

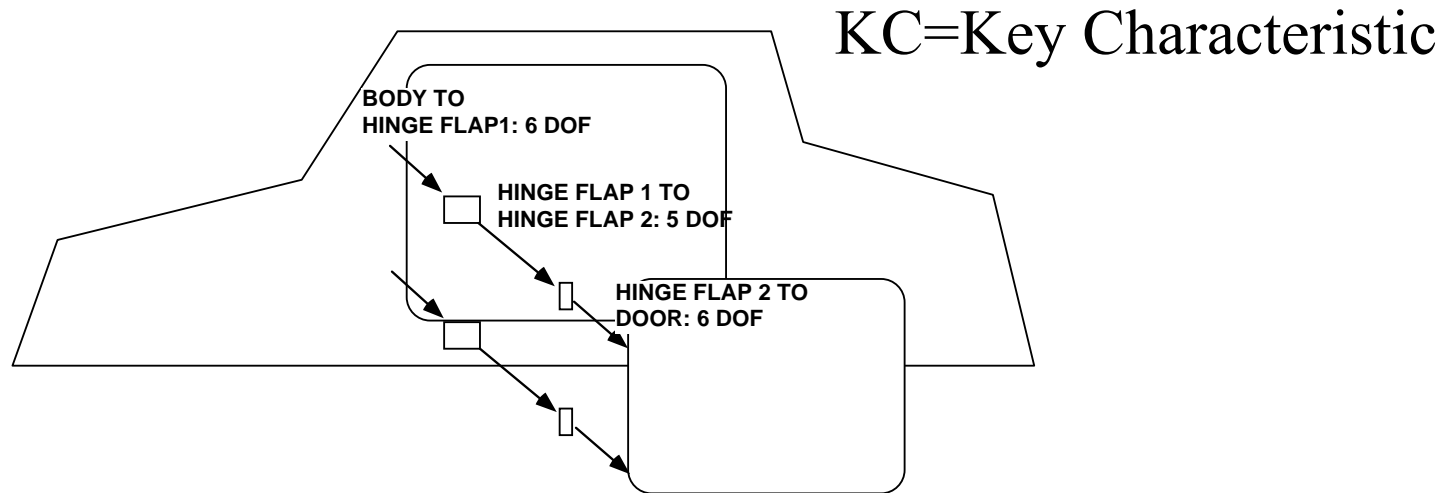
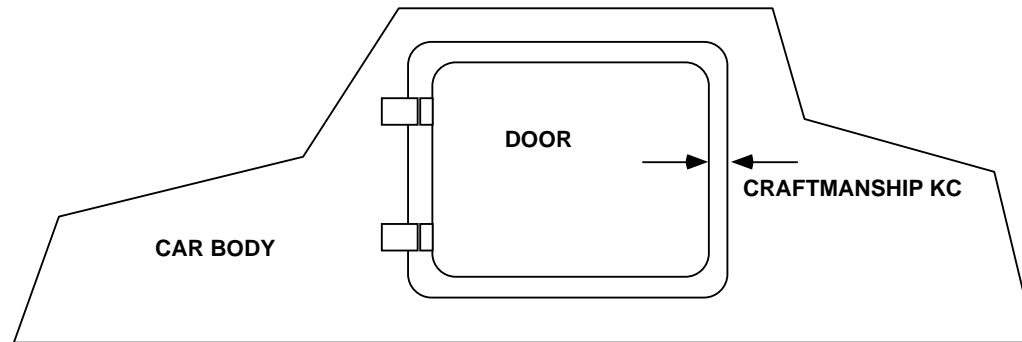
Kinematic Constraint in Assembly

- Topics
 - Assembly as zero-stress location
 - AKA “Exact Constraint,” “Proper Constraint,” or “Kinematic” Design
 - AKA 3-2-1 assembly
 - Assembly features as carriers of constraint, operationalizing the coordinate frames
 - Non-zero-stress assemblies
 - Mathematical analysis of constraint

Assembly = Constraint

- Assembly = removal of dof = application of constraint
- As constraint is applied, degrees of freedom are taken away so that a part gets to where it is supposed to be.
- When parts are where they are supposed to be, the KCs can be delivered, assuming no variation
- This is called the nominal design

Parts Locate Each Other to Deliver Quality at the Customer Level



Definitions of Assemblies

- Whitehead: “An instrument can be regarded as a chain of related parts... any mechanism whose function is directly dependent on the accuracy with which the component parts achieve their required relationships.””The Design and Use of Instruments and Accurate Mechanism,” by Thomas North Whitehead, 1934
- Whitney: “An assembly is a chain of coordinate frames on parts designed to achieve certain dimensional relationships called Key Characteristics between some of the parts or between features on those parts.” “Designing Assemblies,”

Research in Engineering Design, (1999) 11:229-253.

The Three Principles of Statics

- Geometric compatibility
- Force and moment balance
- Stress-strain-temperature relations
- We assume rigid parts, so the 3rd principle does not apply to our work

“Properly” Constrained and Over-constrained Assemblies

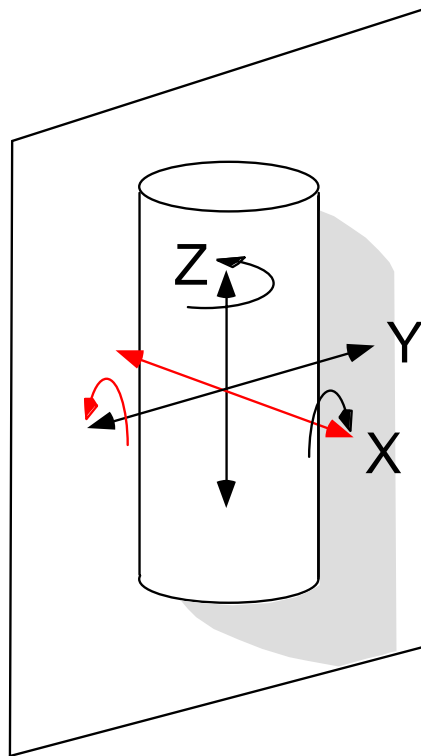
- Assemblies that function by geometric compatibility and force equilibrium alone are called
 - statically determinate
 - “properly” constrained
 - “kinematic” or “semi-kinematic” ~ “3-2-1”
- You “just put them together”
- Assemblies that require stress analysis are called
 - statically indeterminate
 - “over-constrained”
- You can’t “just put them together”
- *Constraint is a property of the nominal design*

Constraint is Accomplished by Surfaces in Contact

The contact permits some dof to move with respect to each other and prevents motion of other dof.

The black ones can move.

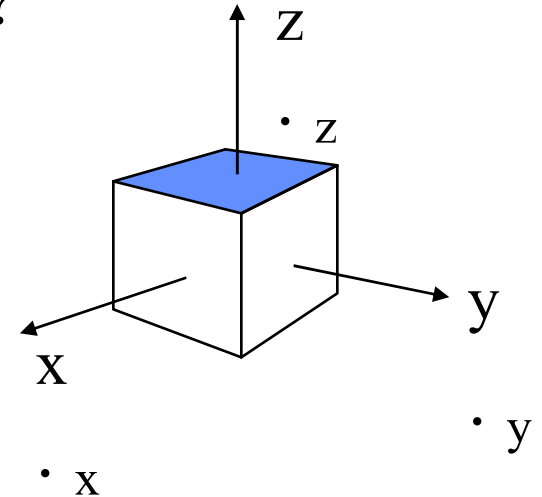
The red ones can't.



Different kinds of surface pairs permit and prevent motion of different dof

Degrees of Freedom

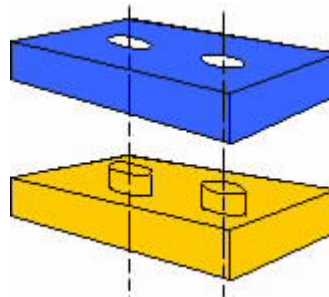
- An object's location in space is completely specified when three translations (X, Y, Z) and three rotations ($\dot{x}, \dot{y}, \dot{z}$) are specified
- How many DOFs are constrained?
 - cube on table (x-y plane)
 - cube at floor-wall interface
 - cube at floor-two walls interface
 - ball on table
 - ball at floor-wall interface
 - round peg in blind round hole
- Think about the constrained ones



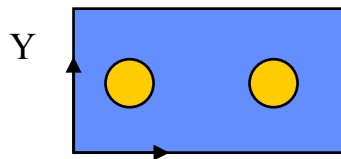
Constraint - 1

- Proper constraint provides **a single value** for each of a body's 6 degrees of freedom
- This is done by establishing surface contacts with surfaces on another part or parts
- If less than 6 dof have definite values, the body is **under-constrained**
- If an attempt is made to provide 2 or more values for a dof, then the body is **over-constrained** because rigid bodies have only 6 dof
- Any extra needed dof must be obtained by deforming the object

Example of Proper and Over Constraint

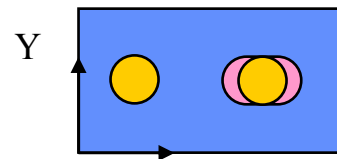


Two pins in holes



X
This is over-constrained
in the X direction

One pin in hole, one pin in slot

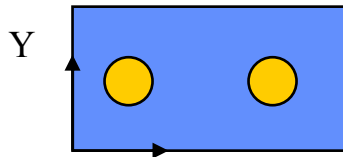


X
This is properly constrained in X

Constraint - 2

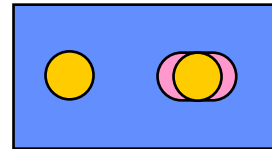
- *Proper* constraint permits an assembly to have unambiguous chains of delivery of KCs

Two pins in holes



X
Which pin determines X
of the blue plate? Can't tell!

