CAD / CAM

SUBJECT CODE: MFPC3005

Computer Aided Design and Manufacturing (CAD/CAM)

PREPARED BY

Dr. Soumya Ranjan Parimanik
Guest Faculty

Dr. Ajit Kumar Pattanaik
Assistant Professor

Department of Mechanical Engineering
Government College of Engineering, Kalahandi

THIS IS FOR EDUCATIONAL PURPOSE

Syllabus and reference books

Module I
08 hours

Fundamentals of CAD: design process, applications of computer for design, creating the manufacturing database, design workstation, graphical terminal, operator input devices, plotters and other devices, CPU, secondary storage

Module II
08 hours

Computer graphics software and database: configuration, graphics packages, constructing the geometry, transformations, database structure and content, wire frame versus solid modeling

Module III
08 hours

CAM introduction, NC and NC part programming: NC coordinate system, motion control, economics of NC, manual and computer-aided programming, APT language, NC programming with interactive graphics

Module IV
08 hours

Problems with conventional NC; NC technology: CNC, DNC, combined DNC/CNC system, adaptive control manufacturing systems

Module V
08 hours

Computer integrated manufacturing system, machine tools and related equipment, materials handling and storage system, computer system

Text books

1. Groover, M. P. and Zimmers, E. W., Computer Aided Design and Manufacturing, Prentice Hall

2. Radhakrishnan, P. and Subramanyam, S., CAD/CAM/CIM, Wiley Eastern Ltd.

3. Amirouche, F., Principles of Computer Aided Design and Manufacturing, Pearson

CAD / CAM

MODULE I

Fundamentals of Computer Aided Design

MFPC3005 Computer Aided Design and Manufacturing (CAD/CAM)

Design process, applications, CAD hardware and devices

Reference: Groover & Zimmers, CAD/CAM: Computer-Aided Design and Manufacturing, Prentice Hall

Module I roadmap

01	**Fundamentals of CAD**	What CAD is and why industry uses it
02	**The design process**	Six phases, concurrent engineering, DFMA
03	**Applications of computers for design**	Geometric modeling, analysis, review and drafting
04	**Creating the manufacturing database**	One design file feeding production
05	**Design workstation and graphics terminal**	How CAD images are generated and displayed
06	**Operator input devices**	Cursor control, digitizers and keyboards
07	**Plotters, CPU and secondary storage**	Hardcopy output, processing and data storage
08	**BPUT previous year questions**	2-, 6- and 16-mark questions

By the end of Module I you should be able to

- Define CAD and explain the role of interactive computer graphics
- Describe the six phases of design and the CAD function that supports each
- Explain how CAD creates the database that manufacturing uses
- Compare storage tube, refresh and raster graphics terminals
- Classify the input and output devices of a CAD workstation
- Describe the CPU and secondary storage options in a CAD system

01

Fundamentals of CAD

What CAD is, how interactive graphics works, and why companies adopt it

CAD uses a computer to create, modify, analyze and optimize a design

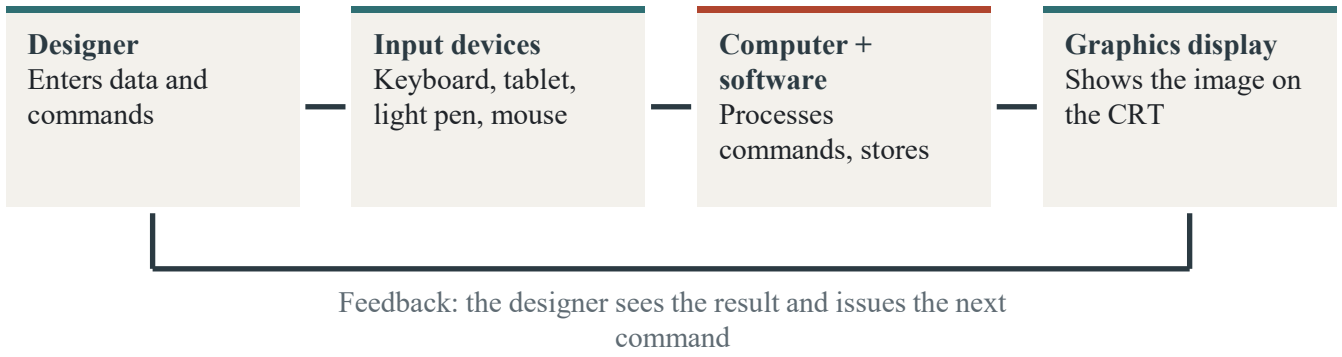
- **Computer-aided design:** the use of a computer system to help create, change, analyze or improve a design. The designer still makes the decisions; the computer stores, calculates and draws.
- **Create:** the designer builds the shape of the part on screen from points, lines, arcs and solids instead of drawing it by hand.
- **Modify:** any dimension or feature can be changed later, and every view and drawing updates from the same model.
- **Analyze and optimize:** the stored model is used for calculations such as weight, stress or fit, so the design can be improved before anything is made.
- **Hardware + software:** a CAD system needs both: a computer with a graphics terminal and input devices, and programs for graphics and engineering work.

Example

A bracket is drawn once as a 3D model. The designer changes the hole diameter from 10 mm to 12 mm, and the front, top and side views, the drawing dimensions and the weight are all updated automatically.

Interactive computer graphics (ICG) is the core of every CAD system

The designer and the computer exchange information through pictures and symbols. The image is built from points, lines and circles, then edited by command.



ICG is one component of CAD: the design itself is still made by the engineer, the computer is the tool.

Remember

ICG is only one part of a CAD system. CAD = ICG + engineering analysis and other design software. A CRT screen and input devices make the link between designer and computer.

- **Interactive:** the designer and computer talk to each other continuously: a command is entered, the picture changes at once, and the designer reacts to what is seen.
- **Image from elements:** the picture is built from basic geometric elements (points, lines, circles) that are stored as numbers in the computer, not as a photo.
- **Transformations:** the stored elements can be enlarged, reduced, moved, rotated or mirrored by a single command, without redrawing.
- **Hardware:** CPU, one or more workstations with a graphics terminal, and peripherals such as plotters and printers.
- **Software:** graphics programs that draw and manipulate the image, plus application programs for the company's engineering work.

Companies adopt CAD for four reasons

- **Productivity:** the designer spends less time on drawing and calculation, so a design is finished faster and lead time shrinks.
- **Quality of design:** the computer allows more alternatives to be analyzed in detail, so the final design is better and has fewer errors.
- **Communication:** drawings are neat, standard and legible, with fewer drafting errors, so production reads them correctly.
- **Manufacturing database:** the geometry, dimensions and bill of materials created during design are stored and reused directly by manufacturing.

Key point

In short, CAD gives four benefits: higher productivity, better design quality, better communication through drawings, and a database for manufacturing.

CAD outperforms manual drafting on speed, accuracy and reuse

Aspect	Manual drafting	CAD
Speed	Every line drawn by hand	Copy, mirror, array and edit existing geometry
Accuracy	Limited by drafter and scale	Stored to computer precision
Changes	Redraw or erase the sheet	Edit the model; views update
Storage	Paper sheets in cabinets	Digital files on disk or tape
Analysis	Separate hand calculations	Mass properties and FEA from the same model
Reuse downstream	Re-read drawings for NC and planning	Same database feeds manufacturing

Example

A gearbox housing needs a wall 2 mm thicker. Manually, three views and the section must be redrawn. In CAD, one dimension is edited and every view, section and dimension updates.

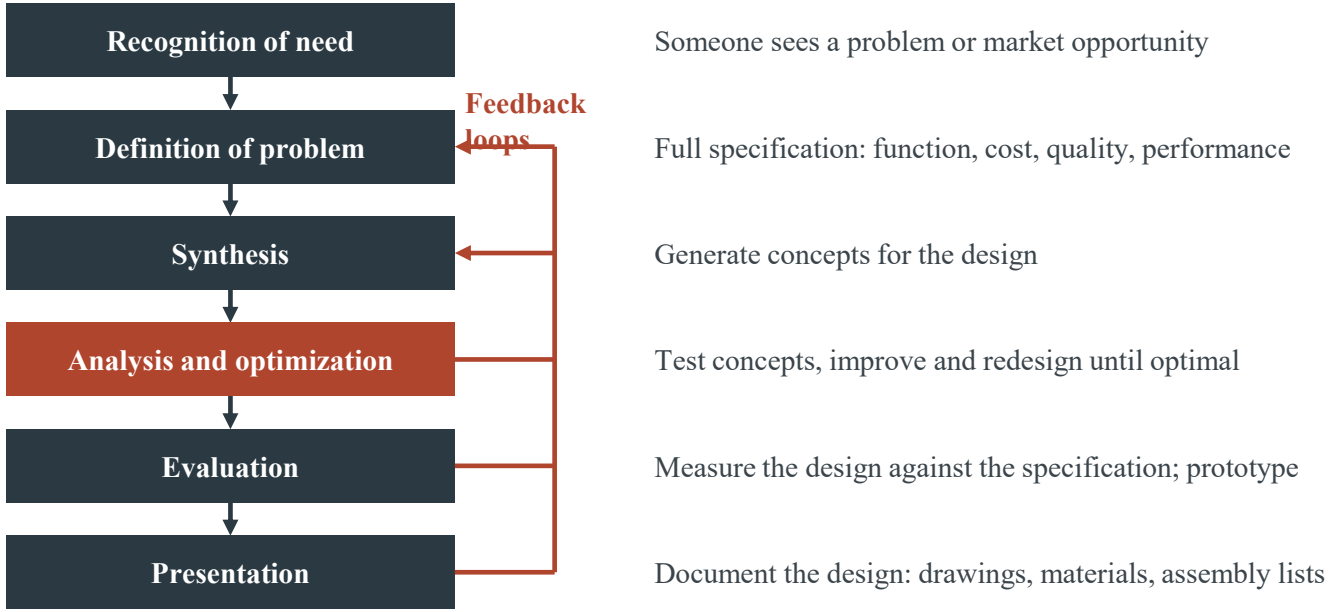
- **Speed:** in manual drafting every view and every change is redrawn by hand; in CAD a change to the model updates all views in seconds.
- **Accuracy:** CAD stores exact coordinates, so dimensions and intersections are precise; hand drawings depend on the draughtsman's skill.
- **Reuse:** standard parts and earlier designs are copied from a library instead of being drawn again from zero.
- **Analysis:** the CAD model can be used for mass properties, stress analysis and interference checks; a paper drawing cannot.
- **Storage and transfer:** files take no drawer space, do not fade, and can be sent instantly to suppliers and the shop floor.

02

The design process

Shigley's six phases, and where the computer helps in each

The design process is iterative: later phases feed back to earlier ones



- **1. Recognition of need:** someone notices a problem or an opportunity: a defect in an existing product, or a market for a new one.
- **2. Definition of problem:** the item is specified completely: physical and functional characteristics, cost, quality and operating performance.
- **3. Synthesis:** the designer proposes a concept or configuration that could meet the specification.
- **4. Analysis and optimization:** the concept is calculated and tested on paper or computer, improved, and redesigned; steps 3 and 4 repeat many times.
- **5. Evaluation:** the design is measured against the specification, often by building and testing a prototype.
- **6. Presentation:** the design is documented with drawings, material specifications and assembly lists.

Why iterative?

Any phase can send the design back to an earlier one. If evaluation finds the prototype too heavy, the designer returns to synthesis or analysis, not just to the end.

Source: Shigley's model of the design process, as presented in Groover & Zimmers

Design starts by recognising a need and defining the problem precisely

1. Recognition of need

The design process begins when someone realises that a need exists. Typical triggers:

- An engineer finds a defect in an existing product
- Customers complain, or sales report falling demand
- A market opportunity appears for a new product

At this stage the need is often vague: people know something must change, not yet what.

2. Definition of problem

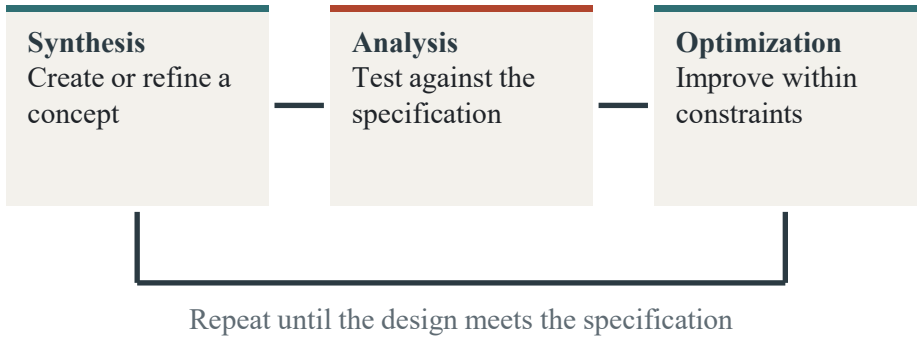
The need is turned into a complete specification of the item:

- Physical and functional characteristics (size, load, speed)
- Cost, quality and reliability targets
- Operating performance and conditions of use

Example: "a bracket that carries 500 N, weighs under 0.3 kg and is made by casting". Every later phase is checked against this specification.

Source: Groover & Zimmers, The design process

Synthesis, analysis and optimization form an iterative loop



Why it loops

Each analysis shows where the concept falls short.

The designer revises and analyses again.

CAD saves the most time here: analysis models come straight from the stored geometry.

Example

A shaft is first sized at 30 mm (synthesis). Stress analysis shows a safety factor of 4, so it is reduced to 25 mm and analyzed again (optimization), until the safety factor is just above the required 2.

- **Synthesis:** creating a design: choosing the form, size, material and arrangement of a component or system. It is the creative step.
- **Engineering analysis:** checking the synthesized design by calculation: stress, deflection, heat transfer, vibration, weight and so on.
- **Optimization:** adjusting the design within its constraints until it performs best, for example lowest weight for a required strength.
- **The loop:** a component is synthesized, analyzed, improved and redesigned; the cycle repeats until the design is optimized.

Evaluation checks the design against the specification; presentation documents it

- **Evaluation:** the finished design is compared with the specification written in the problem-definition phase.
- **Prototype testing:** a physical model is often built and tested to check performance, quality and reliability.
- **Presentation:** the design is communicated to manufacturing through drawings, material specifications and assembly lists.
- **Design database:** the documentation is stored as a database that production, purchasing and quality control all use.

With CAD

Design review and kinematics on screen reduce the number of physical prototypes. Automated drafting produces the presentation drawings directly from the model.

CAD supports four of the six design phases directly

Design phase	CAD function	What the computer does
Synthesis	Geometric modeling	Builds the mathematical description of the part's geometry
Analysis & optimization	Engineering analysis	Mass properties, finite element analysis, simulation
Evaluation	Design review & evaluation	Checks dimensions, interference, kinematics
Presentation	Automated drafting	Produces engineering drawings from the model

How the topics connect

1. Six phases with the Fig 4.2 flowchart. 2. Why the process is iterative. 3. The four CAD functions and the phase each supports. 4. Benefits: productivity, quality, fewer errors, manufacturing database.

- **Recognition and definition:** these phases depend on human judgment about the market and the customer; the computer helps little.
- **Synthesis → geometric modeling:** the designer builds the concept as a computer model of the part's shape.
- **Analysis → engineering analysis:** mass properties and finite element analysis run on the same model.
- **Evaluation → design review:** the model is checked on screen for dimensions, interference and motion before a prototype is built.
- **Presentation → automated drafting:** engineering drawings are generated from the model with dimensions and standard symbols.

Recognition of need and definition of problem remain human judgment. Source: Groover & Zimmers

Concurrent engineering runs design and manufacturing planning in parallel

- **Definition:** product design, manufacturing planning and other functions are carried out at the same time by one team, instead of one after another.
- **Team:** design, manufacturing, quality, purchasing, marketing and sometimes suppliers and customers work together from the start.
- **Why it helps:** manufacturing problems are found while the design can still be changed cheaply, so fewer changes are needed later.
- **Role in product development:** shorter time to market, lower cost, higher quality, and a product that is easier to make and assemble.
- **CAD's part:** a shared CAD database lets every member see and comment on the latest design at once.

Example

While the car door is being designed, the press-shop engineer points out that a deep draw will crack the sheet. The shape is changed on screen before any die is cut.

Concurrent engineering overlaps the stages that sequential engineering runs one after another

Aspect	Sequential engineering	Concurrent engineering
Workflow	Each department finishes, then passes the design "over the wall"	Departments work at the same time as one team
Communication	Little; one-way hand-offs	Continuous, cross-functional
Design changes	Found late, during manufacturing; costly	Found early, during design; cheap
Lead time	Long	Short
Cost and quality	Higher cost, more rework	Lower cost, fewer defects
Tools	Drawings handed between departments	Shared CAD/CAM database, DFMA

Memory aid

Sequential = relay race: one runner at a time. Concurrent = football team: everyone plays together towards one goal.

- **Sequential (over the wall):** each department finishes its work and passes it to the next: design, then process planning, then production. Problems found late send the design back.
- **Concurrent:** departments work in parallel and overlap their stages, sharing information continuously.
- **Time:** sequential is slow because stages wait for each other; concurrent shortens lead time.
- **Cost of changes:** sequential finds problems late, when changes are costly; concurrent finds them early, when changes are cheap.
- **Communication:** sequential has little contact between departments; concurrent needs teamwork and a common database.

Design for manufacture and assembly (DFMA) makes a product easier and cheaper to build

- **Aim:** design the product so that it is easy and cheap to make and to assemble, without losing function.
- **Fewer parts:** every part removed saves making, storing and assembling it. Combine parts where possible.
- **Standard parts:** use standard screws, bearings and materials that are cheap and available.
- **Easy handling and insertion:** add chamfers, symmetry and self-locating features so parts go in one way, from one direction (top-down).
- **Avoid separate fasteners:** use snap fits and press fits instead of screws where strength allows.
- **Suit the process:** choose shapes and tolerances that the chosen process (casting, moulding, machining) can make easily.

Example

A plastic toy cover held by four screws is redesigned with two snap-fit clips. Part count falls from 5 to 1 and assembly time from about 40 s to 5 s.

Reverse engineering builds a CAD model from an existing physical part

Physical part	Measure	Point cloud	CAD model	Use the model
An existing component, often with no drawings	CMM or 3D laser scanner captures surface points	Millions of x–y–z points describing the shape	Surfaces and solids fitted to the points	Manufacture, analyse or improve the part

Purpose
- Reproduce obsolete or worn parts with no drawings
- Study a competitor's product
- Improve or modify an existing design
- Check a manufactured part against its model

Example
A worn turbine blade from an old pump has no drawing. It is laser-scanned, a CAD surface model is built from the point cloud, and a CNC tool path is generated to make a new blade.

Example
The drawings for a 30-year-old pump impeller are lost. The impeller is scanned, a CAD model is built from the scan, and a new impeller is machined on a CNC machine.

- **Definition:** creating a CAD model and drawings from an existing physical part, working backwards from product to design.
- **How it is done:** the part is measured with a coordinate measuring machine or 3D scanner; the point cloud is turned into surfaces and then a solid model.
- **Purpose:** to remake parts with no drawings, to repair or improve old parts, to copy free-form shapes, or to inspect a made part against its model.
- **Result:** a CAD model that can be edited, analyzed and sent to CAM like any other design.

03

Applications of computers for design

Geometric modeling, engineering analysis, design review and automated drafting

Computers serve design through four functions



Geometric modeling
A mathematical description of the part's geometry, stored in the computer



Engineering analysis
Mass properties and finite element analysis run on the model



Design review & evaluation
Check dimensions, interference and motion before building



Automated drafting
Engineering drawings created directly from the database

Key point
Learn the four functions as a set: each has one main purpose, and the next slides give an example of each.

- **Geometric modeling:** the mathematical description of the part's shape is built and stored in the computer. Everything else uses this model.
- **Engineering analysis:** calculations such as mass properties or stress analysis are run on the model to check the design.
- **Design review and evaluation:** the designer checks the model for size, fit, interference and motion before anything is built.
- **Automated drafting:** engineering drawings are produced directly from the model and plotted on paper.

Source: Groover & Zimmers, Application of computers for design

Geometric modeling builds the part from three kinds of commands

- **Generate elements:** commands create basic elements: points, lines, circles, arcs, and in 3D, cubes, cylinders and other solids.
- **Transform elements:** commands scale, rotate, move or mirror elements already created.
- **Join elements:** commands combine simple elements into the full shape of the part.
- **Stored as numbers:** the computer keeps the model as coordinates and equations, so it can be viewed from any angle and used for calculation.

Example

To model a flange: draw a circle of 100 mm (generate), add a 20 mm circle for the bore and copy a bolt hole round the centre six times (transform), then extrude the profile to 15 mm thick (join).

Three model types describe the same part with increasing completeness

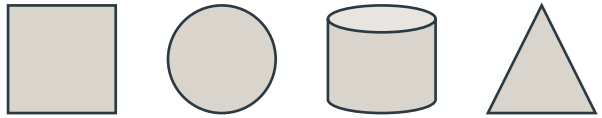
- **Wire-frame:** the part is shown only by its edges, like a model made of wire. Simple and fast, but the picture can be ambiguous and has no surfaces or volume.
- **Surface model:** the faces between edges are defined, so hidden lines can be removed and NC tool paths can follow the surface. The inside is still empty.
- **Solid model:** the part is a complete solid with an inside and outside. Volume, mass and centre of gravity can be calculated, and sections cut.
- **Choice:** more complete models need more computing power but answer more engineering questions.

Remember

Only the solid model can compute mass properties directly. A wire-frame cube and a wire-frame open box look exactly the same.

Solid modeling combines simple primitives with Boolean operations

Building blocks (primitives)



Block Sphere Cylinder Cone

Boolean operations

Union (+)
Joins two solids into one

Difference (−)
Subtracts one solid from another, e.g. drilling a

Intersection (∩)
Keeps only the common volume

Example: block − cylinder = plate with a hole. This is constructive solid geometry (CSG).

Example

A plate with a hole = block − cylinder. An L-bracket = block ∪ block. A rounded end = block ∪ half-cylinder.

- **Primitives:** simple solids the system already knows: block, cylinder, cone, sphere, wedge, torus.
- **Union (+):** two solids are joined into one, keeping all of both volumes.
- **Difference (−):** one solid is subtracted from another, for example to cut a hole.
- **Intersection (∩):** only the volume common to both solids is kept.
- **CSG tree:** the computer records the primitives and the Boolean operations as a tree, so the part can be edited step by step.

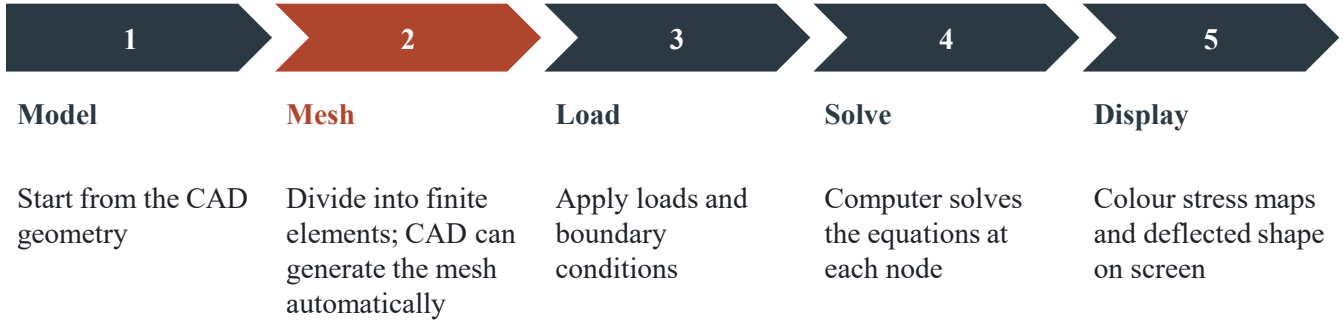
Engineering analysis runs on the stored model instead of a separate hand calculation

- **Mass properties:** from the solid model the computer calculates surface area, volume, weight, centre of gravity and moment of inertia.
- **Finite element analysis:** the part is divided into small elements and stress, deflection, heat flow or vibration is calculated.
- **Other analyses:** tolerance analysis, kinematics of mechanisms, and specialised programs for a company's own products.
- **Benefit:** the same model is used, so no separate hand calculation or physical test is needed at the early stage.

Example

A designer models a crank arm and asks for its mass: 0.84 kg. After adding a pocket, the mass falls to 0.71 kg, and FEA confirms the stress is still below the allowable limit.

FEA turns the model into a mesh, solves it and shows the results



Results feed straight back into the synthesis–analysis loop.

Why a mesh?
A complex shape cannot be solved by one formula. Simple elements can, and the computer combines thousands of them to approximate the whole part.

- **1. Model:** the geometry of the part is taken from the CAD model.
- **2. Mesh:** the part is divided into many small elements joined at nodes. Smaller elements give more accurate results.
- **3. Loads and supports:** forces, pressures, temperatures and fixed points are applied as boundary conditions.
- **4. Solve:** the computer solves a large set of equations for the displacement at every node.
- **5. Results:** stress and deflection are shown as colour maps, so the most stressed region is seen at once.

Source: Groover & Zimmers, Engineering analysis

Design review catches errors on screen before anything is built

- **Automatic dimensioning:** the system measures distances and angles on the model exactly, so dimensions do not need to be checked by hand.
- **Interference checking:** assembled parts are checked to see whether any two occupy the same space.
- **Layering:** different details (e.g. the casting and the machined surfaces) are kept on separate layers and overlaid to compare.
- **Kinematics:** moving parts such as linkages and hinges are animated on screen to check that they clear each other.

Example

In an engine assembly, interference checking shows that a bolt head hits the oil pan by 1.5 mm. The bolt is changed before any part is made.

Automated drafting produces engineering drawings straight from the database

- **What it does:** produces hard-copy engineering drawings directly from the CAD database.
- **Views:** orthographic, sectional and isometric views are generated from the model automatically.
- **Dimensioning:** dimensions, tolerances and labels are added by the system from the stored geometry.
- **Standards:** standard symbols, title blocks and hatching are taken from libraries.
- **Productivity:** drafting is often the most time-consuming manual task; CAD can make it several times faster, especially for revisions.

Remember

Change the model, and the drawing changes with it. Drawings never disagree with each other because they all come from one model.

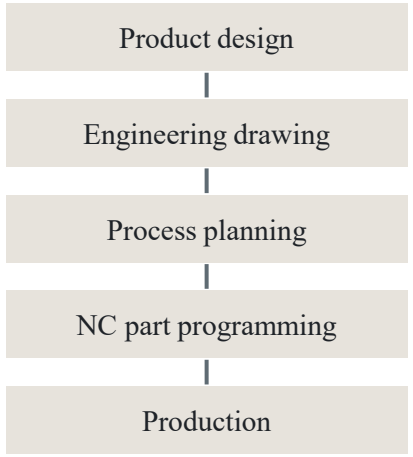
04

Creating the manufacturing database

How one design file replaces repeated re-drawing between design and production

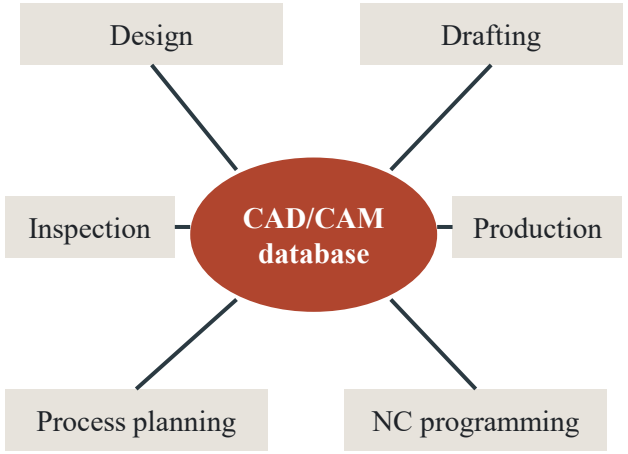
CAD/CAM replaces a chain of separate documents with one shared database

Conventional



Each department re-reads the drawing and recreates its own data

CAD/CAM



In short

A manufacturing database is the integrated CAD/CAM database that contains all product data created in design and needed in production, shared by every department.

- **Definition:** a single computer store holding all the data needed to design and make a product: geometry, dimensions, materials, bill of materials and specifications.
- **Without CAD/CAM:** design produces a drawing; each later department reads it and recreates its own data by hand. This repeats work and introduces errors.
- **With CAD/CAM:** the design data is stored once and every department uses the same database directly.
- **Result:** less duplication, fewer errors, faster changes, and a true link between CAD and CAM.

A manufacturing database holds four classes of data

- **Product data:** what the product is: part geometry, dimensions, tolerances, materials and the bill of materials. Most of it comes straight from the CAD model.
- **Production data:** how the product is made: process plans, routings, the machine tools used, cycle times and NC programs.
- **Operational data:** when and how many are made: production schedules, order quantities, lot sizes and machine assignments.
- **Resources data:** what the factory has to make it with: machines, tooling, fixtures, workers and materials available.

Key point

The four classes are product, production, operational and resources data; each is shown here with an example.

The design database feeds every manufacturing function downstream

- **Process planning:** the part geometry and material decide which operations, machines and sequence are needed.
- **NC part programming:** tool paths are computed directly from the stored surfaces, so the programmer does not re-enter the geometry.
- **Tool and fixture design:** jigs, fixtures, moulds and dies are designed around the part model.
- **Quality control:** inspection programs for coordinate measuring machines are written from the model's dimensions and tolerances.
- **Assembly and purchasing:** the bill of materials and assembly structure drive purchasing, scheduling and robot programs.

Key idea

Data is created once in design and reused everywhere. That reuse is what joins CAD to CAM.

CAD pays off both in the design office and on the shop floor

- **Productivity:** designers produce designs and drawings faster; productivity gains of 3:1 or more are reported for drafting.
- **Shorter lead time:** designs, changes and drawings are finished sooner, so the product reaches the market earlier.
- **Fewer errors:** dimensions come from the model and parts come from libraries, so mistakes in copying and calculation drop.
- **Better designs:** more alternatives can be analyzed and optimized in the same time.
- **Standardisation:** standard parts, symbols and drawing formats are applied automatically.
- **Link to manufacturing:** the design database is used directly by process planning, NC programming and quality control.

Key point

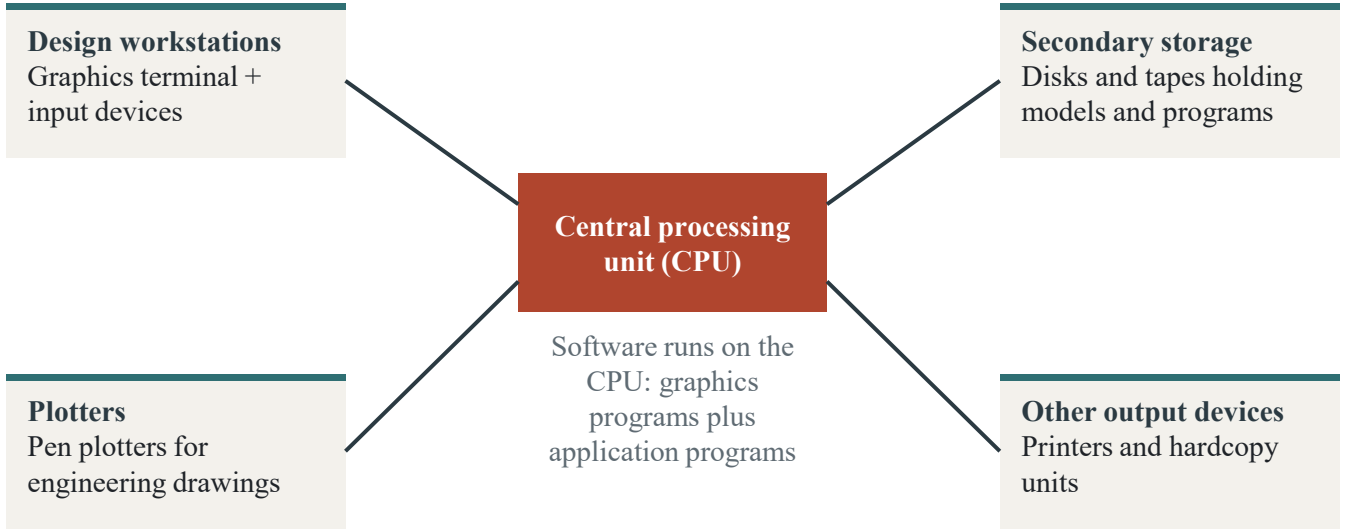
Every benefit has the same source: the design exists once as exact data, and every drawing and calculation is taken from it.

05

CAD hardware and the design workstation

The parts of a CAD system and what the designer's workstation must do

A CAD system links workstations, a CPU, storage and output devices



Key point

Read the diagram from the CPU outwards: every block exchanges data with the CPU, which coordinates the whole system.

- **Central processing unit:** the computer that runs the CAD software, does the calculations and controls the other units.
- **Workstations:** where designers work: each has a graphics terminal plus input devices. Several can share one CPU.
- **Secondary storage:** disks and tapes that hold the programs, models and the design database.
- **Output devices:** plotters and printers that turn the stored design into paper drawings.
- **Block diagram:** draw the CPU in the centre with arrows to workstations, storage, and plotters/printers; arrows show the flow of data.

Source: Groover & Zimmers, Fig. 5.1 configuration of a CAD system

A design workstation must do five things

- **Interface with the CPU:** the workstation passes the designer's commands to the computer and receives results back.
- **Generate a steady image:** the picture must stay on screen without flicker, so the designer can study it.
- **Digital description:** what the designer draws is converted into numbers (coordinates) that the computer can store.
- **Translate commands:** each command typed, picked or pointed at is turned into an operation of the system.
- **User–system communication:** menus, prompts and messages make the dialogue between designer and computer easy.

Why it matters

The workstation is the only part of the system the designer sees and touches. If it is slow or confusing, the whole CAD system is.

The workstation is a graphics terminal plus operator input devices

Graphics terminal

The output side of the workstation: it shows the designer the image of the part.

- Most CAD terminals use a cathode ray tube (CRT)
- The image is drawn by random scan (line strokes) or raster scan (dots, row by row)
- Three types: storage tube, refresh (vector) and raster
- It must give a steady, flicker-free picture

Operator input devices

The input side: they let the designer give commands and data.

- Keyboard: alphanumeric keys for text and values, function keys for commands
- Cursor controls: thumbwheels, joystick, tracker ball, light pen, mouse
- Digitizer or tablet: enters points from an existing drawing

Together with the terminal they form the designer's two-way link with the CPU.

The next two sections take each part in turn: terminals first, then input devices.

CAD software has to satisfy six basic requirements

- **Simplicity:** a designer should learn the system quickly; commands use plain names and do what is expected.
- **Consistency:** similar commands work the same way everywhere, so learning one teaches the others.
- **Completeness:** every function the designer needs is available; no task forces a work-around.
- **Robustness:** a wrong input gives a clear message and the work is not lost.
- **Performance:** response is fast; any delay should come from the hardware, not poor software.
- **Economy:** the program is compact and efficient, so it runs well on the available computer.

Memory aid

S-C-C-R-P-E: Simple, Consistent, Complete, Robust, Performs, Economical.

06

Graphics terminals

How the image is generated, and the three terminal technologies used in CAD

Terminals draw images in one of two ways: random scan or raster scan

- **Stroke writing (random scan):** the beam moves directly from one end point of a line to the other, drawing only the lines in the picture, in any order.
- **Result of stroke writing:** very sharp, smooth lines; best for line drawings; hard to fill solid areas.
- **Raster scan:** the beam sweeps every row of the screen from left to right, top to bottom, switching on only the dots (pixels) that belong to the image.
- **Result of raster scan:** solid areas, shading and colour are easy; sloping lines show small steps (the staircase effect).

Analogy

Stroke writing is like drawing with a pen. Raster scan is like a TV picture or a cross-stitch made of dots, row by row.

A cathode ray tube turns an electron beam into a visible image

1 → 2 → 3 → 4

Electron gun

A heated cathode emits electrons. A control grid with a negative voltage regulates how many pass, which sets the brightness of the spot.

Focusing system

Focusing and accelerating anodes squeeze the electrons into a fine beam and speed it towards the screen, so the spot is small and sharp.

Deflection system

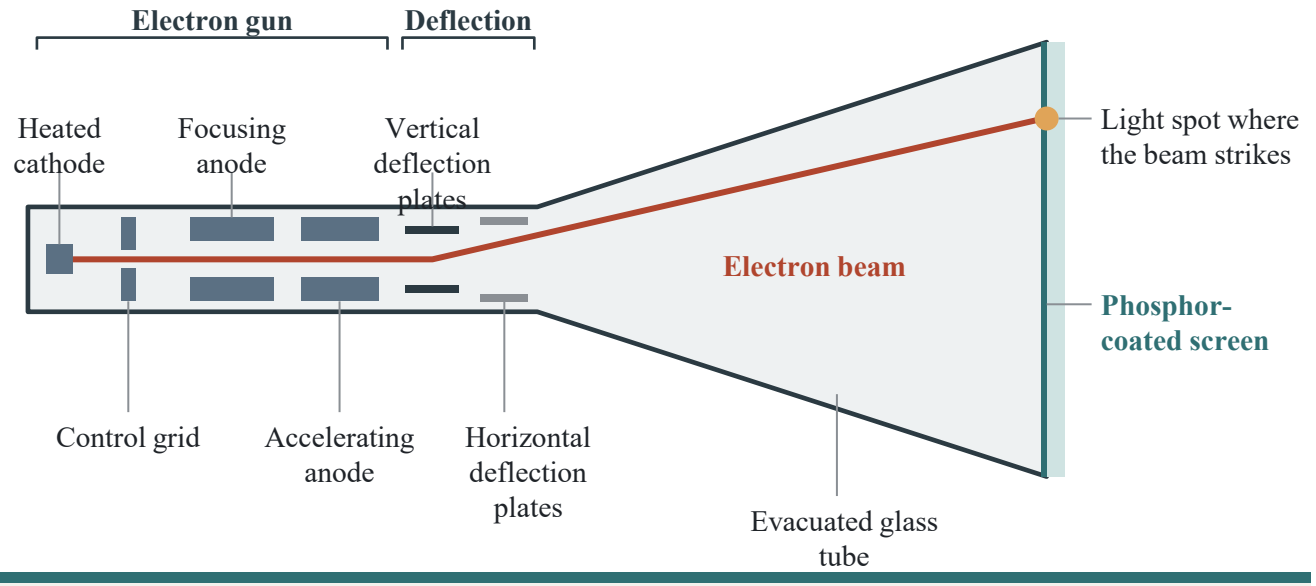
Vertical and horizontal deflection plates bend the beam with electric fields, steering it to any x–y point on the screen.

Phosphor screen

The inside of the screen is coated with phosphor. Where the beam strikes, the phosphor glows and a point of light is seen.

The glow fades quickly, so the image must be stored or redrawn. How it is kept visible separates the terminal types.

In a CRT, the electron gun forms a beam that deflection plates steer onto the phosphor screen



Key point

Follow the beam from left to right: it is formed in the gun, steered by the deflection plates and made visible on the phosphor screen.

- **Heated cathode:** heating the cathode makes it emit electrons.
- **Control grid:** a negative voltage on the grid controls how many electrons pass, which sets the brightness.
- **Focusing and accelerating anodes:** these focus the electrons into a fine beam and speed it up towards the screen.
- **Deflection plates:** voltages on the vertical and horizontal plates bend the beam so it can reach any point on the screen.
- **Phosphor screen:** where the beam strikes, the phosphor coating glows and a spot of light is seen. The glow fades quickly, so it must be renewed.

Source: Groover & Zimmers, Graphics terminal

CAD uses three types of graphics terminal

Direct-view storage tube (DVST)
Uses random scan. A fine storage grid behind the screen holds the charge pattern, so the image stays visible without being redrawn.
• Very high resolution, no flicker, low cost
• Cannot erase a single line: the whole screen is cleared and redrawn
• No colour and no animation

Refresh (vector)
Uses random scan. The lines are kept in a display list in memory and redrawn 30–60 times per second so the image does not fade.
• Selective erase and smooth dynamic motion
• Works with a light pen
• Flickers when the picture has too many lines; expensive

Raster scan
The picture is stored as pixels (dots) in a frame buffer, and the beam sweeps the screen row by row like a television.
• Colour, shading and solid areas
• Selective erase; cost falls as memory gets cheaper
• Sloping lines look stepped (staircase effect)

Source: Groover & Zimmers, Graphics terminal

The storage tube keeps the image on the screen without redrawing it

- **How it works:** the beam writes lines by random scan onto a fine storage grid behind the screen. The grid holds the charge pattern, and flood electrons keep the image glowing.
- **No refresh needed:** the image stays on screen as long as needed without being redrawn, so a cheap, slow computer link is enough.
- **Erasing:** a single line cannot be erased; the whole screen must be cleared and redrawn, which takes time.
- **Use:** low-cost terminals for complex line drawings that change rarely.

Pros and cons

+ Flicker-free, high resolution, low cost
− No selective erase
− No colour, low brightness
− Cannot show motion (no animation)

The refresh terminal redraws its image from a display list many times a second

- **Display list:** the lines of the picture are stored as a list of commands in a memory called the display buffer.
- **Refresh:** the display processor reads the list and redraws every line 30 to 60 times per second, so the eye sees a steady image.
- **Selective erase:** removing an item from the list erases just that line, so editing and animation are easy.
- **Flicker:** if the picture has too many lines, they cannot all be redrawn in time and the image flickers.

Pros and cons

+ Selective erase and animation
+ Bright, sharp lines; works with a light pen
− Flickers with complex pictures
− Expensive display processor

The raster terminal stores the picture as pixels in a frame buffer

- **Pixels:** the screen is divided into a grid of small dots, for example 1024 × 1024 pixels.
- **Frame buffer:** a memory holds one value for every pixel, saying whether it is on (and which colour).
- **Scanning:** the beam sweeps the screen row by row, like television, and lights each pixel according to the frame buffer.
- **Scan conversion:** lines and circles from the model are converted into the pixels that best approximate them.
- **Staircase effect:** sloping lines appear slightly stepped because they are made of square pixels; higher resolution reduces this.

Why it won

Raster terminals give colour, solid shading, selective erase and no flicker, and they use cheap TV technology. Memory cost is their main drawback.

Raster terminals combine the strengths of the other two types

Feature	Storage tube (DVST)	Refresh (vector)	Raster scan
Image generation	Random scan	Random scan	Raster scan
Image kept by	Storage grid on tube	Redrawing display list	Frame buffer in memory
Selective erase	No	Yes	Yes
Colour	No	Limited	Yes
Animation	No	Yes	Yes
Line quality	Sharp	Sharp	Stepped diagonals
Flicker	None	With complex images	None

Source: Groover & Zimmers, Graphics terminal

Colour raster displays use bit planes: n planes give 2^n colours

Each pixel gets n bits of memory, one from each bit plane. The bits select the colour of that pixel; three electron guns (red, green, blue) behind a shadow mask produce it on screen.

2	8	256	16.7M
colours from 1 bit plane	**colours from 3-bit planes**	**colours from 8-bit planes**	**colours from 24-bit planes**
Monochrome: on / off	One bit each for red, green and blue	Often through a colour lookup table	8 bits per gun: full colour

Memory grows with colour: a 1024 × 1024 display with 8 bit planes needs 8 million bits of frame buffer.

Practice

A 512 × 512 display has 3 bit planes. How many colours, and how much memory? Answer: $2^3 = 8$ colours; 512 × 512 × 3 = 786,432 bits = 96 KB.

- **Bit plane:** one layer of frame-buffer memory with one bit per pixel. One plane gives 2 states: on or off (black and white).
- **Rule:** with n bit planes each pixel has n bits, so it can show 2^n colours or grey levels.
- **Worked example:** 4 bit planes → $2^4 = 16$ colours. 8 bit planes → $2^8 = 256$ colours. 24 bit planes → $2^{24} \approx 16.7$ million colours.
- **Memory needed:** memory = pixels × bit planes. A 1024 × 1024 screen with 8 planes needs 1024 × 1024 × 8 bits = 1 MB.

07

Operator input devices

How the designer enters commands, data and geometry into the system

Input devices fall into three categories

Cursor control devices

Move a cursor (cross-hair or arrow) on the screen to point at a location or pick an element.

- Thumbwheels, cursor keys, joystick
- Tracker ball, light pen, mouse

Used for picking, positioning and choosing from menus.

Digitizers

A flat board with a wire grid under its surface senses the position of a stylus or puck and sends x–y coordinates to the computer.

- Digitizer table: enters existing drawings
- Tablet: cursor control and menus

Keyboards

The basic input device on every workstation.

- Alphanumeric keys enter text, dimensions and command names
- Function keys call frequent commands (draw line, zoom, rotate) with one press

Most workstations combine all three: a keyboard, a cursor control device and a tablet or digitizer.

Source: Groover & Zimmers, Operator input devices

Six devices move the cursor, each suited to a different task

Device	How it works	Best for
Thumbwheels	Two wheels move the cross-hair cursor in x and y	Precise horizontal and vertical moves
Direct cursor keys	Arrow keys step the cursor up, down, left, right	Small adjustments
Joystick	A lever tilted in any direction moves the cursor	Fast, free movement
Tracker ball	A ball rotated by the palm drives the cursor	Smooth movement in a small space
Light pen	A pen-shaped sensor pointed at the screen	Picking elements on refresh displays
Mouse	Moved on a flat surface; motion becomes cursor motion	General pointing and menu selection

Two jobs

Every cursor device does two things: positioning (place the cursor at a point) and pointing/picking (identify an element already on screen).

- **Cursor:** a small cross-hair or arrow on the screen that marks a position. Moving it lets the designer point at a location or pick an element.
- **Thumbwheels and cursor keys:** move the cursor separately in x and y; slow but precise for straight moves.
- **Joystick and tracker ball:** move the cursor freely in any direction by tilting a lever or rolling a ball; fast and need little desk space.
- **Mouse:** moved on the desk; its motion is copied by the cursor. Buttons pick items and menus.
- **Light pen:** pointed directly at the screen to pick a displayed element (next slide).

The light pen detects light from the screen to pick an element

- **Construction:** a pen-shaped holder with a photocell (light sensor) and lens at the tip, connected by a cable to the terminal.
- **Sensing:** on a refresh display each line is redrawn many times a second. When the beam passes the point under the pen tip, the photocell sees a flash of light.
- **Identifying the element:** the pulse is sent to the computer at that exact moment. The computer knows which item in the display list it was drawing then, so that item is selected.
- **Tracking:** to locate an empty point, a tracking cross is shown and the pen follows it; the cross position gives the coordinates.

In short

A light pen contains a photocell that detects the light emitted when the CRT beam refreshes the point under the pen. The timing of the pulse identifies the displayed element.

Digitizers convert positions on a drawing into coordinates

- **Purpose:** to enter existing drawings, sketches or maps into the CAD database by tracing them.
- **Construction:** a flat table with a fine grid of wires under the surface, and a stylus or puck with cross-hairs.
- **Working:** when the puck is placed on a point and a button pressed, the grid senses its position and sends x–y coordinates to the computer.
- **Tablet:** a small digitizer beside the terminal; moving the pen moves the cursor, and part of the surface holds a menu of commands.

Example

An old paper drawing of a gasket is taped to the digitizer. The operator clicks each corner and arc point, and the outline appears in the CAD system ready to edit.

Keyboards enter text, numbers and frequently used commands

- **Alphanumeric keyboard:** a typewriter keyboard used to type commands, file names, text notes and exact numeric values such as coordinates and dimensions.
- **Function keyboard:** a set of buttons, each programmed to call one command (draw circle, zoom, rotate). One press replaces typing a whole command.
- **When used:** keyboards are best for exact values and text; cursor devices are best for pointing and picking.

Key point

Input devices fall into cursor controls, digitizers and keyboards; output devices include plotters, printers, hard-copy units and COM.

08

Output devices, CPU and secondary storage

Producing hardcopy, processing the model, and storing the design database

Pen plotters draw engineering drawings with ink on paper

- **Purpose:** to produce accurate, large engineering drawings in ink from the CAD database.
- **Drum plotter:** the paper is wrapped on a drum that rotates for one axis while the pen moves along the drum for the other axis. Compact and suits long drawings.
- **Flatbed plotter:** the paper lies flat and still; the pen moves in both x and y over it. Most accurate, but takes more floor space.
- **Pens:** several pens of different colour or thickness are picked up automatically for different line types.

Compare

Drum: paper moves in one axis, small footprint, lower accuracy.
Flatbed: paper fixed, pen moves in both axes, higher accuracy, large size.

Other output devices trade line accuracy for speed or convenience

- **Printers:** dot-matrix, inkjet and laser printers give fast, cheap copies; line quality is lower than a pen plotter.
- **Electrostatic plotter:** prints the whole drawing as a pattern of dots, row by row like a raster display; much faster than a pen plotter.
- **Hard-copy unit:** copies the picture on the screen directly onto paper in a few seconds.
- **Computer output to microfilm (COM):** stores drawings on microfilm for compact long-term archiving.

Trade-off

Pen plotters: best line accuracy, slow.
Raster-type printers and plotters: fast, slight staircase on sloping lines.

The CPU runs the software and manages the whole CAD system

The CPU executes graphics and application programs, manages the database and controls the workstations and peripherals.

Host-based (mainframe)

One large central computer runs the CAD software for many terminals.

- Large memory and computing power
- Response slows when many designers work at once
- A host failure stops every workstation

Minicomputer (turnkey)

A dedicated minicomputer serves a small group of workstations. The vendor supplies hardware and software as one ready-to-use package.

