Computational Robot Dynamics

Part 1: Dynamic Models of Kinematic Trees

Roy Featherstone

ISTITUTO ITALIANO
DI TECNOLOGIA
ADVANCED ROBOTICS

Model Data Structure

A data structure defining a kinematic tree contains the following information:

- the number of bodies (the number of joints is the same)
- connectivity data – how the bodies are connected together
- joint data – type codes and parameters
- geometry data – the location of each joint in each body
- body data – the spatial inertia of each body

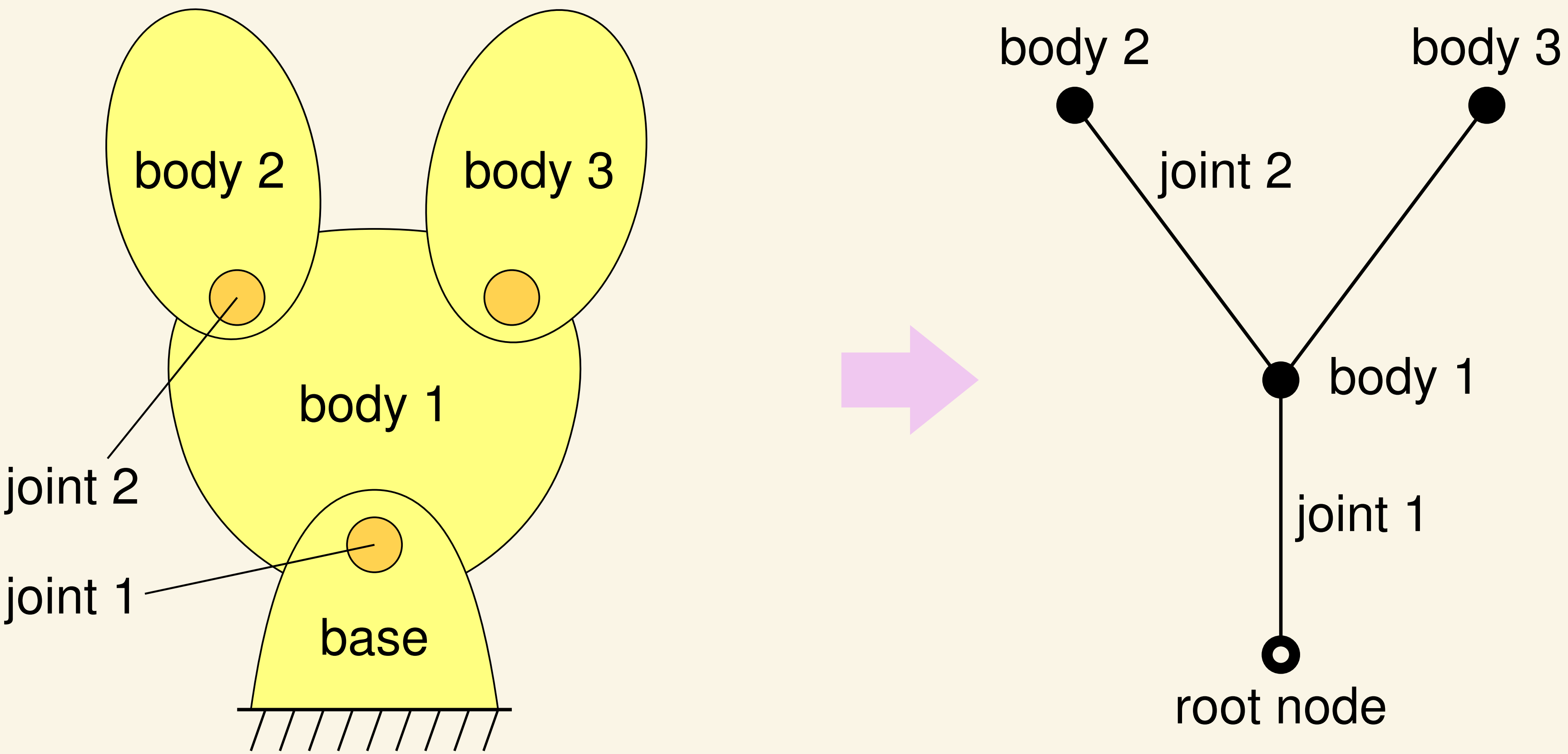
Connectivity Graph

The connections between bodies and joints are described using a connectivity graph in which

- one node represents a fixed base, or fixed reference frame
- this special node is the root node of the graph
- all other nodes represent bodies
- arcs represent joints

The connectivity graph of a kinematic tree is itself a tree.

