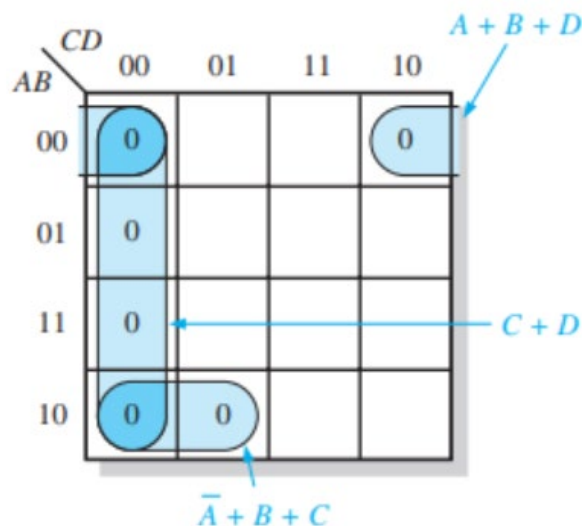


Figure 4.22: Example of mapping a standard POS expression on a K-Map and Grouping the 0's

## Converting Between POS and SOP Using the Karnaugh Map

When a POS expression is mapped, it can easily be converted to the equivalent SOP form directly from the Karnaugh map. Also, given a mapped SOP expression, an equivalent POS expression can be derived directly from the map. This provides a good way to compare both minimum forms of an expression to determine if one of them can be implemented with fewer gates than the other.

For a POS expression, all the cells that do not contain 0s contain 1s, from which the SOP expression is derived. Likewise, for an SOP expression, all the cells that do not contain 1s contain 0s, from which the POS expression is derived.

## Common Pitfalls and Best Practices for K-Maps

1. **Do not Ignore Small Groups:** Even small groups of 1s (like pairs) can help simplify your Boolean expression.
2. **Mark Groups Clearly:** Circle the groups you identify on the K-Map to keep track and avoid mistakes.
3. **Avoid Overlaps:** Make sure the groups do not overlap unnecessarily, as this can lead to errors.
4. **Look for Better Groupings:** Sometimes there is more than one way to group the 1s. Choose the grouping that gives the simplest solution.

## Optimising Robot Control Systems: Practical Example

At this point, you will examine how we will use a practical example to demonstrate how Karnaugh Maps (K-Maps) can be applied to optimise the logic design of robot control systems.

**Real-world Scenario:** *Design a logic system to control a robot's movement. The robot has three sensors (A, B, C) that detect obstacles in front, left, and right, respectively. The robot should move forward (F) only if there are no obstacles in front and either no obstacle on the left or right.*

### Solution

**Step 1 (Understanding the Problem Statement):** First you need to identify the specific problem or functionality that the digital circuit needs to solve or perform and translate the problem statement into a functional requirement.

From the problem statement you we can derive the variables A, B and C which will determine forward movement. For forward movement (F), sensor A needs to sense no obstacles and(.) either no on sensor B or (+) C.

#### Variables

- a. A: Obstacle sensor (front) - 0 (no obstacle), 1 (obstacle)
- b. B: Obstacle sensor (left) - 0 (no obstacle), 1 (obstacle)
- c. C: Obstacle sensor (right) - 0 (no obstacle), 1 (obstacle)
- d. F: Forward movement output - 0 (stop), 1 (move forward)

**Step 2: Develop a truth table:** From the derived problem statement, develop a truth table. The truth table for this our scenario is:

A	B	C	F
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

Figure 4.23

**Step 3 (Derive the Boolean Expression):** Write the initial Boolean expression that represents the logic of the Truth Table. From the truth table in **Figure 4.23**, you should be able to see that  $F = 1$  only when there is no obstacle in front ( $A = 0$ ) and either no obstacle on the left ( $B = 0$ ) or right ( $C = 0$ ). This translates to the Boolean expression:  $F = \overline{A}BC + \overline{A}B\overline{C} + \overline{A}\overline{B}C + \overline{A}\overline{B}\overline{C}$

**Step 4 (Use K-Maps to Simplify Boolean Expressions):** draw a K-map to further simplify the derived Boolean expression. The following K-map was derived from the Boolean expression using the logical steps a to c:

- This K-Map will have  $2^3 = 8$  cells (3 variables).
- Assign A, B, and C to the axes (order does not affect the outcome).
- Mark 1s in the cells where the truth table indicates forward movement ( $F = 1$ ). Here, the 1s will be concentrated in the top row ( $A = 0$ ) and form a horizontal line across B and C (representing either  $B = 0$  or  $C = 0$ ).

		C	
		0	1
AB	00	1	1
	01	1	1
	11	0	0
	10	0	0

**Figure 4.24: Karnaugh map of derived Boolean expression**

**Step 6 (Derive the Simplified Expression):** By Grouping the 1s and simplifying further, the following expression was derived:  $F = \overline{A}$

**Step 7 (Design the Circuit Diagram):** Based on the simplified expression  $F = \overline{A}$ , create the optimised circuit diagram. The circuit will use a NOT gate to invert the input A, allowing the robot to move forward when there is no obstacle in front.

By following these steps, you can learn how to apply K-Maps to optimise the logic design of robot control systems, making them more efficient and effective.

## Benefits of K-Maps

Karnaugh Maps offer a visual and systematic approach to simplifying Boolean expressions, leading to an optimised circuit with fewer gates. This reduces hardware complexity, potentially improving the robot's performance and efficiency.

### Activity 4.2 Exploring Simple Boolean Expressions with K-Maps

1. Simplify the expression  $AB+AB$  using a 2-variable K-Map.
2. You can go about this task by following these steps:
  - a. Create a truth table for the expression.
  - b. Map the values onto a 2-variable K-Map.
  - c. Group the adjacent 1s and write down the simplified expression.
  - d. Explain why this simplified circuit would not need a gate. Write the simplified expression and explain the steps you took.

### Activity 4.3 Solving a Real-World Problem

Critically analyse and select one of the following scenarios to develop truth tables and optimised circuit diagrams.

**Scenario 1:** A robot has three sensors (S1, S2, S3). The output (O) is true if **at least two sensors detect an obstacle**.

1. Formulate the truth table for this scenario.
2. Convert the truth table into a Karnaugh Map.
3. Group adjacent ones and simplify the Boolean expression.
4. Design and draw the optimised circuit diagram based on the simplified expression.

**Scenario 2:** A home security system has three sensors (S1, S2, S3) at entry points. The alarm (A) should trigger if **any sensor detects a breach**.

1. Formulate the truth table for this scenario
2. Convert the truth table into a Karnaugh Map.
3. Group adjacent ones and simplify the Boolean expression.
4. Design and draw the optimised circuit diagram based on the simplified expression.

*You can go about any of these tasks by following these guidelines:*

- a. Formulate truth table: First identify the variables and the possible outputs and plot them onto a truth table.
- b. Convert the truth table into a Karnaugh Map
  - i. Plot the truth table values onto a 3-variable K-Map.
  - ii. Group adjacent 1s to simplify the Boolean expression.
  - iii. Write down the simplified expression and explain how it represents the robot's forward movement.
- c. Draw the optimised circuit diagram based on the simplified expression using basic logic gates.

- d. Present your design to the class. Discuss challenges and how the K-Map helped simplify the system.

## BUILDING AND TESTING BASIC COMBINATIONAL CIRCUITS ON PRINTED CIRCUIT BOARDS

### What is a Printed Circuit Board (PCB)?

Imagine a city. Roads connect different parts of the city, allowing people and vehicles to move around. Similarly, a printed circuit board (PCB) is like a city for electronic components. It's a flat board with copper tracks that connect different electronic components, like transistors, resistors, and capacitors.

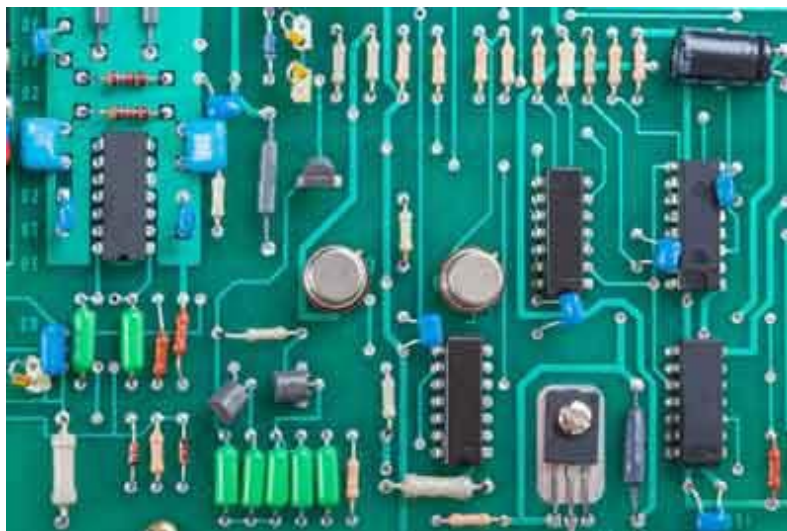


Figure 4.25: A Printed circuit board

### Why Does Electronics Use PCBs?

1. **Organisation:** PCBs keep all the electronic components in one place and organised.
2. **Reliability:** They make sure that the connections between components are strong and reliable.
3. **Efficiency:** PCBs help manufacturers us make smaller and more efficient electronic devices.
4. **Accuracy:** Machines can make PCBs very precisely, reducing errors.



## Parts of a PCB

A PCB can have many layers of copper, like a layer cake. More layers mean more complex circuits, but they also make the PCB more expensive and harder to repair. The most common layers in a PCB are as follows.

- Copper Layers:** These are the electrical pathways, like roads on a city map.
- Substrate:** This is the base material of the PCB, similar to the ground beneath a city.
- Solder Mask:** This protects the copper tracks from damage, like a protective coating on a road.
- Silkscreen:** This layer has labels and markings, like street signs, to help people understand the PCB.

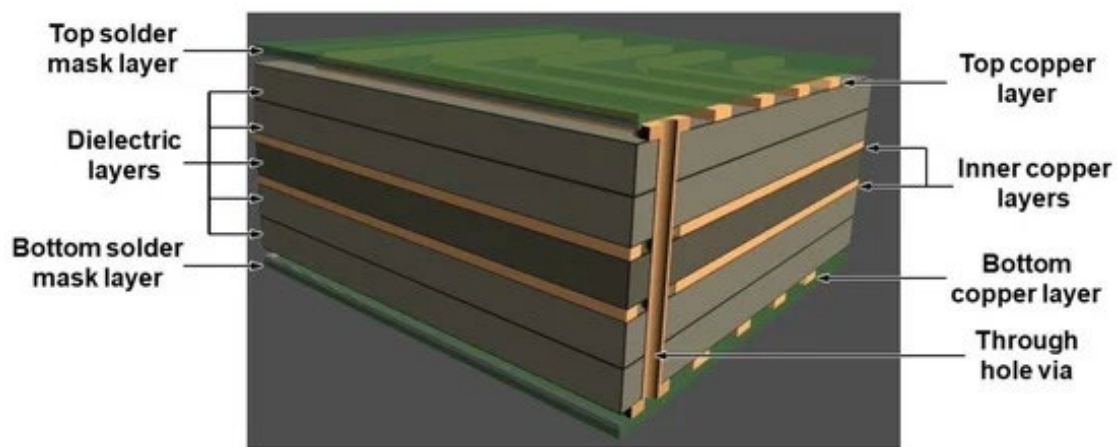


Figure. 4.26: A 3-D schematic design of a multi-layer printed circuit board (PCB) board composed of four copper layers (two internal layers), five dielectric layers, and the top and bottom mask layer.

## Common Components mounted on a PCB

**Resistors:** These are like traffic lights for electricity. They control the flow of electricity.

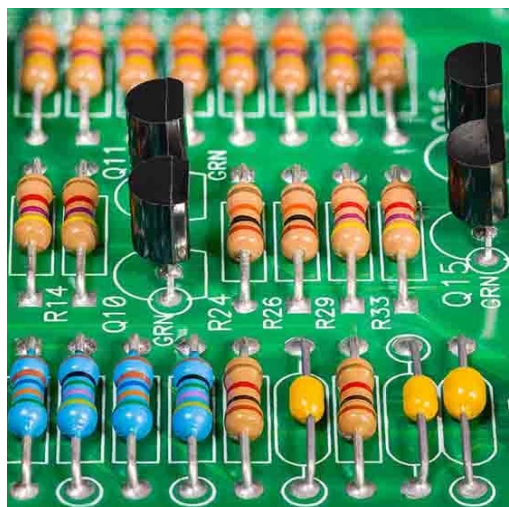


Figure 4.27: Resistors on a PCB

**Capacitors:** These are like tiny batteries. They store and release electrical energy.

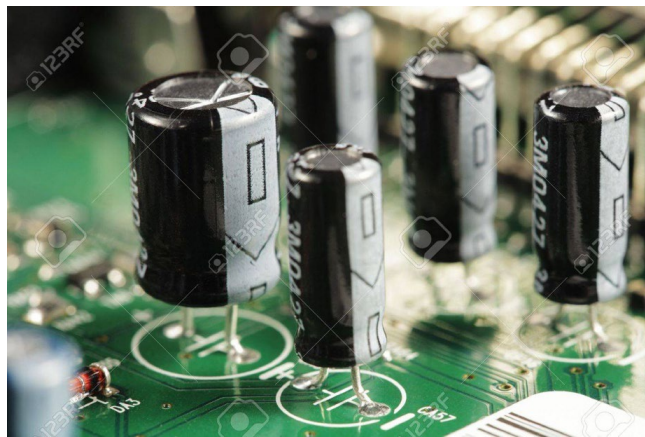


Figure 4.28: Capacitors on a PCB

**Transistors:** These are like electronic switches. They control the flow of electricity.

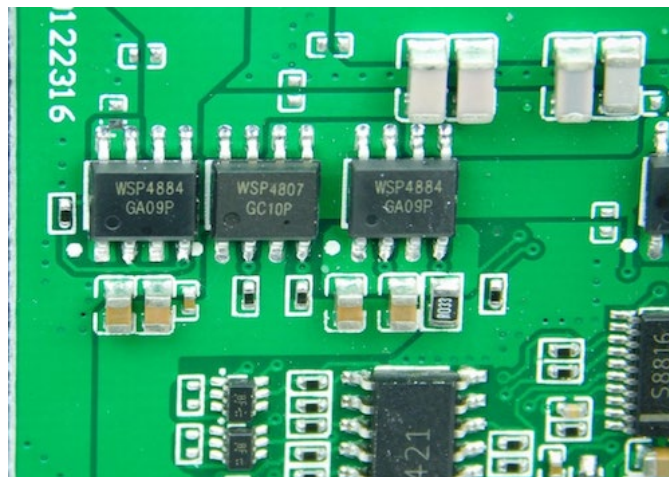


Figure 4.29: Transistors on a PCB

**Integrated Circuits (ICs):** These are tiny chips that contain many transistors and other components. They do complex tasks, like processing information or amplifying signals.

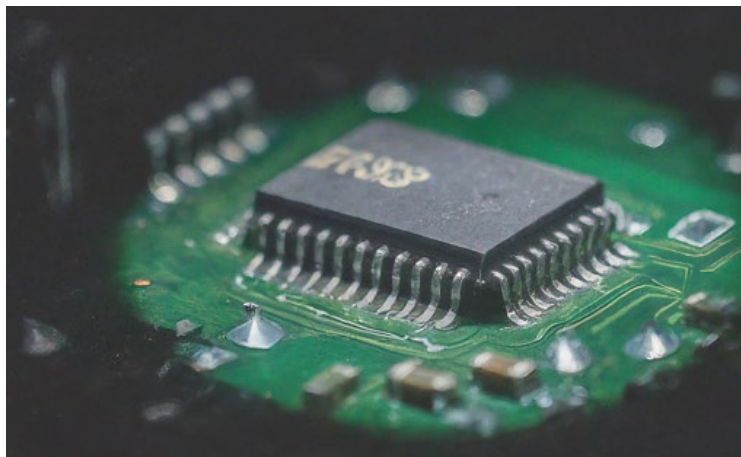
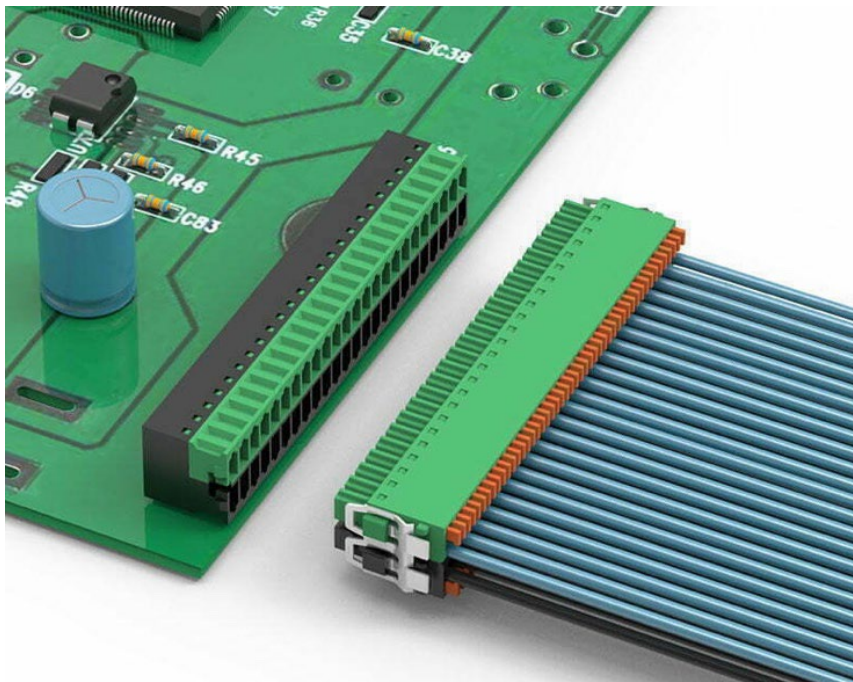


Figure 4.30: An IC on a PCB

**Connectors:** These are used to connect the PCB to other devices, like cables and wires.



**Figure 4.31: Connectors on a PCB**

## Soldering Techniques

Soldering is a fundamental skill in electronics, involving the joining of electronic components using a heated soldering iron and molten solder. It's like welding, but on a much smaller scale. By mastering soldering techniques, you will be able to:

1. **Build Your Own Electronic Projects:** From simple circuits to complex robots, soldering allows you to bring your ideas to life.
2. **Repair Electronic Devices:** Extend the life of your gadgets by fixing broken parts.
3. **Customise Electronics:** Modify existing devices to suit your needs.
4. **Understand How Electronics Work:** Soldering gives you hands-on experience with the inner workings of electronic devices.

## Safety Measures to Observe when Soldering

Before you begin practising soldering techniques, observe the following measures:



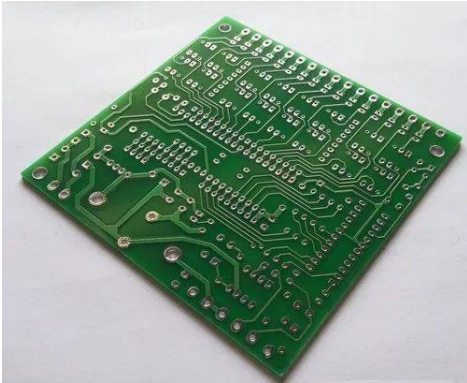
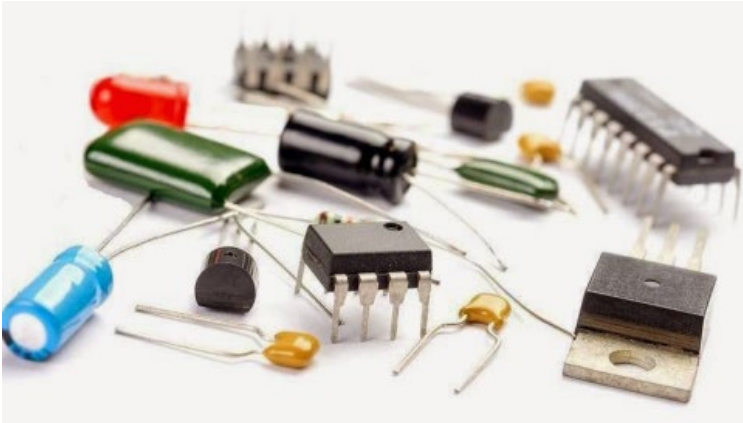
1. **Ventilation:** Always work in a well-ventilated area to avoid breathing in harmful fumes.
2. **Eye Protection:** Wear safety glasses to protect your eyes from hot solder splashes.
3. **Hot Iron:** Be careful not to touch the hot tip of the soldering iron.
4. **Fire Hazard:** Keep flammable materials away from your workspace.



## Tools used in Soldering

The following tools will become helpful when soldering:



Table 4.1: Soldering Tools

Name and function	Image
<i>Soldering Iron:</i> This is used to melt the solder	
<i>Solder:</i> This is a metal alloy that melts and hardens to connect components.	
<i>PCB:</i> This is the board where you will solder components.	
<i>Components:</i> These are the electronic parts, like resistors, capacitors, and ICs.	

<b><i>Tweezers:</i></b> These are used to hold small components.	
<b><i>Solder Wick:</i></b> This is used to remove excess solder.	

## Proper Handling of Soldering Equipment

1. Use a soldering iron with an appropriate wattage (typically 15-30 watts for electronic work).
2. Ensure the soldering iron tip is clean and well-tinned (covered with a thin layer of solder).
3. Use a damp sponge or brass wire cleaner to wipe the tip periodically.
4. Handle the soldering iron by its insulated grip to avoid burns or electric shock.



Figure 4.32: Proper handling of soldering equipment

## Steps in Soldering

Most soldering activities on A PCB will require following the steps below.

1. Prepare the PCB: Make sure the holes in the PCB are clean.
2. Insert Components: Place the components into the holes.



- 3. Heat the Joint: Use the soldering iron to heat both the component lead and the PCB pad.
- 4. Apply Solder: Touch the solder to the heated joint.
- 5. Remove the Iron: Once the solder flows smoothly, remove the iron.
- 6. Cool Down: Let the joint cool naturally.

### Mistakes to Avoid When Soldering

The following are three common mistakes people make when soldering and how to avoid them.


- 1. **Cold Joint:** This happens when the solder doesn't flow properly. To avoid it, make sure both the component and the PCB pad are hot enough.
- 2. **Solder Bridge:** This happens when solder connects two nearby points accidentally. To avoid this, always use less solder and be careful.
- 3. **Overheating:** Too much heat can damage components. To prevent this, do not keep the iron on one spot for too long.

Watch the following videos for tutorials on how to solder on a PCB board.

Description	QR CODE
Soldering Crash Course: Basic Techniques, Tips and Advice!	
HOW TO SOLDER! (Beginner's Guide)	

### Building a circuit (Practical Example)

Scan the QR codes below for videos on practical examples on how to build a circuit

Resource	QR Code
How to Build a Circuit from a Circuit Diagram	

Design and Build a PCB - SMD LED Learn electronics engineering



## Testing and Troubleshooting

### *How to Use a Multimeter and Other Tools to Check if a Circuit Works Properly*

- 1. Select the correct setting on the multimeter.**

Choose the mode for what you want to measure, such as voltage, continuity, or resistance.

- 2. Check for voltage at important parts of the circuit.**

Measure the voltage at different points to make sure power is flowing to all the components.

- 3. Test for good connections using the continuity mode.**

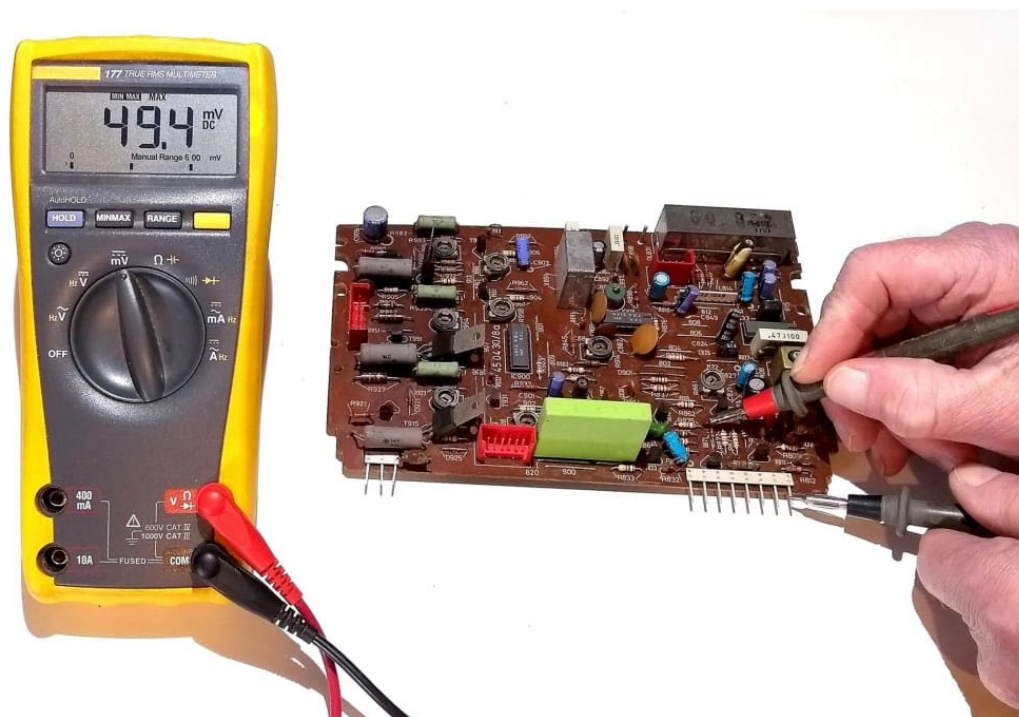
The multimeter can check if the wires and parts are properly connected or if there is a break in the circuit.

- 4. Measure the resistance of resistors.**

Check the resistance of resistors to confirm they have the values written on them.

- 5. Test diodes and transistors.**

Use the diode test mode to make sure diodes and transistors are working correctly.



**Figure 4.33: Using a Multimeter to test PCB components**



## Other Testing Tools

**Oscilloscope:** Use to observe the signal waveforms at various points in the circuit.

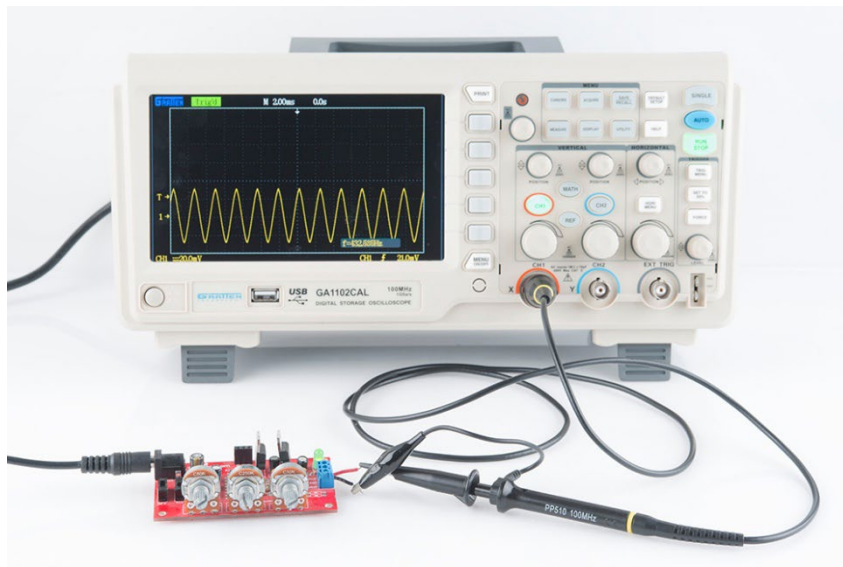


Figure 4.34: Using an Oscilloscope to test PCB components

**Logic Analyser:** Helpful for debugging digital circuits by capturing and displaying multiple signals.

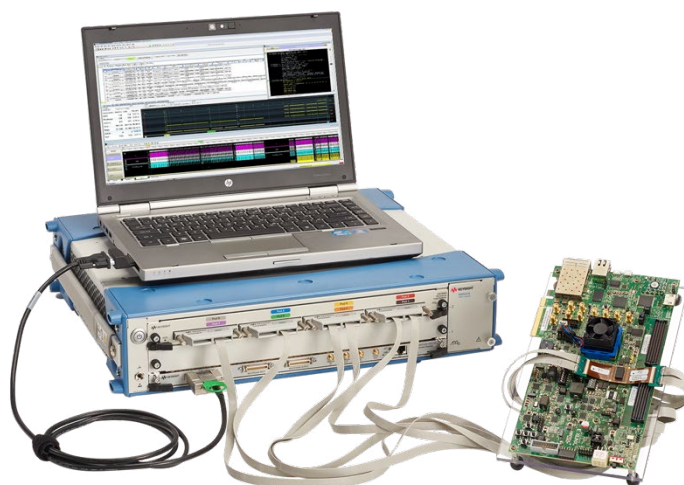


Figure 4.35: Logic Analyser Module

## Common Problems in PCB Assembly and How to Find Them

### 1. Cold (or dry) Joints

- a. **What it looks like:** The circuit works sometimes or does not work at all.
- b. **How to check:** Look at solder joints carefully. If they look dull or cracked, they may need fixing. Reheat the solder to make a smooth and shiny joint.