- Fast response for a design group
- Easy to buy and install

Stand-alone workstation

Each workstation has its own processor, memory and storage.

- Response does not depend on other users
- Networked to share the database and plotters
- Basis of today's distributed CAD

The trend is toward distributed processing: computing power moves from the central host to each workstation.

The CPU controls the whole CAD system and does its computing

- **Runs the software:** executes the graphics and engineering programs that the designer calls from the workstation.
- **Computation:** performs the calculations: geometry, transformations, mass properties, finite element analysis.
- **Control:** directs the flow of data between workstations, storage and output devices, like a traffic controller.
- **Memory management:** holds the active model and programs in main memory and moves data to and from secondary storage.
- **Serving several users:** in a host-based system one CPU serves many workstations by time-sharing.

Parts of the CPU

Control unit: decodes instructions and directs operations.
Arithmetic logic unit (ALU): does calculations and comparisons.
Main memory: holds the program and data in use.

Secondary storage holds the models, programs and design database

Medium	Access	Typical use
Magnetic tape	Sequential, slow	Backup and archiving of design files
Hard disk	Random, fast	Active models, programs and the database
Floppy disk	Random, moderate	Transferring files between systems
Optical disk	Random, high capacity	Large archives of drawings

Example

Today's model is opened from the disk in seconds. Every night, the whole design database is copied to tape and stored in a safe place.

- **Why it is needed:** main memory is limited and is lost when the power is off. Programs, models and the design database must be kept permanently.
- **Magnetic disk:** direct (random) access: any file can be reached quickly. Used for work in progress and the active database.
- **Magnetic tape:** sequential access: data is read in order, so it is slow to find a file but cheap. Used for back-up and archives.
- **Capacity versus speed:** faster storage costs more per byte; systems combine disk for speed with tape for cheap bulk storage.

Secondary storage keeps data when power is off; main memory holds only the work in progress

09

BPUT previous year questions

Module I questions grouped by marks

2-mark questions

1. State the benefits of CAD.
2. Explain the working principle of a light pen used as an input device in CAD.
3. Distinguish between stroke writing and raster scan.
4. Draw the flowchart of the basic design process.
5. What is meant by a manufacturing database?
6. What do you understand by synthesis and engineering analysis in product design?
7. What is the role of concurrent engineering in product development?
8. What is reverse engineering? Explain its purpose with an example.
9. State four classes of data used in manufacturing.

6-mark questions

1. What are the benefits of using CAD over manual drafting?
2. Explain, with the help of a block diagram, the hardware structure of a CAD/CAM workstation.
3. Elaborate on the basic requirements that CAD software has to satisfy.
4. Explain the functions of the central processing unit (CPU) in detail.
5. What are the applications of computers for design?
6. Explain the raster scan technique for generating images.
7. State and explain the various CAD input and output devices.
8. Explain the principles of design for manufacture and assembly (DFMA) with suitable examples.
9. Compare concurrent engineering and sequential engineering methodology.

16-mark question

> State and explain the general design procedure. Explain how computers aid the design process for improved design.

CAD / CAM

Graphics Software and

MODULE II

ded Design and Manufacturing (CAD/CAM)

es, transformations and solid modeling

Reference: Groover & Zimmers, CAD/CAM, Ch. 6; Radhakrishnan & Subramanyam, CAD/CAM/CIM

Module II roadmap

01	**Graphics software configuration**	Three software parts and the functions of a graphics package
02	**Constructing the geometry**	Entities, definition methods, editing and sketch constraints
03	**Curves**	Analytic and synthetic curves: Hermite, Bézier, B-spline
04	**Geometric transformations**	Translation, scaling, rotation, reflection, concatenation; solved numerical
05	**Database structure and content**	Geometry, topology, attributes and Boolean operations
06	**Wireframe, surface and solid models**	Surface types, CSG, B-rep and the rectangular-block example
07	**Modern modeling methods**	Parametric, feature-based, extrude/revolve, kinematic simulation
08	**Data exchange and BPUT PYQ**	IGES, STEP and mark-wise previous year questions

By the end of Module II you should be able to

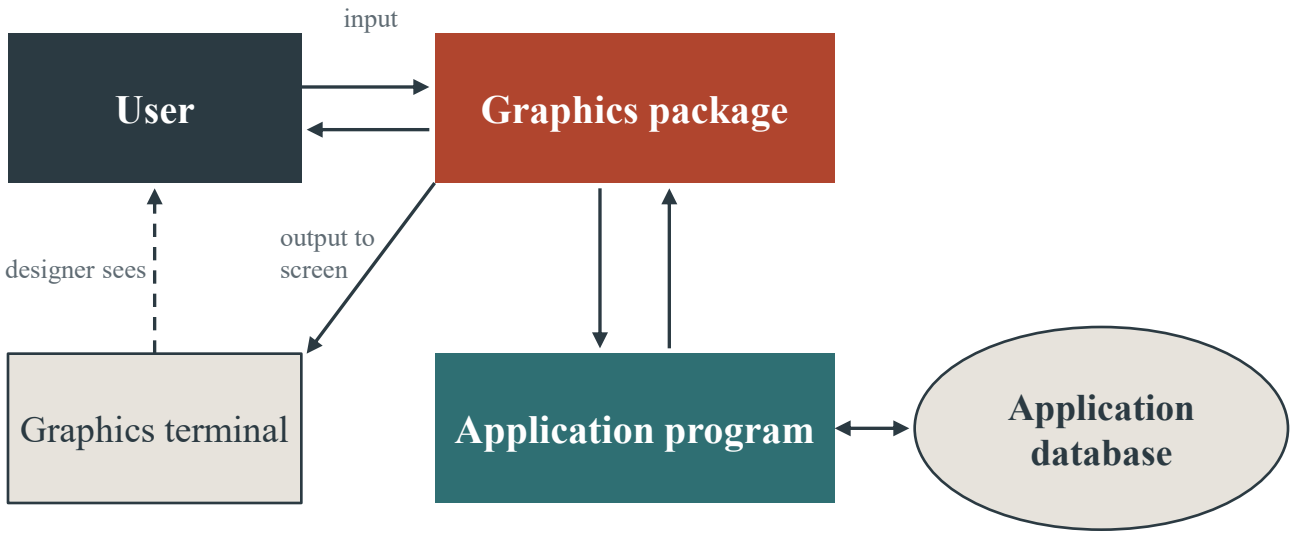
- **Describe** the software configuration of a graphics system and the functions of a graphics package
- **Name** the basic CAD entities and explain constraints in a 2D sketch
- **Differentiate** analytic and synthetic curves, and explain Hermite, Bézier and B-spline curves
- **Apply** translation, scaling and rotation matrices to a line and solve a line-transformation numerical
- **Compare** wireframe, surface and solid models, and CSG with B-rep
- **Explain** parametric and feature-based modeling, extrude, revolve and kinematic simulation
- **Explain** the IGES and STEP data exchange standards

01

Graphics software configuration

How the graphics package, the application program and the database work together

A graphics system has three software parts that work around one database

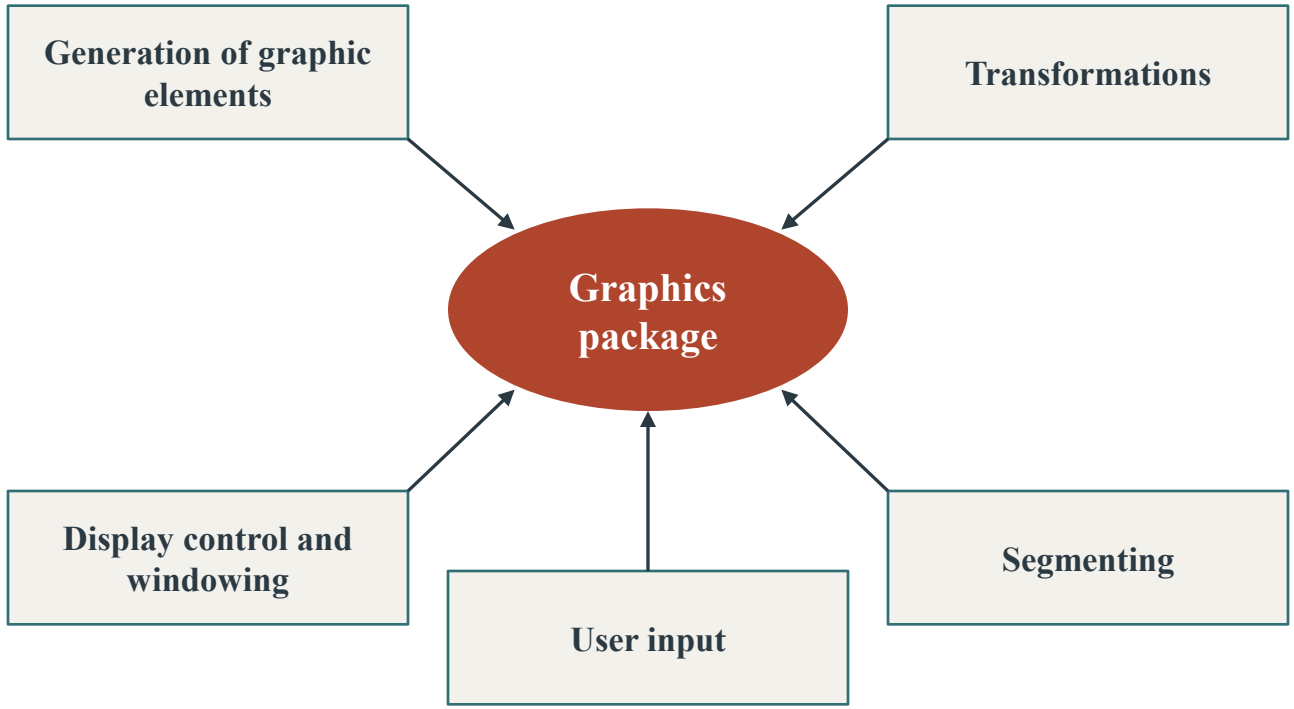


Model of graphics software configuration (after Groover & Zimmers, Fig. 6.1)

Example
Drawing a fillet: the designer picks two lines (user input). The graphics package passes the request to the application program, which computes the arc, stores it in the database and asks the package to display it.

- **Graphics package:** the basic graphics software. It controls the terminal, accepts input from the user and draws the image. It is the link between user and computer.
- **Application program:** the program written for the user's engineering job, such as drafting, analysis or NC programming. It uses the graphics package to talk to the user.
- **Application database:** stores the mathematical description of the model: geometry, attributes and relations. The image on screen is generated from it.
- **Key idea:** the image is only a view of the database; changing the model means changing the database.

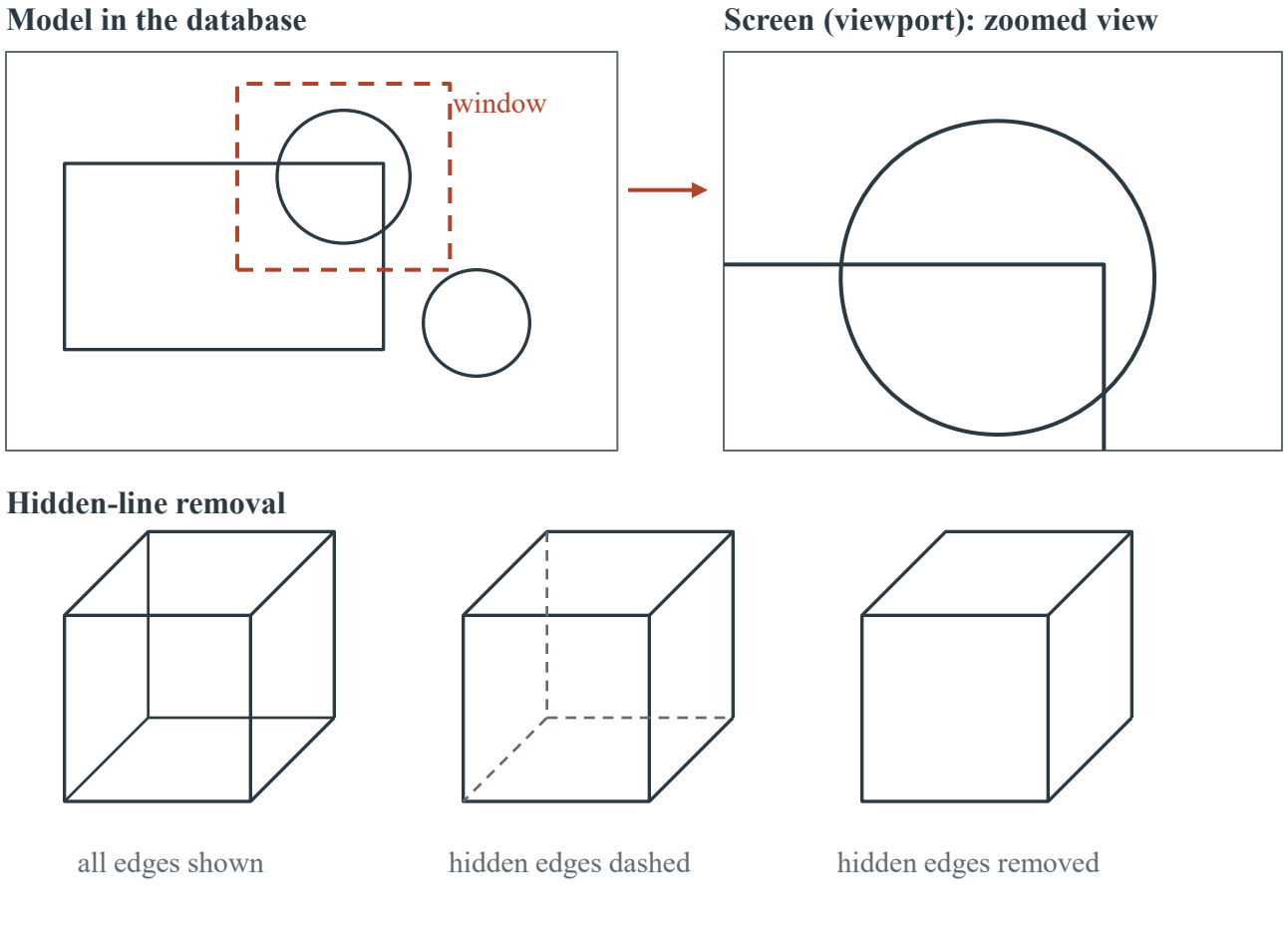
A graphics package performs five groups of functions



- **Generation:** creates points, lines, arcs, circles, curves and text, the basic elements of every model.
- **Transformations:** move, scale, rotate or mirror elements to change the model or the view.
- **Display control and windowing:** choose what part of the model is seen, at what size, and from which direction.
- **Segmenting:** pick a group of elements to replace, delete or highlight without redrawing the rest.
- **User input:** accept commands and data from the keyboard, mouse, tablet or menu.

Key point
Each block is one group of functions. Example: generation draws a circle from its centre and radius; a transformation then moves or scales it.

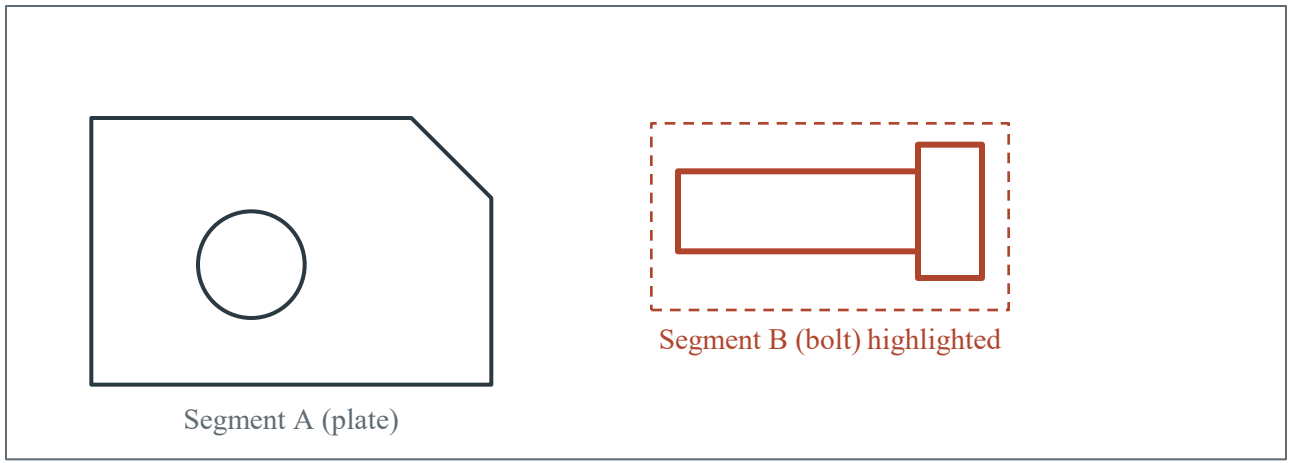
Display control lets the designer view any part of the model at any scale



- **Windowing:** the designer draws a rectangle (the window) around the part of the model to be seen. Only that part is mapped onto the screen area called the viewport.
- **Zoom and pan:** zoom enlarges or reduces the view; pan slides the window across the model. The model itself does not change, only the view.
- **Viewing direction:** the same 3D model can be shown in front, top, side or isometric view.
- **Hidden-line removal:** lines that would be behind a surface are dashed or removed, so a 3D object can be read without confusion.

Segmenting and user input functions let the designer pick and change parts of the image

A drawing made of segments; one segment selected



Delete | **Replace** | **Move** | **Highlight**

User input devices
Keyboard (commands and values), mouse or tracker ball (pointing), digitizer tablet with menu (commands and coordinates), light pen (picking on refresh screens).

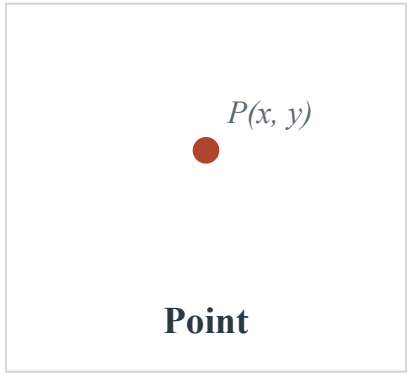
- **Segment:** a single element or a logical group of elements, such as a bolt or a hole, that the software treats as one unit.
- **Segmenting functions:** replace, delete, move or highlight one segment without regenerating the rest of the image.
- **Why it matters:** on a refresh or raster terminal only the changed segment is redrawn, so editing is fast. A storage tube cannot erase selectively.
- **User input functions:** read the designer's commands and data and pass them to the application program.

02

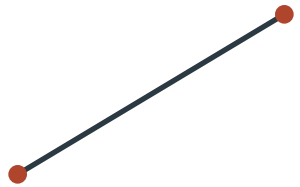
Constructing the geometry

Building a model from basic entities, editing it, and controlling it with constraints

Every CAD model is built from a small set of basic entities



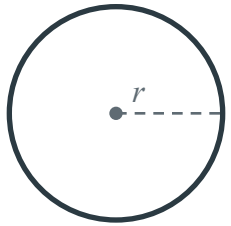
Point



Line



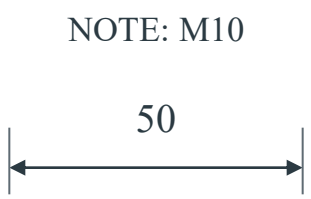
Arc



Circle



Spline curve

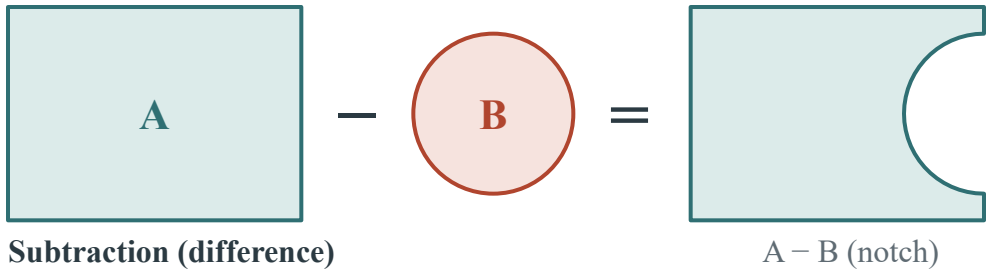


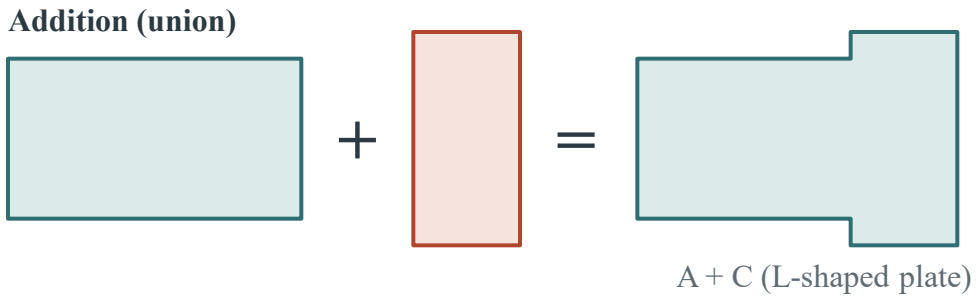
Text and dimension

- **Entity:** the smallest element the CAD database stores and draws: it has a type, coordinates and attributes such as layer, colour and line style.
- **Point and line:** a point is stored as (x, y, z); a line by its two end points.
- **Arc and circle:** stored by centre, radius and, for an arc, start and end angles.
- **Spline:** a smooth curve through or near a set of points, used for free-form shapes.
- **Text and dimensions:** annotation entities that add sizes and notes to the drawing.

In short
Basic entities: point, line, arc, circle, spline curve, text and dimensions (any four).

Complex shapes are built by adding and subtracting simple elements





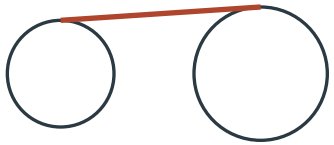
Example (Groover Fig. 6.2)
A rectangular plate with a circular cut-out is made by drawing the rectangle A, the circle B, and asking the system for A − B.

- **Building blocks:** the designer does not draw the final outline line by line; it is assembled from simple elements.
- **Addition:** joins two elements into one region (union).
- **Subtraction:** removes one element from another (difference), e.g. holes, notches, slots.
- **Intersection:** keeps only the area common to both elements.
- **Benefit:** the model stays exact: edges are computed by the system, not drawn by hand, and the same idea extends to 3D solids.

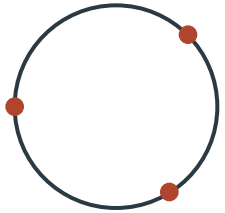
Each entity can be defined in several ways



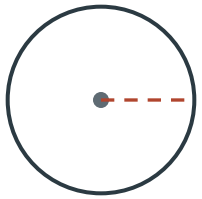
Line: two end points



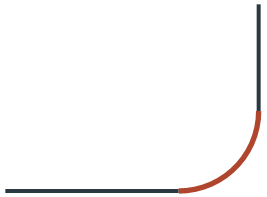
Line: tangent to two circles



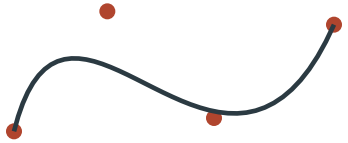
Circle: three points



Circle: centre and radius



Arc: fillet between lines

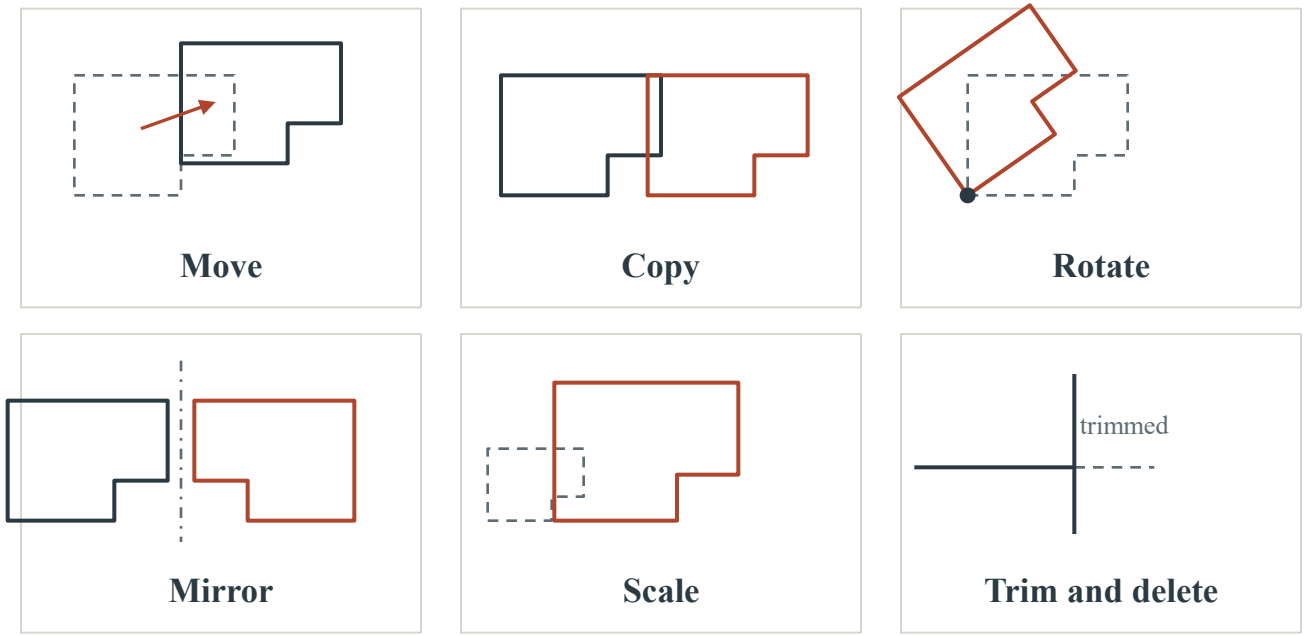


Curve: spline from control points

Why several methods?
The designer uses whatever data the drawing gives. If the drawing shows a centre and a diameter, use centre-radius; if it shows three points on the rim, use the three-point method.

- **Points:** by keyboard coordinates, by cursor on a grid, at the end or middle of an existing line, or at an intersection.
- **Lines:** between two points, parallel or perpendicular to a line, or tangent to circles.
- **Circles and arcs:** by centre and radius, by three points, or tangent to other elements (fillets).
- **Curves:** splines are fitted through a list of points, e.g. a cubic spline.
- **Source:** Groover & Zimmers, Table 6.1, methods of defining elements.

Editing features change the model without redrawing it

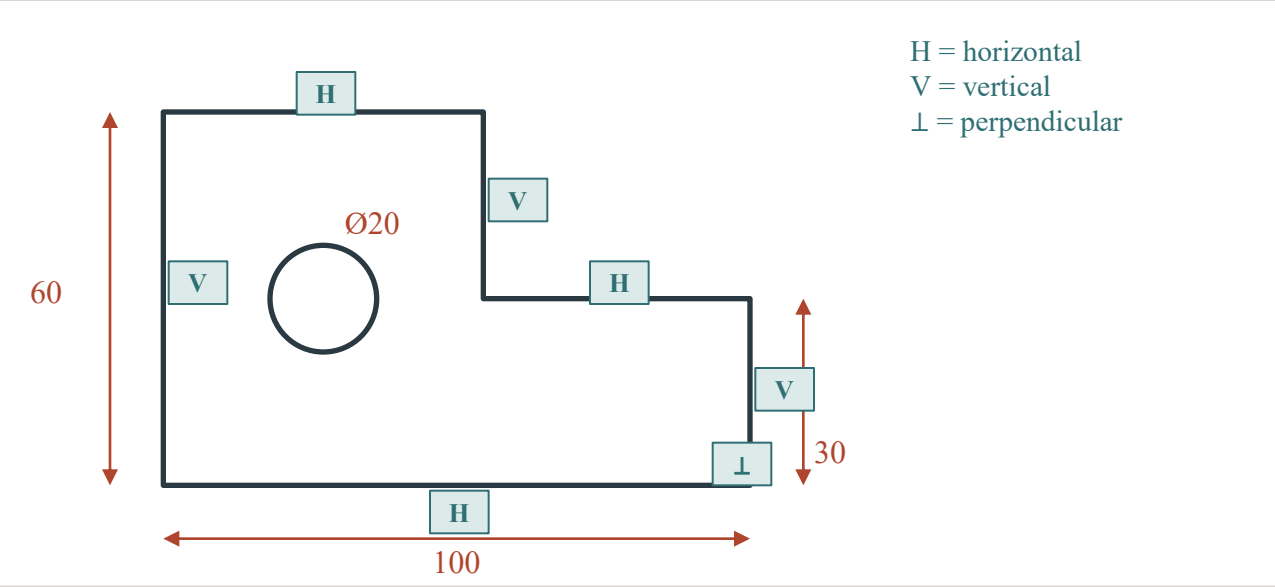


- **Move and copy:** translate selected elements to a new position; copy keeps the original.
- **Rotate:** turn elements about a chosen point by a given angle.
- **Mirror:** create a mirror image about a line or plane, useful for symmetric parts.
- **Scale:** enlarge or reduce elements by a factor.
- **Trim, extend, delete:** cut a line at an intersection, lengthen it, or remove it.
- **Basis:** most editing commands are geometric transformations (section 04).

Example
A flange has six equal holes. Draw one hole, then copy it with a rotate command: "copy, rotate 60°, 5 times". All six holes are exact and evenly spaced.

A constraint is a rule that fixes the shape or size of a sketch

A sketch with geometric and dimensional constraints

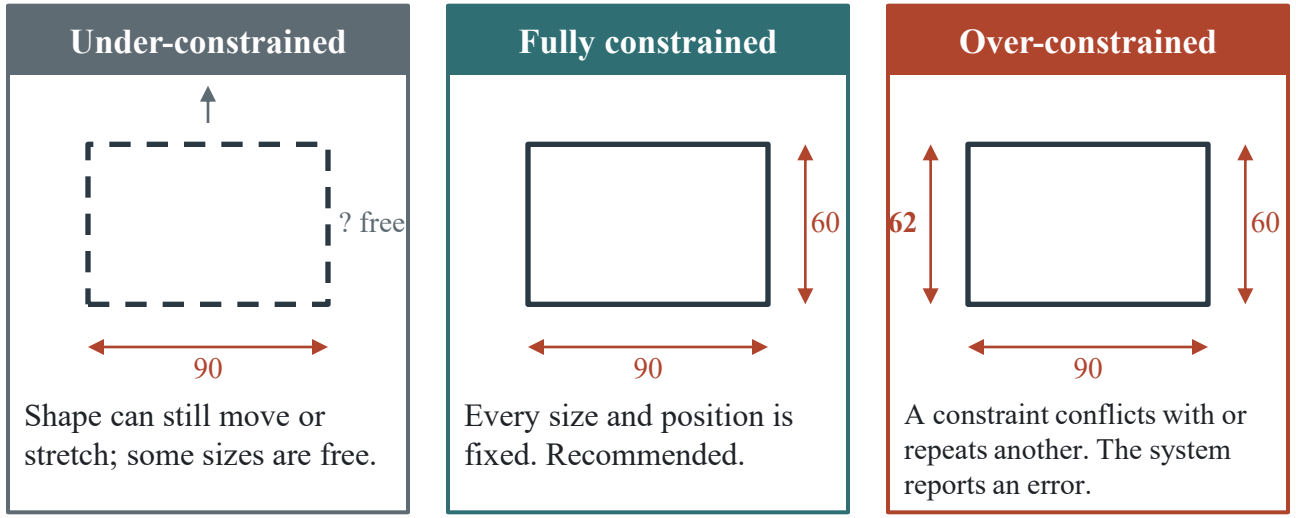


In short

A constraint is a rule applied to sketch entities that fixes their shape (geometric: parallel, perpendicular, tangent, horizontal, concentric) or size (dimensional: length, radius, angle).

- **Geometric constraints:** fix the relation between entities: horizontal, vertical, parallel, perpendicular, tangent, coincident, concentric, equal, symmetric.
- **Dimensional constraints:** fix sizes and positions: length 100, radius 10, angle 30°, distance between centres.
- **Why use them:** the sketch keeps the design intent. Change the 100 mm length to 120 mm and the profile updates while staying square.
- **Basis of:** parametric and feature-based modeling (section 07).

A sketch is under-, fully or over-constrained



How the designer uses this

1. Draw the rough profile.
2. Add geometric constraints (horizontal, vertical, tangent).
3. Add dimensions until the sketch turns fully constrained (most CAD tools change the colour, e.g. blue to black).
4. Delete any duplicate dimension that makes it over-constrained.

- **Degrees of freedom:** each free point in a 2D sketch can move in x and y. Every constraint removes one or more of these freedoms.
- **Under-constrained:** freedoms remain, so the sketch can change shape when dragged.
- **Fully constrained:** zero freedoms: the shape and size are completely defined.
- **Over-constrained:** more constraints than needed; here the height is given as both 60 and 62, so they conflict.

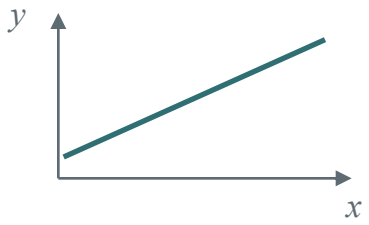
03

Curves

Analytic curves defined by equations, and synthetic curves shaped by control points

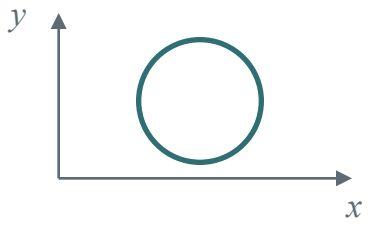
Analytic curves are defined by an exact mathematical equation

Straight line



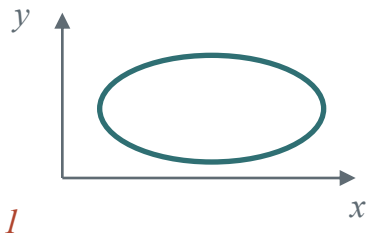
$y = mx + c$

Circle



$x^2 + y^2 = r^2$

Ellipse



$x^2/a^2 + y^2/b^2 = 1$

Parabola



$y^2 = 4ax$

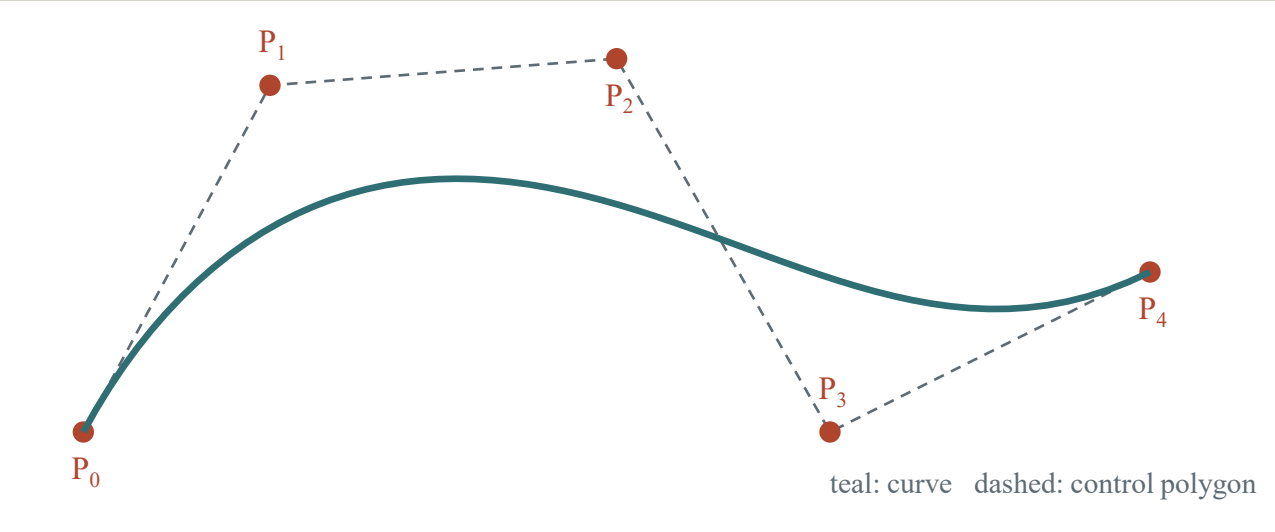
In short

Analytic curves are curves described by an exact analytic equation, e.g. line, circle, ellipse, parabola, hyperbola. Example: circle $x^2 + y^2 = r^2$. A few numbers (centre, radius) define the whole curve.

- **Definition:** a curve whose shape follows exactly from an equation; CAD stores only the equation's parameters.
- **Examples:** line, circle, arc, ellipse, parabola and hyperbola (the conic sections).
- **Storage:** a circle needs only its centre and radius, so the data is small and exact.
- **Limits:** cannot describe free-form shapes such as a car body, a turbine blade or a bottle.
- **Use:** machine parts made of straight edges, holes and arcs.

Synthetic curves are shaped by control points for free-form design

Free-form profile (e.g. a car fender) shaped by control points



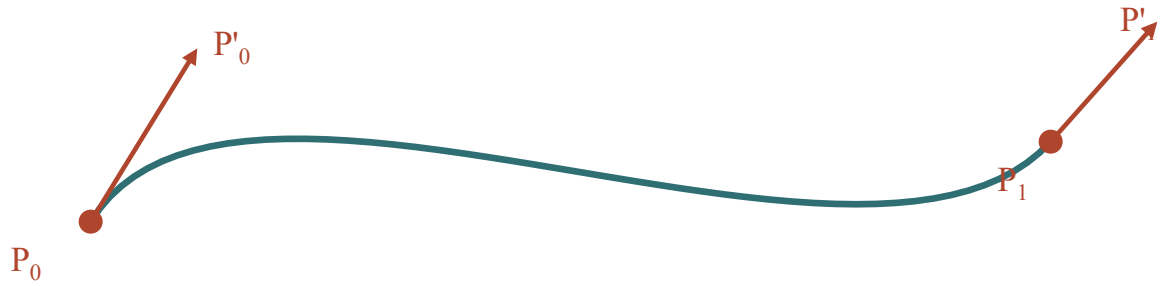
In short

Synthetic curves are curves defined by a set of data or control points rather than one equation. They give free-form shapes that analytic curves cannot. Examples: Hermite cubic spline, Bézier curve, B-spline curve.

- **Definition:** a curve built from polynomial pieces that pass through or near given points.
- **Why needed:** car bodies, aircraft wings, ship hulls and blades cannot be drawn with lines and circles alone.
- **Control points:** moving a point reshapes the curve, so the designer sculpts the shape interactively.
- **Continuity:** pieces join smoothly: same point (C0), same slope (C1), same curvature (C2).
- **Types:** Hermite cubic spline, Bézier, B-spline, NURBS.

A Hermite cubic spline is fixed by two end points and two end tangents

Hermite segment: end points and end tangents



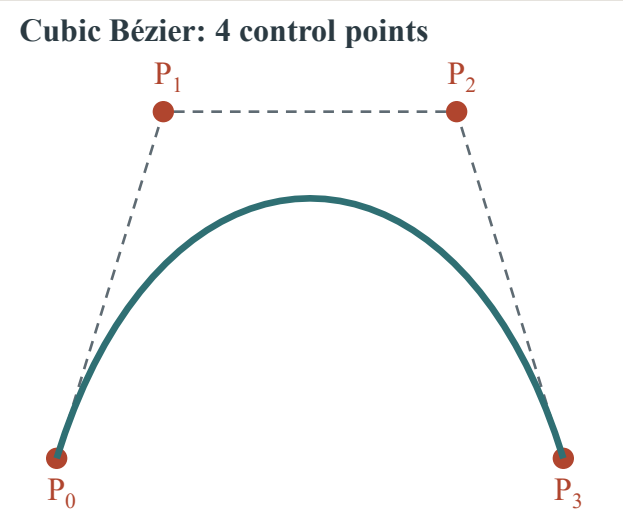
Equation ($0 \le u \le 1$)

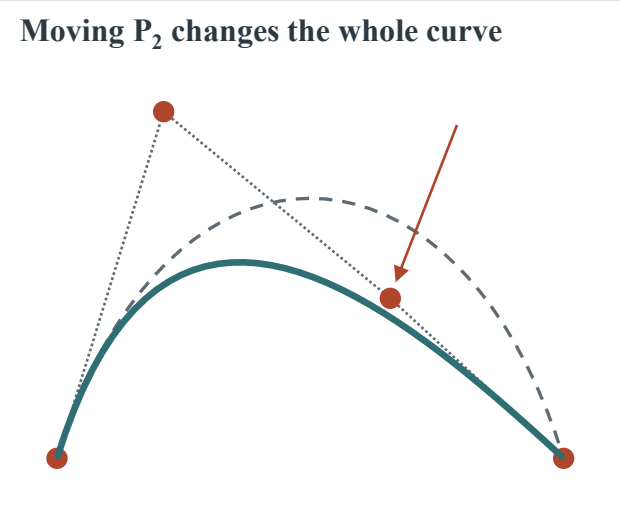
$$P(u) = (2u^3 - 3u^2 + 1)P_0 + (-2u^3 + 3u^2)P_1 + (u^3 - 2u^2 + u)P'_0 + (u^3 - u^2)P'_1$$

At $u = 0$ the curve is at P_0; at $u = 1$ it is at P_1. The tangent vectors P'_0 and P'_1 set the direction and fullness at each end.

- **Inputs:** four conditions: two end points P0, P1 and two end tangents P'0, P'1.
- **Cubic:** the curve is a third-degree polynomial in the parameter u, the lowest degree that can bend in an S-shape.
- **Interpolating:** the curve passes through both end points.
- **Spline:** several Hermite segments joined end to end, with equal tangents at each joint, form a smooth cubic spline.
- **Drawback:** tangent vectors are hard to judge by eye, so designers often prefer Bézier control points.

A Bézier curve is shaped by a polygon of control points





Equation

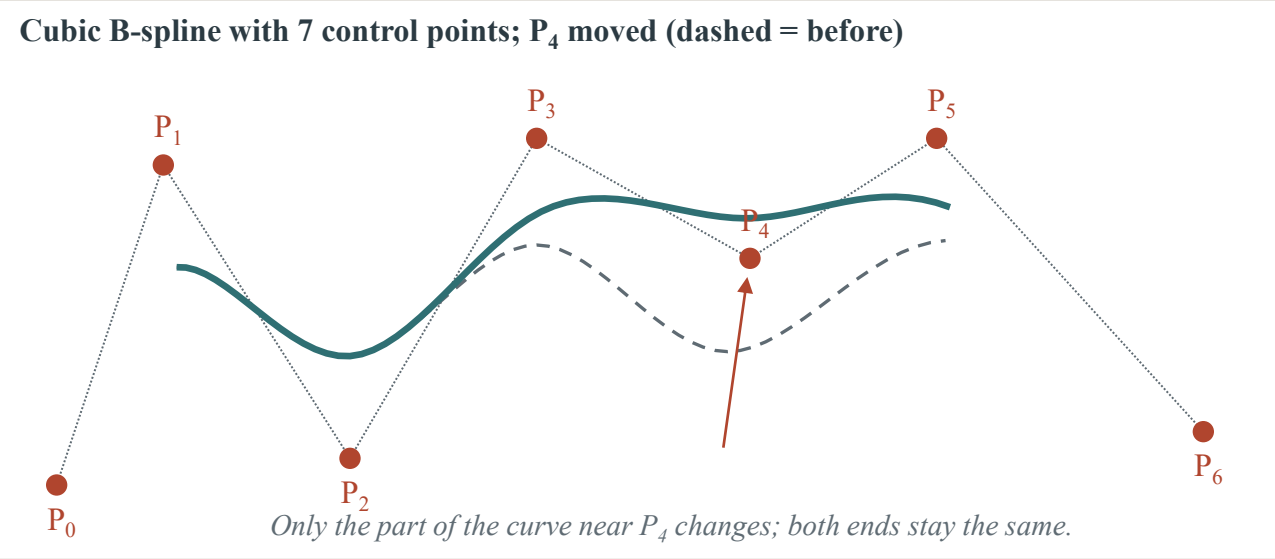
$P(u) = \Sigma\, P_i\, B_{i,n}(u),\ i = 0 \ldots n,\quad 0 \le u \le 1$

$B_{i,n}(u) = C(n, i)\, u^i\, (1 - u)^{n-i}$ (Bernstein polynomial)

For 4 points (n = 3): $P(u) = (1-u)^3 P_0 + 3u(1-u)^2 P_1 + 3u^2(1-u) P_2 + u^3 P_3$

- **Control polygon:** the curve follows the shape of the polygon joining the control points, so it is easy to design by eye.
- **End points:** the curve passes through the first and last points only, and is tangent to the first and last polygon sides there.
- **Degree:** n + 1 control points give a curve of degree n.
- **Convex hull:** the curve always lies inside the polygon's convex hull.
- **Global control:** moving any point changes the whole curve (right figure), a limitation for long curves.

A B-spline curve gives local control: moving one point changes only part of the curve



Bézier vs B-spline

- Bézier: degree fixed by number of points; global control.
- B-spline: degree chosen by the designer (usually cubic) whatever the number of points; local control.
- NURBS (non-uniform rational B-spline) adds weights and can represent circles exactly; it is the standard in modern CAD.

- **Definition:** a curve made of polynomial segments joined smoothly, shaped by a control polygon through basis functions.
- **Local control:** each control point affects only a few neighbouring segments (4 for a cubic).
- **Degree independent:** many control points can be used without raising the degree, so the curve stays well behaved.
- **Ends:** with open (clamped) knots the curve starts and ends at the first and last points.
- **Use:** car bodies, aircraft and turbine blade profiles.

Analytic and synthetic curves suit different design jobs

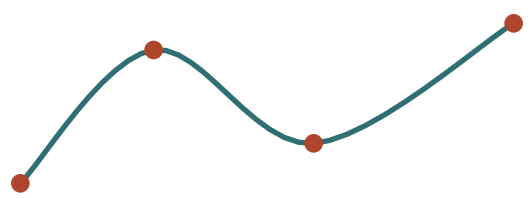
Basis	Analytic curves	Synthetic curves
Definition	Exact equation	Control or data points
Examples	Line, circle, ellipse, parabola	Hermite, Bézier, B-spline, NURBS
Shape range	Regular shapes only	Any free-form shape
Data stored	Few parameters (centre, radius)	Control points and degree
Editing	Change parameters	Drag control points
Typical use	Shafts, holes, plates	Car body, blade, bottle

Tip: sketch one analytic curve (a circle) and one synthetic curve (a Bézier with its control polygon) to see the difference.

- **Analytic example:** a hole of radius 10 mm at (50, 30): stored as centre (50, 30) and r = 10; the circle is exact.
- **Synthetic example:** a car fender profile: stored as 6 control points of a cubic B-spline; the designer drags points until it looks right.
- **Choice:** use analytic entities for machined features and synthetic curves for styled surfaces.
- **In practice:** a single part often uses both: a turbine blade has a B-spline aerofoil and a circular root fillet.

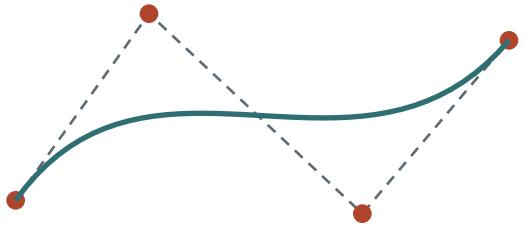
CAD generates curves in several ways

1. Interpolation



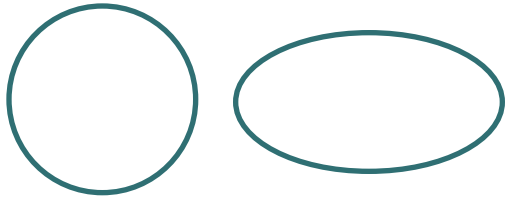
Curve passes through every given point (cubic spline). Used for measured data.

2. Approximation



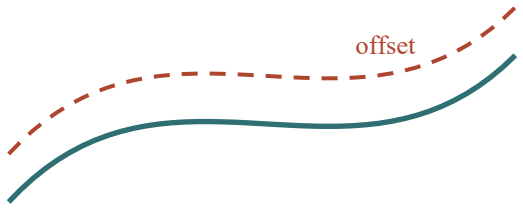
Curve follows a control polygon but passes only near the points (Bézier, B-spline).

3. Conic construction



Line, circle, arc, ellipse from equations and geometric data (centre, radius, tangency).

4. Intersection and offset



New curves from existing geometry: intersection of two surfaces, or a curve offset at a fixed distance.

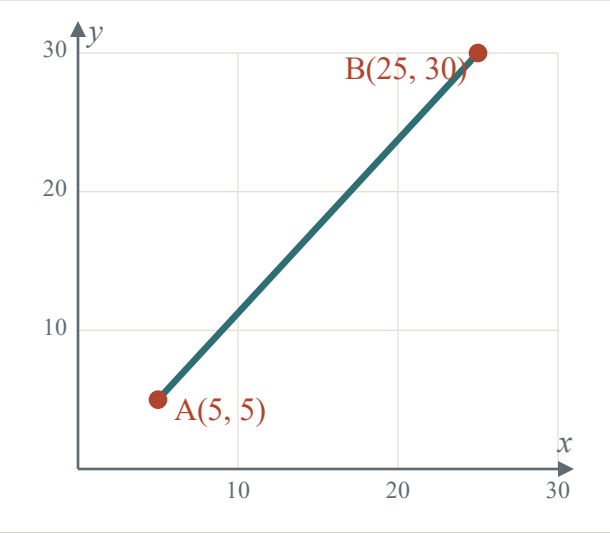
- **Interpolation:** the designer gives points the curve must pass through; the system fits a smooth cubic spline (e.g. digitized profile of a cam).
- **Approximation:** control points act like magnets; the curve is smoother and easier to reshape.
- **Analytic construction:** the classic method for machine parts: centre-radius circles, tangent arcs, fillets.
- **Derived curves:** intersection curves, offsets for tool paths, and projections onto surfaces.
- **In the exam:** name the technique, draw a small sketch, give one example.