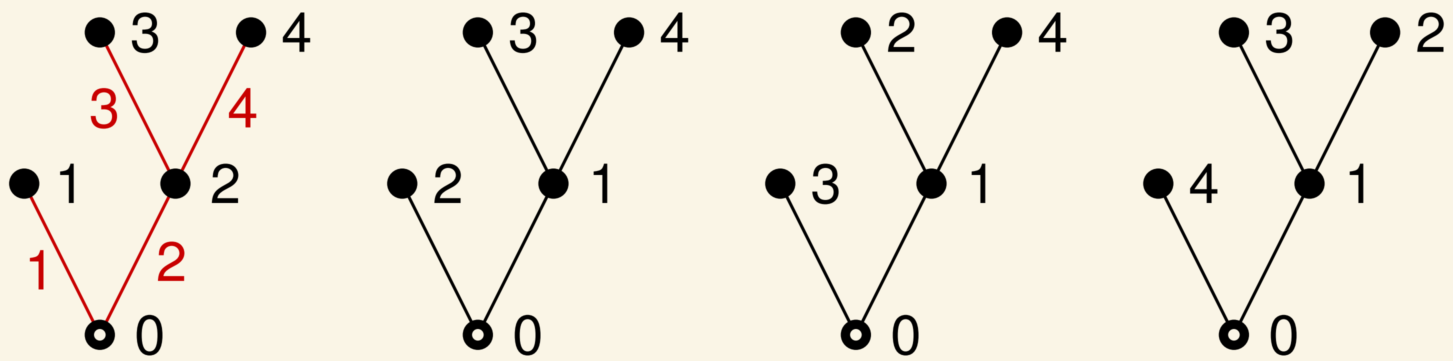
Connectivity Graph — Example



Numbering Scheme

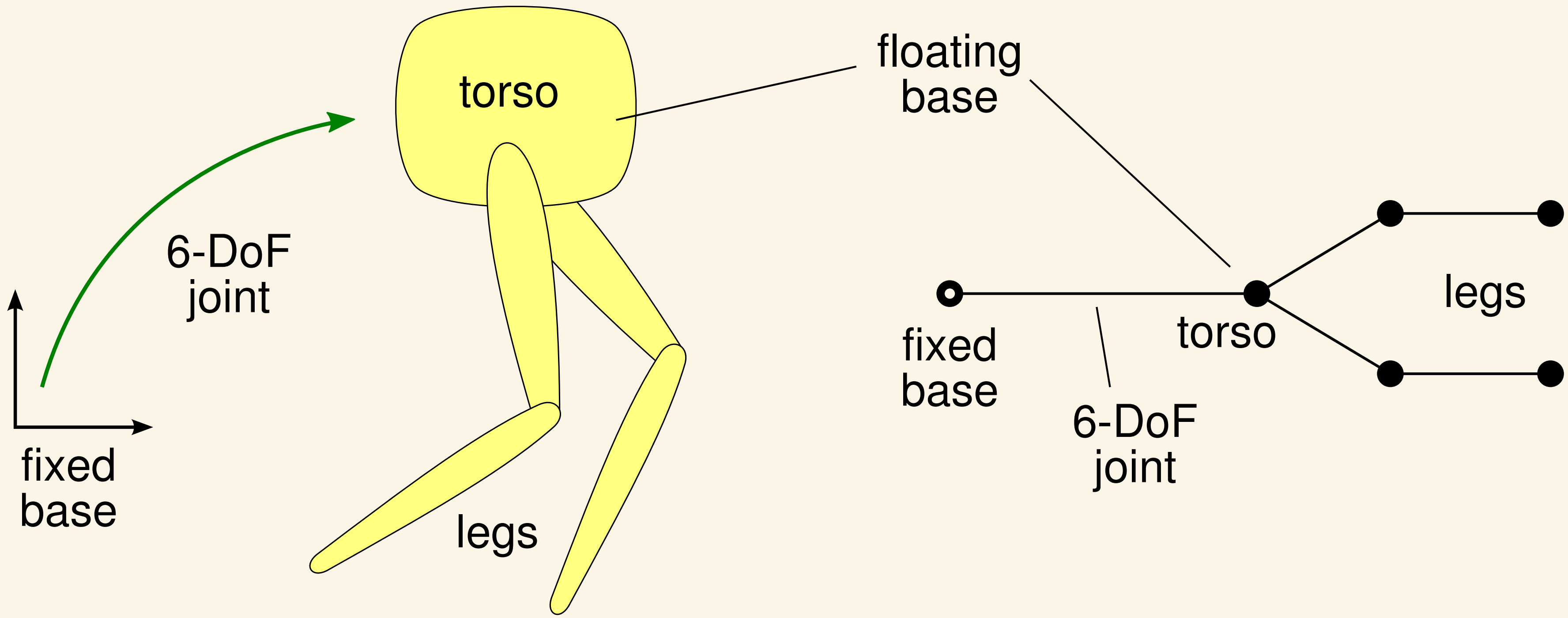
- the root node is numbered 0
- the other nodes are numbered 1 to N in any order such that each node has a higher number than its parent
- arcs are numbered such that arc i connects node i to its parent
- the bodies and joints in the mechanism have the same numbers as the corresponding nodes and arcs in the graph