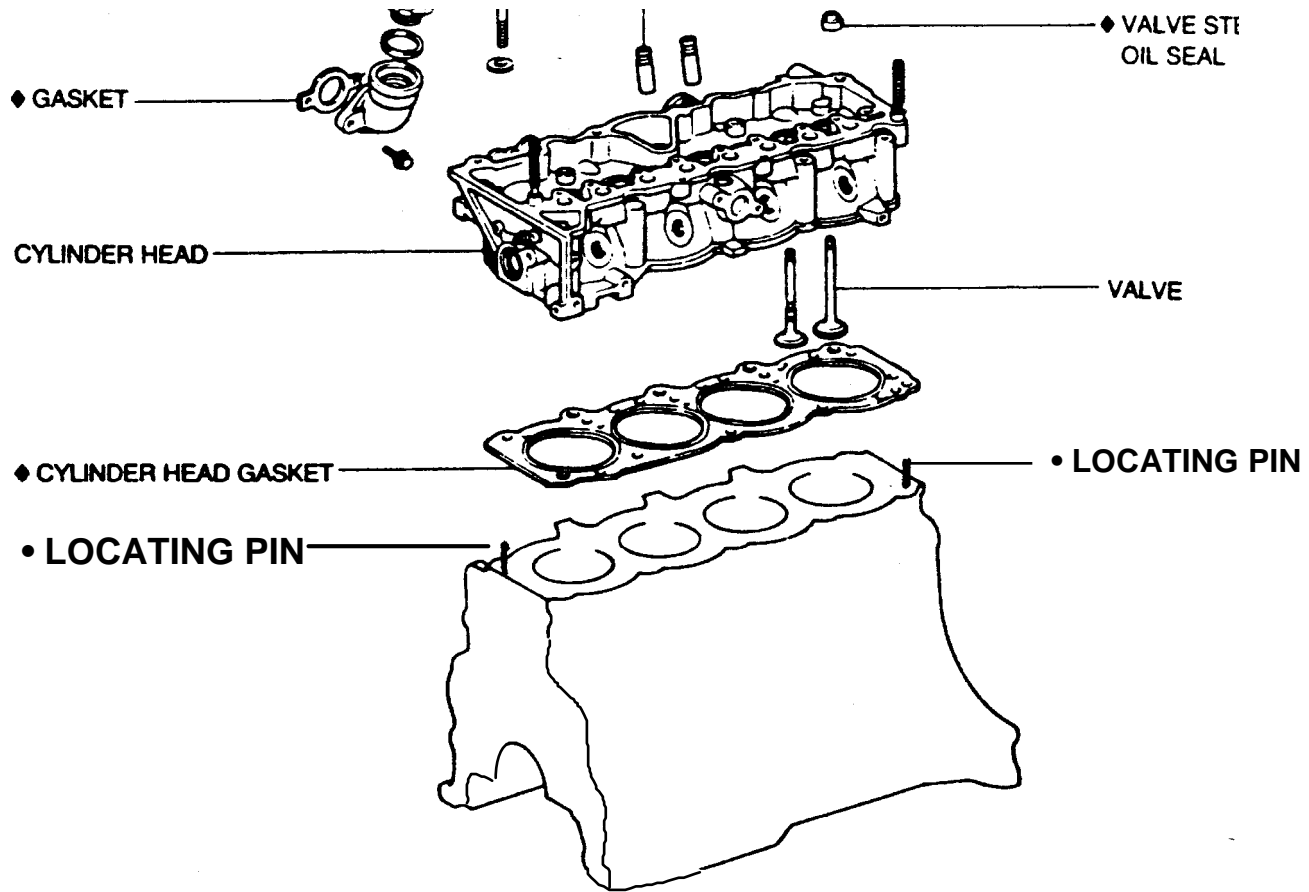
One pin in hole, one pin in slot



The left pin determines X
of the blue plate

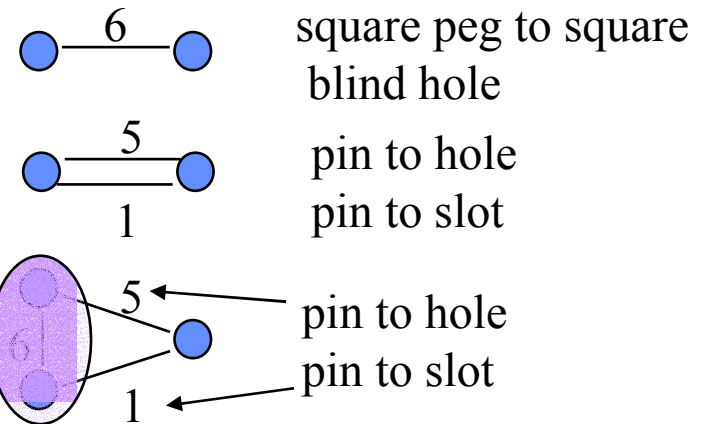
If the X location of the left pin changes, where
will the blue plate go?

Cylinder Head Mate to Block



When Parts are Joined, Degrees of Freedom are Fixed

- Parts join at places called assembly features
- Different features constrain different numbers and kinds of degrees of freedom of the respective parts (symmetrically)
- Parts may join by
 - one pair of features
 - multiple features
 - several parts working together, each with its own features
- When parts mate to *fixtures*, dofs are constrained



Assembly Features Carry Constraint

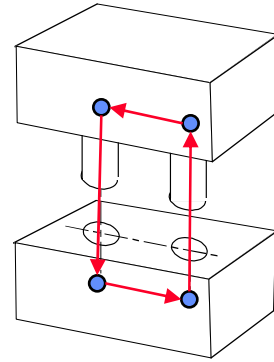
- Kinematic constraint passes from part to part across the feature joints
- The degree of constraint can be calculated using Screw Theory
- Proposed feature designs and KC chains can be examined using Screw Theory to see if they convey the desired amount of constraint and avoid constraint errors

Constraint - 3

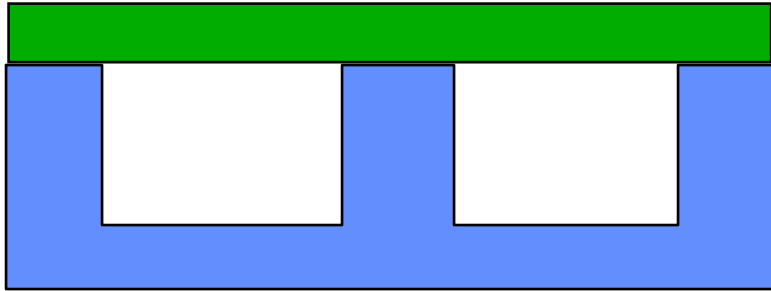
- CAD systems analyze “constraint”
- But CAD systems, developers, and researchers do not mean mechanical constraint as we define it
- They mean geometric location consistency
- Many designs called properly constrained by CAD systems are actually over-constrained
- Different CAD systems do this analysis different ways and can disagree about the same assembly

How CAD Systems Test Constraint

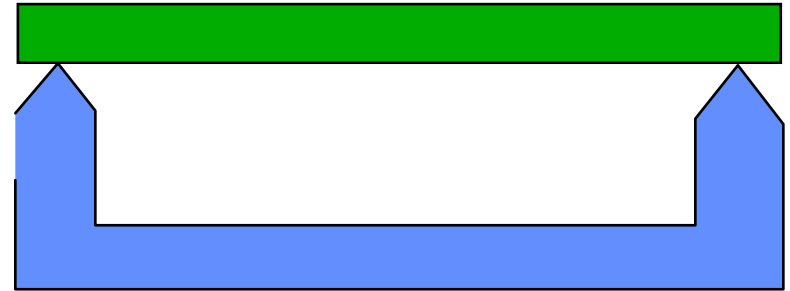
- A closed chain of frames is set up
- A numerical test is done to see if the chain closes
- If, so, the assembly is called “fully constrained”
 - Actually, it should be called “consistent”
- Detailed tests for constraint/consistency problems are done by making small shifts and testing for interference
- Tolerance studies are done the same way
- Analysis requires detailed geometry
- Results depend on how the model was built



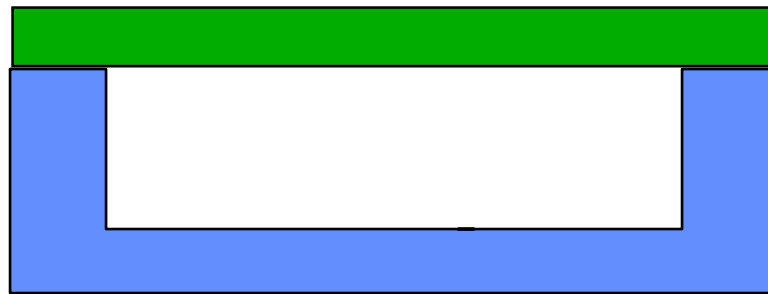
Some Examples



A



B



C

Proper Constraint \approx Zero Stress Assembly

- Kinematic design seeks to determine locations of parts solely or almost solely by means of geometric compatibility
- Locked-in stresses are kept so low that they are negligible
- Exact constraint design is equivalent to “3-2-1” assembly
- In effect, in a kinematic assembly, the parts act as fixtures for each other
- You just put them together

Non-zero Stress Assemblies Can Be OK

- Three-leg stool rests firmly and is fully constrained
- Four-leg stool gives the security of an extra leg and wider footprint but will not rest firmly unless the legs are elastic enough to deform until all four are in contact, or there are screw adjustments
- Other examples
 - shrink fit of wheel on shaft
 - preloaded angular contact ball bearing pairs
 - pre-stressed concrete and case-hardened armor plate
 - planetary gear trains

Constraint Mistakes Happen

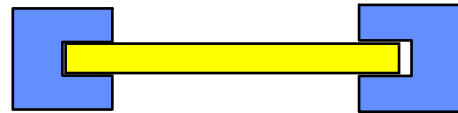
- Designers make constraint mistakes
 - Mostly they make over-constraint mistakes
 - You can see under-constraint mistakes because they permit unwanted motions
 - You can't see the stresses caused by over-constraints
- It would be nice to have a test that looks for over/under constraint

How Airplanes are Built

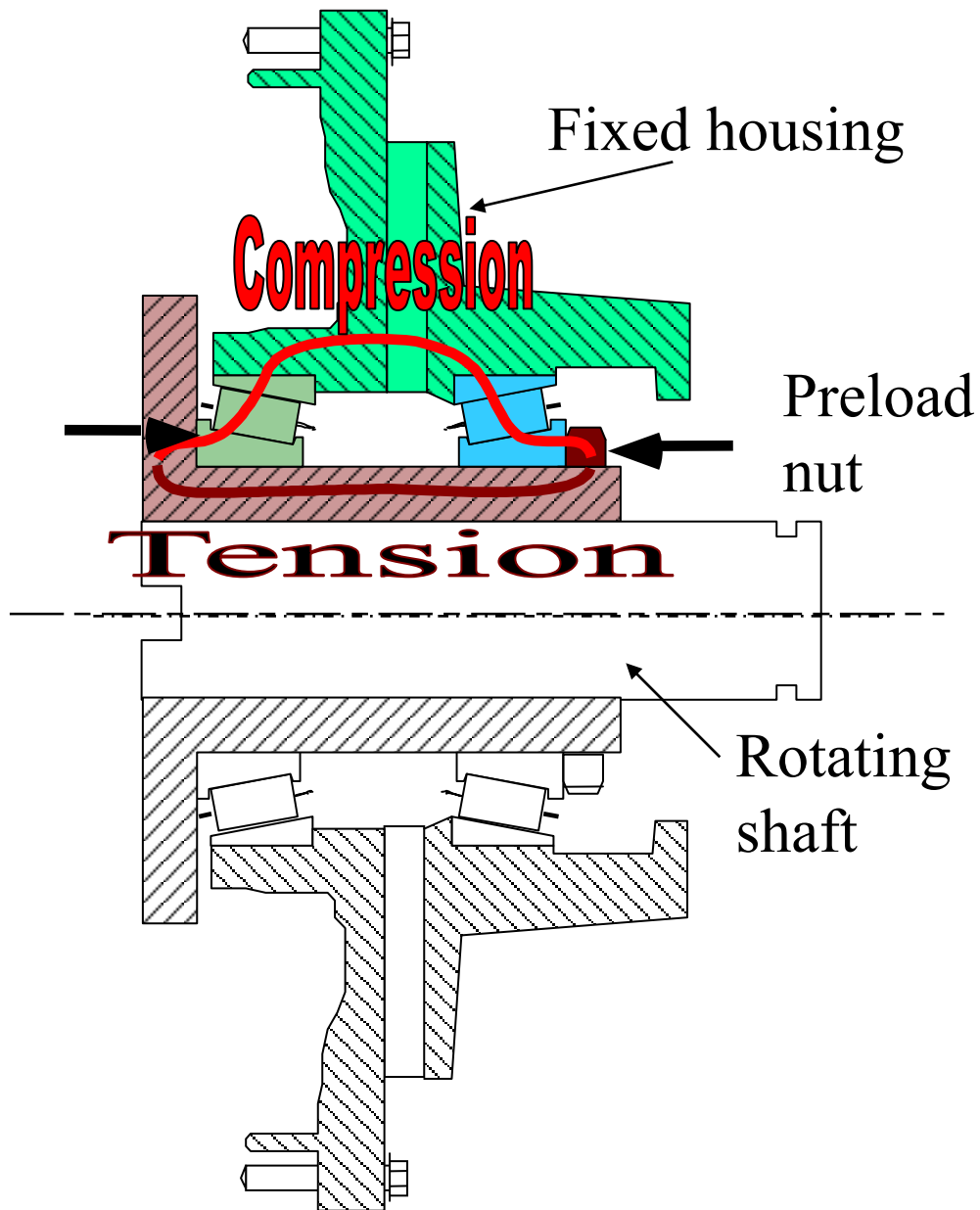
- Boeing:
 - Ensure that there is open space at max material condition
 - Fill the gap with shims, reducing gap to XXX
 - Report remaining gap to Engineering
 - Lately: use better process control to predict gaps and prepare standard shims in as many cases as possible
- Airbus:
 - Make parts from 3D CAD/NC
 - Join them directly
 - “Look, Dr Whitney, no shims!”
- Both attempt to limit locked-in stress

“Good” Over-constrained Assemblies

- Preloaded angular contact bearing systems
 - Preload increases contact stress, creating a stiff bearing system (see next page)
- Planetary gears - redundant locators, no stress
- Shrink fit
 - Heated wheel slips on over shaft, shrinks upon cooling to make a super-tight joint
- Beam built in at both ends
 - It's stiffer for the same cross section than a simply-supported beam because the ends can support a moment
 - A good design permits longitudinal motion at the ends
- In each case there is an underlying properly constrained system!



