Simbody Mobilizers and Constraints (intro)

Michael Sherman (Sherm)

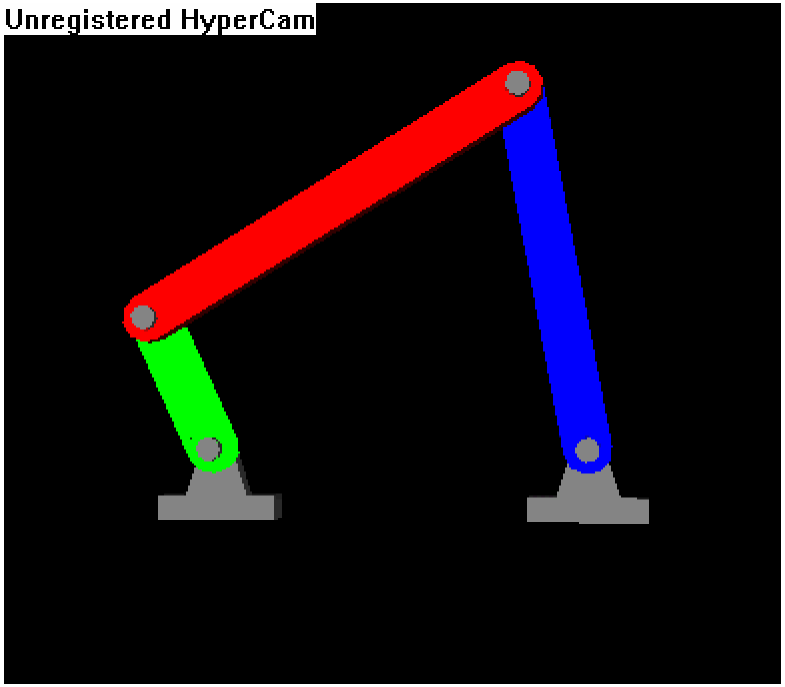
Xulu Entertainment, Inc.

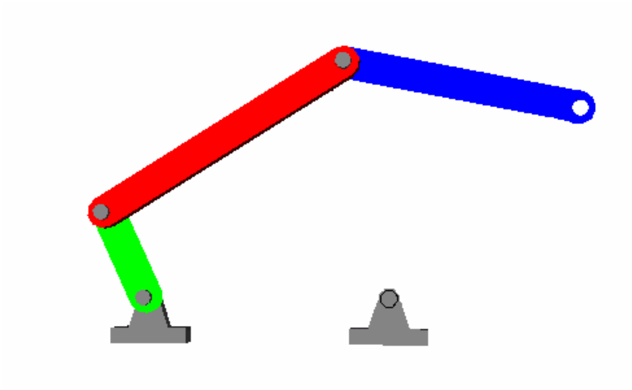
(was: Simbios chief software architect)

SimTK 1.5 Workshop, Sept. 25, 2008

NIH Roadmap
grant U54 GM072970

Joints can *permit* or *restrict* motion



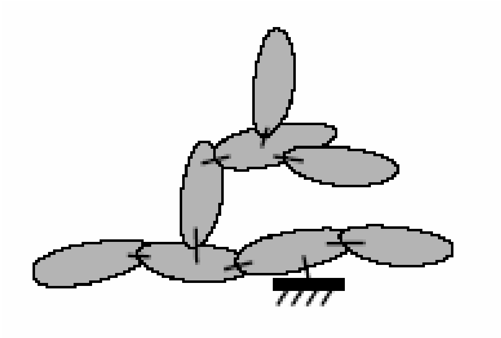


- 4-bar linkage has only 1 DOF
- But has 3 if you remove any one joint

Mobilizers

(joints which *permit* motion)

- In Simbody, bodies do not have inherent mobility
- Mobilizers precisely define the allowable motion relative to parent
- A mobilizer *always* increases the system's mobility
- These define the generalized coordinates q and generalized speeds u
- Bodies + mobilizers form a tree

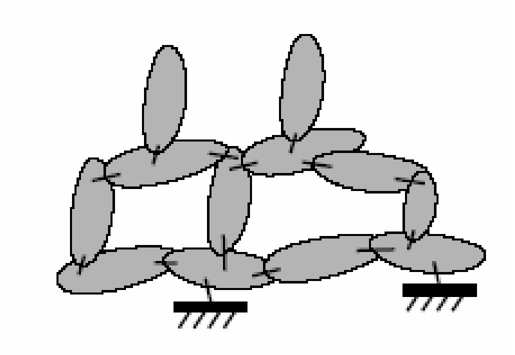


$$\mathbf{M}\ddot{q}=f$$

Constraints

(joints which *restrict* motion)