04

Geometric transformations

Moving, resizing and turning geometry with matrices, with a full worked numerical

A transformation changes coordinates using matrix arithmetic



Point as a matrix

$$\begin{bmatrix} x & y \end{bmatrix}$$

Line AB as a 2 × 2 matrix

$$\begin{bmatrix} 5 & 5 \\ 25 & 30 \end{bmatrix}$$

row 1 = point A
row 2 = point B

Homogeneous coordinates

$$\begin{bmatrix} x & y & 1 \end{bmatrix}$$

adds a 1 so translation can also be a multiplication

General form

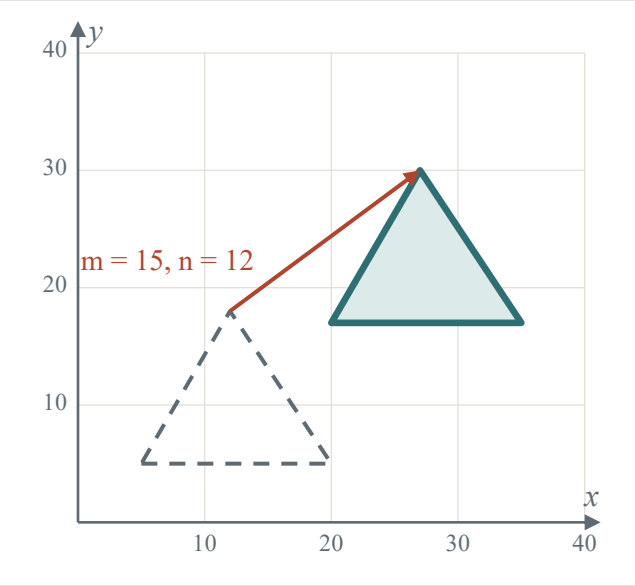
$[x'\ \ y'] = [x\ \ y] \cdot [T]$

- $[x\ \ y]$ = original point (or line matrix)
- $[T]$ = transformation matrix (translation, scaling, rotation…)
- $[x'\ \ y']$ = transformed point

Apply $[T]$ to every point; the whole entity is transformed.

- **Transformation:** an operation that changes the position, size or orientation of an entity by changing its coordinates.
- **Why matrices:** a computer stores every point as numbers, so one matrix multiplication transforms all points at once.
- **In CAD:** used for editing (move, copy, rotate, mirror, scale) and for viewing (zoom, pan, 3D rotation of the view).
- **Basic types:** translation, scaling, rotation; also reflection and their combination (concatenation).

Translation moves every point by the same distance



Equations

$x' = x + m$

$y' = y + n$

Matrix form

[x' y'] = [x y] + [m n]

Homogeneous (3 × 3)

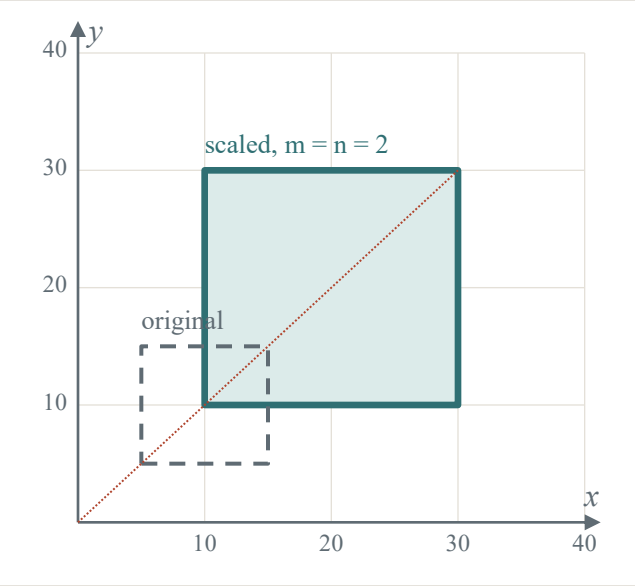
$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ m & n & 1 \end{bmatrix}$$

- **Definition:** moving an entity from one place to another without turning or resizing it.
- **Translation vector:** T = (m, n): m units along x, n units along y.
- **Every point moves equally:** so shape, size and orientation are unchanged.
- **CAD commands:** MOVE and COPY use translation.
- **Note:** in ordinary 2 × 2 form translation is an addition; homogeneous coordinates turn it into a multiplication so it can be combined with other transformations.

Worked example

Line (2, 2)–(5, 6) translated by m = 3, n = 4:
new end points (2+3, 2+4) = (5, 6) and (5+3, 6+4) = (8, 10). The line keeps its length and direction; only its position changes.

Scaling multiplies coordinates by a scale factor



Equations

$x' = m \cdot x$

$y' = n \cdot y$

Matrix form

$$[x' \;\; y'] = [x \;\; y] \begin{bmatrix} m & 0 \\ 0 & n \end{bmatrix}$$

Homogeneous (3 × 3)

$$\begin{bmatrix} m & 0 & 0 \\ 0 & n & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

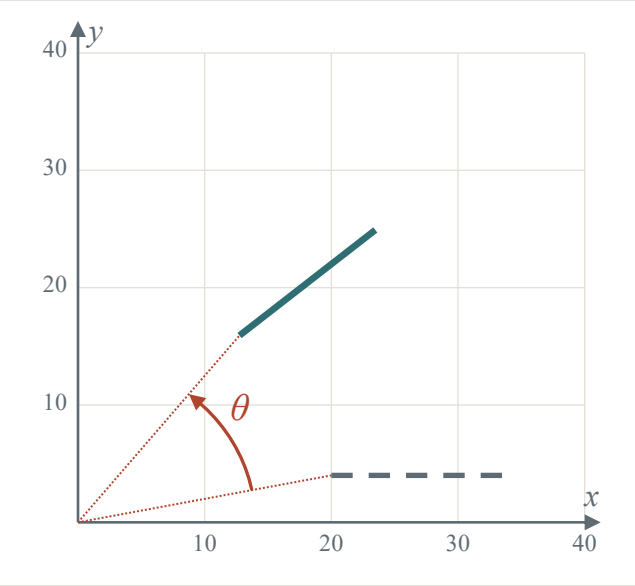
- **Definition:** enlarging or reducing an entity by multiplying its coordinates by scale factors.
- **Uniform scaling:** m = n: shape kept, only size changes. m ≠ n stretches the shape.
- **Factor > 1 or < 1:** greater than 1 enlarges; between 0 and 1 reduces.
- **About the origin:** every point moves along a line from the origin, so the entity also changes position. To scale about another point, translate it to the origin first (concatenation).
- **CAD command:** SCALE; zoom uses the same idea on the view.

Worked example

Line (5, 5)–(25, 30) scaled by 2: [5 5]·S = (10, 10) and [25 30]·S = (50, 60).

The line becomes twice as long and moves away from the origin, because scaling is about the origin.

Rotation turns points about the origin by an angle θ



Equations (counter-clockwise θ)

$$x' = x\cos\theta - y\sin\theta$$
$$y' = x\sin\theta + y\cos\theta$$

Matrix form

$$[x'\ \ y'] = [x\ \ y]\begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

Homogeneous (3 × 3)

$$\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- **Definition:** turning an entity about the origin through an angle θ.
- **Sign convention:** θ positive = counter-clockwise; for clockwise use −θ (the sin terms change sign).
- **Distances kept:** length and shape are unchanged; only orientation changes.
- **About another point:** translate that point to the origin, rotate, then translate back.
- **CAD command:** ROTATE, and rotating a 3D view.

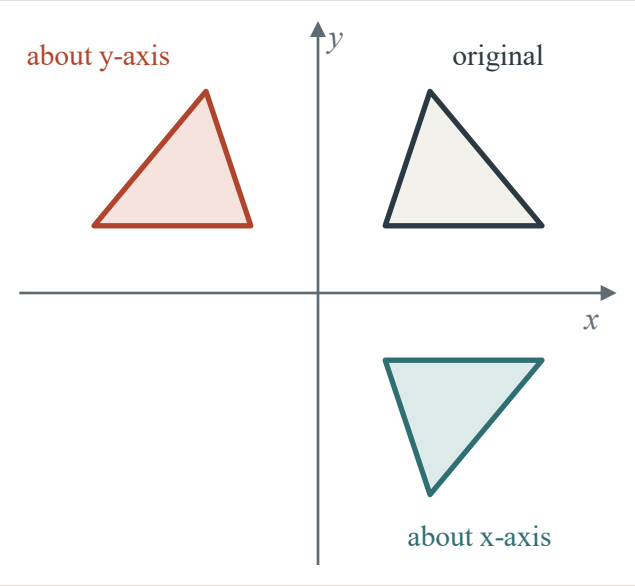
Worked example

Point (10, 0) rotated by 90°: cos 90° = 0, sin 90° = 1

$x' = 10\cdot0 - 0\cdot1 = 0,\ \ y' = 10\cdot1 + 0\cdot0 = 10 \rightarrow (0, 10).$

The point moves from the x-axis to the y-axis, still 10 units from the origin.

Reflection creates a mirror image about an axis



About the x-axis (y → −y)

$$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad x' = x,\ y' = -y$$

About the y-axis (x → −x)

$$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \quad x' = -x,\ y' = y$$

About the line y = x

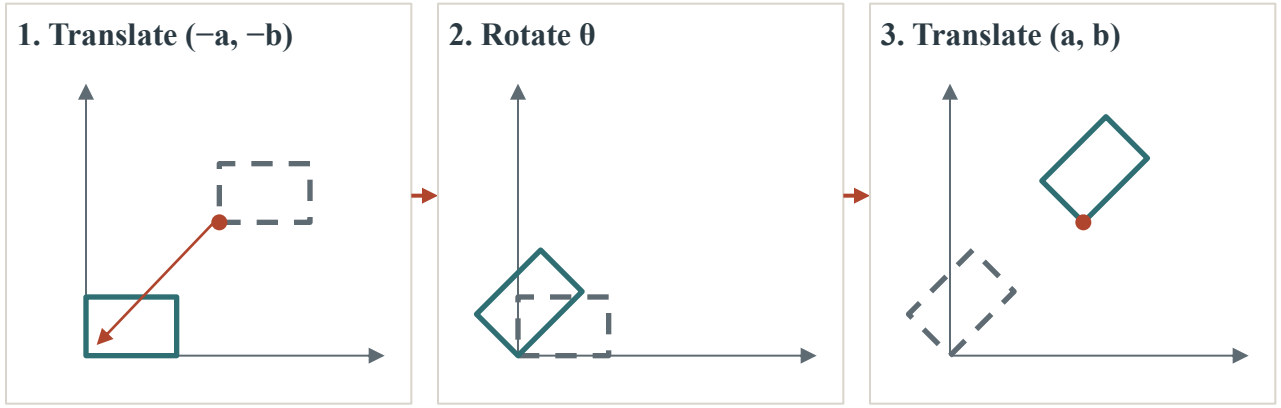
$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad x' = y,\ y' = x$$

Worked example

Point (6, 18) reflected about the x-axis: [6 18]·[[1 0][0 −1]] = (6, −18). About the y-axis it becomes (−6, 18). A symmetric bracket is drawn as one half and mirrored.

- **Definition:** produces the mirror image of an entity about an axis or line.
- **Sign change:** reflection about an axis reverses the sign of the other coordinate.
- **Shape kept:** size and shape stay the same, but the orientation is reversed (left hand becomes right hand).
- **CAD command:** MIRROR: draw half of a symmetric part, then mirror it, saving time and guaranteeing symmetry.

Concatenation combines several transformations into one matrix



Rotating a part by 45° about its own corner P(a, b)

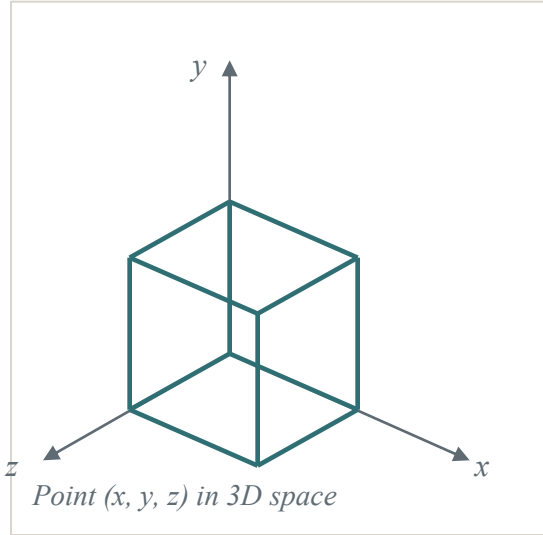
Combined matrix

$[T] = [T_1] \cdot [R] \cdot [T_2]$ (applied left to right, row-vector form)

Example: scale a point by 2 and then rotate it by 90°.
S = [[2 0][0 2]], R = [[0 1][−1 0]] → [T] = S·R = [[0 2][−2 0]]
Point (1, 1): [1 1]·[T] = (−2, 2). Doing the two steps one after another gives the same answer.

- **Concatenation:** multiplying the individual transformation matrices together to obtain one matrix that does all the steps.
- **Why:** the computer multiplies each point by one matrix instead of several, which saves time for models with thousands of points.
- **Order matters:** matrix multiplication is not commutative: rotate-then-translate is not the same as translate-then-rotate.
- **Homogeneous form:** 3 × 3 matrices let translation be multiplied with scaling and rotation.
- **Typical use:** rotation or scaling about any point other than the origin.

Three-dimensional transformations use 4 × 4 matrices



Point (x, y, z) in 3D space

Translation (4 × 4)

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ l & m & n & 1 \end{bmatrix}$$

Scaling

$$\begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotation about the z-axis

$$\begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotation about the other axes

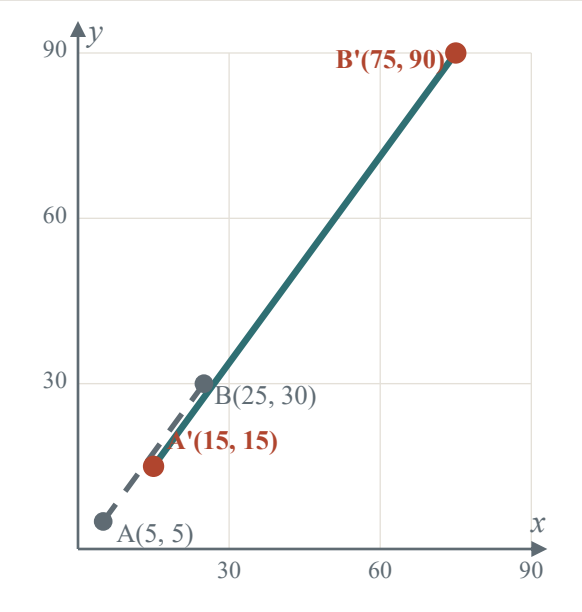
About x: $y' = y\cos\theta - z\sin\theta$, $z' = y\sin\theta + z\cos\theta$ (x unchanged)

About y: $x' = x\cos\theta + z\sin\theta$, $z' = -x\sin\theta + z\cos\theta$ (y unchanged)

Point form: $[x'\ y'\ z'\ 1] = [x\ y\ z\ 1] \cdot [T]$

- **3D points:** each point has three coordinates (x, y, z); homogeneous form adds a fourth term, 1.
- **4 × 4 matrices:** the same idea as 2D: translation by (l, m, n), scaling by (a, b, c), rotation about one axis at a time.
- **Rotation:** the coordinate along the rotation axis does not change.
- **Use in CAD:** rotating a solid model to view it from any direction, and positioning parts in an assembly.

Numerical, part I: scaling the line by a factor of 3



dashed: original solid: transformed

Given line in matrix form

$$[L] = \begin{bmatrix} 5 & 5 \\ 25 & 30 \end{bmatrix}$$ row 1: A, row 2: B

Scaling matrix, factor 3

$$[S] = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix}$$

Multiply [L'] = [L]·[S]

$$\begin{bmatrix} 5 & 5 \\ 25 & 30 \end{bmatrix} \cdot \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix} = \begin{bmatrix} 15 & 15 \\ 75 & 90 \end{bmatrix}$$

Working

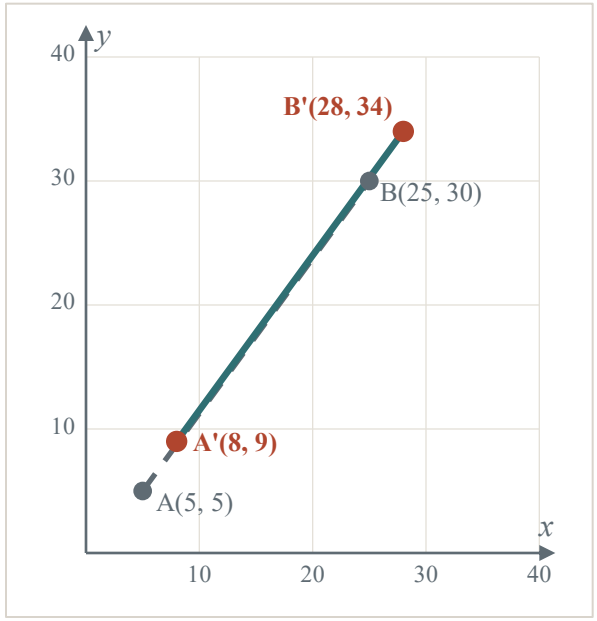
A': $x' = 5\times3 + 5\times0 = 15$, $y' = 5\times0 + 5\times3 = 15$

B': $x' = 25\times3 + 30\times0 = 75$, $y' = 25\times0 + 30\times3 = 90$

Answer: the scaled line runs from (15, 15) to (75, 90).

- **Problem: a line** has end points (5, 5) and (25, 30). (I) scale by 3, (II) translate by 3 in x and 4 in y, (III) rotate 30° counter-clockwise about the origin.
- **Step 1:** write the line as a 2 × 2 matrix, one row per end point.
- **Step 2:** write the scaling matrix with m = n = 3.
- **Step 3:** multiply row by column for each end point.
- **Check:** the length grows from 32.0 to 96.0 units, exactly 3 times.

Numerical, part II: translating the line by (3, 4)



dashed: original solid: transformed

Line in homogeneous form

$$[L] = \begin{bmatrix} 5 & 5 & 1 \\ 25 & 30 & 1 \end{bmatrix}$$

Translation matrix, m = 3, n = 4

$$[T] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 3 & 4 & 1 \end{bmatrix}$$

Multiply [L'] = [L]·[T]

$$\begin{bmatrix} 5 & 5 & 1 \\ 25 & 30 & 1 \end{bmatrix} \rightarrow \begin{bmatrix} 8 & 9 & 1 \\ 28 & 34 & 1 \end{bmatrix}$$

Working

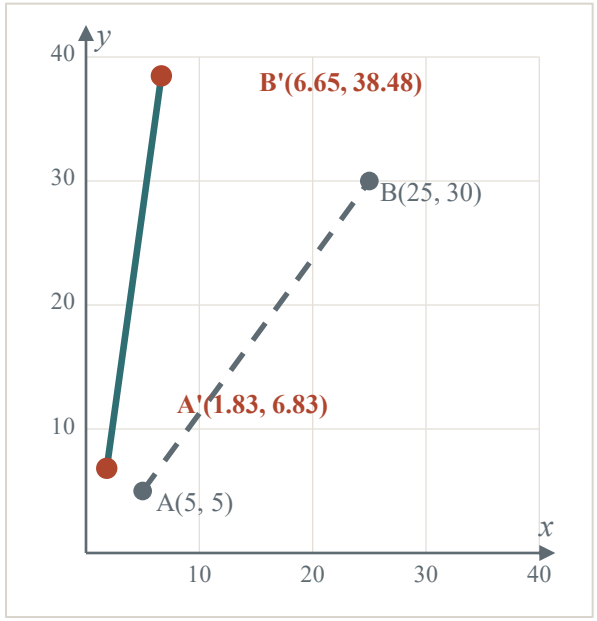
A': $x' = 5 + 3 = 8$, $y' = 5 + 4 = 9$

B': $x' = 25 + 3 = 28$, $y' = 30 + 4 = 34$

Answer: the translated line runs from (8, 9) to (28, 34).

- **Original line:** the problem asks to translate the original line, not the scaled one, so start again from (5, 5)–(25, 30).
- **Short method:** $x' = x + m$, $y' = y + n$ gives the answer directly.
- **Matrix method:** use homogeneous coordinates [x y 1] and the 3 × 3 matrix; examiners expect the matrix.
- **Check:** the length stays 32.0 units and the slope stays 1.25; only the position changes.

Numerical, part III: rotating the line by 30° counter-clockwise



dashed: original solid: transformed

Rotation matrix, θ = 30°

cos 30° = 0.866, sin 30° = 0.5

$$[R] = \begin{bmatrix} 0.866 & 0.5 \\ -0.5 & 0.866 \end{bmatrix}$$

Multiply [L'] = [L]·[R]

$$\begin{bmatrix} 5 & 5 \\ 25 & 30 \end{bmatrix} \cdot \begin{bmatrix} 0.866 & 0.5 \\ -0.5 & 0.866 \end{bmatrix} =$$

$$\begin{bmatrix} 1.83 & 6.83 \\ 6.65 & 38.48 \end{bmatrix}$$ rounded to 2 decimals

Working ($x' = x\cos\theta - y\sin\theta,\ y' = x\sin\theta + y\cos\theta$)

A': $x' = 5(0.866) - 5(0.5) = 1.83,\ y' = 5(0.5) + 5(0.866) = 6.83$

B': $x' = 25(0.866) - 30(0.5) = 6.65,\ y' = 25(0.5) + 30(0.866) = 38.48$

Answer: the rotated line runs from (1.83, 6.83) to (6.65, 38.48).

- **Counter-clockwise:** θ is positive, so use +30° in the standard matrix.
- **Row-vector form:** [x' y'] = [x y]·[R] with R = [[cos θ, sin θ], [−sin θ, cos θ]].
- **Check 1:** length: $\sqrt{((6.65-1.83)^2 + (38.48-6.83)^2)} = 32.0$, unchanged.
- **Check 2:** each end point stays the same distance from the origin, e.g. OA = OA' = 7.07.
- **Write-up:** state the matrix, show the multiplication, give the answer, and sketch the original and rotated lines.

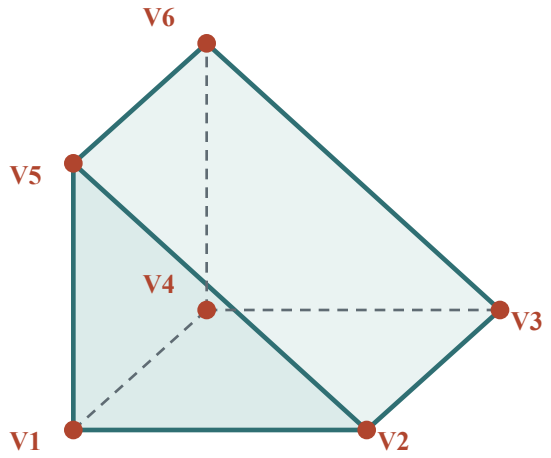
05

Database structure and content

What the CAD database stores, and how solids are combined with Boolean operations

The CAD database stores geometry, topology and non-graphic attributes

Wedge: vertices, edges, faces



Geometry (where)	Topology
V1 (0, 0, 0)	E1 = V1–V2
V2 (60, 0, 0)	E7 = V5–V6
V5 (0, 0, 40) …	Face F1 = E1, E2, E3, E4

Attributes (non-graphic)
Material: mild steel Layer: 2 Colour: blue
Part no.: BR-102 Tolerance: ±0.1 mm

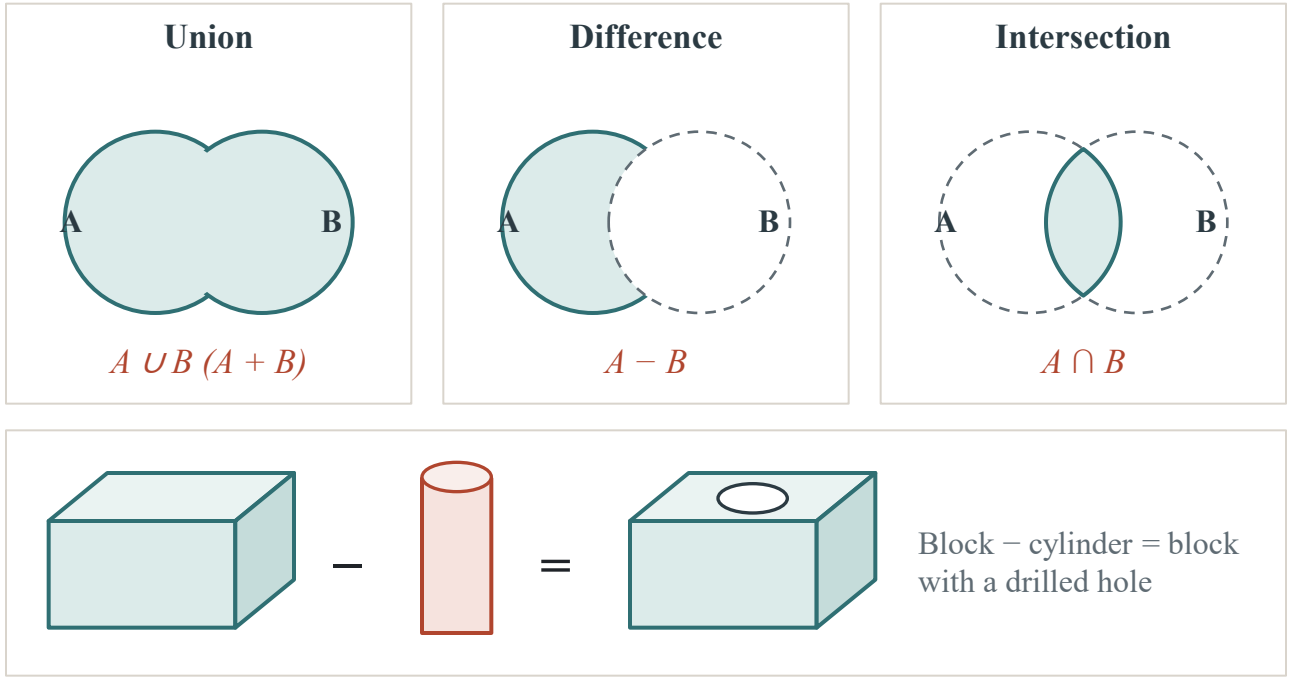
Change V5's coordinates and every edge and face using V5 updates automatically.

How the database is organised

Entities are stored in linked lists: each face points to its edges, each edge to its two vertices. Groups (segments, layers) point to their member entities. Display files for the screen are generated from this data, not stored as pictures.

- **Geometry:** the numbers that fix size and position: coordinates of points, centres and radii, curve coefficients.
- **Topology:** how entities are connected: which edges bound a face, which vertices end an edge. It turns a set of lines into a closed solid.
- **Attributes:** non-graphic data such as material, part number, layer, colour, line type and tolerances.
- **Relations:** links between parts in an assembly and between a drawing and its model.
- **Users:** drafting, analysis, process planning and NC programming all read the same database.

Boolean operations combine primitives into a solid: union, difference and intersection



Groover Fig. 6.7: C(A + B)
Two primitives A and B are first joined (union) and then combined with C to give the final solid. The order of operations is stored as a tree in the database.

- **Primitives:** simple solids such as block, cylinder, sphere, cone, wedge and torus.
- **Union (A ∪ B):** joins two solids into one; overlapping volume counted once.
- **Difference (A − B):** removes B from A: holes, slots and pockets. A − B is not the same as B − A.
- **Intersection (A ∩ B):** keeps only the volume common to both.
- **Result:** any complex part can be built step by step; this is the basis of CSG modeling (section 06).

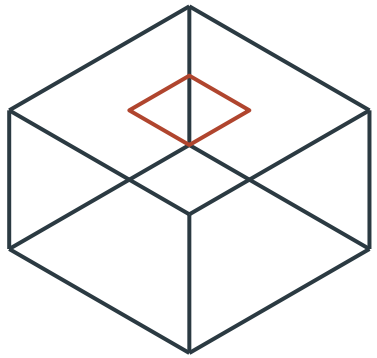
06

Wireframe, surface and solid models

Three ways to describe a part in the computer, and the solid modeling methods

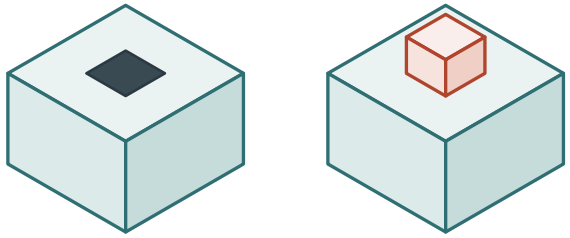
A wireframe model stores only edges, so it can be ambiguous

Wireframe as stored



Is the inner square a hole, a boss or just a line?

Possible interpretations



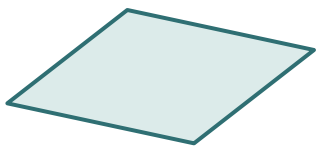
The same edges fit both solids, so the computer cannot tell which part is meant.

Advantages and limits

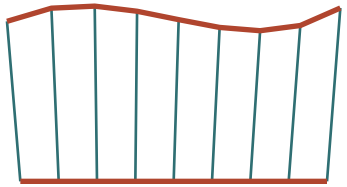
- Simple, fast, needs little memory; good for drafting and early layouts.
- Ambiguous; no surfaces, so no hidden-line removal, shading, or mass properties; cannot generate tool paths over surfaces.

- **Definition:** a 3D model built only from points and edges (lines, arcs, curves) joining them, like a frame made of wire.
- **Stored data:** vertex coordinates and the edges between them; no faces and no volume.
- **Ambiguity:** more than one solid can match the same wireframe (left figure).
- **Uses:** quick concept layouts, 2D drafting, and the control curves of an aircraft (Groover Fig. 6.11).
- **Remedy:** surface or solid modeling adds the missing face and volume information.

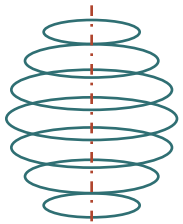
Surface models describe the skin of a part with several types of surface



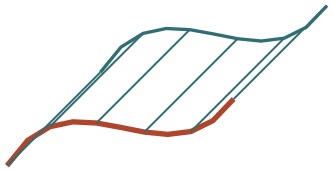
Plane surface



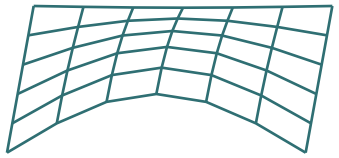
Ruled surface



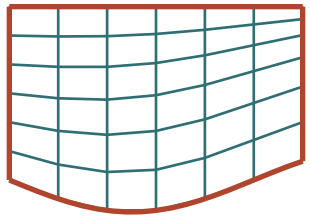
Surface of revolution



Tabulated cylinder



Bézier / B-spline surface



Coons patch

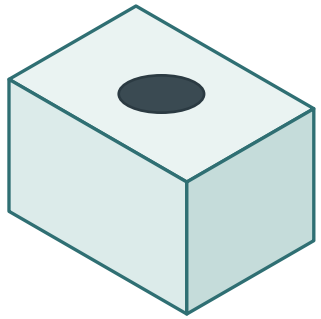
Use

Surface models give the exact skin needed for car bodies, aircraft panels, dies and moulds, and for NC tool paths over 3D surfaces. They still have no volume, so mass properties need a solid model.

- **Plane:** flat surface through three points.
- **Ruled:** straight lines joining matching points of two curves (red).
- **Revolution:** a curve rotated about an axis: bottles, nozzles.
- **Tabulated cylinder:** a curve swept along a straight direction.
- **Bézier and B-spline:** free-form surfaces shaped by a grid of control points.
- **Coons patch:** a surface blended between four boundary curves (red).
- **Fillet and offset:** rounds between surfaces and surfaces at a fixed distance.

Solid modeling gives a complete and unambiguous description of the part

A solid model knows its inside and its outside



Bracket block with a hole

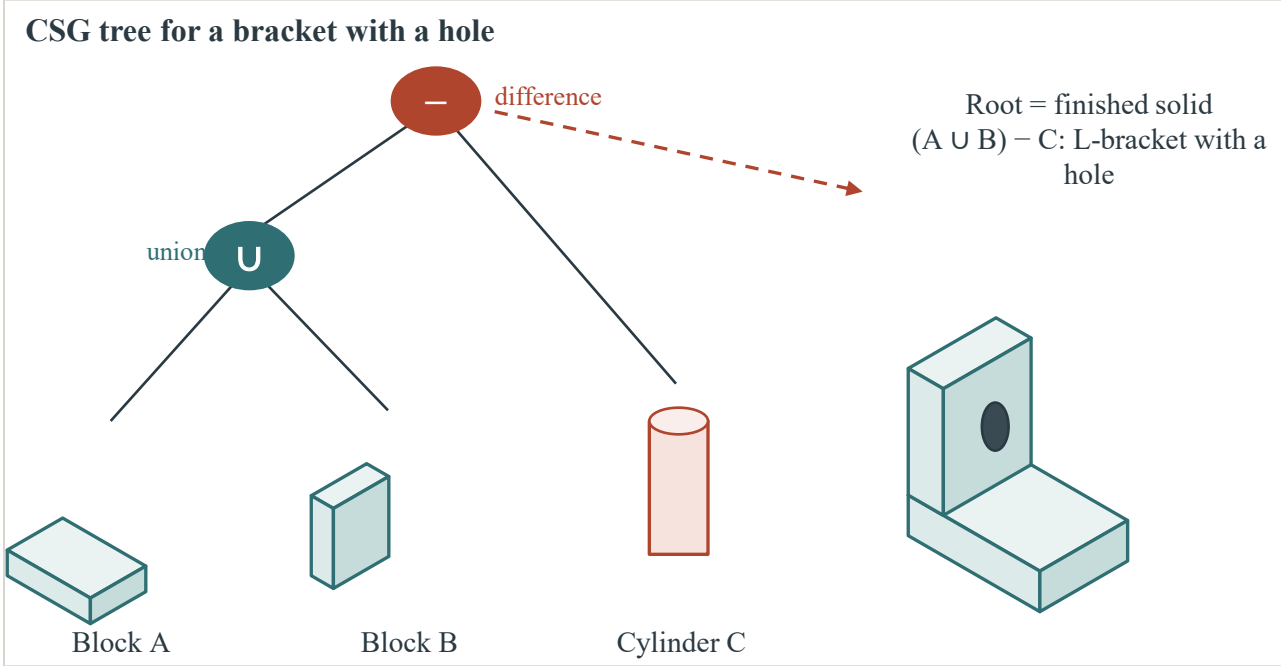
Computed automatically	
Volume	**143.4 cm³**
Mass (steel)	**1.13 kg**
Centre of gravity	**(35, 22.5, 25) mm**
Surface area	**200 cm²**

Significance in engineering design

- Complete, unambiguous model: every point is known to be inside, outside or on the surface.
- Mass properties, FEA meshing, interference checks and NC tool paths come straight from it.
- Realistic shaded images and sections for design review.
- One database for drawings, analysis, CAM and inspection, so fewer errors and shorter lead time.

- **Definition:** a model that represents the complete volume of a part, with faces, edges and the material inside.
- **Valid solid:** closed, with no gaps or dangling edges; the system can check this automatically.
- **What it enables:** volume, weight, centre of gravity, moments of inertia, sections, hidden-line removal and shading.
- **Methods:** constructive solid geometry (CSG), boundary representation (B-rep), sweeping, cell decomposition, hybrid.
- **Example values:** computed for a 70 × 45 × 50 mm steel block with a 20 mm hole; illustrative.

Constructive solid geometry builds a solid from primitives in a tree



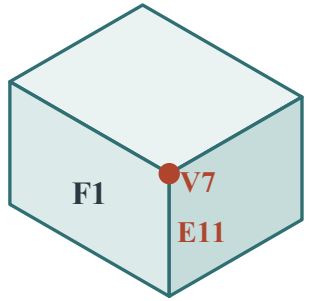
How CSG is stored

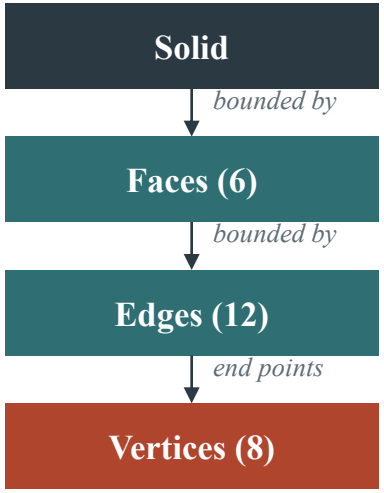
The database stores the tree, not the faces: the leaf primitives with their sizes and positions, and the Boolean operator at each node. The faces are computed when the model is displayed.

- **Idea:** build the part like a child builds with blocks: start with primitives and combine them with union, difference and intersection.
- **Primitives:** block, cylinder, cone, sphere, wedge, torus; each defined by a few dimensions and a position.
- **CSG tree:** leaves are primitives, nodes are Boolean operations, the root is the finished solid.
- **Advantages:** compact data, always a valid solid, easy to edit by changing a primitive size.
- **Limits:** free-form shapes are hard to build; faces and edges must be computed each time they are needed.

Boundary representation stores the faces, edges and vertices of the solid

B-rep of a block: faces → edges → vertices





Euler check: V − E + F = 8 − 12 + 6 = 2

How the designer builds a B-rep

Draw a 2D profile or the input views (Groover Fig. 6.12), then sweep or extrude it. The system computes and stores every face, edge and vertex with their links, so the boundary is explicit and ready for display.

- **Idea:** a solid is described by its boundary: the faces that enclose it, the edges between faces, and the vertices at the ends of edges.
- **Data structure:** geometry (surface equations, coordinates) plus topology (which face uses which edges).
- **Validity:** the Euler formula V − E + F = 2 checks that a simple solid is closed.
- **Advantages:** faces are ready for display, drawing and NC; free-form surfaces fit easily.
- **Limits:** large data; local edits can leave an invalid solid if topology is broken.

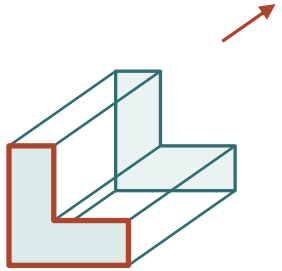
CSG and B-rep differ in data structure, editing and computation

Basis	CSG (C-rep)	B-rep
Data structure	Binary tree of primitives and Boolean operations	Lists of faces, edges and vertices with their links
Data size	Small, compact	Large: every face and edge stored
Ease of editing	Easy: change a primitive or a node	Harder: faces and topology must be updated
Validity	Always a valid solid	Must be checked (Euler rule)
Computation	High at display time: boundary evaluated each time	Low at display time: boundary already stored
Free-form shapes	Difficult	Easy (Bézier, B-spline faces)
Conversion	CSG → B-rep is easy	B-rep → CSG is difficult

Hybrid modelers keep both: the CSG tree for editing and the B-rep for display and manufacturing (Groover & Zimmers, Sec. 6.7).

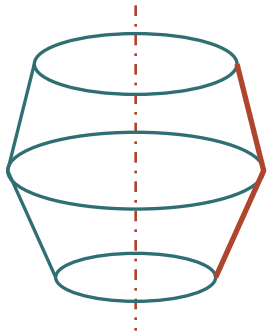
Sweeping, cell decomposition and hybrid modelers complete the solid modeling methods

Translational sweep



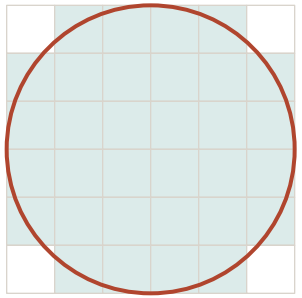
A 2D profile is moved along a straight line to sweep out a solid (extrude).

Rotational sweep



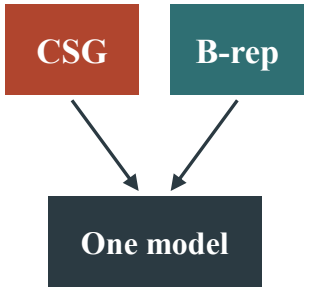
A profile is rotated about an axis to give a solid of revolution (revolve).

Cell decomposition



The solid is divided into small cells (cubes); each cell is marked full or empty. Used in FEA meshing.

Hybrid modeler



Stores the CSG tree and the B-rep together: easy editing plus fast display and machining.

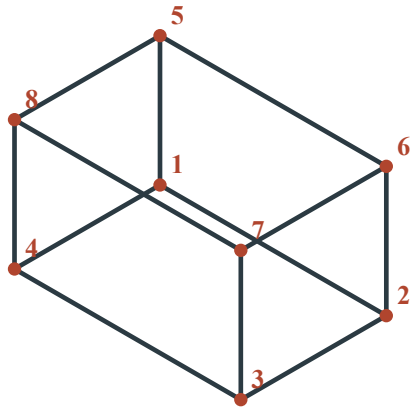
- **Sweep:** simplest way to make prismatic and turned parts.
- **Cells:** accurate only as the cells get smaller.
- **Hybrid:** used by most modern CAD systems.

Summary: solid modeling methods

1. Constructive solid geometry 2. Boundary representation 3. Sweeping (translational, rotational) 4. Cell decomposition 5. Hybrid. Each has its own idea, typical shape and main advantage.

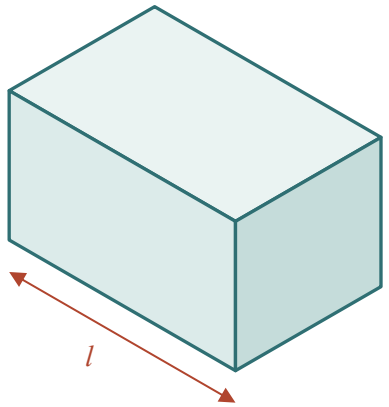
A rectangular block as a wireframe model and as a solid model

(a) Wireframe model



8 vertices + 12 edges; all edges shown, faces not defined

(b) Solid model



6 faces enclose a volume; hidden edges removed, shaded

Data stored for a 70 × 40 × 45 block

Wireframe: 8 vertex coordinates + 12 edge pairs. Solid (CSG): one primitive BLOCK(l = 70, h = 40, w = 45) at the origin. Solid (B-rep): 6 faces, 12 edges, 8 vertices with links.

- **Wireframe:** draw the 12 edges joining the 8 corner points; label the vertices. Every edge is visible, so the drawing can look confusing.
- **Solid:** draw the same block with visible faces shaded and hidden edges removed; the model now has volume.
- **What the solid adds:** volume = 70 × 40 × 45 = 126 cm³, mass, sections and interference checks.
- **Tip: compare both figures side by side, label** vertices on the wireframe, and list three differences below.

Wireframe, surface and solid models suit different engineering uses

Basis	 Wireframe	 Surface	 Solid
Stores	Points and edges	Edges + surfaces (skin)	Edges, faces and volume
Ambiguity	Ambiguous	Mostly clear	Unambiguous
Hidden lines, shading	Not possible	Possible	Automatic
Mass properties	No	Area only	Volume, mass, CG, inertia
Interference check	No	Limited	Yes
Data and computing	Least	Medium	Most
Typical use	Layout, drafting, aircraft control curves	Car body, aircraft skin, dies, NC surface machining	Machine parts, assemblies, FEA, kinematic simulation

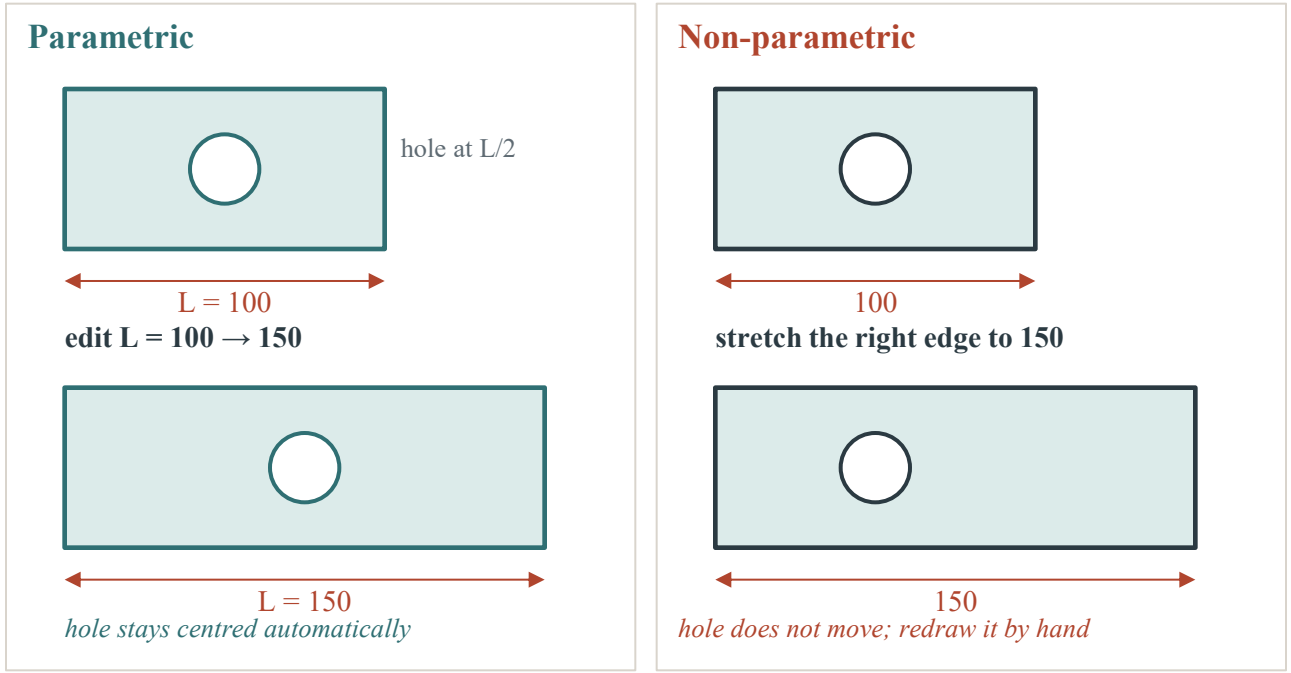
The table compares the three models; the previous slide shows the same rectangular block as a wireframe and as a solid.

07

Modern modeling methods

Parametric and feature-based modeling, extrude and revolve, and assembly simulation

Parametric design drives the model from dimensions; non-parametric design does not

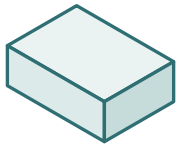


- **Parametric design:** geometry is controlled by named dimensions, constraints and equations; editing a value regenerates the model.
- **Design intent:** relations such as "hole always centred" are stored with the model.
- **History:** features are recorded in order in a model tree and can be edited later.
- **Non-parametric (explicit):** geometry is stored as fixed coordinates; to change a size you move or redraw entities directly.
- **When each is used:** parametric for product families and frequent changes; non-parametric (direct modeling) for quick edits of imported models.

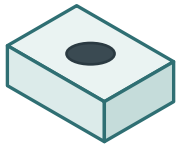
Parametric relations

Dimensions are variables: hole_x = L / 2, thickness = 0.1 × L. Change one value and every related feature updates. A design table can generate a whole family of sizes (M6 … M24 bolts) from one model.

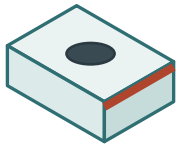
Feature-based modeling builds the part from engineering features such as holes, fillets and shells



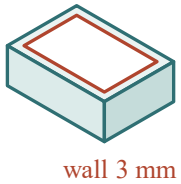
1. Base block



2. Hole



3. Fillet



4. Shell

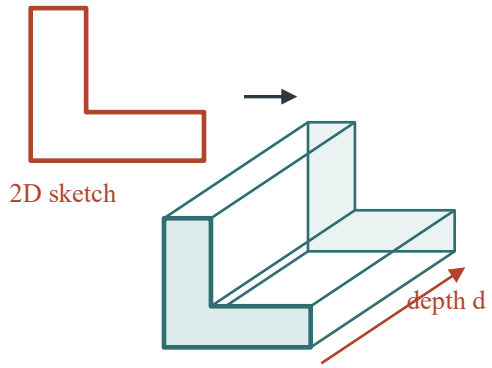
Model tree: Base-Extrude → Hole1 → Fillet1 → Shell1

Hole	Simple, counterbored, countersunk, threaded; placed by centre and depth.
Fillet / chamfer	Rounds or bevels an edge, removing sharp corners.
Shell	Hollows a solid to a uniform wall thickness, leaving chosen faces open (casings, bottles).
Others	Rib, slot, boss, pattern, draft.

- **Definition:** modeling with features: shapes that carry engineering meaning (hole, slot, fillet, rib), not just lines and surfaces.
- **How:** start with a base feature (extrude or revolve), then add or cut features. Each is stored with its parameters.
- **Benefits:** faster modeling, design intent kept, easy changes, and features link to manufacturing (a hole maps to a drilling operation).
- **Example:** a pump cover: extrude base, add bolt-hole pattern, fillet edges, shell to 3 mm.

