Introduction to Kinematic Synthesis & Geometric Constraint Programming (GCP)

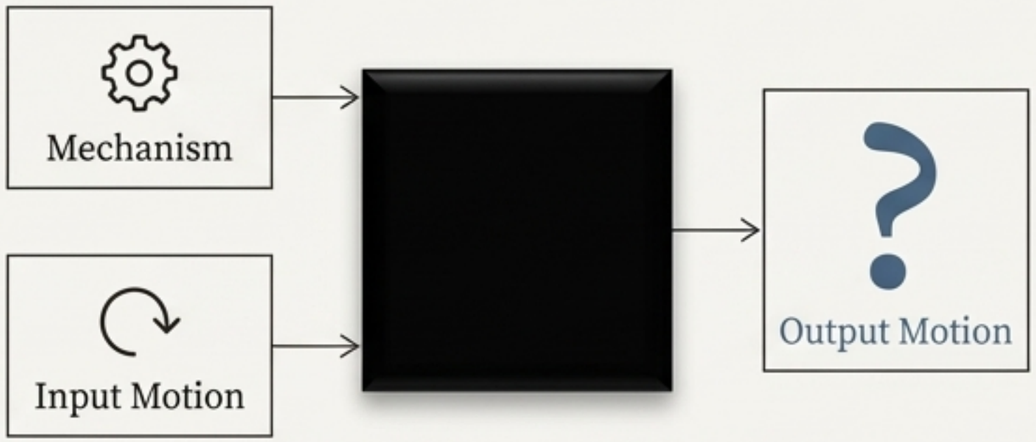
A Modern Approach to Mechanism Design

ME 3751: Kinematics and Mechanism Design

The Two Sides of Kinematics: Analysis vs. Synthesis

Analysis (The “What if?”)

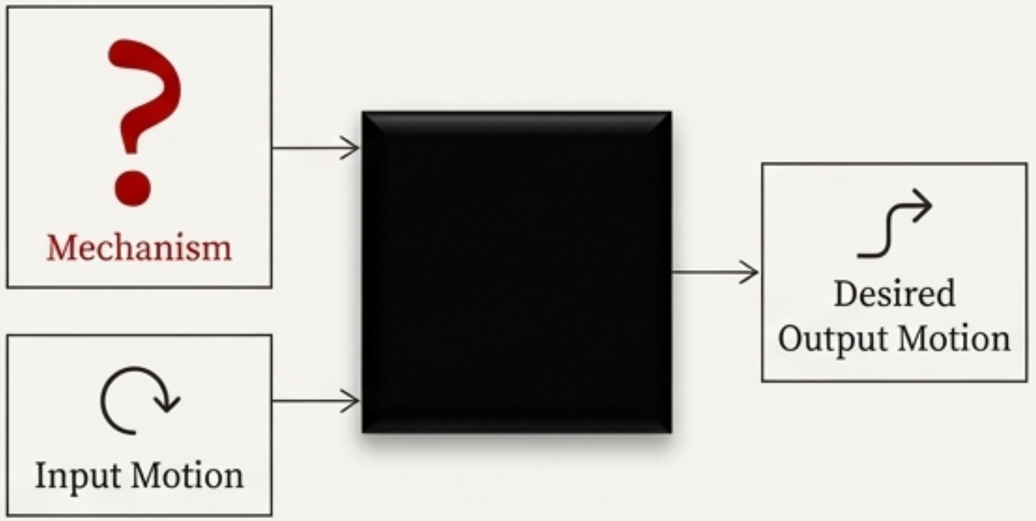
Given a mechanism, find its output motion. This is about evaluating a known design.



Typically has a unique, calculable solution.

Synthesis (The “How to?”)