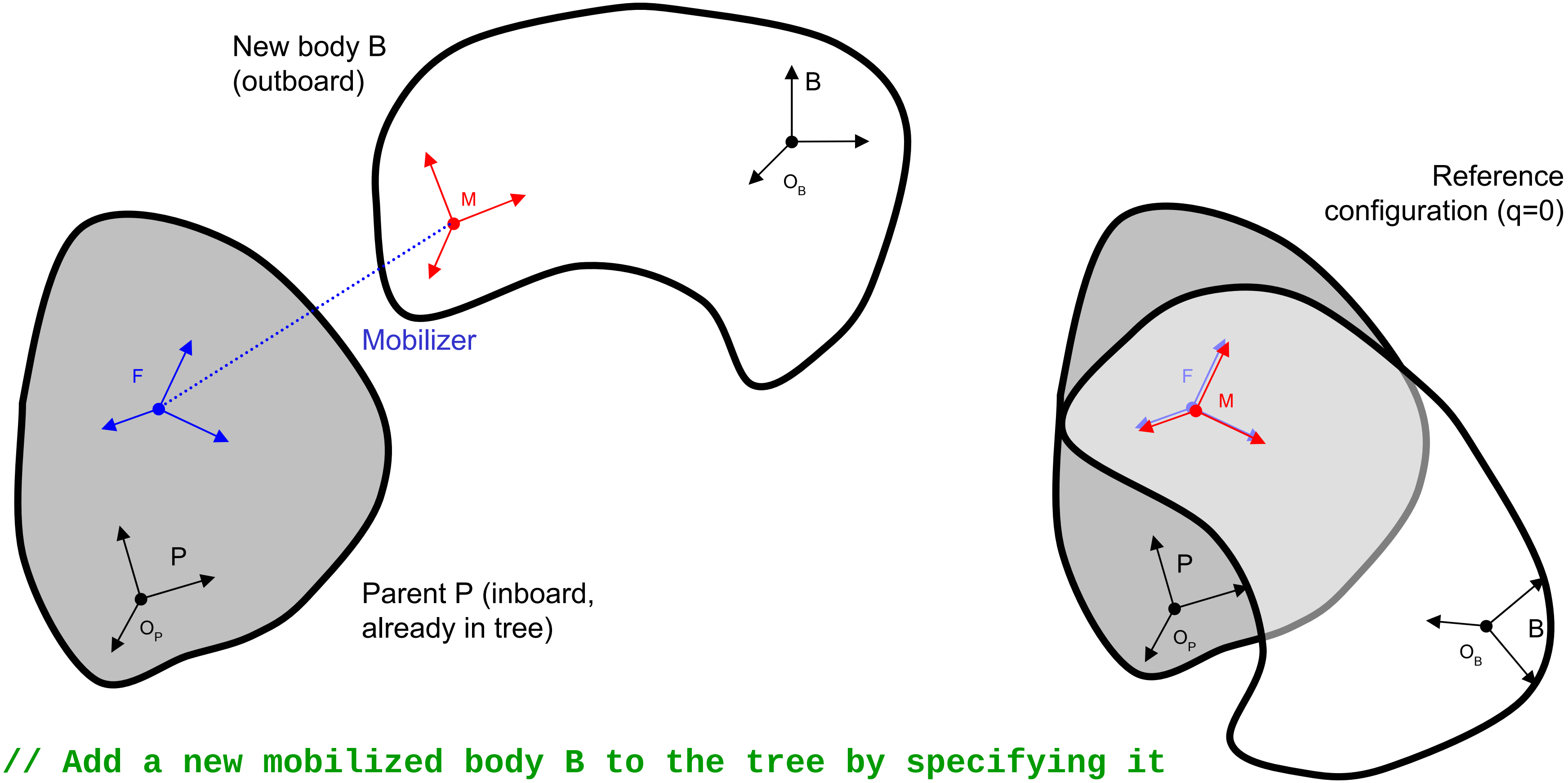
- Trees can be a little too floppy …
- *Constraints* introduce *constraint equations* (1 or more)
  - Restricts allowable motion – like negative mobility
  - E.g., ball constraint adds 3 constraint equations, -3 dofs
- Algebraic invariant relating q’s and u’s: g(q,u)=0
- But … might not be independent
- Must solve assembly problem before simulating
  - Find q such that g(q)=0
- Constraints permit loops, generate additional unknown forces



$$\mathbf{M}\ddot{q} = f - f_c$$

$$\mathbf{g}(q) = 0$$

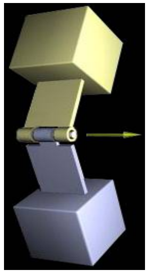
Mobilizers in Simbody



```
// Add a new mobilized body B to the tree by specifying it
// parent (inboard) body P and the mobilizer that grants
// mobility to B relative to P.
MobilizedBody::type B(MobilizedBody   parentP,
                      Transform       parentMobilizerFrameInP,
                      MassProperties  mpropsForB,
                      Transform       childMobilizerFrameInB);
```

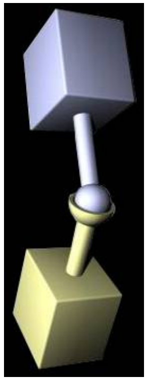
Available Mobilizers in 1.5

- Pin (Torsion)
- Slider (Prismatic)
- Universal
- Cylinder
- Planar
- BendStretch
- Gimbal (3D rotation angles)
- Ball (Orientation, Spherical)
- Translation (Cartesian x-y-z)
- Free, FreeLine
- Screw
- Ellipsoid (thanks, Ajay!)
- More?