Preloaded Angular Contact Bearings



Planetary Gear Systems

Images removed for copyright reasons.

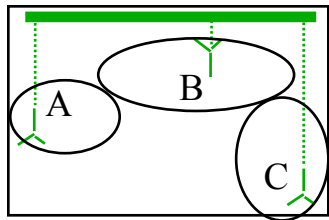
Source:

Figure 4-13 in [Whitney 2004] Whitney, D. E. *Mechanical Assemblies: Their Design, Manufacture, and Role in Product Development*. New York, NY: Oxford University Press, 2004. ISBN: 0195157826.

Why Does Over-Constraint Occur?

- Forces or torques are deliberately inserted, e.g.
 - Shrinking
 - Tightening a lock nut
- The design attempts to fix more than 6 degrees of freedom of a part, e.g.
 - The x position is determined by the part's left end
 - The part's x position is determined by the part's right end
 - There is a fight whose outcome is compression in the x direction and no easy way to calculate the x position

How Different Approaches Deal with Constraint

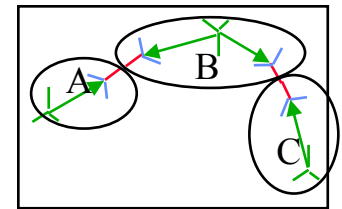


World
Model

Use consistency
check to find
joining errors
↓
Can't detect most
over- or
under-constraints

Use
interference
check to find
shape errors

Local
Feature
Model



Use screw
theory to
find joining
errors: over-
or under-
constraint

What's the Solution to Over-constraint?

- Increase the diameter of one hole
- Increase the diameter of both holes
- Use one hole and one slot
- Match drill one hole pair and use a slot
- Accept a little locked-in stress
- We will use these ideas later when we use features to transfer dimensional constraint and location from part to part using the Datum Flow Chain
 - over-constraint = ambiguity about dimensional transfer
 - requires a stress analysis to decide where the parts are

Tipoffs for Over-constraint

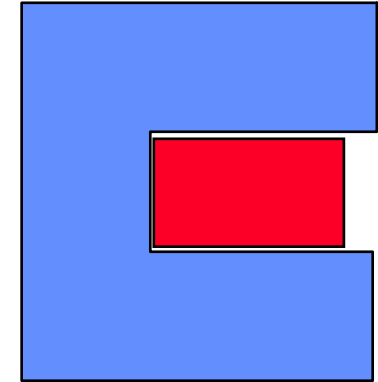
- It takes skill to put the parts together and get them just right
- The assembly task is operator-dependent
- Fasteners have to be tightened in a particular sequence
- It is hard to get welded parts out of the fixture
- Some parts will assemble easily but other “identical” ones will not
- You can never get everything to line up the way you want it to
- Results are inconsistent

Location, Constraint, Stability

- Constraint determines location
 - this is done using “mates” (Whitehead’s “locators”)
- Stability keeps or *effects* location
 - this is done with “contacts” (Whitehead’s “effectors”)
 - screws, wave washers, clamps, welds, glue, chewing gum, etc
- Locating gets it there
- Stabilizing keeps it there
- This area is a common cause of confusion

Location and Stability

You know where it is,
even if it might not stay there



It will stay there (it's welded)
but you don't know where it is



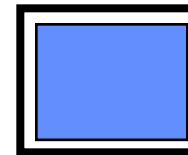
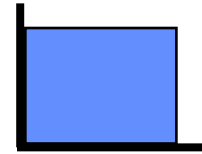
Whitehead's Definitions

(examples use less than 6 dof)

	• Enough Joints	• Too Many Joints
Effected by "small" force	<ul style="list-style-type: none"> – Pure kinematic design <ul style="list-style-type: none"> • 3 legged stool with point-tipped legs – Semi-kinematic design with redundant constraint in the small contact area of each locator <ul style="list-style-type: none"> • 3 legged stool with non-zero contact area at each leg tip 	<ul style="list-style-type: none"> – Redundant constraint <ul style="list-style-type: none"> • 4 legged stool with point-tipped legs or non-zero contact area at each tip • This is really two 3 legged stools - your choice which one!
Effected by a large force	<ul style="list-style-type: none"> – Semi-kinematic design <ul style="list-style-type: none"> • 3 legged stool with non-zero contact area and each leg bolted down tight 	<ul style="list-style-type: none"> – Over-constraint <ul style="list-style-type: none"> • 4 legged stool with each leg bolted down tight

Force Closures and Form Closures

- Force closures are one-sided
 - They support force in one direction at a definite location
 - They can provide proper constraint
- Form closures are two-sided
 - They can support unlimited force
 - They will generate over-constraint unless some clearance is provided
 - If clearance is provided, then the location is no longer definite

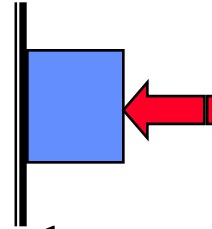


One-Side and Two-Side Constraints

- One-side (AKA force closure)

- Needs an effector

- Gives perfect knowledge of location but can't support an arbitrary force in all directions



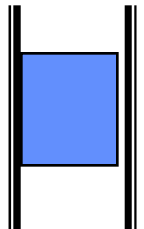
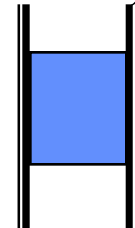
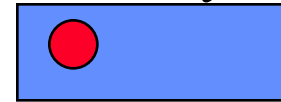
- Two- or multi-side constraint (AKA form closure)

- Needs no effector and can support arbitrary force

- Contains its own stabilizer

- Actually contains over-constraint

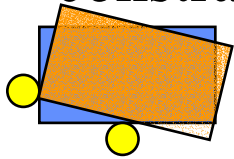
- If we relax this over-constraint with a little clearance then we lose perfect knowledge of location



See “Exact Constraint Design Using Kinematic Principles” by Douglass Blanding

Taxonomy of Assemblies

Under-constrained

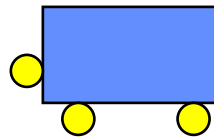


Mechanisms

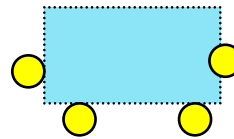
mistakes-attempts to achieve location that lock in stress or fail to locate

Properly Constrained


zero-stress (or almost zero stress) assemblies that deliver KCs by achieving location (using transform T)



Interference and stress

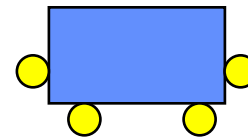


Pre-loaded bearing sets

 = "Noise"

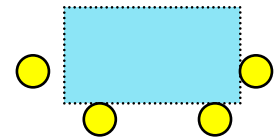
Over-constrained

Line fit



Can't happen but needed for analysis

Clearance AKA redundant



Duplicated arrangements

Mathematical Family of Joints

- It contains 7 kinds of features and 28 possible joints
 - arbitrary surface
 - parallelepiped
 - body of revolution
 - cylinder
 - plane
 - sphere
- They all are one-sided, as are all 28 combinations
- Elements can be used to build features from scratch
- Next slide shows cylinder, plane, and sphere

Basic Surface Contacts and Motions They Allow

Allowed motion
of the black part
is shown, when
the red part is
stationary

Images removed for copyright reasons.

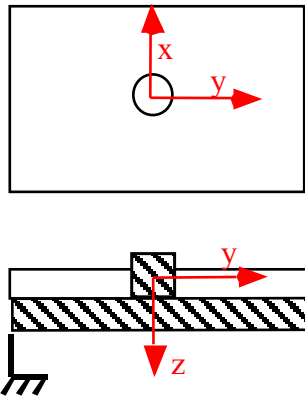
Source:
Table 4-3 in [Whitney 2004] Whitney, D. E. *Mechanical Assemblies: Their Design, Manufacture, and Role in Product Development*. New York, NY: Oxford University Press, 2004. ISBN: 0195157826.

Engineering Family of Assembly Features

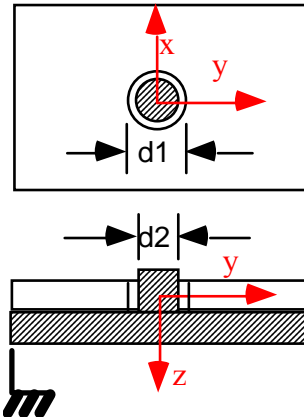
- It contains 17 shapes and counting
 - peg and blind/through hole
 - peg on plane and blind/through hole
 - peg on plane and slot
 - key in keyway
 - etc.
 - You can add any that you want
- They are both one-sided and two-sided
- They are built using basic surfaces

Examples of Engineering Features

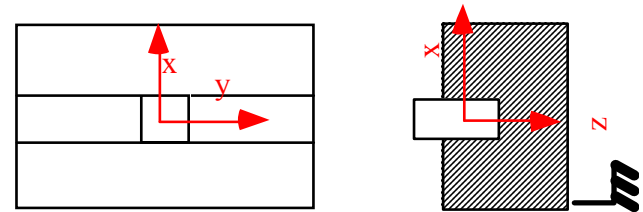
Pin in Hole



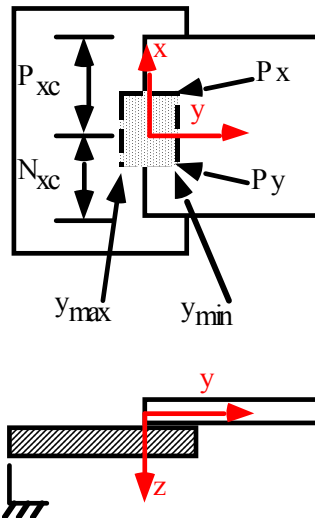
Pin in Oversize Hole



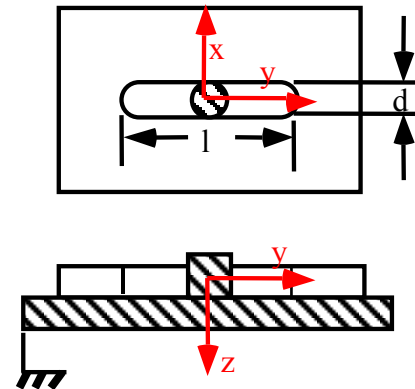
Pin in Prismatic Slot



Two Plates



Pin in Slot



Assembly Feature Construction Options

- Build from elementary surfaces
 - Reveals two-side over-constraints inside a feature
 - Permits detection of over-constraints caused by too many one-side constraints
- Use basic mechanical shapes like holes and slots
 - Suppresses over-constraints inside a feature
 - Permits detection of over-constraints caused by too many one- or two-sided constraints
- The constraint testing methods described next must be applied with caution, depending on which option is used