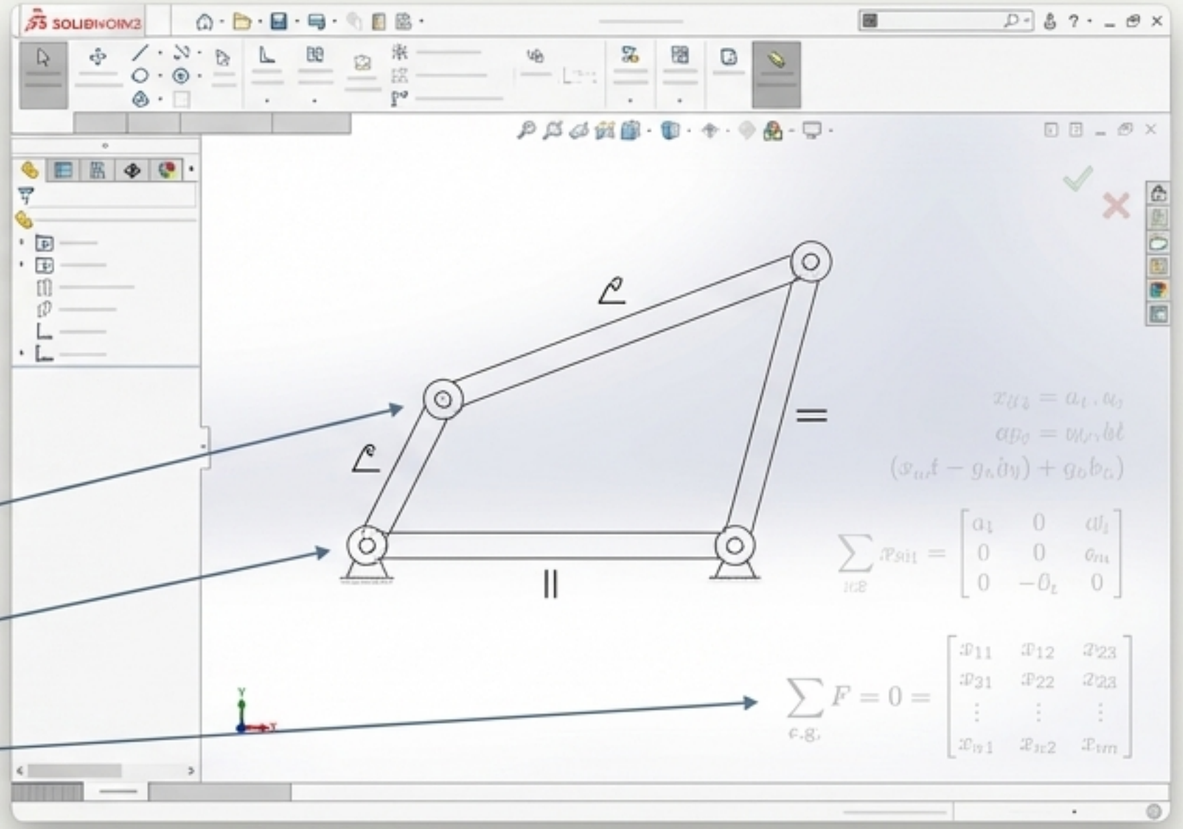
Given a desired output motion, find the mechanism(s) that can produce it. This is the essence of design.



Requires innovation and creative thinking. Often has multiple possible solutions that must be evaluated.

A Modern Solution: Geometric Constraint Programming (GCP)

Core Idea: GCP uses the 2D sketching environment of parametric CAD software (like SolidWorks) to solve complex kinematic synthesis problems graphically.



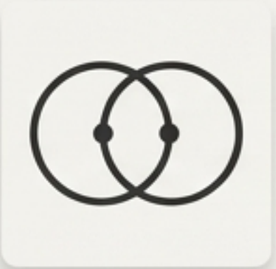
How it Works:

1. “**Geometry** as **Variables**”
2. “**Constraints** as **Equations**”
3. “**Powerful Solvers**”

The Big Picture: We are turning a CAD sketch into a powerful computational tool for mechanism design.

The GCP Toolbox: Essential Geometric Constraints

These are the fundamental rules you apply to your sketch to define the mechanism's behavior. SolidWorks calls them "Relations".

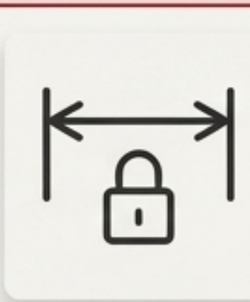


Merge/Coincident: Forces two points to coincide (for revolute joints) or a point to lie on a line (for a pin-in-slot joint).

Position Lock (Fix): Anchors a point or line to the ground, defining the mechanism's frame.

Equality: Constrains two or more lines or arcs to be equal in length or radius.

Parallelism / Perpendicularity: Enforces orientation between lines.

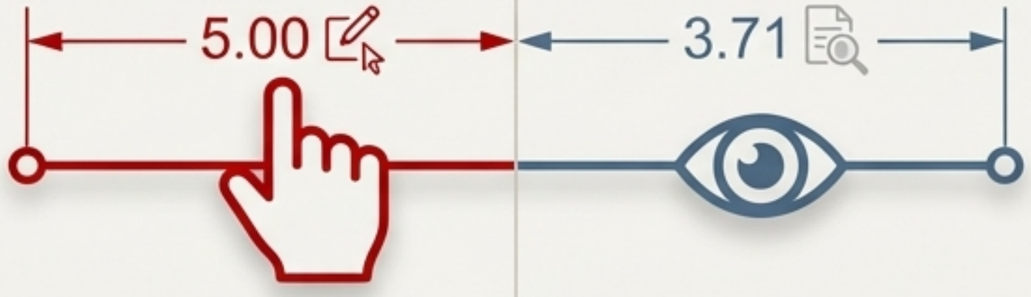


Dimension Lock: Constrains a linear or angular dimension to a fixed value. This is the **most critical** constraint for defining link lengths and angles.

The Engine of GCP: Driving vs. Driven Dimensions

Driving Dimensions

- **Function:** A constraint that *sets* a value.
- **Role:** Defines the independent variables and known parameters of your design problem (e.g., length of the base, required input angle).
- **Behavior:** Changing a driving dimension forces the solver to re-calculate the entire geometry.



