Lesson 2: Planar Transformations and Mathematical Foundations

Modified: Jun. 2, 2026 Published: Sep. 26, 2025



Become Contributor ★ Expert Support

A SCARA robot arm picks a component from a feeder tray and places it on a PCB pad with sub-millimeter accuracy. To do that, the controller must convert a target position in board coordinates into the joint angles that move the end-effector there, accounting for link lengths, joint offsets, and the orientation of every frame in the chain. That conversion is built entirely on 2D transformation matrices. Get the math wrong and the arm places the component on the wrong pad or crashes into the board. In this lesson you will learn planar transformations through SCARA robot programming: complex number rotations, 2D rotation matrices, and forward and inverse kinematics for precision pick-and-place. #PlanarTransformations #SCARA #Kinematics

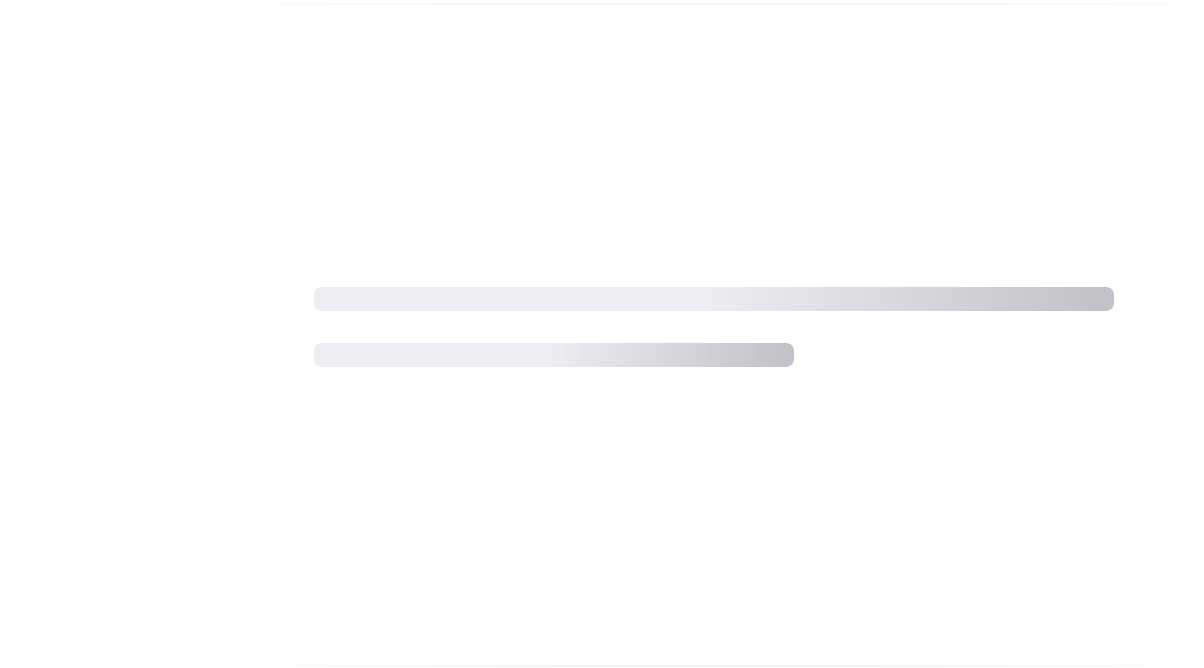
Learning Objectives

By the end of this lesson, you will be able to:

1. **Apply** complex number mathematics for elegant 2D rotation calculations using $e^{i\theta}$
2. **Calculate** forward kinematics from joint angles to end-effector position
3. **Solve** inverse kinematics with elbow-up and elbow-down configurations

4. **Compose** 2D rotation matrices for orientation control
5. **Program** pick-and-place operations with sub-millimeter accuracy

Real-World Engineering Challenge: Planar Motion Control



Planar **mot**ion **syst**ems **sp**an **robo**tics, **manufa**cturing, and **auton**omous **navig**ation. **Fr**om **preci**sion **asse**mbly **rob**ots **plac**ing **elect**ronic **compo**nents with **micro**meter **accu**racy, to **CN**C **mach**ines **cutt**ing **comp**lex **cur**ves in **compo**site **mater**ials, to **mob**ile **rob**ots **navig**ating **wareh**ouses - **al**l **requ**ire **prec**ise **mathem**atical **cont**rol of 2D **posi**tion and **orient**ation.

Representative Systems

Planar Motion Applications:

- **Assembly Robots** (SCARA, delta robots) - pick-and-place with orientation control
- **CNC Machines** (routers, plasma cutters) - tool path generation and offset compensation

- **Mobile Robots** (AGVs, warehouse robots) - trajectory planning and collision avoidance
- **Coordinate Measuring Machines** (CMMs) - precision positioning for inspection
- **Packaging Systems** (conveyors, palletizers) - coordinated multi-axis motion
- **Semiconductor Manufacturing** (wafer handlers) - ultra-precision positioning

The Planar Transformation Challenge

These systems require precise control of:

> **Critical Motion Requirements**
>
> - **Position accuracy** from ±0.01 mm (electronics) to ±1 mm (mobile robots)
> - **Orientation control** for proper tool/part alignment and heading management
> - **Trajectory optimization** for minimum cycle time with smooth, collision-free motion
> - **Workspace utilization** maximizing reachable area within joint/physical limits
> - **Coordinate frame management** relating multiple work zones and reference frames

> **Engineering Question:** How do we mathematically represent and program complex 2D trajectories that combine rotations, translations, and orientations in a systematic, precise manner that works across diverse applications?