Examples

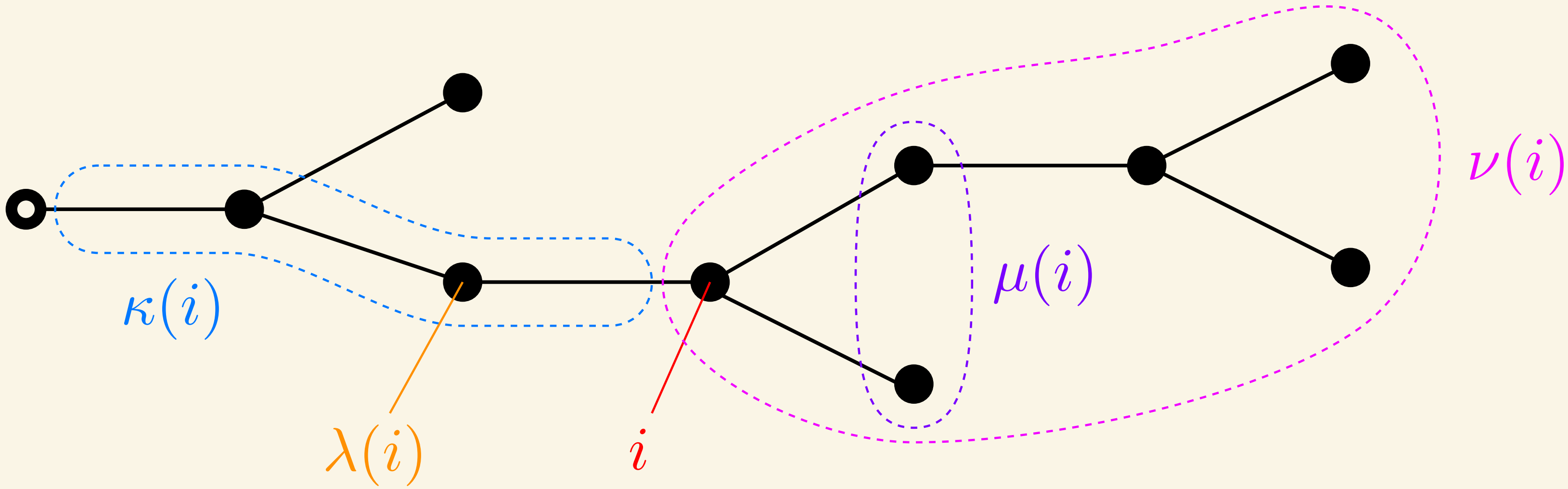


Floating Bases

A mobile robot is connected to a fixed base via a *6-DoF joint* (a joint that does not impose any motion constraints); and the body which is connected directly to this 6-DoF joint is called the *floating base*.