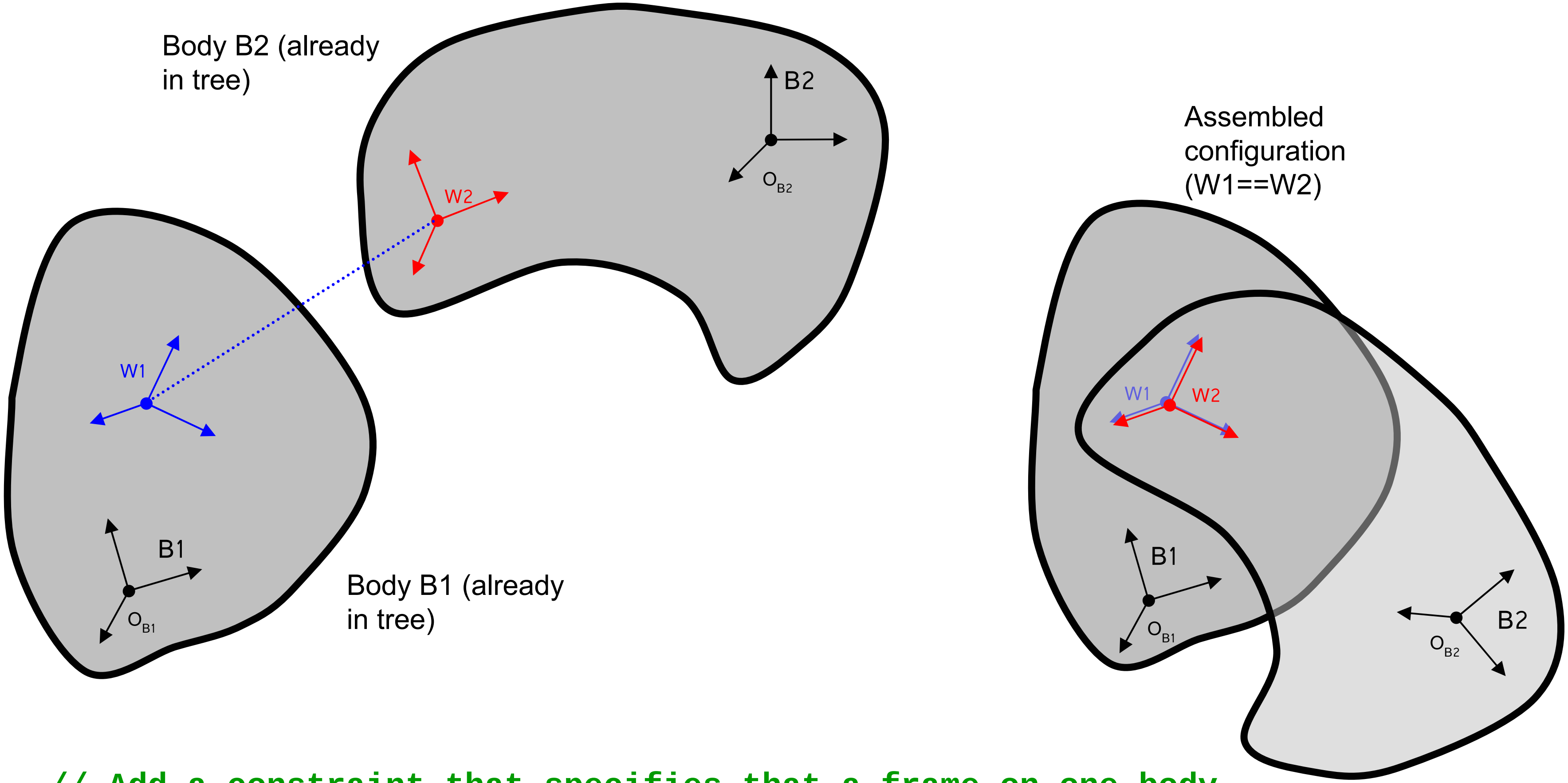
Or, you can make your own!

- Custom Mobilizer



Constraints in Simbody

Example: weld constraint



Body B2 (already in tree)

Body B1 (already in tree)

Assembled configuration (W1==W2)

```
// Add a constraint that specifies that a frame on one body
// remain coincident with a frame on another.
Constraint::Weld(MobilizedBody body1,
                 Transform     frameW1inB1,
                 MobilizedBody body2,
                 Transform     frameW2inB2);
```