Driven Dimensions

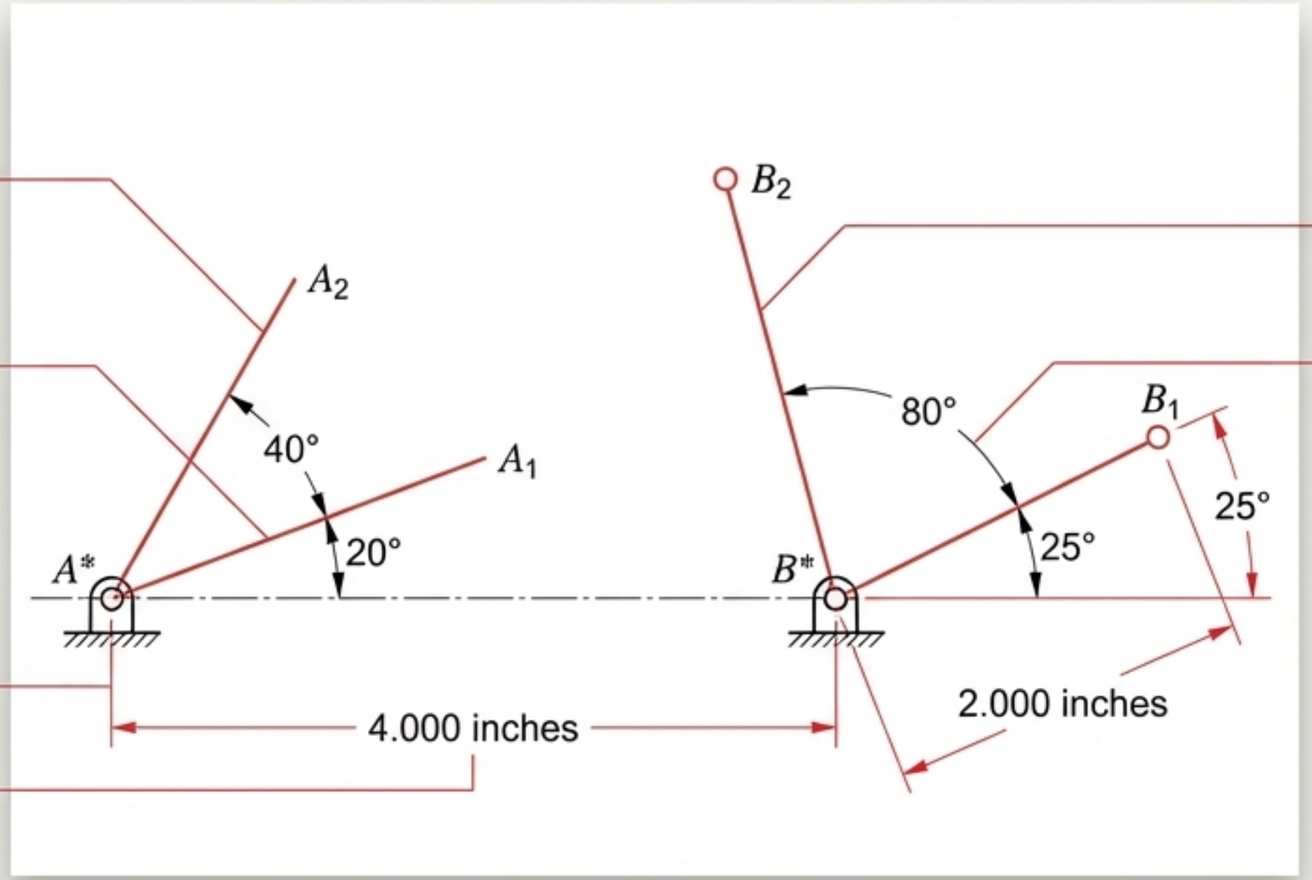
- **Function:** A measurement that *reports* a value. It is NOT a constraint.
- **Role:** Displays the results or dependent variables of your solution (e.g., the calculated length of a coupler link, the current transmission angle).
- **Behavior:** Its value updates automatically as the geometry changes, but it does not control the geometry.

Key Takeaway: Mastering this distinction is how you create a ‘graphical program’ that solves for unknowns and allows for interactive design exploration.

Masterclass: Synthesizing a Double-Rocker Mechanism

Design Specifications (The "Knowns")

- Input link moves through **40°**, starting from **20°.**
- Output link moves through **80°**, starting from **25°.**
- Distance between fixed pivots (base length) is **4 inches.**
- Length of the output link is **2 inches.**



The Goal (The "Unknowns")

- Determine the required length of the **input link.** ?
- Determine the required length of the **coupler link.** ?