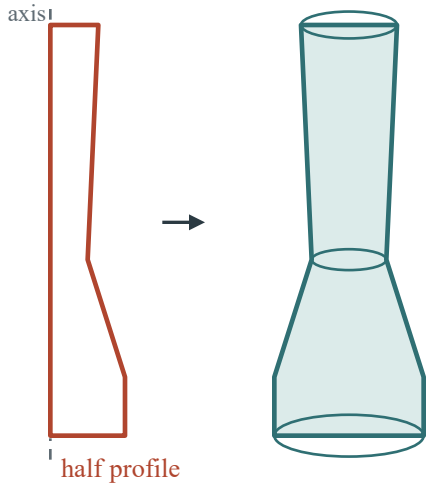
Extrude and revolve turn a 2D sketch into a 3D solid

Extrude



profile moved straight along its normal

Revolve



profile rotated 360° about the axis

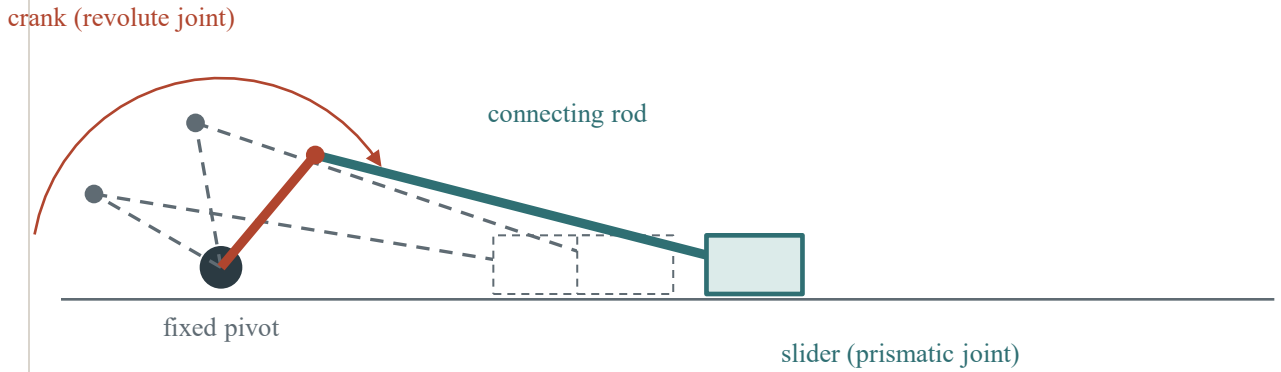
Rules

Extrude: closed profile + depth (one or both sides, or up to a surface); also cut-extrude to remove material. Revolve: closed profile on one side of the axis + angle (360° or less); the profile must not cross the axis.

- **Extrude:** sweeps a closed 2D sketch in a straight line to a set depth. Gives prismatic parts: plates, brackets, rails, gear blanks.
- **Revolve:** rotates a closed 2D sketch about an axis. Gives turned parts: shafts, pulleys, flanges, bottles, nozzles.
- **Add or cut:** both can add material (boss) or remove it (cut, groove).
- **Related features:** sweep (along a path) and loft (between sections).
- **Tip: always show the sketch**, the axis or direction, and the resulting solid for each.

Kinematic simulation moves the parts of an assembly through their motion

Slider-crank assembly: crank at 50° (solid) and two other positions



Outputs: slider displacement, velocity and acceleration vs crank angle; joint forces.

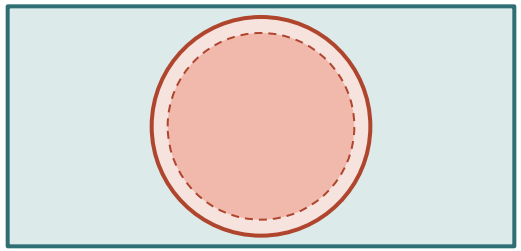
Process of kinematic simulation

1. Build the part models and assemble them. 2. Define joints / mates (revolute, prismatic, cylindrical, spherical, gear, cam). 3. Fix the ground part and add a driver (motor speed or angle). 4. Run the motion study over time steps. 5. Review paths, plots and animation; change the design and re-run.

- **Kinematic simulation:** moving the parts of an assembly on the computer according to their joints, to see how the mechanism behaves without building it.
- **Joints:** each joint removes degrees of freedom; a revolute joint allows only rotation, a prismatic joint only sliding.
- **Driver:** a motor or input motion makes the mechanism move; the software solves positions at each time step.
- **Results:** trajectories, velocities, accelerations, range of motion, and positions where parts clash.
- **Examples:** engine slider-crank, robot arm, door hinge, landing gear.

CAD checks interference and collision before any prototype is built

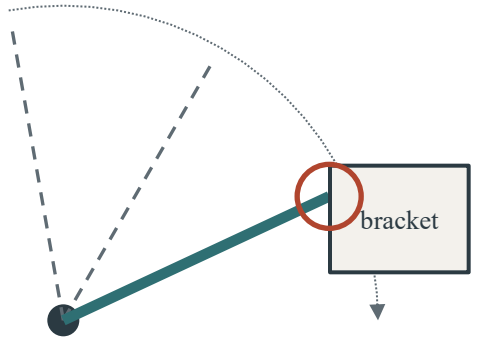
Static interference check



Shaft Ø32 in hole Ø30: overlap volume shown in red

Interference = 1.0 mm on radius → fix the fit

Dynamic collision check



Arm sweeps 100° → 0°; collides with the bracket at about 25°

- **Interference:** two parts occupy the same space in the assembly (wrong fit, wrong position).
- **Collision:** parts that are clear at rest hit each other while moving.
- **Why before a prototype:** a clash found on screen costs minutes; one found in a prototype costs new tooling and weeks of delay.
- **Needs solid models:** only solids have volume, so only they can be intersected; wireframes cannot.
- **Example:** checking that a car door, its hinge and the wiring clear each other through the full opening angle.

How CAD finds them

Static: the system intersects every pair of solids in the assembly and reports overlapping volumes. Dynamic: during the motion study it checks the parts at each time step, stops at the first contact and highlights it. Clearance checks report gaps below a set minimum.

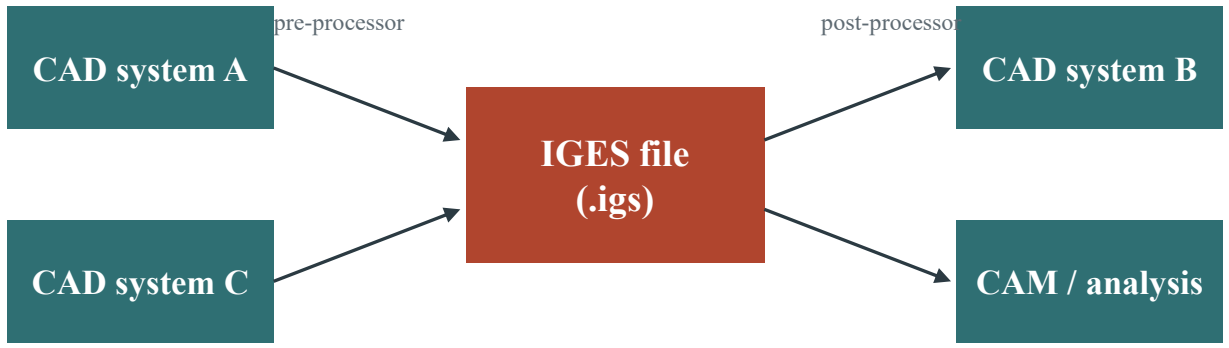
08

Data exchange and CAD/CAM integration

Moving models between CAD systems with IGES and STEP, and other CAD features

IGES lets different CAD systems exchange drawings and models through a neutral file

Neutral-file exchange through IGES



Each system needs only one translator to and from IGES, not one for every other system.

IGES file: 80-column ASCII records in five sections

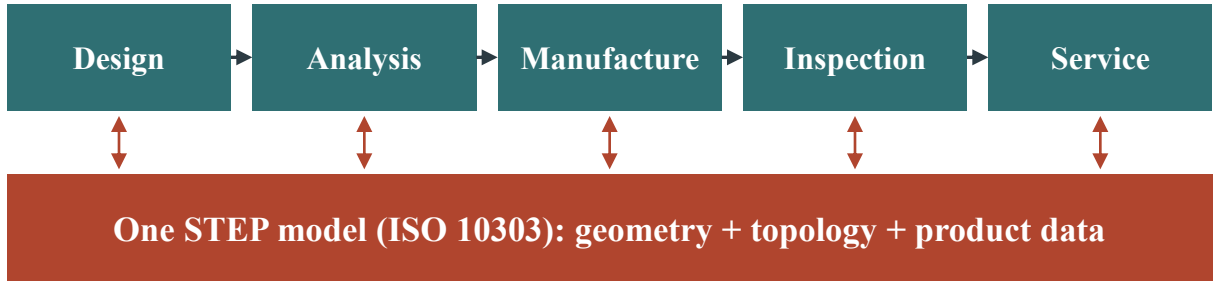
S	G	D	P	T
Start: human-readable notes	Global: units, author, system	Directory entry: one per entity	Parameter data: coordinates, values	Terminate: section counts

In short: IGES = Initial Graphics Exchange Specification, a neutral file format (ANSI, 1980) for exchanging CAD data between different systems.

- **Problem:** each CAD vendor stores models in its own format, so system A cannot read system B's files.
- **Neutral file:** a pre-processor writes the model to a standard IGES file; the receiving system's post-processor reads it.
- **What it carries:** points, lines, arcs, splines, surfaces, annotation and dimensions; later versions add solids.
- **Limitations:** mainly geometry; features, history and product data are lost; some entities translate badly, so models need repair.
- **Developed:** by US NBS with industry (1979–80); widely used for drawings and surfaces.

STEP exchanges the complete product model and is the standard in modern PLM

STEP carries the product model through its life cycle



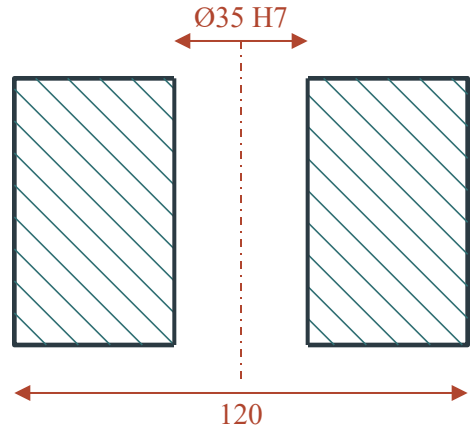
Application protocols: AP203 (mechanical design), AP214 (automotive), AP242 (managed model-based 3D engineering, with PMI)

Basis	IGES	STEP
Full name	Initial Graphics Exchange Specification	Standard for the Exchange of Product model data
Standard	ANSI, 1980	ISO 10303, 1994 onward
Content	Mainly geometry and drawings	Complete product model: solids, assemblies, materials, tolerances
Solids	Limited, often need repair	Full B-rep solids, reliable
PLM use	Legacy data	Modern PLM standard

- **STEP:** Standard for the Exchange of Product model data, ISO 10303. Files usually end in .stp or .step.
- **Scope:** not just shapes: it describes the whole product, including assembly structure, materials, tolerances and annotations.
- **Application protocols:** subsets for each industry, so both systems agree exactly on the content.
- **Used in PLM:** STEP (especially AP242) is used in modern Product Lifecycle Management, because it keeps product data and is an international, vendor-neutral standard.
- **IGES today:** still used for surfaces and older drawings.

Other CAD features finish the drawing and link CAD to manufacturing

A drawing finished with automatic features



No.	Part	Qty
1	Housing	1
2	Shaft	1
3	Bearing 6205	2
4	Circlip	2

Bill of materials, generated

CAD/CAM integration

CAD systems offer links to programming languages (FORTRAN in Groover's time; C++, Python and APIs today) so companies can add their own analysis and planning programs. The same database then feeds NC programming, process planning and inspection.

- **Automatic crosshatching:** section areas are hatched from the model, with pattern and spacing chosen by material.
- **Text:** notes and title blocks with control of size, style and justification.
- **Semi-automatic dimensioning:** dimensions taken from the database; linear or angular, metric or inch, decimal or fraction, with tolerances.
- **Bill of materials:** assembly and material lists generated automatically.
- **Effect:** less drafting time and fewer errors between model and drawing (Groover Sec. 6.8).

09

BPUT previous year questions

Module II question bank, mark-wise

2-mark questions

1. What is a 'constraint' in 2D sketching?

2. What is the full form of the IGES exchange format?

3. What are analytic curves? Give one example.

4. What are the basic entities used in geometric / CAD modeling? Name at least four drawing entities.

5. Explain the geometric transformations (translation, rotation and scaling) in relation to CAD.

6. What are synthetic curves?

6-mark questions

1. What is feature-based modeling? Explain with common features such as hole, fillet and shell.
2. Explain the STEP standard and IGES exchange format. Which format is used in modern PLM systems?
3. Explain the extrude and revolve features with simple sketches.
4. Differentiate between parametric and non-parametric design.
5. Differentiate between analytic and synthetic curves with simple examples.
6. Explain various curve generation techniques with suitable examples.
7. Draw the wireframe model and solid model of a rectangular block.
8. What do you mean by transformation? Explain the transformations used in CAD with matrix notation.
9. Describe the various types of surfaces used in CAD.
10. Explain the various methods of solid modeling techniques used in CAD.
11. Distinguish between wireframe and solid modeling.

16-mark questions

Q1. Explain the process of kinematic simulation within an assembly model. How can CAD tools be used to check for interferences and collisions before a physical prototype is ever built?

Q2. Explain the different types of CAD models: wireframe, surface and solid modeling. Discuss their specific use cases in engineering.

Q3. A line is defined by its end points (5, 5) and (25, 30) in a 2D graphics system. Express the line in matrix notation and (I) scale it by a factor of 3, (II) translate the original line by 3.0 units in x and 4.0 units in y, (III) rotate the original line by 30° counter-clockwise about the origin.

Q4. Explain the significance of solid modeling in engineering design. Compare CSG and B-rep on data structure, ease of editing and computational requirements.

MODULE III

CAM Introduction, NC and NC Part Programming

MFPC3005 Computer Aided Design and Manufacturing (CAD/CAM)

Reference: Groover & Zimmers, CAD/CAM, Ch. 7 and 8

Module III roadmap

01	**Introduction to CAM and NC**	NC basics, the three components and the MCU
02	**The NC procedure**	Five steps from process planning to production
03	**NC coordinate systems**	Axes, fixed/floating zero, absolute vs incremental
04	**NC motion control systems**	Point-to-point, straight-cut and contouring
05	**Applications and economics of NC**	Where NC pays off, advantages and disadvantages
06	**NC words and codes**	Block format, G- and M-codes, offsets, canned cycles
07	**Manual part programming**	Manuscript, textbook example, rectangular-profile program
08	**Computer-assisted part programming**	Programmer and computer tasks, postprocessor, CAM setup
09	**The APT language**	Geometry, motion, postprocessor statements; full programs
10	**NC programming with interactive graphics**	Tool path generation, verification, advantages
11	**BPUT previous year questions**	Mark-wise question bank

By the end of Module III you should be able to

- Explain what NC is, its three basic components and the role of the machine control unit
- Describe the NC procedure and set up NC axes, zero points, absolute and incremental positioning
- Compare point-to-point, straight-cut and contouring motion control systems
- Assess the advantages, disadvantages and economics of NC
- Read and write NC blocks with G- and M-codes, tool offsets, canned cycles and subprograms
- Write a manual part program for a milling profile
- Explain computer-assisted programming, the postprocessor and CAM machining setup
- Write APT geometry, motion, postprocessor and auxiliary statements for a part
- Explain NC programming with interactive graphics and its advantages

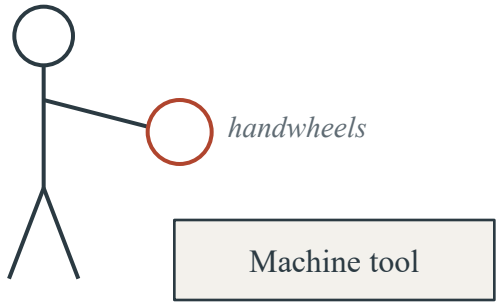
01

Introduction to CAM and NC

What numerical control is, where it came from, and its three basic components

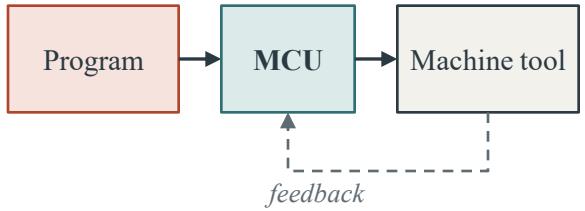
Numerical control operates a machine tool from a coded program instead of by hand

Conventional machining



The operator reads the drawing and moves every slide by hand

Numerical control

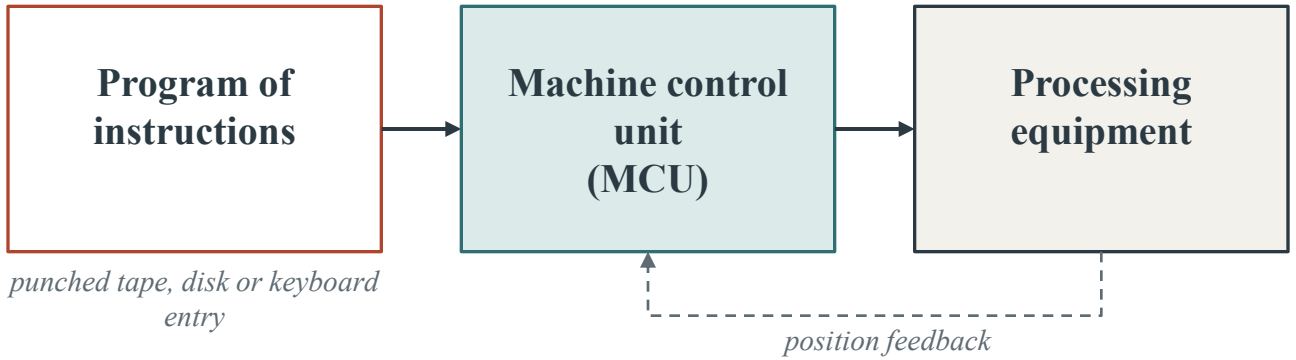


Coded instructions move the slides; the operator loads parts and watches

Example
A plate needs 40 holes at exact positions. Manually, the operator positions the table 40 times from the drawing. With NC, the hole coordinates are programmed once and the machine drills every plate in the batch at the same positions, batch after batch.

- **Numerical control:** a form of programmable automation in which the machine is controlled by numbers, letters and symbols coded into a program.
- **Origin:** developed at MIT for the U.S. Air Force around 1952 to machine complex aircraft parts.
- **CAM link:** NC was the first step of computer-aided manufacturing: CAM began with NC.
- **Change of job:** a new part needs only a new program, not new tooling, so NC suits small and medium batches.

Every NC system has three basic components

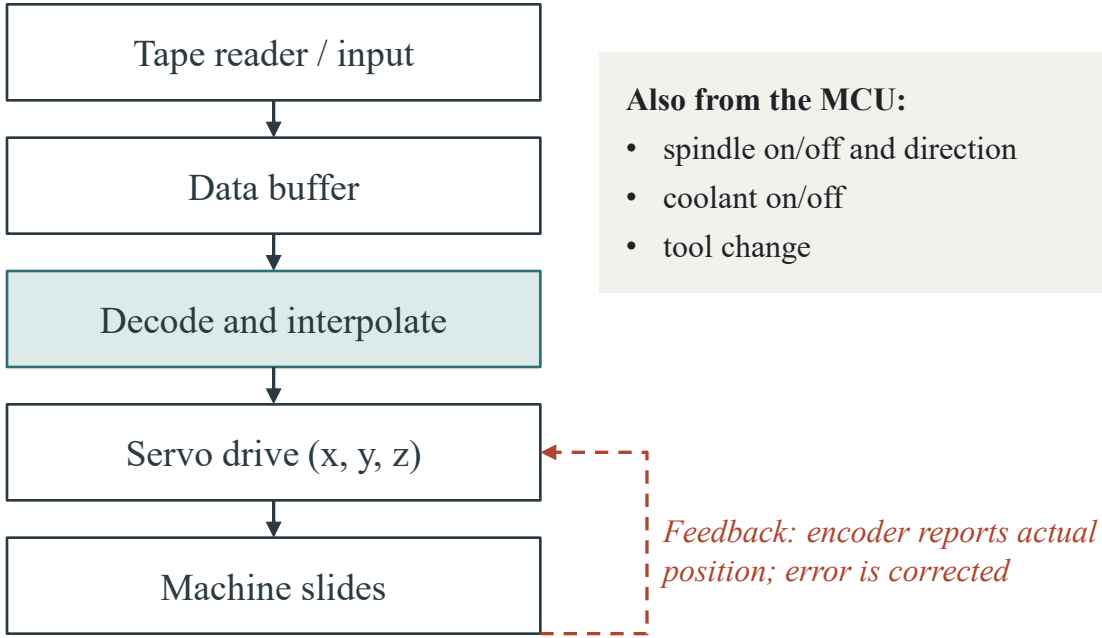


Basic components of an NC system (after Groover Fig. 7.1)

Example: drilling a flange
Program: the hole positions, spindle speed and feed coded as blocks.
MCU: reads each block, sends motion commands to the table drives and starts the spindle.
Processing equipment: the drilling machine with its table, spindle and drive motors makes the holes.

- **Program of instructions:** the detailed step-by-step commands that tell the machine what to do: positions, speeds, feeds, tool changes. Stored on punched tape or in memory.
- **Machine control unit:** electronics and hardware that read and interpret the program and convert it into mechanical actions of the machine.
- **Processing equipment:** the machine tool that does the useful work: the worktable, spindle, motors and controls needed to machine the part.

The machine control unit reads the program and drives the machine



Inside the machine control unit

- **Reads:** the program block by block from tape, disk or memory and stores it in a buffer.
- **Decodes:** separates each block into words: motion (G, X, Y, Z), speed (S), feed (F), tool (T) and miscellaneous (M) functions.
- **Interpolates:** computes intermediate points so the tool follows a line or arc.
- **Drives and checks:** sends signals to the servo motors and compares actual with commanded position (closed loop).

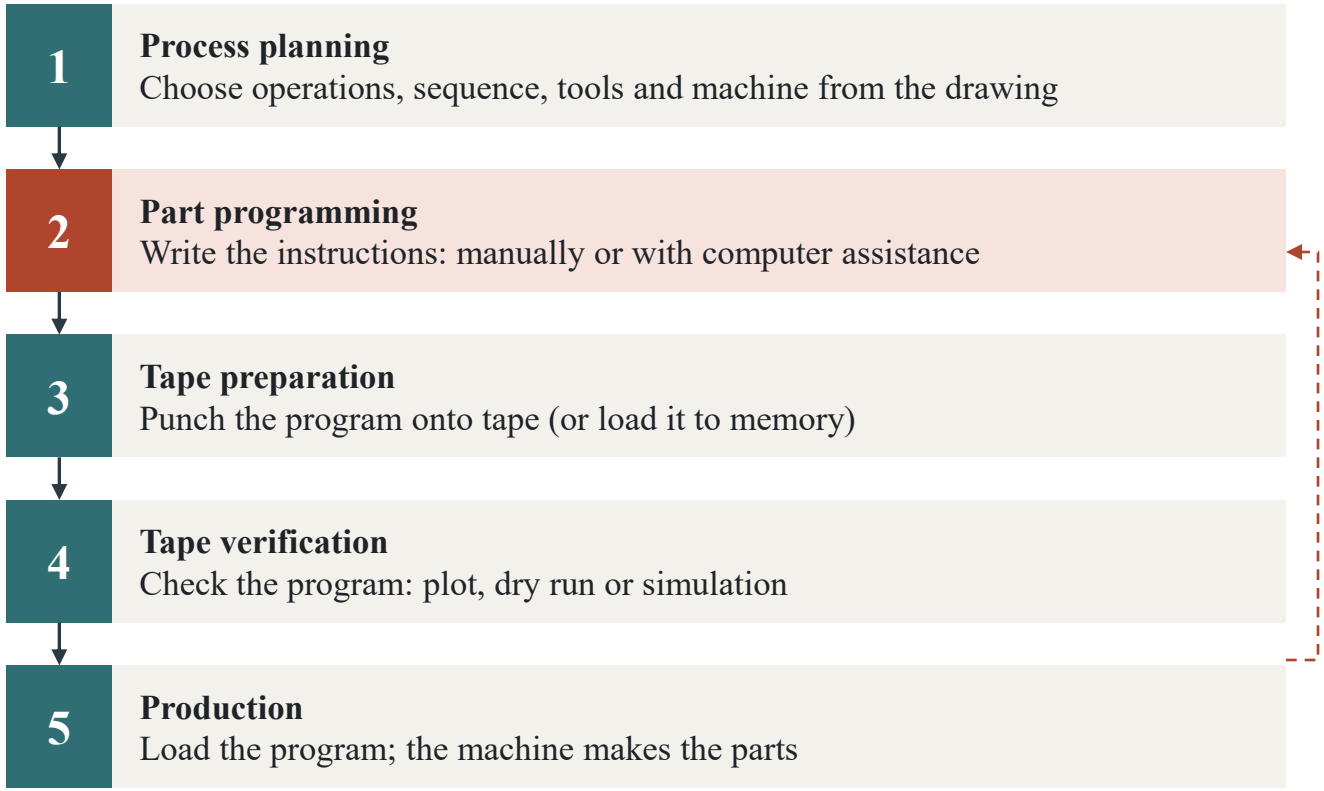
In short
The MCU reads and decodes the part program, interpolates the tool path, drives the axes through servo motors, uses feedback to correct position, and switches spindle, coolant and tools.

02

The NC procedure

The five steps from a part drawing to a finished part

The NC procedure has five steps from process planning to production



If verification finds an error, the program is corrected and verified again before production starts.

- **Process planning:** the planner decides how the part is made; NC is used for the operations it suits.
- **Part programming:** converts the plan into coded NC instructions.
- **Tape preparation:** in older systems the program was punched on 1-inch paper tape; today it is transferred electronically.
- **Verification:** catches collisions and wrong dimensions before metal is cut.
- **Production:** the operator sets up the workpiece and tools and runs the program.

A part program can be written manually or with computer assistance

Manual part programming

- Programmer calculates every coordinate by hand
- Writes blocks on a manuscript form
- Suits simple point-to-point jobs
- Slow and error-prone for contours

Computer-assisted programming

- Programmer describes geometry and tool path in a language such as APT
- Computer does the arithmetic, cutter offset and postprocessing
- Suits complex parts and contours

Example
Drilling 3 holes in a plate is quick to program by hand. Milling a cam with a curved profile needs hundreds of points, so a computer language or CAD/CAM system is used.

- **Part programming:** planning and documenting the sequence of processing steps the NC machine must perform.
- **Manual:** all calculations and coding are done by the programmer; the result is written directly in machine code.
- **Computer-assisted:** the programmer writes English-like statements; the computer converts them to machine code.
- **With interactive graphics:** geometry comes from the CAD model and the tool path is built on screen (section 10).

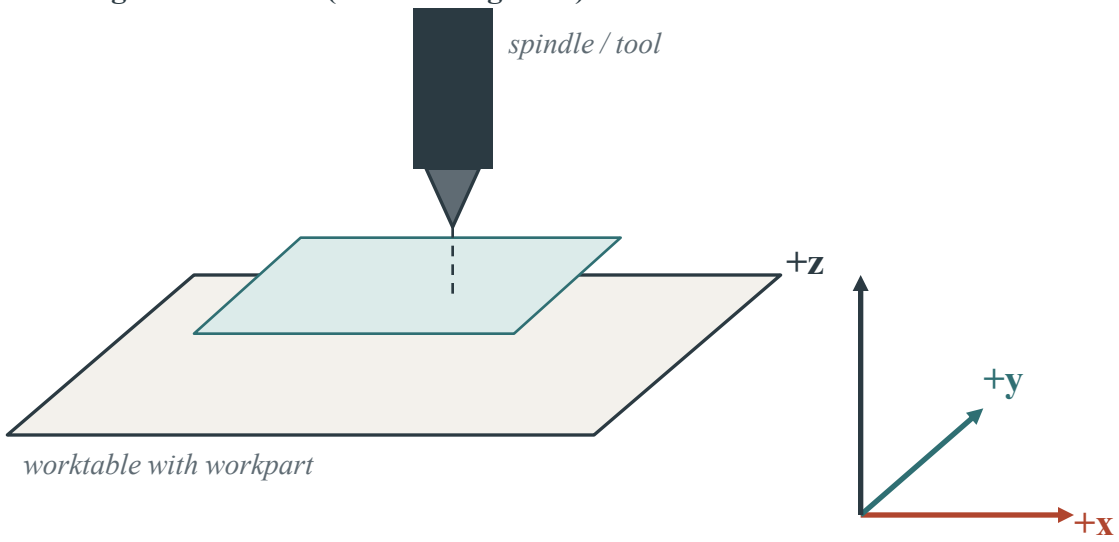
03

NC coordinate systems

Machine axes, zero points, and absolute versus incremental positioning

NC axes follow the right-hand rule: x, y and z for drills and mills

Drill press / milling machine axes (Groover Fig. 7.5a)



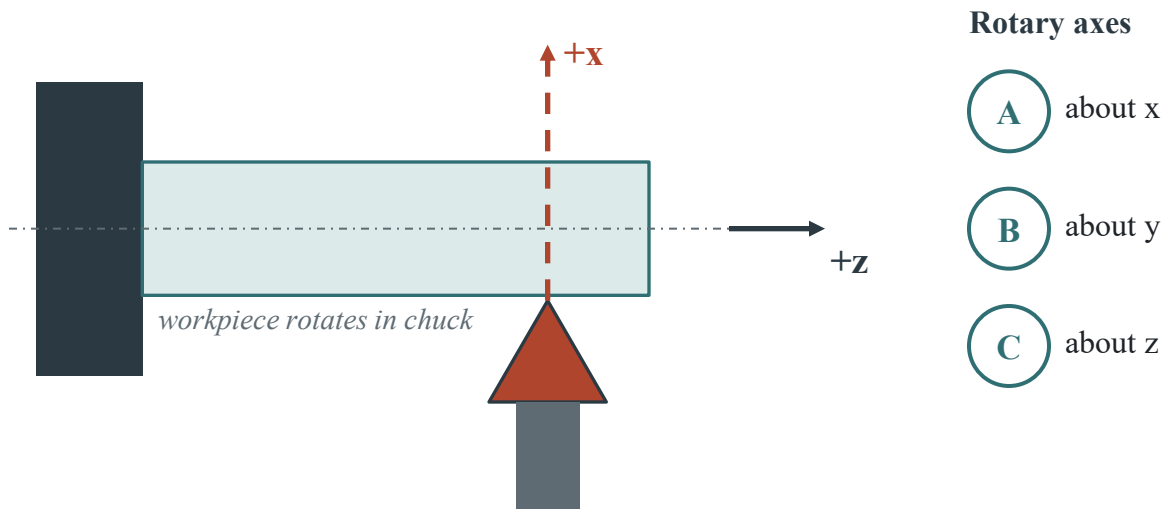
z is the spindle axis; +z moves the tool away from the work

- **x and y:** the two horizontal axes of the worktable; x is usually the longer travel.
- **z axis:** always parallel to the spindle; on a drill or vertical mill it is vertical.
- **Direction:** +z is away from the workpiece, so a negative z move feeds the tool into the part.
- **Use:** the programmer writes every position as x, y, z coordinates in this system.

Right-hand rule
Point the thumb along +x and the index finger along +y; the middle finger then points along +z. All NC machines use this convention.

A lathe uses two axes, and rotary axes A, B and C turn about x, y and z

Lathe (turning) axes



Rotary axes

- **A** about x
- **B** about y
- **C** about z

z is along the spindle (work) axis; x is the radial cross-slide

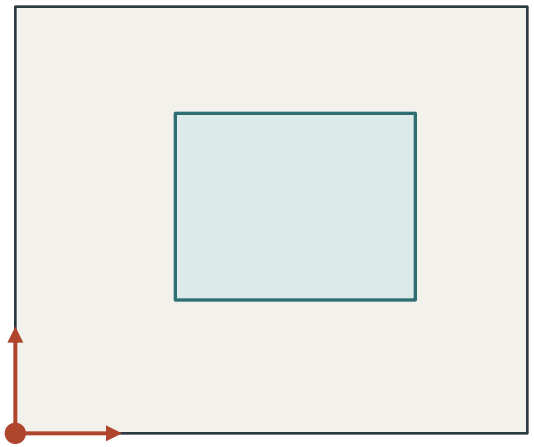
- **z axis:** on a lathe the z axis is the axis of rotation of the work, running along the bed.
- **x axis:** the cross-slide direction, perpendicular to the work axis; +x moves the tool away from the centre.
- **Rotary axes:** A, B and C are rotations about x, y and z, used on indexing tables and 4- and 5-axis machines.
- **Rule:** the right-hand rule still applies to every axis.

Example

Turning a 40 mm shaft down to 36 mm: the tool moves −x by 2 mm (radius) and then feeds along −z for the length of the cut.

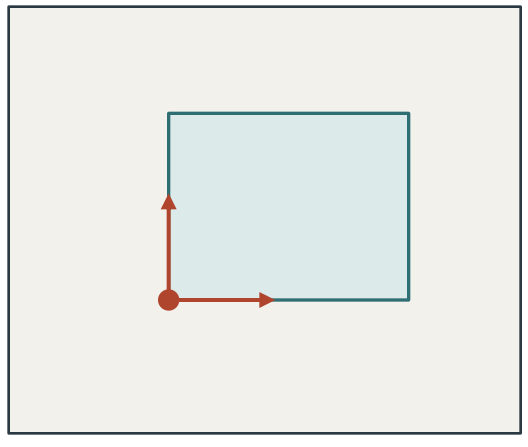
The zero point can be fixed by the machine or floated by the operator

Fixed zero



origin set by the machine builder, e.g. lower-left corner of the table

Floating zero



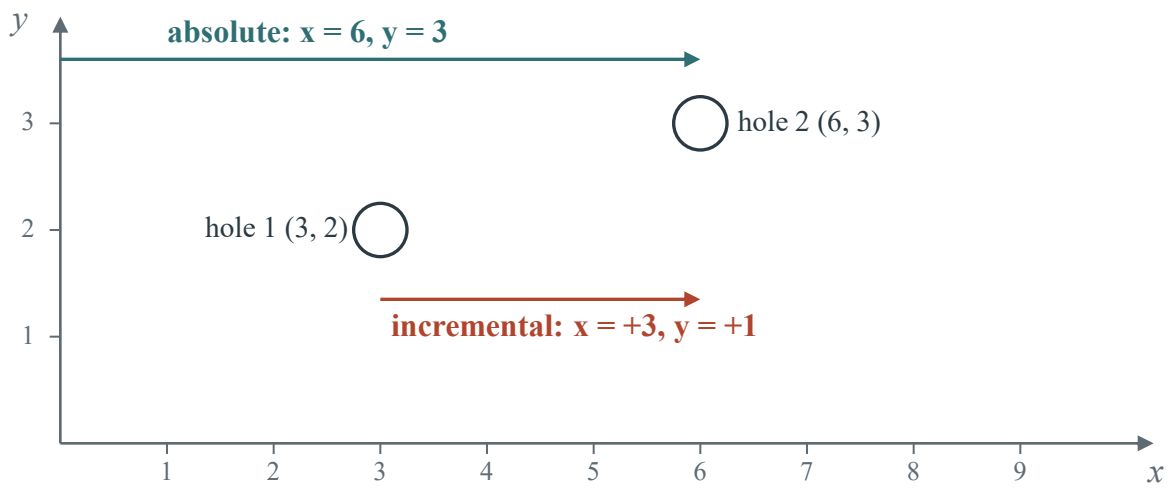
origin placed by the operator at a convenient point on the part

- **Fixed zero:** the origin is always at the same place on the machine table; all coordinates are positive from it.
- **Floating zero:** the operator can set the origin anywhere on the table; more flexible and common on modern machines.
- **Choice:** the programmer picks a zero that makes the part's dimensions easy to use, often a datum corner.
- **Benefit:** the same program works wherever the part is clamped, once the zero is set.

Target point
With floating zero the programmer names a target point (for example a corner of the part) and the operator moves the tool there and sets it as zero before starting.

Absolute positioning measures from the origin; incremental measures from the last point

Moving from hole 1 to hole 2 (after Groover Fig. 7.6)



Absolute (G90)
+ an error in one block does not carry into the next
+ easy to check against the drawing
− all positions must be known from one origin

Incremental (G91)
+ simple for repeated patterns and subprograms
+ convenient for step moves
− one wrong move shifts every later position

- **Absolute positioning:** every position is given as coordinates measured from the fixed or floating zero.
- **Incremental positioning:** each position is given as the distance and direction from the previous position.
- **Example:** hole 2 at (6, 3): absolute x = 6, y = 3; incremental from hole 1 at (3, 2) is x = +3, y = +1.
- **In programs:** G90 selects absolute and G91 incremental; many programs mix both.

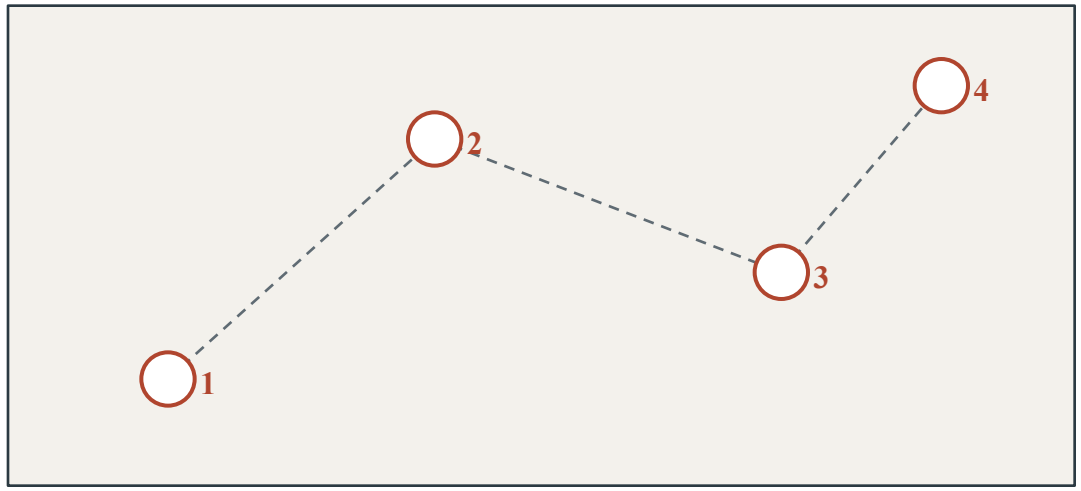
04

NC motion control systems

Point-to-point, straight-cut and contouring control

Point-to-point NC moves the tool to a position, then the operation is done

Point-to-point (PTP) drilling (after Groover Fig. 7.7)



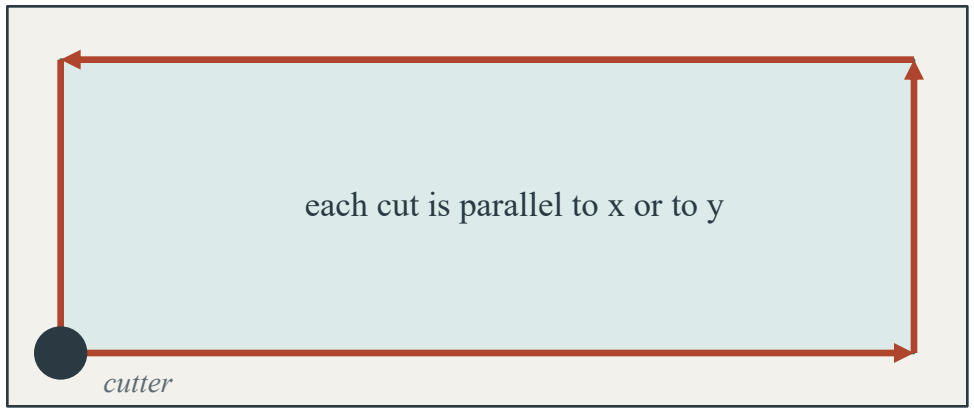
dashed: rapid move, path not controlled circle: drill here

Example
Drilling 4 holes: the table moves at rapid traverse to hole 1, stops, the drill feeds down and retracts, then the table moves to hole 2. The path between holes does not matter.

- **Also called:** positioning system.
- **How it works:** the tool is moved quickly to each programmed position; cutting happens only when it has stopped.
- **Path:** not controlled; the machine may move one axis at a time or both together.
- **Uses:** drilling, reaming, tapping, boring, punching and spot welding.
- **Cost:** the simplest and cheapest NC control.

Straight-cut NC cuts while moving parallel to one machine axis

Straight-cut milling (after Groover Fig. 7.8)



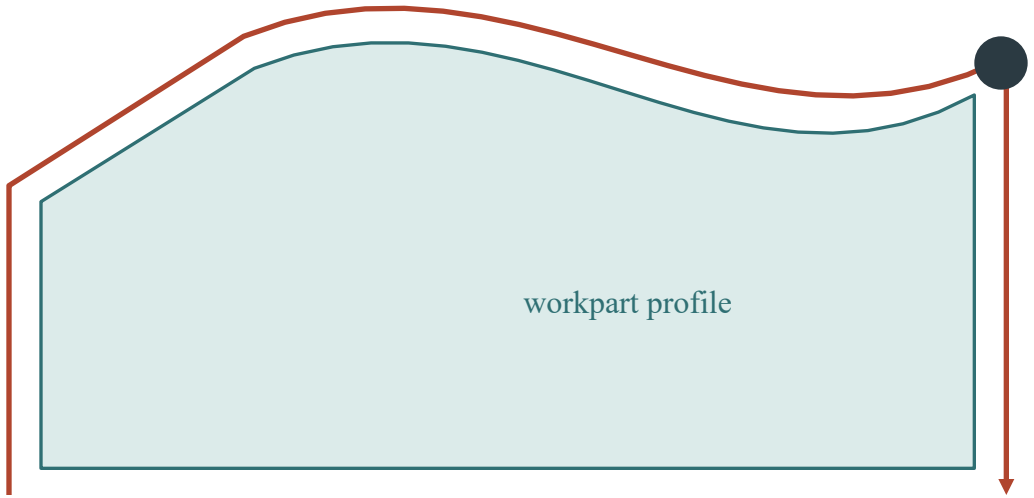
red: cutting at controlled feed along one axis at a time

- **How it works:** the tool cuts at a controlled feed rate while moving along one axis at a time.
- **Path:** straight lines parallel to the x, y or z axis only; no angled or curved cuts.
- **Uses:** milling rectangular shapes, slots, steps and pockets with square sides.
- **Capability:** includes all PTP functions plus controlled-feed cutting.

Example
Face-milling the four edges of a rectangular plate: the cutter moves along x at the feed rate, then along y, then back along x and y. A 45° chamfer cannot be cut, because that needs two axes moving together.

Contouring NC controls several axes at once to follow any path

Contouring (continuous path) (after Groover Fig. 7.9)



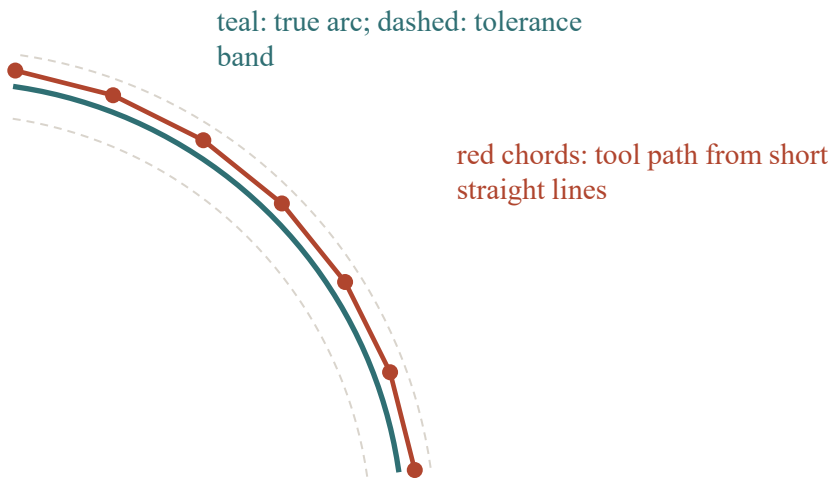
red: tool path; x and y move together at controlled rates

- **Also called:** continuous-path NC.
- **How it works:** two or more axes are controlled at the same time, with their speeds coordinated by the MCU.
- **Path:** any shape: angled lines, circles, curves, and 3D surfaces with 3- to 5-axis control.
- **Uses:** milling profiles, dies and moulds, turning tapers and contours, flame and laser cutting.
- **Cost:** the most complex and expensive control.

Example
Milling a cam profile or a turbine blade edge: the MCU changes the x and y speeds continuously so the cutter follows the curve. Angled lines, arcs and 3D surfaces all need contouring.

Contouring approximates a curve with short straight or circular segments

Linear interpolation of an arc, with tolerance band



outside tolerance + inside tolerance = band the path must stay in

- **Interpolation:** the MCU calculates intermediate points between programmed points so the tool moves smoothly.
- **Linear:** straight segments between points; any curve is approximated by many short lines.
- **Circular:** the MCU generates an arc directly from its end points and centre (G02, G03).
- **Tolerance:** inside, outside or band tolerance limits how far the segments may deviate from the true curve.

Example
An arc of radius 50 mm with a tolerance of 0.01 mm: the MCU splits it into many chords so that no chord strays more than 0.01 mm from the true arc. A tighter tolerance means more, shorter segments.

The three motion control systems differ in capability and cost

Feature	Point-to-point	Straight-cut	Contouring
Cutting during motion	No	Yes, along one axis	Yes, along any path
Axes controlled together	None (only end position)	One at a time	Two to five
Tool path	Not controlled	Parallel to an axis	Lines, arcs, curves, surfaces
Typical operations	Drilling, tapping, punching, spot welding	Face and slot milling, steps	Profile milling, dies, moulds, cams
Control complexity	Simple	Medium	Complex
Cost	Lowest	Medium	Highest

Each higher system can do everything the lower ones can (Groover & Zimmers, Sec. 7.5).

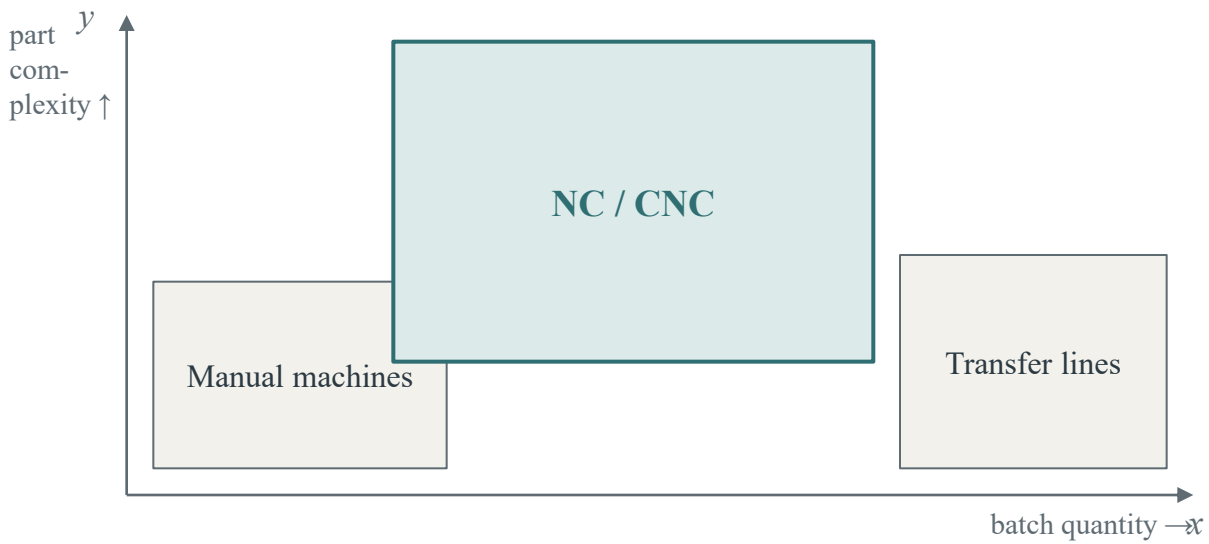
05

Applications and economics of NC

Where NC is used, the jobs it suits, and its advantages and disadvantages

NC is used in machining and many other processes, and suits certain kinds of jobs

Where NC fits: batch size and part complexity



- **Machining:** milling, drilling, turning, boring and grinding are the main NC uses.
- **Other processes:** sheet-metal punching and bending, welding, flame and laser cutting, inspection, assembly, drafting.
- **Jobs that suit NC:** parts made in small or medium batches, in repeat orders, with complex geometry or close tolerances, needing many operations, and changing often.
- **Not suited:** very high volumes of one simple part, where hard automation is cheaper.

Example
An aircraft wing rib is made in batches of 20, has 60 pockets and many curved edges, and changes with every design revision. It is an ideal NC job.

NC cuts non-productive time, fixturing and lead time

Share of time the spindle is actually cutting



Illustrative split: NC removes much of the time spent setting up, positioning and checking between cuts.