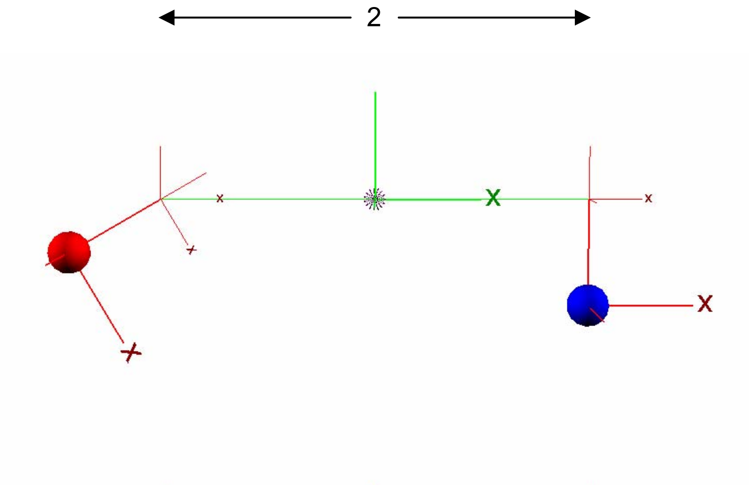
Available Constraints in 1.5

- Rod (Distance)
- Ball (CoincidentPoints)
- Weld (CoincidentFrames)
- PointInPlane
- PointOnLine
- ConstantAngle
- ConstantOrientation
- NoSlip1D (Gear)
- Prescribed motion
- More?

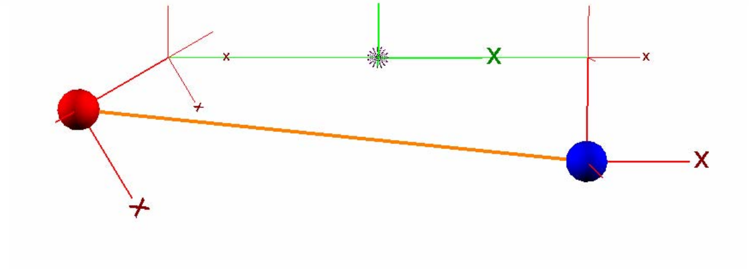
Or, you can make your own!

- Custom Constraint

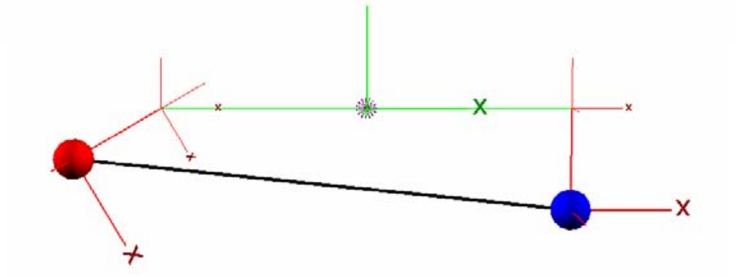
Rod constraint example



- No connection
- Left pendulum has initial condition of -60 degrees



- Connected by a spring of natural length 2, plus damping
- Spring is initially stretched



- Connected by a rod (distance) constraint d=2
- Constraint is initially violated; must be assembled

That’s enough hand waving …

sherm@xulu.com

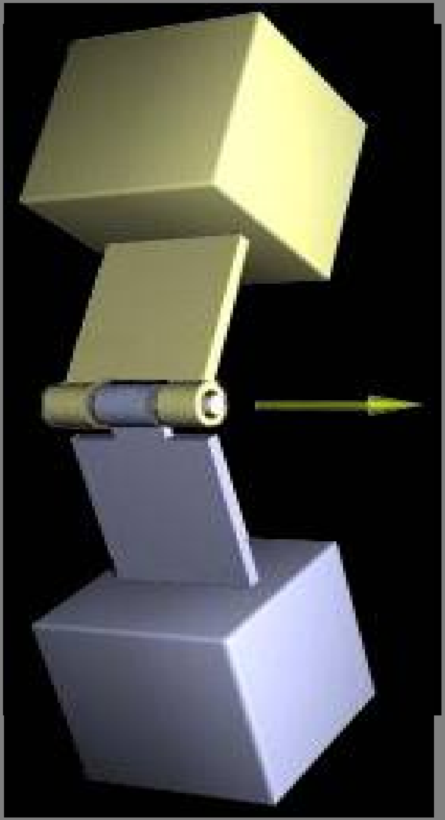
Optional – custom mobilizer

Biological joints via Simbody mobilizers

- Current tools adapted from mechanical engineering
  - Joints designed for ideal behavior, ease of manufacturing
- Biological joints are different
  - Similar goals but different materials, growth instead of manufacture, etc.
  - Well approximated with low dofs
  - But motion is very complex
- Thanks to Ajay Seth for the following slides …

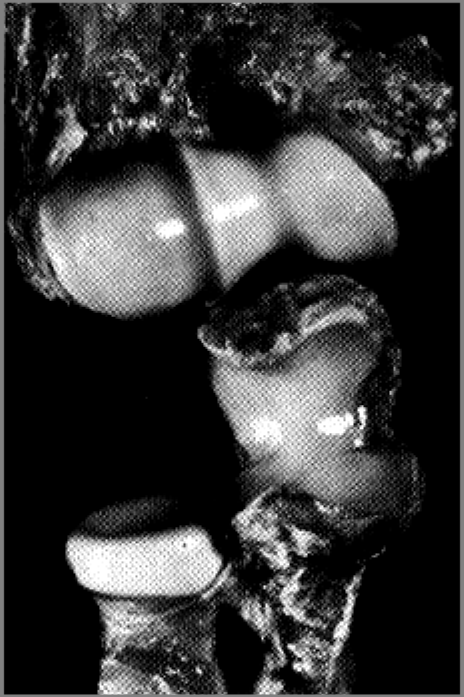
Some 1-dof joints

Hinge (pin joint)