Why Planar Mathematics Matters

Consequences of Poor Mathematical Foundation:

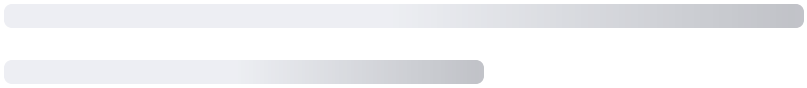
- **Programming errors** leading to collisions, missed targets, or navigation failures
- **Inefficient trajectories** with unnecessary motion and longer cycle times
- **Orientation errors** causing assembly failures or incorrect tool angles
- **Workspace limitations** from inadequate mathematical modeling

- **Calibration difficulties** without systematic transformation framework

Benefits of Systematic Planar Analysis:

- **Precise motion programming** using mathematical transformation tools
- **Optimal trajectory planning** through systematic matrix composition
- **Consistent coordinate frame management** across different work areas
- **Scalable programming methods** applicable to complex 3D systems
- **Unified approach** for diverse applications (robotics, CNC, navigation)

Fundamental Theory: Planar Transformation Mathematics



Complex Number Representation of Planar Motion

Complex **numb**ers **prov**ide an **eleg**ant **mathem**atical **frame**work for **repres**enting 2D **rotat**ions and **transl**ations. **Eve**ry **poi**nt in the **pla**ne can be **repres**ented as a **comp**lex **num**ber z = x + iy, and **transf**ormations **bec**ome **sim**ple **algeb**raic **opera**tions.

🔢 Complex Number Planar Point

$$z = x + iy = re^{i\theta}$$

Where:

- z = Complex number representing 2D point
- x, y = Cartesian coordinates (real and imaginary parts)
- r = Distance from origin (magnitude)
- θ = Angle from positive real axis (phase)

Physical Meaning: Every 2D point corresponds to a unique complex number, enabling algebraic manipulation of geometric transformations.

Pure Rotation | Pure Translation | Combined Motion

🔄 Complex Rotation Operator

Rotation by angle α: $z' = z \cdot e^{i\alpha} = z \cdot (\cos\alpha + i\sin\alpha)$

Matrix form: $\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$

Physical Meaning: Multiplying by $e^{i\alpha}$ rotates any point counterclockwise by angle α about the origin.

Matrix Representation of 2D Rotations

🔄 2D Rotation Matrix

Rotation by angle θ about the origin:

$$R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

Applied to a point: $\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$

Expanded form:

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

Physical Meaning: This matrix rotates any 2D vector counterclockwise by angle θ about the origin.

Rotation Matrix Properties

1. **Orthogonal matrix:** $R^T R = I$ (transpose equals inverse)
2. **Determinant:** $\det(R) = 1$ (preserves area and orientation)
3. **Composition:** $R(\alpha) \cdot R(\beta) = R(\alpha + \beta)$ (rotations add)
4. **Inverse:** $R^{-1}(\theta) = R(-\theta) = R^T(\theta)$ (reverse rotation)

Application 1: PCB Assembly Pick-and-Place Programming (Electronics Manufacturing)

A SCARA robot must pick an SMD (Surface Mount Device) component from a component tray and place it on a PCB with precise position and orientation control.

System Problem Statement

- **Configuration:** SCARA robot with 2 rotary joints + vertical prismatic + wrist rotation
- **Task:** Transfer component from tray to PCB with rotation during transfer



What we need to calculate:

1. **Forward kinematics** using complex number mathematics
2. **Inverse kinematics** with elbow-up and elbow-down configurations
3. **Rotation composition** using complex multiplication for orientation control
4. **Placement verification** for sub-millimeter accuracy

Key Question: What are the joint angles needed to achieve ±0.05mm position accuracy, and how do we use complex numbers to verify the solution?

Equivalent System Model

SCARA Robot Configuration

Given:

- **Robot parameters:** $L_1 = 350$ mm, $L_2 = 250$ mm
- **Component tray position:** (450, 200) mm, orientation 30°
- **PCB position:** (-200, 300) mm, orientation -45°
- **Required component rotation:** 75° during transfer
- **Placement accuracy:** ±0.05 mm position, ±0.1° orientation

Step 1: Forward Kinematics - Complex Number Approach

> **Click to reveal forward kinematics calculations**

Step 2: Inverse Kinematics - Elbow Configurations

> **Click to reveal inverse kinematics calculations**

Step 3: Component Rotation Using Complex Numbers

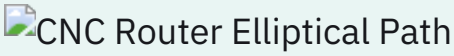
> **Click to reveal rotation composition calculations**

Application 2: CNC Router Elliptical Tool Path Generation (Manufacturing)

A CNC router must cut an elliptical slot in a composite panel at a 25° angle with a precise tool offset to account for cutter diameter.