Michigan survey: lead time cut by 26–44% after moving jobs to NC

Example
A part needs 5 operations. On manual machines it is set up 5 times with jigs. On an NC machining centre it is clamped once in a simple fixture and all 5 operations run from one program.

- **Less non-productive time:** fewer setups, faster positioning and automatic tool changes raise the share of time spent cutting.
- **Simpler fixtures:** the program, not a jig, locates the tool, so fixtures only hold the part; cheaper and quicker to make.
- **Shorter lead time:** fewer setups and quicker changeover reduce the time from order to finished part.
- **Less operator effort:** the operator loads, unloads and monitors instead of guiding every cut.

NC also improves flexibility and quality, and reduces inventory and floor space

Flexibility
A new part needs a new program, not new tooling

Quality
More consistent parts; 31–44% less scrap reported

Less inventory
Smaller batches and shorter lead times mean less stock

Less floor space
One NC machine replaces several conventional ones

Easier changes
Design changes are made by editing the program

Less inspection
Repeatable accuracy reduces checking between parts

Example
A batch of 50 shafts is needed every month. With NC, 50 are made per order instead of a year's supply of 600, so stock and storage space fall.

- **Flexibility:** the same machine handles many different parts; switching jobs takes minutes.
- **Better quality control:** the machine repeats the programmed path, so variation between parts falls.
- **Reduced inventory:** work-in-process and finished stock are lower because batches can be smaller.
- **Reduced floor space:** fewer machines and fewer fixtures need less shop space.
- **Survey data:** Groover reports 31–44% scrap reduction (Michigan survey).

NC has disadvantages: higher investment, maintenance and skill requirements

– **Higher investment:** An NC machine costs several times a conventional one

– **Higher maintenance:** More complex electronics; maintenance cost up 48–63%

– **Skilled people:** Part programmers and trained maintenance staff are needed

– **Higher utilization needed:** The high cost is recovered only if the machine runs most shifts

Economics in short
NC pays off when its savings in setup, lead time, scrap and inventory are larger than its higher investment and running costs, which usually means high machine use.

- **Investment:** the machine and control are expensive, and spares and tooling cost more.
- **Maintenance:** needs specialists in electronics and hydraulics as well as mechanics.
- **Programming:** programs must be written and verified before the first part; a cost for every new job.
- **Utilization:** to repay the cost the machine is often run two or three shifts a day.

06

NC words and codes

How a program block is built from words, and the main G- and M-codes

An NC block is a line of words, each a letter address followed by a number

One block, word by word



Word	Meaning
N	sequence (block) number
G	preparatory function: type of motion
X, Y, Z	coordinate data
F	feed rate (mm/min)
S	spindle speed (rpm)
T	tool number
M	miscellaneous function
EOB (;)	end of block

- **Block:** one complete instruction to the machine, ended by the end-of-block character.
- **Word:** a letter address plus a number, e.g. X50.0. Each word gives one item of data.
- **Word address format:** words carry their letter, so they can be written in any order and unchanged words can be left out. This is the standard format today.
- **Other tape formats:** fixed sequential (every word in a fixed place) and tab sequential (words separated by TAB).
- **Standards:** EIA RS-274 and ISO 6983 define the codes; tape uses EIA RS-244 or ASCII.

G-codes are preparatory functions that set the type of motion and mode

G00	Rapid positioning	**G28**	Return to reference point
G01	Linear interpolation (feed)	**G40**	Cancel cutter radius compensation
G02	Circular interpolation, clockwise	**G41 / G42**	Cutter compensation left / right
G03	Circular interpolation, counter-clockwise	**G43**	Tool length compensation
G04	Dwell	**G80**	Cancel canned cycle
G17	XY plane selection	**G81**	Drilling canned cycle
G20 / G21	Inch / metric input	**G90 / G91**	Absolute / incremental

Example
N020 G01 X60.0 Y20.0 F120 ; feed in a straight line to (60, 20) at 120 mm/min.

- **Meaning:** G-codes prepare the control for the motion that follows: the type of interpolation, the plane, units and positioning mode.
- **Modal codes:** most G-codes stay active until another code of the same group replaces them (G01 stays on for following blocks).
- **Motion:** G00 moves at rapid speed without cutting; G01, G02 and G03 cut at the programmed feed.
- **Example of an arc:** G02 X40 Y20 I10 J0 cuts a clockwise arc to (40, 20) with its centre 10 mm to the right of the start.

M-codes are miscellaneous functions that switch the spindle, coolant and program

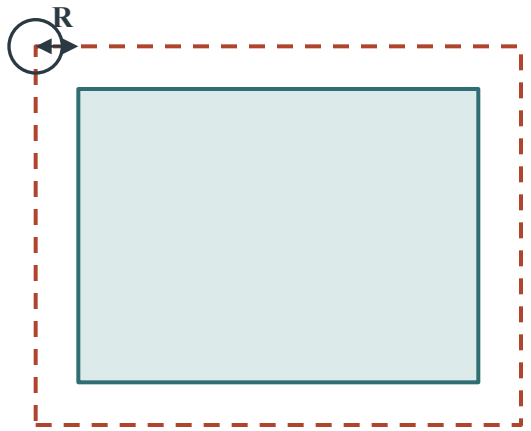
M00	Program stop	**M06**	Tool change
M01	Optional stop	**M08**	Coolant on
M02	End of program	**M09**	Coolant off
M03	Spindle on, clockwise	**M30**	End of program, rewind / reset
M04	Spindle on, counter-clockwise	**M98**	Call subprogram
M05	Spindle stop	**M99**	End subprogram, return

Example
N010 T01 M06 ; change to tool 1
N020 S800 M04 ; spindle 800 rpm counter-clockwise (e.g. left-hand tap)
N030 M08 ; coolant on … N200 M30 ; end and rewind

- **Meaning:** M-codes switch machine functions on and off; they are not motions.
- **M04:** starts the spindle counter-clockwise, used for left-hand tools, left-hand taps and some turning set-ups.
- **M08:** turns flood coolant on to cool the tool and work, flush chips and improve finish; M09 turns it off.
- **M30:** ends the program, stops spindle and coolant, and rewinds or resets to the start ready for the next part.

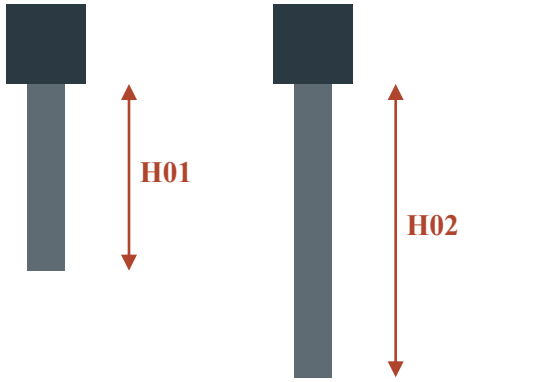
Tool offsets let one program work for cutters of different size and length

Cutter radius offset (G41/G42)



teal: part outline (programmed)
red: tool-centre path, offset by the cutter radius

Tool length offset (G43)



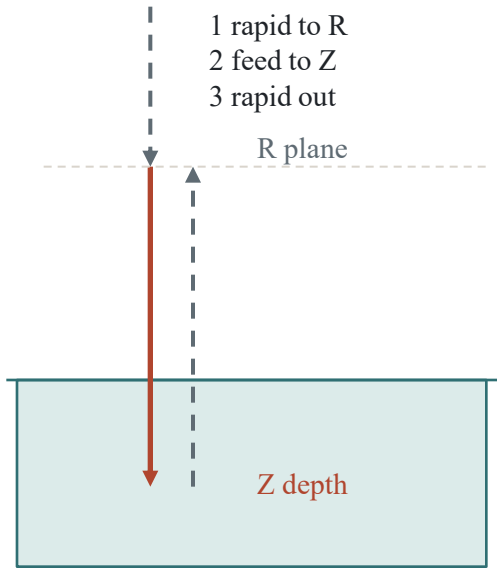
each tool's length is stored, so z = 0 means the work surface for every tool

- **Tool offset:** a correction value stored in the MCU and applied to the programmed path for the actual tool.
- **Radius compensation:** G41 (tool left of path) and G42 (right) shift the path by the cutter radius; G40 cancels it.
- **Length compensation:** G43 with an H number adds the tool's length so every tool reaches the right depth.
- **Why:** the programmer writes part dimensions, not tool-centre positions; worn or different tools need no reprogramming.

Example
A profile is programmed on the part outline. With a 10 mm cutter the offset register holds R = 5; after regrinding to 9.8 mm the operator enters 4.9 and runs the same program.

A canned cycle performs a whole drilling or tapping sequence from one block

G81 drilling cycle

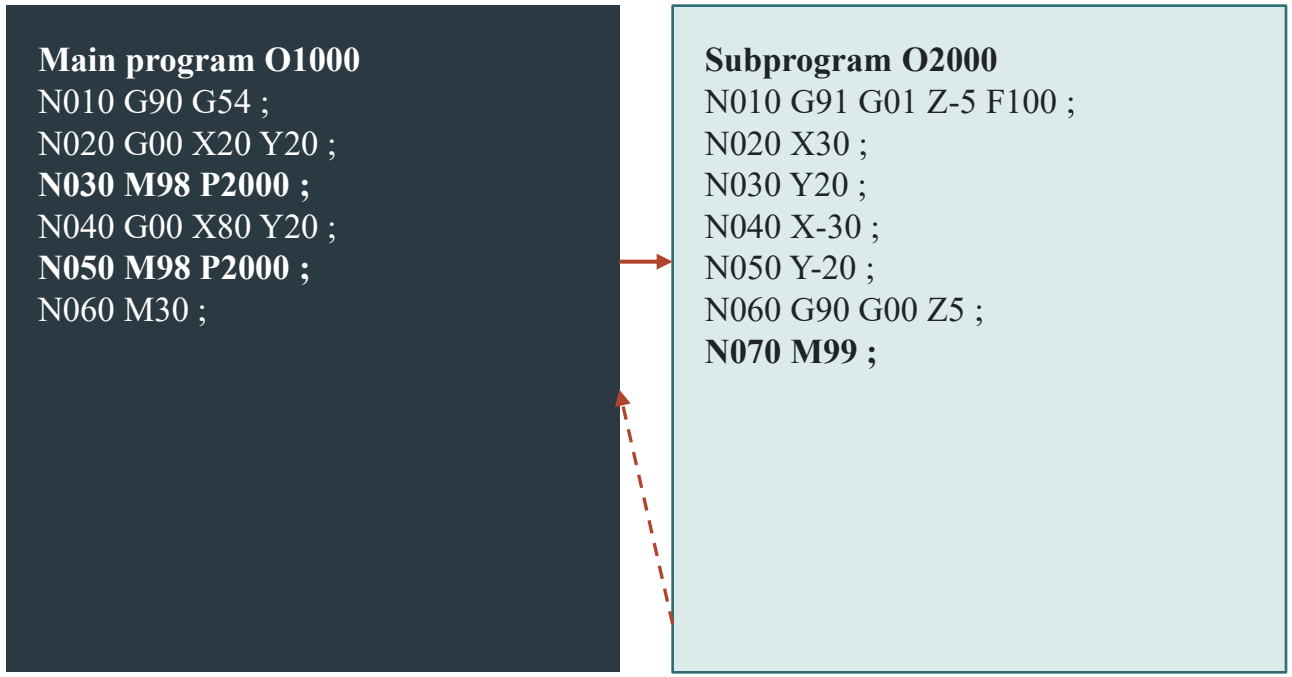



```
N010 G90 G21 G17 ;
N020 T02 M06 ;
N030 S1200 M03 ;
N040 G00 X20 Y20 Z5 M08 ;
N050 G81 X20 Y20 Z-15 R2 F80 ;
N060 X60 ;
N070 X100 ;
N080 G80 ;
N090 G00 Z50 M09 ;
N100 M30 ;
```

- **Canned cycle:** a stored sequence of moves called by one G-code; the MCU fills in the individual motions.
- **Example:** G81 is a simple drilling cycle: rapid to the R plane, feed to depth Z, rapid out.
- **Other cycles:** G82 drill with dwell, G83 peck drilling, G84 tapping, G85 boring; G80 cancels any cycle.
- **Benefit:** shorter programs, fewer errors, and easy repetition at many hole positions.

Example
Blocks N050–N070 drill three 15 mm deep holes at x = 20, 60 and 100. Only the new x is written for the second and third holes; G80 cancels the cycle.

Subprograms and macros store repeated machining sequences

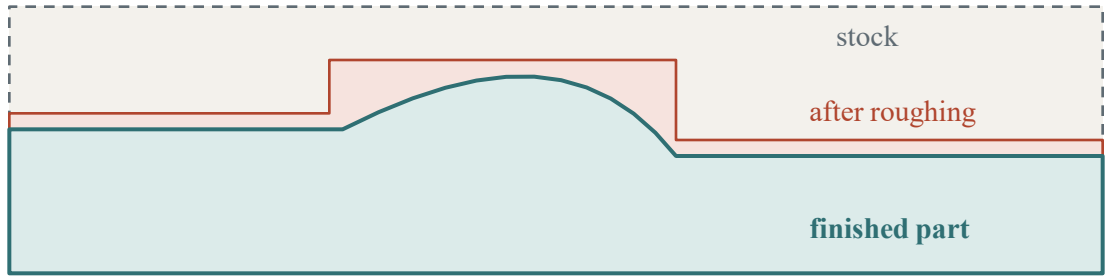


Example
The same 30 × 20 mm pocket is needed at two places. It is written once as subprogram O2000 in incremental moves and called twice with M98 P2000.

- **Subprogram:** a separate program for a repeated operation, called from the main program with M98 P(number) and ended with M99, which returns control.
- **Macro:** a subprogram with variables (#1, #2…) and logic, e.g. a bolt-circle macro that takes the number of holes and radius as inputs.
- **When used:** repeated features, families of similar parts, and standard operations such as bolt circles and pockets.
- **Benefit:** shorter, clearer programs; one change updates every use.

Roughing removes most of the material quickly; finishing gives final size and surface

Stock → roughing → finishing



	Roughing	Finishing
Purpose	Remove bulk material	Final size and surface
Depth of cut	Large (2–5 mm)	Small (0.2–0.5 mm)
Feed	High	Low
Speed	Lower	Higher
Allowance left	0.3–1 mm for finishing	None
Surface finish	Rough	Smooth, to tolerance

- **Roughing:** heavy, fast cuts that remove most of the stock in the shortest time, leaving a small even allowance.
- **Finishing:** light cuts at higher speed and lower feed that bring the part to final size, tolerance and surface finish.
- **Why two stages:** heavy cuts cause tool deflection and heat; a separate finishing pass gives accuracy.
- **Tools:** roughing uses strong, often larger cutters; finishing uses sharp tools, sometimes a ball-nose end mill for curves.

07

Manual part programming

Writing the program by hand, block by block, with worked examples

In manual part programming the programmer writes every block by hand

Part programming manuscript (layout)

N	G	X	Y	Z	F	S	T	M	Remarks
010	21								metric
020							01	06	tool change
030						800		03	spindle on
040	00	20.0	20.0	5.0				08	rapid, coolant
050	01			-10.0	60				drill
060	00			5.0					retract
070								30	end

When manual programming is used
Simple point-to-point jobs (drilling, tapping) and straight-cut milling with few positions. For contours with arcs and angles the coordinate arithmetic becomes long and error-prone, so computer-assisted programming is preferred.

- **Definition:** the part programmer works out every tool position and machine function and writes the program directly in machine code (G- and M-code blocks).
- **Manuscript:** a printed form with one column per word; each line is one block.
- **Programmer's tasks:** choose the zero point, list the coordinates, select speeds, feeds and tools, and write the blocks in order.
- **Output:** the manuscript is typed into the MCU or punched on tape.

Example: drilling three holes by manual programming

Part with three holes (dimensions in inches)

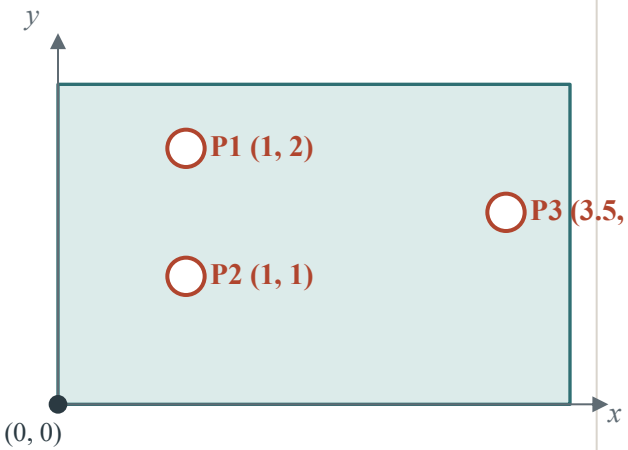


plate 1/2 in thick; drill through

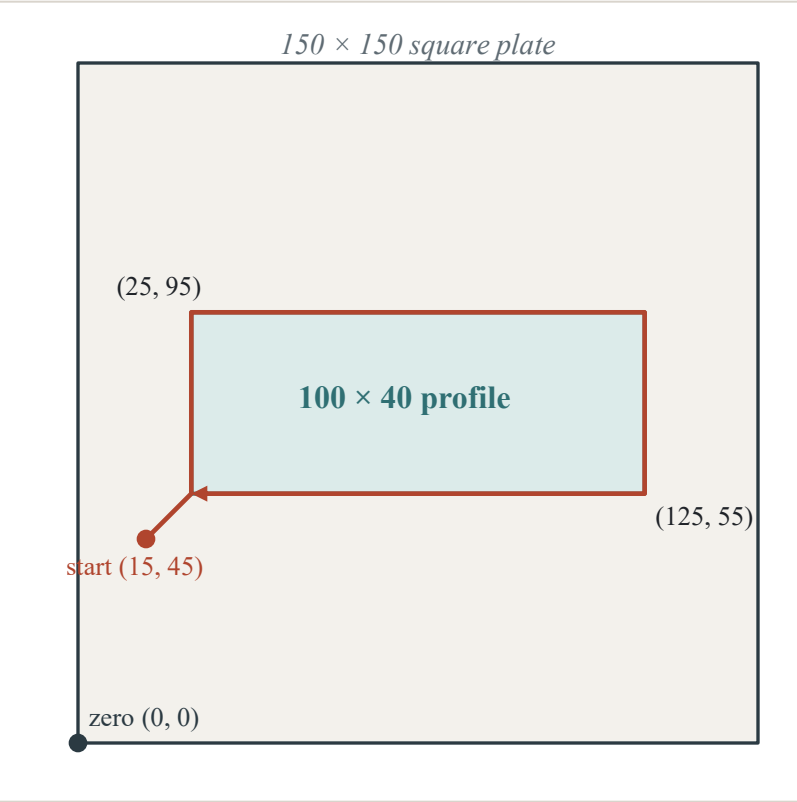
```
N010 G20 G90 ;
N020 T01 M06 ;
N030 S1000 M03 ;
N040 G00 X1.0 Y2.0 Z0.1 M08 ;
N050 G01 Z-0.6 F4.0 ;
N060 G00 Z0.1 ;
N070 Y1.0 ;
N080 G01 Z-0.6 ;
N090 G00 Z0.1 ;
N100 X3.5 Y1.5 ;
N110 G01 Z-0.6 ;
N120 G00 Z1.0 M09 ;
N130 M30 ;
```

- **Job:** drill three holes in a 1/2 in plate on a point-to-point NC drill (Groover Ex. 8.1 and 8.3 part).
- **Coordinates:** measured from the zero at the lower-left corner: P1 (1, 2), P2 (1, 1), P3 (3.5, 1.5).
- **Drill depth:** Z−0.6 takes the drill 0.1 in beyond the bottom of the plate.
- **Textbook version:** Groover writes the same job on a fixed-format tape manuscript with RWS, m06, m13 and m30 codes; this slide shows it in modern word-address form.

Reading the program

N040: rapid to P1 just above the work. N050: drill through at 4 in/min. N060: retract. N070–N090: move to P2 and drill. N100–N110: P3. N120–N130: retract, coolant off, end. Modal G00/G01 and coordinates carry over between blocks.

Example: milling a 100 × 40 mm rectangular profile, set-up and tool path



Set-up data
Material: mild steel plate 150 × 150
Tool T01: Ø10 mm end mill
Speed: 1000 rpm
Feed: 150 mm/min (80 plunge)
Depth of cut: 5 mm
Zero: lower-left plate corner, top face z = 0

- **Problem:** write a manual part program to mill a 100 × 40 mm rectangular profile on a 150 mm square plate, with spindle start, tool change and linear interpolation.
- **Step 1: zero:** set x = 0, y = 0 at the lower-left plate corner and z = 0 on the top face.
- **Step 2: corners:** (25, 55), (25, 95), (125, 95), (125, 55).
- **Step 3: tool and path:** Ø10 end mill, clockwise path with G41 D01 (D01 = 5 mm radius).
- **Step 4: entry:** rapid to (15, 45) outside the part, plunge, then feed on to the first corner.

The profile is centred on the plate: x from 25 to 125 and y from 55 to 95. The cutter runs clockwise round the outline with cutter compensation G41 so the programmed points are the part corners.

Example: the complete manual part program for the rectangular profile

O0001 ;	program number
N010 G21 G90 G17 G40 ;	metric, absolute, XY plane, comp off
N020 T01 M06 ;	tool change: Ø10 end mill
N030 S1000 M03 ;	spindle start 1000 rpm, clockwise
N040 G00 X15 Y45 Z5 M08 ;	rapid to start point, coolant on
N050 G01 Z-5 F80 ;	plunge to 5 mm depth
N060 G41 D01 X25 Y55 F150 ;	comp left on, feed to corner 1
N070 Y95 ;	linear cut up the left side
N080 X125 ;	cut along the top
N090 Y55 ;	cut down the right side
N100 X25 ;	cut along the bottom
N110 G40 G00 X15 Y45 ;	comp off, move clear
N120 Z50 M09 ;	retract, coolant off
N130 M05 ;	spindle stop
N140 M30 ;	end of program, rewind

Bold: tool change, spindle start and G01 cuts. Without G41, program the tool centre: (20, 50), (20, 100), (130, 100), (130, 50).

The same profile written in absolute and in incremental mode

Absolute (G90)

```
N060 G90 G41 D01 G01 X25 Y55 ;
N070 Y95 ;
N080 X125 ;
N090 Y55 ;
N100 X25 ;
```

every value = position from zero

Incremental (G91)

```
N060 G90 G41 D01 G01 X25 Y55 ;
N065 G91 ;
N070 Y40 ;
N080 X100 ;
N090 Y-40 ;
N100 X-100 ;
N105 G90 ;
```

every value = move from last point

Check

Both versions end at (25, 55). In incremental form the four moves add up to zero (+40 − 40 in y, +100 − 100 in x), which is a quick check that the profile closes.

- **Absolute: advantages:** easy to read against the drawing; an error affects one point only; easy to restart mid-program.
- **Absolute: disadvantages:** every coordinate must be calculated from the one zero point.
- **Incremental: advantages:** short numbers for repeated features; ideal for subprograms that can run at any location.
- **Incremental: disadvantages:** an error carries into every later position; harder to check.
- **Practice:** most programs use G90 and switch to G91 only for repeated patterns.

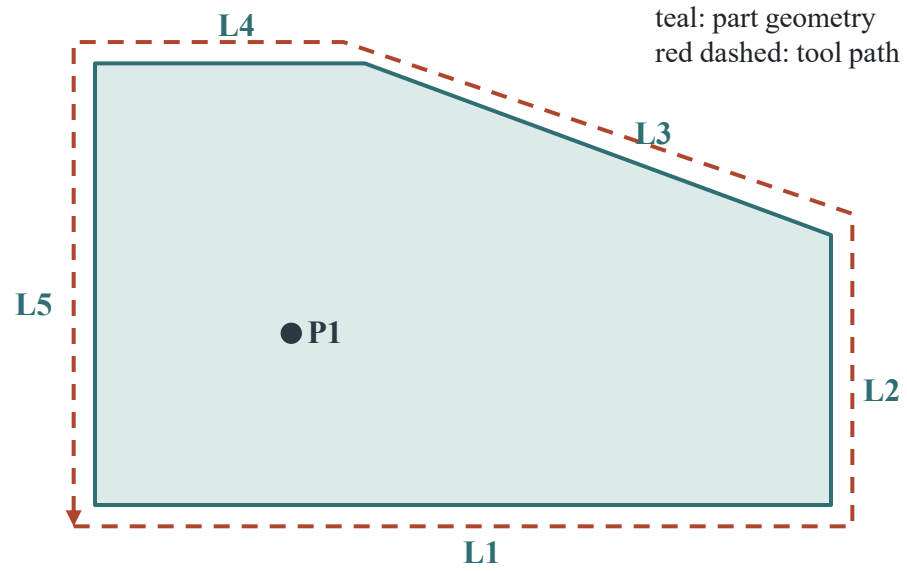
08

Computer-assisted part programming

Sharing the work between the part programmer and the computer

The part programmer defines the part geometry and the tool path

Geometry elements and the cutter path (after Groover Fig. 8.6)

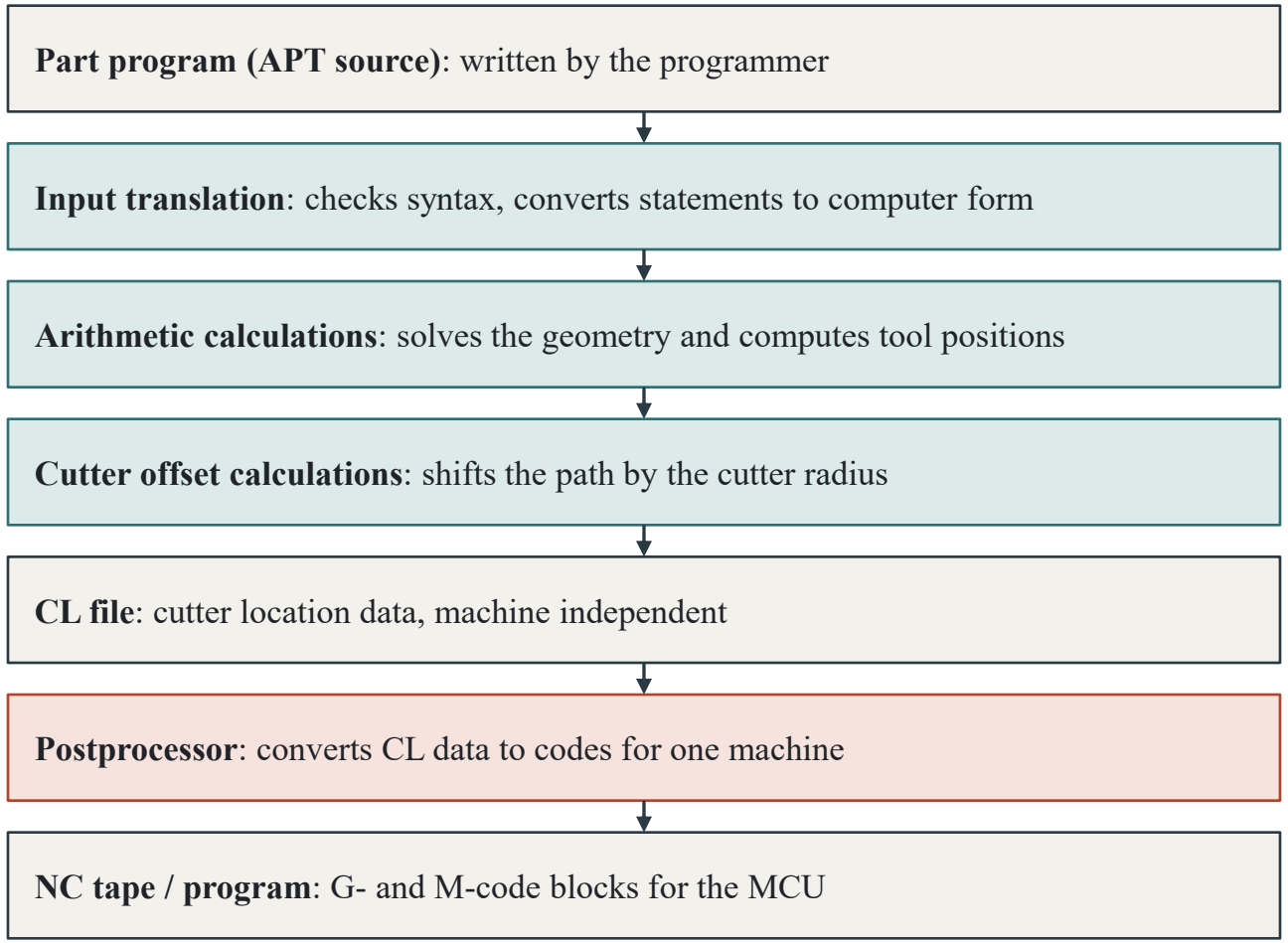


- **Defining the geometry:** every point, line, circle and plane on the part that the tool will touch is named and described (geometry statements).
- **Specifying the tool path:** the programmer writes the sequence of cutter motions along those elements (motion statements).
- **Other instructions:** feeds, speeds, cutter size, tolerances and coolant (postprocessor and auxiliary statements).
- **Result:** an English-like source program, much shorter than the machine code.

Example

The programmer writes L1 = LINE/P0, P2 and so on for every edge, then a motion sequence such as GORGT/L1, PAST, L2. The computer, not the programmer, works out the cutter-centre coordinates.

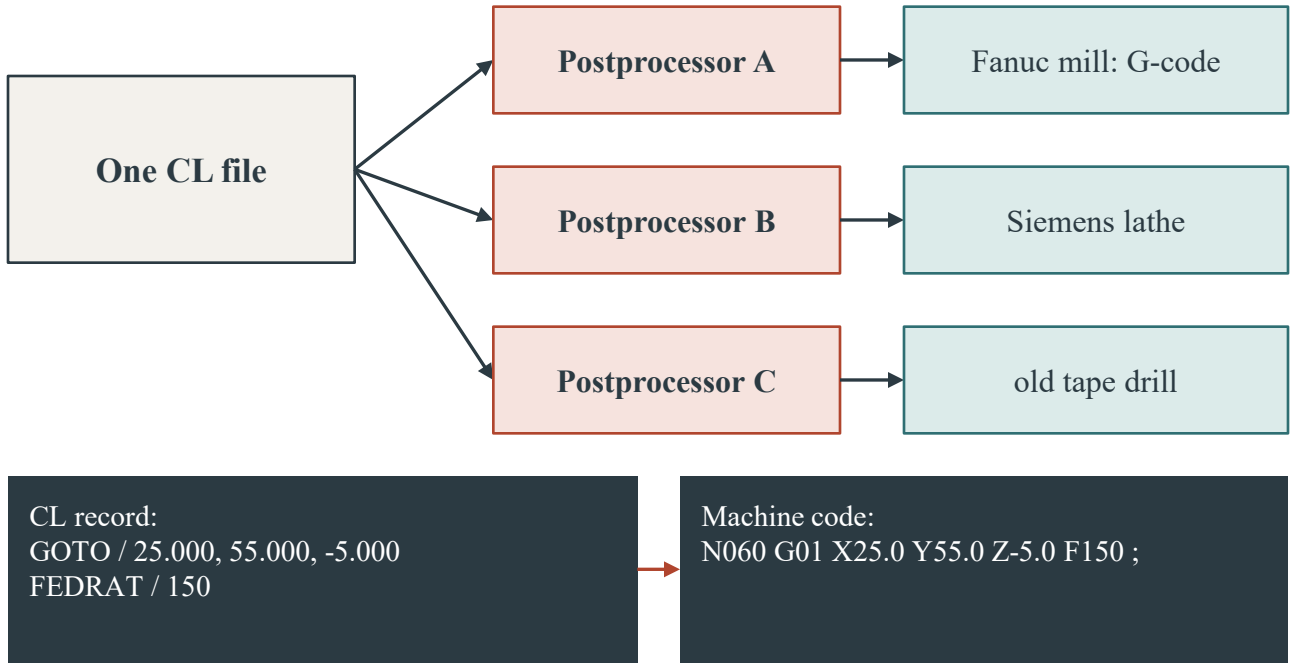
The computer translates the input, does the arithmetic and offsets the cutter



Computer tasks in computer-assisted part programming (after Groover Fig. 8.7)

- **Input translation:** the source program is read, checked for errors and converted into a form the computer can process.
- **Arithmetic:** the computer does the geometry: intersections, tangents, and every point along the path. This is the main saving over manual work.
- **Cutter offset:** the computer offsets the path so the cutter edge, not its centre, follows the part outline.
- **Postprocessing:** the general CL file is turned into the codes and format of the particular machine tool.

The postprocessor converts the general tool path into codes for one machine



In short
The postprocessor is the program that converts the general cutter-location data from APT or CAM software into the exact codes, format and limits of one particular machine tool and its controller.

- **Why needed:** every machine and control has its own codes, word format, axis limits and features; APT output is machine independent.
- **What it does:** converts CL data to G/M codes, adds sequence numbers, applies feed and speed limits, and formats the output.
- **Postprocessor statements:** words such as SPINDL, FEDRAT and COOLNT in the APT program are passed to it.
- **Benefit:** one part program can run on different machines by choosing a different postprocessor.

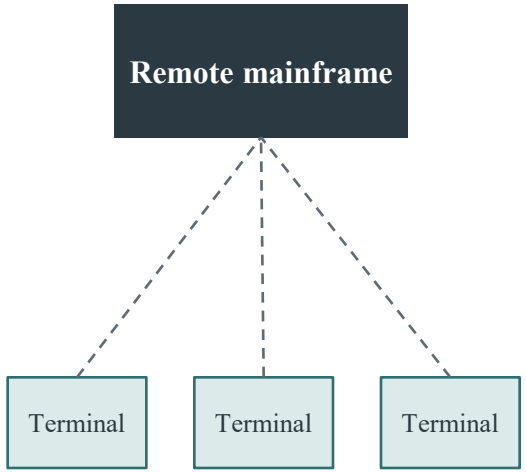
Many NC programming languages have been developed, with APT the most widely used

Language	Developed by	Main use
APT (Automatically Programmed Tools)	MIT, 1956; later IIT Research Institute	General: up to 5-axis contouring; the standard language
ADAPT	IBM, for the U.S. Air Force	Subset of APT for smaller computers; 2-D contouring
EXAPT	Germany (Aachen, Berlin)	PTP and contouring, adds machining technology data
UNIAPT	United Computing Corp.	APT for mini-computers
SPLIT	Sundstrand	PTP and contouring for their machines
COMPACT II	Manufacturing Data Systems	Time-sharing language for PTP and contouring
PROMPT	Weber N/C Systems	Interactive programming for several machine types
CINTURN II	Cincinnati Milacron	Turning (lathe) programs

Source: Groover & Zimmers, Sec. 8.5. APT is covered in detail in the next section.

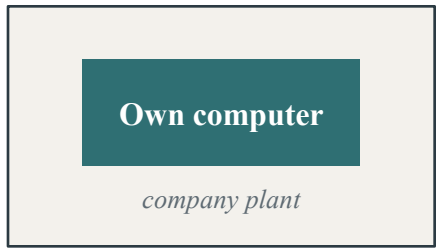
Programs can be processed on a time-shared or an in-house computer

Time-sharing



many companies share one big computer over phone lines

In-house computer

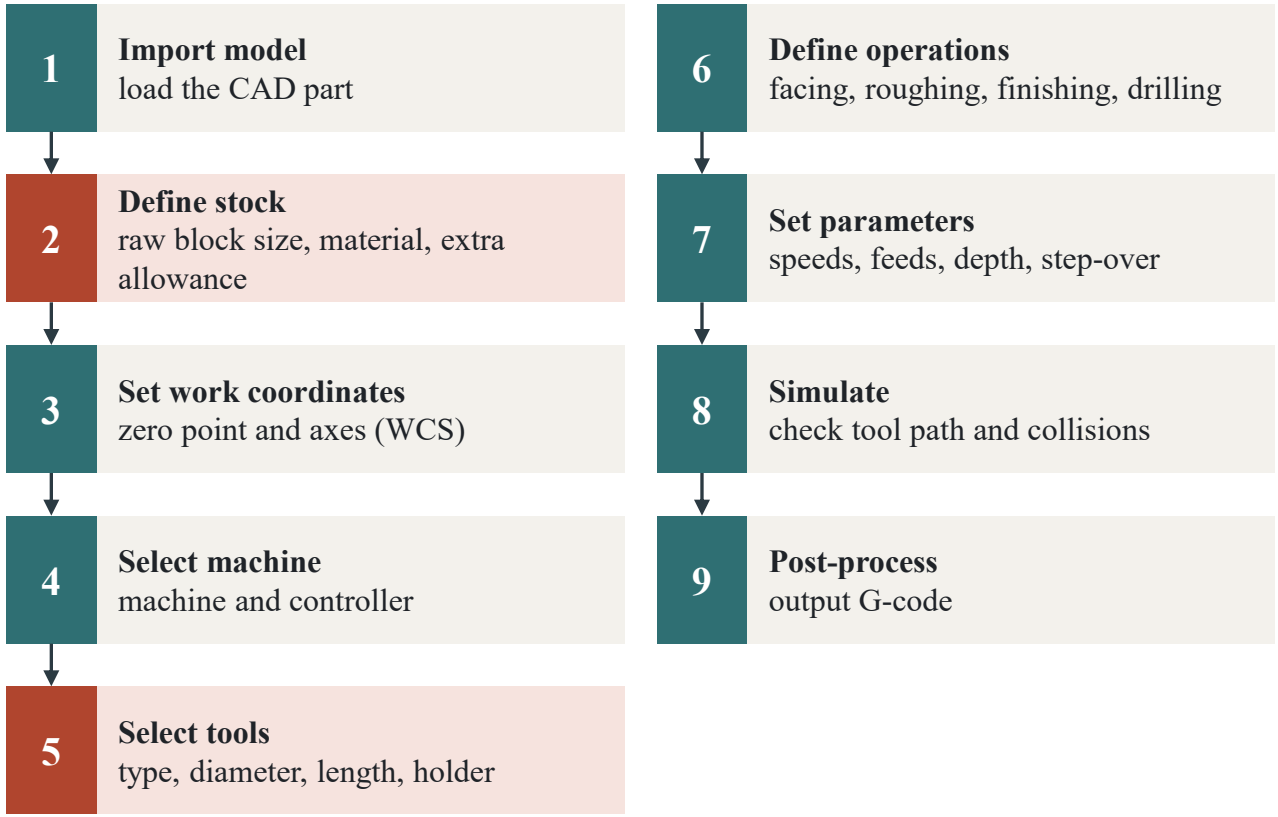


the company runs APT on its own computer or CAD/CAM system

- **Time-sharing:** the company pays for computer time on a service bureau's large computer; low entry cost, wide choice of languages.
- **Disadvantages of time-sharing:** running cost rises with use; data travels outside the company.
- **In-house computer:** the company owns the computer and software; higher first cost but low cost per program and full control.
- **Trend:** in-house CAD/CAM systems have replaced time-sharing for NC programming.

Choice
A small job shop with a few NC machines rents time-sharing. A large plant with many programmers buys its own computer, which today is usually a CAD/CAM workstation.

Machining set-up in CAM software runs from stock definition to tool selection



Steps 2 (stock) and 5 (tool selection) are highlighted

- **Stock:** the raw material the part is cut from: a block, bar or casting, larger than the part. The software removes stock until the part shape remains.
- **Work coordinate system:** the zero point set on the stock so the CAM program and the machine agree.
- **Tool selection:** each operation is given a tool from the library; diameter and length decide the cutter offset.
- **Operations and parameters:** rough then finish, with speed, feed, depth of cut and step-over.
- **Simulation and post:** verify, then post-process to G-code for the machine.

09

The APT language

Geometry, motion, postprocessor and auxiliary statements, with complete programs

APT uses four types of statement, each in a fixed format

Geometry statements: define the part's points, lines, circles, planes
P1 = POINT/1.0, 2.0, 0

Motion statements: describe the path the cutter follows
GOTO/P1

Postprocessor statements: control the machine: speed, feed, coolant
SPINDL/573 FEDRAT/2.29

Auxiliary statements: cutter size, tolerances, part name, end
CUTTER/.500 FINI

Format of a geometry statement:
symbol = geometry type / descriptive data
e.g. P1 = POINT/5.0, 4.0, 0.0

- **APT:** Automatically Programmed Tools: an English-like language for computer-assisted part programming, developed at MIT in the 1950s.
- **Geometry:** the programmer names each element (P1, L1, C1) once and refers to it by name later.
- **Motion:** written as command / descriptive data, e.g. GOTO/P1.
- **Slash:** the slash separates the main word from its data; words without a slash need no data.
- **Punctuation:** commas separate data items; the symbol must start with a letter and be up to six characters.

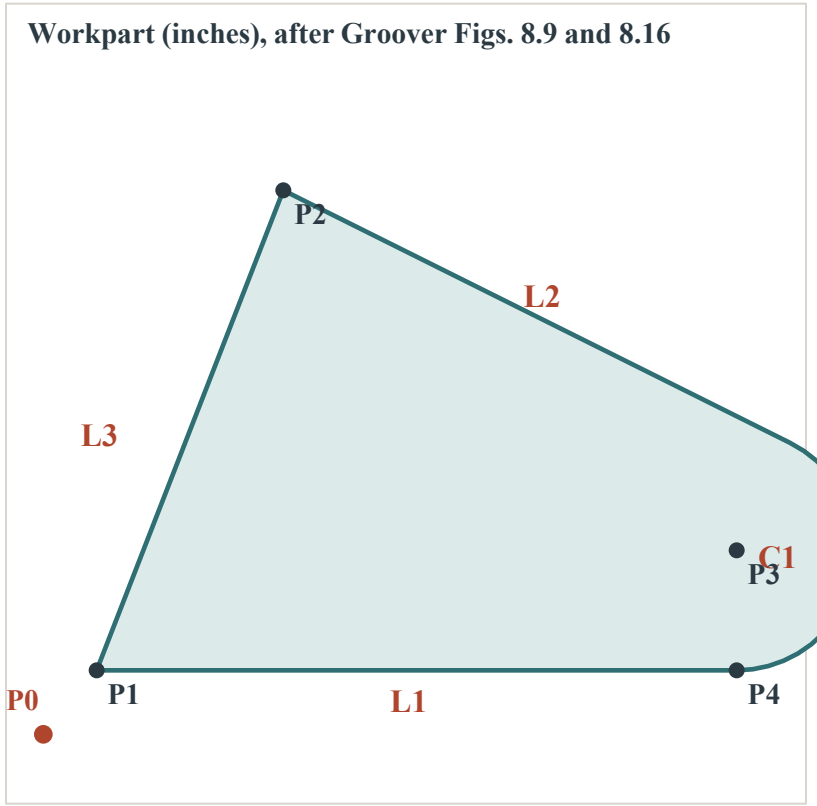
Geometry statements define the points, lines, circles and planes of the part

Element	APT statement	Meaning
Point	P1 = POINT/5.0, 4.0, 0.0	by x, y, z coordinates
Point	P2 = POINT/INTOF, L1, L2	intersection of two lines
Line	L1 = LINE/P3, P4	through two points
Line	L2 = LINE/P5, PARLEL, L3	through a point, parallel to a line
Line	L4 = LINE/P1, LEFT, TANTO, C1	through a point, tangent to a circle
Plane	PL1 = PLANE/P1, P4, P5	through three points
Plane	PL2 = PLANE/0, 0, 1.0, -0.5	coefficients a, b, c, d of ax + by + cz = d
Circle	C1 = CIRCLE/CENTER, P1, RADIUS, 5.0	by centre and radius
Circle	C2 = CIRCLE/3.0, 3.0, 0, 3.0	by centre x, y, z and radius

Source: Groover & Zimmers, Sec. 8.6, geometry statements.

- **Ground rule 1:** coordinate data are given in the order x, y, z.
- **Ground rule 2:** an element must be defined before it is used in another statement.
- **Ground rule 3:** a symbol can be used for only one element in a program.
- **Ground rule 4:** the words LEFT, RIGHT, LARGE, SMALL, XLARGE, etc. pick one of several possible solutions.
- **Ground rule 5:** lines and planes are unbounded: they extend to infinity in both directions.

Example: geometry statements for a workpart




```
P0 = POINT/-0.5, -0.6, 0
P1 = POINT/0, 0, 0
P2 = POINT/1.75, 4.5, 0
P3 = POINT/6.0, 1.125, 0
P4 = POINT/6.0, 0, 0
L1 = LINE/P1, P4
C1 = CIRCLE/CENTER, P3,
RADIUS, 1.125
L2 = LINE/P2, LEFT,
TANTO, C1
L3 = LINE/P1, P2
PL1 = PLANE/0, 0, 1.0, -1.0
```

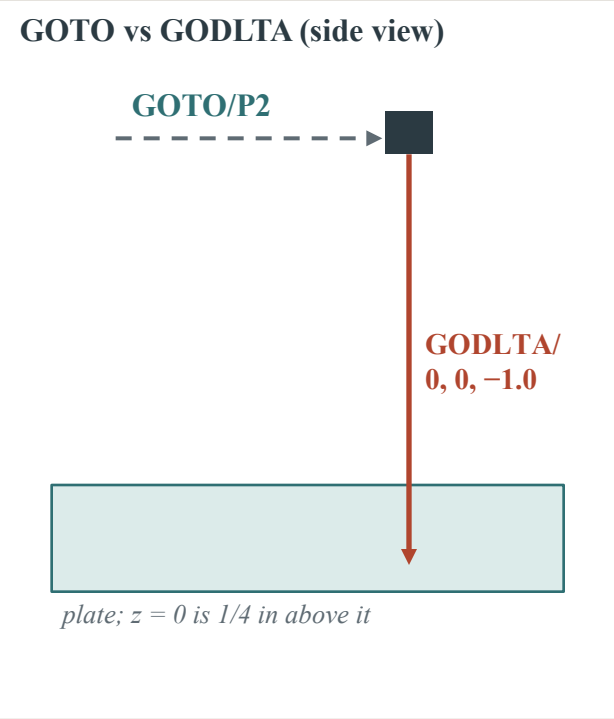
- **L1:** the bottom edge, through P1 and P4.
- **C1:** the rounded corner: centre P3, radius 1.125 in; it touches L1 at P4.
- **L2:** the top edge: from P2, tangent to C1 on the LEFT side as seen from P2.
- **L3:** the left edge, through P1 and P2.
- **PL1 and P0:** the part surface for the cutter bottom, and the target (start) point.
- **Note:** coordinates are taken from the textbook figures; the APT form follows Groover Ex. 8.2.

PL1 is the plane z = −1.0, set below the bottom of the plate so the cutter machines the full thickness (the part surface need not be a real face of the part).

Point-to-point motion uses FROM, GOTO and GODLTA

Example 8.3 (drilling)

```
P1 = POINT/1.0, 2.0, 0
P2 = POINT/1.0, 1.0, 0
P3 = POINT/3.5, 1.5, 0
P0 = POINT/-1.0, 3.0, 2.0
FROM/P0
GOTO/P1
GODLTA/0, 0, -1.0
GODLTA/0, 0, +1.0
GOTO/P2
GODLTA/0, 0, -1.0
GODLTA/0, 0, +1.0
GOTO/P3
GODLTA/0, 0, -1.0
GODLTA/0, 0, +1.0
GOTO/P0
```



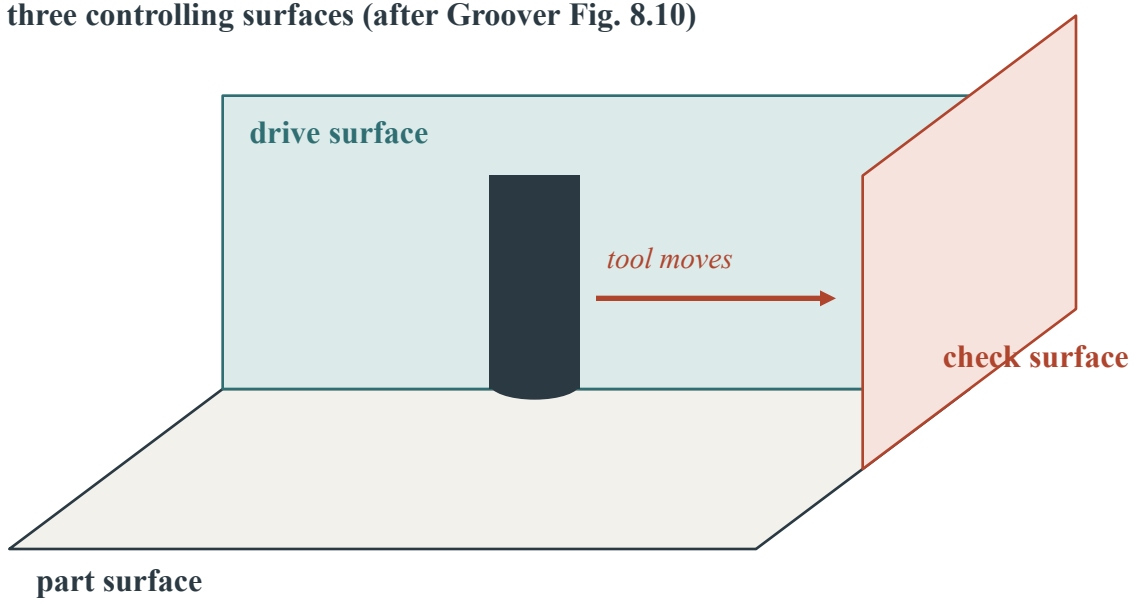
- **FROM:** FROM/P0 (or FROM/x, y, z) gives the starting point where the operator has set the tool; it comes only once, at the start of motion.
- **GOTO:** GOTO/P2 or GOTO/2.0, 7.0, 0.0 sends the tool to that point (absolute move).
- **GODLTA:** GODLTA/2.0, 7.0, 0.0 moves 2 in in x and 7 in in y from the present position (incremental move).
- **Drilling:** GOTO to the hole, GODLTA down by the depth, GODLTA back up.