## 2. Bridging

- a. **What it looks like:** The circuit shows strange behaviorbehaviour, or some parts stop working because of short circuits.
- b. **How to check:** Look for extra solder connecting two pads that should not be joined. Use a desoldering tool to remove the extra solder.

## 3. Wrong Component Placement

- a. **What it looks like:** The circuit does not work as planned.
- b. **How to check:** Compare how the components are placed with the circuit diagram (schematic). Make sure polarised components, like diodes or capacitors, are in the correct direction.

## 4. Damaged Components

- a. **What it looks like:** Some parts of the circuit do not respond or behave incorrectly.
- b. **How to check:** Use a multimeter to test each component. Replace any component that does not give the correct reading.

# Practical Tips for Troubleshooting PCBs

## Step-by-Step Approach

### 1. Start with a visual check.

Look for obvious problems like solder bridges or components in the wrong place.

### 2. Test the power supply.

Check if the circuit is getting the correct amount of power.

### 3. Follow the schematic.

Use the circuit diagram to trace the flow of signals and check each section step by step.

### 4. Divide and test.

Break the circuit into smaller sections and test each part separately to find where the problem is.

## Keep Good Records

1. Write down your measurements and observations while troubleshooting.
2. Compare your results with the expected values from the schematic or the datasheet of the components.

## Be Patient and Persistent

1. Solving problems can take time. If you are stuck, take a break and come back with fresh eyes.
2. Work with classmates or ask your teacher for help if needed.

## Why Precision is Important in PCB Assembly

1. **Ensures the Circuit Works Correctly**
  - a. **Correct Component Placement:** Placing components in the right spot and direction ensures the circuit performs as designed. Misplaced parts can cause failures or even damage components.
  - b. **Accurate Soldering:** Proper soldering prevents problems like cold joints or solder bridges, ensuring reliable electrical connections.
2. **Reduces Errors and Saves Time**
  - a. **Fewer Defects:** Being careful during assembly reduces mistakes, saving time and effort on repairs.
  - b. **Consistent Quality:** Precision leads to high-quality circuits that meet the required standards, whether for small projects or large-scale production.
3. **Improves Durability and Reliability**
  - a. **Long-Lasting Circuits:** Careful assembly ensures the circuit works well over time.
  - b. **Avoids Failures:** Precision is especially important in critical applications, like medical devices or car electronics, where failures can be serious.

## Why PCB Skills Are Useful

### Professional Benefits

1. **Better Job Opportunities:** Knowing how to assemble PCBs can lead to careers in electronics design, manufacturing, or repair. Industries like robotics and telecommunications need skilled workers.
2. **Advanced Roles:** Mastering PCB skills can help you move into roles in project management, quality control, or research and development.
3. **Understanding Standards:** Knowing industry rules for PCB assembly makes you more credible and increases job opportunities.

### Personal Benefits

1. **DIY Projects:** With PCB skills, you can create your own electronic devices, from simple gadgets to complex systems, which encourages creativity.
2. **Problem-Solving Skills:** Designing and fixing PCBs improves your ability to think critically and solve problems.
3. **Lifelong Learning:** Working with PCBs helps you stay updated with new technologies, fostering continuous learning.

### Activity 4.4 Identifying and Understanding Electronic Components

1. Watch a short video or presentation introducing components like resistors, capacitors, LEDs, diodes, and transistors. Take note of the symbols and functions of each component.
2. **Divide yourselves into groups of 3-5 and get a PCB to work with. This will be also used for following activities.**
  - a. In your small groups, examine your PCB or loose components. Use a magnifying glass (if available) to closely observe each part. Discuss the purpose of each component in the circuit.
  - b. Each group will explain one or two components they identified, including their ratings and significance after the discussion.
3. Independently research one specific component (e.g., resistor, diode) and prepare a short presentation. Explain the component's role in circuits, its rating, and why it is important.
4. Write a summary of 100 – 150 words of the components you explored, including their functions and key features.

### Activity 4.5 Measuring Component Ratings with a Multimeter




1. Observe the teacher's demonstration on how to use a multimeter to measure resistance, capacitance, and continuity or watch a video on same topic. Take notes on the steps and safety precautions.
2. Work with a friend and in pairs, use a multimeter to measure the values of various components like resistors and capacitors on the PCB provided at your station. Record your readings in a table and compare them with the standard values printed on the components.
3. Use the multimeter to test faulty components you have access to provided by your teacher. Identify and document the errors or inconsistencies.
4. Share your findings with the class, explaining common issues you encountered and how you resolved them.
5. Reflect on what you learned about using a multimeter and its importance in troubleshooting circuits.

### Activity 4.6 Soldering a Simple Circuit on a PCB

1. Learn how to read a schematic diagram by watching a demonstration from a video or from your teacher. Pay attention to symbols, connections, and component placement.

2. **In your groups**, plan your layout for a simple circuit, such as a **buzzer operated by a switch**. Sketch your layout and mark where each component will go.
3. Follow your schematic and layout to solder the components onto the PCB. Ensure proper soldering techniques to avoid issues like cold joints or bridging.
4. Use a multimeter to test the soldered circuit. Check for continuity, correct voltage, and proper functionality.
5. If the circuit does not work, identify and fix the problem using the multimeter. Present your circuit to peers for feedback on design and soldering quality.
6. Demonstrate your working circuit to the class, explaining the purpose of each component and how the circuit works.
7. Reflect on your experience, including challenges faced and what you learned about soldering and testing circuits.

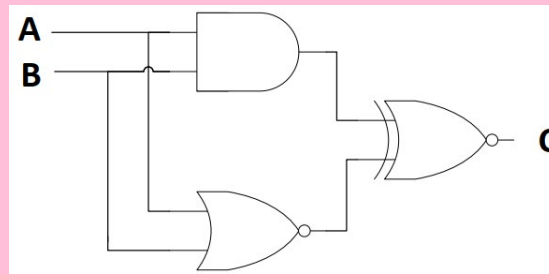
EXTENDED READING

Resource	QR Code to
<p>Online tutorials for Boolean algebra with interactive examples</p> <p><a href="https://www.youtube.com/watch?v=EPJf4owqwdA&amp;list=PLTd6ceoshpreTJdg5AI6i2D2gZR5r8_Aw">https://www.youtube.com/watch?v=EPJf4owqwdA&amp;list=PLTd6ceoshpreTJdg5AI6i2D2gZR5r8_Aw</a></p>	
<p>Boolean Algebra 2 – Simplifying Complex Expressions</p>	
<p>Digital Logic - Implementing a logic circuit from a Boolean expression.</p>	

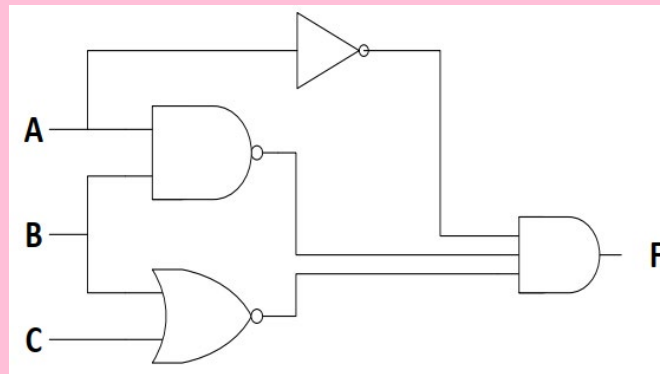
## REVIEW QUESTIONS 4.1

1. Generate the truth table and the final output expression for the following combinational logic circuits.

a.



b.



2. Simplify the following Boolean expressions using applicable Boolean theorems and laws:
  - a.  $(A+B+C) ( A+ B+C)(A+B+C)$
  - b.  $AB+ABC+ABC$
  - c.  $ABC+ABC+ABC+ABC$

## REVIEW QUESTIONS 4.2

1. What is a Karnaugh Map (K-Map)?
2. Explain the purpose of Karnaugh Map in digital system optimisation.
3. Fill out the K-Map for the Boolean expression  $AB + AB$ . Circle the adjacent 1s and write the simplified Boolean expression.
4. A robot with three sensors detects obstacles. The robot should move forward (F) only if there is no obstacle in front ( $S1 = 0$ ) and at least one of the side sensors ( $S2, S3$ ) detects no obstacles.
  - a. Derive the Boolean expression for the forward movement output (F) based on the scenario.
  - b. Simplify the expression using a K-Map and provide the optimised circuit design using logic gates.



## REVIEW QUESTIONS 4.3

1. What is the purpose of a resistor in an electronic circuit?
2. Explain how to use a multimeter to check the rating of a resistor. What steps will you follow to ensure your measurement is accurate?
3. Explain the steps involved in measuring the capacitance of a capacitor using a multimeter.
4. After soldering a simple LED blinking circuit onto a PCB, you find it is not working. Explain how you would use a multimeter to identify and fix potential faults.
5. Why is it important to ensure precision when soldering components onto a PCB? Provide two examples of potential problems that could arise from poor soldering techniques and how they can be avoided.
6. Create a detailed plan for converting a complex schematic diagram into a soldered circuit on a PCB. Include steps for layout planning, component placement, and soldering. Explain how you would test the circuit to ensure it functions correctly.



SECTION

# 5

## TOOLS AND APPS FOR ROBOT DESIGN 2

# ROBOT DESIGN METHODOLOGIES

## Tools & Apps for Robot Design

### INTRODUCTION

This section focuses on building a strong foundation in digital design and its practical application through 3D printing. You will begin by learning how to use computer-aided design (CAD) software to create precise three-dimensional models of robotic components. You will then use slicing software to convert these models into g-codes, which are instructions that a 3D printer can follow. By integrating digital design and fabrication, you will gain hands-on experience in transitioning from conceptualising robotic designs to creating tangible prototypes. Through these activities, you will not only understand the design-to-production process but also develop innovation and problem-solving skills essential for robotics development. By the end of this section, you will be equipped to design, print, and refine robotic parts, laying the groundwork for advanced design and engineering challenges.

#### KEY IDEAS

- **Additive Manufacturing:** Additive manufacturing, like 3D printing, creates objects layer by layer. By designing digital models and printing them with FDM printers, you can quickly make and test robotic parts.
- **Computer-Aided Design (CAD):** CAD software such as Tinkercad and Fusion 360 enables the creation of precise 2D and 3D models for robotic components. It allows users to start with basic designs and progressively develop more advanced simulations and manufacturing-ready models.
- **Digital-to-Physical Workflow:** CAD models are turned into physical objects using slicing software and 3D printing methods like FDM and SLA. FDM is often chosen because it is safe, easy to use, and ideal for learning environments.
- **3D Printing Workflow:** Making functional robotic parts involves several steps, starting with designing in CAD software, preparing the model with slicing software, and printing it with a 3D printer. This process helps bring digital designs to life.
- **Material Selection for 3D Printing:** Materials like PLA, ABS, and PETG are used for 3D printing. The choice depends on what the part needs to do, such as being strong, durable, or suitable for certain environments.
- **Parametric Modelling and Assemblies:** Parametric modelling and assembly techniques allow you to adjust design details to make sure components fit together and work properly. These techniques also help improve designs by checking for overlaps (interference checks) and ensuring smooth movement through motion analysis.

# UNDERSTANDING COMPUTER-AIDED DESIGN (CAD)

Computer-Aided Design (CAD) is an essential tool in modern robotics. In this section, you will learn how CAD works as a virtual workshop, helping you design robot parts. By understanding how CAD works, you will be able to turn your ideas for robots into detailed designs, which can later be used for building actual prototypes.

## What is CAD?

Computer-Aided Design (CAD) is a digital tool that helps people create, change, and improve designs. It has changed the way designs are made in many industries, including robotics. CAD software allows engineers and designers to move from traditional drawing methods (like using pencil and paper) to digital tools, making designs more accurate, efficient, and easier to share with others.

## Core Principles of CAD for Robotics

Here are some of the main ideas that show how CAD helps designers create and improve robotic parts:


1. **2D and 3D Modelling:** CAD software allows you to create both 2D sketches (like blueprints) and 3D models of robot parts. This helps you see and understand your designs from different angles.
2. **Parametric Modelling:** Some CAD systems allow you to create designs using parameters or numbers. This makes it easier to change designs quickly and test different ideas.
3. **Assemblies:** CAD can combine different parts into one digital model, helping you check if the parts fit together, move correctly, or interfere with each other. For example, interference checks in CAD help make sure parts do not collide when assembled. Kinematic analysis looks at how the robot moves, like how its joints work and how fast it moves, without worrying about forces or motors.
4. **Drawing Creation:** CAD can automatically create detailed engineering drawings from your 3D models. This ensures that the parts can be made accurately.
5. **Simulation and Analysis:** Advanced CAD software can test designs virtually, simulating things like stress, movement, and other factors to make sure everything works before creating a physical model.

## 2D vs. 3D in CAD

A model is a simplified version of a real-world object or system. In CAD, there are two types of models: 2D and 3D. Understanding the difference between these two is important for using CAD tools effectively when designing.

	2D Model	3D Model
<b>Stands for</b>	Two-Dimensional	Three-Dimensional
<b>Definition</b>	A flat representation of an object, defined by length and width.	A representation of an object with length, width, and depth, providing a realistic view.
<b>Appearance</b>	Like a drawing on paper, lacking depth.	Similar to a physical object, offering a complete visual representation.
<b>Use in CAD</b>	Primarily for creating blueprints, schematics, and technical drawings. Useful for initial design concepts and documentation.	Essential for creating detailed models of complex objects like robotic parts. Allows for analysis, simulation, and visualisation from multiple angles.
<b>Examples</b>	Floor plans, electrical circuit diagrams, orthographic projections.	Solid models of robot arms, gears, and enclosures.

It is quite difficult to show the clear differences of 2D and 3D models on a plain sheet of paper, hence the following video resource below has been provided.

Resource	QR Code
What is 1D, 2D and 3D? <a href="https://www.youtube.com/watch?v=0TUmVaga95o">https://www.youtube.com/watch?v=0TUmVaga95o</a>	

## CAD in the Robotics Workflow

The use of Computer-Aided Design (CAD) is a key part of the robotics development process. CAD helps turn ideas into real designs that can be made into actual robot parts. Here is how CAD fits into the overall robotics workflow.

### 1. Conceptual Design

CAD helps quickly turn rough ideas into visual designs. You can create quick sketches or simple models that help you see how your robot or robot parts might look and function. This step helps you decide which direction to take with your design.

## 2. Detailed Design

Once you have a basic concept, you use CAD to create precise 3D models of the parts. This means adding details like size, shape, and how each part fits together, which is crucial for the actual building of the robot.

## 3. Digital Prototyping

With CAD, you can test your design without actually building it. You can virtually check if parts fit together, move as expected, and identify potential problems before spending time and resources on physical prototypes.

## 4. Manufacturing Integration

After finalising the design, CAD software helps you generate the data needed to make the parts. This might include creating instructions (like G-code) for machines to use, helping ensure that the manufacturing process is smooth and efficient.

# Popular CAD Software for Robotics

Different CAD software programs are available for different needs. Below are some common options.

1. **Tinkercad:** A beginner-friendly, cloud-based tool that is great for those just starting with CAD, often used in schools and for simple designs.
2. **SolidWorks:** A powerful tool used by professionals for designing complex mechanical parts. It is widely used in robotics for its detailed and precise designs.
3. **Fusion 360:** A versatile tool that combines CAD (for design), CAM (for manufacturing), and CAE (for engineering analysis). It is great for robotics because it covers the entire design-to-manufacture process.
4. **Autodesk Inventor:** A CAD program focused on product design and manufacturing. It is used for detailed, industrial-level designs.
5. **FreeCAD:** A free, open-source CAD tool with a lot of features, useful for those who want a customisable, no-cost solution.

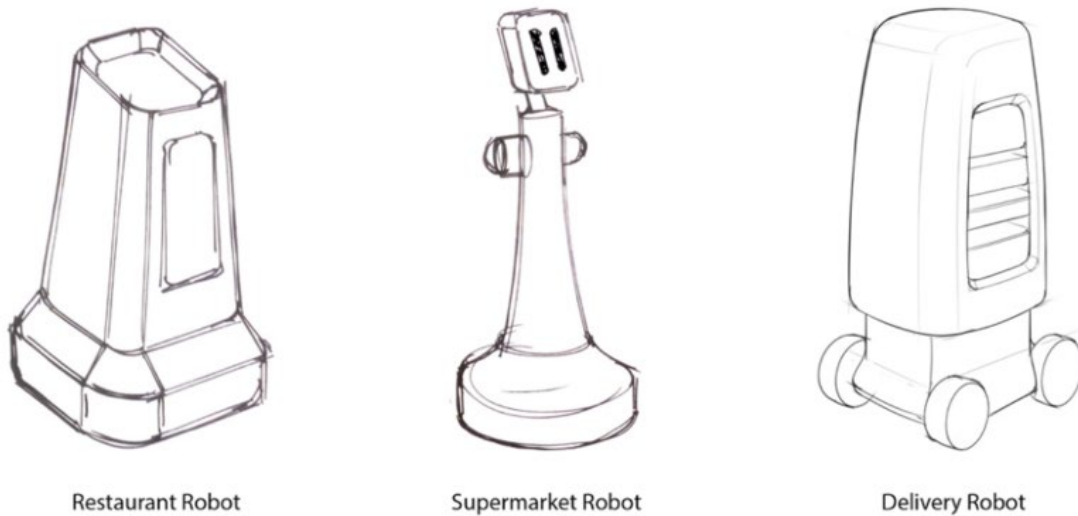
In robotics, CAD is used for creating models, while **Computer-Aided Manufacturing (CAM)** takes those models and turns them into instructions for machines like 3D printers or CNC machines. **Computer-Aided Engineering (CAE)** helps test and optimise designs by simulating how they'll behave in real life.

# Freehand Sketch

While CAD software is powerful for precise designs, starting with a **freehand sketch** can be a helpful part of the design process. Before jumping into CAD, it is good to sketch ideas on paper. This helps you quickly visualise your concept and make changes easily. Sketching can be a great way to brainstorm with your team, get feedback, and refine ideas. Though CAD provides accuracy, freehand drawing often sparks creativity and can help in coming up with innovative ideas that you might not think of otherwise. In robotics, the



initial sketching phase is an important part of thinking through the design and refining it before going digital.



**Figure 5.1: Freehand sketches of robots (source)**

The following resource may be helpful in developing this skill of freehand sketch.

Resource	QR Code
Marklin's Freehand Sketching for Engineers Playlist <a href="https://www.youtube.com/playlist?list=PLbgj-zgqid54GYCrOXtzHYzOF3to1A2M_">https://www.youtube.com/playlist?list=PLbgj-zgqid54GYCrOXtzHYzOF3to1A2M_</a>	

## Getting Started with CAD in Robotics

Choosing the right CAD software can feel overwhelming because there are so many options. For beginners, **Tinkercad** is an excellent starting point. It has a simple, user-friendly interface, making it easy to learn and use. However, because Tinkercad is cloud-based, it might be difficult to access in areas with unreliable internet connections.

In such cases, **Fusion 360** is a good alternative. It is more advanced than Tinkercad but includes offline capabilities, making it more flexible for students in areas without consistent internet. Fusion 360 is also a versatile tool, as it combines design, manufacturing, and analysis features, which are useful for robotics projects.



Mastering either software takes time and practice. To help you get started, step-by-step tutorials for both Tinkercad and Fusion 360 are included in the resources below. These tutorials are designed to guide you through the basic tools and functions, ensuring a smooth learning experience.



## Tinkercad Tutorial Resources:

Resource	QR Code
<p>TinkerCAD - Tutorial for Beginners in 10 MINS! [ FULL GUIDE 2024 ]</p> <p><a href="https://www.youtube.com/watch?v=QIn9c5TjrKk">https://www.youtube.com/watch?v=QIn9c5TjrKk</a></p>	
<p>Tinkercad Tutorial - Complete Guide</p> <p><a href="https://www.youtube.com/playlist?list=PL90LC6zq_Lzf9tHyFPzX_9OA35BFTfEBs">https://www.youtube.com/playlist?list=PL90LC6zq_Lzf9tHyFPzX_9OA35BFTfEBs</a></p>	
<p>[1DAY_1CAD] EOD ROBOT (Tinkercad : Know-how / Style / Education)</p> <p><a href="https://www.youtube.com/watch?v=Qi0lpcgP87c">https://www.youtube.com/watch?v=Qi0lpcgP87c</a></p>	
<p>[1DAY_1CAD] ROBOT DOG SPOT (Tinkercad : Know-how / Style / Education)</p> <p><a href="https://www.youtube.com/watch?v=kjLXoR1G3cY">https://www.youtube.com/watch?v=kjLXoR1G3cY</a></p>	

## Fusion 360 Tutorial Resources:

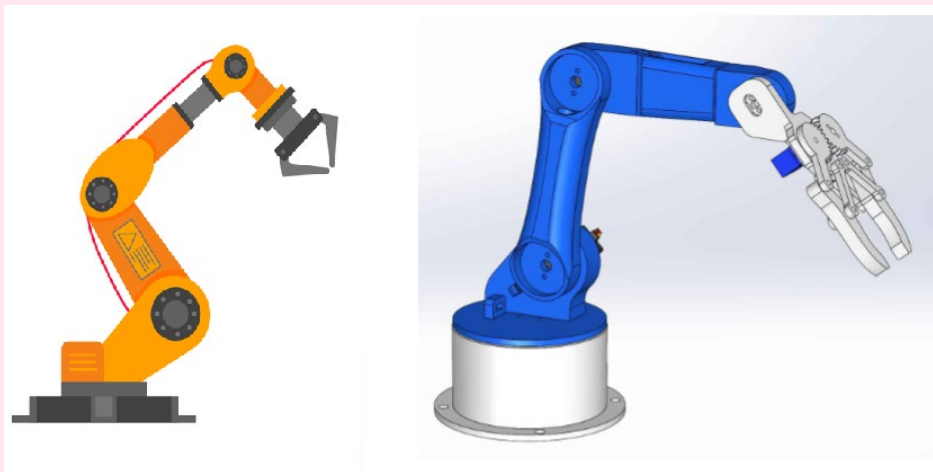
Resource	QR Code
<p>Download and Install - Fusion 360 for Beginners</p> <p><a href="https://www.youtube.com/watch?v=XXEei95rmbI">https://www.youtube.com/watch?v=XXEei95rmbI</a></p>	
<p>Learn Autodesk Fusion 360 in 30 Days for Complete Beginners! 2023 EDITION</p> <p><a href="https://www.youtube.com/playlist?list=PLrZ2zKOTC_-C4rWfapngoe9o2-ng8ZBr">https://www.youtube.com/playlist?list=PLrZ2zKOTC_-C4rWfapngoe9o2-ng8ZBr</a></p>	

### Activity 5.1 Understanding CAD and Its Importance in Robotics

1. Organise yourselves into Your teacher will put you into groups of 3-5 for this activity.
2. Participate in a brainstorming session based on the prompts: “How do you think robots are designed and built? What steps do you think engineers and designers take to bring a robot from an idea to a physical machine?”. Share your ideas about how robots are designed and built. Use the statement starter: “I think robots are designed by...”.
3. Follow-Up: Watch a short video on how CAD is used to design robotic components. After watching, write down at least three benefits of using CAD in robotics.

### Activity 5.2 Exploring 2D and 3D Models

1. In your assigned groups, examine the images in **Figure 5.2** below and discuss the differences between 2D and 3D models. Record your observations in a table comparing the two.



**Figure 5.2: 2D and 3D Models of Robotic Arms**

2. Identify how each of these models is used in design.

### Activity 5.3 Freehand Sketching for Concept Development

1. Sketch a simple design for a robotic component, such as a wheel or a gripper (or any other robotic component), on paper. Focus on capturing the basic shape and dimensions.
2. Pair up with a classmate and exchange sketches. Provide feedback on how the sketches could be improved for clarity or accuracy.

### Activity 5.4 Getting Hands-On with CAD Software

#### 1. Exploring Basic CAD Features

- a. Pair with another person in your group and use Tinkercad (or another CAD platform) to create and manipulate basic 2D shapes, such as squares and circles. Combine these shapes to form a simple robot base.
- b. Share your completed 2D model with your group and explain the steps you used to create it.

#### 2. Building 3D Models

- a. Extend your skills to build a simple 3D model, such as a cube or cylinder. Experiment with resizing, rotating, and combining shapes to create robotic components.
- b. Save screenshots of your designs at different stages to document your process.

### Activity 5.5 Modelling a Robotic Component or Assembly

1. In your assigned groups, using your CAD software, model a simple robotic component of your choice, such as a gripper, wheel, or robot base. Alternatively, create a basic assembly like a robot arm or a line-following robot.
2. Write a 100-150-word short description of your design choices and challenges you faced. Prepare to present your model to the class or set it up in a design gallery for peer feedback.
3. Present your completed model to the class. Explain the steps you followed, the tools you used, and the decisions you made during the design process. Highlight any difficulties and how you overcame them.
4. Provide feedback to your peers by exploring their designs and noting what you learned from their approaches.

## EXPLORING 3D PRINTING IN ROBOTICS

After learning how to create digital designs using CAD tools, you are ready to take the next step: turning those virtual designs into real objects using **3D printing**. This exciting technology allows you to create physical items from your digital designs, bridging the gap between what you imagine on a computer and what you can hold in your hands. Over the next section, you will learn how to use tools like **slicing software** to turn a CAD model of a robot part into the instructions (called **G-codes**) needed for a 3D printer. Then, you will get to print your design and bring it to life.

## Why is 3D Printing Important?

3D printing is changing how manufacturers create and build things. It has become a key tool in many industries and is especially valuable in robotics. Here is why it is important

1. **Rapid Prototyping:** You can quickly create and test mechanical parts, speeding up the design process.
2. **Hands-On Learning:** It helps you understand how designs work in real life.
3. **Encourages Creativity:** You can experiment with different ideas and see how they turn out.

By turning your digital designs into real objects, 3D printing makes learning about robotics more fun and practical.

## Understanding Manufacturing Processes

To make any product, raw materials need to go through a **manufacturing process**. These processes are ways of turning materials into finished items. There are three main types:

1. **Subtractive Manufacturing:** Material is removed to shape an object. E.g. Carving a chair out of a block of wood.
2. **Formative Manufacturing:** Materials are shaped without removing anything. E.g. Making pots from clay or casting metal parts like wheels.
3. **Additive Manufacturing (3D Printing):** Objects are built layer by layer by adding material. E.g. Using a 3D printer to create a robot part.

Before 3D printing, most manufacturing used subtractive or formative methods. Additive manufacturing is now popular because it is flexible, less wasteful, and ideal for creating complex shapes.

## What is a 3D Printer?

A **3D printer** is a machine that turns digital designs into physical objects. It works by using the **additive manufacturing** method, where material (like plastic) is added layer by layer until the object is complete. Think of it like stacking sheets of paper to form a 3D shape.

## Why is 3D Printing Special?

1. **Design Freedom:** You can create shapes that are difficult or impossible to make with other methods.
2. **Less Waste:** It uses only the material needed for the object, which reduces waste.
3. **Quick Results:** You can rapidly turn ideas into real prototypes, making it perfect for robotics.



## 3D Printing in Real Life

3D printing is used in many fields, including

1. **Manufacturing:** To make tools and parts.

2. **Healthcare:** For custom prosthetics (artificial limbs) and implants (hip implants).
3. **Art and Design:** For creating sculptures and models.

In this lesson, you will learn the basics of 3D printing, explore its practical uses, and print a part you have designed, helping you understand the journey from concept to creation.

Resource	QR Code
What is 3D printing? <a href="https://www.youtube.com/watch?v=bcTzyx35odY">https://www.youtube.com/watch?v=bcTzyx35odY</a>	
What Is 3D Printing and How Does It Work?   Mashable Explains <a href="https://www.youtube.com/watch?v=Vx0Z6LplaMU">https://www.youtube.com/watch?v=Vx0Z6LplaMU</a>	

## Types of 3D Printers

There are different kinds of 3D printers, each using a unique method to create objects. Below is the explanation of the two most common types of 3D printers:

### Fused Deposition Modelling (FDM)

**How it works:** FDM printers use a heated nozzle to melt a plastic filament, which is then squeezed out in thin layers to build up the shape of the object.

**Why it is popular:** FDM is the most widely known and used type of 3D printer. It is easy to use, affordable, and can print with a variety of materials like plastic.


**Example:** You can think of it like a hot glue gun that builds up layers of plastic to create a model.

### Resin 3D Printing (SLA)

**How it works:** SLA printers use a special liquid resin. A laser shines into the liquid resin, hardening the material layer by layer to form an object.

**Why it is great for details:** Resin 3D printers are perfect for making highly detailed parts because the laser can create fine details that other printers might not be able to.


**Example:** Imagine a laser painting each layer of your design on the surface of the liquid, creating a solid object step by step.