System Problem Statement

- **Configuration:** 3-axis CNC router with 12mm diameter end mill
- **Task:** Generate G-code waypoints for rotated ellipse with tool offset compensation



What we need to calculate:

1. **Parametric ellipse** using complex number representation
2. **Rotation by 25°** using complex multiplication $z' = z \cdot e^{i25^\circ}$
3. **Tool offset** using normal vectors (cutter radius compensation)
4. **Entry spiral** using Archimedean spiral mathematics
5. **Waypoint discretization** at 16 equally-spaced points

Key Question: How do we use complex numbers to generate precise CNC waypoints for a rotated ellipse with proper tool offset?

Equivalent System Model



Given:

- **Ellipse parameters:** a = 80 mm (semi-major axis), b = 50 mm (semi-minor axis)
- **Center position:** (150, 100) mm
- **Rotation angle:** α = 25° (counterclockwise)

- **Tool diameter:** d = 12 mm (radius r = 6 mm)
- **Entry spiral:** 2 revolutions, 0.5 mm pitch
- **Number of waypoints:** 16 (every 22.5°)
- **Tolerance:** ±0.02 mm position accuracy

Step 1: Parametric Ellipse in Complex Form

> **Click to reveal ellipse parametrization**

Step 2: Tool Offset Using Normal Vectors

> **Click to reveal tool offset calculations**

Step 3: 16-Point Waypoint Discretization

> **Click to reveal waypoint calculations**

Step 4: Entry Spiral Path (Archimedean Spiral)

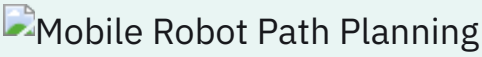
> **Click to reveal spiral entry calculations**

Application 3: Mobile Robot Navigation Path Planning (Autonomous Systems)

An autonomous mobile robot must navigate from a starting position to a goal position while avoiding a circular obstacle using arc trajectories.

System Problem Statement

- **Configuration:** Differential drive mobile robot (two independently driven wheels)
- **Task:** Calculate collision-free path using complex number rotations and translations



What we need to calculate:

1. **Forward kinematics** for differential drive using complex numbers
2. **Arc trajectory** from rotation center using polar representation
3. **Collision detection** using complex magnitude (distance calculation)
4. **Path waypoints** with positions and orientations every 0.5m
5. **Velocity profile** for smooth motion control

Key Question: How do we use complex numbers to plan smooth arc trajectories and verify collision-free paths?

Equivalent System Model



Given:

- **Starting pose:** $(x_0, y_0, \theta_0) = (0, 0, 0^\circ)$
- **Goal pose:** $(x_g, y_g, \theta_g) = (8, 6, 45^\circ)$ meters

- **Obstacle:** Circle centered at $(4, 2)$ m with radius $r_{obs} = 1.5$ m
- **Safety clearance:** 0.5 m minimum distance from obstacle
- **Robot dimensions:** Wheelbase L = 0.6 m, width W = 0.5 m
- **Waypoint spacing:** 0.5 m along path

Step 1: Path Strategy - Three-Segment Arc Trajectory

> **Click to reveal path planning strategy**

Step 2: Arc Trajectory Mathematics Using Complex Numbers

> **Click to reveal arc trajectory calculations**

Step 3: Collision Detection Using Complex Distance

> **Click to reveal collision detection calculations**

Step 4: Waypoint Generation with 0.5m Spacing

> **Click to reveal waypoint generation**

Summary and Next Steps

In this lesson, you mastered planar transformation mathematics through three applications: SCARA robot kinematics, CNC elliptical toolpaths, and mobile robot navigation.

Key Skills Developed:

1. **Apply** complex number mathematics: $z = x + iy = re^{i\theta}$ for elegant 2D transformations
2. **Compose** rotations and translations: $z' = z \cdot e^{i\alpha} + \Delta z$
3. **Calculate** forward/inverse kinematics, parametric curves, and collision-free paths
4. **Generate** waypoints with tool offset compensation and orientation control

Critical Formulas:

- **Complex rotation:** $z' = z \cdot e^{i\theta}$ where $e^{i\theta} = \cos\theta + i\sin\theta$
- **2D rotation matrix:** $R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$
- **Distance:** $d = |z_1 - z_2|$ (Euclidean distance in complex form)

Key Insights:

- Rotation composition: e^(iα) · e^(iβ) = e^(i(α+β))
- Inverse kinematics yields multiple solutions (elbow-up/down)
- Normal vectors enable tool offset compensation

Coming Next: Lesson 3 extends to 3D space with homogeneous transformation matrices, Euler angles, and 6-DOF spatial robot control.

Comments

Loading comments...

© 2021-2026 SiliconWit®. All rights reserved.

Previous
Lesson 1: Kinematic Joints and Degrees of Freedom

Next
Lesson 3: 3D Rotations and Transformations

Privacy · Cookie settings