Finger

Elbow



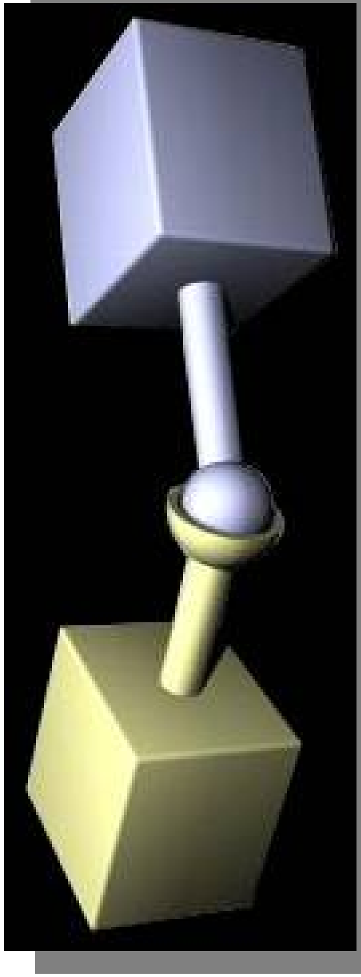
Ajay Seth, 2007

Pure rotation

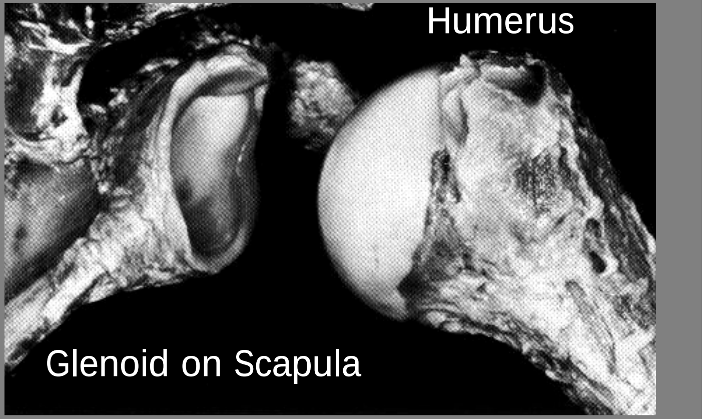
Rotation + translation

Some 3-dof joints

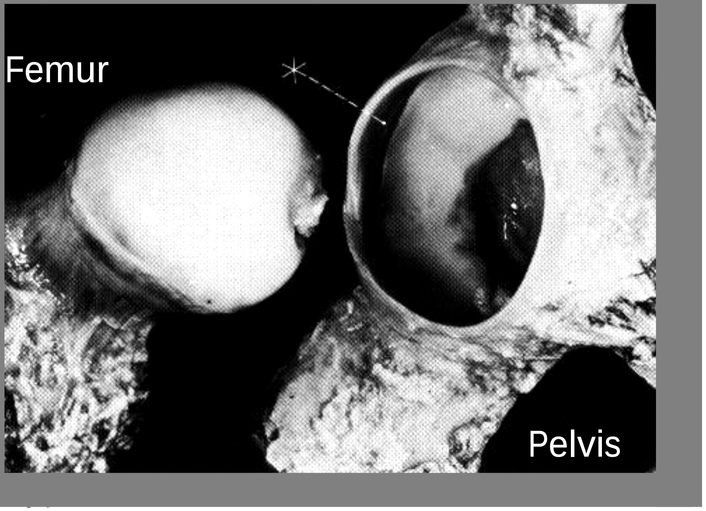
Ball & socket



Pure rotation



Shoulder



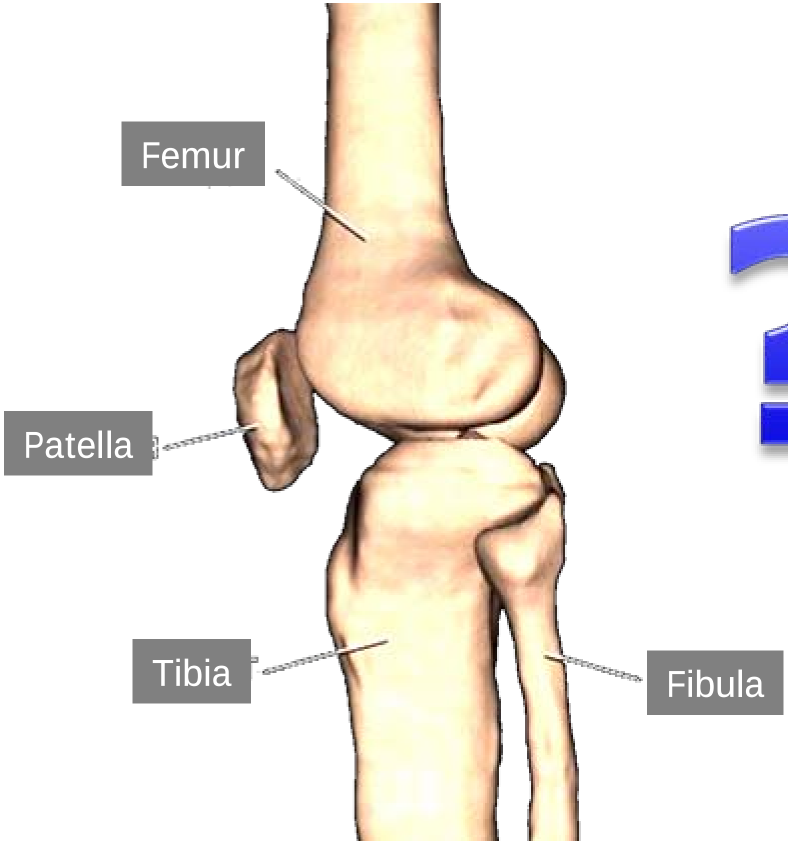
Hip

Ajay Seth, 2007

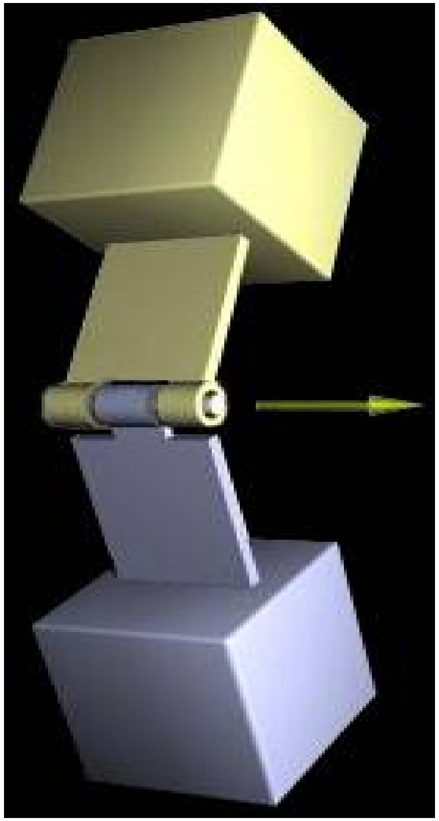
Rotation + translation

Is The Knee a Hinge?