In short

GOTO moves to an absolute position (a named point or x, y, z). GODLTA moves by an increment from where the tool is, which makes drilling down and back up easy.

Contouring guides the cutter with drive, part and check surfaces

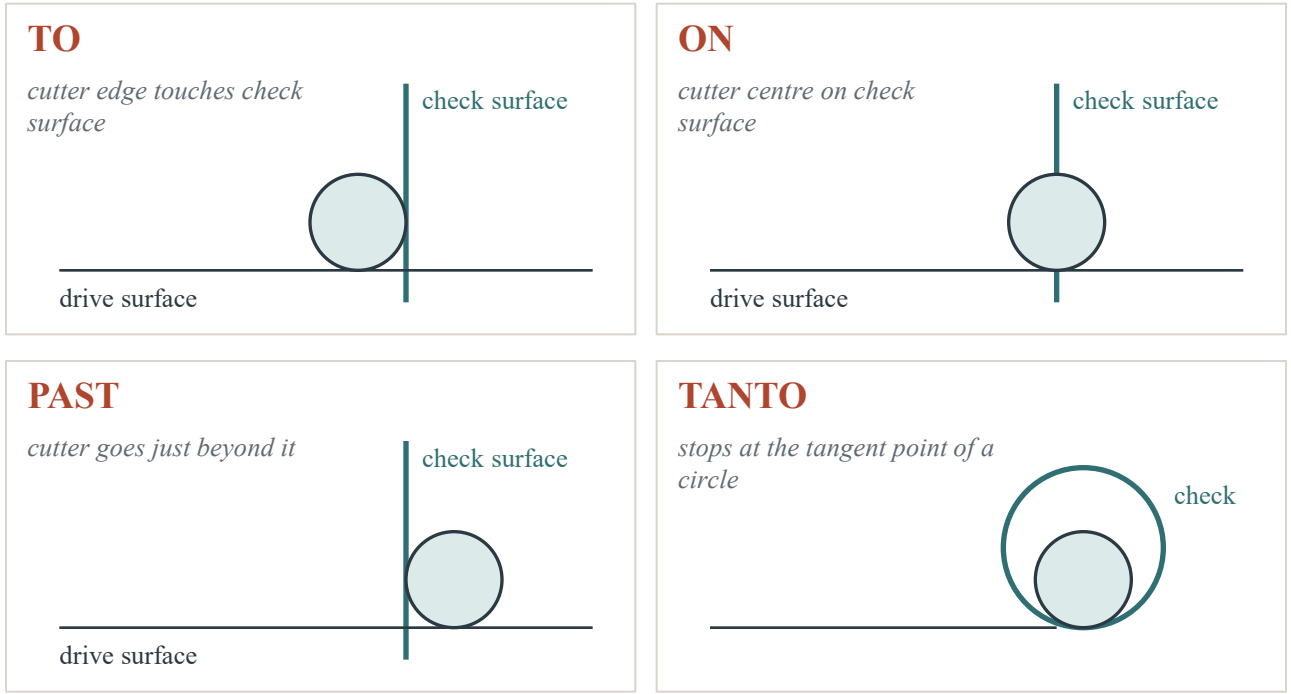
The three controlling surfaces (after Groover Fig. 8.10)



- **Why three surfaces:** in contouring the tool position must be controlled all the time, so APT guides it between two intersecting surfaces and stops it at a third.
- **Drive surface:** guides the side of the cutter along the path.
- **Part surface:** the surface on which the bottom (end) of the cutter rides; it may not be an actual surface of the part.
- **Check surface:** the surface that stops the current motion; it often becomes the next drive surface.

Example
To mill the bottom edge of a plate: the drive surface is the edge line L1 (side of the cutter), the part surface is a plane below the plate (bottom of the cutter), and the check surface is the next edge, where the move ends.

Modifier words TO, ON, PAST and TANTO say where the cutter stops

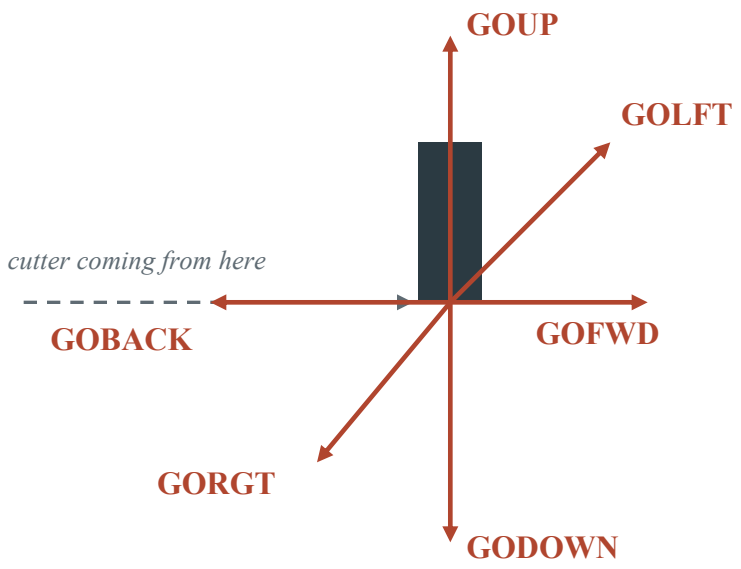


- **Modifier word:** tells APT where to stop the cutter relative to the check surface.
- **TO:** the cutter's edge just reaches the check surface.
- **ON:** the cutter's centre is on the check surface.
- **PAST:** the cutter moves just past the check surface; used at sharp outside corners.
- **TANTO:** the cutter stops at the point of tangency between the drive surface and a circular check surface.

Example: GORGT/L1, TANTO, C1 moves the cutter along line L1 until it reaches the point where L1 is tangent to circle C1 (Groover Fig. 8.12).

Six motion words steer the cutter; GO/TO starts a contour, GOTO moves to a point

The six motion words (after Groover Fig. 8.13)



GO/TO vs GOTO

GOTO/P1: point-to-point move to a point.

GO/TO, L1, TO, PL1, TO, L3: the first contouring move; brings the cutter to the drive, part and check surfaces at the start of a contour.

- **GOFWD:** go forward, continuing in the same direction.
- **GOBACK:** reverse the direction of motion.
- **GOLFT / GORGT:** turn left or right, seen as if riding on the cutter in its direction of travel.
- **GOUP / GODOWN:** move up or down.
- **Viewpoint:** the workpiece is stationary and the tool is imagined to move relative to it.

Start of a contour

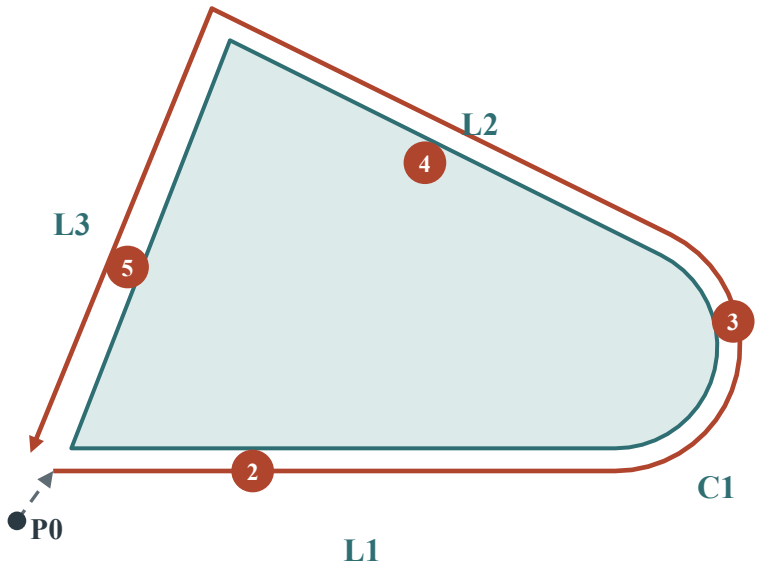
FROM/TARG

GO/TO, PL1, TO, PL2, TO, PL3

The tool moves from the target point to the intersection of the three surfaces; after that, GOLFT, GORGT, GOFWD and the others continue the contour.

Example: a contouring motion sequence round the workpart

Cutter path round the part (red)




```
FROM/P0
1 GO/TO, L1, TO, PL1,
TO, L3
2 GORGT/L1, TANTO, C1
3 GOFWD/C1, PAST, L2
4 GOFWD/L2, PAST, L3
5 GOLFT/L3, PAST, L1
GOTO/P0
```

- **1:** start: cutter to L1 (drive), PL1 (part) and L3 (check), at the lower-left corner.
- **2:** turn right and drive along L1 until tangent to C1.
- **3:** go forward round the arc C1 until past L2.
- **4:** go forward along L2 until past L3.
- **5:** turn left down L3 until past L1, back to the start; then return to P0 (Groover Ex. 8.4).

PL1 is placed below the bottom of the plate so the cutter cuts the full thickness. Cutter size, feed and speed are added by the postprocessor and auxiliary statements (next slide).

Postprocessor and auxiliary statements complete the program

Postprocessor statements

SPINDL/573, CLW	spindle on at 573 rpm, clockwise; SPINDL/OFF stops it
FEDRAT/2.29	feed rate in in/min (or mm/min)
COOLNT/ON	coolant on; COOLNT/OFF
RAPID	next move at rapid traverse
TURRET/2	select tool 2
MACHIN/MILL, 1	choose the postprocessor for machine 1
END	end of machine operation

Auxiliary statements

PARTNO …	part identification, printed on the output
CLPRNT	print the cutter-location data
CUTTER/.500	cutter diameter: path is offset by 0.250 in
INTOL/.001 · OUTTOL/.001	inside and outside tolerances for curves
FINI	last statement of the program

- **Postprocessor statements:** control the machine tool: spindle, feed, coolant and other features; they are passed to the postprocessor.
- **With or without slash:** words with a slash (FEDRAT/) need data after it; words without (RAPID, END) are complete.
- **FEDRAT:** in drilling the feed is along the drill axis; in end milling it is perpendicular to the cutter axis.
- **Auxiliary statements:** define the cutter, tolerances and part name, and end the program.
- **CUTTER:** CUTTER/.500 tells APT the diameter, so the tool path is offset 0.250 in from the part outline.

Example: a complete APT program

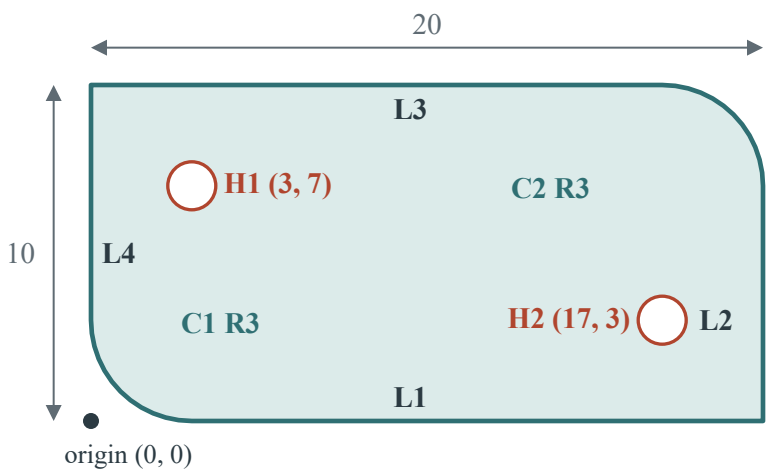
```
PARTNO EXAMPLE 8.5 PLATE
MACHIN/MILL, 1
CLPRNT
CUTTER/0.500
OUTTOL/0.005
INTOL/0.005
P0 = POINT/-0.5, -0.6, 0
P1 = POINT/0, 0, 0
P2 = POINT/1.75, 4.5, 0
P3 = POINT/6.0, 1.125, 0
P4 = POINT/6.0, 0, 0
L1 = LINE/P1, P4
C1 = CIRCLE/CENTER, P3, RADIUS,
1.125
```

```
L2 = LINE/P2, LEFT, TANTO, C1
L3 = LINE/P1, P2
PL1 = PLANE/0, 0, 1.0, -1.0
SPINDL/573, CLW
FEDRAT/2.29
COOLNT/ON
FROM/P0
GO/TO, L1, TO, PL1, TO, L3
GORGT/L1, TANTO, C1
GOFWD/C1, PAST, L2
GOFWD/L2, PAST, L3
GOLFT/L3, PAST, L1
RAPID
GOTO/P0
COOLNT/OFF
SPINDL/OFF
END
FINI
```

- **Job:** contour-mill the plate of the previous examples: low-carbon steel, already rough-cut to shape.
- **Tool and conditions:** 1/2 in end mill, 573 rpm, 2.29 in/min (Groover Ex. 8.5).
- **Order:** identification and set-up, then geometry, then machine settings, then motion, then shut-down and FINI.
- **Cutter offset:** CUTTER/0.500 makes APT offset the path 0.250 in outside the outline.
- **Note:** statement order follows Groover; the program listing is reconstructed in textbook style.

APT program: contour milling and drilling a 20 × 10 mm plate

Plate 20 × 10 mm (all dimensions in mm)



Assumptions

Top face z = 0, so PL1 (z = −5) gives the 5 mm depth. Ø6 end mill for contouring, Ø3 drill for holes. Feed 0.15 mm/rev × 1000 rpm = 150 mm/min.

```
PARTNO PLATE 20X10
MACHIN/MILL, 1
CLPRNT
UNITS/MM
CUTTER/6.0
OUTTOL/0.01
INTOL/0.01
P0 = POINT/-10, -10, 10
P1 = POINT/0, 0, 0
P2 = POINT/20, 0, 0
P3 = POINT/20, 10, 0
P4 = POINT/0, 10, 0
L1 = LINE/P1, P2
L2 = LINE/P2, P3
L3 = LINE/P3, P4
L4 = LINE/P4, P1
C1 = CIRCLE/3, 3, 0, 3
C2 = CIRCLE/17, 7, 0, 3
H1 = POINT/3, 7, 2
H2 = POINT/17, 3, 2
PL1 = PLANE/0, 0, 1, -5
```

- **Problem:** write an APT program to contour-mill the plate 5 mm deep and drill the two Ø3 mm holes; feed 0.15, speed 1000 rpm.
- **Lines:** L1 bottom, L2 right, L3 top and L4 left edges, each through two corner points.
- **Circles:** C1 and C2 are the R3 corners, centres (3, 3) and (17, 7); tangent to their two edges.
- **Holes:** H1 and H2 are 2 mm above the top face, the start point for drilling.
- **Part surface:** PL1 is the plane z = −5, the bottom of the cut.

APT program for the plate: contouring and drilling statements

Contour milling (Ø6 end mill)

```
SPINDL/1000, CLW
FEDRAT/150
COOLNT/ON
FROM/P0
GO/TO, L1, TO, PL1, TANTO, C1
GORGT/L1, PAST, L2
GOLFT/L2, TANTO, C2
GOFWD/C2, TANTO, L3
GOFWD/L3, PAST, L4
GOLFT/L4, TANTO, C1
GOFWD/C1, TANTO, L1
GODLTA/0, 0, 15
RAPID
GOTO/P0
```

Drilling the holes (Ø3 drill)

```
CUTTER/3.0
TURRET/2
SPINDL/1000, CLW
FEDRAT/150
RAPID
GOTO/H1
GODLTA/0, 0, -7
GODLTA/0, 0, 7
RAPID
GOTO/H2
GODLTA/0, 0, -7
GODLTA/0, 0, 7
RAPID
GOTO/P0
COOLNT/OFF
SPINDL/OFF
END
FINI
```

- **Start:** from P0 the cutter goes to L1 on PL1, at the tangent point with C1 (lower-left corner).
- **Contour:** right along L1 past the sharp corner, up L2 to C2, round C2 to L3, along the top past L4, down L4, round C1 back to L1 (counter-clockwise, tool outside).
- **Retract:** GODLTA/0, 0, 15 lifts the cutter clear before the rapid return.
- **Drilling:** tool change, GOTO each hole 2 mm above the top, GODLTA −7 (2 mm clearance + 5 mm depth) and back up.
- **End:** coolant and spindle off, END, FINI.

10

NC programming with interactive graphics

Generating and checking the tool path on a CAD/CAM graphics terminal

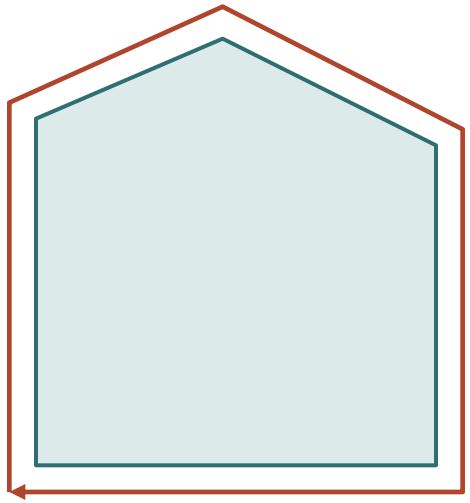
CAD/CAM part programming starts from the design geometry and builds the tool path on screen

1. **Geometry definition**: use the part model created during design, or build it on the terminal; label surfaces and elements (APT geometry is generated automatically)
2. **Tool selection**: pick tools from the tool library, or define a new one (diameter, corner radius, length)
3. **Tool path generation**: step by step: define a start point and move the cutter along the displayed surfaces; or use automatic routines
4. **Postprocessor statements**: feeds, speeds, coolant are inserted at the right points
5. **Output**: an APT program or CLFILE, postprocessed to the NC program

- **Innovations:** interactive graphics, voice programming and manual data input (MDI) changed NC programming from the late 1970s.
- **Integration:** the same geometric data that defined the part in design are used to construct the tool path.
- **Labelling:** the programmer labels surfaces in reply to prompts; the system generates the APT geometry statements.
- **Example:** in Groover Ex. 8.8 an Applicon system labels the part edges L182–L185 and the arc C158, then lists the APT program.

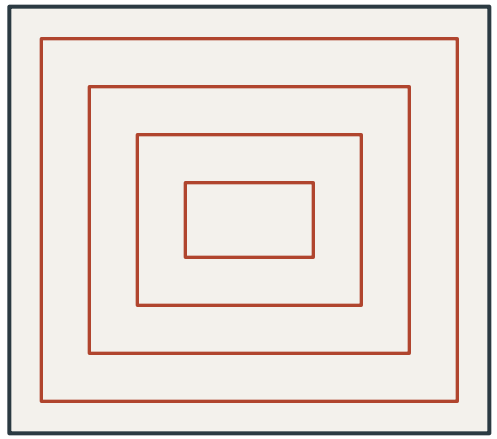
Automatic routines generate profiling and pocketing tool paths

Profiling



cutter follows the outline at an offset

Pocketing

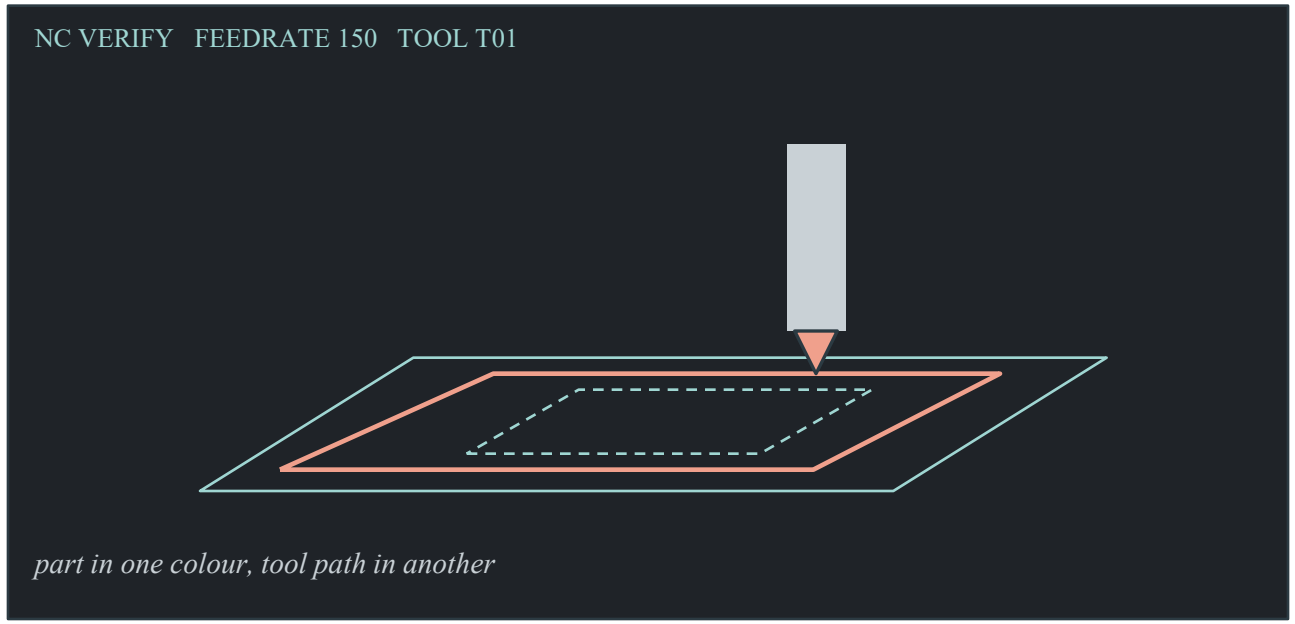


material removed inside the boundary

- **Automatic routines:** MACRO-type subroutines built into the CAD/CAM software for common jobs; the programmer only enters the parameters.
- **Profiling:** generates the cutter path round a chain of geometry elements picked by the user.
- **Pocketing:** removes material inside a closed boundary; the programmer gives the sides and bottom of the pocket.
- **Others:** end-milling a pocket, point-to-point drilling, punch press work and surface contouring.

Example (Groover Fig. 8.20)
On a Computervision system the programmer picks the boundary and bottom of each pocket; the routine generates the nested passes that clear it, then a profile pass finishes the outside.

Dynamic simulation lets the programmer verify the tool path before cutting



part in one colour, tool path in another

- **Colour graphics:** the part is shown in one colour and the tool path in another, so they are easy to tell apart.
- **Dynamic verification:** the tool and holder move over the workpiece on screen in 3D (Computervision, Groover Fig. 8.21).
- **Four display modes:** high speed, actual speed, freeze and stepping.
- **Benefit:** collisions, gouges and missed areas are found before any metal is cut.

1. High speed: fast run to check the whole path

2. Actual speed: tool moves at the commanded feed

3. Freeze: stops motion for close inspection

4. Stepping: shows the path in discrete steps

CAD/CAM gives five advantages in NC part programming

1. **Savings in geometry definition**: the part geometry already exists from design, so it is not redefined as in conventional APT

2. **Immediate visual verification**: the tool path is shown at once, so most errors are found and corrected as they are made

3. **Automatic programming routines**: MACRO routines for profiling and pocketing cut programming time sharply

4. **One-of-a-kind jobs**: programming is so quick that NC becomes economical even for a single part

5. **Integration with other functions**: design, tool design, process planning, set-up instructions and part families share one database

Source: Groover & Zimmers, Sec. 8.8, advantages of CAD/CAM in NC programming.

11

BPUT previous year questions

Module III question bank, mark-wise

2-mark questions

1. Define a Canned Cycle in CNC/NC programming and give one example.
2. What is the primary role of a Post-processor in a CAM/NC workflow?
3. Define the term 'Stock' in CAM.
4. What is the purpose of a Coolant (M08) code in CNC?
5. Explain the concept of Tool Offset in CNC machining.
6. What do you mean by G-codes? Explain with a suitable example.
7. Distinguish between GOTO and GO/TO statements in relation to NC programming.
8. What is manual part programming?
9. What is the function of an MCU (Machine Control Unit) in NC?
10. State the specific functions of the miscellaneous codes M04 and M30.

6-mark questions

1 Describe the steps involved in machining setup within a CAM software, from stock definition to tool selection.

2 Explain Subprograms and Macros in CNC programming. When are they used?

3 Explain the difference between Roughing and Finishing in CNC machining.

4 Explain the following in relation to NC programming: (i) NC words (ii) Drive Surface, Check Surface, and Part Surface.

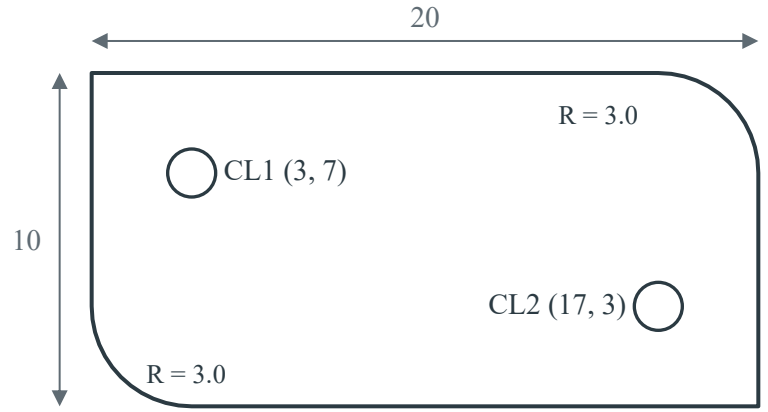
5 Differentiate between absolute and incremental coordinate systems used for manual part programming. Explain their advantages and disadvantages.

16-mark questions

1 Write a manual part program for a milling operation to cut a rectangular profile of size 100 mm × 40 mm on a square plate of 150 mm size. Include spindle start, tool change, and linear interpolation codes.

---

2 Develop an APT program for the figure. Do contour machining for a depth of 5 mm and drilling of holes of diameter 3 mm. Assume Feed 0.15 mm/min, speed 1000 rpm. All dimensions are in mm.



MODULE IV

NC Technology: CNC, DNC and Adaptive Control

MFPC3005 Computer Aided Design and Manufacturing (CAD/CAM)

Reference: Groover & Zimmers, CAD/CAM, Ch. 9

Module IV roadmap

01	**Problems with conventional NC**	Six built-in problems and how NC controllers evolved
02	**Computer numerical control (CNC)**	Configuration, functions, compensation, diagnostics, advantages
03	**Direct numerical control (DNC)**	Central computer, two configurations, functions, advantages
04	**Combined DNC/CNC systems**	Hierarchical, distributed NC and what the combination adds
05	**Adaptive control machining**	Where AC pays, sources of variability, ACO vs ACC, benefits
06	**BPUT previous year questions**	Mark-wise question bank

By the end of Module IV you should be able to

- List the problems of conventional NC and trace the generations of NC controllers
- Explain the configuration of a CNC system and how its controller, drives, motors and encoders work together
- Describe the functions of CNC: control, in-process compensation, programming features and diagnostics
- Compare hybrid and straight CNC, and state the advantages of CNC over conventional NC
- Draw the DNC configuration and compare behind-the-tape-reader and special-MCU systems
- State the functions and advantages of DNC, and distinguish NC, CNC and DNC
- Explain how combined DNC/CNC systems form a hierarchical, distributed NC
- Define adaptive control, identify where it applies and list the sources of variability
- Distinguish ACO from ACC, explain the working of an ACC system and its benefits

01

Problems with conventional NC

Why hard-wired, tape-fed NC fell short, and how its controllers evolved

Conventional NC suffers from six built-in problems

01

Part programming mistakes

Syntax and arithmetic errors in a manual program; a tape often needs several corrections before it runs correctly.

02

Nonoptimal speeds and feeds

Feed and speed are fixed in the program, so the controller cannot respond to the actual cutting conditions.

03

Punched tape

Paper tape tears and wears with repeated use; Mylar tape lasts longer but costs more.

04

Tape reader

The least reliable component of the NC system; a reader fault stops the machine.

05

Hard-wired controller

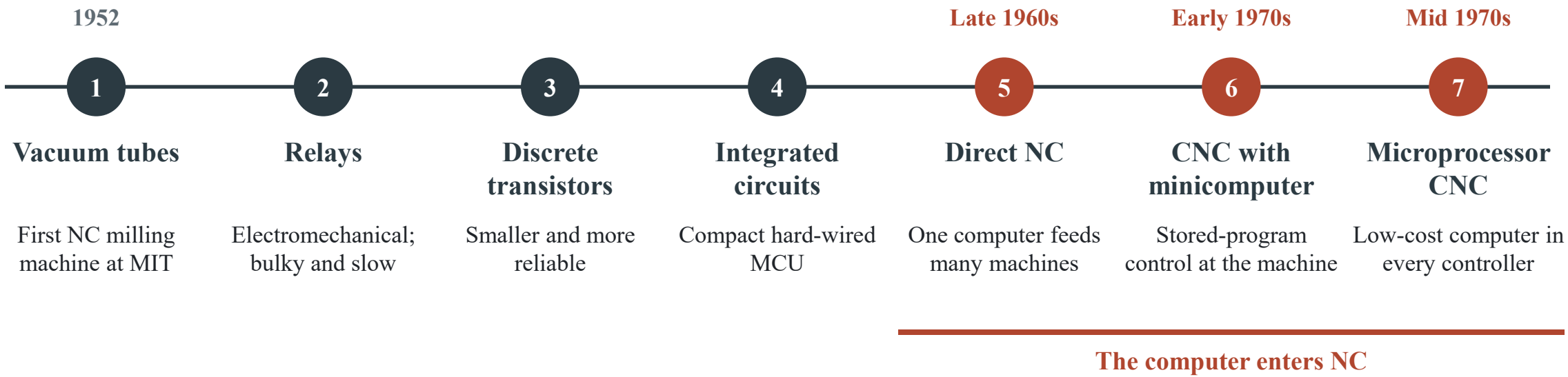
Logic is built into circuits, so a new feature needs new hardware, not a software change.

06

Management information

No timely data on parts produced, cycle times, tool wear or machine breakdowns.

NC controllers passed through seven hardware generations from 1952 to the mid-1970s



Why the generations matter

Generations 1 to 4: the controller was hard-wired; each step only made the circuits smaller and more reliable.

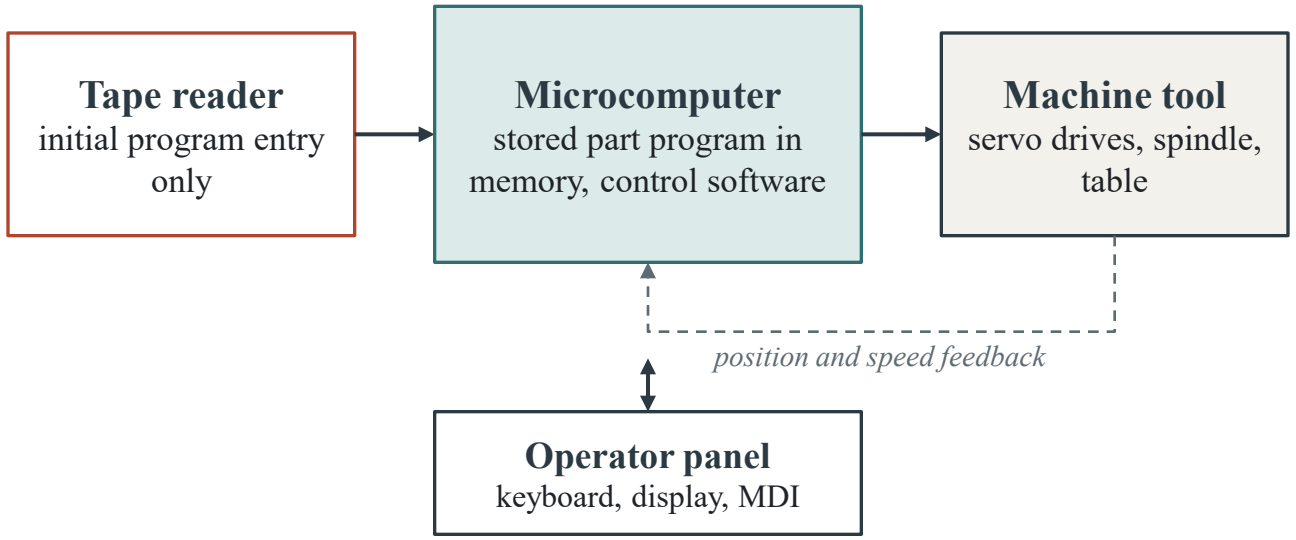
Generations 5 to 7: a computer became part of NC, which led to DNC, CNC and adaptive control, the subject of this module.

02

Computer numerical control (CNC)

A dedicated computer on each machine: configuration, functions and advantages

CNC uses a dedicated stored-program computer at each machine tool



Configuration of a CNC machine tool system

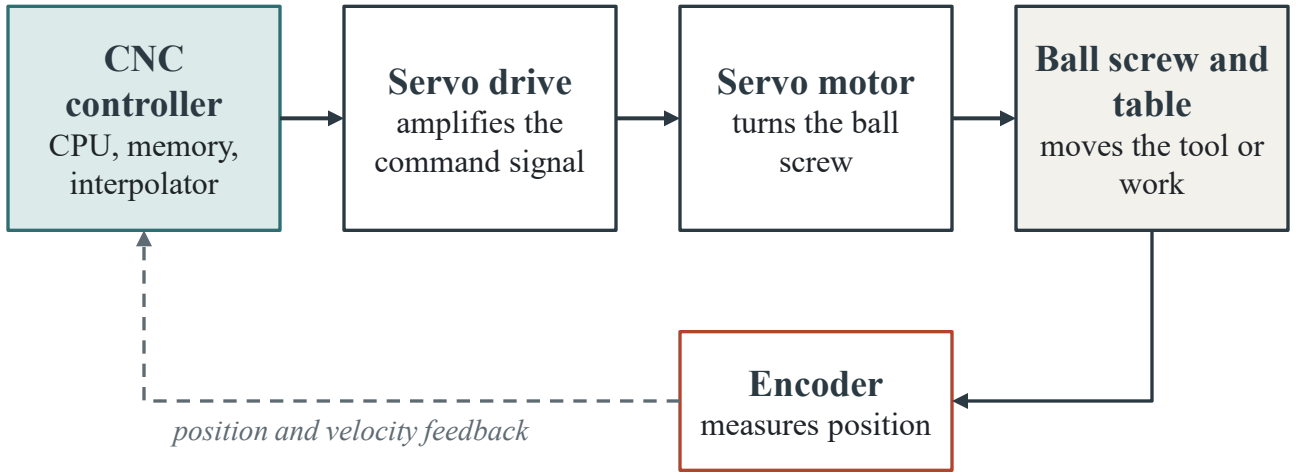
Example: changing a feed on the shop floor

NC: the programmer edits the source, punches a new tape and reloads it.

CNC: the operator edits the stored program at the control panel and runs it at once.

- **Definition:** an NC system that uses a dedicated stored-program computer to perform some or all of the basic NC functions.
- **Tape read once:** the program is entered once and held in memory, so the unreliable tape reader is not used during machining.
- **Soft-wired control:** control logic is software, so features change by reprogramming rather than rewiring.
- **Typical computer:** a minicomputer in early systems; a microprocessor in modern controllers.

A CNC system moves the tool through a closed loop of controller, drive, motor and encoder



Closed-loop control of one machine axis in a CNC system

Example: moving the table 50 mm along X

1. The controller interpolates the move and sends a velocity command.
2. The drive amplifies it and the motor turns the ball screw.
3. The encoder reports position; the controller corrects any error until the table reaches 50 mm.

- **Controller:** reads the stored program, interpolates the tool path and issues position and velocity commands for each axis.
- **Drives:** power amplifiers that convert the low-power command into motor current.
- **Motors:** servo or stepper motors that turn the ball screws to move the axes and spindle.
- **Encoders:** rotary or linear sensors that feed actual position back, closing the control loop.

CNC performs four principal functions

01
Machine tool control

Converts the stored program into axis motion and spindle commands. Interpolation is done in hard-wired logic (hybrid CNC) or in software (straight CNC).

02
In-process compensation

Corrects the tool motion during machining for errors measured by probes and gauges, changes in datum, tool offsets and variable cutting conditions.

03
Programming and operating features

Soft-wired control adds program editing at the machine, tool-path graphics, extra interpolation types, unit conversion, macros and manual data input.

04
Diagnostics

Identifies the cause of a malfunction, warns that a component is about to fail and switches to redundant circuits to cut downtime.

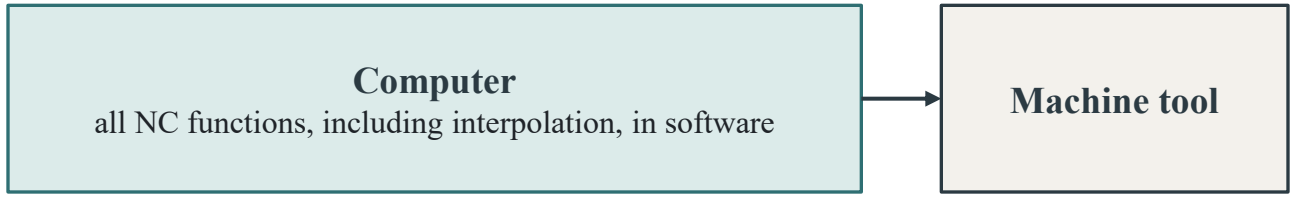
Hybrid CNC keeps interpolation in hard-wired logic; straight CNC does everything in software

Hybrid CNC



Hard-wired logic does the fast, repetitive arithmetic; the computer does the rest

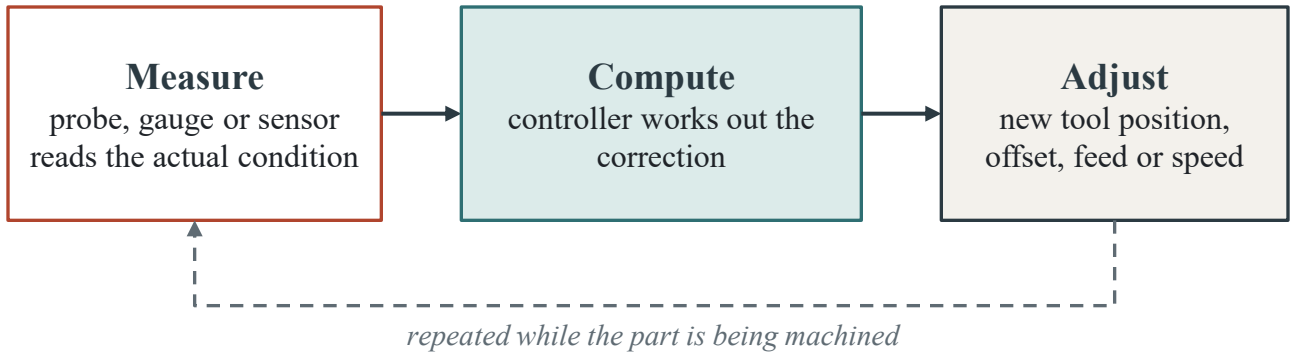
Straight CNC



Software does everything; only the servo interface is hardware

Basis	Hybrid CNC	Straight CNC
Interpolation	Hard-wired circuits	Computer software
Cost	Lower: cheap circuits, smaller computer	Higher: needs a faster computer
Flexibility	Limited to the wired functions	Full: any function can be reprogrammed
Speed	Fast interpolation in hardware	Depends on computer speed

In-process compensation corrects the tool motion while the part is being machined



In-process compensation loop

Example: a worn end mill

After 40 parts the cutter has worn by 0.02 mm.

An in-process probe measures the pocket width as undersize.

The controller adds 0.02 mm to the tool offset, and the next part is machined to size without stopping the machine.

- **Probe and gauge errors:** adjusts for dimensional errors sensed by in-process inspection probes and gauges.
- **Datum changes:** recomputes axis positions when the datum location changes.
- **Tool offsets:** adjusts for changes in tool length and radius.
- **Adaptive control:** changes speed and feed to suit the actual cutting conditions.
- **Tool life:** predicts tool life and selects an alternative tool when one wears out.

Soft-wired control gives CNC programming and operating features that NC lacks

- **Editing at the machine**
  Correct or optimize the stored program at the control panel, without a new tape.
- **Tool-path graphics**
  Display the programmed path on screen and check it before cutting metal.
- **More interpolation types**
  Circular, parabolic and cubic interpolation, not only linear.
- **Metric and inch units**
  Convert the program between unit systems in software.
- **Macros and subroutines**
  Store a repeated machining sequence once and call it by name.
- **Several programs in memory**
  Switch between jobs without reloading a program.
- **Manual data input**
  Enter or change blocks directly from the keyboard.
- **Operator prompts**
  Messages on the display guide set-up and operation.

CNC diagnostics find faults, warn of failures and switch to spare circuits

Find the cause

When the machine stops, the diagnostic program identifies which component failed, so the technician goes straight to the fault.

Warn before failure

The controller monitors components and signals that one is close to failing, so it can be replaced during planned maintenance.

Use redundancy

Critical circuits are duplicated; when one fails, the controller switches to the spare and the machine keeps running.

Why diagnostics matter

Downtime: a CNC machine is expensive, so every hour it waits for repair is costly.

Effect: faster fault finding and planned replacement raise machine utilization.

CNC has six advantages over conventional NC

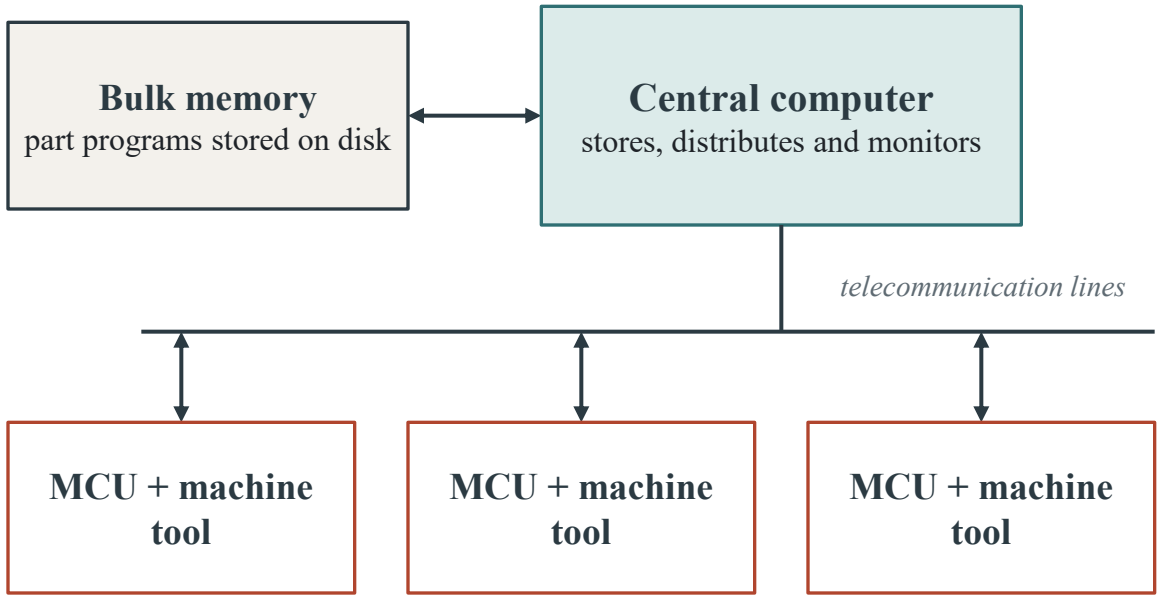
Advantage	Conventional NC	CNC
Tape read once	Tape read for every part; reader errors repeat	Program read once into memory; machining runs from memory
Editing at the machine	A change needs a new tape	Operator edits the stored program at the control
Metric conversion	Units fixed in the program	Software converts between inch and metric
Greater flexibility	New features need new hardware	New features added by changing software
User-written programs	Only the built-in functions	Macros and subroutines written by the user
Total manufacturing system	Stand-alone machine	Computer can link to DNC and plant computers

03

Direct numerical control (DNC)

One central computer serving many machine tools in real time

DNC connects many machine tools to one central computer in real time



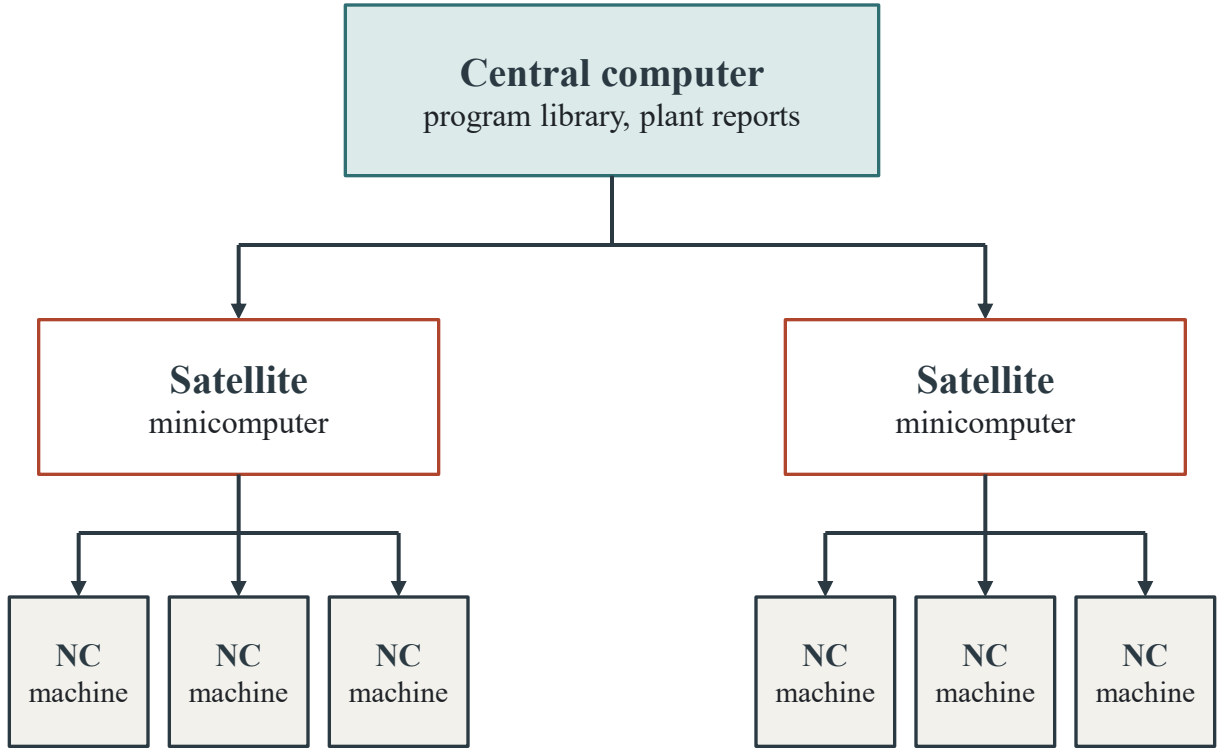
General configuration of a DNC system

Example: a shop of 20 NC machines

All 20 part programs sit on the central computer's disk. When an operator starts a job, the computer sends that program to the machine on demand.

- **Definition:** a system that connects a set of NC machines to a common computer for program storage and on-demand distribution of data.
- **Central computer:** manages storage, sends programs to machines and collects shop data.
- **Bulk memory:** holds the part program library.
- **Telecommunication lines:** carry data between the computer and the machines in real time.
- **Machine tools:** each with its own MCU that receives the blocks.

Satellite minicomputers take the real-time load off the central DNC computer

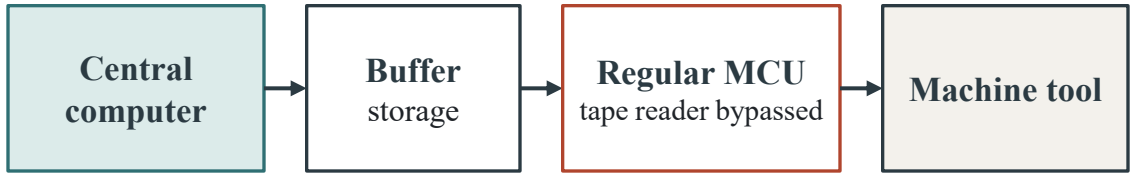


DNC with satellite minicomputers: each satellite serves a group of machines

- **Load sharing:** each satellite looks after several machines, so the central computer is not tied up sending blocks in real time.
- **Local storage:** the satellite holds the programs its machines need for the shift.
- **Fewer long lines:** only the satellites connect to the central computer.
- **Scale:** more machines can be added by adding another satellite.

DNC links to the machine either behind the tape reader or through a special MCU

Behind-the-tape-reader (BTR) system



- The computer connects to the existing NC controller in place of the tape reader.
- Blocks pass through buffer storage so the machine never waits for data.
- Cheap retrofit, but the controller's functions stay those of ordinary NC.

Special machine control unit



- A controller designed for DNC replaces the regular NC controller.
- Two-way communication: the machine reports status back to the computer.
- Easier data collection and more flexibility, at higher cost.

Which one to choose

Existing NC machines: BTR is the cheaper way to bring them under DNC without replacing their controllers.

New installations: a special MCU gives two-way communication and better shop-floor data.

DNC performs four functions

01 NC without punched tape

Programs travel from the central computer straight to each machine's controller, so tape and tape reader are no longer needed.

02 Part program storage

Active programs are kept ready for immediate transmission; the full library sits in secondary storage and is called up when needed.

03 Data collection, processing and reporting

The computer gathers piece counts, machine running time and tool data from the shop floor and turns them into reports for management.