Describing Connectivity



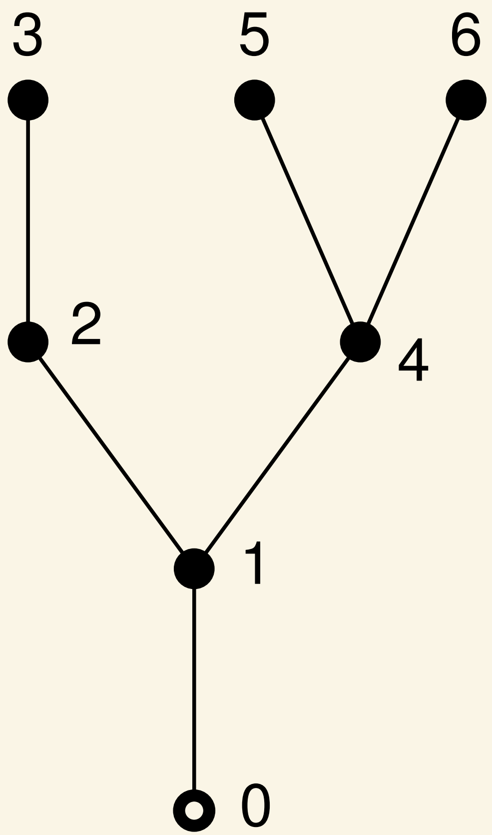
$\kappa(i)$ — all the joints that support body $\boldsymbol{i}$