Step 1: Defining the Problem Kinematically

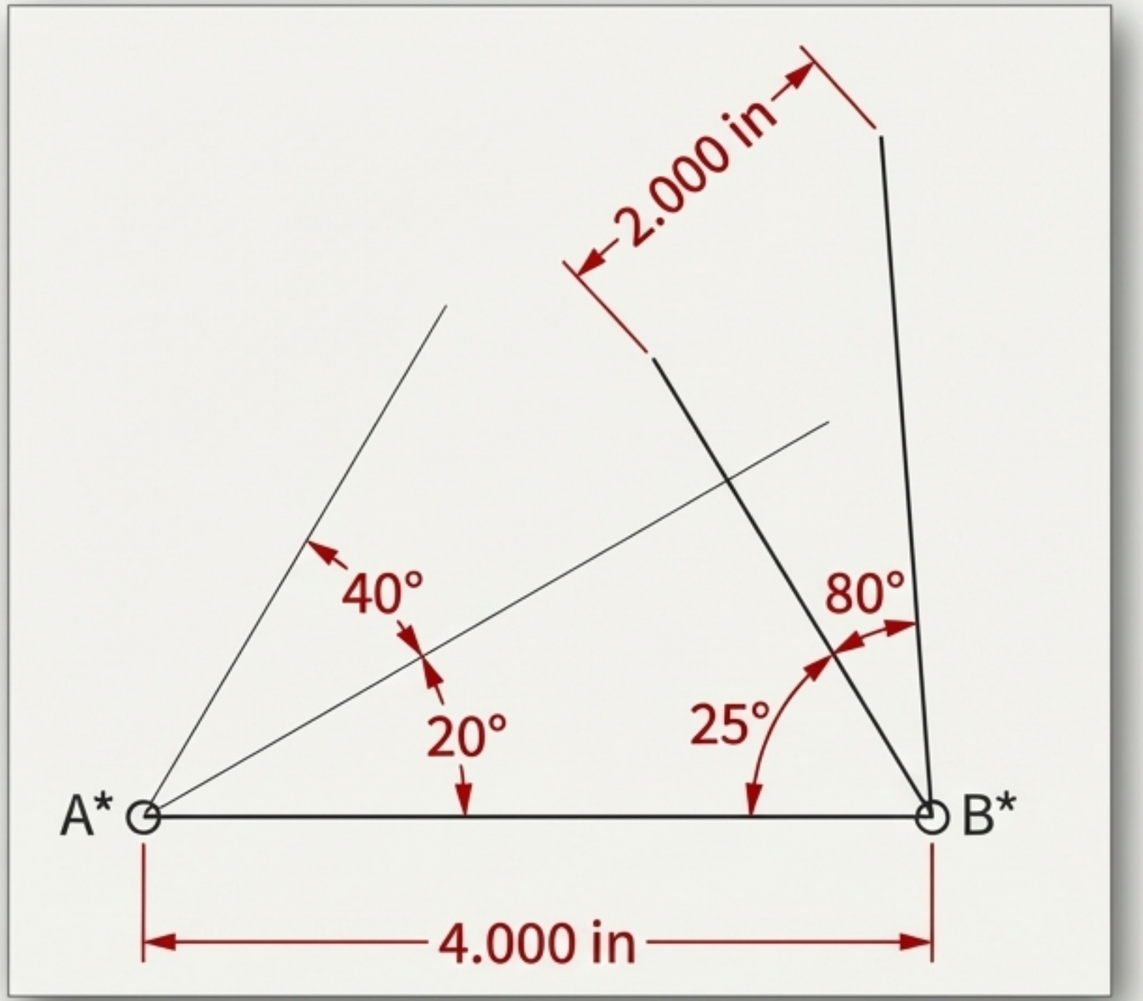
Action

In a 2D sketch, lay out all the known information from the problem statement using driving dimensions.

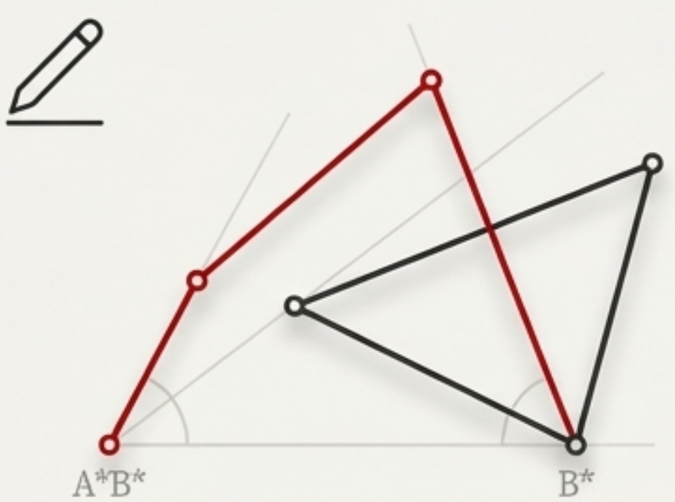
Construction Details

1. Draw a horizontal line and fix its length to **4.000 in**. This is the ground link (A* to B*).
2. At pivot A*, draw lines representing the input link's start (20°) and end (20° + 40° = 60°) positions.
3. At pivot B*, draw lines representing the output link's start (25°) and end (25° + 80° = 105°) positions.
4. Constrain the length of the output link lines to **2.000 in**.

Status: This sketch now geometrically represents the complete motion envelope required by the design.