Assembly Feature Construction Options

Elementary surfaces
(TTRS) - one-sided
Have location, no size

Intersect these
to make new features.
They will contain over-
constraint if 2-sided.
They will have location and size
if 2-sided

Make new features
from scratch. They will not
contain over-constraint
even if 2-sided.
They will have location and size
if 2-sided

Need to add clearance
on size to relieve internal
over-constraint

Intersect these
to make compound
features - will
contain over-
constraint if 2-sided
or if mistakes are made

Kinematicians' Approach to Constraint

- Kinematicians are interested in things that move
- So they are interested in “mobility” M
- Assemblies may or may not move
- We are interested in redundancy or negative mobility
- The Kutzbach criterion attempts to analyze both but it can give the wrong answer

$$M = 6(n - g - 1) + \sum f_i$$

n = number of links

g = number of joints

$\sum f_i$ = total dof of all joints

$M = 0$. proper constraint

$M < 0$. over - constraint

$M > 0$. under - constraint

If the linkage is planar,
the Grübler criterion is used.
It is the same as the Kutzbach
Criterion except that “6” is
replaced by “3”

Does the Kutzbach Criterion Work?

Images removed for copyright reasons.

Source:

Figure 4-4, 4-5, 4-6 in [Whitney 2004] Whitney, D. E. *Mechanical Assemblies: Their Design, Manufacture, and Role in Product Development*. New York, NY: Oxford University Press, 2004. ISBN: 0195157826.

$$M = 3(n - g - 1) + \sum f_i$$

n = number of links

g = number of joints

$\sum f_i$ = total dof of all joints

$M = 0$. proper constraint

$M < 0$. over - constraint

$M > 0$. under - constraint

Our Approach

- Mobility and constraint require separate analyses
- If an assembly is not over-constrained **and** it is not under-constrained, then and only then is it properly constrained
- Mobility and constraint analyses are done using Screw Theory
- The Kutzbach criterion is a naïve attempt to do what Screw Theory can do

Basics of Screw Theory

- Kinematic joints permit motion and resist force
- Each degree of freedom of motion is called a twist
- Each degree of force resistance is called a wrench
- A twist has the form $T = [. \ x \ . \ y \ . \ z \ v_x \ v_y \ v_z]$
- A wrench has the form $W = [f_x \ f_y \ f_z \ M_x \ M_y \ M_z]$
- Twists and wrenches are *reciprocals* of each other (under certain conditions)
- That is, directions that allow motion cannot support force and vice versa

Constraint is Accomplished by Surfaces in Contact

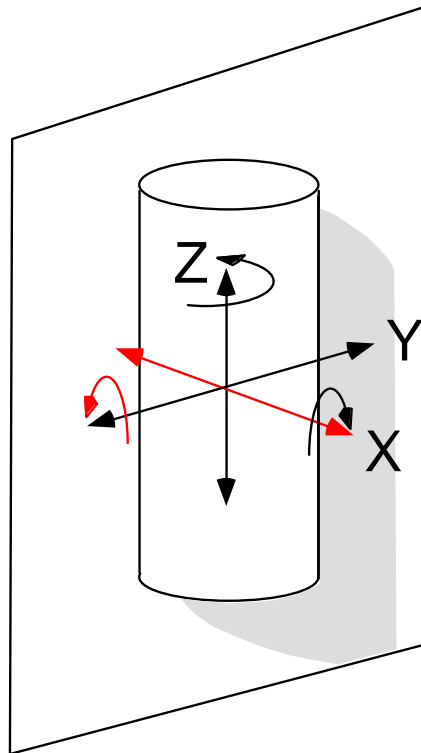
The contact permits some dof to move with respect to each other and prevents motion of other dof.

The black ones can move.

The red ones can't.

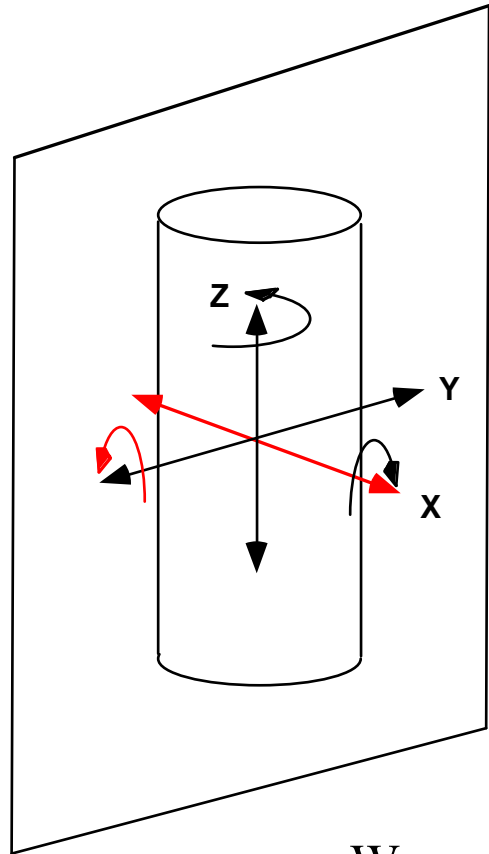
The red ones can support force.

The black ones can't.



Different kinds of surface pairs permit and prevent motion of different dof

Twist Space and Wrench Space Describe the Behavior of Two Surfaces in Contact



WRENCH SPACE

TWIST SPACE

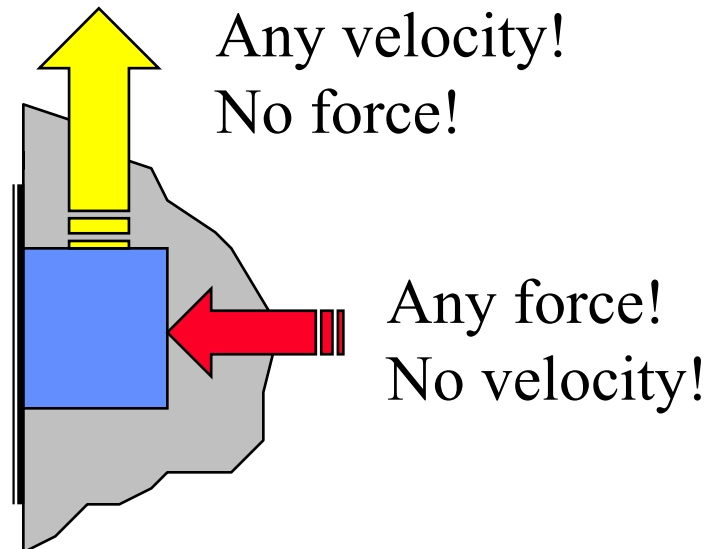
Wrench space and twist space are reciprocal

Reciprocal of Screw

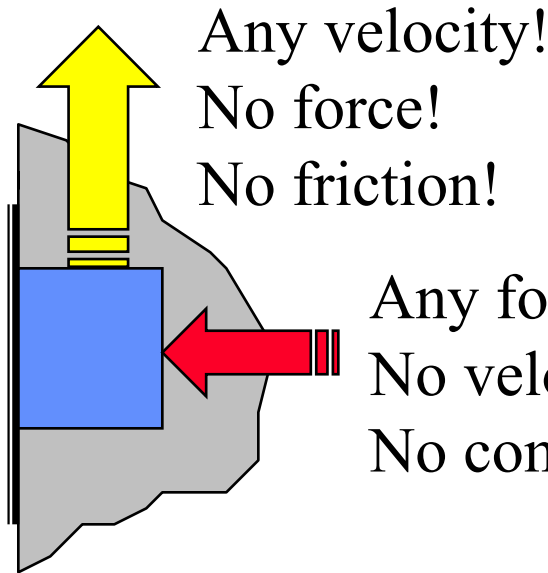
- Reciprocal of a twist is a wrench
- Reciprocal of a wrench is a twist
- Twist and wrench are reciprocal if wrench cannot do work along direction of twist (no friction)
- This means: $\text{Twist} \cdot \text{Wrench} = 0$ (\cdot = dot product)
- Equivalently, wrench is in the null space of twist
- Rank of twist matrix + rank of its reciprocal wrench matrix = 6

Forces and Velocities in Constraint

- In general, force and velocity (or any pair of variables whose product = power) are related by an impedance ($v=iR$, for example)
- When parts are rigid and friction is zero, all impedances are zero or infinite.



Relation Between Motions (Twists) and Forces (Wrenches)

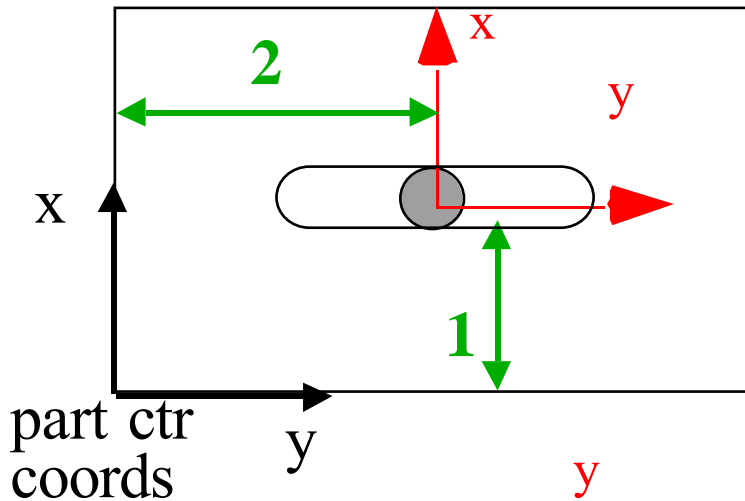


*also
Recipe
Recipe
Recipe*

$$T = \text{recip}(W) = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

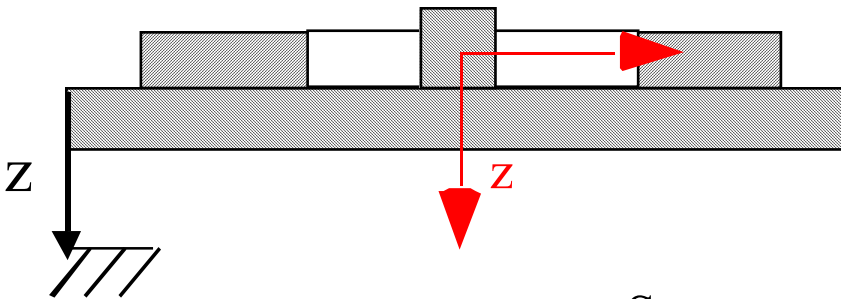
$$W = \begin{matrix} \underbrace{\hspace{2cm}}_F & \underbrace{\hspace{2cm}}_M \\ \text{(recip}(T)) \\ = \begin{bmatrix} 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \end{matrix}$$