04 Communications

Links the central computer with the machine tools, satellite computers, operator terminals and other plant computers.

DNC removes the tape and turns the shop into a source of management data

6

advantages of DNC over stand-alone NC

01 **No punched tape:** Tape and tape readers, the least reliable parts of NC, are removed.

02 **More computing power:** The central computer handles editing and calculations a hard-wired MCU cannot.

03 **Programs kept as computer files:** Easy to store, retrieve, edit and protect.

04 **Programs stored as CLFILE:** One program serves several machines; the computer postprocesses it for whichever machine needs it.

05 **Shop performance reports:** Real-time data on production, machine use and downtime for management.

06 **Framework for the automated factory:** The first step toward computer-integrated manufacturing.

NC, CNC and DNC differ in where the computer sits and what it controls

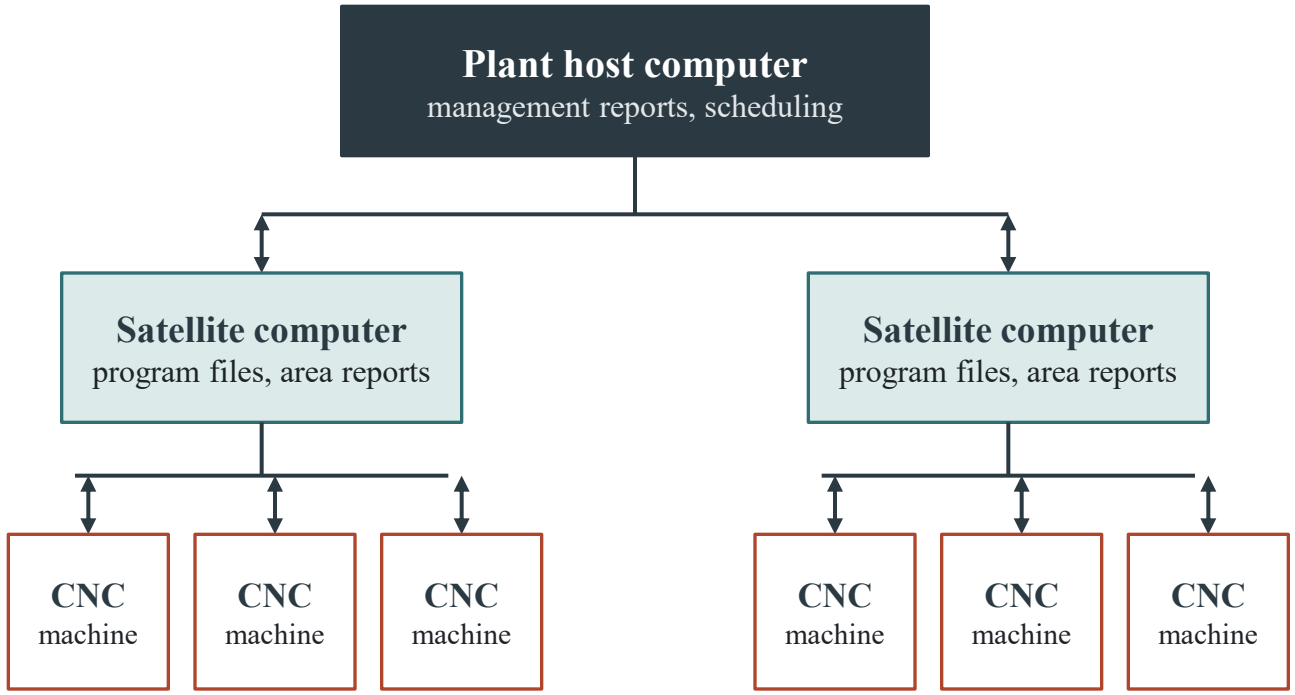
Basis	NC	CNC	DNC
Control	Hard-wired logic circuits	Dedicated computer at each machine	Central computer serving many machines
Program input	Punched tape, read for every part	Tape or keyboard, read once into memory	Sent from the central computer on demand
Program storage	On tape only	Controller memory	Bulk memory of the central computer
Editing	Punch a new tape	At the machine control	At the central computer
Flexibility	Low: change needs rewiring	High: change by software	High, across the whole shop
Management data	None	Limited, at each machine	Full shop-floor reports
Cost	Lowest	Moderate	Highest

04

Combined DNC/CNC systems

From direct to distributed NC: a hierarchy of computers

Combined DNC/CNC replaced central control with a hierarchy of computers



Hierarchical (distributed) DNC: CNC machines report up through satellite computers

- **Why it replaced 1970 DNC:** early DNC was expensive and rigid in its reports and hardware.
- **Flexible reporting:** the information system is tailored to what the firm needs.
- **Built in stages:** subsystems are installed one at a time, so cost is spread and each pays back as it is added.
- **Easier changes:** corrections are made as the system grows.

Distributed NC: the DNC idea survives, but control is spread across many computers, so DNC now stands for distributed numerical control.

Combining DNC with CNC adds program download, backup and better communication

01

Whole-program download

The DNC computer sends the complete program into CNC memory once, instead of block by block.

No punched tape at the machine and far less traffic on the lines. Next step: a postprocessor inside the CNC, so the program can be sent as a CLFILE.

02

Redundancy

If the central computer fails, each CNC machine can still run on its own.

Backup needs a file of punched tapes duplicating the stored programs, and a tape reader (fixed or portable) at each machine.

03

Better communication

Computers sit at both ends of every link between the office and the shop floor.

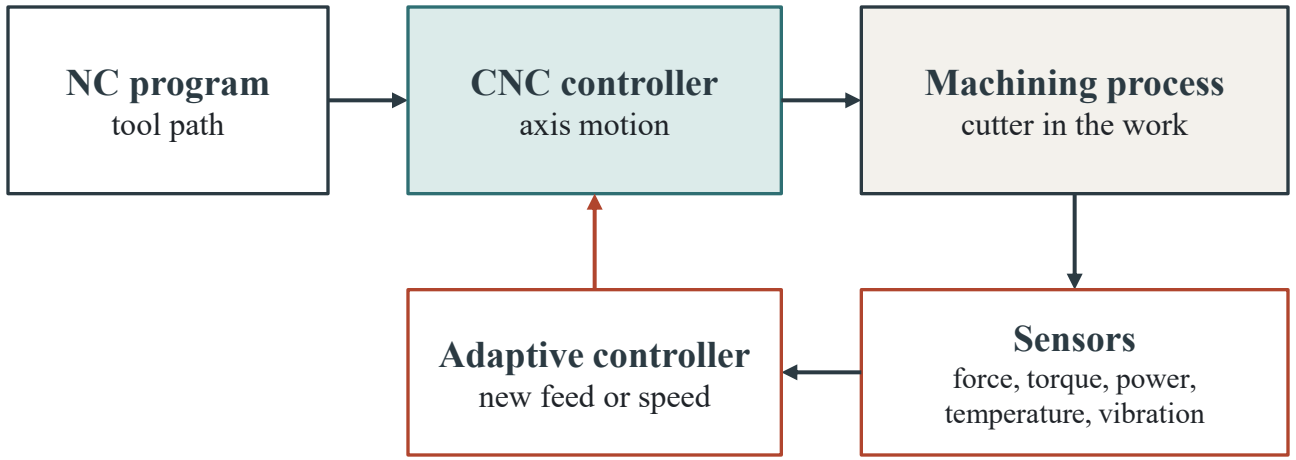
Computer-to-computer links are easier than links to hard-wired devices, so richer management information systems become possible.

05

Adaptive control machining

Adjusting feed and speed during the cut from measured process variables

Adaptive control sets feed and speed from process variables measured during the cut



NC sets the path; adaptive control sets feed and speed from what the sensors measure

Example: milling a casting with hard spots

NC: feed is chosen for the hardest spot, so most of the cut runs slowly.

AC: the controller cuts feed only at the hard spots and runs faster everywhere else.

- **Definition:** a control system that measures output process variables and uses them to control speed and/or feed.
- **Origin:** research at the Bendix Research Laboratories in the early 1960s, sponsored by the U.S. Air Force; first systems were analog.
- **Today:** microprocessor-based and integrated with an existing CNC system.
- **Aim:** run the process more efficiently, judged by metal removal rate and cost per volume of metal removed.

Adaptive control pays off when the cutter is engaged most of the time on variable, costly jobs

> 40%

of machine time with the cutter engaged in the work: the rule of thumb for justifying adaptive control (Mathias)

NC vs AC: NC cuts non-productive time (handling, setup, tool changes). AC cuts the in-process time, while the tool is actually cutting.

1. **In-process time is a large share of the cycle**
   Long cuts give AC time to pay back.

2. **The job has sources of variability**
   AC adapts feed and speed to changing conditions; the more variable, the better the fit.

3. **The machine tool is costly to operate**
   High equipment investment makes every minute saved valuable.

4. **The work material is hard to machine**
   Steel, titanium and high-strength alloys; cast iron and aluminum are candidates but easier to cut.

Five sources of variability make a machining job suitable for adaptive control

Source of variability	What happens	Adaptive control response
Variable geometry of cut	Depth or width of cut changes along the path, as in profile milling and contouring	Feed adjusted to keep the cutting load constant
Variable hardness and machinability	Hard spots or difficult areas in the workpiece	Speed or feed reduced to avoid early tool failure
Variable workpiece rigidity	Thin or poorly supported sections deflect under the cutting force	Feed reduced to keep deflection within limits
Tool wear	Forces rise as the cutter dulls	Feed reduced to hold the force at the set value
Air gaps	The cutter passes through holes, slots or gaps in the part	Feed increased to cross the gap quickly

ACO optimizes a performance index; ACC only holds measured variables within limits

Adaptive control optimization (ACO)
research systems

Index of performance = **metal removal rate ÷ tool wear rate**

- **Goal:** maximize an economic index of performance during the cut.
- **Problem:** tool wear rate cannot be measured on-line with today's sensors.
- **Status:** mostly confined to research until reliable sensors exist.

Adaptive control constraint (ACC)
production systems

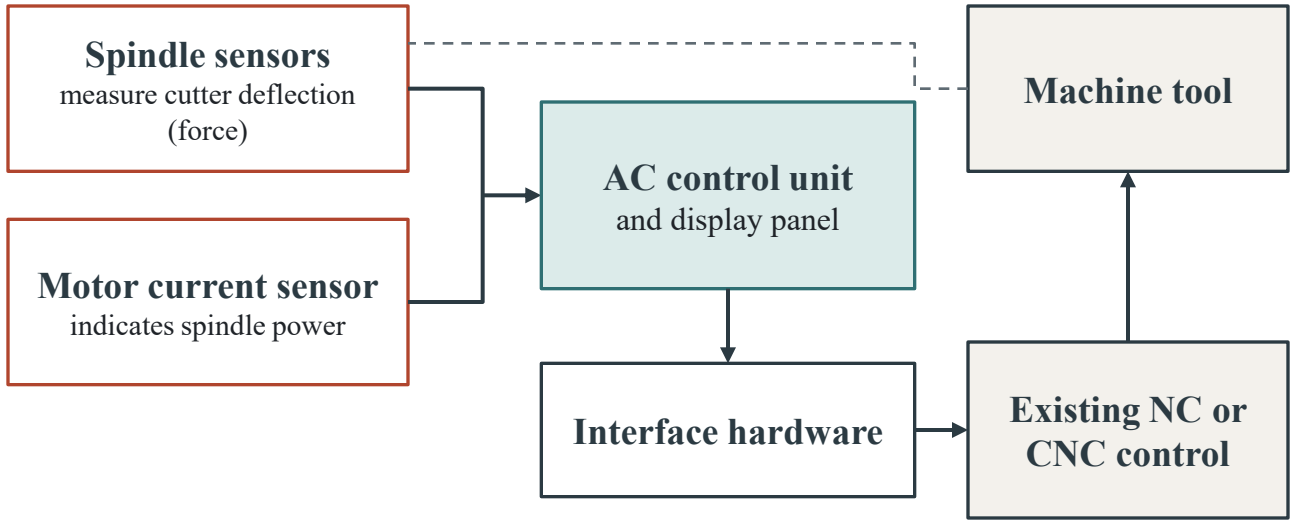
Keep force, power or torque **at or below its limit**

- **Goal:** manipulate feed and/or speed so measured variables stay within their constraint limits.
- **Simpler:** less sophisticated and less expensive than ACO.
- **Status:** nearly all adaptive control machining in industry is ACC.

Why ACC dominates: sensor problems make ACO impractical, so production systems control what they can measure: cutting force and spindle power.

An ACC system adds sensors, a control unit and a machinability program to an NC machine

Hardware (retrofit package)



Software

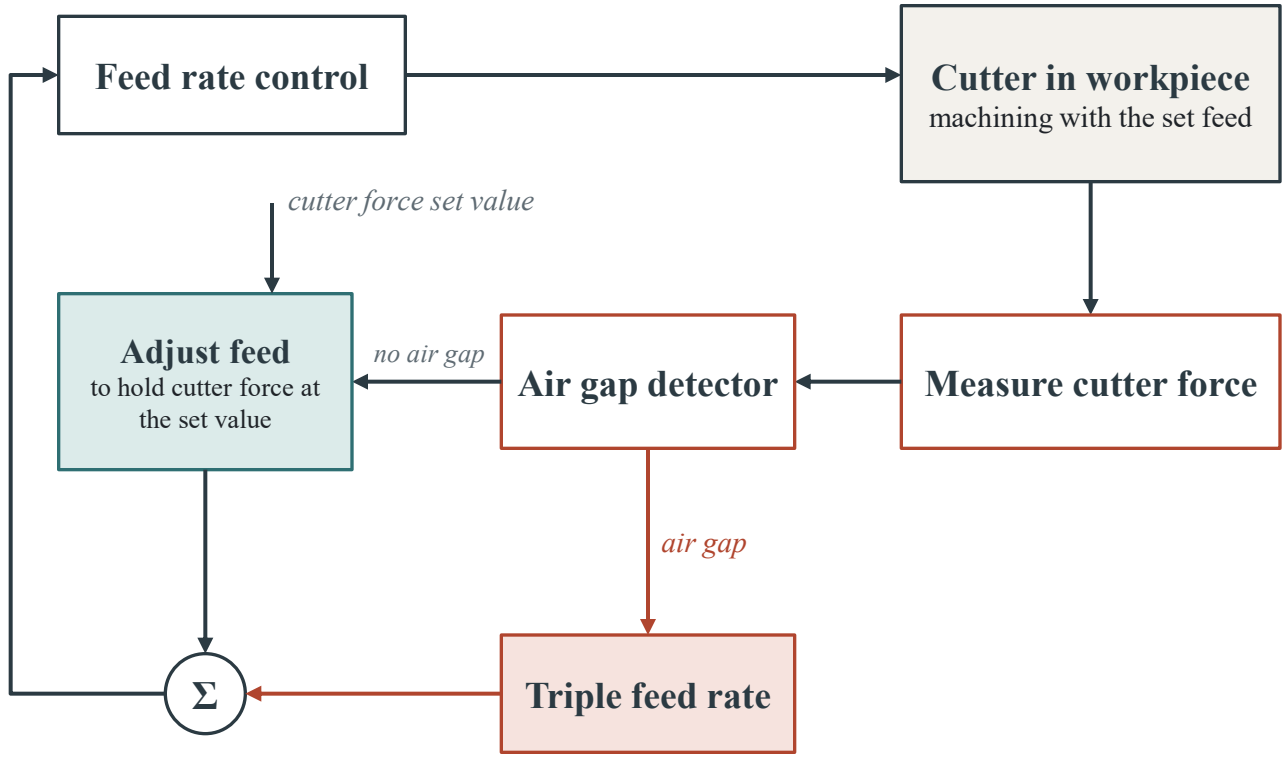
Inputs
cutter size and geometry, work material hardness, size of cut, machine tool characteristics

Machinability program
called as an APT MACRO statement in the part program

Outputs
starting feed and speed, and the cutter-force limit the controller must hold

Typical application: profile or contour milling on an NC machine. Feed is the controlled variable; cutter force and spindle power are the measured variables.

The force loop cuts feed under heavy load and triples it through air gaps

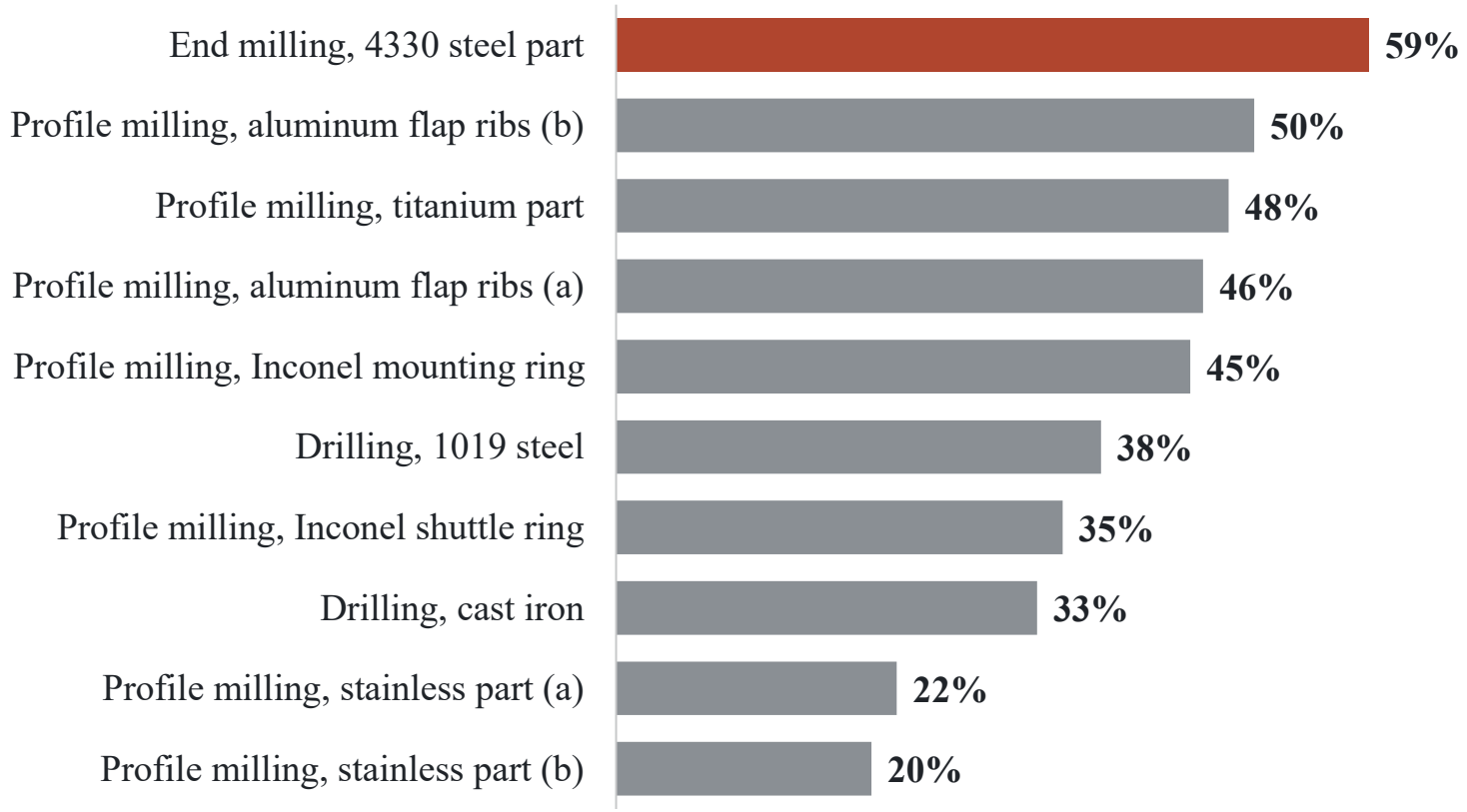


Adaptive control loop that uses cutter force as the measured variable

- **Measure:** a spindle sensor measures cutter force (deflection) continuously.
- **Check for air:** the air gap detector decides whether the cutter is actually in metal.
- **In metal:** feed is raised or lowered to hold cutter force at the set value; more force means less feed.
- **In an air gap:** feed is tripled so the cutter crosses empty space quickly.
- **Variables used:** Bendix tried temperature, torque and vibration; Macotech uses cutter load and spindle power, which protects the motor from overload.

Adaptive control cut machining time by 20% to 59% in production trials

Machining time saved by adaptive control compared with NC, by job



Source: Macotech Corp., as reported by Groover and Zimmers

- **Milling:** 20% to 59% less time on aluminum, stainless steel, Inconel, titanium and alloy steel parts.
- **Drilling:** 33% to 38% less time.
- **Smallest gains:** the two stainless steel aerospace parts, at about 20%.

Adaptive control gives five benefits in the jobs where it is justified

01

Higher production rate

On-line adjustment runs at the highest metal removal rate the conditions allow, so more parts are made per hour.

02

Longer tool life

Feed is reduced before the cutter is overloaded, so fewer tools break and wear is more uniform.

03

Greater part protection

The force limit can be set from the part's size tolerance, which guards against out-of-tolerance cuts.

04

Less operator intervention

Control moves from the machine operator to management, through the part programmer.

05

Easier part programming

The controller selects feed, so the programmer need not plan for the worst case; fewer tryouts are needed.

Where it works: milling, drilling, tapping, grinding and boring; only limited success in turning.

06

BPUT previous year questions

Module IV question bank, mark-wise

2-mark questions

1 Differentiate between NC machine and CNC machine.

---

2 What are the common problems encountered with conventional NC?

6-mark questions

1	Describe the architecture and components of a CNC system (Controller, Motors, Drives, and Encoders) and explain how they work together to move the tool.
2	Draw the configuration diagram of a DNC system and explain the function of each component.
3	What are the functions of CNC? State the advantages of CNC over conventional NC.
4	Distinguish between NC, CNC, and DNC systems.
5	What is meant by Adaptive Control in machining systems? Explain two approaches to adaptive control.
6	What are the problems with conventional NC machines?

16-mark question

1 What are the sources of variation during machining? Explain, with a neat sketch, the working principle of adaptive control.

MODULE V

Computer Integrated Manufacturing Systems

MFPC3005 Computer Aided Design and Manufacturing (CAD/CAM)

Reference: Groover, Automation, Production Systems and Computer-Integrated Manufacturing

Module V roadmap

01	**Computer integrated manufacturing system**	Production system, CIM components, database, automation types
02	**Machine tools and related equipment**	CNC centres, industrial robots, PLC, CMM and quality control
03	**Group technology, cells and process planning**	Part families, coding, PFA, rank order clustering, CAPP
04	**Flexible manufacturing systems**	Types, components, layouts, benefits, bottleneck model
05	**Materials handling system**	Principles, equipment, AGVS, conveyors, cranes, analysis
06	**Storage system**	Performance, conventional storage, AS/RS, carousels, analysis
07	**Computer system**	Hierarchy, networks, ERP, data collection, FMS control
08	**BPUT previous year questions**	Mark-wise question bank

By the end of Module V you should be able to

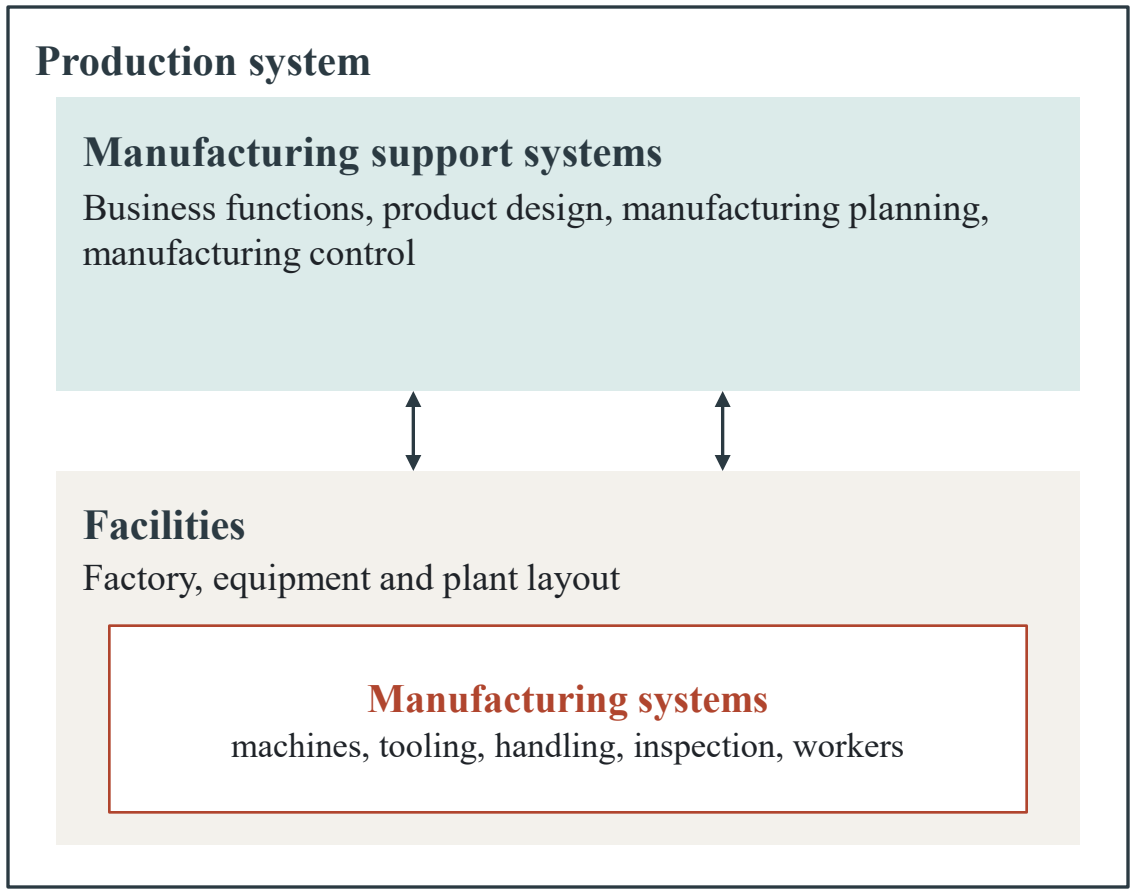
- Explain the production system, and define CIM with its objectives and components
- Compare fixed, programmable and flexible automation, and state the reasons for automating
- Describe CNC machining centres, industrial robots, PLCs and CMMs used in CIM
- Form part families using classification and coding, PFA and rank order clustering
- Explain cellular manufacturing, and distinguish variant and generative CAPP
- Explain the types, components, layouts and benefits of FMS, and apply the bottleneck model
- State the principles of material handling, and explain AGVs, conveyors and cranes
- Describe AS/RS and carousel storage, and compute AS/RS capacity and throughput
- Explain the computer hierarchy, networks, ERP and FMS computer control in CIM

01

Computer integrated manufacturing system

Production systems, CIM components, the common database and types of automation

A production system combines facilities with the support systems that manage them



Production system: support systems over facilities

- **Facilities:** the factory, its production machines and tooling, material handling, inspection equipment and the plant layout.
- **Manufacturing systems:** the groups of equipment and workers inside the facilities that touch the product.
- **Support systems:** the procedures that manage production and solve its technical and logistics problems.
- **Information cycle:** business functions take the order, product design defines the part, planning schedules it and control executes it on the shop floor.

CIM uses computers to integrate every function from design to shipment

Computer integrated manufacturing (CIM)

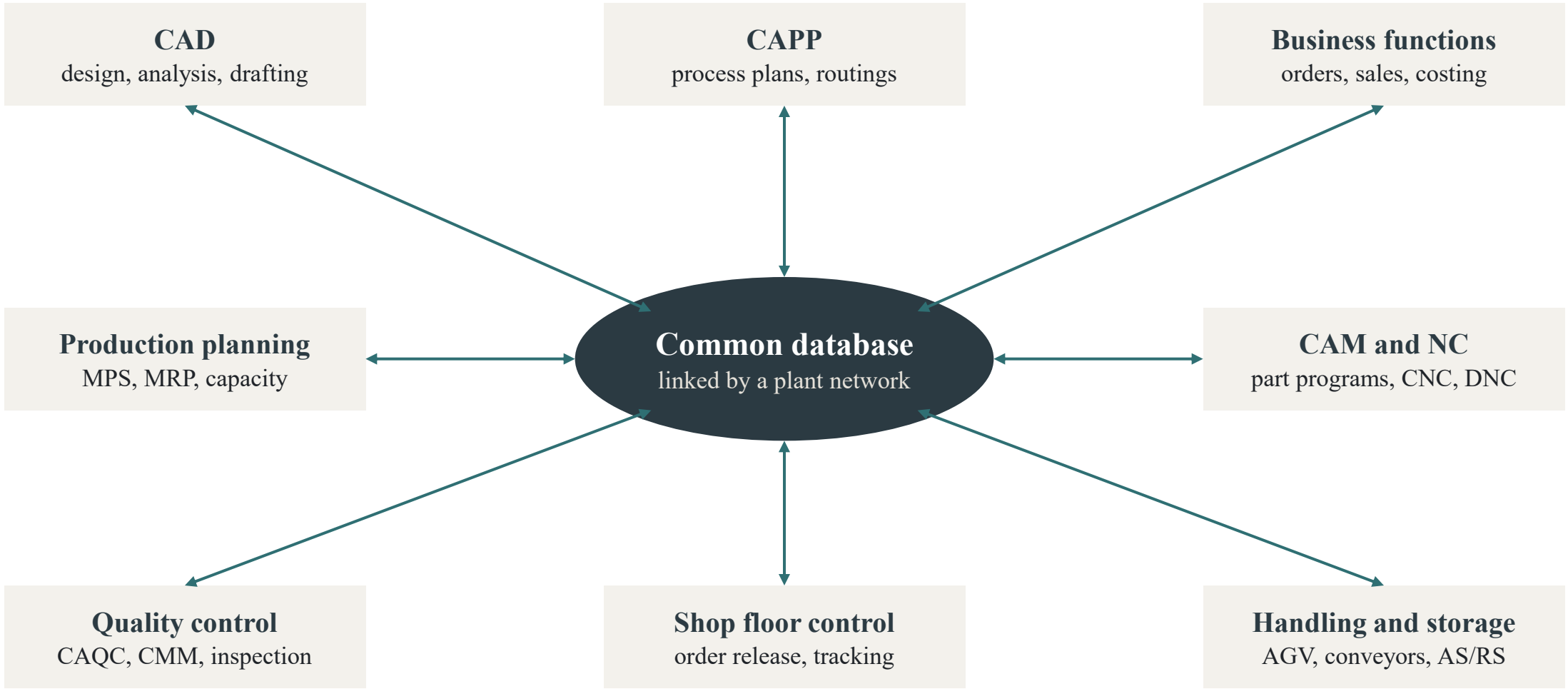
The use of computers to plan, control and link all functions of a firm (business, design, manufacturing planning and shop floor) through one shared database.

Objectives of CIM

- Shorter manufacturing lead time
- Lower inventory and work-in-process
- Better quality from consistent data
- Higher utilisation of equipment
- Faster response to design and demand changes
- Less paperwork and lower production cost

Basis	CAM	CIM
Scope	Shop-floor operations	Whole firm: business, design and production
Functions	NC programming, process planning, machine control	CAD, CAPP, CAM, PPC, CAQC, handling, business
Data	Separate files for each function	One common database
Integration	Islands of automation	All functions linked by a network
Benefit	Faster, accurate machining	Shorter lead time, lower inventory, faster response

CIM links eight functions through one common database and a network



A centralised database lets every CIM function work from the same data

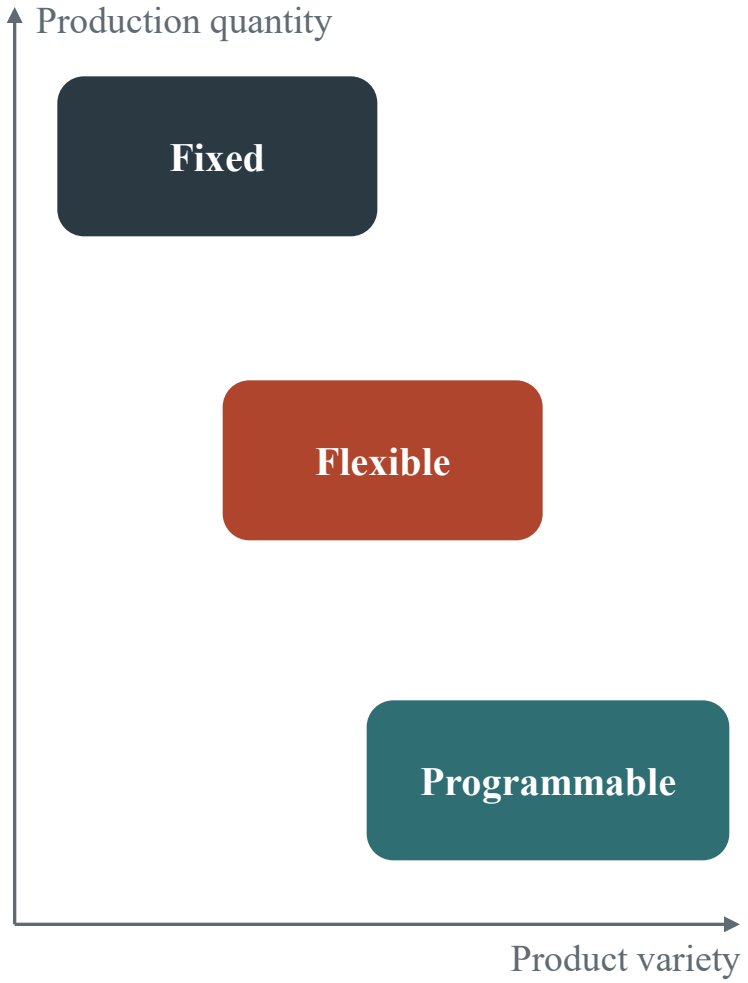
- **One source:** design, process and production data stored once and shared by all functions.
- **No re-entry:** CAPP, NC programming and inspection read the part geometry from the CAD model.
- **Consistency:** an engineering change reaches every function at once.
- **Traceability:** order status, quality records and tool data held together for reports.
- **Control:** the DBMS manages security, backup and many users at once.

Design attributes	Manufacturing attributes
Basic external shape	Major process
Basic internal shape	Operation sequence
Length/diameter ratio	Batch size
Material type	Annual production
Part function	Machine tools
Major dimensions	Cutting tools
Tolerances and finish	Production time

Design attributes describe what the part is; manufacturing attributes describe how it is made. Group technology codes and CAPP use both.

The type of automation follows production quantity and product variety

Type	Features	Examples
Fixed	High investment in custom equipment; high production rate; cannot change product	Transfer lines, automated assembly
Programmable	Program changed for each batch; setup time lost between batches	NC and CNC machines, robots, PLCs
Flexible	Changeover with no lost time; can produce a mix of products	FMS, flexible assembly lines



Companies automate for nine reasons

Raise labour productivity
More output per labour hour

Reduce labour cost
Machines replace rising wage costs

Ease labour shortages
Fill jobs workers cannot be found for

Remove routine tasks
Free workers from boring manual work

Improve worker safety
Move people away from hazardous work

Improve product quality
Uniform output and fewer defects

Cut manufacturing lead time
Faster response to customer orders

Do what manual work cannot
Precision, size or complexity beyond hand work

Avoid the cost of not automating
Stay competitive with automated rivals

02

Machine tools and related equipment

Automated machines, robots, controllers and inspection equipment that do the work in a CIM plant

Manufacturing systems range from manual work to fully automated machines

Manual work system

One or more workers using hand tools, with no powered equipment.

Example: filing, hand assembly, manual inspection

Worker-machine system

A worker operates powered equipment; each does what it does best.

Example: an operator running an engine lathe or press

Automated system

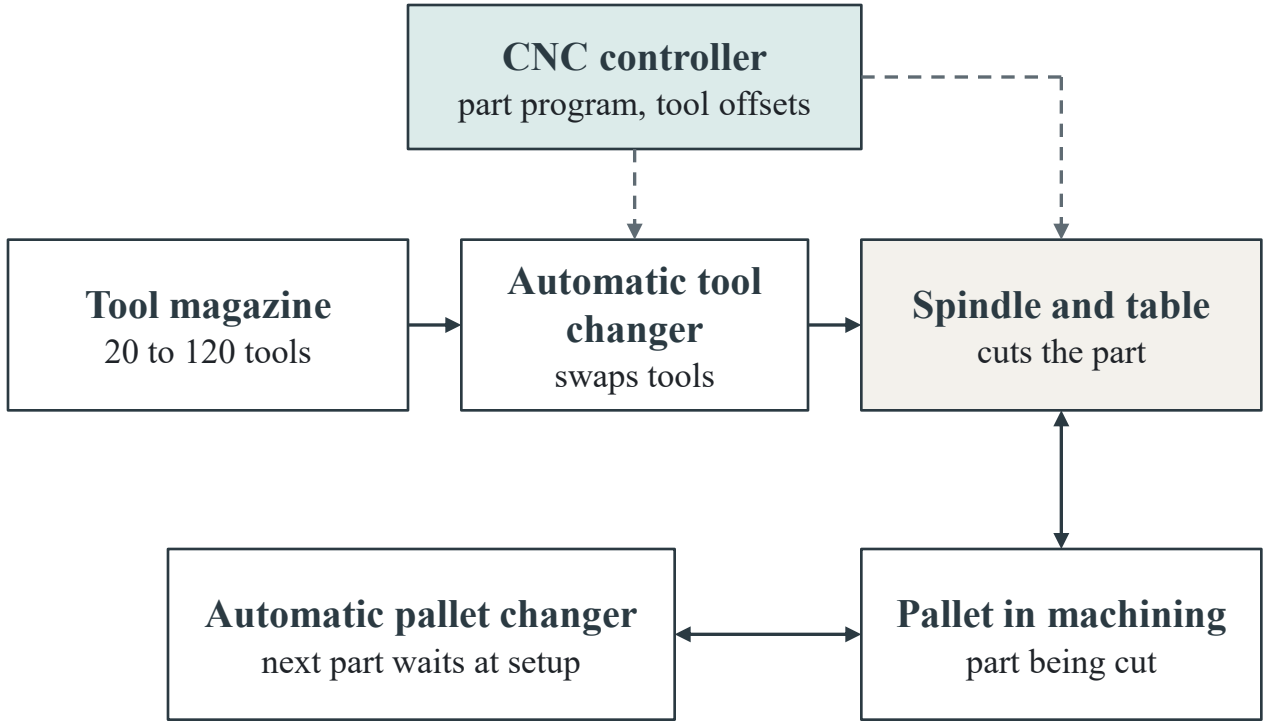
The process runs without direct worker participation.

Semi-automated: worker loads and unloads each cycle. **Fully automated:** runs unattended for long periods.

Automated systems used in a CIM plant

Automated machine tools (CNC machining and turning centres), transfer lines, automated assembly systems, industrial robots, and automated inspection such as CMMs and vision systems.

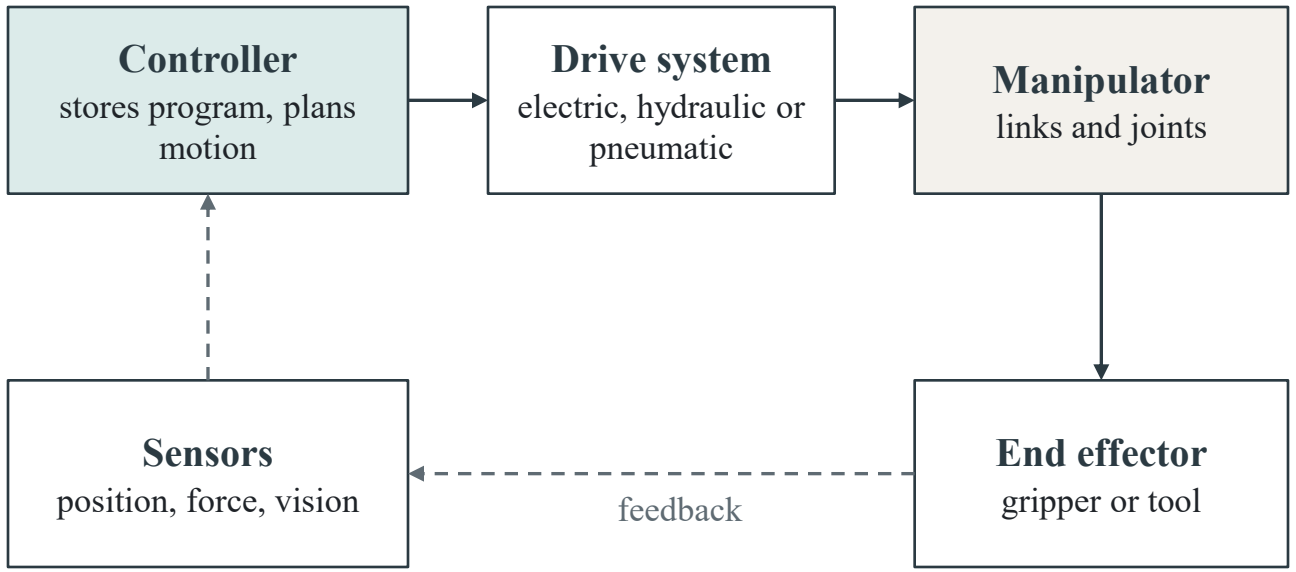
CNC machining centres change tools and pallets without stopping for the operator



Tool and pallet changing on a machining centre

- **Machining centre:** a CNC machine for milling, drilling and boring, built with horizontal or vertical spindles and three to five axes.
- **Automatic tool changer:** a robot arm exchanges tools between the magazine and the spindle in seconds.
- **Pallet changer:** the operator fixtures the next part while the current one is cut.
- **Turning and mill-turn centres:** CNC lathes that add rotating tools, so one setup finishes the part.

An industrial robot is a programmable manipulator built from five components



Components of an industrial robot system

- **Manipulator:** a chain of links joined by rotary or linear joints that positions the wrist.
- **End effector:** the gripper or tool (welding gun, spray gun) fixed to the wrist.
- **Drive system:** actuators that move each joint.
- **Controller:** stores the program and coordinates the joints.
- **Sensors:** internal sensors report joint position; external sensors see or feel the work.

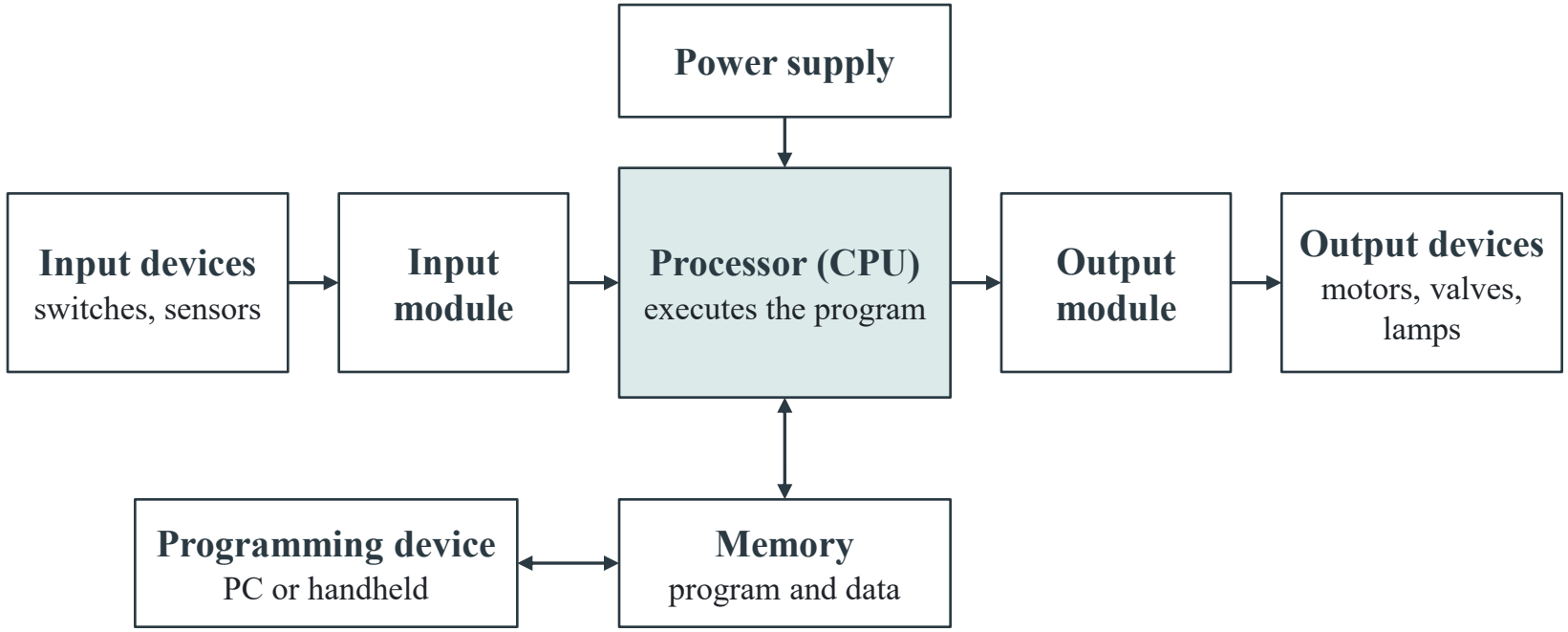
Robots are programmed by leading them through the task or by writing a program

Method	How it works	Typical use
Powered lead-through	Operator drives each joint with a teach pendant; the controller records the points	Spot welding, machine loading
Manual lead-through	Operator moves the arm by hand through the path; the controller records it	Spray painting, arc welding
Textual programming	Program written in a robot language such as VAL or AML	Assembly, sensor-based tasks
Off-line programming	Program built and simulated on a computer, then downloaded	New cells; no lost production time

Programming functions

Define points and paths; choose joint, straight-line or circular motion; set speed; open and close the gripper; read sensors and interlocks with other machines; branch and call subroutines.

A PLC replaces relay panels with a stored program that scans inputs and sets outputs



Block diagram of a programmable logic controller

Scan cycle

1. Read all inputs
2. Execute the ladder logic program
3. Update all outputs

Repeat every few milliseconds

Applications

- Machine and conveyor sequencing
- Transfer lines and assembly
- Batch process control

A CMM measures part geometry by moving a probe along three axes

Components

01 **Probe head and probe**
touch-trigger or scanning probe senses the surface

02 **Mechanical structure**
bridge, cantilever, gantry or horizontal arm on a granite table

03 **Drive system and scales**
moves each axis and reads x, y, z position

04 **Computer and software**
computes dimensions, form and position, and compares them with CAD

Benefits

- Inspection time cut by a factor of 5 to 10 over manual methods
- Higher accuracy and repeatability
- One machine measures many part types
- Results stored for statistical process control
- Programs generated directly from the CAD model

Computer-aided quality control moves inspection from the gauge room into the process

Objectives of CAQC

- Inspect 100% of parts instead of samples
- Detect defects at the process, not after it
- Cut inspection time and labour
- Feed quality data back to correct the process
- Link inspection to the CAD/CAM database

Basis	Fixed gauging	Flexible inspection system
Flexibility	One gauge for one part	Reprogrammed for new parts
Part variety	Single part	Whole part family
Cost	Low per gauge; new gauge per part	High initial; low per new part
Inspection	Go/no-go, often sampled	Full dimensional data, every part

100% online inspection

An FIS (CMM, probes and vision under computer control) checks every part within the process cycle, so a drifting process is caught and corrected at once.

03

Group technology, cells and process planning

Grouping similar parts into families and machines into cells, and planning their processes by computer

Group technology groups similar parts into families that share setups and machines

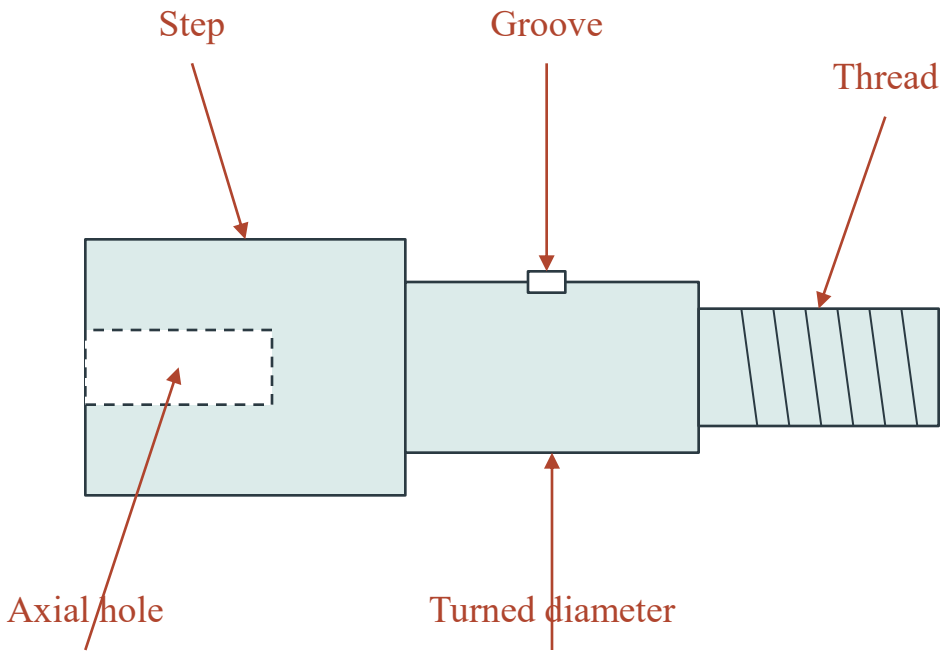
Group technology (GT)

A philosophy that identifies similar parts and groups them into **part families** so their likeness can be used in design and production.