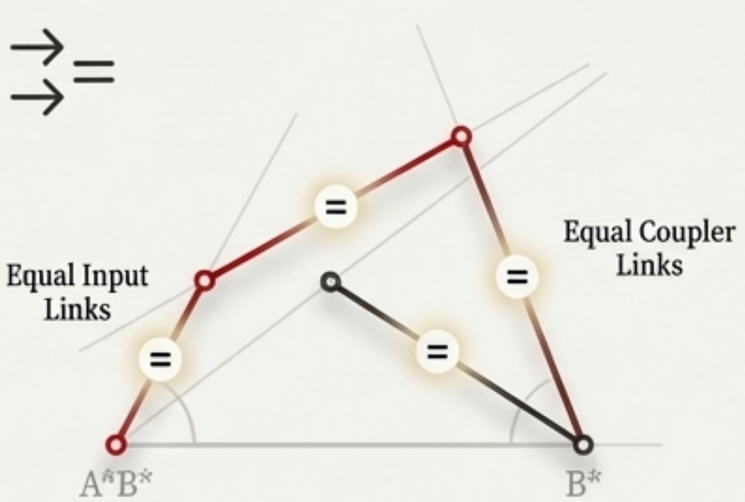


Step 2: The Geometric Construction



Draw Arbitrary Linkages

Action: Draw two separate, arbitrary four-bar linkages representing the start (position 1) and end (position 2) states of the mechanism.

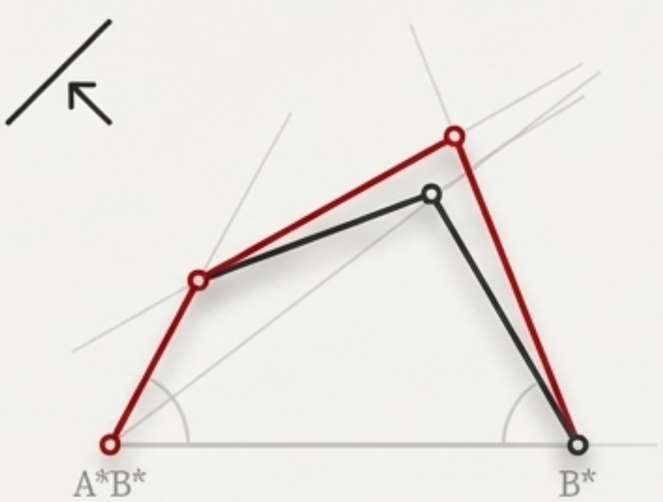


Apply Equality Constraints

Applying the Constraints (The Logic): Because both drawings must represent the *same* linkage, we enforce the following equality constraints:

1. **Equal Input Links:** The length of the input link in position 1 must equal the length in position 2.

2. **Equal Coupler Links:** The length of the coupler in position 1 must equal the length in position 2.



Apply Collinearity

3. **Collinearity:** Snap the links of each sketched linkage to be collinear with the guideline angles from Step 1.

The Result: As these constraints are applied, the CAD solver works in the background, forcing the initially arbitrary linkages to conform to a single geometric solution that satisfies all conditions simultaneously.

Step 3: The Solution Emerges

Action

After all constraints are applied, the sketch is fully defined. The geometry is now solved.