Twist Representation of Plate and Slotted Pin Joint Shows What Motions are Allowed



$$\text{twist} = [\cdot \ x \cdot \ y \cdot \ z \ v_x \ v_y \ v_z]$$

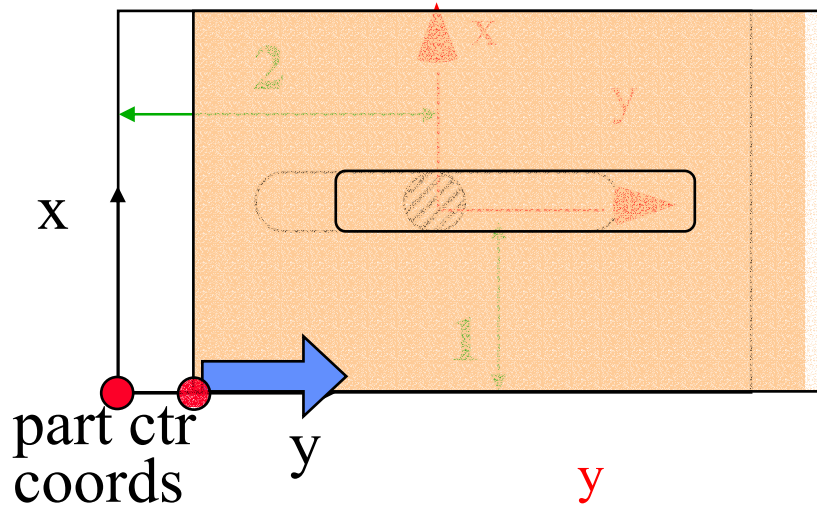
$$= \begin{bmatrix} 0 & 0 & 1 & 2 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{matrix} \text{Z rotation} \\ \text{Y translation} \end{matrix}$$



The twist has two rows because the feature allows two different motions

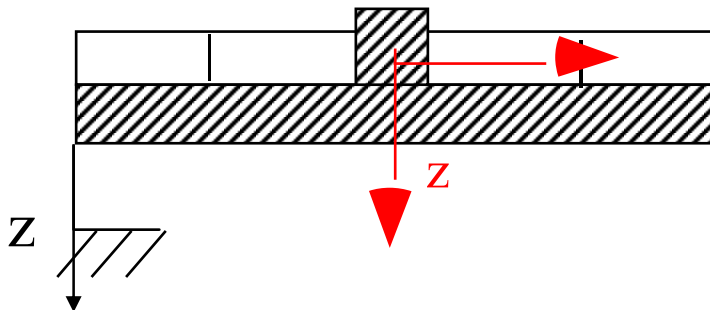
See next two slides for explanation

The Translation Part of the Twist

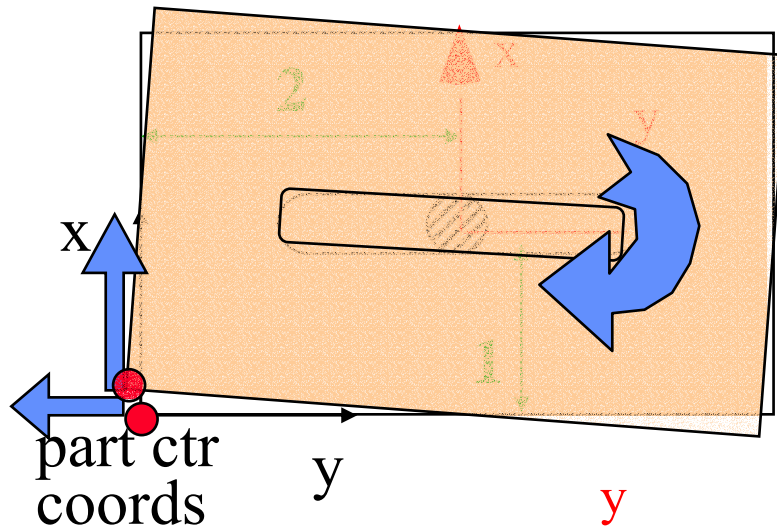


How the red dot moves:

$$[\underbrace{0 \ 0 \ 0 \ 0}_{\cdot} \ \underbrace{1 \ 0}_{\mathbf{v}}]$$



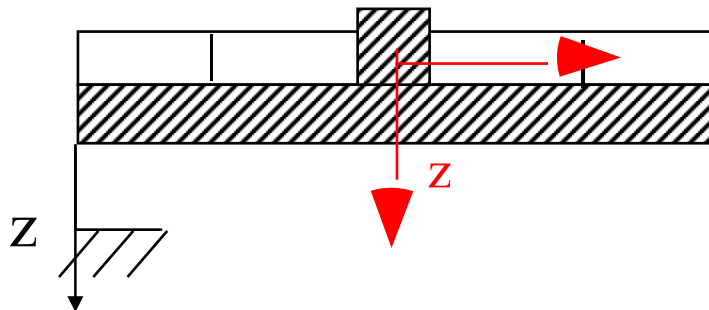
The Rotation Part of the Twist



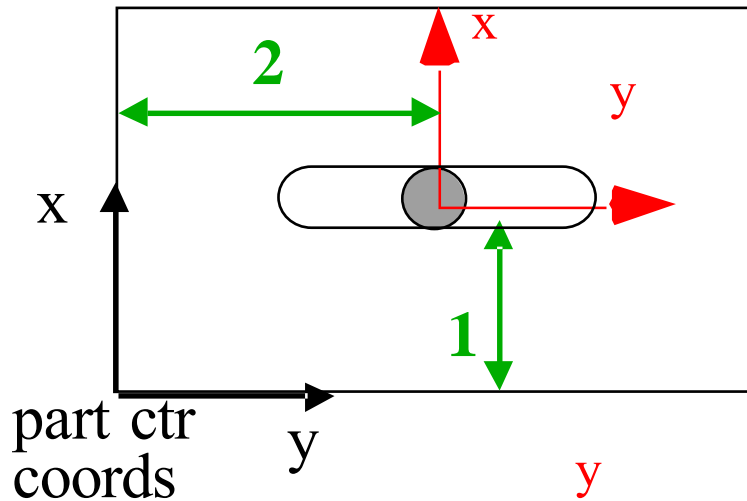
How the red dot moves:

$$[0 \ 0 \ 1 \ 2 \ -1 \ 0]$$

$\underbrace{\hspace{2em}}$ $\underbrace{\hspace{2em}}$
 . v



Wrench Representation of Plate and Slotted Pin Joint Shows What Forces and Moments Can Be Resisted



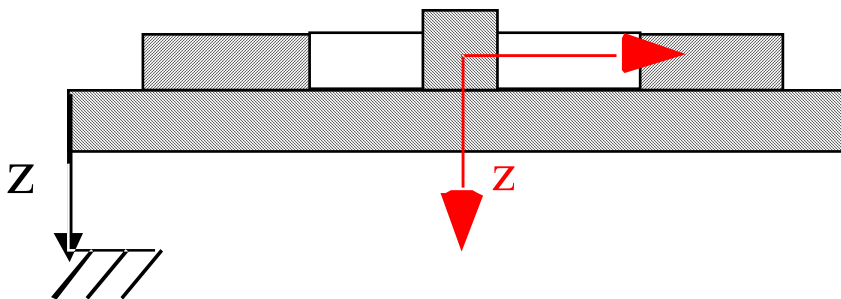
$$\text{wrench} = [f_x \ f_y \ f_z \ M_x \ M_y \ M_z]$$

$$= 1 \ 0 \ 0 \ 0 \ 0 \ -2 = X \text{ force at pin}$$

$$0 \ 0 \ 1 \ 0 \ 0 \ 0 = Z \text{ force}$$

$$0 \ 0 \ 0 \ 1 \ 0 \ 0 = X \text{ Moment}$$

$$0 \ 0 \ 0 \ 0 \ 1 \ 0 = Y \text{ Moment}$$



The wrench has four rows because the feature can resist four different forces

Motion and Constraint Analysis When Parts are Joined by Several Features

- Parts are joined by features
- Multiple features may constrain properly or they may contain over/under constraints
- **Motion** analysis permits detection of **under-**constraint
- **Constraint** analysis permits detection of **over-**constraint
- Proper constraint = not *over-* and not *under-*constrained

Motion and Constraint Analysis Using Screw Theory-1

- Each feature is represented by a twist that shows the motions it allows, expressed in part center coordinates as: $[\cdot \ x \cdot \ y \cdot \ z \ v_x \ v_y \ v_z]$
- If a part joins another part via several features, then the **intersection** of all the features' twists shows the net allowed motion
- If a net motion is allowed, then all the features allow that motion
- All feature twists and the twist intersection are expressed in the same coordinate system

Motion Analysis (Twist Intersection) Algorithm

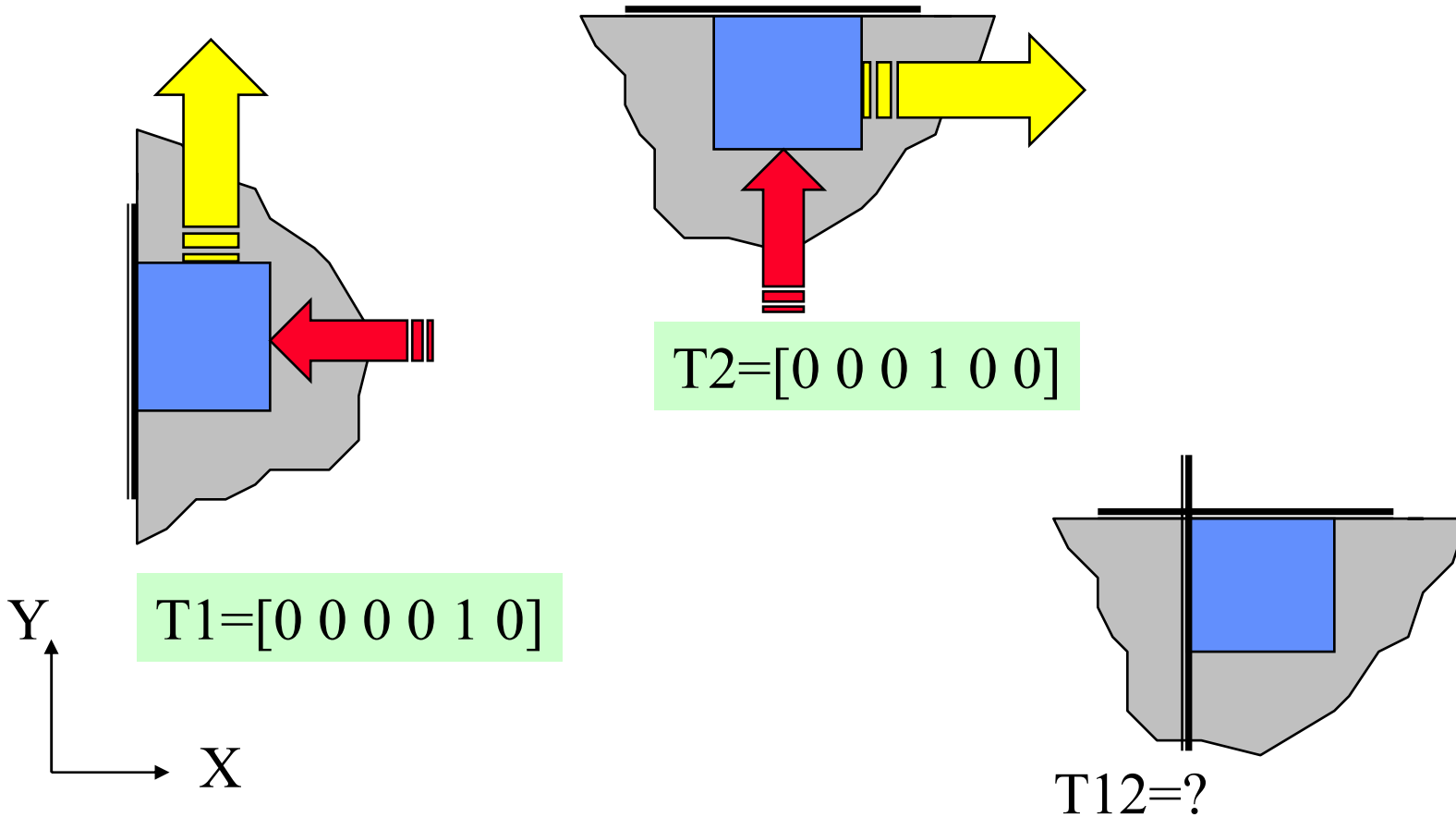
- For each feature i joining two parts, having twist T_i
 - find the associated wrench W_i using
 - $W_i = \text{recip}(T_i)$
- Form the union of all the $W_i = W_U$
 - $W_U = [W_1; W_2; \dots]$ (using MATLAB notation)
- Find resultant net twist from
 - $T = \text{recip}(W_U)$
 - $T_R = \text{rref}(T)$ (rref simplifies the result for easy reading)
- If any motion exists in T_R , then every joint this part has allows this motion