Resource	QR Code
<p>The 10 Main Types of 3D Printer Explained</p> <p><a href="https://www.3dsourced.com/3d-printers/main-types-of-3d-printer-explained/">https://www.3dsourced.com/3d-printers/main-types-of-3d-printer-explained/</a></p>	

## FDM vs. Resin Printing

FDM and Resin printing are two different types of 3D printing technologies. Below are the main differences between them.

FDM Printing	Resin Printing
<b>How it works:</b> FDM uses a spool of plastic filament, which melts and is laid down layer by layer to create the object.	<b>How it works:</b> Resin printing uses a vat of liquid resin, which is hardened by UV light, one layer at a time.
<b>Popularity:</b> FDM printers are more common and are used by most people.	<b>Popularity:</b> Resin printers are less common than FDM printers
<b>Resolution:</b> FDM printers produce objects with lower detail and rougher surfaces. This means they're not ideal for printing very small models.	<b>Resolution:</b> Resin printers create objects with higher detail and smoother surfaces. This makes them great for small or intricate models.
<b>Safety:</b> FDM printers do not use harmful materials, so they do not require special cleaning after printing.	<b>Safety:</b> Resin printers use chemicals that can be toxic, so they need extra steps to clean and cure the print. They are not as safe for children.
<b>Colour:</b> Some FDM printers can print in multiple colours during one print. multiple colours within one print.	<b>Colour:</b> Resin printers generally can not print in multiple colours in one print.

Resource	QR Code
<p>DM vs Resin 3D Printing - Which is Better?</p> <p><a href="https://www.youtube.com/watch?v=gSvjzGnAosI">https://www.youtube.com/watch?v=gSvjzGnAosI</a></p>	

Resin-based 3D printers are great for producing detailed prints, but the liquid resins they use can be harmful. Because of this, they require special safety equipment. This discussion focuses on FDM (Fused Deposition Modelling) printers, which are safer and use solid plastic filaments.

## Parts of an FDM (Fused Deposition Modelling) 3D Printer

An FDM (Fused Deposition Modelling) 3D printer has several important parts that work together to create 3D objects. Here are the main components:

1. **Hotend:** This part contains the nozzle and heating element. It is responsible for melting the plastic filament so it can be laid down in layers.
2. **Extruder:** The extruder pushes the melted filament into the hotend. It has a motor and **gears** that control the speed at which the filament is fed.
3. **Build Plate:** This is the flat surface where the 3D object is printed. It provides a stable base for the melted filament to cool and harden.
4. **Motion System:** This system uses motors to move the print head in different directions (left-right, forward-backward, up-down). This movement helps create the shape of the object.
5. **Controller Board:** Think of this as the “brain” of the printer. It reads the instructions (called G-code) and tells the printer’s parts what to do. It also controls the temperature and power.
6. **Power Supply:** This part gives the printer the electrical energy it needs to work.
7. **Spool Holder:** The spool holder keeps the roll of filament in place as it feeds into the printer.

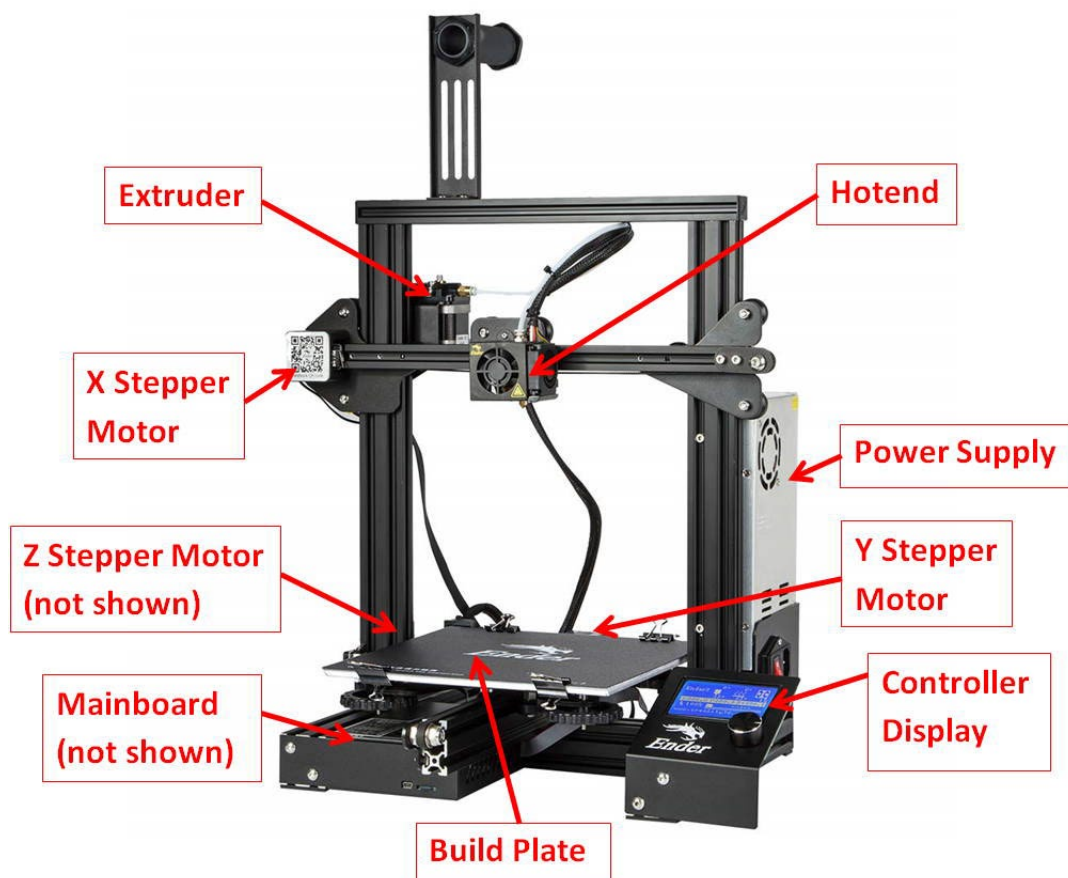


Figure 5.3: The parts of an FDM 3D Printer (source)



Resource	QR Code
<p>How 3D Printing Works</p> <p><a href="https://www.3dmakerengineering.com/blogs/3d-printing/how-3d-printing-works">https://www.3dmakerengineering.com/blogs/3d-printing/how-3d-printing-works</a></p>	

## Filament Materials for FDM (Fused Deposition Modelling) 3D Printing

FDM (Fused Deposition Modelling) 3D printers use different types of plastic filaments to create 3D objects. These filaments are heated, melted, and then laid down layer by layer to form the shape. There are several types of filament, and each one has its own features and uses.

### 1. **Polylactic Acid (PLA):**

PLA is one of the most common materials used in 3D printing. It is easy to work with, non-toxic, and can break down naturally (biodegradable). This makes it good for making prototypes, models, and simple parts.

### 2. **Acrylonitrile Butadiene Styrene (ABS):**

ABS is a strong and durable filament. It is great for parts that need to withstand impact or stress. However, ABS can bend and warp during printing, so it works best with a heated build plate (the surface the object prints on) to prevent this issue.

### 3. **Polyethylene Terephthalate Glycol (PETG):**

PETG combines the good properties of both PLA and ABS. It is tough, clear, and sticks well to the build plate. It is perfect for both prototypes and functional parts that need to be durable but also have good clarity.

### 4. **Acrylonitrile Styrene Acrylate (ASA):**

ASA is similar to ABS in strength, but it is better for outdoor use because it can handle UV light and weather changes without breaking down. This makes it a great choice for parts that will be exposed to the sun or outdoor elements.

## Choosing the Right Filament

When deciding which filament to use, consider what your 3D object will be used for. For example:


1. Will it be used outdoors?
2. Does it need to be tough or flexible?
3. Is it a prototype or a final product?

Different filaments also have different melting temperatures, so be sure to check that before loading the filament into the printer to avoid issues during printing.



**Figure 5.4:** Examples of filaments of different colours.

The resource in the table below demonstrates the right way to load and remove the filament spool in an FDM printer.

Resource	QR Code
Loading & Removing 3d Printer Filament - A Beginner’s Guide <a href="https://www.youtube.com/watch?v=mwdT-XLLnIU">https://www.youtube.com/watch?v=mwdT-XLLnIU</a>	

**General 3D Printing Workflow**

The process of 3D printing follows a set of steps. Here is an overview of the workflow:

**Step One: Design Creation**

The first step in 3D printing is creating or getting a 3D model. This model is the digital version of the object you want to print. You can create the model yourself using **Computer-Aided Design (CAD)** software, or you can download pre-made models from websites like Thingiverse, MyMiniFactory, or Printables. Some models are free, while others need to be bought.

The models are saved in special file formats, most commonly as **STL** files. STL stands for Stereolithography, and it contains information about the shape and structure of the object. These files usually end with “.stl.” Other file types like “.obj” or “.gltf” exist, but they are not used as often.

## Step Two: File Preparation (Slicing)

Once you have your digital 3D model, you need to prepare it for printing. This step is called **slicing**. Slicing is the process of converting the 3D model into a set of instructions for the 3D printer, called **G-code**.

1. Specialised software, called **licer software**, is used for this process. Popular slicers include **Cura**, **PrusaSlicer**, and **Simplify3D**.
2. The slicing software divides the model into horizontal layers. Each layer is a thin slice of the object. The software then creates G-code, which tells the 3D printer how to move, how much material to extrude, and other important details.

There are some important settings (called **slicing parameters**) that affect the print:



- a. **Layer Height:** This controls how thick each layer is. Thinner layers give a smoother finish but take more time to print.
- b. **Infill:** This controls how solid the inside of the object is. More infill makes the object stronger but takes longer to print and uses more material.
- c. **Support Structures:** These are extra materials added to support parts of the object that hang over (like the arms of a robot). They are removed after printing.
- d. **Nozzle Temperature:** This setting controls how hot the printer's nozzle needs to be, depending on the material you are using.
- e. **Build Plate Adhesion:** This ensures that the object sticks properly to the print bed (the surface where the print is made).

## Step Three: Printing

After the slicing software has created the G-code, it is transferred to the 3D printer. This can be done by connecting the printer to a computer with a **USB cable** or by inserting an **SD card** with the G-code file.

1. The printer reads the G-code and follows the instructions to print the object. It does this layer by layer, starting from the build plate and working upwards.
2. The print head moves around, laying down the material to create the object, while the printer follows the path outlined in the G-code.

The following resources are very helpful in getting youlearners started with printing their 3D designed models using an FDM 3D printer.

Resource	QR Code
3D Print Your Own Designs for Free with Tinkercad <a href="https://www.youtube.com/watch?v=n-MOwGsUZ68">https://www.youtube.com/watch?v=n-MOwGsUZ68</a>	
3D PRINTING 101: The ULTIMATE Beginner's Guide <a href="https://www.youtube.com/watch?v=2vFdww4U1VQ&amp;t=616s">https://www.youtube.com/watch?v=2vFdww4U1VQ&amp;t=616s</a>	

### Activity 5.6 Exploring 3D Printing Fundamentals

1. Start by watching a short video on the fundamentals of 3D printing. Pay attention to the history, basic principles, and how 3D printing is used in different industries.
2. Organise Your teacher will then divide yourselves into groups of 3-5 to have a brief discussion. Use the following questions as a guide:
  - a. What materials are typically used in 3D printed robotic components (e.g., grippers, gears)?
  - b. How do you think these components are manufactured in the real world?
  - c. Imagine you need to create parts for a new robot. How would 3D printing help you prototype these parts quickly and efficiently?
3. In your groups, **compare additive manufacturing** (3D printing) to **subtractive manufacturing** and **formative manufacturing**. Discuss their differences and unique advantages. How does 3D printing fit into these processes? Create a chart comparing these methods.
4. Write a short reflection on how 3D printing might impact the future of robotics.

### Activity 5.7 Understanding FDM 3D Printers and Filament Materials

1. In your small groups, research one of the following filament materials: **PLA**, **ABS**, **PETG**, or **ASA**. Focus on:
  - a. Material properties
  - b. Best uses in robotics (e.g., grippers, structural parts)
  - c. Pros and cons of using the material in different conditions (indoor vs outdoor, strength, flexibility)
2. Share your findings with the class. Use the following questions to guide your presentation:
  - a. Which material would you choose for a strong robot arm and why?
  - b. How does temperature affect the choice of filament for 3D printing?
3. Using illustrations or videos, identify the main components of an FDM 3D printer (nozzle, extruder, build plate). Discuss the role of each part in the printing process.

### Activity 5.8 Preparing a 3D Model for Printing

1. In your small groups, choose or design a simple robotic component (e.g., a gear or bracket) using **CAD software**. You may either create your own model or choose a pre-designed model from an online repository like Thingiverse.
2. **Slicing the Model:**

- a. Open the slicing software (e.g., **Cura** or **PrusaSlicer**).
  - b. Load the CAD model and adjust the slicing settings. Focus on:
    - i. **Layer height**
    - ii. **Infill** (e.g., 20% or 50%)
    - iii. **Support structures** (if necessary)
    - iv. Nozzle temperature and build plate adhesion
  - c. Prepare the model for printing by generating the **G-code**.
3. Working in small groups, collaborate to prepare the model for printing. Discuss the settings you chose and why. Test different slicing parameters with the group and compare the outcomes.

### Activity 5.9 Printing and Post-Processing

1. In your pre-existing small groups, using the FDM 3D printer, load the G-code generated in the previous activity. Print the robotic component.
2. As the print progresses, monitor the printing process and record the time it takes for each layer to be printed. Make sure the build plate is level and that the filament is feeding correctly.
3. **Post-Print Activities:**
  - a. Once the print is complete, carefully remove the part from the build plate.
  - b. If support structures were used, remove them carefully.
  - c. Clean up any excess filament or rough edges on the printed part.
4. After completing the print, reflect on the following:
  - a. What challenges did you face during the printing process?
  - b. What improvements can be made in the next print?

### Activity 5.10 Presentation and Peer Feedback





1. Prepare a short presentation about the component you printed. In your presentation:
  - a. Describe the **design process** using CAD software.
  - b. Explain the **slicing settings** you chose and why.
  - c. Discuss the **material** you selected and its relevance to the robot component.
  - d. Highlight any **challenges** you faced and how you solved them.

2. **Peer Feedback:** Set up a “**robot design gallery**” where you and your peers can display your printed models and documentation. Walk around and provide constructive feedback to others, focusing on:
  - a. The functionality of the printed part
  - b. The choice of material and its appropriateness for the robotic function
  - c. Suggestions for improving the print in future attempts

### Activity 5.11 Final Reflection and Self-Assessment

1. Using the rubric provided by your teacher, Assess your performance in the following areas:
  - a. Understanding of 3D printing principles
  - b. Effectiveness in using CAD software
  - c. Successful use of slicing software
  - d. Quality of the printed model
2. Write a brief reflection on:
  - a. What you learned throughout the activities
  - b. Any challenges you faced and how you overcame them
  - c. How you might apply 3D printing in future robotics projects

## EXTENDED READING

Resource	QR CODE
3D Printing Basics - 3D Hubs: A comprehensive guide to 3D printing technologies, materials, and troubleshooting. <a href="https://www.hubs.com/knowledge-base/">https://www.hubs.com/knowledge-base/</a>	
<b>Autodesk Tinkercad Learning Portal:</b> A free platform offering step-by-step lessons and project ideas for CAD beginners. <a href="https://www.youtube.com/@MakersMuse/playlists">https://www.youtube.com/@MakersMuse/playlists</a>	
<b>Fusion 360 Tutorials by Lars Christensen:</b> A series of beginner to advanced tutorials specifically tailored for CAD design. <a href="https://www.youtube.com/@cadcamstuff/playlists">https://www.youtube.com/@cadcamstuff/playlists</a>	
3D Printing 101 by Thomas Sanladerer: An excellent video series introducing 3D printing processes, tools, and common challenges. <a href="https://www.youtube.com/user/ThomasSanladerer">https://www.youtube.com/user/ThomasSanladerer</a>	

## REVIEW QUESTIONS 5.1

1. List three benefits of using CAD in robotics.
2. State two popular CAD software options used for robotics.
3. Explain the difference between 2D and 3D models and provide an example of each used in the robotics design process.
4. Compare and contrast the use of freehand sketching and CAD software in the initial design phase of a robot.
5. Develop a step-by-step guide for a peer who has limited technical experience on how to use a chosen CAD software to create a simple 3D model of a robotic component.



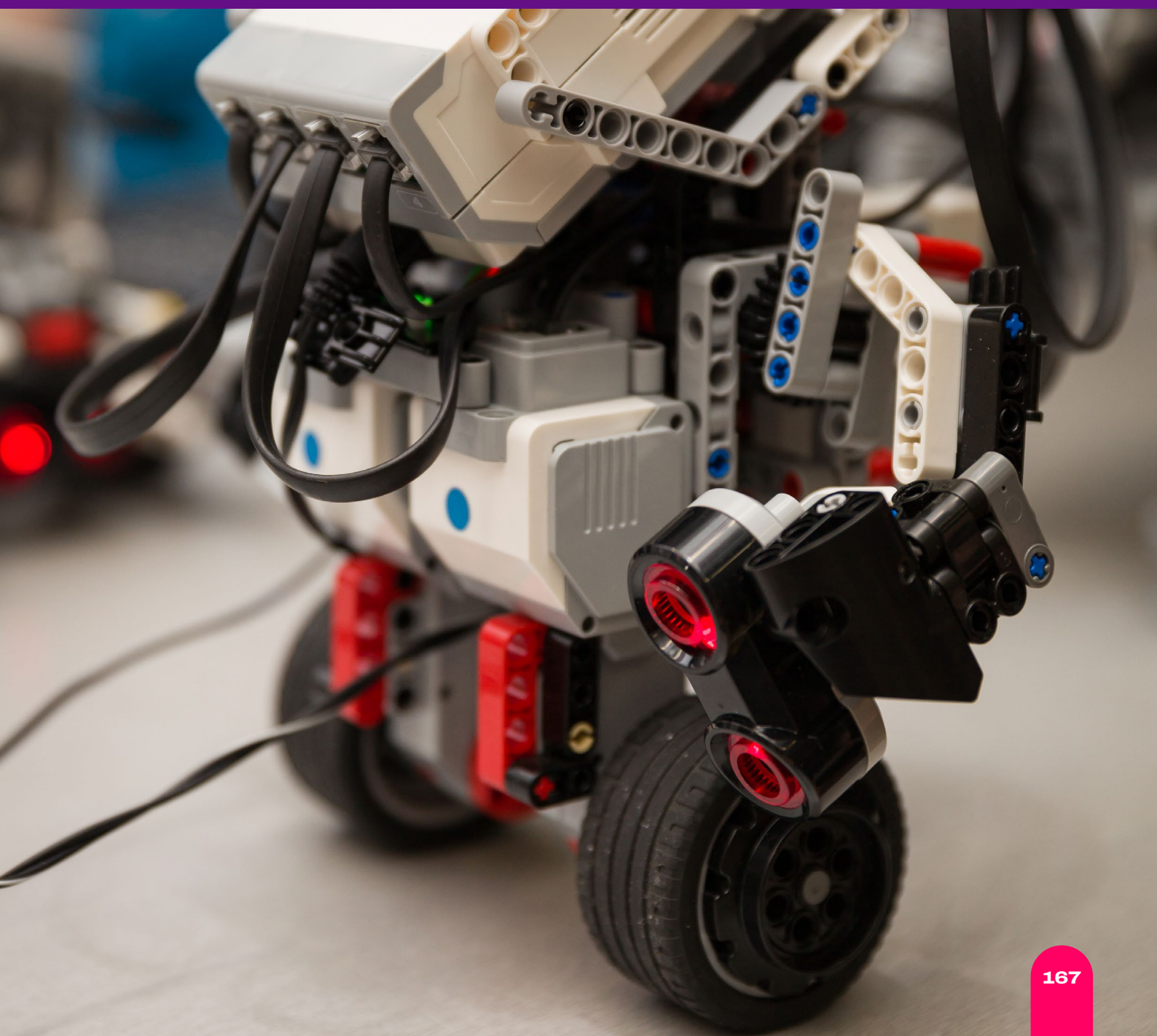
## REVIEW QUESTIONS 5.2

1. Identify the three main types of manufacturing and provide an example for each.
2. Compare and contrast FDM and Resin 3D printing based on factors like material, resolution, and safety considerations.
3. Analyse a table comparing different filament materials (PLA, ABS, PETG) used for FDM printing and recommend the most suitable filament for printing a strong gear component giving reasons for your choice.
4. A student encounters warping issues during their 3D print. Based on the knowledge of factors influencing printing success, suggest a reason for the warping and a solution to address the issue.
5. A robotics team needs to design a custom gripper for their robot. Using CAD software, design a simple gripper that could be 3D printed. Explain the design considerations and justify your material selection for the gripper.

SECTION

# 6

## HIGHER ORDER DESIGN THINKING



# ROBOT CONSTRUCTION & PROGRAMMING

## Higher Order Design Thinking

### INTRODUCTION

In this section, your understanding of how to control robots will be built on, focusing on more advanced ways to guide and manage robots. Ideas about how robots make decisions and change their actions based on feedback from their surroundings will be explored. Simple codes (called pseudocode) and flowcharts will be used to help solve problems in robotics.

A project will be given where a solution to a robot problem will be designed. This will start with basic ideas and then develop into clear instructions for the robot. You will work with your classmates to review ideas, make changes, and improve your solutions. Through this, skills in problem-solving and building strong, efficient robots will be improved.

First, some basic ideas about control systems will be reviewed to get started.

#### KEY IDEAS

- **Drawing Flowcharts to Explain a Process:** Flowcharts use symbols like arrows, circles, and rectangles to show actions and decisions as steps in solving a problem or controlling a system.
- **Understanding Algorithms:** These are step-by-step plans that explain how to solve a problem. They are very important for writing computer programs and creating systems that work automatically.
- **Using Pseudocodes for Planning:** Pseudocodes explain the steps of an algorithm using simple words. It is not a computer program but helps to show how the program will work.

### REVIEWING KEY CONCEPTS

#### Fundamentals of Control Principles

##### Feedback and Non-Feedback Loop Systems

Control systems are systems that are used to manage, direct, or control the actions of other devices or systems. An important part of how they work is called a *feedback loop*.

- **Feedback Loop Systems:** These systems use information from sensors to keep adjusting their actions, making sure the right result is achieved. An example is

an automatic washing machine. The machine uses sensors to detect the water level, temperature, and load size. Based on this information, it adjusts the water amount, washing time, and spin speed. Feedback systems are stable, accurate, and can adapt to changes.

- b. **Non-Feedback Loop Systems:** These systems follow set instructions without using real-time data. They carry out actions as planned, even if the outcome changes. An example is a garden sprinkler system that turns on and off at scheduled times. It waters the garden regardless of whether it has rained or not. Non-feedback systems are simpler but lack the flexibility and precision of feedback systems.

## Basic Principles in Automation and Robotics

### Problem-Solving Framework

To solve problems in robotics and automation, a clear and organised method is always used. First, the information needed to start the system (called inputs) is identified. Then, the steps for how the system will work (called processes) are planned. Finally, the results that the system should produce (called outputs) are decided. For example, in a simple robot that sorts fruits, the input could be the type of fruit, the process could be the robot sorting them into different baskets, and the output would be the fruits arranged correctly. This organised method helps to make systems that work well and are easy to understand.

### Algorithms, Pseudocode, and Flowcharts

To solve problems in robotics and automation, certain tools are used to plan and organise solutions.

- a. **Algorithms:** These are step-by-step plans that explain how to solve a problem. They are very important for writing computer programs and creating systems that work automatically. For example, a robot that makes juice might follow an algorithm like this: pick a fruit, wash it, cut it, and squeeze it to get juice.
- b. **Pseudocode:** Pseudocodes explain the steps of an algorithm using simple words. It is not a computer program but helps to show how the program will work. It makes it easier to understand and write the program later. For example, the pseudocode for sorting fruits could look like this: “If the fruit is red, place it in basket A. If the fruit is yellow, place it in basket B.”
- c. **Flowcharts:** Flowcharts are drawings that show the steps in an algorithm. They use symbols like arrows, circles, and rectangles to show actions and decisions. For example, a flowchart for a traffic light system could show when the light should turn red, yellow, or green.

# Introduction to Advanced Control and Feedback Systems

## Feedback Loop Systems

Feedback loop systems use real-time information from sensors to adjust what they are doing. This helps them work very accurately and handle complex tasks.

### Example 1: Autonomous (self-driving) Drones

Autonomous dDrones use feedback loop systems to stay balanced and fly safely. Sensors collect data about the drone's height, speed, and position. This information is sent to a processor, which makes decisions like whether to move up, down, or stop. The drone's motors (actuators) then carry out these decisions.

1. **Sensors:** Measure things like height, speed, and location (for example, gyroscopes, accelerometers, and GPS).
2. **Processors:** Decide what to do based on the sensor data (for example, microcontrollers or onboard computers).
3. **Actuators:** Make the drone move by changing the speed of the motors.

These systems help drones fly smoothly and respond to changes in their surroundings.

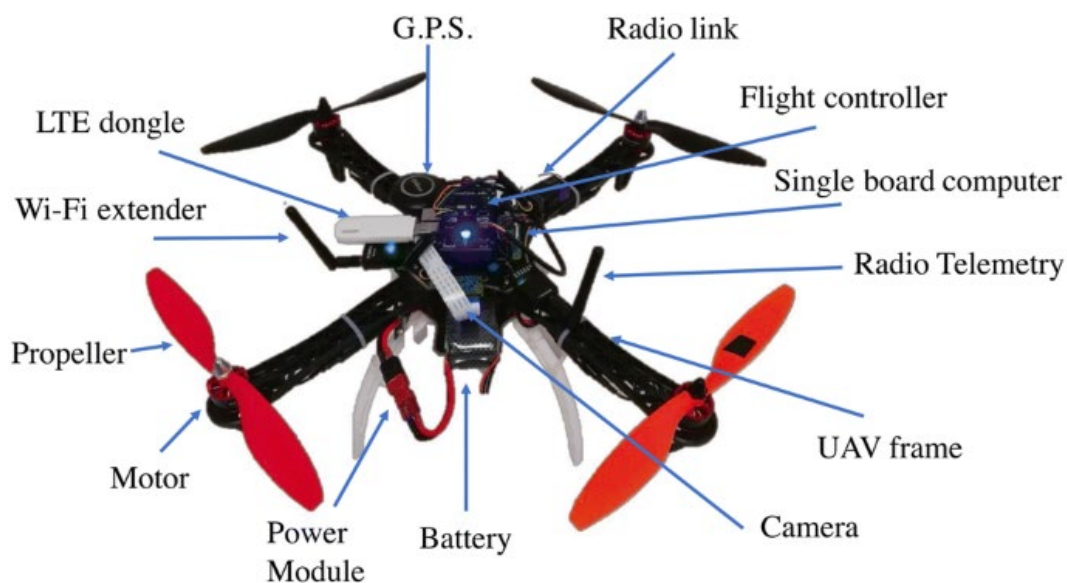


Figure 6.1: Labelled Drone


## Control Mechanism

Real-time feedback is used to help the drones respond to changes in their surroundings. For example, when a drone delivers packages, sensors are used to detect obstacles, such as trees or buildings, and to adjust its height and direction to avoid them.



Drones in busy areas use advanced systems that combine information from different sensors. These systems ensure the drone flies safely and efficiently. For instance, the drone can slow down if there is strong wind, stop if its path is blocked, or adjust its height to fly over an obstacle. By using these systems, smooth and safe flight is ensured.

Scan the QR code below or visit the internet to watch a video on how drones operate as a control system.

Link	QR Code
<a href="https://www.youtube.com/watch?v=2BwUMk10WqI">https://www.youtube.com/watch?v=2BwUMk10WqI</a>	

## Example 2: Farming Robots

Farming robots are often used to help with tasks like planting seeds, watering crops, and picking fruits. These robots work in fields where accuracy and the ability to adjust to changes are very important. A feedback loop system is used to help these robots perform their tasks properly.


Some important parts of farming robots include

1. **Sensors:** These are used to measure things like the soil's moisture level, the size of fruits, or the position of plants.
2. **Controllers:** The sensor data is used to decide how the robot should adjust its movements and actions.
3. **Actuators:** These make the robot carry out the adjustments, like watering dry soil or gently picking ripe fruits.

## Control Mechanism

Continuous feedback is used to help the robot stay accurate. For example, the robot can change how much water it sprays based on the soil's moisture or adjust its grip to pick fruits without damaging them. This helps the robot work efficiently and adapt to different farming conditions.

Scan the QR code below or search on the internet to watch a video on farm robotics and automation.

Link	QR Code
<a href="https://www.youtube.com/watch?v=4qrlFse5IUU">https://www.youtube.com/watch?v=4qrlFse5IUU</a>	

## Non-Feedback Loop Systems

Non-feedback loop systems work by following instructions that are set in advance. They do not make real-time changes but are good for simple and repetitive tasks.