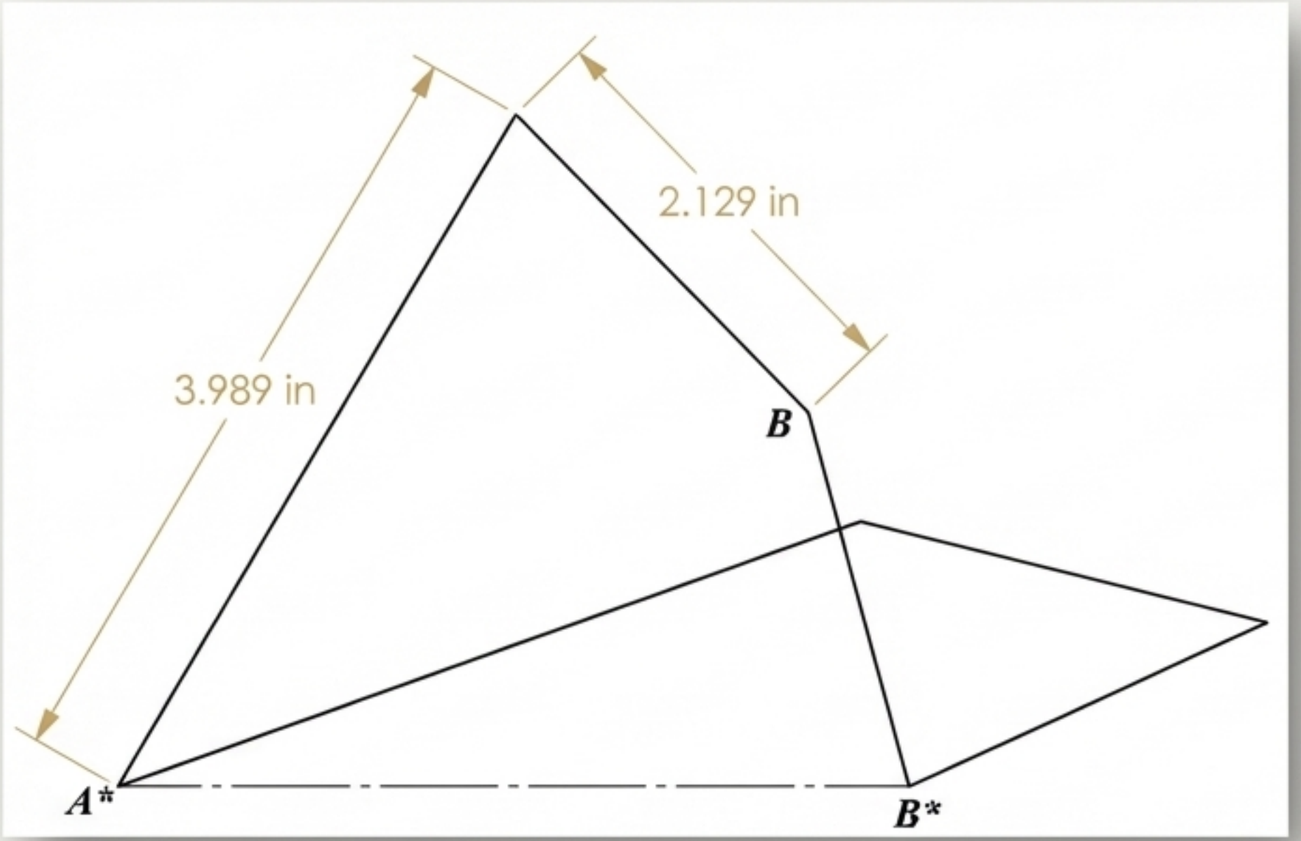
Extracting the Solution

Use driven dimensions to measure the lengths of the now-defined input and coupler links.

The Answer

Input Link Length (A*A): 3.989 in

Coupler Link Length (AB): 2.129 in



Key Insight: We have solved a complex, non-linear system of equations without writing any of them down. The geometric construction *is* the program.

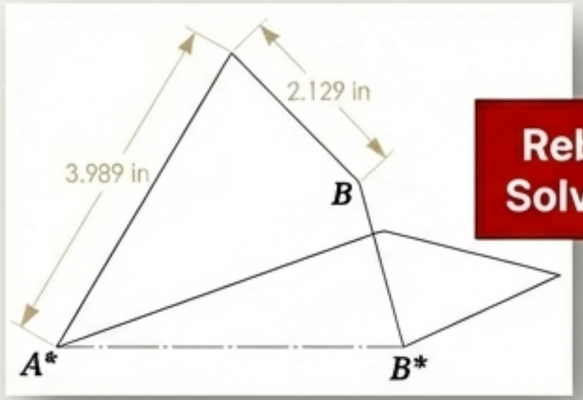
From Static Solution to a Dynamic Model

Action:

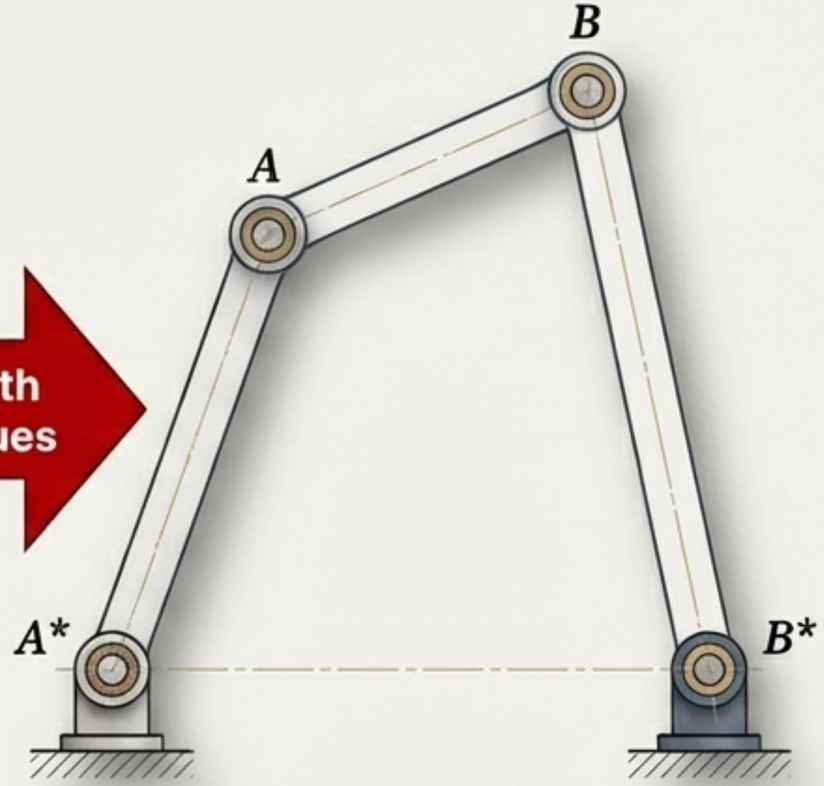
Now that we know the link lengths, we create a *new* sketch of the final linkage.

Construction:

1. Draw a four-bar linkage (ground, input, coupler, output).
2. Apply **driving dimensions** using the solved lengths:
   - Ground = 4.000 in
   - Input = 3.989 in
   - Coupler = 2.129 in
   - Output = 2.000 in
3. Fix one of the ground pivots (e.g., A*).



Rebuild with Solved Values



The Result: A single, fully-defined four-bar linkage model. Because it is constrained by length but not by a specific angle, the input link is free to rotate, and the mechanism can be animated.

The Payoff: A Graphical Program for Design Exploration

- **Core Concept:** The construction sketch from Step 2 is now a reusable “graphical program.” Any of the initial **driving dimensions** can be modified, and the solver will instantly find a new solution.
- **The Power:** This allows for rapid trial-and-error and optimization without needing to resolve complex equations manually for each change.