$\lambda(i)$ — the parent of body $\boldsymbol{i}$

$\mu(i)$ — the children of body $\boldsymbol{i}$

$\nu(i)$ — all the bodies in the subtree supported by joint $\boldsymbol{i}$

Describing Connectivity



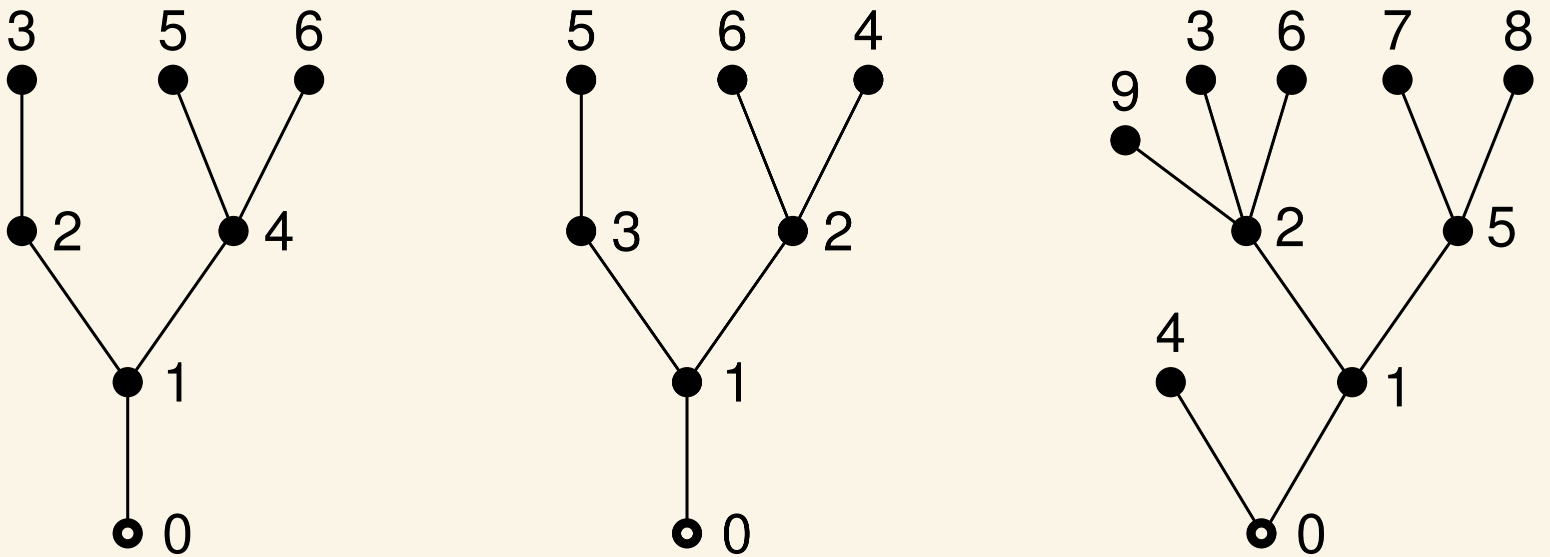
$$\mu(0) = \{1\}$$

$$\lambda(1) = 0 \qquad \mu(1) = \{2, 4\}$$
$$\lambda(2) = 1 \qquad \mu(2) = \{3\}$$
$$\lambda(3) = 2 \qquad \mu(3) = \{\}$$
$$\lambda(4) = 1 \qquad \mu(4) = \{5, 6\}$$
$$\lambda(5) = 4 \qquad \mu(5) = \{\}$$
$$\lambda(6) = 4 \qquad \mu(6) = \{\}$$

$$\kappa(1) = \{1\} \qquad \nu(1) = \{1, 2, 3, 4, 5, 6\}$$
$$\kappa(2) = \{1, 2\} \qquad \nu(2) = \{2, 3\}$$
$$\kappa(3) = \{1, 2, 3\} \qquad \nu(3) = \{3\}$$
$$\kappa(4) = \{1, 4\} \qquad \nu(4) = \{4, 5, 6\}$$
$$\kappa(5) = \{1, 4, 5\} \qquad \nu(5) = \{5\}$$
$$\kappa(6) = \{1, 4, 6\} \qquad \nu(6) = \{6\}$$

Describing Connectivity

We can collect the parent numbers into a single array of integers, $\lambda = [\lambda(1), \lambda(2), \ldots, \lambda(N)]$, which is called the *parent array*. This array provides a complete description of both the connectivity and the numbering scheme.



$\lambda = [0, 1, 2, 1, 4, 4]$ $\quad$ $\lambda = [0, 1, 1, 2, 3, 2]$ $\quad$ $\lambda = [0, 1, 2, 0, 1, 2, 5, 5, 2]$

Describing Connectivity

- As λ provides a complete description of the connectivity, it follows that the sets $\kappa(i)$, $\mu(i)$ and $\nu(i)$ can all be calculated directly from λ.

- Most algorithms only need λ.

- Many algorithms rely on the property $0 \leq \lambda(i) < i$.

Joint Data

For each joint, the following data is required:

- a *type code*, which identifies the type of joint (e.g. revolute or prismatic)
- zero or more *joint parameters*, depending on the joint type (e.g. the pitch of a helical joint)

Joint Model

The joint data is used to calculate the *joint transform* and the *motion subspace matrix*.