### Example 1: Sprinkler Systems

In farms or gardens, sprinkler systems water plants at fixed times. These systems do not change how much water is used, even if it rains.

1. **Timer:** Decides when the sprinklers should start and stop.
2. **Pump Motor:** Moves the water through the pipes and sprinklers at a steady rate.

### Control Mechanism

The timer controls how long the sprinklers work, ensuring water is sprayed at the same time each day without checking the soil's moisture or the weather.

### Example 2: Traffic Lights

Traffic lights change colours at regular intervals to manage traffic flow. They do not adjust to the number of vehicles on the road.

1. **Programmable Timer:** Sets the duration for each light (red, yellow, and green).
2. **Bulbs and Signals:** Show the colours to guide vehicles and pedestrians.

### Control Mechanism

The timer runs a set pattern of lights, ensuring that the sequence repeats in the same way every time, keeping the process simple and predictable.

## DEVELOPING ADVANCED ALGORITHMS

Advanced algorithms are very important for helping smart machines work accurately and quickly. These algorithms process a lot of information from sensors, make decisions in real time, and control parts like motors to complete tasks properly.

## Understanding Complex Problems

Some tasks are difficult and need advanced algorithms to solve them. These problems show why smart solutions are important for systems that use control and feedback.

### Example 1: Drones Delivering Packages

A delivery drone must fly in a busy neighbourhood. It has to avoid trees, buildings, and birds while carrying packages to the right house. The drone needs to plan its path, find obstacles, and adjust its flight in real time.



## Example 2: Farm Robots Picking Fruits

Robots on a farm can be used to pick fruits. These robots need to work together to pick fruits from trees without bumping into each other. They also need to adapt if a new task is added or for instance, if a tree is taller than expected.

## Steps for Designing Advanced Algorithms

Creating an advanced algorithm involves these steps.

1. **Problem Definition:** The problem must be explained clearly, including what is needed and what the goal is.  
*Example:* A drone delivering a package must safely reach the destination without crashing into obstacles.
2. **Data Collection and Processing:** The sensors and the type of information needed must be identified. This information is then processed to make it useful.  
*Example:* A farm robot uses cameras to detect where the ripe fruits are located.
3. **Decision- Making Logic:** The rules for how the machine should act based on the data are created. These rules are made smarter using control methods or learning techniques.  
*Example:* A farm robot calculates how to pick fruits gently without damaging them.
4. **Controlling the Actions:** The instructions for motors and other parts are made based on the decisions. The actions are planned to be smooth and safe.  
*Example:* Use Proportional-Integral-Derivative (PID) controllers to make the robot move its arm carefully to pluck a fruit without shaking the tree too much.
5. **Testing and Validation:** The algorithm is tested in a safe environment, like a computer simulation, to check for mistakes. It is then tried in real situations to make sure it works well.  
*Example:* The robot's fruit-picking algorithm is tested in a model farm before being used in a real farm.

## PSEUDOCODE DEVELOPMENT

Pseudocode is a simple way of writing down the steps of a task in plain language. It acts as a bridge between how humans think and how computers work. Pseudocode makes it easier to understand and improve an algorithm before turning it into a program. This section explains how to create pseudocode by breaking down complex algorithms into easy steps.

## Understanding the Algorithm

Before writing pseudocode, it is important to understand the task completely. This means studying how the algorithm works and what steps are involved.

## Steps to Understand the Algorithm

1. **Breaking it Down (Decomposition):** The task is divided into smaller, simpler parts. This helps to clearly see all the parts of the algorithm.  
*Example:* A robot sorting items into boxes can be broken down into steps like picking up an item, checking its size, and placing it in the correct box.
2. **Identifying Inputs and Outputs:** The information needed for the algorithm (inputs) and the expected results (outputs) must be written down.  
*Example:* For the sorting robot, the inputs are the items, and the output is the sorted boxes.
3. **Key Steps Identification:** The important actions that make the task work are identified. These steps will form the main parts of the pseudocode.  
*Example:* The robot must scan each item, compare its size, and place it in the correct box.

## Writing the Pseudocode

Once the algorithm is understood, the next step is to write the pseudocode. This involves putting the key steps into simple, clear instructions.

## Steps to Write Pseudocode

1. **Plain Language Conversion:** Each step of the task is written in easy-to-understand English. This keeps the instructions clear while staying true to the logic of the algorithm.  
*Example:* “Check the size of the item” or “Move the item to the correct box.”
2. **Pseudocode Structure:** The steps are written neatly, using proper spacing and indentation to make it easy to read. Special keywords like IF, THEN, and LOOP can be used to show decisions or repeated actions.  
*Example:*  

```

IF item is small
    Place in Box A
ELSE
    Place in Box B

```
3. **Logical Flow Preservation:** The pseudocode must show how the algorithm flows, including decisions, loops, or repeated actions. Comments can be added to explain tricky parts of the logic.

*Example:*

```

LOOP through all items
    Check the size of each item
    Place in the correct box
END LOOP

```

## Example: Sorting Robot

**Task: Write pseudocode for a robot to sort items into two boxes based on size.**

### Steps

1. Input: A list of items.
2. Output: Items sorted into Box A (small) or Box B (large).
3. Actions: Check the size of each item and place it in the correct box.

### Pseudocode

```

FOR each item in the list
  IF item size is small
    Place in Box A
  ELSE
    Place in Box B
  END IF
END FOR
  
```

This pseudocode shows the task clearly and can now be used to create the program.

## CREATING DETAILED FLOWCHARTS

Flowcharts are diagrams that show the steps and decisions in a process. They are very useful in robotics and automation because they make it easier to understand how a system works. In this section, you will learn how to create detailed flowcharts for systems with complex control and feedback. You have already learnt about basic flowchart symbols in year one, so this will teach you how to use them to draw flowcharts for more advanced systems.

### How to Develop a Flowchart

Once you understand how the system works, you can start drawing your flowchart. Here are the steps to follow.

1. **High-Level Overview:** Start with a basic version of the flowchart that shows the main parts of the system. For example, if you are creating a flowchart for a traffic light system, begin by showing the steps: green light, yellow light, red light.
2. **Add More Details:** Break the system into smaller steps and add these to the flowchart. Use symbols like rectangles for actions and diamonds for decisions. For example, in the traffic light system, you can include steps like “Check if cars are present” or “Wait for pedestrians to cross.”

3. **Include Decision Points:** Add decision points to show where the system chooses between two or more options. For instance, the traffic light might check if the road is clear before changing the signal.
4. **Show Repeating Actions (Loops):** Use loops to show actions that happen repeatedly. In the traffic light system, this could be the cycle of lights changing repeatedly.
5. **Use Sub-Flowcharts for Complex Parts:** If a part of the flowchart is too detailed, draw a separate flowchart for that part and connect it to the main one. For example, you can draw a smaller flowchart just for how the traffic light handles pedestrians.

## How to Make a Good Flowchart



A good flowchart is clear and easy to understand. Follow these tips.

1. **Symbol Consistency:** Always use the same symbol for the same type of action. For example, do not use a rectangle for one action and a circle for another similar action.
2. **Clear Labelling:** Write clear and simple labels for all parts of the flowchart. For example, instead of writing “Change signal,” you can write “Change from red to green light.”
3. **Organise Neatly:** Arrange the flowchart in a way that makes it easy to follow. Draw the steps in a logical order from top to bottom or left to right.
4. **Visual Hierarchy:** Use colours or bold lines to show the most important steps. For example, you can highlight the decision points in yellow.
5. **Check and Improve:** Review your flowchart to make sure it is correct. Ask yourself if someone else could understand it easily. Make changes if necessary to improve it.

Flowcharts will help you explain how systems work and make it easier to fix problems or improve them.

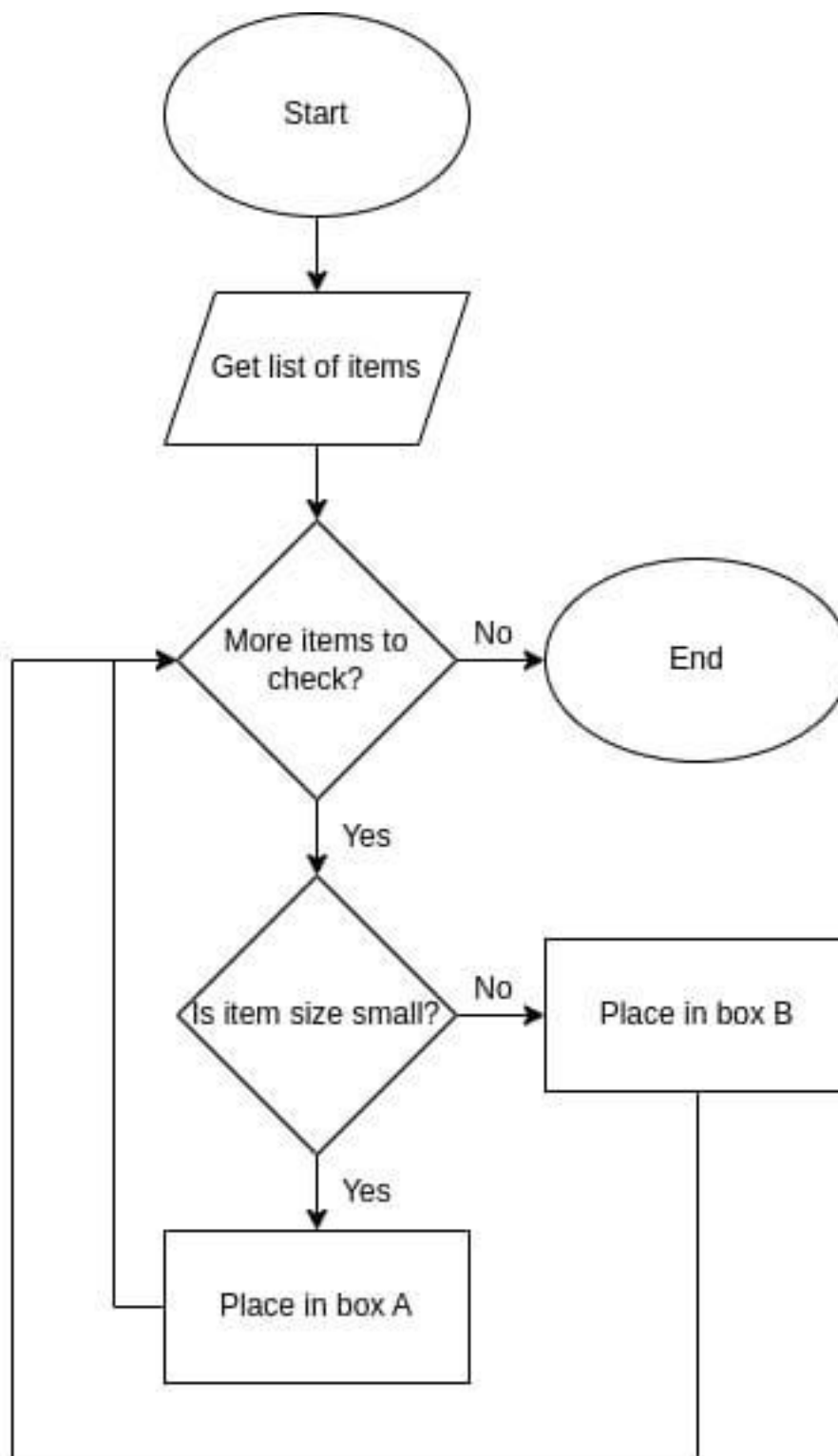
When designing flowcharts, it is helpful to use computer software to make the work neat and organised. Some examples of software that can be used include **Google Drawings**, **Canva**, **PowerPoint**, **Draw.io**, **Lucidchart**, **Microsoft Visio**, **SmartDraw** and **Paint**. These tools make it easy to draw flowcharts and include symbols, labels, and arrows.

Scan the QR codes for a tutorial on using Draw.io and Lucidchart to draw flowcharts.

LINK	QR CODE
<a href="https://www.youtube.com/watch?v=_zZczZxyXKM">https://www.youtube.com/watch?v=_zZczZxyXKM</a>	
<a href="https://www.youtube.com/watch?v=C-qnybZf09Y">https://www.youtube.com/watch?v=C-qnybZf09Y</a>	

## Example: Sorting Robot

Based on the developed algorithm and Pseudocode for the sorting robot example, the flowchart below was developed



**Figure 6.2:** Flowchart for sorting Robot Algorithm

### Activity 6.1 Designing a Smart Robot for Everyday Tasks



You must organise yourselves teacher will put you into groups and make a mat provide you with a mat that has places for obstacles and, lines for navigation for and objects on it for this activity.

1. In your group, you are to design a robot that can follow the lines on the mat from point A to B.
  - a. Write an algorithm for the robot's task
  - b. Write a pseudocode based on your algorithm.
  - c. Draw a flowchart for your proposed solution.
  - d. Present your flowchart to the class.
  - e. Build your robot and program it.
  - f. Test your robot and adjust where necessary.
2. In your group design a robot that follows lines on the mat, and can avoid obstacles as it moves from point A to B.
  - a. Write an algorithm for the robot's task
  - b. Write a pseudocode based on your algorithm.
  - c. Draw a flowchart for your proposed solution.
  - d. Present your flowchart to the class.
  - e. Build your robot and program it.
  - f. Test your robot and make adjustments where necessary.
3. In your group design a robot that can pick up an object at point A, navigate a path on the mat, avoid obstacles and deliver the object to point B
  - a. Write an algorithm for the robot's task
  - b. Write a pseudocode based on your algorithm.
  - c. Draw a flowchart for your proposed solution.
  - d. Present your flowchart to the class.
  - e. Build your robot and program it.
  - f. Test your robot and make adjustments where necessary. Avoid obstacles and move objects to specific locations determined by your teacher.

*Use the following prompts to complete the activity:*

- i. Write an algorithm for the robot's task
- ii. Write a pseudocode based on your algorithm.
- iii. Draw a flowchart for your proposed solution.
- iv. Present your flowchart to the class.
- v. Build your robot and program it.
- vi. Test your robot and adjust where necessary.

## EXTENDED READING

Resource	QR Code
<a href="https://www.youtube.com/watch?v=aYcIoxZOWkI">https://www.youtube.com/watch?v=aYcIoxZOWkI</a> How to create a flowchart using PowerpointPowerPoint	
<a href="https://www.youtube.com/watch?v=6UWac5e0tPA">https://www.youtube.com/watch?v=6UWac5e0tPA</a> What is an Algorithm	
“Everything You Need to Ace Computer Science and Coding in One Big Fat Notebook”  This reading resource provides a simplified explanation of algorithms, pseudocode and flowcharts	
“Algorithm, Pseudocode and Flowchart: Learn Algorithm in Simple Steps. For absolute beginners, ICT, KS3, GCSE, A-Level, Under Graduate, Kids, SAT & for all IT Enthusiast.”  This reading resource uses a simplified language to explain algorithms, pseudocodes and flowcharts. to young learners.	



## REVIEW QUESTIONS 6.1

1. What is a flowchart, and why is it important in robotics?
2. What is pseudocode, and how is it different from a flowchart?
3. What is the difference between a feedback loop and a non-feedback loop in robotic control systems?
4. Describe how a robot can navigate from a starting point to a target while avoiding obstacles.
5. Create a flowchart for a robot navigating from a starting point to a target point on a mat, avoiding three obstacles. Include decision points for obstacle detection.

SECTION

# 7

## ROBOT CONSTRUCTION



# ROBOT CONSTRUCTION & PROGRAMMING

## Robot Construction

### INTRODUCTION

---

In this section, you will learn how robots move and how their parts work together to make them move in a certain way. You will also understand how speed and changes in speed (velocity and acceleration) are important for robots to move carefully and do their tasks correctly. This is very helpful when designing robots that need to work in tricky places or do difficult jobs.

#### KEY IDEAS

- **Building Fun Robots:** You can build robots using kits or simple materials to create machines that move and do helpful jobs.
- **How Robots Move:** Robots can travel in straight lines, turn, or crawl, using wheels, tracks, or legs to move on different surfaces.
- **Robots Use Smart Parts:** Gears and swinging parts help robots perform tasks like picking things up or opening doors.

## UNDERSTANDING ROBOT MOVEMENT: CLOSED CHAINS, VELOCITY, AND TRAJECTORY.

### What Are Closed Chains in Robots?

Closed chains are systems in robots where the parts, called links, are connected in a loop. Because of this, when one part moves, the other parts are affected. These closed chains are different from open chains, where the parts are not connected in a loop. Closed chains are more stable and allow robots to move with great care.

Here are some examples of closed chains:

1. **A Crane with a Mechanical Arm:** A crane's arm has parts that form a loop. This helps it lift and move objects very precisely.



Figure 7.1: Crane with a mechanical arm showing a closed loop mechanism

2. **A Windshield Wiper System:** The bars (wiper arms) in a wiper are connected in a loop, making the wiper blades move back and forth smoothly.

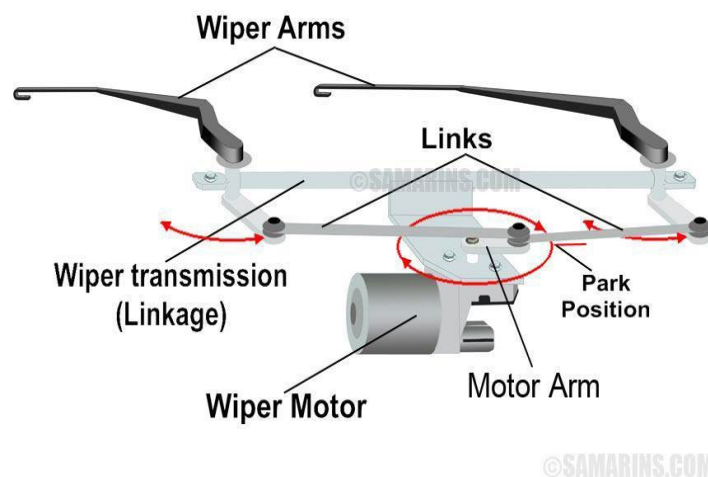


Figure 7.2: Wiper system showing closed loop mechanism [Source]

3. **A Machine That Mixes Dough:** The parts of a dough mixer form a closed loop, helping it mix the ingredients evenly.



Figure 7.3: A dough mixing machine as a closed mechanism

## Characteristics of a Closed Chains

1. **Linked Movements:** When one part moves, it makes all the other parts move in a certain way.
2. **Constraints:** The movements are limited because of how the parts are connected.
3. **Very Steady:** These systems are steady and allow for precise control.
4. **Complicated Movements:** Figuring out how the parts move together can be tricky because they all depend on one another.

## Degrees of Freedom in Closed Chains

Degrees of Freedom (DoF) are used to describe how many different ways a machine or system can move. In machines like robots, parts called **links** and **joints** are used to control these movements. The kind of joints, like ones that turn (revolute joints) or slide (prismatic joints), also affect how the machine moves.

For example:

- a. A **bicycle's pedal** can only turn in circles because the joints and links limit it to that movement.
- b. A **fan blade** spins around its centre, but it cannot move up or down.

In a robot with many joints, movements like turning, lifting, or sliding can be done. However, when the links form a loop (a closed chain), the movements are limited to certain patterns.

Think about a **pair of scissors**. The blades move in a set path because the handles and blades are connected in a loop. This makes it easier to cut paper in a straight line.

The figure below shows several 3D joints with their degrees of freedom stated within the pair of brackets.

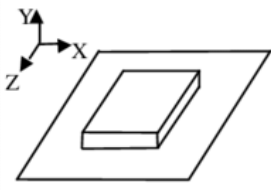
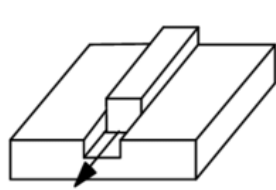
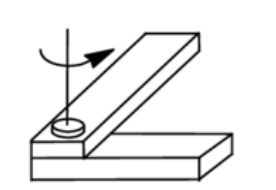


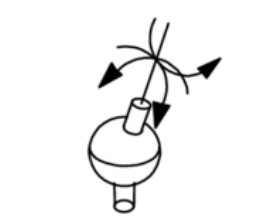
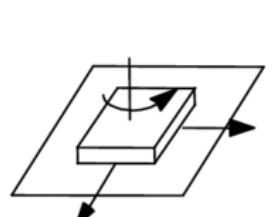
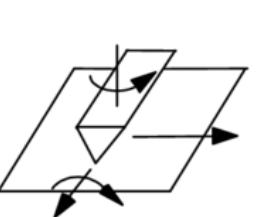
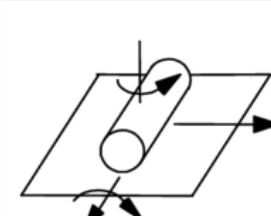
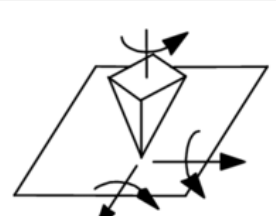
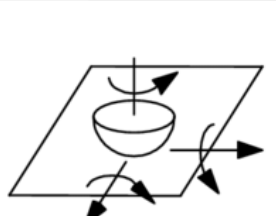
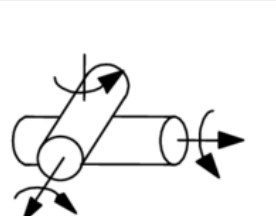
			
Rigid (no motion)	Prismatic (1)	Revolute (1)	Parallel Cylinders (2)
			
Cylindrical (2)	Spherical (3)	Planar (3)	Edge Slider (4)
			
Cylindrical Slider (4)	Point Slider (5)	Spherical Slider (5)	Crossed Cylinders (5)

Figure 7.4: 3D kinematic joint types with their degrees of freedom. [Chase et al, 1996.]

## Application of Closed Chain Mechanisms in Robots

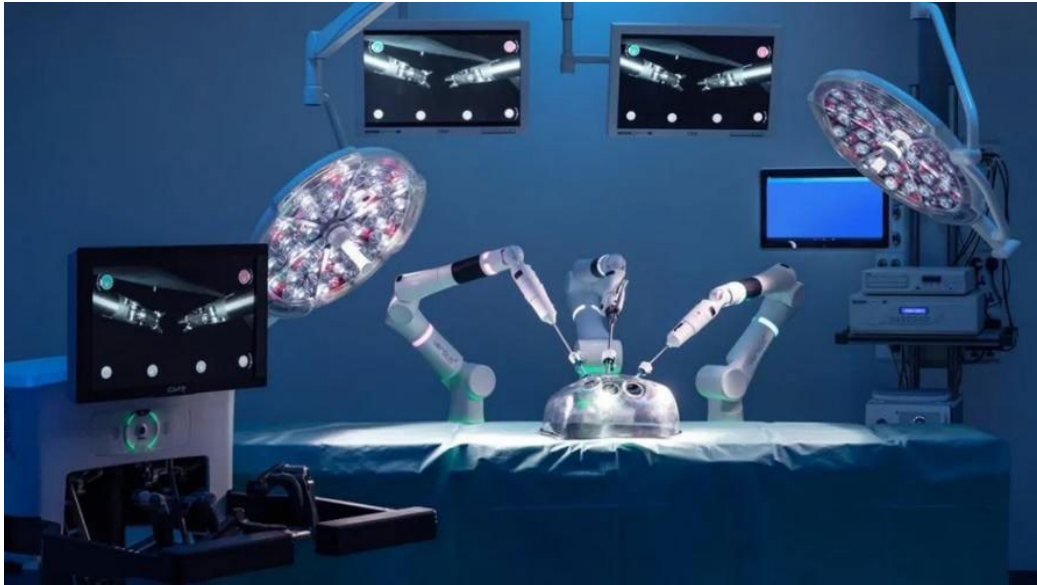
Closed chain mechanisms are used in robots for many important tasks. These tasks often need the robot to be very steady and precise. Here are some examples of where closed chain mechanisms are helpful:

### 1. Careful and Accurate Work

Closed chain mechanisms are used in robots that must do very careful tasks.

**Example:** A robot used by doctors to help in surgeries can move very precisely to perform delicate operations.





**Figure 7.5: Surgical robot using a closed chain mechanism**

## 2. Carrying Heavy Things

Because closed chains are strong and steady, they are used in robots that carry or lift heavy objects.

**Example:** A warehouse robot with a strong arm can pick up and move big boxes safely.



**Figure 7.6: Warehouse robot with a closed chain arm designed for lifting**

## 3. Training and Practice

Closed chain mechanisms are used in machines that help people learn new skills by providing realistic movements.

**Example:** A flight simulator used by pilots to practise flying a plane moves in ways that feel like a real flight.





**Figure 7.7: Flight simulator using a closed chain mechanism**

### Activity 7.1 Closed Chain Mechanisms

*For this activity you should work with a friend.*

1. Explain to your friend what is meant by a closed chain mechanical system.
2. Draw a four rigid bar linkage closed chain mechanical system. Label each of the rigid bars (links) and the connecting pivots (joints). Explain why this is a closed chain mechanical system.
3. Explain in your own words why a bicycle pedal mechanism has two degrees of freedom.
4. A robot arm has a base, a two-link arm and an actuator. Sketch the robot arm and label the linkages, connecting pivots. Explain why this robot arm has three degrees of freedom.
5. Use diagrams to explain how the movement of one link affects other parts of the robot arm mechanical system you sketched for the previous question.
6. Explain how closed chains contribute to the stability and precision of robotic systems.

## Velocity in Mechanics

Velocity shows how quickly something changes its position over time. It also tells the direction in which it is moving. For example, if a car is moving at 50 km/h towards Kumasi, its velocity is 50 km/h in that direction. Unlike speed which is a scalar quantity,

velocity always has both size (how fast) and direction which makes it a vector quantity. It is usually measured in metres per second (m/s).

## Types of Velocity

### 1. Linear Velocity

**Linear velocity** is how fast something moves along a straight path. Imagine a robot moving straight across a flat mat. The speed at which it moves forward in a straight line is its linear velocity.

#### How to Calculate Linear Velocity

The formula for linear velocity ( $v$ ) is:

$$v = \text{distance (d)} / \text{time (t)}$$

**Example:** If a robot moves 20 metres in 10 seconds, its velocity can be found like this:

$$v = 20 / 10, v = 2 \text{ m/s}$$

### 2. Angular Velocity

Angular velocity shows how quickly something turns or spins around a point or an axis. Imagine a robot turning to face a new direction. The speed at which it turns is its angular velocity.

#### How to Calculate Angular Velocity

The formula is:

$$\omega = \text{angle } (\theta) / \text{time (t)}$$

where:

- $\omega$  is the angular velocity.
- $\theta$  is the angle the object turns (measured in radians).
- $t$  is the time it takes to turn.

## How Robots Turn

When a robot turns, its wheels on either side move at different speeds.

1. The **inner wheel** travels a shorter distance.
2. The **outer wheel** travels a longer distance.

*Some important terms*

- a. **Turn Radius (r):** The radius of the circle the robot's centre follows during a turn.
- b. **Inner Wheel Radius ( $r_0$ ):** The radius of the circle that the inner wheel follows.

$$r_0 = r - d/2$$

- c. **Outer Wheel Radius ( $r_1$ ):** The radius of the circle that the outer wheel follows.

$$r_1 = r + d/2$$

Here,  $d$  is the distance between the robot's two wheels referred to as the wheelbase.

### Path Lengths During a Turn

The distance each wheel travels during a turn depends on the angle ( $\theta$  in radians) of the turn.

#### Inner Wheel Path Length ( $l_0$ ):

$$l_0 = r_0 \times \theta$$

By substituting  $r_0 = r - d/2$  into the formula above, the inner wheel path length can be found as:  $l_0 = (r - d/2) \times \theta$

#### Outer Wheel Path ( $l_1$ )

$$l_1 = r_1 \times \theta$$

By substituting  $r_1 = r + d/2$  into the formula above, the outer wheel path length can be found as:  $l_1 = (r + d/2) \times \theta$

### Converting Degrees to Radians

Turns are often measured in degrees, but for calculations, radians are used. To convert degrees to radians it is worth noting that  $\pi$  radians is equal to  $180^\circ$ :

$$\text{Angle in Radians} = \text{Angle in Degrees} \times \pi/180$$

#### Example

If a wheel turns  $90^\circ$ , convert the turn angle to radians:

$$\text{Angle in radians} = 90 \times \pi/180 = \pi/2 \approx 1.5708 \text{ rad.}$$

**Example calculation on Robot turn:** Assume a robot with a wheelbase ( $d$ ) of 1.5 metres is making a 60-degree turn with a turn radius of 2 metres. Find

- the equivalent turn angle in radians
- the radius of the inner wheel,
- the radius of the outer wheel,
- the path length each wheel needs to travel to complete the turn.

#### Solution:

- Turn angle =  $60^\circ$ , Angle in radians =  $60^\circ * \pi/180 = \pi/3 \approx 1.0472 \text{ rad.}$
- Inner wheel radius:  $r_0 = r - d/2$ ,  $r = 2\text{m}$ ,  $d = 1.5\text{m}$

$$r_0 = 2 - 1.5/2$$

$$r_0 = 1.25\text{m}$$

- Outer wheel radius:  $r_1 = r + d/2$ ,  $r = 2\text{m}$ ,  $d = 1.5\text{m}$

$$r_1 = 2 + 1.5/2$$

$$r_1 = 2.75\text{m}$$

d.