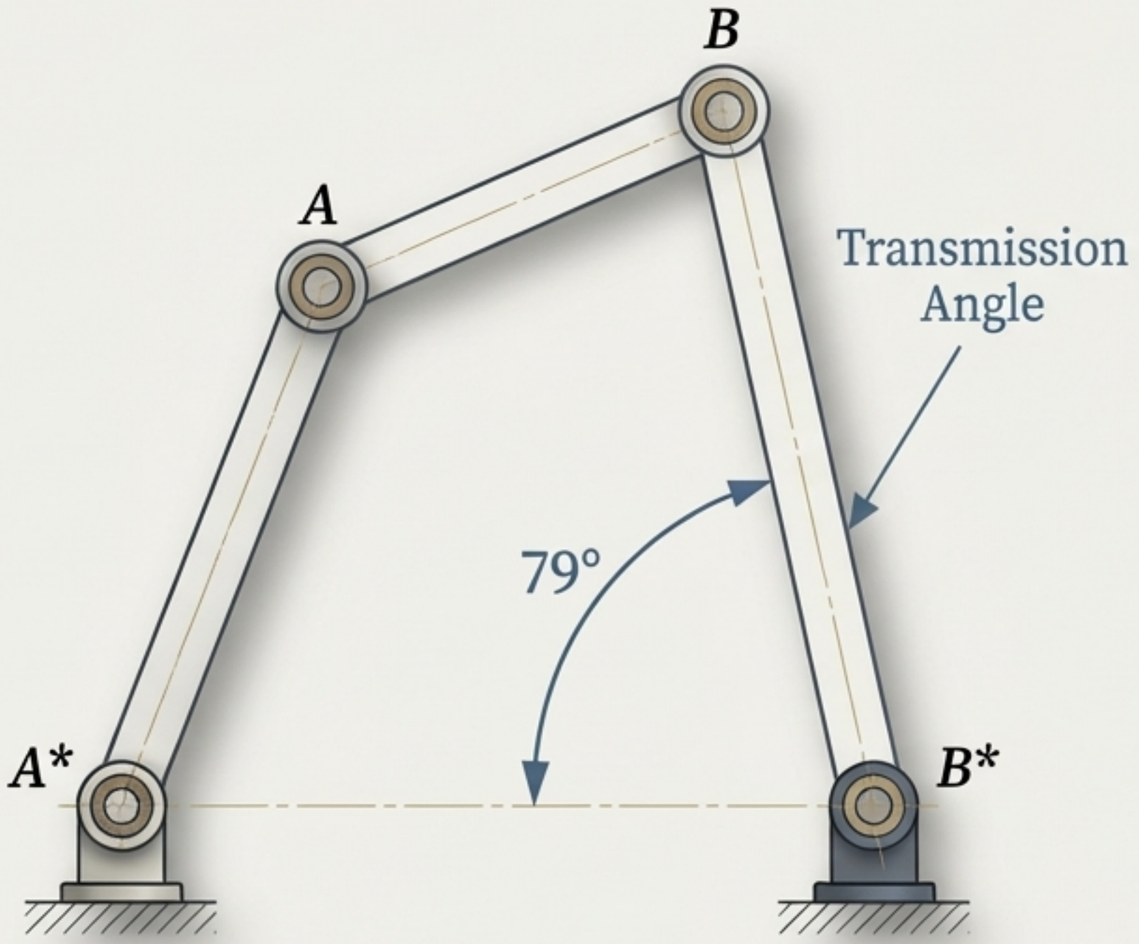
“What if the output link was **1.000 in** long?”

“What if the base was **3.000 in**?”

“What if the start angle was **10°**?”

“What if the input range was **60°**?”

Beyond Geometry: Evaluating Design Quality



The Question: Is the linkage a good design? One key metric is the transmission angle.

Transmission Angle: The angle between the output link and the coupler. It indicates how effectively force is transmitted through the linkage.

- Ideal: As close to 90° as possible.
- Problematic: Angles near 0° or 180° cause the mechanism to lock up or bind.

Implementation in GCP:

1. In the dynamic model, add an angular driven dimension to measure the transmission angle.
2. As you drag the input link through its range of motion, this dimension will update in real-time.

Result for our Design: For this specific solution, the transmission angle varies between 39° and 151°.

GCP: Your New Framework for Mechanism Synthesis

The Workflow Recap

1

Define

Clearly state your design problem with knowns (driving dimensions) and unknowns.

→

2

Construct

Sketch the mechanism in its key positions (e.g., start and end).

→

3

Constrain

Apply equality constraints to link the different positions into a single coherent system.

→

4

Solve

Let the CAD solver find the geometry. Measure the unknowns using driven dimensions.

→

5

Model & Analyze

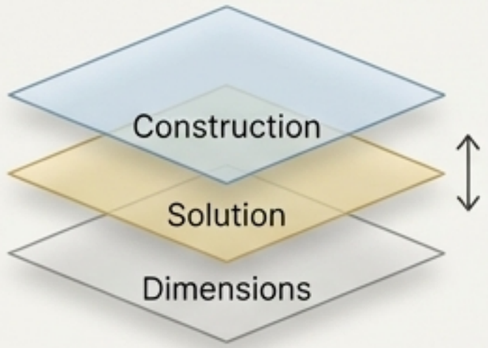
Build a final, movable model to verify motion and analyze performance metrics like the transmission angle.

