15-494/694: Cognitive Robotics

Dave Touretzky

Lecture 8:

Review, and SLAM

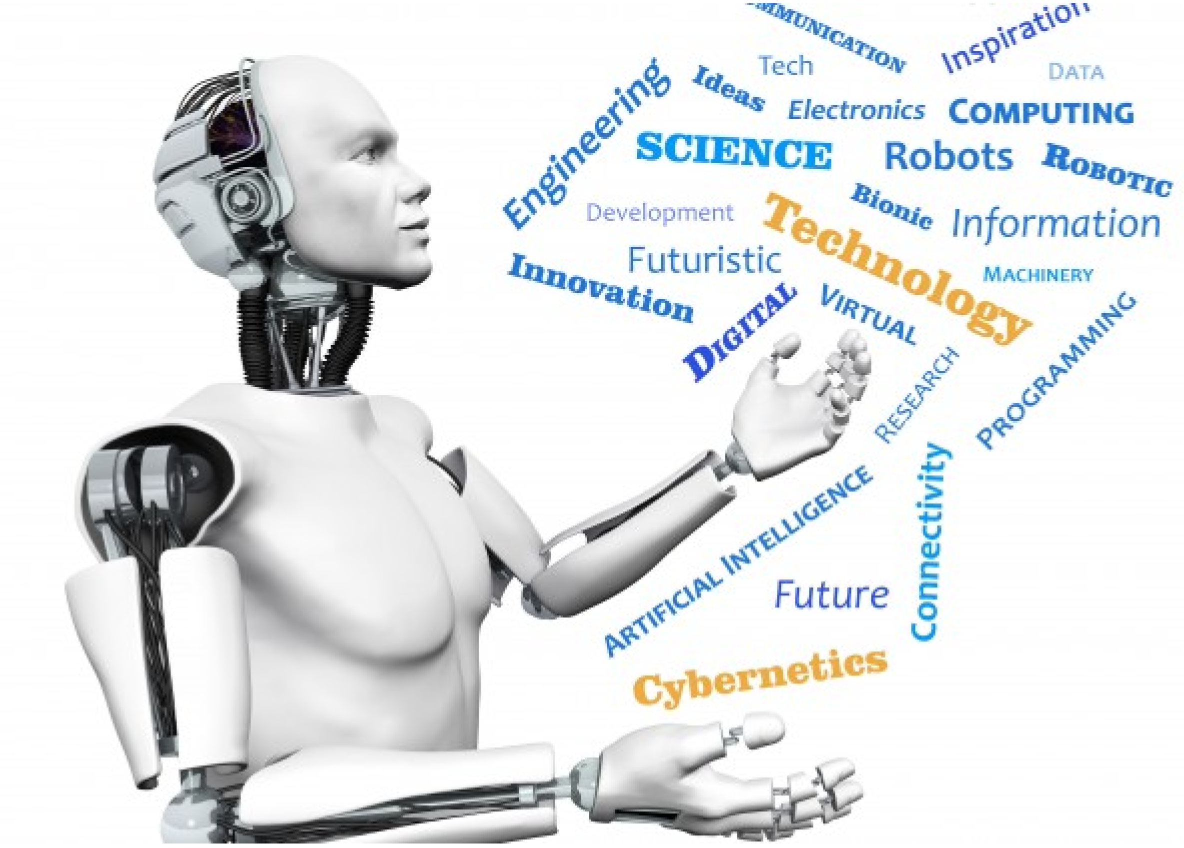


Image from http://www.futuristgerd.com/2015/09/10

Kinematics Again

- Why we need a kinematics engine (Tower of Hanoi demo).

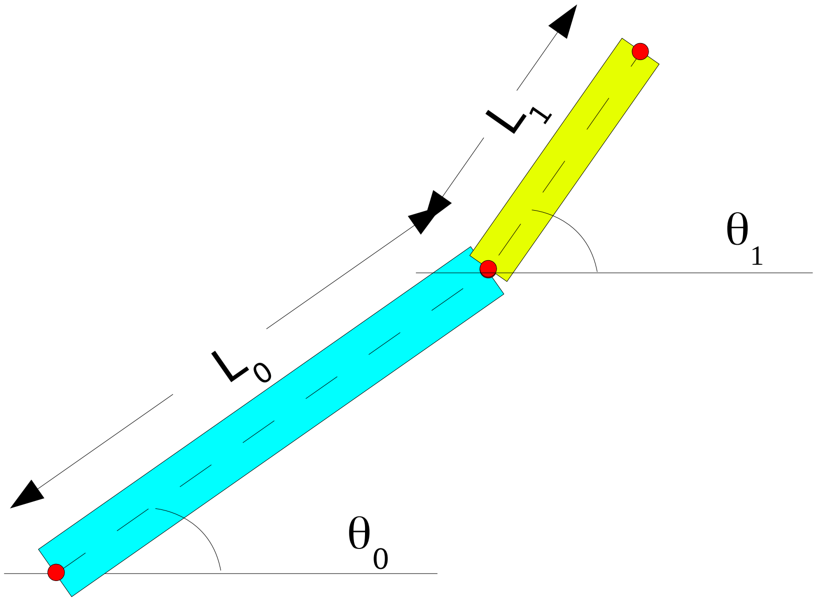
- But we need path planning too.