.m files for Screw Calculations

```
function R = recip(T)
% Takes the reciprocal of a screw matrix
p = (null(T))';
[i,j]=size(p);
if i>0
    R = flip(p);
    R=rref(R);
else
    disp('empty matrix')
    R=zeros(0);
end
% rref finds row-reduced echelon form
% ' takes transpose
```

```
function W = flip(WU)
% FLIPs columns of WU
% col 1 becomes 4, 2 becomes 5, and 3 becomes 6
% col 4 becomes 1, 5 becomes 2, and 6 becomes 3
[i,j] = size(WU);
if j == 6
    for l=1:i
        for k=1:3
            W(l,k) = WU(l,k+3);
            W(l,k+3) = WU(l,k);
        end
    end
end
W;
else
end
```

Example: Cube on Floor at Wall(s)



Example Twist Intersection

»T1=[0 0 0 0 1 0]

T1 =

0 0 0 0 1 0

»T2=[0 0 0 1 0 0]

T2 =

0 0 0 1 0 0

»W1=recip(T1)

W1 =

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

»W2=recip(T2)

W2 =

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

» WU=[W1;W2]

WU =

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

»T=recip(WU)
 empty matrix

T =

[]

»

So it can't move

Motion and Constraint Analysis Using Screw Theory-2

- Each twist has a reciprocal called a wrench expressed in part center coordinates as $[f_x \ f_y \ f_z \ M_x \ M_y \ M_z]$
- It represents all the forces and torques that the feature can transmit to a mating part
- The intersection of some wrenches shows if there is/are direction(s) that they all constrain
- To find if a part is over-constrained, it is necessary to intersect all combinations of wrenches until all over-constrained directions have been found

Constraint Analysis (Wrench Intersection) Algorithm

- For all features joining two parts, each having T_i
 - Form their union $TU = [T_1; T_2; \dots]$
 - find $W = \text{recip}(TU)$
 - find $WR = \text{rref}(W)$ - for ease of interpretation
- If a wrench appears in WR then every joint can exert that wrench, which means over-constraint
- But if WR is empty, it does not mean that there is no over-constraint!
- You have to check all pairs, triplets, etc.

Summary of Twist and Wrench Intersection

$$v(X_i) = \text{recip}\{. [\text{recip}(X_i)]\}$$

Intersection of twists:

$$\left. \begin{array}{l} T1 . \quad W1 \\ T2 . \quad W2 \\ T3 . \quad W3 \\ \text{etc.} \end{array} \right\} . (Wi)=WU . \quad TR$$

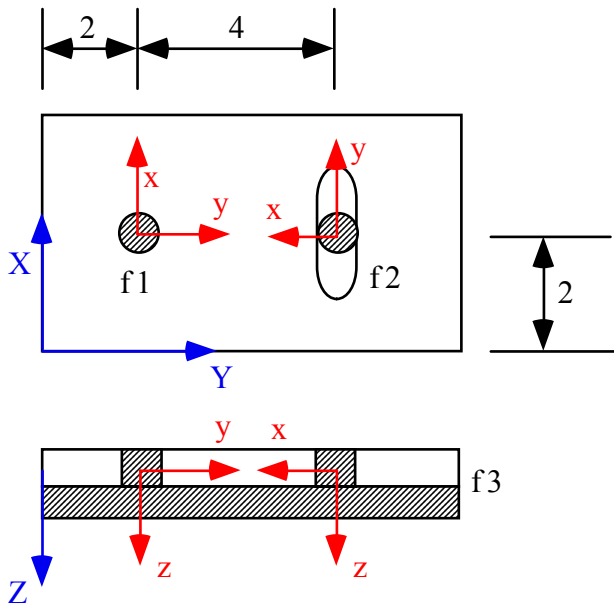
Intersection of wrenches:

$$\left. \begin{array}{l} W1 . \quad T1 \\ W2 . \quad T2 \\ W3 . \quad T3 \\ \text{etc.} \end{array} \right\} . (Ti)=TU . \quad WR$$

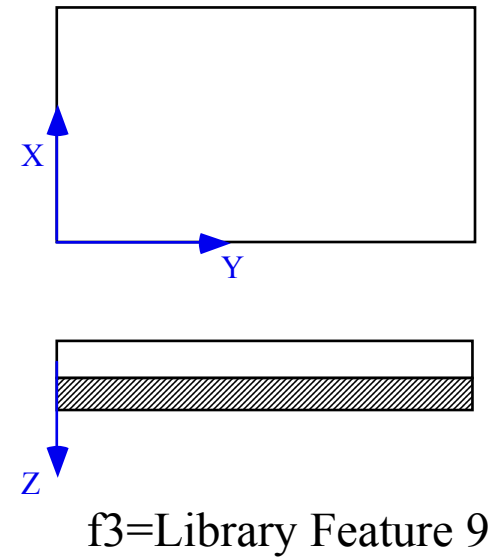
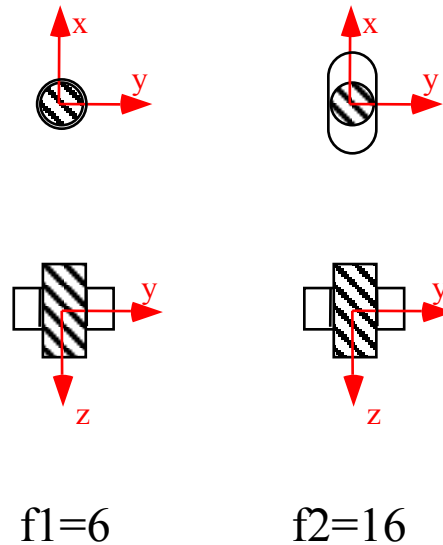
$$. = \text{reciprocal} \quad . = \text{union}$$

Example: Assembly Made by Combining Several Features

The assembly



The features used to make it



Motion Analysis Results

»T1=[0 0 1 2 -2 0;0 0 0 0 0 1]

T1 =

```
0 0 1 2 -2 0
0 0 0 0 0 1
```

»T2=[0 0 1 6 -2 0;0 0 0 1 0 0;0 0 0 0 0 1;0 1 0 0 0 2]

T2 =

```
0 0 1 6 -2 0
0 0 0 1 0 0
0 0 0 0 0 1
0 1 0 0 0 2
```

»T3=[0 0 0 1 0 0;0 0 0 0 1 0;0 0 1 0 0 0]

T3 =

```
0 0 0 1 0 0
0 0 0 0 1 0
0 0 1 0 0 0
```

»W1=recip(T1)

W1 =

```
1.0000 0 0 0 0 -2.0000
0 1.0000 0 0 0 2.0000
0 0 0 1.0000 0 0
0 0 0 0 1.0000 -0.0000
```

»W2=recip(T2)

W2 =

```
0 1.0000 0.0000 0 -0.0000 2.0000
0 0 0 1.0000 0 0
```

»W3=recip(T3)

W3 =

```
0 0 1 0 0 0
0 0 0 1 0 0
0 0 0 0 1 0
```

»WU=[W1;W2;W3]

WU =

```
1.0000 0 0 0 0 -2.0000
0 1.0000 0 0 0 2.0000
0 0 0 1.0000 0 0
0 0 0 0 1.0000 -0.0000
0 1.0000 0.0000 0 -0.0000 2.0000
0 0 0 1.0000 0 0
0 0 1.0000 0 0 0
0 0 0 1.0000 0 0
0 0 0 0 1.0000 0
```

»TU=recip(WU)

TU =

```
0 0 1.0000 2.0000 -2.0000 0.0000
```

Part can rotate about Z
Rotation center is at (2,2)

Constraint Analysis Results

Intersect all twists:

»TU123=[T1;T2;T3]

TU123 =

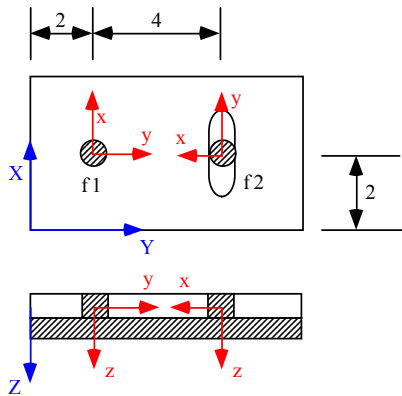
$$\begin{bmatrix} 0 & 0 & 1 & 2 & -2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 6 & -2 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

»WU123=recip(TU123)

WU123 =

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

M_x



Intersect all pairs of twists:

»TU12=[T1;T2]

TU12 =

$$\begin{bmatrix} 0 & 0 & 1 & 2 & -2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 6 & -2 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 2 \end{bmatrix}$$

»WU12=recip(TU12)

WU12 =

$$\begin{bmatrix} 0 & 1.0000 & -0.0000 \\ 0 & 0 & 0 & 1.0000 \end{bmatrix}$$

»TU13=[T1;T3]

TU13 =

$$\begin{bmatrix} 0 & 0 & 1 & 2 & -2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

»WU13=recip(TU13)

WU13 =

$$\begin{bmatrix} 0 & 0 & 0 & 1.0000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.0000 & 0.0000 \end{bmatrix} \begin{matrix} M_x \\ M_y \end{matrix}$$

»TU23=[T2;T3]

TU23 =

$$\begin{bmatrix} 0 & 0 & 1 & 6 & -2 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

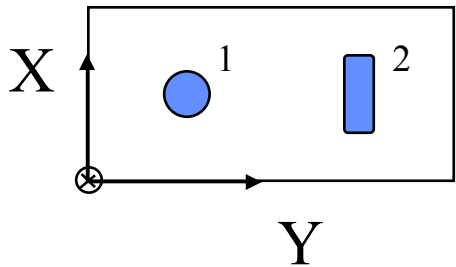
»WU23=recip(TU23)