- i. Inner path length:  $l_0 = r_0 \times \theta$ ,  $r_0 = 1.25\text{m}$ ,  $\theta = \pi/3$

$$l_0 = 1.25\text{m} * \pi/3$$

$$l_0 = 1.25\text{m} * 1.0472 = 1.309\text{m}$$

- ii. Outer path length:  $l_1 = r_1 \times \theta$ ,  $r_1 = 2.75\text{m}$ ,  $\theta = \pi/3$

$$l_1 = 2.75\text{m} * \pi/3$$

$$l_1 = 2.75\text{m} * 1.0472 = 2.8798\text{m}$$

## Velocity Calculations

When a robot moves, both of its wheels rotate (angular velocity,  $\omega$ ) and push it forward (linear velocity,  $v$ ). The speed at which the robot moves forward depends on how fast its wheels are spinning and how large the wheels are.

For a robot to turn, the wheels must rotate at different speeds. The different velocities can be calculated by combining the linear and angular velocities as described below:

- a. **Inner Wheel Linear Velocity ( $v_0$ ):** This can be calculated for a wheel with radius  $r_0$  using the formula;

$$v_0 = r_0 * \omega$$

- b. **Outer Wheel Linear Velocity ( $v_1$ ):** This can be calculated for a wheel with radius  $r_1$  using the formula;

$$v_1 = r_1 * \omega$$

It can be seen from the formulas above that the linear velocity of a wheel is given by the product of the wheel's radius and the angular velocity of the wheel.

**Example:** If a wheel with a radius of 20 cm rotates at 8 rad/s, calculate its linear velocity.

**Solution**

$$\begin{aligned}\text{Linear velocity } (v) &= r \text{ (m)} * \omega \text{ (rad/s)}, r = 20 \text{ cm} = 0.2 \text{ m}, \omega = 8 \text{ rad/s} \\ v &= 0.2 * 8 = 1.6 \text{ m/s}\end{aligned}$$

This means that the robot moves with a speed of 1.6m/s in the forward direction.

## Understanding Average Motion Concepts in Robotics

Average motion, helps describe how something moves over a period. Robots often move from one place to another. To study their movement, it is important to understand three key

ideas: average position, average velocity, and average acceleration. These help to predict how a robot moves and behaves during its tasks.

## Average Position

This is the middle point between where the robot starts and where it ends.

**Formula:**

$$\text{Average position} = \text{Initial position} + \text{Final Position} \div 2$$

**Example:** A robot moves from position A = 4m to position B = 12m. Calculate the average position of the robot.

$$\text{Average position} = 4 + 12 \div 2 = 8\text{m}$$

## Average Velocity

This is how fast the robot changes its position over a period of time.

**Formula:**

$$\text{Average velocity} = \text{Displacement (d)} \div \text{Time taken (t)}$$

**Example:** A robot moves 16 metres in 2 seconds. Calculate the average velocity of the robot.

$$\text{Average velocity} = 16\text{m} \div 2\text{s} = 8\text{m/s}$$

## Average Acceleration

This is how much the robot's velocity changes over time.

**Formula:**

$$\text{Average acceleration} = \text{change in velocity } (\Delta v) \div \text{time taken (t)}$$

**Example:** A robot accelerates from a velocity of 4m/s to 8m/s in 2 seconds. Calculate the average acceleration.

$$\text{Average acceleration} = (8\text{m/s} - 4\text{m/s}) \div 2\text{s} = 2\text{m/s}^2$$

## Understanding Instantaneous Motion Concepts in Robotics

Instantaneous motion tells you the speed or position at one exact moment. It is like taking a snapshot of movement. One key difference between **average motion** and **instantaneous motion** is that, while average motion is about the whole journey or movement, instantaneous motion is about a single moment. i.e. Average motion gives a big-picture idea whereas instantaneous motion is like focusing on just one part of the journey. To understand instantaneous motion, let us take a look at three key concepts, namely, instantaneous position, instantaneous velocity and instantaneous acceleration.

## Instantaneous Position

**Position** refers to the exact location of an object at a specific time. Think of it as a point on a map.

You can We often use a number line or a graph to show position. For example:

- If a robot starts at **0 metres** and moves to **5 metres**, its position changes.
- The **instantaneous position** tells you us where the robot is at a single moment, such as at **5 seconds** into the journey.

On a graph, the position at a specific time can be found by looking at the point where the time value meets the position curve.

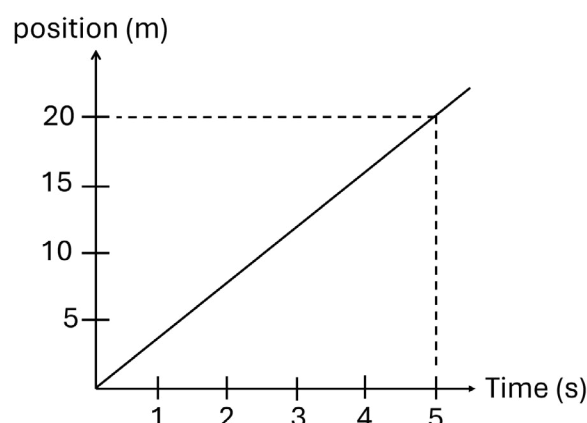


Figure 7.8: Position curve showing instantaneous position of 20m at 5s.

## Instantaneous Velocity

**Instantaneous velocity** is the velocity of an object at one specific moment in time. Imagine riding your bicycle down a hill. If you look at the speedometer when it says **15 km/hour**, this is your **instantaneous velocity** at that moment.

Instantaneous velocity is found using a graph of position versus time as shown in **Figure 7.9**.

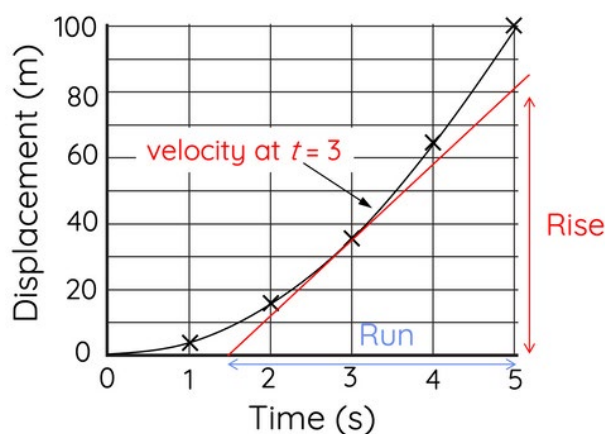


Figure 7.9: Position versus time graph showing instantaneous velocity at 3s

- The **slope** (steepness) of the curve at a specific point tells you the instantaneous velocity. This can be calculated by dividing the **Rise** as shown in the figure by the **Run** (i.e. change in y over change in x).
- If the slope is steep, the object is moving fast. If the slope is flat, the object is not moving.

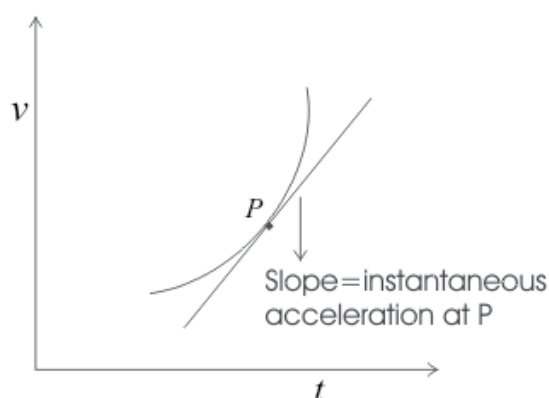
## Instantaneous Acceleration

**Instantaneous acceleration** is the acceleration of an object at a specific time. It shows how fast the velocity is changing at that moment.

Imagine a sprinter starting a race:

- At the very start, the sprinter's acceleration is high as their speed increases rapidly.
- Later, when they reach a steady speed, the instantaneous acceleration might drop to **0** because their velocity is not changing anymore.

Instantaneous acceleration is found by looking at the **slope of the velocity-time graph**:



**Figure 7.10: Velocity vs Time graph showing instantaneous acceleration**

- A steep slope means high acceleration.
- A flat line means no acceleration (constant velocity).

Resource	QR Code
<a href="https://www.youtube.com/watch?v=pfTTHx9kCHk">https://www.youtube.com/watch?v=pfTTHx9kCHk</a> A video on velocity	
<a href="https://www.youtube.com/watch?v=Ed58JZKiGEM">https://www.youtube.com/watch?v=Ed58JZKiGEM</a> A video on acceleration	



<https://www.youtube.com/watch?v=4dCrkp8qgLU&t=31s>



A video on Position, velocity and acceleration

## Understanding Wheel Rotations for Robot Turns

A two-wheeled differential robot can turn in one of two ways, namely:

1. Spin turn, where both wheels move with the same speed but move in opposite directions (i.e. one wheel moves with a velocity that is the negative of the other wheel)
2. Swing turn, where one wheel moves while the other remains stationary OR one wheel moves faster than the other thereby creating an arc.

The number of times each wheel must rotate during the turn can be calculated using the formula below where  $r$  is the radius of the wheel:

$$\text{Rotations} = \frac{\text{path length}}{2\pi r}$$

Where path length = (distance between two wheels/2) \* (robot turn angle) for a spin turn.

### Example

A two-wheeled differential robot has wheels with a radius of 14 cm. The distance between the wheels is 28 cm. If the robot makes a 90-degree turn, how many rotations does each wheel need to make for a spin turn?

$$\begin{aligned} \text{Path Length} &= d \times \theta / 2 = 0.28 \times 2.2 \\ &= 0.2199\text{m.} \end{aligned}$$

where:

$$\begin{aligned} d &= \text{distance between two wheels} = 28\text{cm} \\ &= 0.28\text{m} \\ \theta &= \text{turn angle in radians} = 90^\circ \\ &= 2\text{radians} \end{aligned}$$

$$\begin{aligned} \text{Rotations} &= \frac{\text{path length}}{2\pi r} \\ &= \frac{0.21992\pi}{2\pi \times 0.14} \\ &= 0.25 \text{ rotations} \end{aligned}$$

In simpler terms, the rotations for a spin turn can be calculated as;

$$\begin{aligned} \text{Rotations} &= \frac{360R}{\theta r} = \frac{90 \times 360}{140 \times 14} = 0.25 \text{ rotations} \\ \text{where;} \\ \theta &= \text{the turn angle in degrees} \\ R &= \text{half the distance between the two wheels} \\ r &= \text{the radius of the wheel} \end{aligned}$$

Each wheel must rotate with 0.25 rotations with the same speed but in opposite directions to make a 90-degree spin turn. How will the robot move if both wheels are given the same speed and direction as well as the same number of rotations as calculated above?

## Converting RPM to Linear Speed

### What is RPM?

RPM means **Revolutions Per Minute**. It shows how many times a wheel or motor rotates in one minute. Robots often have motors rated in RPM, and this can be used to find how fast a wheel moves forward.

### Formula for Linear Speed

The speed of a robot's wheel in metres per second (m/s) can be calculated using:

$$v = \text{RPM} \times 2\pi \times r / 60; \text{ where } r = \text{radius of the wheel and } v = \text{linear speed in m/s}$$

**Example:** A robot wheel with a **60 cm diameter** turns at a rate of **50 RPM**. Calculate the wheel's linear speed in m/s

**Solution:**

$$\text{Radius (r)} = \text{diameter}/2 = 60/2 = 30\text{cm} = 0.3\text{m}$$

$$\text{Linear speed (v)} = 50 \times 2\pi \times 0.3 / 60 = 30\pi / 60 \approx 1.57\text{m/s}$$

### Why is this Useful?

Understanding this helps to:

- Calculate how fast a robot can move.
- Decide which motor is best for a robot.
- Control a robot's speed during specific tasks.

### Trial Question

A robot with a wheelbase ( $d$ ) of 1.5 metres is makes a 45-degree turn with a turn radius ( $r$ ) of 1 metre. Find the radii of the inner wheel and the outer wheel and well as the path length each wheel needs to travel to complete the turn.

## Trajectory in Mechanics

A trajectory is the path that an object follows as it moves through space over time. For example, when a robot moves, it may follow a straight path, a curved path, or a mix of both. In robotics, planning a trajectory means deciding how the robot will move to complete its task.

## Important Concepts

1. **Position:** Where the robot is at any moment.
2. **Velocity:** How fast and in what direction the robot is moving.
3. **Acceleration:** The rate of change of velocity along the path.

## Types of Trajectories

1. Linear Trajectory (Straight Path)

Robots sometimes move in straight lines. The distance they travel can be found using this formula:

$$\text{Distance, } d = v \cdot t.$$

where  $v$  = velocity in m/s and  $t$  = time taken in seconds

**Example:** Consider a robot moving in a straight line for 15 seconds with a constant velocity of 5m/s. Calculate the distance covered.

### Solution

$$\begin{aligned} \text{Distance } d &= v \cdot t \text{ (where, } v = 5\text{m/s and } t = 15\text{s)} \\ &= 5 \cdot 15 \\ &= 75\text{m} \end{aligned}$$

2. Circular Trajectory (Circular Path)

Robots sometimes turn or follow a circular path. This involves,

- a. **Angular Velocity ( $\omega$ ):** How fast the robot is turning.

$$\omega = \frac{\theta}{t}, \text{ where } \theta = \text{angle of the turn in radians, } t = \text{time taken in seconds}$$

- a. **Linear Velocity ( $v$ ):** How fast the robot moves along the circle.

$$v = r \cdot \omega, \text{ where } r = \text{wheel radius in metres, } \omega = \text{angular velocity in rad/s}$$

**Example:** A robot needs to move from point A to point B in a straight line, then make a 90-degree turn, and proceed to point C.

- a. Calculate the velocity required to cover the 20-metre distance from point A to B in 4 seconds.
- b. Calculate the velocity of turning 90 degrees at B if it takes 4 seconds to complete the turn with radius of 2 metres.
- c. Calculate the distance from B to C if the robot will need to travel in a straight line for 4 seconds with a constant velocity of 2.5 m/s to complete it.

### Solution

- a. Since the distance from A to B is a straight line, the linear velocity will be calculated as:

$$v = \text{distance/time} = 20\text{m}/4\text{s} = 5\text{m/s}$$

- b. Since the robot turns, the velocity to be calculated is the angular velocity.  
The turn is **90 degrees**, which is  $\pi/2$  radians, and it takes **4 seconds** to complete.

$$t = 2\text{radians}/4\text{s} = 8\text{rad/s}$$

- c. Since the movement from B to C is a linear trajectory, the distance can be calculated as:

$$d = v \cdot t = 2.5\text{m/s} \cdot 4\text{s} = 10\text{m}$$

These calculations help you to:

- a. **Predict Future Positions of the Robot:** The robot's future position can be estimated by using its current speed and the direction it is moving in. For example, if a robot moves in a straight line at a speed of 2 metres per second, you can calculate how far it will be in 5 seconds.
  - b. **Control the Robot's Movement:** Robots can avoid obstacles and follow paths by adjusting their speed. For example, if a robot is about to hit a wall, its speed can be reduced, or it can turn to avoid the wall.
  - c. **Plan the Robot's Path:** A robot can calculate the best path to reach a target. It looks at its own movements and the space around it. For example, if there is an obstacle in its way, the robot can choose a different route that avoids the obstacle but still gets to the target.
  - d. **Perform Precise Tasks:** Robots are used for jobs where accuracy is important, like placing items in exact positions. For example, a robot in a factory may need to pick up boxes and arrange them neatly on a shelf.
3. **Polynomial Trajectory:** This is a smooth movement described by polynomial equations. It allows a robot to move gently between two points. For example, if a robotic arm needs to pick up a ball and place it in a box, it may use a polynomial path to avoid knocking into anything.
  4. **Other Types of Trajectories**
    - a. **Elliptical Trajectory:** This path looks like an oval. It is useful for objects or robots that need to move in circular patterns. For example, a toy car moving around a playground in a loop could follow an elliptical trajectory.
    - b. **Spline Trajectory:** This path is smooth and flexible, often used when robots need to make very precise movements. For example, in a workshop, a machine may cut wood along a smooth, curved line using this type of path.

In summary we have discussed 5 types of trajectories or paths, namely: **Linear, circular, polynomial, elliptical** and **spline trajectories**. Robots need these different types of paths for different jobs. By planning carefully, they can avoid obstacles, follow smooth routes, and complete tasks quickly and accurately.

### Activity 7.2 Planning and Understanding Robot Movements

Organise yourselves into small groups. Your teacher will put you in groups of 3-5 persons. You will need a long tape measure for this activity. and provide you with materials for this activity.

In your group, use the tape measure provided by your teacher to measure the length and width of your classroom and note it.

1. Calculate how fast a two-wheeled differential robot has to move from one end of the classroom to the other in 20 seconds if it is to move in a straight line.
2. If the robot is to make a 90-degree turn to prevent it from hitting a wall, state what type of velocity you would have to calculate for the robot and calculate how fast the robot will have to move to make the turn in 5 seconds
3. Calculate how much time it will take the robot to move across the width of the classroom if it moves at 10m/s.
4. Present your findings to the class and explain why you used the formulas you chose for the respective velocities you found.

### Activity 7.3 Understanding Instantaneous and average motion

In the same groups from **Activity 7.2**, explain in your own words the difference between average and instantaneous motion.

Imagine a sprinter running a 60-metre race. She starts very quickly, slows down in the middle, and then speeds up again at the end. At different times, her speeds are:

1. At 2 seconds: 4 m/s
2. At 4 seconds: 3 m/s
3. At 6 seconds: 5 m/s

What is her average speed for the whole race?

What was her instantaneous speed at 4 seconds?

#### Plot the Motion on a Graph Sheet

- a. On the horizontal axis (x-axis), write the time (seconds).
- b. On the vertical axis (y-axis), write the speed (m/s).
- c. Plot the speeds at 2 seconds, 4 seconds, and 6 seconds.

#### Measure Your Robot's Motion: Build and program a two-wheeled robot to move forward with some speed

- a. Use a stopwatch and measure how far the robot can move in 10 seconds.
- b. Write down the time and distance, and distance and calculate the average speed.

- c. Put light obstacles at different positions in the robot's path to slow the robot down occasionally. Record the speed at specific points (like at 3 seconds, 6 seconds, and 9 seconds). Write these as your instantaneous speeds.
- d. In your group, discuss the following and present your findings to the class:
  - i. What do you notice about how speed changes over time?
  - ii. How is average motion different from instantaneous motion in the sprinter's story and your robot's activity?

## PERFORMING BASIC CALCULATIONS INVOLVING VELOCITY AND TRAJECTORY MOTIONS AND APPLYING TRAJECTORY CALCULATION TO ROBOT NAVIGATIONS

This section will help you learn how to calculate and understand velocity and trajectory. These ideas are important for predicting and controlling how robots move and can be used in real-life activities, like guiding a robot to follow a path.

### Analysing Velocity and Trajectory

**Direction of Velocity:** The direction of velocity at any moment shows where an object is going. Velocity is a quantity that has both size (speed) and direction. At any given time, the direction of velocity is **tangent to the object's path**, meaning it points along the curve or straight line the object is following at that moment.

For example, imagine a car driving around a bend. At every point on the bend, the car's velocity points in the exact direction it is moving at that time – this direction is along the curve of the road.

You can also think of what happens when one throws a stone in a curved path through the air. At any moment, the direction of velocity is along the path the stone is following. This is why a stone curves smoothly through the air.

This understanding helps in robotics when designing paths for robots, as knowing the direction of velocity allows the robot to move smoothly along its intended trajectory.

**Direction of Acceleration:** Acceleration describes how quickly something changes its speed. It is not just about speeding up; it can also involve slowing down or changing direction. It is a vector quantity, which means it has both size (how much change) and direction (where the change is happening). It represents the rate of change in velocity.

1. **Acceleration in the Same Direction as Velocity:** When something is going faster, it is said to be accelerating in the same direction as its velocity. For example, if a bicycle

is pedalled faster down a hill, the acceleration is in the same direction as the movement of the bicycle.

1. **Acceleration in the Opposite Direction as Velocity:** When something is slowing down, it is said to be accelerating in the opposite direction to its velocity. This is often called deceleration. For instance, when a car is stopping at a traffic light, it is slowing down, so the acceleration is opposite to its direction of travel.
2. **Acceleration Perpendicular to Speed:** Sometimes, acceleration can happen without changing the speed but by changing the direction. This can be seen when a person swings in a circle on a merry-go-round. Even if the person is moving at a steady speed, the acceleration is directed towards the centre of the merry-go-round. This type of acceleration is known as centripetal acceleration.

Imagine a toy car moving in a straight line and suddenly stopping. The deceleration points in the direction opposite to the toy car's motion. If the car moves in a circle, the acceleration always points towards the centre of the circle. In summary, the direction of acceleration indicates how the velocity is changing. It can cause an object to speed up, slow down, or change direction.

## Equations of Motion

Equations of motion are important tools used to understand how objects move. They help you see the connection between an object's position, speed, and acceleration over time. These equations are especially useful in robotics, where they help in predicting how robots will move.

### Linear Motion Equations

1. **Position as a Function of Time:** This equation shows how far an object is from its starting point after a certain time. The formula used is:

$$p(t) = p_0 + v_0 t + \frac{1}{2} at^2, \text{ where } p(t) \text{ is the position at a given time } t, p_0 \text{ is the initial or starting position,}$$

$v_0$  is the initial or starting velocity  $a$  is the constant acceleration.

2. **Velocity as a Function of Time:** This equation tells you how fast an object is moving at a certain time. The formula is:

$$v(t) = v_0 + at, \text{ where } v(t) \text{ is the velocity at time } t.$$

### Example Scenarios

**Scenario 1:** Imagine a robot starting from a stop at the entrance of a warehouse. The robot begins at position  $p_0 = 0$  metres and starts moving with no initial speed ( $v_0 = 0$ ). The robot accelerates at a steady rate of  $0.5 \text{ m/s}^2$  to deliver a package. Where will the robot be after 15 seconds.



**Solution**

Initial position, ( $p_0$ ) = 0m

Initial velocity, ( $v_0$ ) = 0m/s

Acceleration, ( $a$ ) = 0.5 m/s<sup>2</sup>

Time, ( $t$ ) = 15s

Using the formula for position:  $p(t) = p_0 + v_0t + \frac{1}{2}at^2$ ,

$$\begin{aligned} p(15) &= 0 + 0*15 + \frac{1}{2} * 0.5 * 15^2 \\ &= \frac{1}{2} * 0.5 * 225 \\ &= 56.25\text{m} \end{aligned}$$

After 15 seconds, the robot will be 56.25 metres away from where it started.

**Scenario 2:** A robot is to be built and programmed to move from one point on a mat to another. Assume it starts at point A with no speed ( $= 0$ ) and accelerates at a steady rate of 0.5 m/s<sup>2</sup> as it moves to pick up an object. How fast will the robot be moving after 10 seconds?

**Solution**

Initial velocity, ( $v_0$ ) = 0m/s

Acceleration, ( $a$ ) = 0.5 m/s<sup>2</sup>

Time, ( $t$ ) = 10s

Using the formula for velocity:  $v(t) = v_0 + at$

$$\begin{aligned} v(10) &= 0 + 0.5*10 \\ &= 5\text{m/s} \end{aligned}$$

After 10 seconds, the robot will be moving at a speed of 5 metres per second.

**Application in Robotics**

Understanding these calculations is very important for robots. Here are some ways they help.

- Path Planning:** Robots need to know the best way to move so they can reach their destination safely and quickly.
- Obstacle Avoidance:** Robots must calculate how to slow down (decelerate) or speed up (accelerate) to avoid crashing into things in their way.
- Motion Control:** It is necessary to control how fast a robot moves by setting specific velocities and accelerations.
- Pick-and-Place Operations:** When robots need to pick up or drop off objects, it is important to know the right speed to avoid dropping or breaking anything.

## Angular Motion Equations

Angular motion refers to how objects rotate. Just like you can track how far something moves in a straight line, you can also track how far something rotates. There are special equations that help you understand this motion.

1. **Angular Position as a Function of Time:** This equation shows the angle of an object at a certain time. The formula is:  

$$\theta(t) = \theta_0 + \omega_0 t + \frac{1}{2} a t^2$$
 where  $\theta(t)$  is the angular position at time  $t$ ,  $\theta_0$  is the initial angular position,  $\omega_0$  is the initial angular velocity, and  $a$  is the constant angular acceleration.
2. **Angular Velocity as a Function of Time:** This equation tells you how fast an object is rotating at a certain time. The formula is:  

$$\omega(t) = \omega_0 + a t$$
 where  $\omega(t)$  is the angular velocity at time  $t$ .

## Example Scenarios

**Scenario 1:** Imagine a robot that needs to move around a factory to inspect various sections. The wheel of the robot starts at an angle of 45 degrees ( $\theta_0 = 45^\circ$ ) and has an initial angular velocity of  $20^\circ/\text{s}$  ( $\omega_0 = 20^\circ/\text{s}$ ). The wheel accelerates at a constant rate of  $4^\circ/\text{s}^2$  ( $a = 4^\circ/\text{s}^2$ ). Find the angular position of the wheel after 10 seconds.

### Solution

Initial angle ( $\theta_0$ ) =  $45^\circ$

Initial angular velocity ( $\omega_0$ ) =  $20^\circ/\text{s}$

Angular acceleration ( $a$ ) =  $4^\circ/\text{s}^2$

Time ( $t$ ) = 10s

Using the formula for angular position:  $\theta(t) = \theta_0 + \omega_0 t + \frac{1}{2} a t^2$ ,

$$\begin{aligned}\theta(10) &= 45 + 20 * 10 + \frac{1}{2} * 4 * 10^2 \\ &= 445^\circ\end{aligned}$$

Therefore, after 10 seconds, the angular position of the robot's wheel will be  $445^\circ$

### Note

Because the answer is greater than 360 degrees the wheel has rotated more than one full revolution.

**Scenario 2:** Imagine a robot in a warehouse that starts rotating from an initial angular position of  $0^\circ$  ( $\theta_0 = 0^\circ$ ) with an initial angular velocity of  $10^\circ/\text{s}$  ( $\omega_0 = 10^\circ/\text{s}$ ). The wheel accelerates at a constant rate of  $5^\circ/\text{s}^2$  ( $a = 5^\circ/\text{s}^2$ ). Find the angular position of the wheel after 5 seconds.

### Solution

Initial angular position ( $\theta_0$ ) =  $0^\circ$

Initial angular velocity ( $\omega_0$ ) =  $10^\circ/\text{s}$

Angular acceleration ( $a$ ) =  $5^\circ/\text{s}^2$

Time ( $t$ ) = 5s

$$\begin{aligned}
 \text{Using the formula for angular position: } \theta(t) &= \theta_0 + \omega_0 t + \frac{1}{2} at^2, \\
 \theta(5) &= 0 + 10 \cdot 5 + \frac{1}{2} \cdot 5 \cdot 5^2 \\
 &= 50 + 62.505 \\
 &= 112.5^\circ
 \end{aligned}$$

So, after 5 seconds, the angular position of the rotating base will be **112.5°**.

### Application in Robotics

Understanding angular position and velocity of rotating parts of a robot is very important for several reasons:

- a. **Precision Sorting:** Robotic arms need to position themselves accurately to sort packages correctly. This means knowing exactly where to move to pick up and place items.
- b. **Synchronisation:** The movement of the robot's rotating base must be coordinated (in sync) with other parts. This helps improve the efficiency of sorting tasks.
- c. **Control Systems:** Control systems are designed to manage how the robot's base turns. This ensures smooth and reliable movements during operations.

In a warehouse, knowing the angular position of the robotic arm allows programmers to set the right angles for sorting tasks. This results in efficient and accurate operations.

## Applying Trajectory Calculations to Robot Navigation

Using trajectory calculations can greatly benefit how robots navigate. Here are some advantages:

1. **Path Prediction:** Calculating velocity and trajectory helps predict where a robot will go next. For example, this can show the path a robot will take based on its current speed and direction.
2. **Motion Control:** Trajectory calculations help adjust the robot's velocity and trajectory. This ensures the robot can navigate around obstacles and reach its destination accurately. For instance, if a robot needs to avoid an object while moving towards a target, trajectory calculations can help it change direction smoothly.
3. **Robotic Arm Movement:** These calculations also help determine how fast and in what direction a robotic arm should move to pick up objects precisely.
4. **Mobile Robot Navigation:** Trajectory calculations assist in planning and executing paths for mobile robots. This makes their routes more efficient and safer.

# Planning and Controlling Trajectories in Robotic Systems

## Trajectory Planning

Trajectory planning involves deciding the best path for a robot to follow. This includes considering the starting and ending positions and any obstacles along the way. Here are the steps to follow:

1. **Define the initial and final positions:** Decide where the robot starts and where it needs to go.  
*Example:* A robot begins at *point A* in a classroom and needs to pick up an eraser from *point B*.
2. **Choose the type of trajectory:** A robot's path can be straight (linear), curved (circular), or a mix of both.  
*Example:* The robot moves in a straight line to the middle of the room, then follows a curved path to the eraser.
3. **Determine intermediate waypoints if needed:** Sometimes, the robot may need to stop or turn at certain points.  
*Example:* The robot stops at a chair before reaching the eraser to avoid bumping into it.
4. **Set Speed and Safety Rules:** The robot should move at a safe speed and start or stop slowly to avoid falling over.  
*Example:* The robot moves slowly when picking up the eraser to avoid dropping it.