Knee

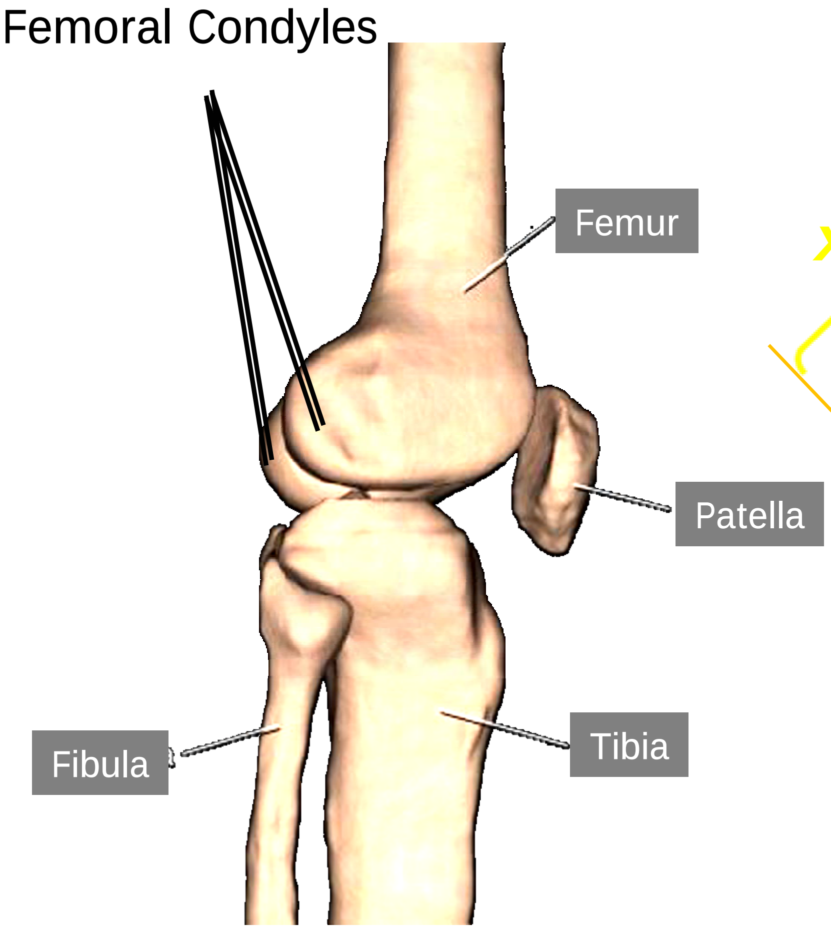


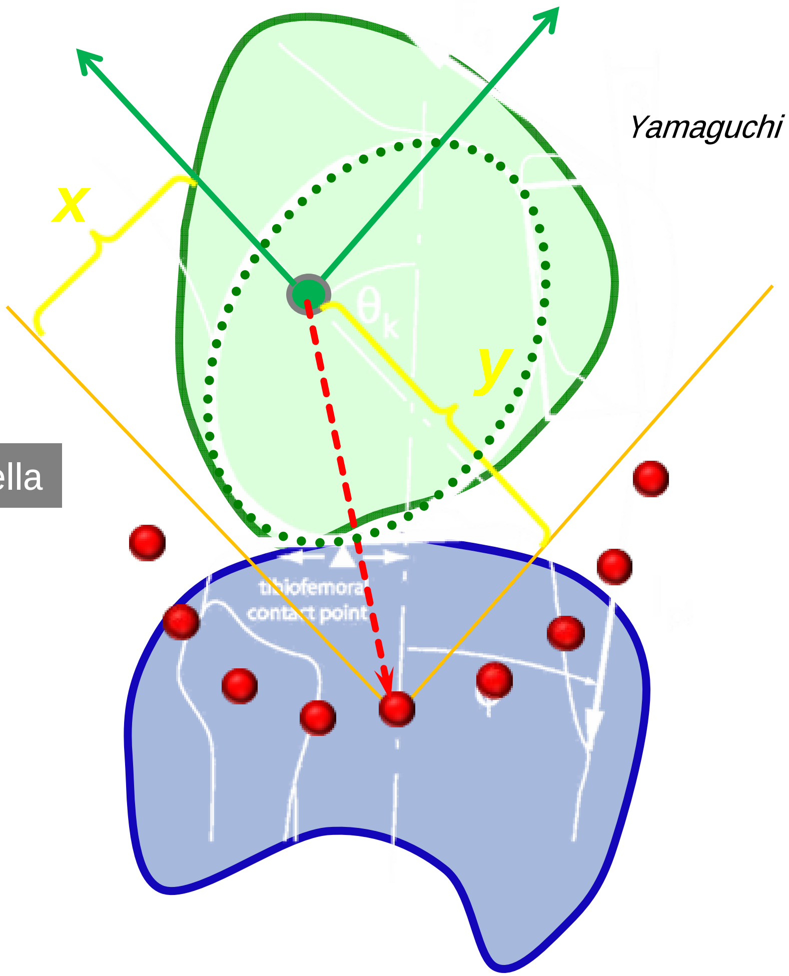
Hinge



Ajay Seth, 2007

Human Knee Anatomy





Yamaguchi & Zajac 1989

Result: biological joints must be "faked" in mechanical codes

- Use extra mechanical coordinates & constraints
  - E.g., common planar model takes 5 equations
    - 3 coordinates x,y,θ
    - 2 constraints to relate translation to rotation (Yamaguchi & Zajac, 1989)
- Can do much better in Simbody using Custom Mobilizer – 1 unconstrained dof

Creating Constraints in SimTK

Peter Eastman

SimTK Workshop, September 25, 2008

A Constrained System

(ExampleGears.cpp)

```
int main() {
   // Create the system.

   MultibodySystem system;
   SimbodyMatterSubsystem matter(system);
   Body::Rigid gearBody(MassProperties(1.0, Vec3(0), Inertia(1)));
   gearBody.addDecoration(Transform(Rotation(0.5*Pi, CoordinateAxis::XCoordinateAxis())),
      DecorativeCylinder(1.0, 0.1));