WU23 =

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

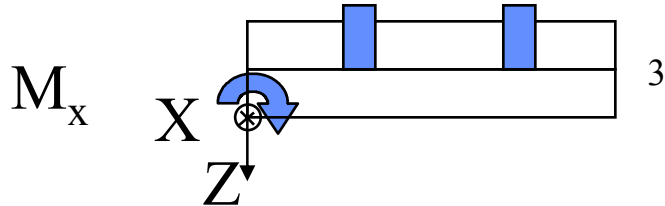
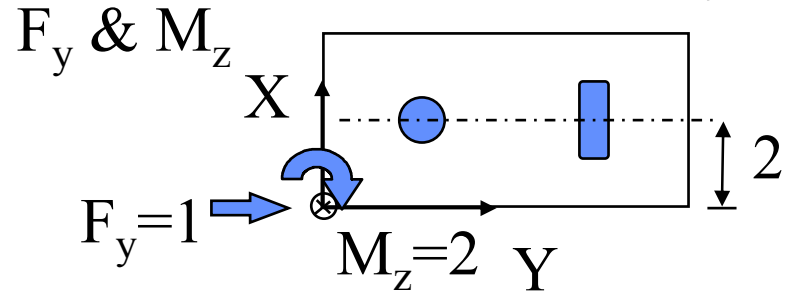
M_x

Constraint Analysis Interpretation



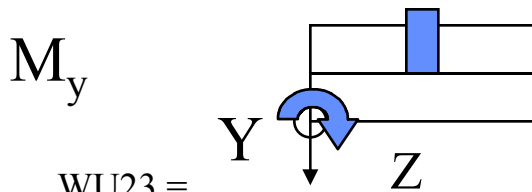
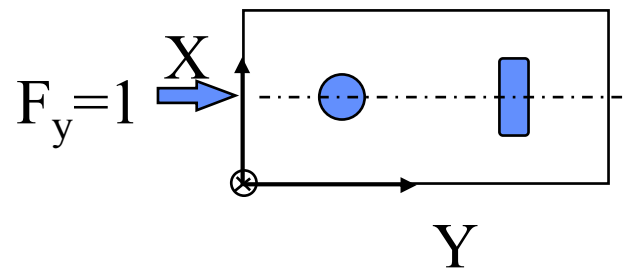
$$WU_{12} = \begin{matrix} 0 & 1 & 0 & 0 & 0 & 2 \end{matrix}$$

What MATLAB says:



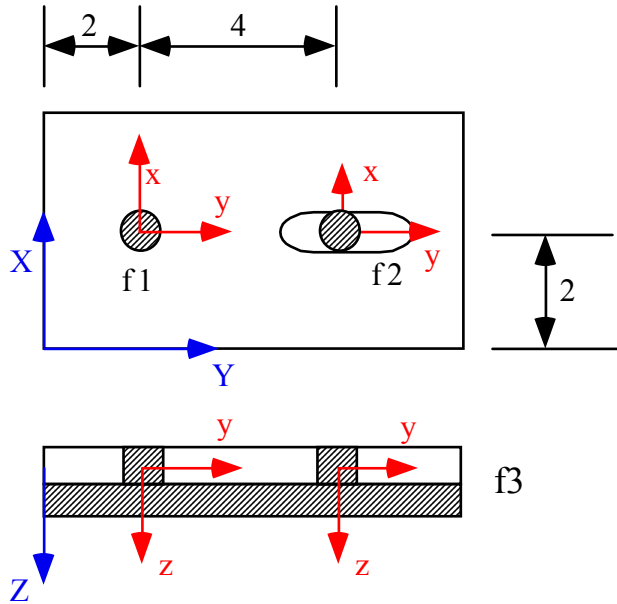
$$WU_{12} = \begin{matrix} 0 & 0 & 0 & 1 & 0 & 0 \end{matrix}$$

What it means:



$$WU_{23} = \begin{matrix} 0 & 0 & 0 & 1 & 0 & 0 \end{matrix}$$

Second Example



- Analysis results:
 - No motion is possible
 - Over-constraint exists about **X** and **Y**

$$WU_{123} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$WU_{12} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$WU_{13} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$WU_{23} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$T_2 =$

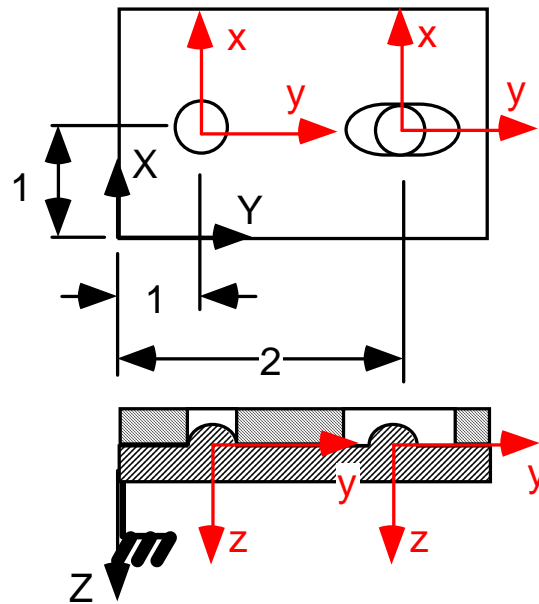
$$\begin{bmatrix} 0 & 0 & 1 & 6 & -2 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & -6 \end{bmatrix}$$

T_1 & T_3 the same

$TU = []$

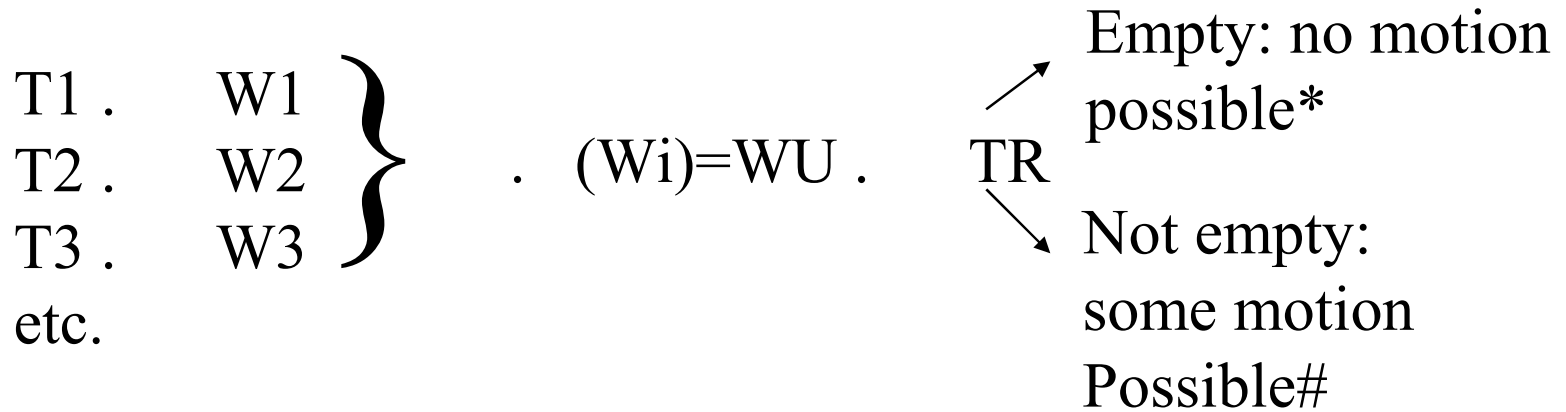
empty matrix

A Way to Eliminate Some Two-Side Over-constraints



Use library features 18 and 19 instead of 6 and 16. These X-Y locators do not fight with each other over orienting the top plate and do not fight with the bottom plate over orienting the top plate

How to Handle More Than Two Features: Motion Analysis

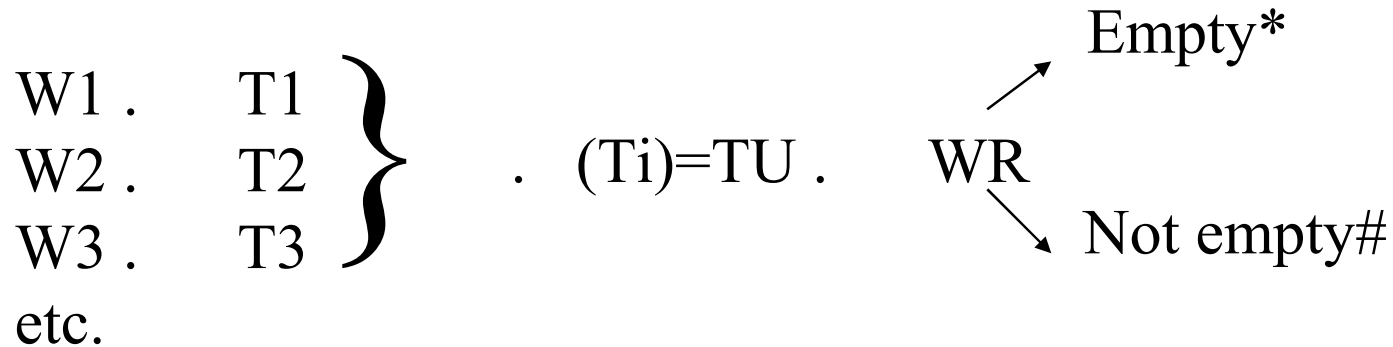


. = recip

*No direction of motion can be provided by all these features

#One or more directions of motion can be provided by all these features

How to Handle More Than Two Features: Constraint Analysis



. = recip

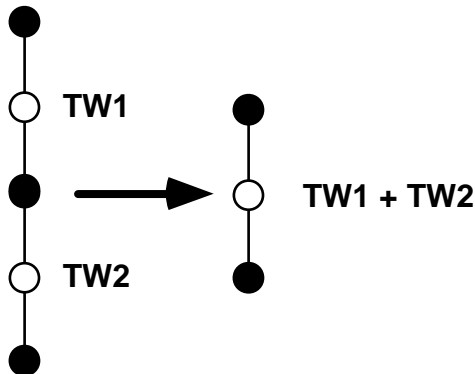
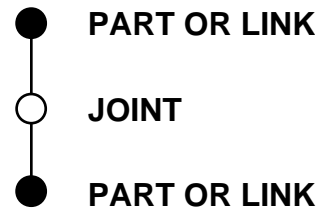
*No force or moment can be resisted by *all* these features

#One or more forces or moments can be resisted by *all* these features

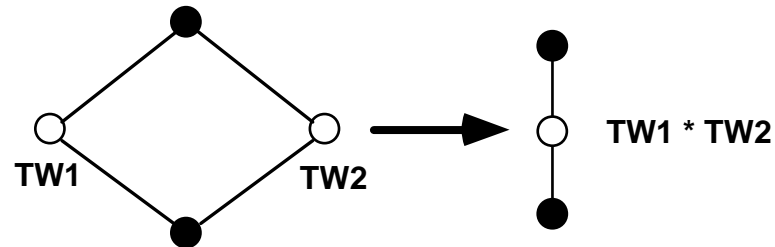
General Problem of Mechanism Freedom

- Our solution is based on Screw Theory and a method due to Konkar for intersecting twists
- It is based on tracing paths through the mechanism from a link of interest to a fixed link
- Separate motion and constraint analyses are done
- Our method works for a class of mechanisms
- It can be applied to a succession of subassemblies in an assembly sequence
- A complete solution method exists - see the CD

Path Method for Motion and Constraint Analysis - 1: Series and Parallel Reduction