## Controlling a Robot's Movement Path

After planning how the robot should move, the next step is controlling its movement. This means making sure the robot follows the planned path correctly while adjusting for unexpected changes in its surroundings. Here is a simple explanation of how it works:

### Steps to Control a Robot's Movement Path

1. **Making the Robot Move**
  - a. **Linear Movement:** The robot uses its wheels or motors to move in a straight line. For example, it moves from point A to point X on a classroom floor.
  - b. **Turning:** At point X, the robot uses its wheels to turn towards point B, following a curved (circular) path.
2. **Feedback Mechanism**
  - a. **Sensors:** Special tools like cameras, gyroscopes (to measure turning angles), or encoders (to count wheel rotations) are used to check the robot's position, orientation and surroundings.
  - b. **Real-Time Adjustments:** If the robot is slightly off its path, the sensors help it correct itself, so it stays on track.

### 3. Task Execution

- a. **Object Detection:** When the robot reaches point B, its sensors help it detect objects and the actuators help pick it up (e.g., a pencil).
- b. **Returning Safely:** After picking up the object, the robot moves back to point A using the same planned path. It slows down and turns carefully to avoid dropping the object.

### 4. Error Handling

- a. **Unexpected Obstacles:** If the robot finds something blocking its path (like a book), it recalculates its path to go around the obstacle.
- a. **Staying on Course:** If the robot veers off the path by mistake, it uses its sensors to find the correct direction and adjusts its movements.

## How the Robot Does This

**Step 1: Trajectory Planning:** The robot knows where to start, where to end, and how to move in between (straight or curved paths).

**Step 2: Real-Time Control:** Sensors help it check its position and make changes instantly.

**Step 3: Smooth Navigation:** The robot adjusts its speed, turns, and path to complete its task without problems.

### Activity 7.4 Robot Starting from Rest

**Organise** Your teacher will put yourselves into small groups of 3-5 for this activity. In your group, attempt the following problems. Use the following guidelines to undertake this activity.

1. Understand the problem
2. Select the appropriate formula
3. Extract the data from the problem and plug it into the formula
4. Calculate for the answer.

#### Problems:

- a. A delivery robot begins from rest at a gate and speeds up by  $0.5 \text{ m/s}^2$ . Calculate how far it moves after 10 seconds.
- b. If the robot must move from **A** to **B** which is 10 metres away in 5 seconds, calculate the velocity required.
- c. At **B**, the robot turns 90 degrees in a circular path (radius = 1 metre, time = 3 seconds). Calculate angular velocity.
- d. The robot moves 1.5 metres from **B** to **C** in 0.5 seconds. Find the velocity.
- e. At **C**, it turns 135 degrees with a radius of 0.3 metres in 1 second. Calculate angular velocity.

- f. From **C** to **D**, it moves at 0.8 m/s for 0.75 seconds. Find how far it travels.
- g. Use a toy car to visualise the motion. Discuss what happens when the robot moves faster, turns sharper, or faces obstacles.

### Trial Question

A robot on wheels needs to move on a mat to pick up a small box from a position at coordinates (0.5, 0.5) metres and place it on a platform at coordinates (1, 0.5) metres.

Plan the trajectory the robot will follow, considering factors like type of path (linear, circular), velocity, and acceleration.

#### Guide:

*Step 1:* Define the initial position and the final position.

*Step 2:* Choose a linear trajectory for simplicity.

*Step 3:* Calculate the waypoints and ensure the trajectory meets constraints (e.g., maximum velocity of 0.2 m/s, maximum acceleration of 0.1 m/s<sup>2</sup>).

*Step 4:* Use a linear interpolation formula to determine intermediate positions along the path.

## CREATING ROBOTS USING ROBOTIC KITS AND LOCAL MATERIALS

In this lesson, you will learn how to design and create amazing robots using kits and materials found around you. You will explore how robots work by learning about important mechanical parts, such as gears, swinging, and moving back and forth (reciprocating). You will also make vehicles with wheels or crawlers and robots that can move without using tyres. By working on fun projects like shooting robots, automatic doors, and wind machines, you will gain skills and understand how robots are built and controlled.

### Advanced Gear Systems

#### Gear Trains

Gear trains are groups of gears that work together to move energy from one part of a machine to another. They can make things spin faster or slower and can even change the direction of the spin. For example, they are used in clocks to control how the hands move.

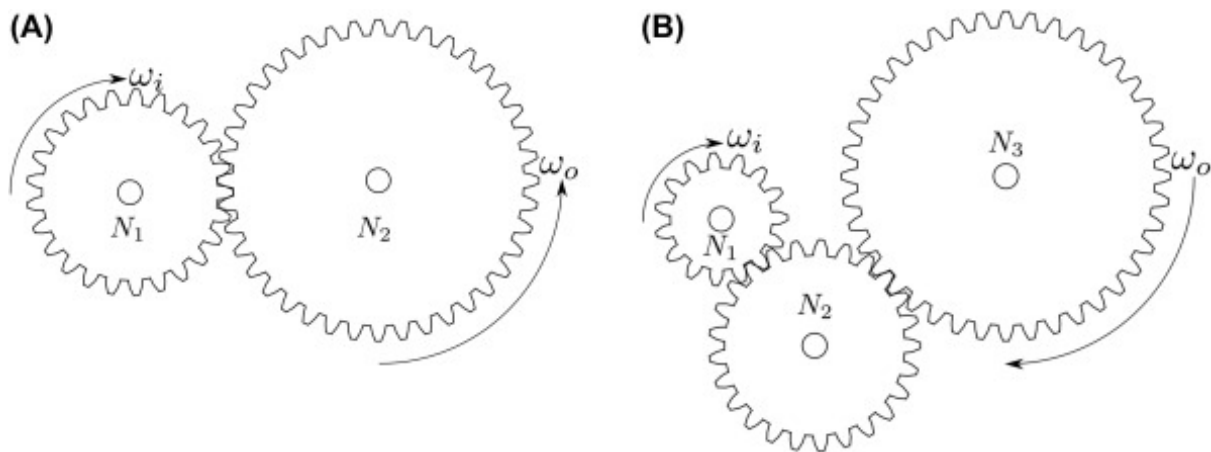


Figure 7.11: Gear trains

## Compound Gear Trains

Compound gear trains use more than one gear on the same shaft, which helps to create special gear setups for controlling speed or force. Two or more gears are connected on different shafts and are linked together. This setup allows for more choices in how fast or slow something moves and is compact, saving space in machines.

*Applications in Robotics:*

- In robotic arms, they are used to move parts very accurately.
- In wheels, they help control how fast or slow the robot moves.
- In machines, they are helpful for creating strong turning power (torque).

## Gear Ratio

A gear ratio shows how two gears work together to change speed or strength (force). It compares the number of teeth on one gear to the number of teeth on another gear it is connected to.

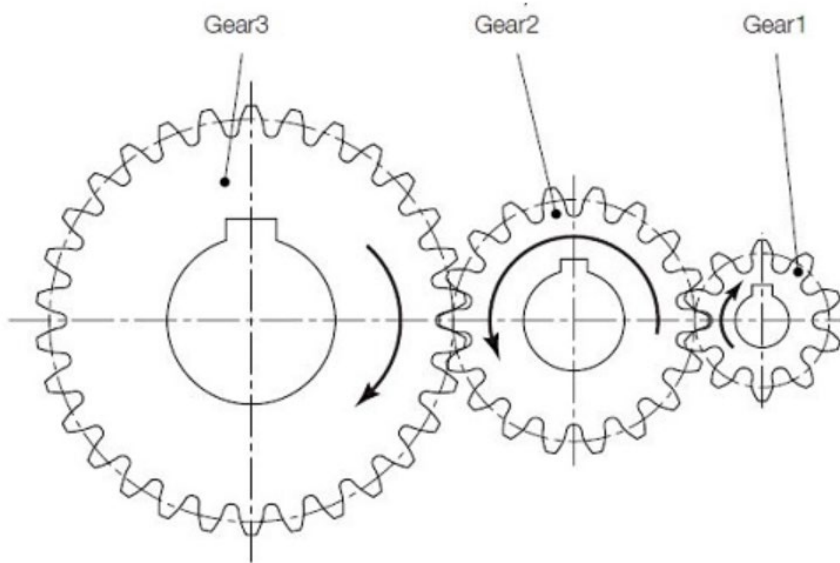
*How It Works*

- If a big gear with 40 teeth connects to a smaller gear with 20 teeth, the smaller gear will spin faster because it has fewer teeth.
- This is called a **2:1 gear ratio**, meaning the big gear turns once for every two turns of the smaller gear.

**Figure 7.12** shows a gear train with three gears:

- Input Gear (Gear 1): 10 teeth
- Intermediate Gear (Gear 2): 18 teeth
- Output Gear (Gear 3): 30 teeth





**Figure 7.12: Compound gear train with 3 gears**

**To find the total gear ratio, the gear ratio for each pair of gears must be calculated**

**Gear 1 to Gear 2:** The gear ratio is found by dividing the number of teeth on Gear 2 by the number of teeth on Gear 1:

Gear Ratio = Teeth on Gear 2 ÷ Teeth on Gear 1

Gear Ratio =  $18 \div 10 = 1.8$

**Gear 2 to Gear 3:** Gear Ratio = Teeth on Gear 3 ÷ Teeth on Gear 2

Gear Ratio =  $30 \div 18 = 1.67$

## Overall Gear Ratio

To find the overall gear ratio, multiply the two gear ratios:

Overall Gear Ratio = Gear Ratio 1 × Gear Ratio 2

Overall Gear Ratio =  $1.8 \times 1.67 = 3$  (rounded)

*What Does It Mean?*

The overall gear ratio is 3:1. This means Gear 1 (the input gear) turns 3 times for every 1 turn of Gear 3 (the output gear). The output gear moves slower but with more strength.

## Types of Gear Trains

### Planetary Gears

A planetary gear train is a special gear system made up of a central gear called the sun gear and smaller gears called planet gears that move around it. These planet gears are attached to a carrier that helps them rotate around the sun gear. The planet gears also work with an outer gear called the ring gear.

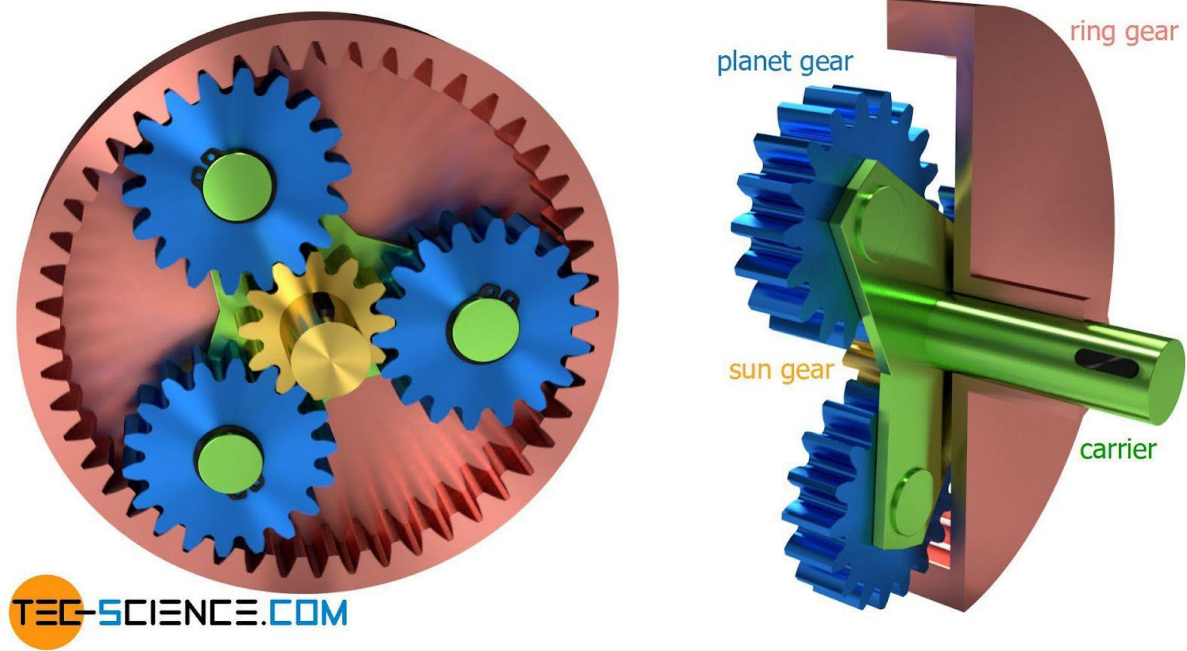


Figure 7.13: Planetary gear

### Planetary Gear Sets

This system includes:

- a. Sun Gear: The central gear.
- b. Planet Gears: The smaller gears that rotate around the sun gear.
- c. Ring Gear: The outer gear that the planet gears mesh with.

### Advantages

- a. High Torque Density: These gears can produce a lot of power without taking up much space.
- b. Compact Size: They fit well in small machines.
- c. Multiple Gear Ratios: Different speeds can be set by changing the arrangement of gears.
- d. High Efficiency: They work well without wasting much energy.
- e. Balanced Load Distribution: The weight is spread evenly across the gears.

### Applications in Robotics

Planetary gears are used in robots for

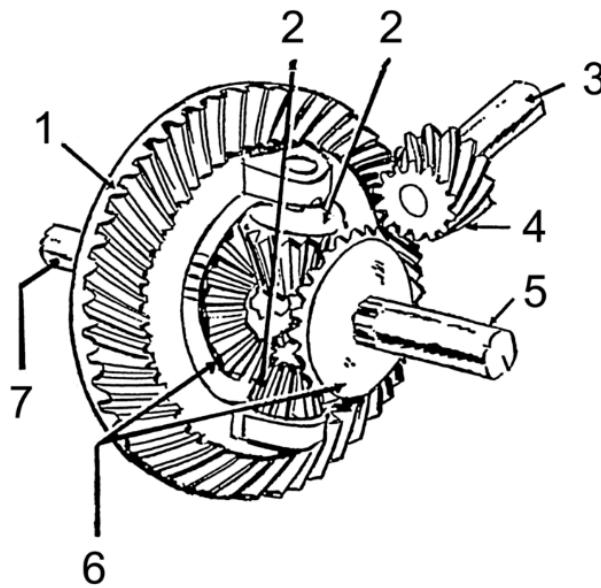
1. Mobility: Helping robots move smoothly.
2. Robotic Arms: Providing precision and strength for lifting and moving objects.
3. Compact Actuators: Making small devices that need to fit in tight spaces.

### Example Configurations

- Simple Planetary Gear Set: The sun gear drives the planet gears, which move within the ring gear. The speed depends on the sizes of the gears.
- Compound Planetary Gear Set: This has multiple sets of planetary gears for a wider range of speed and power adjustments.

## Differentials

A differential is a special gear system that allows power to move from one place to another at different angles. This system is important in vehicles, as it lets the wheels turn at different speeds when going around corners.



**Figure 7.14: Differential Gear [(1) Ring gear, (2) Pinions, (3) Pinion shaft, (4) Drive pinion, (5) Right axle, (6) Side gears, (7) Left axle]**

### Functionality and Applications in Robotics

The differential gear system splits power evenly between two outputs, which allows them to turn at different speeds. This is very important for vehicles when they turn, as the inside wheels and outside wheels need to move differently. Typically, bevel gears are used in this system.

## Advantages

- Smooth Turning: The gears allow wheels to rotate at different speeds, making it easier to turn.
- Efficient Torque Distribution: Power is shared well between the wheels.
- Improved Manoeuvrability: The robot can turn and move more easily.

### Example Applications

- Mobile Robot with Differential Drive:** In this type of robot, each wheel is powered by its own motor. The differential ensures that when the robot turns, the wheels can rotate at different speeds to keep it stable.
- Articulated Robot:** Differential gears are used in the joints of these robots to help them move smoothly and work together. This makes it easier for them to perform complex tasks accurately.

## Swinging Mechanisms

Swinging mechanisms are mechanical systems that use a rotating element to create a back-and-forth motion. This type of motion can be used for various purposes, such as moving objects, creating a pendulum effect, or even generating power.

### Types of Swinging Mechanisms

There are several types of swinging mechanisms, but some of the most common include:

#### 1. Pendulum

A pendulum consists of a weight, known as a bob, that is hung from a fixed point. The bob is free to swing back and forth due to gravity. The time it takes for the pendulum to complete one full swing is called the period of oscillation. This time depends on the length of the string or rod and the gravitational force.

#### Uses of Pendulums

- Timing Devices:** Pendulums are used in clocks because they move in a regular and predictable way.
- Sensing and Measurement:** Pendulums are found in devices like accelerometers and seismometers, which are used to detect motion and vibrations.
- Robotic Applications:** Pendulums are used in robots for balance and stability, helping robotic arms or walking robots stay upright.

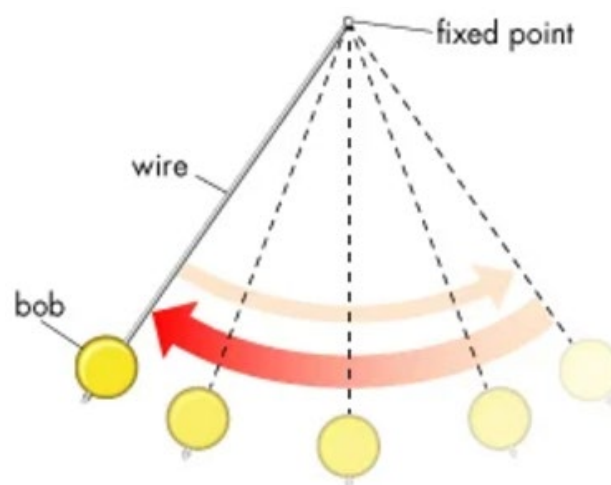


Figure 7.15: A Pendulum

## 2. Crank and Rocker Mechanism

A crank and rocker mechanism are a system that changes rotating (circular) motion into back-and-forth (linear) motion. It uses a crank (a rotating handle or wheel) connected to a rod. The other end of the rod is attached to a rocker arm, which swings back and forth as the crank turns.

### How it Works

1. The crank rotates continuously.
2. The connecting rod moves because of the crank.
3. The rocker arm swings back and forth in response to the rod's motion.

### Uses of the Crank and Rocker Mechanism

- a. **Engines:** Used in car engines to change the motion of the pistons into the rotation of the crankshaft.
- b. **Windshield Wipers:** Converts the motor's rotation into the back-and-forth motion of the wipers.
- c. **Robotic Movements:** Helps robotic arms or joints move in controlled ways, like gripping or waving.

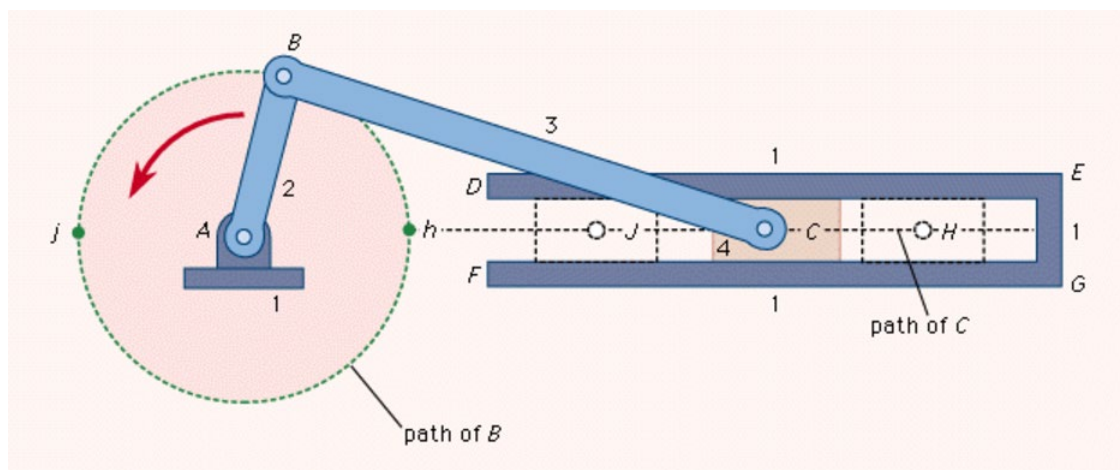


Figure 7.16: Crank Mechanism

## 3. Cam Mechanism

A cam mechanism is a system that changes rotating motion into straight or stop-and-go movement. It uses a cam, which is a rotating object with a shaped surface. As the cam turns, it pushes a follower, making the follower move in a specific way.

### Where it is Used

- a. **Engines:** Cams control the opening and closing of valves in car engines.
- b. **Automatic Machines:** Used in sewing machines to move parts in precise patterns.

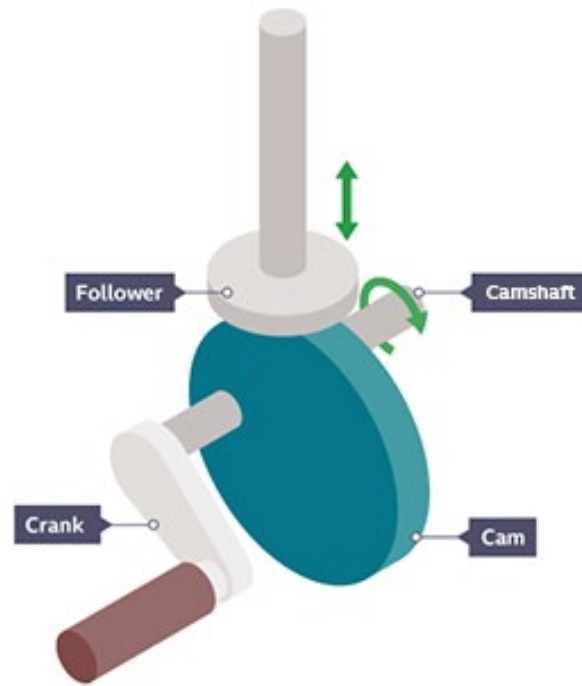


Figure 7.17: Cam Mechanism

#### 4. Geneva Drive

A Geneva drive changes continuous rotation into stop-and-go rotation. It has a wheel with slots (the Geneva wheel) and a pin that fits into the slots. The wheel rotates only when the pin moves into the next slot, creating an intermittent motion.

##### Where it is Used

- a. **Vending Machines:** Ensures items drop one at a time.
- b. **Film Projectors:** Moves film frame by frame for proper display.

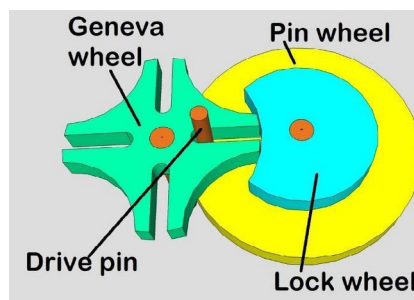


Figure 7.18: Geneva Drive

#### 5. Rack and Pinion

A rack and pinion are a system that changes rotary motion into straight-line motion. It has a pinion (a rotating gear) that moves along a rack (a straight bar with teeth).

##### Where it is Used

- a. **Steering Systems:** Allows cars to turn their wheels smoothly.
- b. **Machinery:** Used in cutting tools and other equipment for precise movement.





Figure 7.19: Rack and Pinion

## Application of Swinging Mechanisms in Robotics

Swinging mechanisms are used in a variety of robotic applications such as

1. **Manipulators:** Swinging mechanisms are used to create robot arms that can reach into small places and pick up or move things. Imagine a robot helping to clean a tight corner of a room.
2. **Locomotion:** Swinging mechanisms allow robots to walk, run, and even climb over things. Think of a robot that can go up a steep hill or jump over a small obstacle.
3. **Power Generation:** Swinging mechanisms can be used to change movement into electricity. For example, a swinging device can capture energy from wind or water movement to power lights.
4. **Actuation:** Swinging mechanisms are also used to make parts that move very accurately. This can help robots do tasks like assembling small pieces together..

## Reciprocating Mechanisms

Reciprocating mechanisms change spinning motion into back-and-forth movement. They are made up of three important parts:

1. **Crank:** This is a part that spins around.
2. **Connecting Rod:** This part connects the crank to another part.
3. **Piston:** This part moves back and forth.



When the crank spins, it makes the connecting rod move, which then causes the piston to move. These mechanisms are seen in many machines.



## Common Types

1. **Crank and Slider Mechanism:** Converts rotation into a straight motion, such as in pumps.
2. **Four-Bar Linkage:** Uses four connected parts to change between rotary and straight motion.
3. **Slider-Crank Mechanism:** Similar to the crank-slider but used in machines where smooth linear motion is needed.
4. **Geneva Mechanism:** Provides stop-and-go motion, often used in clocks.

Scan the QR codes below or visit the internet to search for and watch demonstrations of different kinds of Reciprocating Mechanisms.

Resource	QR Code
<a href="https://www.youtube.com/watch?v=ve9M8d6KfdI">https://www.youtube.com/watch?v=ve9M8d6KfdI</a>  Mechanisms for converting rotational movement into linear	
<a href="https://www.youtube.com/watch?v=4vXRuvVWufk">https://www.youtube.com/watch?v=4vXRuvVWufk</a>  Reciprocating Mechanisms using Lego Technic parts	

## Practical Applications of Reciprocating Mechanisms

Reciprocating mechanisms are used in many everyday devices such as:

1. **Bicycles:** When pedals are pushed, they make the wheels turn, which helps the bicycle to move forward.
2. **Water Pumps:** These mechanisms help move water from one place to another, like when drawing water from a well.
3. **Compressors:** These devices squeeze air into a smaller space, such as in a spray can or a bicycle pump.
4. **Robots:** In robots, reciprocating mechanisms help the arms move and can also help open and close their hands.
5. **Machines:** Many machines, like sewing machines, use reciprocating mechanisms to move needles up and down.

## Cam Mechanisms

Cam mechanisms are special devices that change spinning motion into straight or “stop and go” motion. They consist of:


1. **Cam:** This is a piece that rotates and has a specific shape.
2. **Follower:** This part follows the shape of the cam and moves up and down or side to side.
3. **Base:** This is the stable part that holds everything in place.

## Types of Cams

There are two main types of cams

1. **Disk Cams:** These are round and rotate around a middle point. They are very common and used in many machines, such as in clocks where the hands move.
2. **Linear Cams:** These are flat and slide back and forth. They are less common but can be found in devices that need straight movements, like in some types of doors.

Scan the QR code below or visit the internet to search for demonstrations of the types of cams and how they work:

Resource	QR Code
<a href="https://www.youtube.com/watch?v=HsXWewecMLE">https://www.youtube.com/watch?v=HsXWewecMLE</a>  How Do Cam and Follower Mechanisms Work?	

## Functional Applications of Cam Mechanisms

Cam mechanisms are used in many machines and devices to control how things move. Here are some important uses.

1. **Internal Combustion Engines:** In internal combustion engines, cams help open and close the valves that let air and fuel in and exhaust fumes out. This is essential for the engine to work properly.
2. **Machine Tools:** Cams control how tools move in machines that shape or cut materials. For example, they help drill holes or cut shapes in metal.
3. **Automatic Cars:** In automatic transmissions, cams help change gears smoothly so that cars can speed up or slow down easily.
4. **Printing Machines:** Cams control the movement of paper and ink in printing machines. They ensure that the right amount of ink is applied at the right time.
5. **Textile Machines:** In machines that make fabric, cams control how the fabric moves, helping to weave or knit materials together.

## Designing Cam Profiles

To make sure a cam mechanism works well, the design of the cam profile is very important. The cam profile is the shape of the cam, and it needs to be designed carefully. Here are some things to think about when designing it.

1. **Type of Motion:** What kind of movement is needed? Should it be quick or slow?

2. **Speed of the Cam:** How fast will the cam rotate? This affects how quickly the follower (the part that moves) reacts.
3. **Load on the Follower:** How heavy is the load that the follower will carry? The design must support this weight.
4. **Material Choices:** What materials will be used for the cam and follower? Different materials can affect strength and wear.

## Implementation in Robotics

Cam mechanisms are important in robotics, where they help control movements in various ways.

1. **Robotic Arms:** Cams can be used to guide how robotic arms move, allowing them to pick up or place objects accurately.
2. **End Effectors:** Cams control the movements of end effectors, which are tools or grippers at the end of a robot's arm that perform tasks.
3. **Actuators:** Cams can create actuators, which are devices that move objects very precisely, like opening a door or lifting a lid.

## Intermittent Mechanisms

Intermittent mechanisms create movement that starts and stops at regular times. This kind of motion is useful in machines where constant movement is not needed. Here are some common types of intermittent mechanisms:

1. **Geneva Drives:** Used to create step-by-step movement.
2. **Ratchet Mechanisms:** Allow movement in one direction only.
3. **Index Mechanisms:** Help move items a set distance at a time.
4. **Escapement Mechanisms:** Control the release of energy in clocks and watches.