   MobilizedBody::Pin gear1(matter.updGround(), Transform(Vec3(1, 0, 0)), gearBody, Transform());
   MobilizedBody::Pin gear2(matter.updGround(), Transform(Vec3(-1, 0, 0)), gearBody, Transform());
   Body::Rigid rodBody(MassProperties(1.0, Vec3(0), Inertia(1)));
   rodBody.addDecoration(Transform(Vec3(0, 1, 0)), DecorativeCylinder(0.05, 1.0));
   MobilizedBody::Pin rod(gear2, Transform(Vec3(0, 0.8, 0.1)), rodBody, Transform());
   Constraint::ConstantSpeed(gear1, 0.1);
   Constraint::NoSlip1D(matter.updGround(), Vec3(0), UnitVec3(0, 1, 0), gear1, gear2);
   Constraint::PointOnLine(matter.updGround(), UnitVec3(0, 1, 0), Vec3(0, 0, 0.1), rod, Vec3(0, 2, 0));
   system.updDefaultSubsystem().addEventReporter(new VTKEventReporter(system, 0.05));

   // Initialize the system and state.

   system.realizeTopology();
   State state = system.getDefaultState();

   // Simulate it.

   RungeKuttaMersonIntegrator integ(system);
   TimeStepper ts(system, integ);
   ts.initialize(state);
   ts.stepTo(1000.0);
}
```