Core Principle

By thinking in terms of geometric constraints, you can leverage powerful computational tools to solve complex design problems intuitively and efficiently.

Best Practices for Effective GCP

Stay Organized with Layers

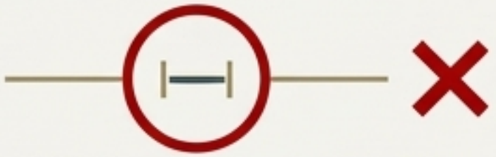


Isolate different parts of your construction onto separate layers that can be toggled for visibility.

Example Layers: 'InputDimensions', 'SolutionConstruction', 'FinalLinkage', 'DrivenDimensions'.

This prevents clutter and makes your graphical program much easier to read and debug.

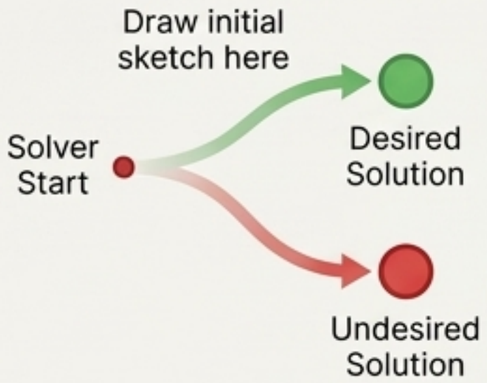
Watch for Unintended Constraints



CAD programs often "snap" to constraints like horizontal, vertical, or perpendicular automatically.

If your sketch becomes over-constrained or behaves unexpectedly, look for and delete these unwanted automatic constraints.

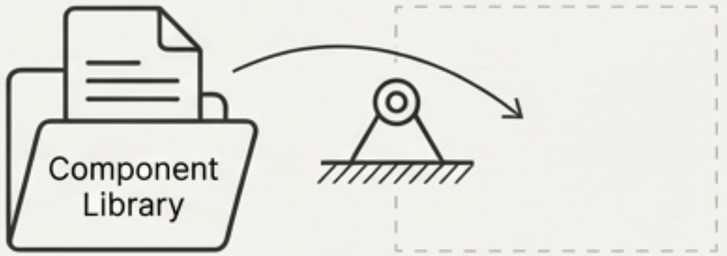
Start Close to the Solution



For highly non-linear problems with multiple solutions, the solver might find an undesired one.

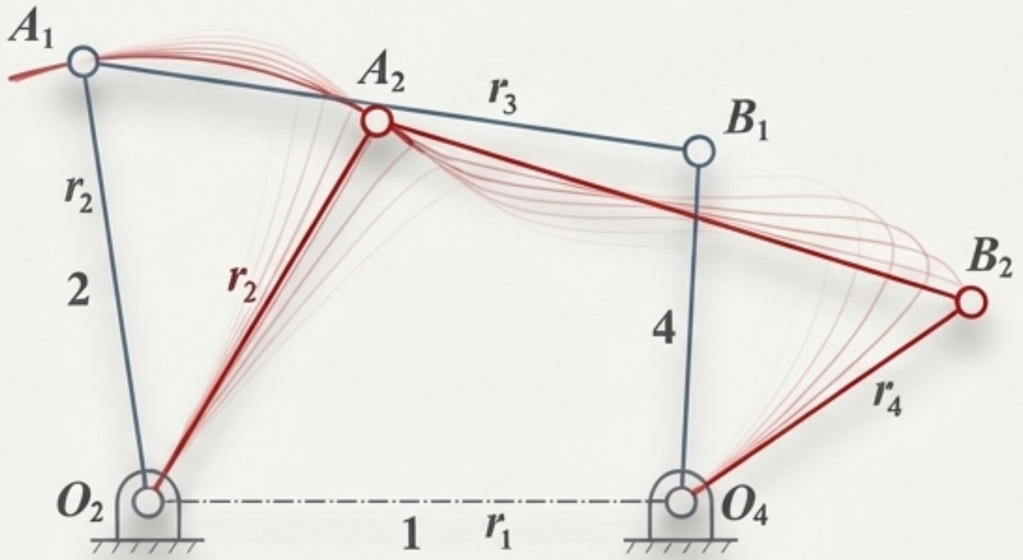
Drawing your initial "arbitrary" linkages reasonably close to what you expect the final solution to look like can help guide the solver to the correct answer.

Build Reusable Components



For common elements like ground pivots or slider lines, create them in a separate file and copy/paste them into your designs to save time and ensure consistency.

You’re Not Just Drawing a Linkage—
You’re Building a Design Engine



The Old Way

- Calculate a single solution for a single set of inputs. If the inputs change, you start over.

The GCP Way

- Create a robust, constrained “graphical program” where the relationships between components are defined.

The Result: A **flexible system** that **instantly provides new solutions** as you explore the design space by changing input variables. This is the foundation of modern, **parametric mechanism design**.