Part family: parts similar in shape and size, or in the processing steps they need.

Benefits

- Fewer setups and shorter lead times
- Design retrieval avoids duplicate parts
- Standard process plans and tooling
- Cell layouts cut handling and work-in-process



Composite part concept

One machine cell set up for the composite part can make any member of the family.

The Opitz system codes a part in nine digits: a form code and a supplementary code

Form code (digits 1–5)

Supplementary code (digits 6–9)

1	2	3	4	5	6	7	8	9
Part class	Main shape	Rotational machining	Plane surface machining	Auxiliary holes, gear teeth	Dimensions	Material	Original shape of raw material	Accuracy

A **secondary code** (letters A–D) may follow, defined by each company for its own production needs. Digit 1 separates rotational from nonrotational parts, and it decides how digits 2–5 are read.

Hierarchical (monocode)

Each digit's meaning depends on the digit before it. Compact, but hard to build.

Chain (polycode)

Each digit has a fixed meaning in every position. Easy to read, but longer.

Hybrid

Mixes both structures, as Opitz does. The most common choice.

Production flow analysis forms families from routings instead of drawings

Stage 1

Data collection

Part numbers and operation routings from the route sheets

Stage 2

Sort routings

Group parts with identical machine sequences into packs

Stage 3

PFA chart

Plot each pack against the machines it visits: the incidence matrix

Stage 4

Cluster analysis

Rearrange rows and columns into blocks of part families and machine groups

Strengths

- Uses route sheets the plant already has
- No drawing classification or coding needed
- Gives machine groups for cells directly

Limits

- Routings may be outdated or not optimal
- Families depend on how parts happen to be made
- Large matrices need a clustering method such as ROC

Rank order clustering reorders the incidence matrix into part families and machine groups

1. Read each row of 1s and 0s as a binary number, and compute its decimal value.
2. Rank the rows in decreasing order of value, and reorder them.
3. Read each column as a binary number from top to bottom, and compute its value.
4. Rank the columns in decreasing order of value, and reorder them.
5. If no row or column moved, stop. Otherwise return to step 1.

Row value

$$V_i = \sum_{j=1}^{n} b_{ij} 2^{n-j}$$

where b = 1 if part j visits machine i, and n is the number of parts (columns). Columns use the same rule with m rows.

Result: 1s gather into blocks along the diagonal. Each block is a part family and the machine group that forms its cell.

Worked example: rank order clustering finds two cells in a 5 × 6 matrix

Initial incidence matrix

	A	B	C	D	E	F	Value
M1	1		1			1	41
M2		1		1	1		22
M3		1			1		18
M4	1					1	33
M5				1	1		6

Matrix after clustering

	A	F	C	E	B	D
M1	1	1	1			
M4	1	1				
M2				1	1	1
M3				1	1	
M5				1		1

Pass 1: row values M1 41, M4 33, M2 22, M3 18, M5 6, so rows are reordered. Column values are then A 24, F 24, C 16, E 7, B 6, D 5, so columns are reordered.

Pass 2: no row or column moves, so the method stops.

Result: two cells Parts A, C, F on machines M1, M4; parts B, D, E on machines M2, M3, M5

Cellular manufacturing dedicates a group of machines to one part family

Objectives of a cell

- Shorter throughput time
- Less work-in-process
- Fewer setups and simpler tooling
- Better quality: one team owns the parts
- Simpler scheduling and less handling

Cell design criteria

- Part family size and production volume
- Number and type of machines needed
- Layout: in-line, U-shaped or loop
- Handling within the cell
- Utilisation of the key machine
- Multi-skilled operators per cell

Lean manufacturing

Production with the least waste: overproduction, waiting, transport, inventory, motion and defects. Cells, just-in-time and pull scheduling are its shop-floor tools.

CAPP builds process plans by retrieving a family plan or by generating one from rules

Conventional process planning: a route sheet built by the planner in six steps

Interpret the drawing	Select processes	Select machines	Sequence operations	Select tooling	Estimate times

Basis	Variant (retrieval) CAPP	Generative CAPP
Approach	Retrieves the standard plan for the part's family and edits it	Creates a new plan from part data and decision logic
Input	GT code of the new part	Part geometry from CAD, material, tolerances
Database	File of family plans	Knowledge base of process and machine rules
Planner's role	Edits the retrieved plan	Little or no intervention
Strength	Simple and cheap to set up	Consistent plans; no family plans needed
Limit	Needs families; planner skill still needed	Complete logic is hard to build

04

Flexible manufacturing systems

Computer-controlled groups of CNC machines with automated handling, built for mid-volume, mid-variety production

An FMS links CNC workstations by automated handling under one computer system

Flexible manufacturing system (FMS)
A group of CNC workstations joined by an automated material handling and storage system, all under distributed computer control, that can process a mix of parts at once.

FMS fits when

- Batch production leaves machines idle between setups
- A manned GT cell is ready to be automated
- Parts form a family by similar geometry or by a common product
- Annual volume is about 5,000 to 75,000 parts

Where FMS sits among production systems



A system is flexible if it can identify parts, change programs and change setups quickly

Identify and tell apart each part | **Change operating instructions quickly** | **Change the physical setup quickly**

Part variety
Can it process different parts in a mix?

Schedule change
Can it accept changes in mix and quantity?

Error recovery
Can it keep running after a breakdown?

New part
Can a new part design be added easily?

Type	Part variety	Schedule change	Error recovery	New part
Single machine cell	Yes, one part at a time	Yes	Limited: one machine	Yes
Flexible manufacturing cell (2–3 machines)	Yes, several at once	Yes	Limited: few machines	Yes
FMS (4 or more machines)	Yes, several at once	Yes	Spare machines cover breakdowns	Yes

A dedicated FMS trades flexibility for production rate; a random-order FMS does the reverse

Dedicated FMS

- Limited part variety, known in advance
- Special-purpose machines
- Higher production rate, close to a transfer line

Random-order FMS

- Large family; new designs introduced
- General-purpose CNC machines
- Complex control; lower production rate

How a manufacturing cell differs from an FMS

Basis	Manufacturing cell	FMS
Machines	Usually 1 to 3	4 or more
Handling	Often manual loading and transfer	Automated handling and storage
Control	Local CNC; little system control	Central computer coordinates the system
Part variety	A small family	A wider family, processed at once

Every FMS has four components

01

Workstations

Load/unload stations, CNC machining and turning centres, assembly stations, CMM and other inspection, cleaning and chip removal.

02

Material handling and storage

Moves parts between stations, buffers work-in-process, and presents parts on pallet fixtures.

03

Computer control system

A central computer and station microcomputers coordinate machines, handling, tools and data.

04

Human resources

People load and unload parts, change tools, maintain and repair equipment, program parts and manage the system.

The FMS handling system moves parts between stations and holds them as a buffer

Five functions of the handling system

01 Move parts between any stations, in any order

02 Handle different part shapes on pallet fixtures

03 Store work-in-process temporarily

04 Give easy access for loading and unloading

05 Work under computer control

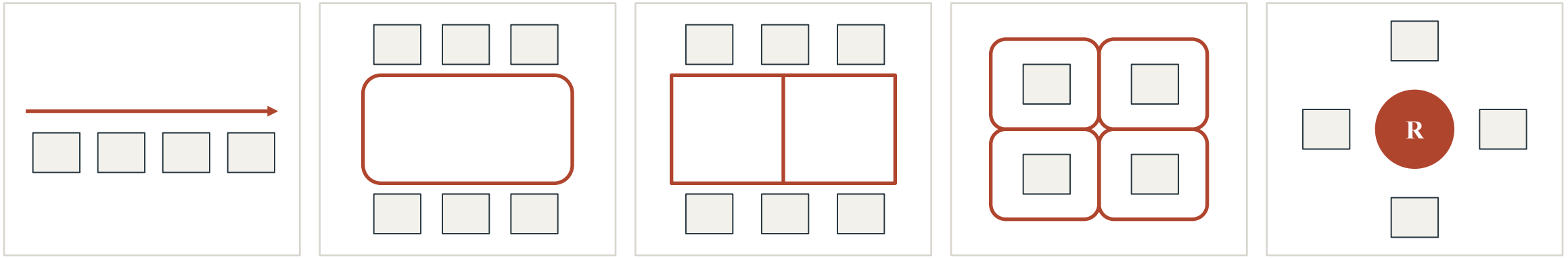
Primary handling system

- Sets the layout of the FMS
- Moves parts between stations
- Conveyors, AGVs, rail-guided carts

Secondary handling system

- Shuttles at each station
- Transfers the part from the primary system to the machine
- Queues parts so the machine is never idle

Five FMS layouts suit different routing variety and handling equipment



In-line

Little variation in sequence

Equipment: Conveyor, shuttle

Loop

Higher variety; flow in one direction

Equipment: Conveyor, towline

Ladder

Rungs shorten travel between machines

Equipment: Conveyor, AGV, rail cart

Open field

Large family; many routes

Equipment: AGV, in-floor towline

Robot-centred

Rotational parts on a few machines

Equipment: Industrial robot

FMS layout types: red shows the handling path, boxes are workstations

FMS raises machine utilisation to 80–90% and cuts inventory by 60–80%

Applications

- **Machining:** nonrotational parts on machining centres, the most common use
- **Turning:** rotational parts, less mature
- **Sheet metal:** flexible fabricating systems with an AS/RS, rail carts and CNC punch presses
- **Assembly:** flexible lines with robots and PLC control

80–90%
machine utilisation, against far less in a batch shop

40–50%
less floor space

60–80%
less work-in-process inventory

30–50%
higher labour productivity

Fewer
machines needed for the same output

24 h
unattended operation on night shifts

Planning, design and operating decisions decide whether an FMS performs

Planning

- Part family to be made
- Processing requirements
- Size and weight of parts
- Production volume

Design

- Types of workstations
- Routing variety and layout
- Handling system
- WIP and storage capacity
- Tooling
- Pallet fixtures

Operation

- Scheduling and dispatching
- Machine loading
- Part routing
- Part grouping
- Tool management
- Pallet and fixture allocation

The bottleneck model estimates FMS production rate from station workloads

Terms

- **p_j:** fraction of output that is part j
- **s_i:** number of servers at station i
- **t_{ijk}:** time of operation k of part j at station i
- **f_{ijk}:** frequency of that operation
- **t_{n+1}:** mean transport time per move
- **WL*, s*:** workload and servers at the bottleneck station, which has the largest WL_i/s_i

Part mix	$\sum_j p_j = 1.0$
Station workload	$WL_i = \sum_j \sum_k t_{ijk} f_{ijk} p_j$
Transports per part	$n_t = \sum_i \sum_j \sum_k f_{ijk} p_j - 1$
Handling workload	$WL_{n+1} = n_t t_{n+1}$
Maximum production rate	$R_p^* = \frac{s^*}{WL^*}$
Rate of part j	$R_{pj}^* = p_j R_p^*$
Utilisation of station i	$U_i = \frac{WL_i}{s_i} R_p^*$
Busy servers at station i	$BS_i = WL_i R_p^*$

Worked example: milling is the bottleneck, so the FMS makes 3.33 parts per hour

Part data (times in min)

Part	Mix	Load	Mill	Drill	Unload
A	0.4	4	30	10	2
B	0.6	4	40	15	2

Routing 1 → 2 → 3 → 1 for both parts, so $n_t = 3$ moves of 3.0 min.
Servers: load/unload 1, mill 2, drill 1, carriers 4. All f = 1.0.

Station results

Station	s	WL	WL/s	U	BS
Load/unload	1	6.0	6.0	33.3%	0.33
Mill	2	36.0	18.0	100%	2.00
Drill	1	13.0	13.0	72.2%	0.72
Handling	4	9.0	2.25	12.5%	0.50

WL in min per part; milling has the largest WL/s.

$$R_p^* = \frac{2}{36.0} = 0.0556 \text{ pc/min} = 3.33 \text{ pc/hr}$$

$$R_{pA}^* = 3.33(0.4) = 1.33 \text{ pc/hr}, \qquad R_{pB}^* = 3.33(0.6) = 2.00 \text{ pc/hr}$$

05

Materials handling system

Principles, equipment, AGV systems, conveyors, cranes and the analysis of vehicle-based systems

Material handling accounts for 20–25% of manufacturing labour cost

Material handling

The movement, storage, protection and control of materials throughout manufacturing, distribution, consumption and disposal. It adds cost but no value to the product, so it should be kept to the minimum.

20–25%

of total manufacturing labour cost goes on handling material

External logistics

- Transport between facilities
- Rail, truck, air, ship and pipeline
- Planned by distribution management

Internal logistics

- Movement and storage inside a facility
- Between machines, stores and docks
- The subject of factory material handling

Ten principles guide the design of a material handling system

01
Planning
Define needs, goals and methods before buying equipment

02
Standardisation
Standard methods, equipment and controls

03
Work
Cut handling work: fewer, shorter moves

04
Ergonomic
Fit tasks to people; avoid heavy, repetitive lifts

05
Unit load
Move material in pallets and containers

06
Space utilisation
Use floor and overhead space well

07
System
Coordinate handling with storage and production

08
Automation
Automate to raise efficiency and control

09
Environmental
Save energy; reduce waste and noise

10
Life cycle cost
Judge equipment on its total cost of ownership

Handling equipment falls into four categories chosen by the material, its flow and the layout

Equipment categories

01 **Transport**
trucks, AGVs, rail vehicles, conveyors, cranes

02 **Storage**
racks, shelving, AS/RS, carousels

03 **Unitizing**
pallets, tote boxes, palletizers

04 **Identification and tracking**
bar codes, RFID, machine vision

Design factors

- **Material:** physical state, size, weight, shape, condition and safety risk
- **Flow rate:** quantity moved per hour
- **Routing:** pickup and drop-off points and distances
- **Scheduling:** urgency of each delivery
- **Layout:** process, product or fixed-position

Unit load principle
Move the largest practical load at once, on a standard pallet or container, to cut the number of trips.

Six types of transport equipment differ in cost, routing flexibility and load

Equipment	Features	Typical applications
Manual industrial trucks	Low cost; low deliveries per hour	Light loads over short distances
Powered trucks (forklift, walkie, tractor)	Medium cost; flexible routing; needs a driver	Pallet loads in warehouses and factories
Automated guided vehicles	High cost; battery powered; computer controlled; flexible routing	FMS, mixed-model assembly, warehousing
Monorails and rail-guided vehicles	High cost; fixed rail routing; flexible in speed	Assembly lines, moving large loads
Conveyors	Fixed route; mechanised; many types	Products along a line; sorting
Cranes and hoists	Lift and move heavy loads	Large items in factories and yards

An AGVS moves loads on driverless, battery-powered vehicles along defined paths

01

Driverless train

A towing vehicle pulling several trailers; suits heavy loads over long distances.

02

AGV pallet truck

Picks up and drops pallet loads at floor level in warehouses.

03

Unit load carrier

Carries one unit load on its deck; transfers loads to stations by roller or lift.

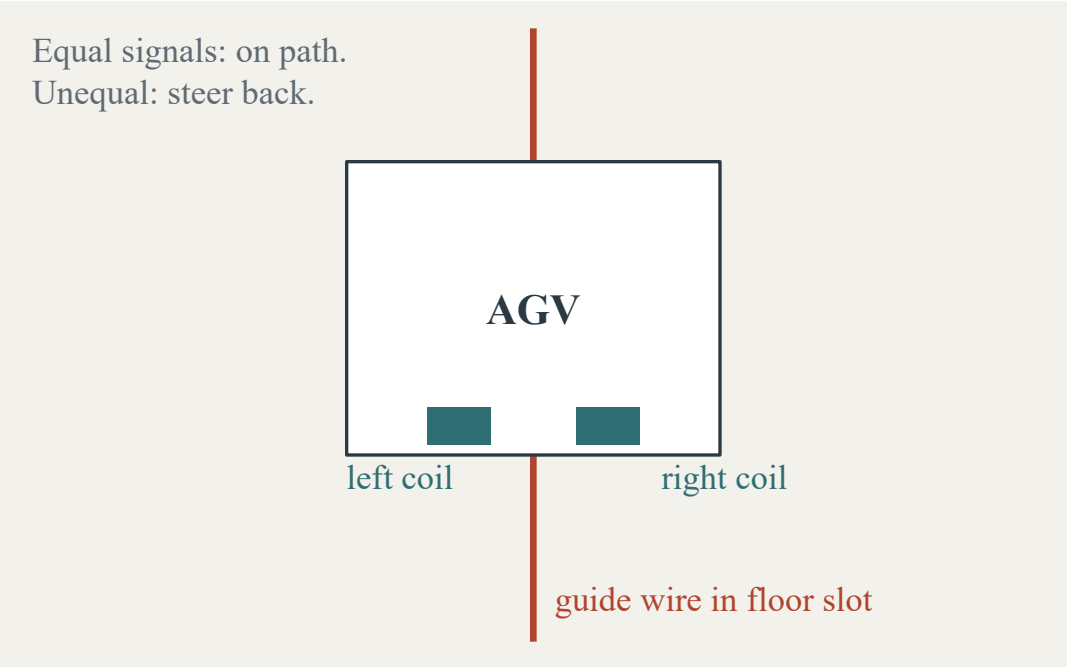
Applications

- Driverless train operations
- Storage and distribution
- Assembly line work carriers
- Flexible manufacturing systems

Role and advantages in FMS

- Random routing to any station
- Paths changed without new hardware
- Floor stays clear for people
- Computer dispatch and load tracking

AGVs follow a guide wire by balancing the signal between two sensor coils



Guide-wire sensing with two coils

Other guidance: paint strips read optically, and self-guided vehicles using dead reckoning with beacons.

- **Guidance:** a frequency in the wire creates a magnetic field that the coils sense.
- **Routing:** at a branch the vehicle chooses by frequency select or path switch select.
- **Traffic control:** on-board forward sensing, or zone control that allows one vehicle per zone.
- **Dispatching:** on-board panel, remote call stations, or a central computer.
- **Safety:** speed below walking pace, obstacle sensors, emergency bumpers and warning lights.

Conveyors move parts along a fixed path, powered or by gravity

Type	How it moves the load
Roller	Load rides on rotating rollers, powered or gravity
Skate-wheel	Wheels on shafts; light loads, gravity
Belt	Continuous flat or troughed belt
In-floor towline	Chain in the floor pulls four-wheel carts
Overhead trolley	Trolleys on an overhead rail, pulled by a chain
Power-and-free	Trolleys can be released from the chain to stop
Cart-on-track	Carts on a track driven by a spinning tube

Operation

- **Powered vs nonpowered:** a drive moves the load, or gravity and workers do
- **Continuous vs asynchronous:** constant speed, or stop-and-go
- **Single direction:** loads move one way, from loading to unloading
- **Continuous loop:** carriers return empty to the start
- **Recirculating:** parts not unloaded go round again

Rail-guided vehicles, cranes and hoists cover fixed routes and heavy lifts

01

Monorails and rail-guided vehicles

Self-propelled vehicles on a fixed rail, overhead or on the floor, powered through the rail.

More flexible than conveyors, less than AGVs.

02

Cranes

Bridge crane: girder spans the bay on overhead rails.

Gantry crane: legs run on floor rails.

Jib crane: arm swings round a column.

03

Hoists

Lift loads vertically by pulleys and chain or rope.

Mechanical advantage equals the number of rope lines supporting the load.

Worked example: an AGVS loop needs four vehicles for 40 deliveries per hour

Delivery cycle time	$T_c = T_L + \frac{L_d}{v_c} + T_U + \frac{L_e}{v_e}$
Available time per vehicle-hour	$AT = 60AT_fE$
Deliveries per vehicle-hour	$R_{dv} = \frac{AT}{T_c}$
Vehicles needed	$n_c = \frac{R_f}{R_{dv}}$

A = availability, Tf = traffic factor, E = worker efficiency, Rf = required deliveries per hour.

Data

Load 0.75 min, unload 0.50 min; loaded travel 110 m, empty 80 m at 50 m/min; A = 0.95, Tf = 0.90, E = 1.0; Rf = 40 per hour.

Solution

$$T_c = 0.75 + 2.20 + 0.50 + 1.60 = 5.05 \text{ min}$$

$$AT = 60(0.95)(0.90)(1.0) = 51.3 \text{ min}$$

$$R_{dv} = \frac{51.3}{5.05} = 10.16$$

$$n_c = \frac{40}{10.16} = 3.94 \rightarrow 4 \text{ vehicles}$$

06

Storage system

Storage performance, conventional methods, AS/RS and carousels, and AS/RS sizing and throughput

Storage systems are judged on six measures, and randomized storage needs fewer locations

Measure	Meaning
Capacity	Number of loads or volume the system can hold
Density	Capacity per unit of floor area
Accessibility	Ease of reaching any stored item
Throughput	Loads stored and retrieved per hour
Utilisation	Share of capacity in use; 80–90% is the target
Reliability	Share of time the system is available

Location strategies

Randomized
Any load goes to the nearest empty slot; space sized for the average inventory

Dedicated
Each item has fixed slots; space sized for each item's maximum

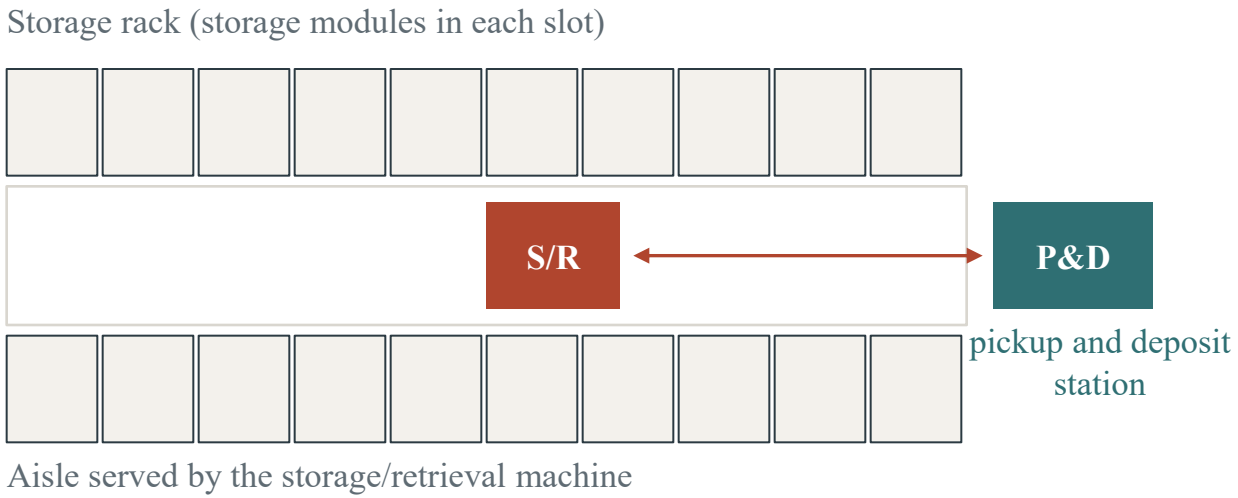
Class-based
Dedicated zones by activity class, randomized within each zone

For the same items, randomized storage may need about 3,000 locations where dedicated storage needs 5,500.

Conventional storage methods trade density against access and cost

Method	Advantages	Disadvantages	Typical items
Bulk storage	Lowest cost; high density	Poor access; stacking limits	Pallet loads, crates
Rack systems	Good density and access; low cost	Racks fixed in place	Pallet loads
Shelves and bins	Low cost; good for small items	Floor space for large items	Small parts, tools
Drawer storage	High density; good access	Higher cost; small items only	Tools, small parts
Automated (AS/RS, carousel)	High throughput; computer control	High investment	WIP, unit loads, small parts

An AS/RS stores and retrieves loads with a computer-controlled machine in each aisle



Components

- **Storage structure:** steel rack that holds the loads
- **S/R machine:** carriage that stores and retrieves loads
- **Storage modules:** pallets, tote bins or drawers
- **P&D station:** where loads enter and leave
- **Controls:** computer that sets each slot and logs every move

Top view of one AS/RS aisle

S/R machine drives: horizontal travel along the aisle, vertical lift on the mast, and a shuttle that moves the load into and out of the slot.

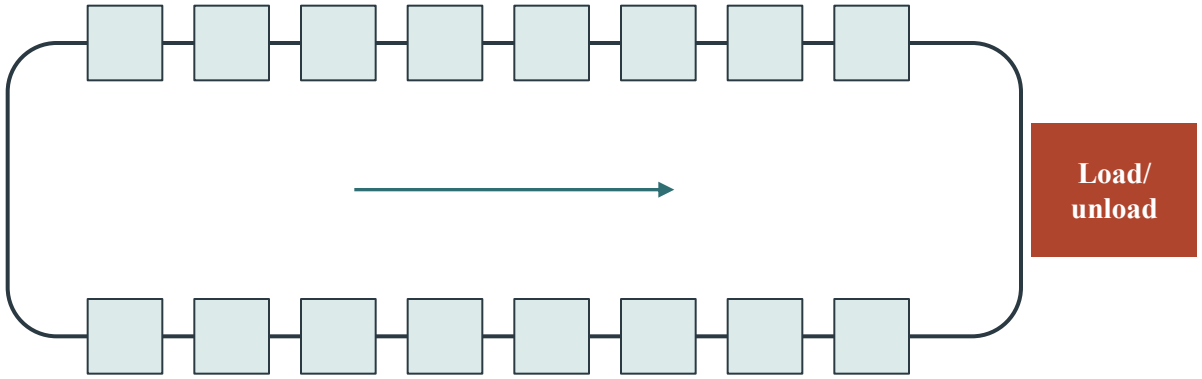
Item location file: the computer records what is in every slot.

Six AS/RS types cover loads from full pallets to single small items

Type	Description	Typical use
Unit load	Large automated system for pallet loads; S/R machine in each aisle	Warehousing, distribution
Deep-lane	High-density; loads stored many deep, fed in one end	Few items in large quantities
Miniload	Small loads in bins or drawers	Small parts, tools
Man-on-board	Operator rides the carriage and picks single items	Order picking
Automated item retrieval	Items pushed from lanes onto a conveyor	Order picking of small items
Vertical lift module	Trays in a vertical column brought to an opening	Small parts, saves floor space

Applications: unit load storage and handling, order picking, and work-in-process storage between production stages.

A carousel brings stored bins to the operator on an oval conveyor loop



Horizontal carousel: bins on an overhead loop

Basis	AS/RS	Carousel
Motion	S/R machine moves to the load	Loads move to the operator
Structure	Fixed racks	Bins on a moving loop
Capacity added by	More aisles	More carousels

- **Horizontal or vertical:** horizontal loops run overhead; vertical loops save floor space.
- **Drive:** top-driven or bottom-driven.
- **Control:** manual switch, keyboard, or computer linked to inventory.
- **Applications:** storage and retrieval, transport and accumulation, WIP buffers, and electrical testing of parts as they circulate.

Worked example: a four-aisle AS/RS holds 5,760 loads and handles about 90 transactions per hour

$$Capacity\ per\ aisle = 2n_y n_z$$

$$W = 3(x + a), \qquad L = n_y(y + b), \qquad H = n_z(z + c)$$

$$T_{cs} = 2 \max\left(\frac{L}{v_y}, \frac{H}{v_z}\right) + 2T_{pd}$$

$$T_{cd} = 2 \max\left(\frac{1.5L}{v_y}, \frac{1.5H}{v_z}\right) + 4T_{pd}$$

$$R_{cs}T_{cs} + R_{cd}T_{cd} = 60U, \qquad R_t = R_{cs} + 2R_{cd}$$

x, y, z: load size; a, b, c: clearances; Tpd: pickup and deposit time.

Data

4 aisles, 60 bays × 12 tiers per side; load 42 × 36 × 48 in; a = 6, b = 8, c = 10 in; vy = 400 ft/min, vz = 60 ft/min; Tpd = 0.4 min; U = 90%, equal single and dual cycles.

Size

Capacity = 2(60)(12) = 1,440 per aisle, 5,760 in total. W = 3(48) = 144 in = 12 ft; L = 60(44) = 220 ft; H = 12(58) = 58 ft.

Throughput per aisle

Tcs = 2(0.967) + 0.8 = 2.73 min; Tcd = 2(1.45) + 1.6 = 4.50 min. R(2.73 + 4.50) = 54, so R = 7.47 of each cycle and Rt = 22.4 transactions/hr.

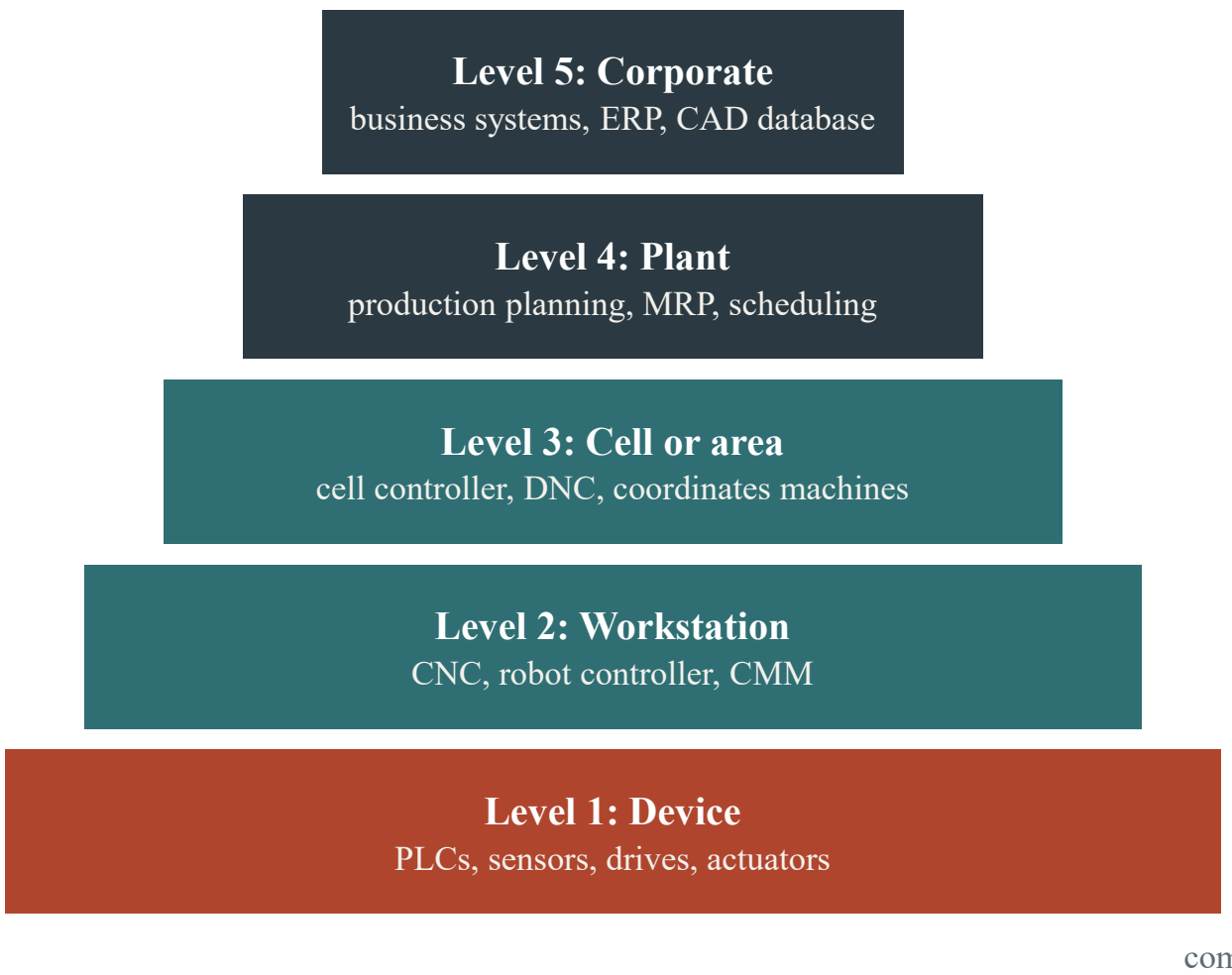
System: 4 × 22.4 ≈ 90 transactions per hour.

07

Computer system

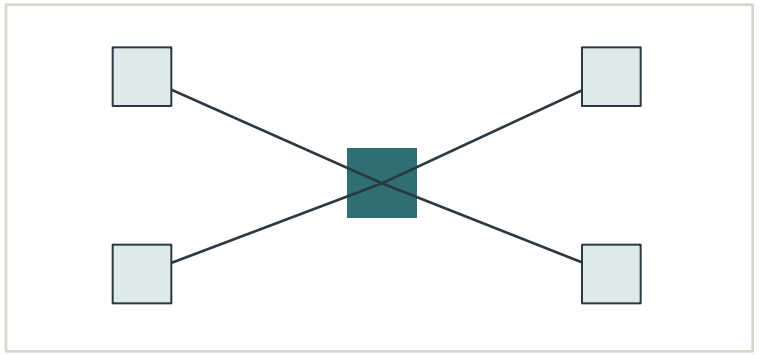
How computers are organised, networked and used to plan and control a CIM plant

CIM computers form a five-level hierarchy from corporate systems down to devices

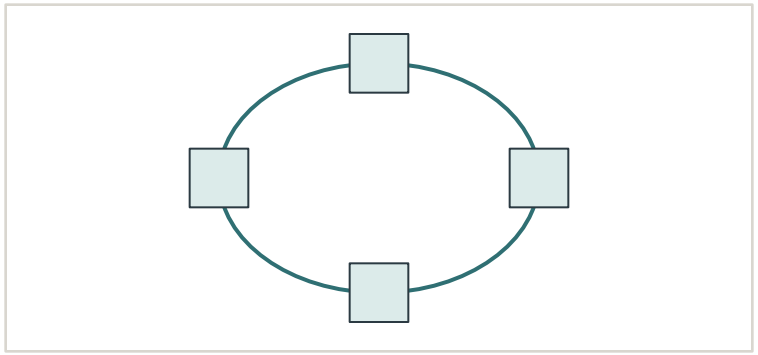


- Each level plans over a longer time horizon than the one below it
- Commands and schedules flow down
- Status, counts and quality data flow up
- Lower levels must answer in real time, in milliseconds

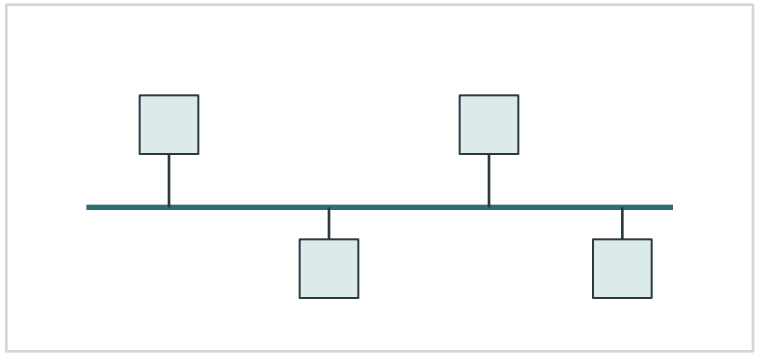
A plant network lets every computer in the hierarchy share data in real time



Star
all nodes wired to a central hub



Ring
each node passes data to the next



Bus
all nodes share one cable

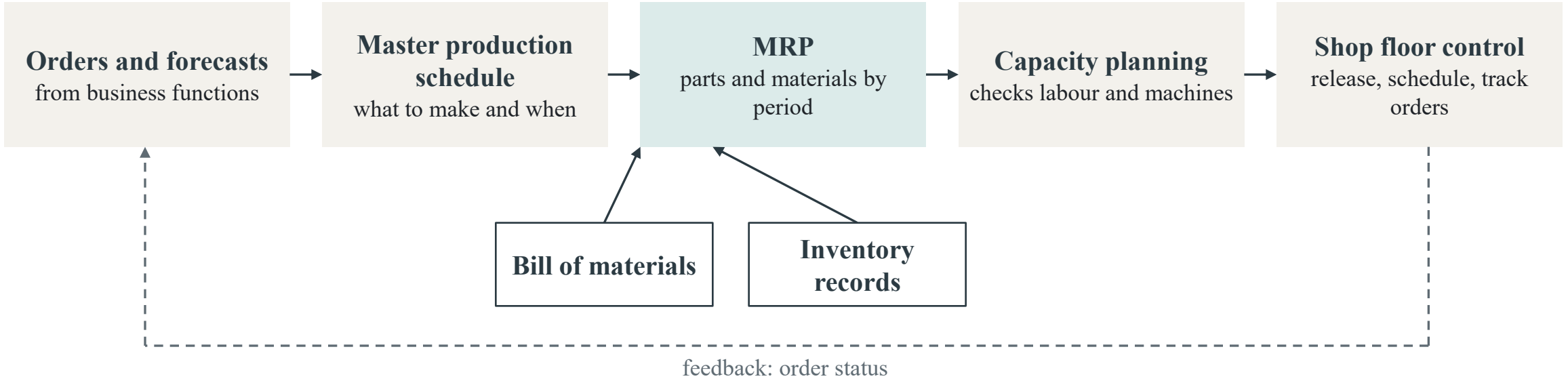
Why networking is vital

- One database reached by all functions
- Programs downloaded to machines (DNC)
- Real-time status from the shop floor

Common protocols

- MAP for the factory floor; TOP for offices
- Ethernet with TCP/IP across the plant
- Fieldbus (Profibus, DeviceNet) for devices

Manufacturing planning and control turns customer orders into a shop-floor schedule



Process planning

Route sheet for each part

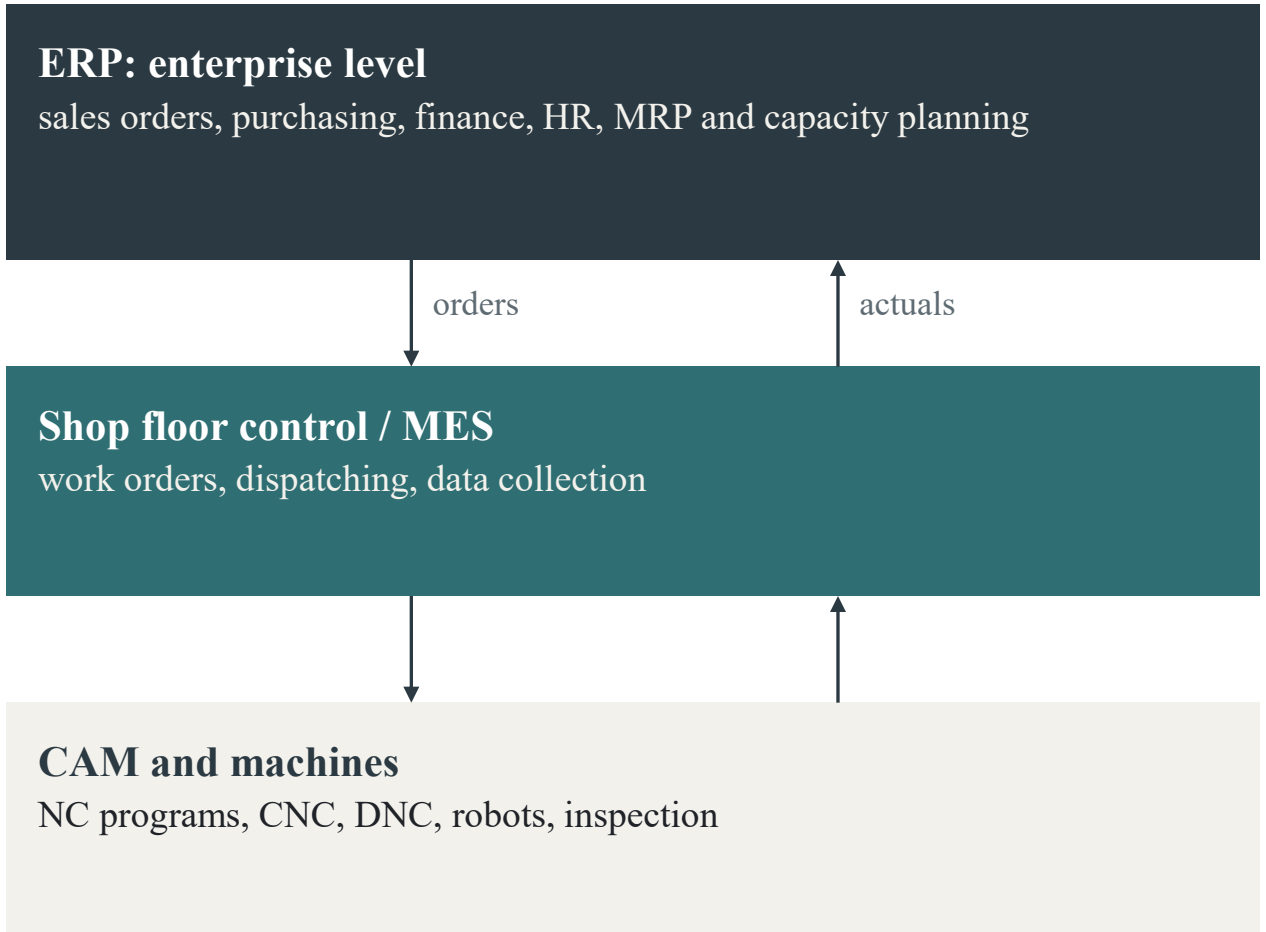
Inventory control

Right stock at least holding cost

Quality control

Inspection, SPC and corrective action

ERP connects the business, the CAM workflow and the factory floor in one system



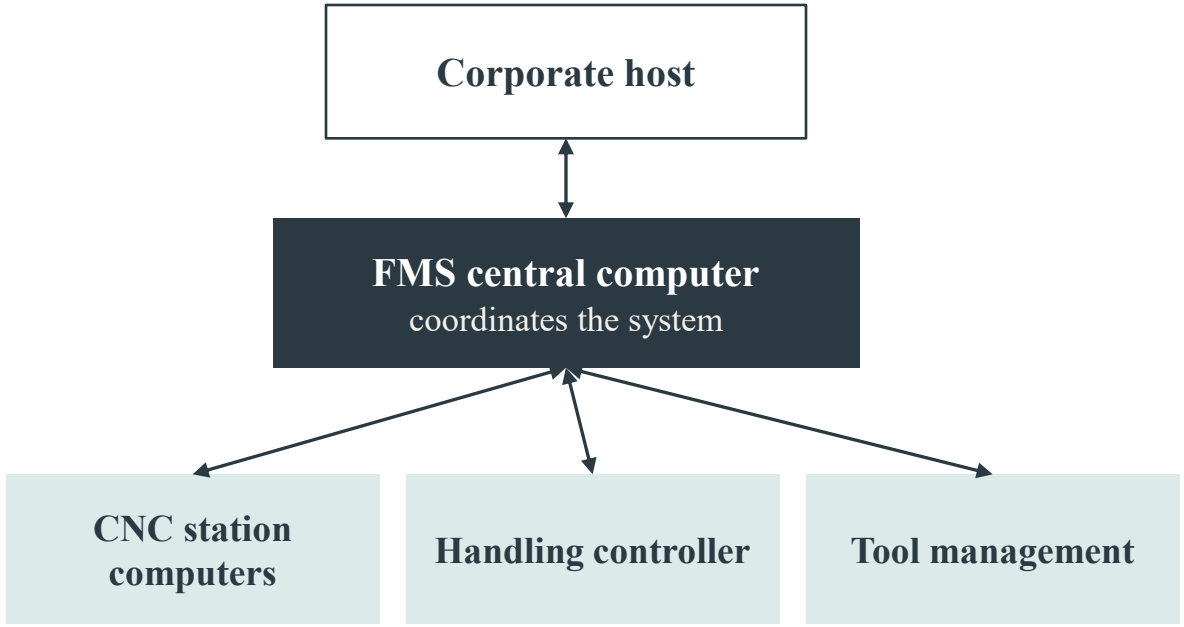
- **One plan:** a sales order becomes material, capacity and work orders without re-entry.
- **CAM link:** routings and NC programs are tied to each work order.
- **Feedback:** completions, scrap and machine time post back to costing and inventory.
- **Visibility:** managers see order status and costs as production runs.

Shop floor data collection feeds the computer system with what actually happened

Method	How data is captured	Typical use
Data terminals	Workers key in order, operation and quantity	Job start and finish, labour time
Bar codes	Labels scanned on parts, totes and job tickets	Order tracking, inventory moves
RFID tags	Radio tags read without line of sight	Pallets, tools, AGV loads
Direct machine links	PLC and CNC report counts and states	Utilisation, downtime, piece counts
Vision and sensors	Cameras and gauges record measurements	Quality data, 100% inspection

Data collected: order status, piece counts, scrap, machine utilisation and downtime, labour hours and quality results.

An FMS computer system is a DNC network that performs nine control functions



Two-way DNC structure of FMS control

Commands and programs go down to each machine; execution and performance data come back up to the central computer.

1. Workstation control
2. Distribution of control instructions
3. Production control
4. Traffic control
5. Shuttle control
6. Workpiece monitoring
7. Tool control
8. Performance monitoring and reporting
9. Diagnostics

Tool control and performance reports keep the FMS running and visible to managers

Tool control

Tool location

If a tool needed by a part is missing at its station, the computer routes the part to another station that has it, or tells the operator to load the tool.

Tool life monitoring

A tool life is stored for every cutter and machining time is logged. When the time reaches the life, the computer calls for replacement.

Report	Contents
Availability	Uptime of each workstation and reasons for downtime
Utilisation	Use of each station and the FMS average over a period
Production	Daily and weekly output by part, against schedule
Tooling	Tools at each station and their life status
Status	Snapshot of current operation, on demand

Summary

01	**CIM system**	One database and network link design, planning, production and business.
02	**Machine tools and equipment**	CNC centres, robots, PLCs and CMMs carry out and check the work.
03	**Group technology and CAPP**	Part families and cells cut setups; CAPP plans by variant or generative logic.
04	**FMS**	CNC stations, automated handling and computer control give mid-volume flexibility; the bottleneck sets output.
05	**Materials handling**	Ten principles and the right equipment, from trucks to AGVs, conveyors and cranes.
06	**Storage**	AS/RS and carousels raise throughput; sizing and cycle times set capacity.
07	**Computer system**	A five-level hierarchy, plant networks, ERP and FMS control tie it together.

08

BPUT previous year questions

Module V question bank, mark-wise

2-mark questions

1. What is the main function of a Programmable Logic Controller (PLC)?
2. What is meant by a 'lead through' programming method in robotics?
3. Define the term 'Part Family' as used in Group Technology. What is the composite part concept?
4. State two objectives of Computer Aided Quality Control.
5. What are the main objectives/goals of a Computer Integrated Manufacturing (CIM) system?
6. Differentiate between Design Attributes and Manufacturing Attributes in a CIM environment.

2-mark questions

7 State three classifications of automation systems.

8 How is Group Technology related to Computer Aided Process Planning (CAPP)?

9 Why is networking vital in modern manufacturing industries?

10 What do you mean by lean manufacturing?

11 List any two types of Automated Material Handling Systems.

6-mark questions

1	Discuss the role of ERP (Enterprise Resource Planning) in integrating the CAM workflow with a factory floor.
2	Discuss the role and importance of a centralized database in a CIM environment.
3	What is an AGV (Automated Guided Vehicle)? Explain its role and advantages in a Flexible Manufacturing System (FMS).
4	Outline the basic structure and steps of conventional process planning.
5	Explain the different automation strategies used for various production systems.
6	Differentiate between Computer Integrated Manufacturing (CIM) and Computer-Aided Manufacturing (CAM). Discuss the strategic and operational benefits of adopting a full CIM framework over an isolated CAM setup.
7	State the major components and benefits of a Coordinate Measuring Machine (CMM).

6-mark questions

9	Define the concept of cellular manufacturing. Explain its operational philosophy and discuss the primary objectives of implementing a cellular layout on a factory floor.
10	What is Computer Aided Quality Control? Discuss its objectives in a CIM system.
11	Explain the Opitz classification system used for part classification and coding in Group Technology.
12	Discuss the various modules required for the integration of CAD/CAM with an inspection system.
13	What are the different methods of robot programming? Explain.
14	How is a manufacturing cell different from an FMS? Discuss the criteria used in the design of a machine cell.
15	Write a short note on Flexible Inspection Systems (FIS) and Coordinate Measuring Machines (CMM).

16-mark questions

1	What is a Flexible Inspection System (FIS)? Discuss how an FIS differs from traditional fixed gauging techniques in terms of flexibility, part variety, and cost. Explain the strategic role of FIS in achieving 100% online inspection.
2	What is Group Technology? Describe its various benefits and explain the Production Flow Analysis (PFA) method in detail, outlining its key stages and how it groups machines and part families using manufacturing routings.
3	Explain the components of an industrial robotic system. Describe the various programming methods and programming functions used in robotic operations.
4	Sketch a neat block diagram showing the main components of a Computer Integrated Manufacturing (CIM) system and briefly explain each component.
5	What is the basic structure of a process plan? Explain the necessity of Computer Aided Process Planning (CAPP) in manufacturing systems and its role in CAD/CAM integration.

16-mark questions

6	What is the difference between variant and generative process planning methodologies? Explain in detail.
7	Sketch the layout of a Flexible Manufacturing System (FMS) and explain its major subsystems. Also state the advantages of FMS.
8	Explain the steps involved in the rank order clustering technique. Apply the technique to a part-machine incidence matrix to identify logical part families and machine groups.
9	What are the components of a CIM system? Discuss how CIM integrates various factory functions, including its role in production planning & control and quality control.
10	State the principles of material handling. Explain the working principle of Automated Guided Vehicles (AGVs).