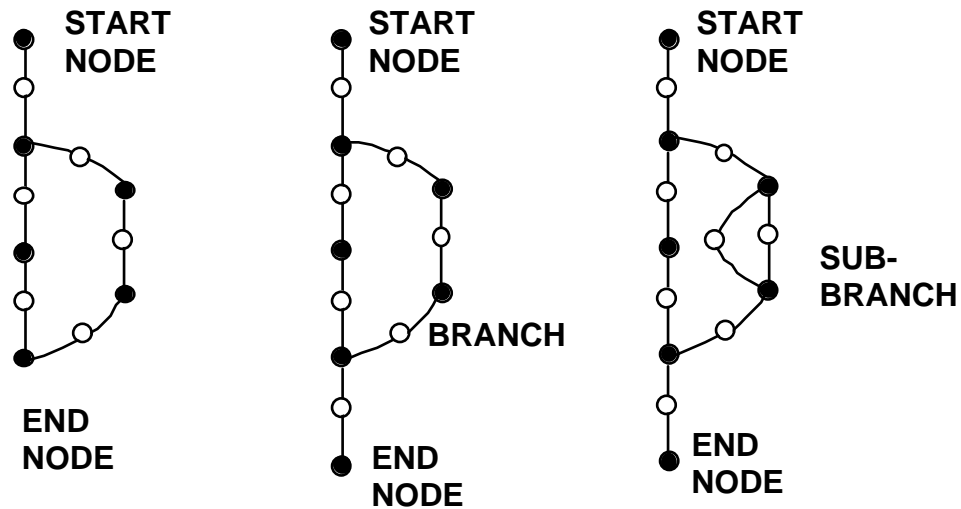


Joints in series are reduced using union



Joints in parallel are reduced using intersection

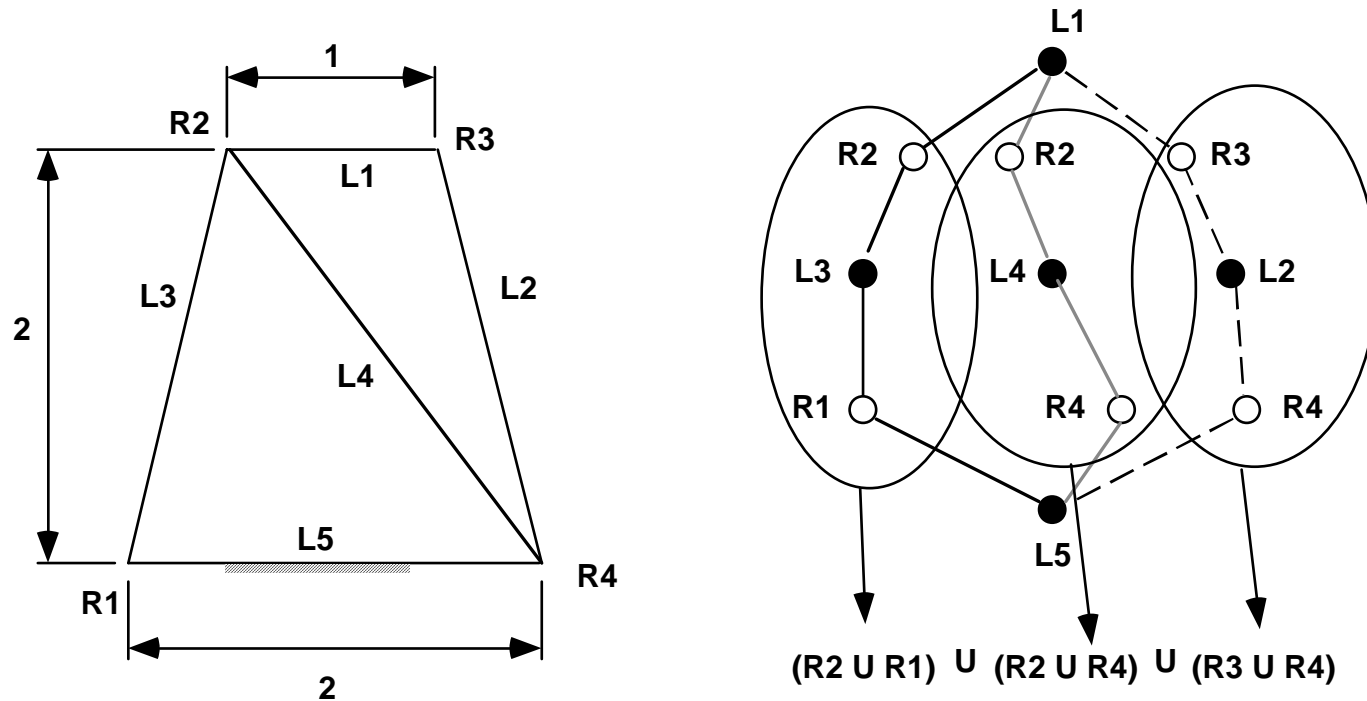
Path Method for Motion and Constraint Analysis - 2: Paths and Branches



Path Method for Motion and Constraint Analysis - 3: The Process

- Identify the link to be analyzed and call it “start”
- Call the fixed link “end”
- Identify all paths and branches from start to end
- Form union of twists along each path and branch
- Intersect twist unions from branches and paths using the reduction rules until the mechanism consists of “start” and “end joined by one equivalent joint

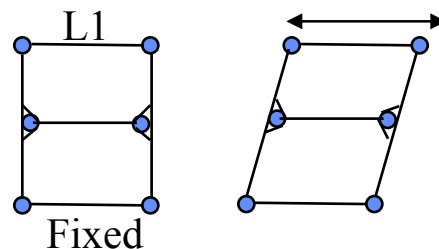
Path Method for Motion and Constraint Analysis - 4: An Example



The required unions and intersections can be written as a Boolean expression and fed right into Matlab

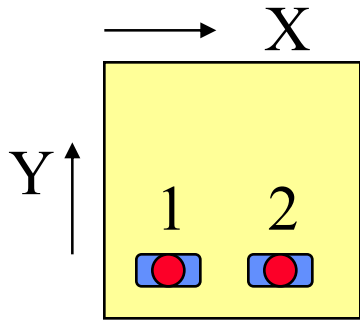
Conditions for Success of This Method

- Paths must be independent at link of interest
 - All the dof of joints connected to the link of interest must be independent of each other
- Same as saying there cannot be any cross-links between paths near link of interest



This mechanism is both over- and under-constrained. Our method does not work on it. The Kutzbach criterion also gives the wrong answer.

Constraint Analysis with Multiple Features

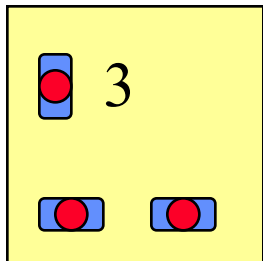


$F1 \vee F2 =$ under-constrained

$TR = \vee (T_1, T_2) = recip[. (W_1, W_2)]$ is not empty

X motion is allowed

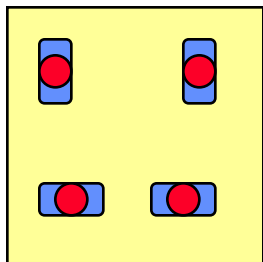
$WR = recip[. (T_1, T_2)]$ is empty



$\vee [F1, F2, F3] =$ properly constrained

$WR = recip[. (T_1, T_2, T_3)]$ is empty

$TR = recip[. (W_1, W_2, W_3)]$ is empty



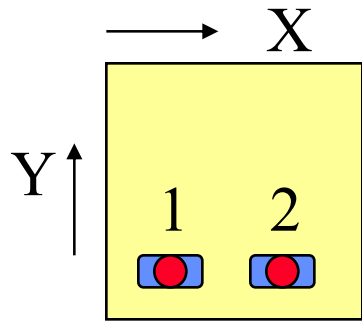
4

$\vee [F1, F2, F3, F4] =$ not over-constrained,

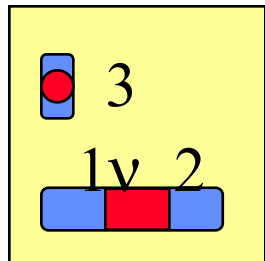
$WR = \vee (W_1 \cdots W_4) = recip[. (T_1, T_2, T_3, T_4)]$ is empty

Even tho we know it IS over-constrained

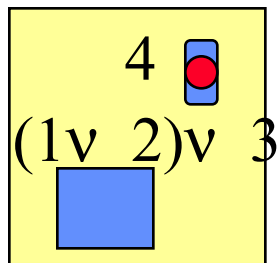
Constraint Analysis with Multiple Features - 2



$F1 \vee F2 = \text{under-constrained}$
 $TR_{12} = \vee (T_1, T_2) = \text{recip}(. (W_1, W_2))$
 X motion is allowed
 $WR = \text{recip}[\cdot (T_1, T_2)]$ is empty

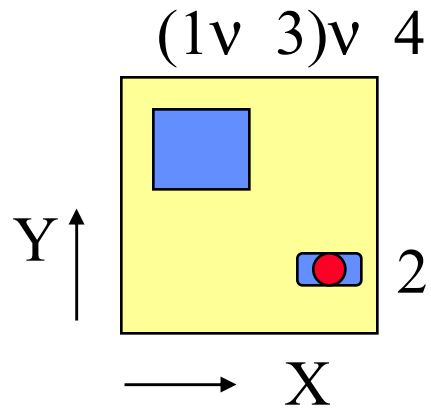


$F3 \vee (F1 \vee F2) = \text{not over-constrained}$
 $TR_{123} = \vee (T_1, T_2, T_3) = \text{recip}[\cdot (W_1, W_2, W_3)]$ is empty
 $WR = \text{recip}\{ [T_3, TR_{12}] \}$ is empty



$\{[(F1 \vee F2) \vee F3] \vee F4\} = \text{over-constrained}$
 $TR_{123} = \vee (T_1, T_2, T_3) = \text{recip}[\cdot (W_1, W_2, W_3)]$
 $WR = \text{recip}\{ [T_4, TR_{123}] \}$ is not empty

What Direction is Over-constrained?



References for Kinematic Design

Slocum A. H., *Precision Machine Design*, New York: Prentice-Hall, 1991.

Smith, S. T., and D. G. Chetwynd, *Foundations of Ultraprecision Mechanism Design*, Philadelphia: Gordon and Breach, 1992

Whitehead, T. N., *The Design and Use of Instruments and Accurate Mechanism*, New York: Dover Press, 1954.

Publications

- Whitney, D. E., Gilbert, O., and Jastrzebski, M., "Representation of Geometric Variations Using Matrix Transforms for Statistical Tolerance Analysis in Assemblies, Research in Engineering Design, (1994) 6: pp 191-210
- Mantripragada, R., Cunningham, T. W., and Whitney, D. E., "Assembly-oriented Design: A New Approach to Designing Assemblies, Proceeding, IFIP WG5.2 Workshop on Geometric Modeling in CAD, May 19 - 23, 1996.
- Mantripragada, R. and Whitney, D. E., "The Datum Flow Chain," Research in Engineering Design, v 10, 1998, pp 150-165.
- Mantripragada, R. and Whitney, D. E., "Modeling and Controlling Variation Propagation in Mechanical Assemblies using State Transition Models," IEEE Trans on Robotics and Automation, 1, no 1, Feb, 99, pp 124-140.
- Adams, J. D. and Whitney, D. E., "Application of Screw Theory to Constraint Analysis of Assemblies of Rigid Parts," 1999 IEEE ISATP
- Whitney, D. E., Mantripragada, R., Adams, J. D., and Cunningham, T. W., "Use of Screw Theory to Detect Multiple Conflicting Key Characteristics in Complex Mechanical Products," 1999 ASME DFM Conf