joint transform

function in spatial_v2

joint position variable

```
[Xj,S] = jcalc(jtyp,q);
```

motion subspace matrix

type code and parameters

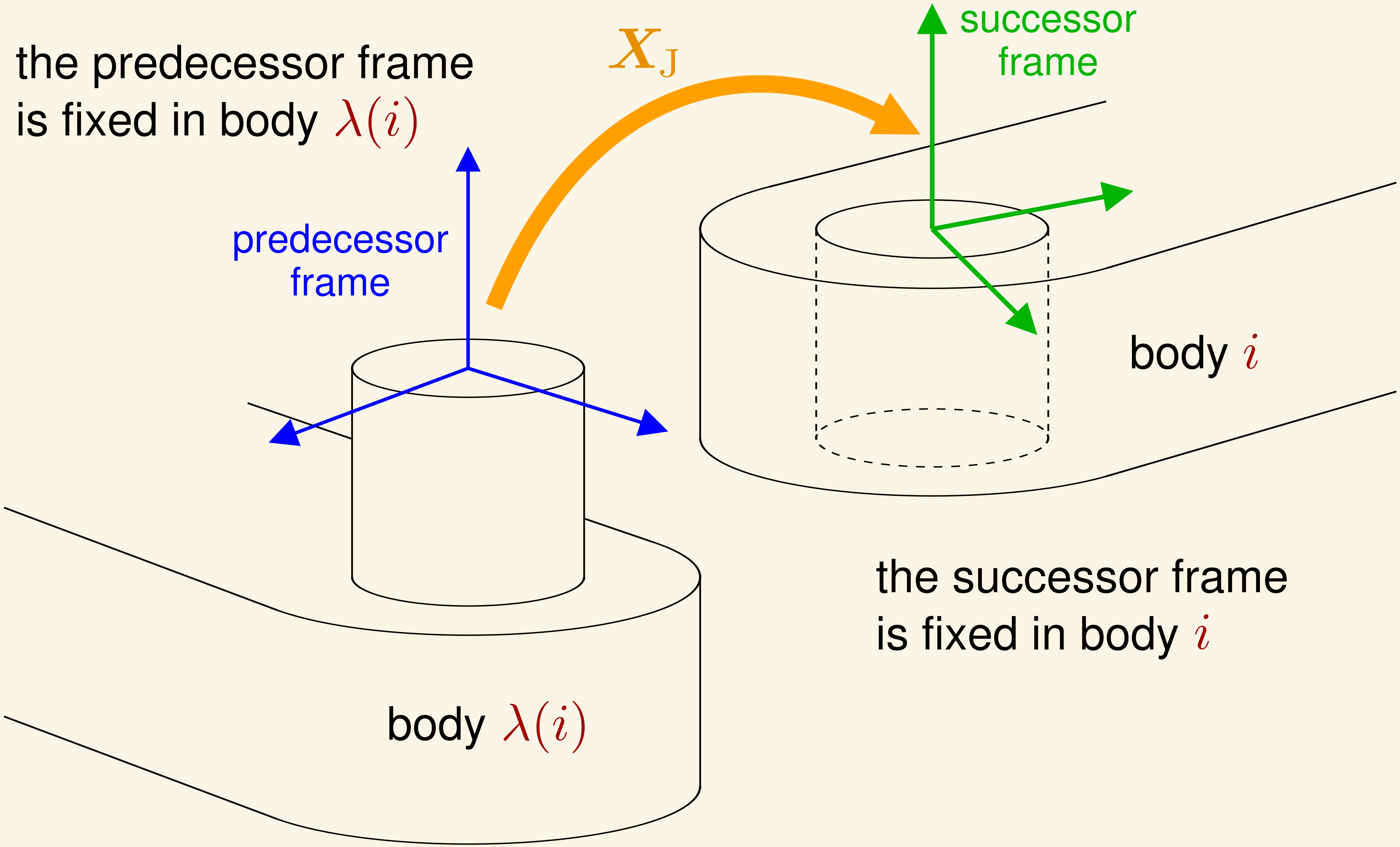
Joint Model

The purpose of a joint is to allow relative movement between two bodies. Therefore, for each joint, we introduce a *predecessor frame* and a *successor frame*, each fixed in one of the two bodies. The joint transform is then defined as follows:

> *The joint transform is the coordinate transform from the predecessor frame to the successor frame.*

For joint i, the predecessor frame is fixed in body $\lambda(i)$, and the successor frame is fixed in body i.

Joint Model


the predecessor frame
is fixed in body $\lambda(i)$
X_{J}
successor
frame
predecessor
frame
body i
the successor frame
is fixed in body i
body $\lambda(i)$

Joint Model

The joint motion subspace is a matrix that maps the joint velocity variable to a spatial velocity:

$$\boldsymbol{v}_{\mathrm{J}} = \boldsymbol{S}\,\dot{\boldsymbol{q}}$$

where $\boldsymbol{v}_{\mathrm{J}}$ is, by definition, the velocity of the successor body relative to the predecessor body. So, for joint i,