## Ratchet Mechanisms: Function and Applications

A ratchet mechanism is a special device that lets a shaft turn in one direction but stops it from turning back.

### Parts of a Ratchet Mechanism

- a. **Ratchet Wheel:** This is a round gear with teeth that only allow movement in one direction.
- b. **Pawl:** This is a small piece that fits into the teeth of the ratchet wheel, preventing it from turning the wrong way.
- c. **Spring:** The spring keeps the pawl pressed against the ratchet wheel, making sure it holds its position.

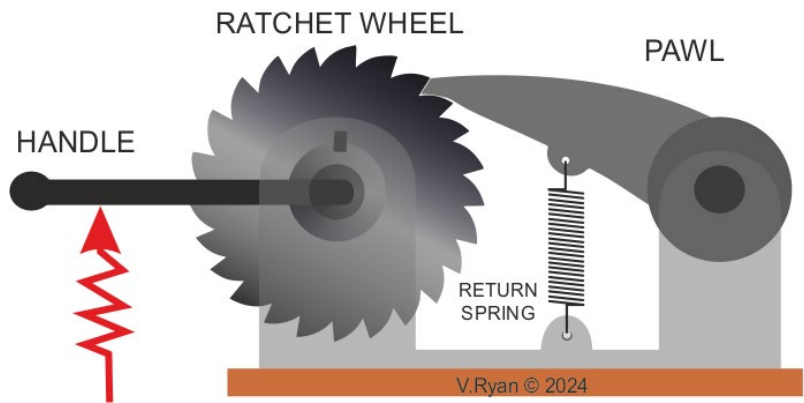




Figure 7.20: Ratchet Mechanism

Uses of Ratchet Mechanisms

Ratchet mechanisms are useful in many everyday items. They can be found in tools like socket wrenches, allowing the tool to turn a bolt in one direction while stopping it from going back. They are also used in elevators to prevent the lifts from falling backward. They are applied in robotic systems for controlled linear or rotational motion, such as in feeding mechanisms, locking systems, and actuators. This makes tasks easier and more efficient.

Scan the QR Code below to learn more on ratchet mechanism.

Resource	QR Code
<a href="https://technologystudent.com/cams/ratch1.htm">https://technologystudent.com/cams/ratch1.htm</a>  Reading resource on ratchet mechanism	
<a href="https://www.youtube.com/watch?v=1B5fytq0dkY">https://www.youtube.com/watch?v=1B5fytq0dkY</a>  A video resource on how to build a ratchet mechanism using cardboard.	

Vehicles in Robotics

In robotics, vehicles can be designed in different ways to help them move effectively. Two important types of vehicles are those that use caster wheels and those that use crawlers.

Vehicles Using Caster Wheels

Caster wheels are special wheels that help vehicles move easily and turn smoothly. They are often seen on things like shopping carts, office chairs, and hospital beds. In robotics, caster wheels are used in various machines to improve movement and stability.



**Figure 7.21: Caster Wheels**

### Design Principles of Caster Wheels

Caster wheels are usually made from metal or plastic and are designed to rotate freely. They have a part called a swivel mechanism at the top, which helps them turn 360 degrees. This mechanism often uses ball bearings or roller bearings to allow smooth rotation.

*When designing caster wheels, several factors are important for their performance*

1. **Diameter:** Larger wheels provide better stability than smaller ones.
2. **Width:** Wider wheels are less likely to tip over.
3. **Tire Material:** The type of material affects how well the wheel grips the ground and how easy it rolls.

### Practical Applications

*Caster wheels are used in many areas, including*

1. **Healthcare:** Caster wheels are found on hospital beds and wheelchairs, making it easier to move patients and equipment within healthcare facilities.
2. **Manufacturing:** In factories, caster wheels are used on industrial robots and machinery, allowing them to be moved around the factory floor effortlessly.
3. **Retail:** Shopping carts and grocery carts have caster wheels, which help customers easily transport products around stores.
4. **Food Service:** In restaurants, caster wheels are used on food delivery carts and kitchen equipment, enabling the smooth movement of food and supplies.

### Caster Wheel Configurations

There are different ways to set up caster wheels, and each type is useful for different purposes:

**1. Rigid Caster Wheels:**

- a. Do not swivel, so they only move in straight lines.
- b. Good for heavy loads that need stability.

**2. Swivel Caster Wheels:**

- a. Can turn 360 degrees.
- b. Useful when easy turning is needed, like on office chairs.

**3. Locking Caster Wheels:**

- a. Can be locked to stop the wheel from moving.
- b. Good for things that need to stay in place sometimes, like hospital beds.

**4. Swivel and Locking Caster Wheels:**

- a. Can both swivel and lock in place.
- b. Used when flexibility is most important.

**Implementing Caster Wheels**

When adding caster wheels to vehicles like robots or carts, several factors should be considered.

1. **Weight of the Vehicle:** Heavier robots may require larger or stronger caster wheels.
2. **Terrain:** The type of surface the vehicle will be used on affects the choice of tire material. For example, soft tires are better for rough surfaces.
3. **Desired Manoeuvrability:** The level of movement needed will influence the type of caster wheel configuration selected. Swivel wheels for instance will help robots turn easily.

*For robots, additional factors to consider include*

- a. **Centre of Gravity:** The robot's centre of gravity should be balanced to maintain stability.
- b. **Speed:** Faster robots may need different sizes or types of caster wheels. Wheels that roll more smoothly can be used.
- c. **Environment:** The surroundings can determine the best tire material to use.

**Crawlers**

Crawlers, also known as tracked vehicles, are special machines that use continuous tracks to move over different types of surfaces. These tracks help distribute weight and provide better grip, making crawlers very useful in various fields.

Crawlers are designed to move smoothly over challenging terrains such as mud, snow, sand, and rocky surfaces. Because of their unique design, they are commonly used in;

1. **Construction:** For moving heavy materials and equipment over rough and uneven surfaces.
2. **Agriculture:** To help with tasks like planting and harvesting.
3. **Military:** For transporting troops and equipment in difficult environments.



4. **Exploration:** To navigate rough landscapes, such as in space missions or deep-sea exploration.



**Figure 7.22: Crawler (Tracked vehicle)**

### Track Systems

The track system is the most important part of a crawler. It consists of several linked tracks that go around the vehicle's drive wheels. This design increases the area in contact with the ground, helping to spread out the weight and improve traction (grip).

### Advantages of Crawlers with Track Systems

Crawlers have several benefits compared to vehicles with wheels:

1. **Superior Traction:** Crawlers can easily move over tough terrains like mud and sand, where wheeled vehicles may struggle.
2. **Reduced Ground Pressure:** The wide tracks help reduce the pressure on the ground. This means they are less likely to sink into soft surfaces or damage delicate areas.
3. **Manoeuvrability:** Crawlers can turn and navigate well in small spaces because each track can move independently.
4. **Stability:** Continuous tracks provide better stability on uneven ground, making it less likely for the vehicle to tip over.

### Designing Crawlers

*To design an effective crawler, these factors should be considered*

1. **The Terrain:** The tracks must be made of materials and sizes that can handle the type of ground the crawler will move on.
2. **Payload (Weight):** The robot must be strong enough to carry its load. Its engine or motor should have enough power to handle the weight.
3. **Manoeuvrability:** The crawler should be able to turn easily and clear any obstacles in its path.




4. **Power:** Select an engine or motor that provides sufficient power for the intended application.
5. **Track Material:** Tracks must be tough enough to last a long time and flexible enough to move over uneven surfaces.

### Building a Crawler Robot

To create a crawler robot, follow these steps

1. **Design:** Start with a detailed plan that includes the robot's size, components, and electronic parts.
2. **Frame:** Build a strong frame using materials like aluminium or steel to support the robot.
3. **Track System:** Assemble the tracks and drive wheels, making sure everything is properly aligned and tight.
4. **Motors:** Install powerful motors that will drive the tracks.
5. **Electronics:** Add microcontrollers, sensors, and a power supply to control the robot and help it navigate.
6. **Programming:** Write software that will control the movements and actions of the robot.
7. **Testing:** Test the crawler robot on different terrains to make adjustments and improve its performance.

Scan the QR Code below or search on the internet to watch a video on crawler robots using the Lego platform.

Resource	QR Code
<a href="https://www.youtube.com/watch?v=aRuFJi-hR4A">https://www.youtube.com/watch?v=aRuFJi-hR4A</a>  Building instructions for a tracked Lego EV3 Robot vehicle	

## MOVING WITHOUT TYRES IN ROBOTICS

Robots can move in many creative ways that do not require traditional tyres. These alternative methods are important for moving in environments where wheels may not work well. In this part of section 7 you We will now explore different ways robots can move without tyres, how these movements work, and where they are used.

### 1. Inchworm-like Motion

Inchworm-like motion copies the movement of an inchworm. This involves a series of steps where the robot extends its body and then contracts to move forward.

How It Works

- a. Linear actuators or pneumatic systems are used to create the stretching and contracting motion.
- b. Gripping or suction systems at both ends help the robot hold on to surfaces, allowing it to pull itself forward.

This type of movement is useful for tasks that require precise positioning.

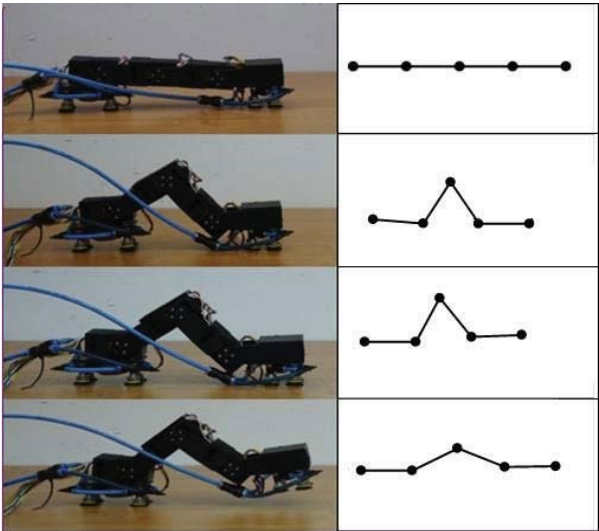



Figure 7.23: Inchworm robot motion on a vertical plane

Scan the QR code below or search online to watch a video on how inchworm robots move

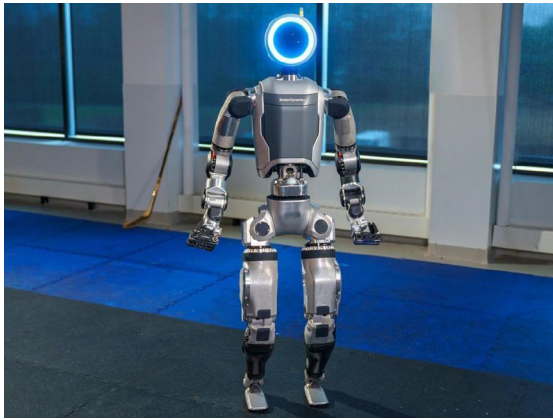
Resource	QR Code
<a href="https://www.youtube.com/watch?v=Tb7n8Xmgzaw">https://www.youtube.com/watch?v=Tb7n8Xmgzaw</a>  inchworm robot motion	

2. Legged Locomotion

Legged locomotion involves using legs to move around. This method is especially helpful for robots that need to move in rough or uneven terrains.

Types of Legged Robots

- a. **Bipedal Robots:** These robots walk on two legs, similar to humans.
- b. **Quadrupedal Robots:** These robots have four legs, making them stable and able to jump or navigate obstacles easily.



(a) A bipedal robot



(b) A quadrupedal robot

Figure 7.24: Legged robots

## Creating a Moving Robot Without Tyres

When designing a robot that does not use tyres, careful planning is essential. Here are some important factors to consider:

1. **Type of Movement:** Decide what kind of movement the robot will need, as this will influence the design of the mechanism.
2. **Terrain:** Understand the environment where the robot will operate. This will determine the kind of legs or other movement systems needed.
3. **Power Source:** Choose a power source that suits the motors and mechanisms you will use.
4. **Control System:** Plan how the robot will be controlled, including the software and sensors needed.

Once these factors are taken into account, a robot can be built using various materials and components.

## Arms, Wings, and Other Robotic Mechanisms

Robots often incorporate specialised components to perform tasks such as shooting, opening doors, raking, or generating wind. These mechanisms enhance functionality across different industries.

### 1. Shooting Robots

Robots designed to shoot or launch objects are used in many areas, from entertainment to industrial applications.

#### Design and Applications:

- a. **Spring-Loaded Mechanisms:** These use tension springs to store energy and launch objects.
- b. **Pneumatic Systems:** These systems use compressed air to propel items.

- c. **Electromagnetic Launchers:** These use electromagnetic forces to speed up objects.

#### **Applications:**

- a. Sports: Robot football or target games.
- b. Industrial Lines: Placement of parts or materials.

## **2. Automatic Doors**

Robotic automatic doors improve convenience in homes and industries.

#### **How They Work**

- a. **Sensor Integration:** Motion, pressure, or infrared sensors detect when someone is nearby.
- b. **Actuation Mechanisms:** Electric motors, pneumatic actuators, or hydraulic systems open and close the doors.
- c. **Control Systems:** Special algorithms manage how sensors and actuators work together.

## **3. Raking Mechanisms**

Robotic raking systems are useful for gathering leaves, soil, or debris, making them valuable in gardening, agriculture, and cleaning.

#### **Function and Implementation:**

- a. **Design Considerations:** Choose materials and designs for raking tines or blades that work well.
- b. **Motion Systems:** Use linear actuators or rotating parts to move the raking tools.

**Applications:** These mechanisms are found in robotic lawn mowers, agricultural robots for preparing soil, and cleaning robots that collect debris.

## **4. Wind Generation**

Some robots can create wind using motors and blades, which can be useful for cooling systems or simulating environments.

#### **How It Works:**

- a. **Blade Design:** Blades are shaped for optimal airflow and efficiency.
- b. **Motor Selection:** Motors need to have the right power and speed for driving the blades effectively.

**Applications:** Wind-generating robots can be used in cooling systems, ventilation robots, and environmental simulation setups.

# **Building and Testing Prototypes**

#### **To create effective mechanisms**

1. **Plan and Design:** Create blueprints detailing dimensions and components.
2. **Choose Materials:** Select durable, lightweight parts suitable for the task.

3. **Construct Prototypes:** Assemble all elements securely.
4. **Program Control Systems:** Write algorithms to manage sensors and actuators.
5. **Test and Refine:** Evaluate performance and adjust, as necessary.

### Activity 7.7 Building Robots using Caster Wheels

Organise yourselves into groups of 3-5 for this activity. Your teacher will put you in groups of 3-5 for this activity. You will be provided with a Robotics Kit or will need access to the materials required to make mechanisms.

#### Swinging mechanism

1. In your groups build a simple motor driven pendulum that swings back and forth.
2. Include sensors to control the angle of swing or position
3. Ensure the pendulum is stable and observe its motion.
4. Document the design process and the materials used.
5. Present your pendulum to the class, explaining how it works and what principles of physics are at play.




#### Caster Wheel to Navigate

1. Design a small robot that uses four caster wheels for navigation.
2. Ensure the robot is stable and can manoeuvre efficiently.
3. Conduct tests on different terrains and document how the caster wheels perform.
4. Present your robot to the class, highlighting the design considerations and the results of your tests.

#### Intermittent motion mechanisms

1. In your groups, watch instruction videos from your robotics kit or online and build a robot gripper using a Geneva drive or ratchet mechanism. Ensure the mechanism can perform smooth intermittent motion.
2. Attach your mechanism to a simple robot arm mounted on a chassis or frame.
3. Write down each step of your construction process. Note any challenges (e.g., alignment issues, gear slipping) and how you solved them.
4. Run your robot and observe how the intermittent motion works. Does the motion align with your expectations? What changes could improve the mechanism's efficiency?
5. Show your classmates how the intermittent motion works.
  - a. Describe how the Geneva drive or ratchet mechanism functions.
  - b. Highlight its real-world applications, such as in watches, film projectors, or industrial machinery.
  - c. Discuss the challenges and solutions from the building process.

## EXTENDED READING

Resource	QR Code
<a href="https://www.youtube.com/watch?v=pfTTHx9kCHk">https://www.youtube.com/watch?v=pfTTHx9kCHk</a> A video on velocity	
<a href="https://www.youtube.com/watch?v=Ed58JZKiGEM">https://www.youtube.com/watch?v=Ed58JZKiGEM</a> A video on acceleration	
<a href="https://www.youtube.com/watch?v=4dCrkp8qgLU&amp;t=31s">https://www.youtube.com/watch?v=4dCrkp8qgLU&amp;t=31s</a> A video on Position, velocity and acceleration	

## REVIEW QUESTIONS 7.1

1. Define average velocity.
2. What is instantaneous acceleration?
3. Describe the relationship between linear and angular velocity.
4. A car travels 150 metres north in 10 seconds. Calculate its average velocity.



## REVIEW QUESTIONS 7.2

1. If a robot's wheel has a radius of 0.2 metres and it rotates at 5 radians per second, what is its linear velocity?
2. What is the formula for angular velocity when a constant angular acceleration is given?
3. A spinning wheel has an initial angular velocity of  $10^\circ/\text{s}$  and an angular acceleration of  $2^\circ/\text{s}^2$ . What is the angular velocity after 5 seconds?
4. A robot needs to move from Point A (0,0) to Point B (6,3). It accelerates at  $0.4 \text{ m/s}^2$  until it reaches a speed of 1 m/s, then continues at that speed. How much time will the journey take?

## REVIEW QUESTIONS 7.3

1. What is the main purpose of a crank and rocker system?
2. How does a Geneva drive work?
3. How does a cam mechanism change the type of motion?
4. What is the main difference between a pendulum mechanism and a crank and rocker system?
5. Why might a planetary gear system be useful in a robotic arm?

## SECTION

# 8

## PROGRAMMING ROBOTS 2

# ROBOT CONSTRUCTION & PROGRAMMING

## Programming Robots

### INTRODUCTION TO PID CONTROLLERS

Imagine that while riding your bicycle, you are trying to ride exactly on the boundary white lines on the side of a tarred road. You do not want to go off the line to the left or the right, but exactly on that line. Chances are that if you do this without much care or attention, you may not be able to achieve this. You will have to rely on constant feedback from your eyes, which will be processed by your brain to determine how best to control your balance and speed to accurately achieve this. To achieve a similar level of accuracy in robots and other automated solutions, a PID controller does the job that your eyes and brain do in order to keep you on the line is needed. The acronym PID stands for Proportional, Integral, and Derivative, which represent the three components that contribute to the controller's overall performance. It helps robots stay on track, whether it is following a line, avoiding obstacles, or picking up objects with precision. In this section, you will learn more about how PID Controllers work.

#### KEY IDEAS

- **PID Controllers:** Are electronic devices that often Proportional, Integral, and Derivative (PID) control work together to achieve desired performance in several automated solutions like such as robotics. They are crucial for accurate robot movement and positioning.
- **Troubleshooting in Robotics:** Troubleshooting is essential for identifying and fixing issues in robotic systems. To achieve this, one needs to develop problem-solving approach is used which skills and learn to follows an established systematic approaches.

### CONTROL SYSTEMS

Earlier in year 2 you explored Previously; we introduced the concept of **open-loop control systems**. These systems work like a one-way system which has no feedback. You give a command, and the robot follows it without checking if it is doing the right thing. They may be helpful in certain scenarios however, more often than not, in robotic solutions, they may not be helpful. For example, if you tell a robot to move forward for 10 seconds, it will do just that, even if there is an obstacle in the way. This can lead to mistakes. Remember that an open-loop control system is represented in **Figure 8.1**.

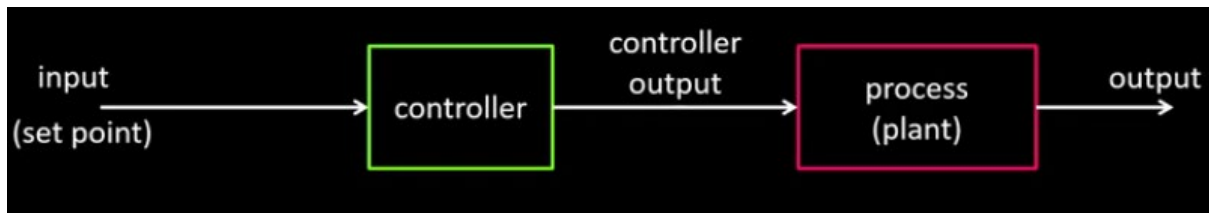


Figure 8.1: An Open-loop control system

To fix this, you can use a closed-loop or feedback control system. It is like a robot checking its work as it goes. The robot can keep checking if there is an obstacle in its path as it moves the distance instructed to move and adjust its movements accordingly. This feedback loop helps the robot carry out the instruction while avoiding mistakes. Such a control system is represented in **Figure 8.2**.

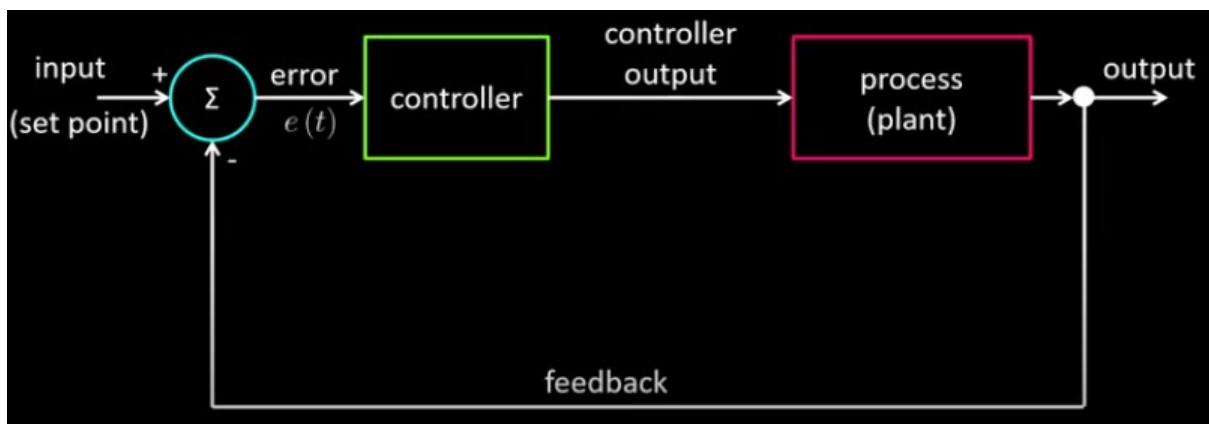


Figure 8.2: A Closed-loop/Feedback control system

## How PID Controllers Work

A PID controller helps control a system by calculating the difference between where the system is and where it should be (called the “setpoint”). For example, if a robot should move to a specific position but hasn’t reached it yet, the PID controller works to correct the difference using three parts.

1. **Proportional (P):** This part of the controller looks at how big the error is right now. If the robot is far from the target, the proportional component applies a stronger correction. It helps the robot get closer to the target faster. However, if only the proportional part is used, the robot might overshoot or keep oscillating around the target without stopping precisely on it.
2. **Integral (I):** The integral part considers all the small errors over time and adds them together. If the robot stops just short of the target every time, the integral part will gradually adjust the system to fix this issue. It ensures the robot eventually reaches the exact target. However, if not carefully adjusted, it might cause the robot to overshoot or become unstable.
3. **Derivative (D):** The derivative part looks at how quickly the error is changing. If the error is decreasing too fast, it predicts that the robot might overshoot the target and



adjusts accordingly to smooth the movement. This helps reduce wobbles and makes the system more stable.

## Steps the PID Controller Follows

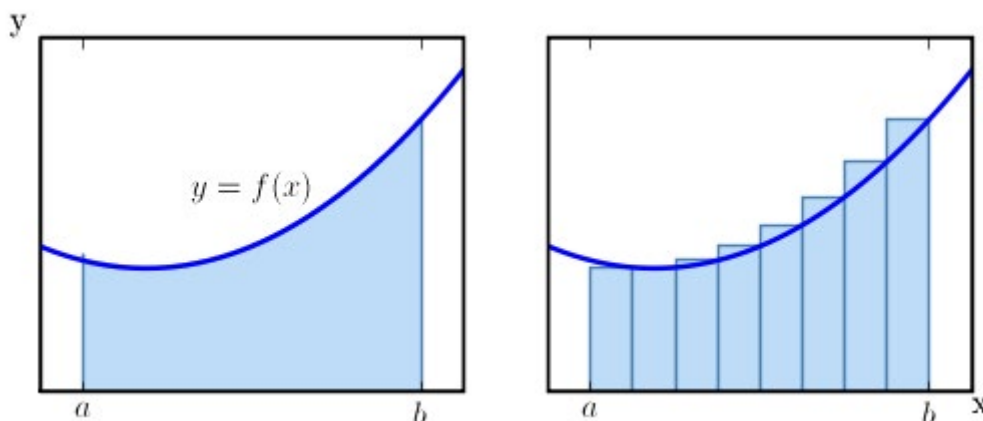
The PID controller works by constantly following the steps below:

1. **Calculate the Error:** The controller calculates the difference between the setpoint and the system's current state.
2. **Proportional Adjustment:** It multiplies the error by a constant ( $K_p$ ) to make an initial correction.
3. **Integral Adjustment:** It adds up the past errors and multiplies them by a constant ( $K_i$ ) to fix long-term errors.
4. **Derivative Adjustment:** It predicts future errors by looking at how fast the error is changing and multiplies this by a constant ( $K_d$ ).
5. **Combine Outputs:** The three adjustments (P, I, D) are added together to create the final control signal.
6. **Apply the Signal:** The control signal is sent to the system to correct its behaviour.

## Understanding Integral and Derivative

The terms “integral” and “derivative” originate from mathematics and are fundamental concepts in the branch called *field of calculus*.

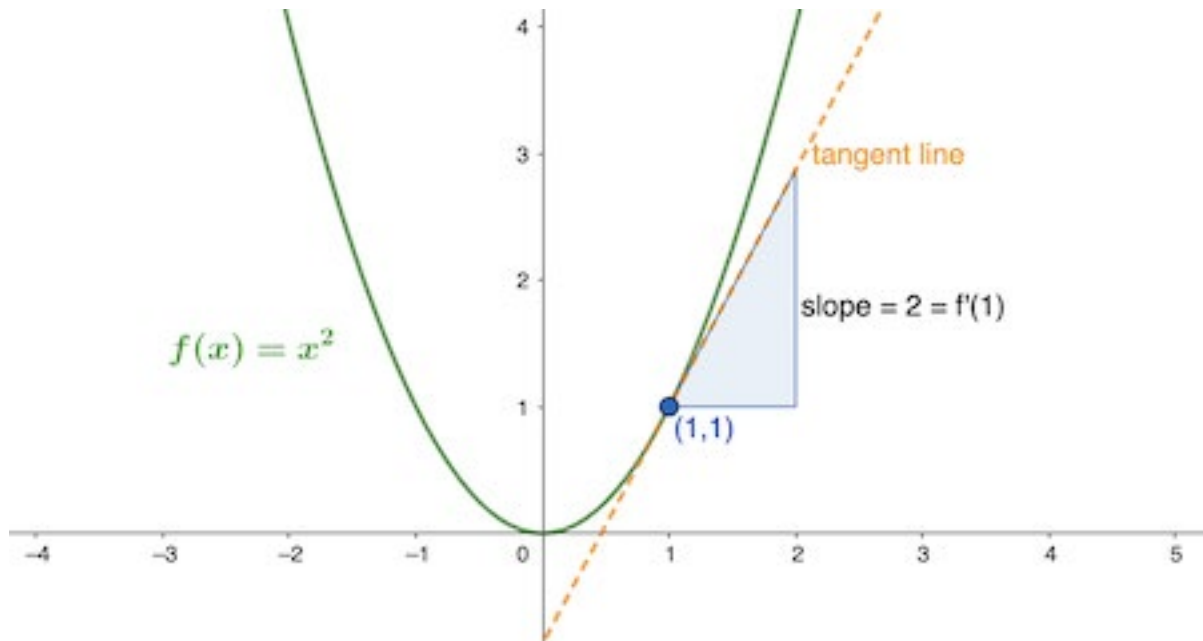
1. **Integral:** The operation of integration or the “integral” adds up the total area under a curve in a graph. In **Figure 8.3**, a graph is presented showing a car's speed measured over a period of time, travelling between points “a” and “b”. If you want we desire to **calculate the distance travelled between those two points** it will be the **same** as finding the integral which is to **calculate the total area under the curve**. It is like adding up infinitely many tiny rectangles to find the total area, just as depicted in **Figure 8.3**. In mathematics, this integral operation will be represented as “ba”.



**Figure 8.3:** Finding the area under the curve using integration

2. **Derivative:** The Derivative measures how fast something is changing at a specific moment. Using the same example above, the derivative would show how quickly the

car's speed is increasing or decreasing (its acceleration) at any point on the curve. To do that will be like drawing a tangent line to that point on the curve and then measuring its slope. This is demonstrated in **Figure 8.4**. In mathematics, the Derivative operation is usually represented by an expression similar to “ $d(y)dx$ ”.