ConstantSpeed

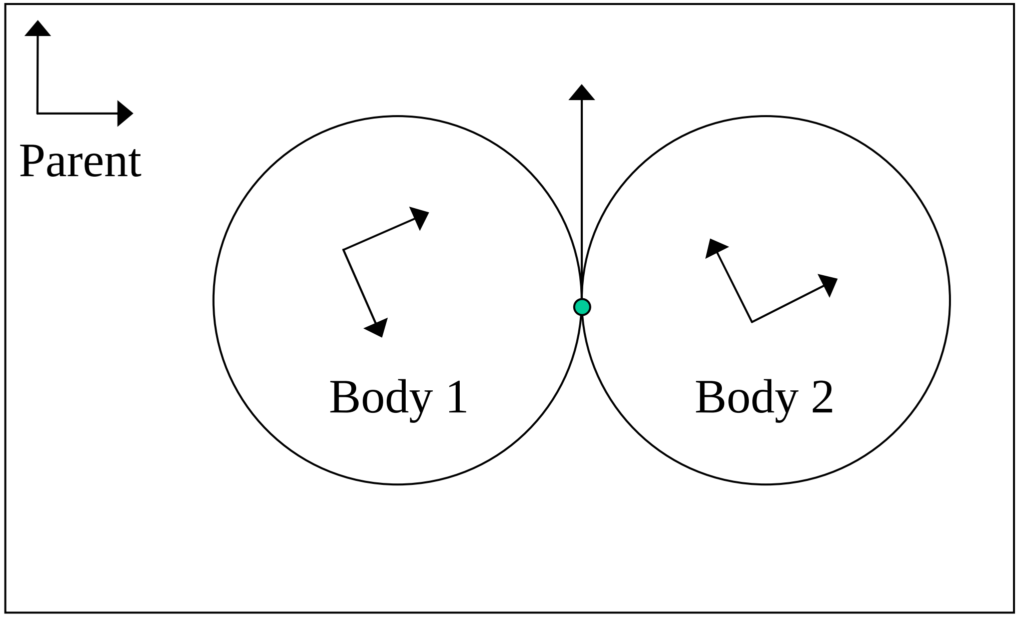
```
Constraint::ConstantSpeed(gear1, 0.1);
```

- Applies to a single generalized speed
- Forces it to have a constant value

NoSlip1D

```
Constraint::NoSlip1D(matter.updGround(), Vec3(0), UnitVec3(0, 1, 0), gear1, gear2);
```

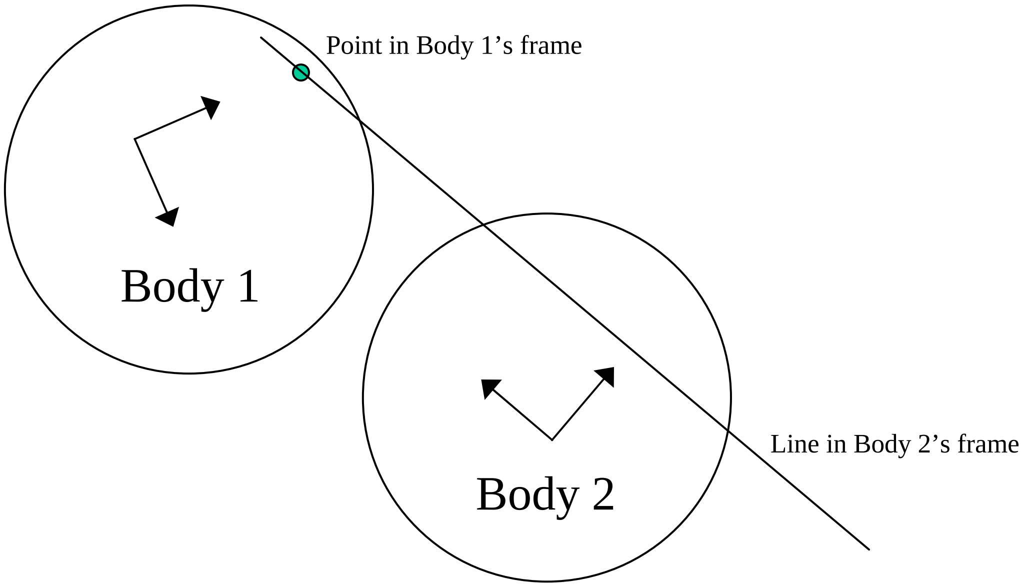
- Useful for rolling bodies
- Both bodies must have equal velocity *at a point in space* in a particular direction



PointOnLine

```
Constraint::PointOnLine(matter.updGround(), UnitVec3(0, 1, 0), Vec3(0, 0, 0.1), rod, Vec3(0, 2, 0));
```

- Requires a point in one body’s frame to remain on a line in another body’s frame



Exercises

- Increase the rotation speed to 0.2 radians/sec
- Constrain the end of the rod to lie on the line x=-1, z=0.1