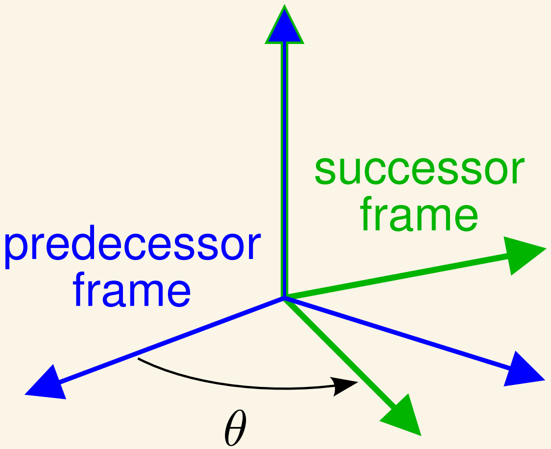
$$\boldsymbol{v}_{\mathrm{J}i} = \boldsymbol{v}_i - \boldsymbol{v}_{\lambda(i)}$$

hence

$$\boldsymbol{v}_i = \boldsymbol{v}_{\lambda(i)} + \boldsymbol{S}_i\,\dot{\boldsymbol{q}}_i$$

Joint Model — Examples

revolute joint on *z* axis

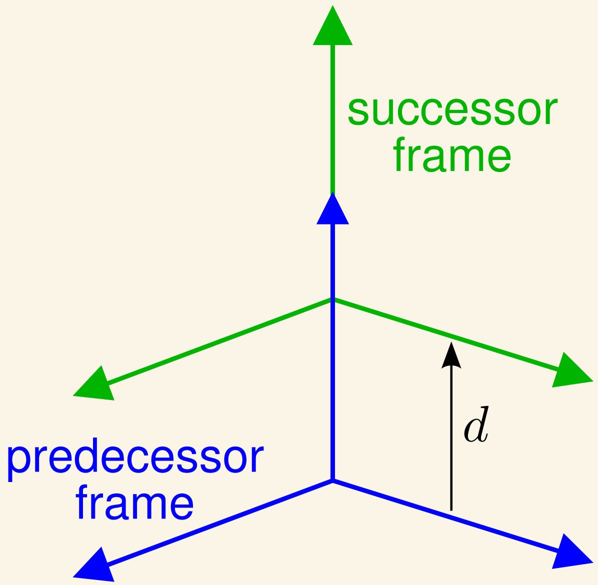


$$\boldsymbol{X}_{\mathrm{J}}(\theta) = \begin{bmatrix} \boldsymbol{E} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{E} \end{bmatrix}$$

$$\boldsymbol{E} = \begin{bmatrix} c & s & 0 \\ -s & c & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \boldsymbol{S} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$c = \cos(\theta)\,, \quad s = \sin(\theta)$$

prismatic joint in *z* direction



$$\boldsymbol{X}_{\mathrm{J}}(d) = \begin{bmatrix} \mathbf{1} & \mathbf{0} \\ -\boldsymbol{r}\times & \mathbf{1} \end{bmatrix}$$

$$\boldsymbol{r}\times = \begin{bmatrix} 0 & -d & 0 \\ d & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad \boldsymbol{S} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Geometry Data

The geometry data defines the location of each joint relative to the *body coordinate frames* of its predecessor and successor bodies. The first step is therefore to define the body coordinate frames. We do this using the following convention:

1. The body coordinate frame of body i is defined to be the successor frame of joint i.

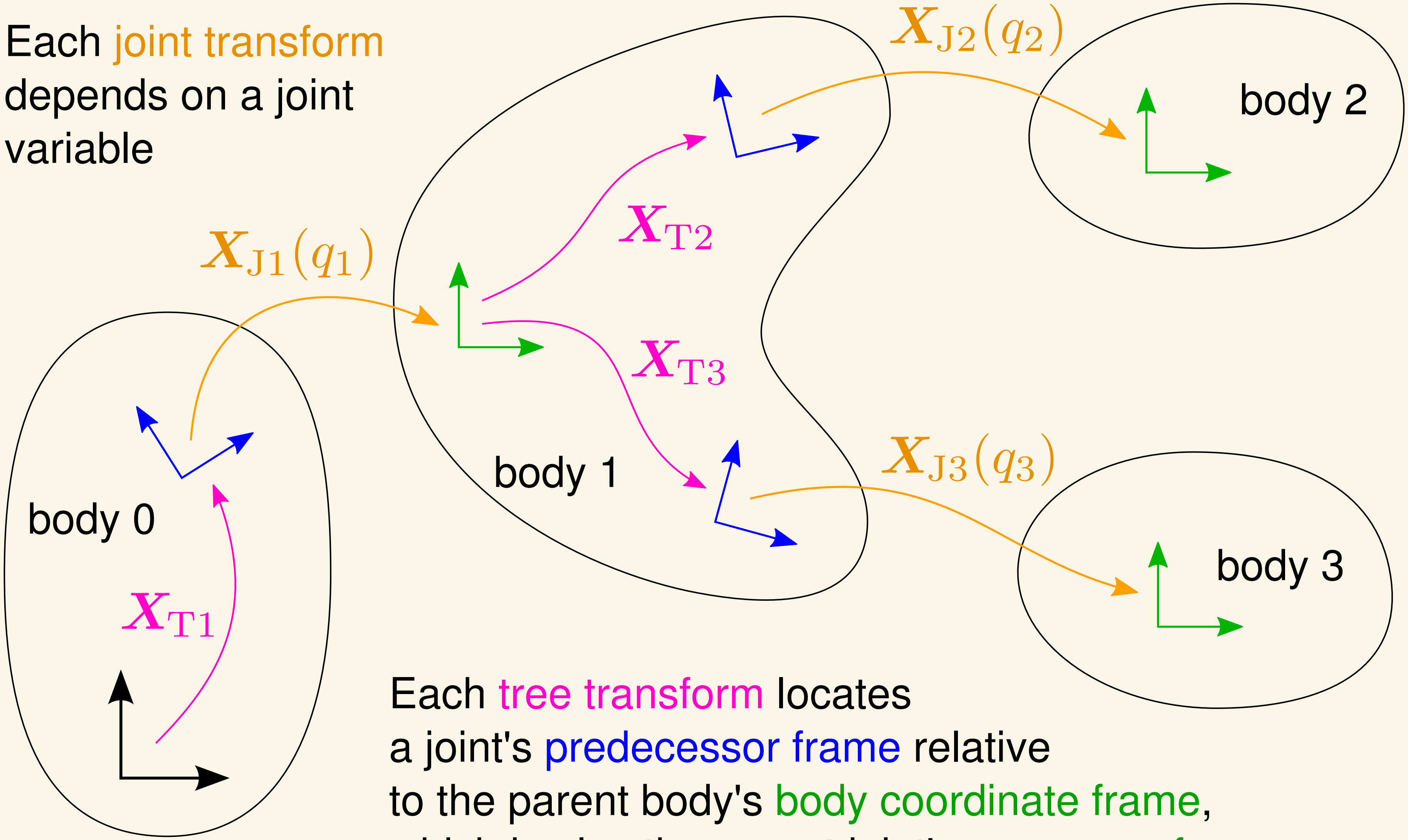
2. The body coordinate frame of body 0 is the reference frame.

Geometry Data

Having defined the body coordinate frames, the geometry data consists of one *tree transform* per joint, defined as follows:

- The tree transform for joint i is the coordinate transform from the body coordinate frame of body $\lambda(i)$ to the predecessor frame of joint i.

Each joint transform
depends on a joint
variable



Each tree transform locates
a joint's predecessor frame relative
to the parent body's body coordinate frame,
which is also the parent joint's successor frame
(or the reference frame in the case of body 0)

Body Data

For *dynamics*, the only data required is the *spatial inertia* of each body, expressed in body coordinates.

However, if you want to see the robot then you must also supply drawing instructions for at least some of the bodies; and if you want to simulate contact dynamics (not supported in spatial_v2) then you will also have to describe the shapes of the bodies (e.g. by importing CAD files).

Summary

A model data structure defining a kinematic tree contains the following information:

	field names in spatial_v2
• the number of bodies	`.NB`
• connectivity data – the parent array (λ)	`.parent`
• joint data – type codes and parameters	`.jtype`
• geometry data – the list of tree transforms	`.Xtree`
• body data – the spatial inertia of each body	`.I`

Now try the question set for Part 1 -- Modelling