**Figure 8.4:** Finding the derivative of a function on a graph

**So, in simpler terms:**

- a. **Integral** = Total area under the curve – the mathematical concept of integration.
- b. **Derivative** = How quickly something is changing – the mathematical concept of differentiation..

The PID controller combines these mathematical ideas to efficiently adjust and control the system. In a robotic system, this ensures smooth and accurate movements, as illustrated in **Figure 8.5**.



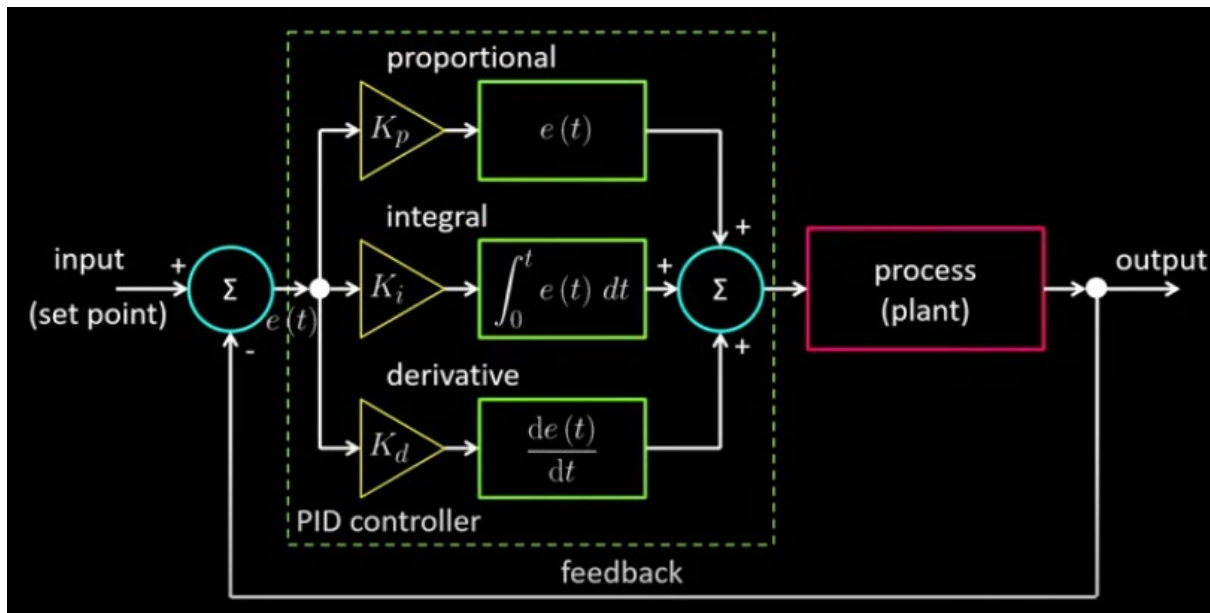


Figure 8.5: A PID Controller in a closed-loop control system

### Tuning PID Controllers

For a PID controller to work well, three settings need to be adjusted:  $K_p$ ,  $K_i$ , and  $K_d$ . These are called “gains,” and they control how much influence the proportional, integral, and derivative parts have on the system:

1.  **$K_p$  (Proportional Gain):** This controls how strongly the system reacts to the current error. A high  $K_p$  makes the system respond faster but can cause it to become unstable or oscillate too much.
2.  **$K_i$  (Integral Gain):** This focuses on fixing small, long-term errors by adding up past errors. It helps the system reach the target exactly but can make it unstable if set too high.
3.  **$K_d$  (Derivative Gain):** This looks at how quickly the error is changing. It helps smooth out the system’s response and reduces overshooting, but too much  $K_d$  can make the system overly sensitive to small changes or noise.

Finding the right values for  $K_p$ ,  $K_i$ , and  $K_d$  is often done through tuning (trial and error) or by using advanced tuning techniques like the Ziegler-Nichols PID Controller Tuning Method. Tuning means adjusting these gains to find the right balance, so the system reaches its goal quickly and smoothly, without too much delay or wobbling.

## How gain concepts can be used, for example, to tune a PID controller for a robotic arm

1. Start with  **$K_p$**  to make the arm move toward the target.
2. Adjust  **$K_i$**  so the arm stops exactly at the target without overshooting or falling short.
3. Fine-tune  **$K_d$**  to smooth out any remaining instability or jerky movements.

By carefully adjusting these settings, the robotic arm can move efficiently and accurately. Note that not all control systems need the full PID structure. Sometimes, using just a Proportional (P) controller, a Proportional-Integral (PI) controller, or a Proportional-

Derivative (PD) controller is sufficient. The choice depends on the specific requirements of the system.

## How gain concepts can be used to tune a PID controller for a line following robot

### 1. Proportional Gain (Kp)

*Scenario:* Imagine the robot is slightly off the line.

*High Kp:* The robot will quickly correct its position but may overshoot and oscillate around the line.

*Low Kp:* The robot will correct its position slowly and may take longer to return to the line.

### 2. Integral Gain (Ki)

*Scenario:* The robot has a persistent small error due to imperfect calibration.

*High Ki:* The robot will eliminate the small error quickly but may introduce oscillations if too aggressive.

*Low Ki:* The robot will slowly correct the small error and might not fully eliminate it over time.

### 3. Derivative Gain (Kd)

*Scenario:* The robot encounters a sharp turn on the line.

*High Kd:* The robot will react cautiously to sharp changes, reducing overshoot and oscillations but may respond slowly to the turn.

*Low Kd:* The robot will react quickly to sharp changes, which might result in overshooting the turn and oscillations.

Summary Table

Gain	High Gain Behavior	Low Gain Behavior
Kp	Quick corrections, possible overshoot	Slow corrections, longer return to the line
Ki	Quick elimination of small errors, oscillations	Slow elimination, persistent small errors
Kd	Cautious response, reduced overshoot	Quick response, potential overshoot

By tuning the gains (Kp, Ki, Kd), you can achieve a balanced response where the robot accurately follows the line, minimizes overshoot, and quickly corrects any deviations.

### Activity 8.1 Implementing A Pid Controller

Organise yourselves into Your teacher will put you in groups of between 3 and 5 for this activity. You will need and provide you with a robotic kit/simulation tool and a line following mat.

In your groups, watch the video in the link below or go online and watch a video on implementing a PID controller using your available kit.

Resource Link	QR Code
PID Line follower for Spike Prime! <a href="https://www.youtube.com/watch?v=HGwXER0hRMg">https://www.youtube.com/watch?v=HGwXER0hRMg</a>	
PID Line Follower for EV3 <a href="https://www.youtube.com/watch?v=AMBWV_HGYj4">https://www.youtube.com/watch?v=AMBWV_HGYj4</a>	

### 1. The role of PID controllers

- Define PID controllers and their role in achieving precise movement and positioning in robots.
- Explain how each of the following might be used to refine robot control: Proportional gain (P), Integral gain (I), and Derivative gain (D).

### 2. Exploring PID functionality, tuning and simplifying

- Build a line following robot programmed to move from point A to point B on a line mat.
- Test your program by running your robot using about 45% of its maximum speed on the mat provided by your teacher.
- Experiment using the PID simulation software and record what happens when you change the following:
  - Adjust the  $K_p$  values and note how it affects the performance of the robot.
  - Adjust the  $K_d$  values and note how it affects the performance of the robot.
  - Adjust the  $K_i$  values and note how it affects the performance of the robot.

Record the  $K$  values that made your robot move smoothly from point **a** to point **b** on the mat. Discuss your findings with your classmates and explain the effects that the various gains you implemented have on the robot's performance.

# A SIMPLE GUIDE TO TROUBLESHOOTING IN ROBOTICS

In robotics, troubleshooting means finding and fixing problems when things do not work as they should. No matter how well-designed a robot is, issues can happen. Learning how to solve these problems in an organised logical way, step by step if you like, not only saves time but also helps you understand your robot better and improve its design in the future. Here is an easy guide to help you troubleshoot your robot:

## Step 1: Follow a Routine

Start with a checklist and go through it whenever there is a problem. This ensures you do not miss anything important. Here's a basic troubleshooting routine:

- a. **Check the Power**
  - Is the robot switched on?
  - Are the batteries charged, or is there enough power?
- b. **Check Mechanical Parts**
  - Are the motors, wheels, or joints working smoothly?
  - Is anything stuck, loose, or misaligned?
- c. **Check Electrical Connections**
  - Are all wires firmly connected?
  - Are there any damaged or frayed wires?
- d. **Check the Software**
  - Is the program running as it should?
  - Are there any error messages or strange behaviours?

By going through these steps one by one, you can quickly narrow down the issue.

## Step 2: Isolate the Problem

Robots have different parts, like motors, sensors, and code, all working together. Troubleshooting is easier when you focus on one part at a time. For example,

- a. **If the robot is not moving:**
  - Check if the motors are getting power.
  - Look for anything blocking the wheels.
  - See if the code controlling movement has errors.
- b. **If the robot is not sensing properly:**
  - Check if the sensors are connected.
  - Test if they are giving the right data to the program.

By testing parts individually, you can figure out where the problem is. For example, if the motors are fine but the robot still doesn't move, the issue might be in the code or electrical connections.

### Step 3: Pay Close Attention

Observation is one of the best tools for troubleshooting. Use your senses to find clues

- a. **Look:** Are any parts broken, bent, or out of place? Is the robot moving smoothly or wobbling?
- b. **Listen:** Do you hear unusual sounds like grinding or clicking? This could mean something is stuck or wearing out.
- c. **Feel:** Is the robot vibrating more than usual? Are any parts getting too hot?

For example, if a motor feels unusually hot, it might be working too hard or not connected properly.

### Step 4: Know What to Expect

Sometimes, what looks like a problem might not actually be one. Robots have limitations based on their design or programming.

For instance, if a robotic arm does not reach all the way to the edge of a table, it might be because the program limits how far it can extend—not because something is broken. Knowing how your robot is supposed to work helps you tell the difference between normal behaviour and real problems.

### Step 5: Understand How the Robot Works

The better you understand the robot, the easier it is to troubleshoot. This means knowing how the hardware (like motors and sensors) works with the software (like programs and algorithms).

- a. **Hardware and Software:** Sensors collect data and send it to the controller. The program processes the data and tells the motors what to do.
- b. **Feedback Loops:** Some robots adjust their actions based on feedback. For example, if a robot moves too far, it might use a sensor to correct itself.

If you understand these connections, you can trace problems back to their source. For example, if the robot turns too far, it could be a coding error or a sensor not sending accurate data.

## Common Issues in Robots

There are three main types of problems you might face:

1. **Algorithm Problems:** These are mistakes in the logic that controls the robot. For example, if a robot overshoots its target, the control algorithm might need adjusting. Fix it by testing different settings or running simulations.

2. **Design Problems:** Sometimes, the robot's structure causes issues. For example, if the robot tips over easily, its centre of gravity might be too high. Fix it by testing the physical design and making adjustments.
3. **Coding Errors:** Bugs in the program can cause the robot to act unexpectedly. For example, if the robot ignores sensor data, there might be a mistake in how the code reads the sensors. Fix it by checking error messages or testing smaller sections of the code.

### Note

Algorithm issues are about the plan for how the robot should act, while coding errors are mistakes in how that plan is written into the program.

## FIXING IDENTIFIED PROBLEMS

Troubleshooting robots is not just about finding what is wrong. It is also about fixing those problems and improving the robot until it works properly. This involves testing, making changes, and testing again. Let us break down how you can fix different kinds of problems step by step.

### Fixing Algorithm Problems

An algorithm is the “brain” of the robot. It's a set of rules that tells the robot how to act. If the robot's behaviour is wrong, the algorithm might not be working well. Here's how to fix algorithm problems:

#### 1. Understand What the Robot Should Do

Think about what the robot is supposed to achieve. For example, if the robot is following a line on the floor, it should be able to handle turns, stop when needed, and avoid obstacles.

#### 2. Compare What It Does vs. What It Should Do

Watch the robot closely. Is it following the line, or is it wandering off? Is it stopping too late? If its behaviour doesn't match what's expected, the algorithm might need fixing.

#### 3. Break the Algorithm Into Parts

Look at each part of the algorithm. For example:

- a. Does the robot calculate distances correctly?
- b. Does it know when to turn?
- c. Does it adjust its speed when approaching a corner?
- d. Test each part separately to find where it's going wrong.

**4. Make Adjustments**

Once you find the problem, tweak the algorithm. For example:

- a. If the robot is moving too fast near a turn, program it to slow down.
- b. If it's not detecting an obstacle, update the code to include that scenario.

**5. Test Again**

After making changes, test the robot in different conditions to make sure the algorithm now works as expected.

## Fixing Design Problems

Sometimes, the robot's physical structure causes issues. For example, parts might be too weak, poorly balanced, or not fit for the job. Here's how to fix these problems

**1. Inspect the Robot's Structure**

Look for obvious problems. Are the wheels straight? Is anything loose or wobbly? Does the frame seem unstable?

**2. Test the Design Virtually (if possible)**

Use computer simulations or simple models to test the robot's design. This can help find weak points. For example, if the robot tips over during turns, the centre of gravity might be too high.

**3. Match the Design to the Job**

Check if the robot is built for its task. For instance:

- a. If it's meant to carry heavy loads but struggles, the motors might not be strong enough.
- b. If it's slipping on smooth floors, the wheels might need better traction.

**4. Make Adjustments**

Fix the issues you've identified:

- a. Replace weak parts with stronger materials.
- b. Adjust the frame to improve balance.
- c. Upgrade components like motors or batteries if needed.

**5. Test the Robot Again**

After fixing the design, test the robot in real-world conditions to ensure it works better and hasn't developed new problems.

## Fixing Coding Errors

Coding errors are mistakes in the program that controls the robot. These can cause the robot to behave strangely or not work at all. Here's how to address coding issues:



### 1. Notice What's Going Wrong

Pay attention to when and how the robot is misbehaving. Does it freeze, move erratically, or react too slowly? Note when the issue happens.

### 2. Check the Code

Look at the part of the program controlling the problem. For example:

- a. If the robot isn't stopping when it should, check the section that handles stopping.
- b. Look for typos, missing commands, or mistakes in the logic.

### 3. Debug the Code

Use debugging tools or add print statements to see what's happening inside the program. For example, you can check if the robot is correctly reading sensor data or sending the right signals to the motors.

### 4. Fix the Code

Once you find the problem, rewrite the code to fix it. For instance:

- a. If the robot isn't stopping, make sure the condition for stopping is correct.
- b. If the sensor data is being misinterpreted, adjust how the code processes it.

### 5. Test in Different Scenarios

Run the robot through various situations to make sure the fix works everywhere, not just in one specific test.

## Document Your Work

As you troubleshoot and fix problems, it's important to keep track of what you did. A clear record helps others understand the process and makes it easier to solve similar problems in the future.

A troubleshooting document can include

1. **Title Page:** This is the first page of the document. It includes the title of the document, the name of the project or robot, the date, and the names of the people who worked on solving the problem.
2. **Table of Contents:** This section lists all the main sections and subsections of the document, along with their page numbers. It helps the reader find specific parts quickly.
3. **Introduction:** This section gives a brief overview of the problem. It explains what went wrong and how the robot was not working as it should.
4. **System Overview:** This part describes how the robot is designed. It includes information about the hardware (like sensors and motors) and the software (like the control system). It also gives a summary of the algorithm the robot uses to perform its tasks.
5. **Observations:** This section explains what you noticed about the problem. It describes in detail the signs or symptoms that showed the robot was not working properly.

6. **Flaw or Error Identification and Root Cause Analysis:** This section explains what caused the problem. It categorises the issue as an algorithm flaw, design flaw, or coding error. It also gives details about the root cause and the steps taken to find it.
7. **Iteration Process for Fixing:** Here, you describe how you tried to fix the problem. Start with the first attempt and explain the changes you made after testing, until the issue was resolved.
8. **Testing and Validation:** This section explains how you tested the robot after making changes to ensure the problem was fixed. If the corrections improved the robot's overall performance or efficiency, include that information here as well.
9. **Lessons Learned:** This part highlights the important things you learned during the troubleshooting process. It also explains how similar problems can be avoided in the future.
10. **Conclusion:** This is a short summary of the entire troubleshooting process. It includes how the problem was fixed, the results, and the key lessons learned.
11. **Appendices:** This section contains any extra materials, such as code, diagrams, or references, that which might help someone understand the problem and the solutions better.

### Activity 8.2 Troubleshooting a Robot



In the same groups from **Activity 8.1**, your teacher will need provide you with a troubleshooting guide and a worksheet to record your observations and findings.

1. Go back to the robot you programmed in **Activity 8.1** and run it again using the maximum speed possible and the same  $K_p$ ,  $K_i$  and  $K_d$  values on the line following mat. Use your worksheet to record any strange behaviours. For example:
  - a. Does the robot move smoothly?
  - b. Does it follow the line at all times, or does it veer off the line at times?
2. Think about what might be causing the robot's problem. Break the robot system into parts.
  - a. Hardware: Are all parts attached and working as they should?
  - b. Software: Does the program have any mistakes?
  - c. Algorithm: Is the robot following the correct steps?
3. Follow the troubleshooting guide to test each part of the program one gain at a time to narrow down the problem.
  - a. Tune the  $K_p$  value
  - b. Tune the  $K_i$  value
  - c. Tune the  $K_d$  value

- Write down what you discover; for example: “The robot shakes a lot because the  $K_p$  value is too high”.

Fix the problem and use the troubleshooting guide to prepare a troubleshooting and error correction technical organise your report. Present your findings and solution to your class and explain what worked as well as what did not work.

## EXTENDED READING

Resource	QR CODE
<a href="https://www.youtube.com/watch?v=UR0hOmjaHp0">https://www.youtube.com/watch?v=UR0hOmjaHp0</a>  PID Control - A brief introduction	
<a href="https://people.duke.edu/~hpgavin/SystemID/References/Astrom-Feedback-2006.pdf">https://people.duke.edu/~hpgavin/SystemID/References/Astrom-Feedback-2006.pdf</a>  Feedback Systems: An Introduction for Scientists and Engineers  Page 301 - 320	

## REVIEW QUESTIONS 8.1

1. What does a PID controller stand for, and what are its three components?
2. List three advantages of using PID controllers in robotics compared to open-loop control systems.
3. If a robot consistently overshoots its target, which PID gain would you adjust, and why?
4. How would a PID controller be used differently in a line-following robot versus a wall-following robot?
5. What are some limitations of using PID controllers in robotics?
6. Design a simple experiment to test the effect of increasing the proportional gain ( $K_p$ ) on a line-following robot.

## REVIEW QUESTIONS 8.2

1. What are the three main categories of issues that can arise when troubleshooting a robot?
2. Why is it important to follow a routine when troubleshooting a robot?
3. How can you tell the difference between a design flaw and a coding error?
4. If a robot moves too far past its target, what might be the cause, and how could you fix it?
5. What are some steps you can take if a robot is not responding to commands?

## REFERENCES

1. Aaron Maurer (2021). Smart robotics with LEGO MINDSTORMS Robot Inventor. Packt Publishing.
2. All3DP. (n.d.). Choosing filament: PLA vs ABS vs PETG. Retrieved from <https://all3dp.com>
3. Autodesk. (n.d.). Tinkercad Learning Portal. Retrieved from <https://www.tinkercad.com/learn>
4. Bonnin, M. (2021). Design engineering for 3D printing: Scanning, generative design, and optimization. CRC Press. Retrieved from <https://www.routledge.com>
5. Brown, S., & Vranesic, Z. (2013). Fundamentals of digital logic with Verilog design (3rd ed.). McGraw-Hill.
6. Cline, L. S. (2018). Fusion 360 for makers: Design your own digital models for 3D printing and CNC fabrication. Make Community, LLC. Retrieved from <https://www.amazon.com>
7. Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2022). Introduction to algorithms (4th ed.). MIT Press.
8. Craig, J. J. (2018). Introduction to Robotics: Mechanics and Control (4th ed.). Pearson.
9. Edutopia. (n.d.). 3D Printing in Education: Bridging Concepts and Prototyping. Retrieved from <https://www.edutopia.org>
10. Eldrid, S. (2020). 3D Printing and Maker Lab for Kids: Create amazing projects with CAD design and STEAM ideas. Quarry Books. Retrieved from <https://quarto.com>
11. Eun Jung Park (2014). Exploring LEGO Mindstorms EV3 Tools and Techniques for Building and Programming Robots. Wiley.
12. Farrell, J. (2017). Programming Logic and Design, Comprehensive (9th ed.). Cengage Learning.
13. Floyd, T. L. (2021). Digital fundamentals (12th ed.). Pearson.
14. Horowitz, P., & Hill, W. (2015). The art of electronics (3rd ed.). Cambridge University Press.
15. Hubs, 3D. (n.d.). Fused Deposition Modeling (FDM): Basics and Applications. Retrieved from <https://www.hubs.com/knowledge-base>
16. Katz, R. H., & Borriello, G. (2014). Contemporary logic design (2nd ed.). Pearson.
17. Korfhage, R. R. (2019). Boolean algebra and its applications in digital circuits. Springer.
18. Maker's Muse. (n.d.). Beginner's Guide to CAD and 3D Printing [YouTube channel]. Retrieved from <https://www.youtube.com/channel/UCxQbYGpbdrh-b2ND-AfIybg>
19. Malvino, A. P., & Brown, J. A. (2020). Digital computer electronics (4th ed.). McGraw-Hill.
20. Mano, M. M., & Ciletti, M. D. (2017). Digital design: With an introduction to the Verilog HDL, VHDL, and system Verilog (6th ed.). Pearson.
21. McCarthy, J. M., & Soh, G. S. (2011). Geometric Design of Linkages. Springer.

22. Norton, R. L. (2020). *Design of Machinery: An Introduction to the Synthesis and Analysis of Mechanisms and Machines* (6th ed.). McGraw-Hill.
23. Robotics Co., Ltd. (2018). *ROBOTIS Design and Development Manual*. Robotics Learning Materials.
24. Russell, S., & Norvig, P. (2020). *Artificial intelligence: A modern approach* (4th ed.). Pearson.
25. Sanjay and Arvind Seshan (2020). *PID LINE FOLLOWER*, Prime Lessons.
26. Siciliano, B., & Khatib, O. (Eds.). (2016). *Springer Handbook of Robotics* (2nd ed.). Springer.
27. Siegwart, R., Nourbakhsh, I. R., & Scaramuzza, D. (2011). *Introduction to Autonomous Mobile Robots* (3rd ed.). MIT Press.
28. Skiena, S. S. (2020). *The Algorithm Design Manual* (3rd ed.). Springer.
29. Thai, C. (2017). *Exploring Robotics with ROBOTIS Systems*. CRC Press.
30. Tietze, U., Schenk, C., & Gamm, E. (2015). *Electronic circuits: Handbook for design and applications* (2nd ed.). Springer.
31. Tocci, R. J., Widmer, N. S., & Moss, G. L. (2017). *Digital systems: Principles and applications* (12th ed.). Pearson.
32. Ultimaker. (n.d.). *Cura Slicing Software Guides*. Retrieved from <https://ultimaker.com/software/ultimaker-cura>
33. Wakerly, J. F. (2017). *Digital design: Principles and practices* (5th ed.). Pearson.
34. Wei Lu (2016). *Beginning Robotics Programming in Java with LEGO Mindstorms*. Apress.



## GLOSSARY

- **3D Printing Workflow:** The step-by-step process of designing, slicing, printing, and post-processing a 3D-printed object.
- **9.Gain:** The values used in a PID controller to determine how much the Proportional, Integral, and Derivative parts affect the robot's actions.
- **Actuator:** A device in a robot or system that carries out actions, like moving a part or turning a wheel.
- **Additive Manufacturing:** A manufacturing process, such as 3D printing, where objects are created by adding material layer by layer.
- **Algorithm:** A step-by-step set of instructions that a robot follows to complete a task.
- **Angular Velocity:** The rate at which an object rotates, measured in radians per second.
- **Assistive Robotics:** Robots designed to aid individuals, especially in healthcare or elderly care, such as Elli Q or robotic nursing assistants.
- **Boolean Algebra:** A mathematical system used to analyse and simplify digital logic circuits using binary values (0 and 1).
- **Caster Wheels:** Wheels that swivel in all directions, helping robots or carts turn and move easily.
- **Closed Chains:** Systems where parts are connected in a loop, making their movements depend on each other, like a crane's arm.
- **Closed-Loop Control:** A system where the robot continuously monitors its progress and makes adjustments as needed to stay on track.
- **Combinational Circuit:** A digital circuit in which the output depends only on the current inputs, without memory or feedback.
- **Computer-Aided Design (CAD):** A digital tool used to create, modify, and optimize 2D and 3D models of robotic components.
- **Control System:** A mechanism that manages and directs actions in machines or processes, often using sensors and processors.
- **Crawler:** A robot or vehicle with tracks instead of wheels, designed to move on uneven or soft surfaces.
- **Debugging:** Finding and fixing mistakes in a robot's program to make it work correctly.
- **Degrees of Freedom (DoF):** The number of independent ways a robot or machine can move, such as turning or sliding.
- **DeMorgan's Theorem:** A set of Boolean algebra rules that transform expressions involving NOT, AND, or OR for simplification.
- **Digital Accessibility:** Ensuring robots are user-friendly and accessible to individuals with varying levels of technological proficiency.
- **Digital Prototyping:** The process of testing and refining a design in a virtual environment before creating a physical model.

- **Error Correction:** The process of identifying deviations from desired system behaviour and adjusting controls to correct them.
- **Ethical Programming:** The process of embedding moral principles into a robot's behaviour through software design.
- **Feedback Loop:** A system where the robot checks its performance and adjusts its actions to match the desired goal. (Note: Consolidated duplicate definitions for clarity.)
- **Filament:** The thermoplastic material used in FDM 3D printing, available in types like PLA, ABS, and PETG, each with unique properties.
- **Freehand Sketching:** A preliminary design technique where rough, hand-drawn sketches help visualize ideas before creating detailed CAD models.
- **Fused Deposition Modelling (FDM):** A common 3D printing technique that uses melted thermoplastic filament to create objects layer by layer.
- **G-code:** A machine-readable language that provides instructions for 3D printers, CNC machines, and other automated manufacturing tools.
- **Human-Machine Interaction (HMI):** The ways in which humans and robots communicate and work together, ensuring efficient and safe collaboration.
- **Karnaugh Map (K-Map):** A visual tool used to simplify Boolean expressions and optimize digital circuit designs.
- **Linear Velocity:** How fast an object moves in a straight line, calculated as distance divided by time.
- **Logic Gate:** A basic electronic component that performs a logical operation, such as AND, OR, or NOT, in digital circuits.
- **Non-Feedback Loop:** A system that follows set instructions without adjusting based on real-time information.
- **Obsolescence:** The process by which robots become outdated due to advancements in technology or changes in user needs.
- **Open-Loop Control:** A system where the robot follows commands without checking if it is performing them correctly.
- **Oscillation:** A repeated back-and-forth movement, often caused by incorrect settings in the robot's control system.
- **Parametric Modelling:** A CAD technique that allows designers to adjust dimensions and constraints to modify designs dynamically.
- **Planetary Gears:** A compact gear system used to control speed and torque in robotic arms or vehicles.
- **Printed Circuit Board (PCB):** A board that connects and supports electronic components using conductive pathways.
- **Product-of-Sums (POS) Form:** A Boolean expression format where multiple ORed terms are combined using AND operations.
- **Real-Time Adjustments:** Changes made to a system's behaviour as it operates, based on current data or sensor inputs.
- **Reciprocating Mechanism:** A system that converts rotary motion into back-and-forth movement, like in water pumps.

- **Set Point:** The target value a system aims to maintain, such as a specific temperature in a heating system. (Note: Consolidated duplicate definitions for clarity.)
- **Slicing Software:** A program that converts a 3D CAD model into G-code, which instructs a 3D printer on how to build the object layer by layer.
- **Soldering:** The process of joining electronic components by melting solder to create a conductive bond.
- **Stability Analysis:** Examining whether a system can maintain its desired behaviour over time without oscillations or divergence.
- **Sum-of-Products (SOP) Form:** A Boolean expression format where multiple ANDed terms are combined using OR operations.
- **Swinging Mechanism:** Systems that create back-and-forth motion, used in tools like windshield wipers or robotic arms.
- **Thermal Capacity:** A property of materials or systems that describes their ability to store heat, important for temperature control models.
- **Trajectory:** The path a robot or object follows while moving, which can be straight, curved, or complex.
- **Truth Table:** A table that displays all possible input combinations and their corresponding outputs for a logic circuit.
- **Tuning:** The process of adjusting the settings of a PID controller to make the robot perform better.

## Acknowledgements



Ghana Education  
Service (GES)



## List of Contributors

Name	Institution
Kwame Oteng Gyasi	Kwame Nkrumah University of Science and Technology
Isaac Nzoley	Wesley Girls' High School, Cape Coast