Kinematics Review

- What is a kinematic chain?

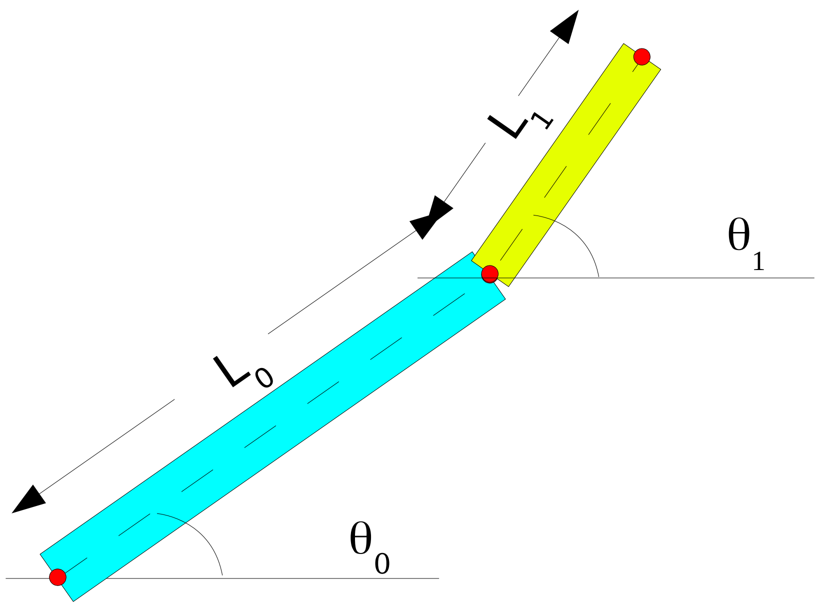
Kinematics Review

- What is a kinematic chain?
  - An alternating sequence of joints and links.
  - The transformation between reference frame *i* and reference frame *i+1* is described by DH parameters.



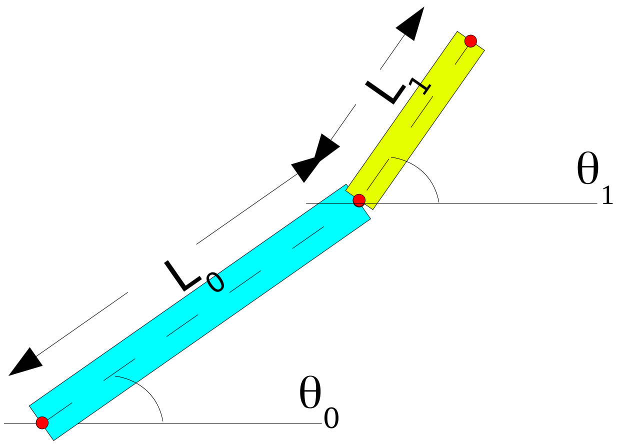
Kinematics Review (2)

- What defines a reference frame?



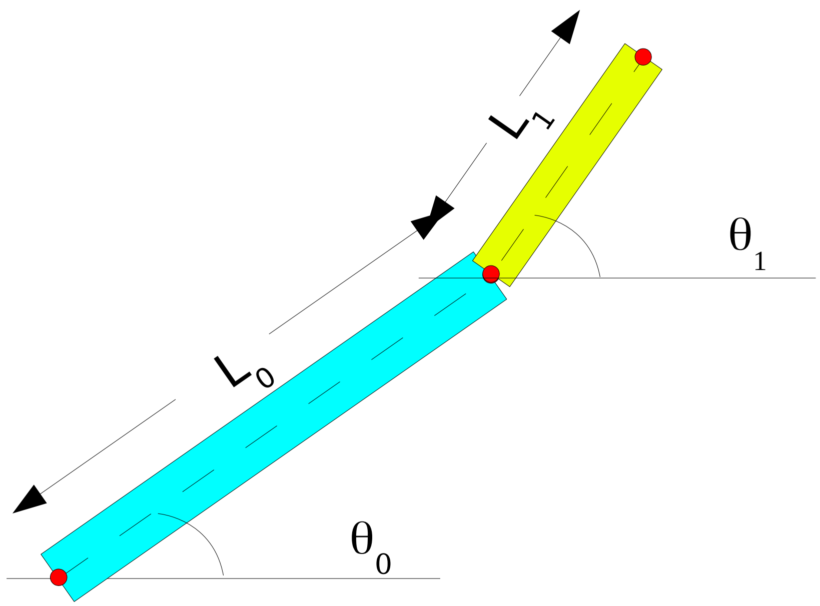
Kinematics Review (2)

- What defines a reference frame?
  - An origin (x,y,z) and a 3D orientation.
  - The orientation can be described in terms of a 3D rotation matrix.
  - We could also use Euler angles, or a quaternion.



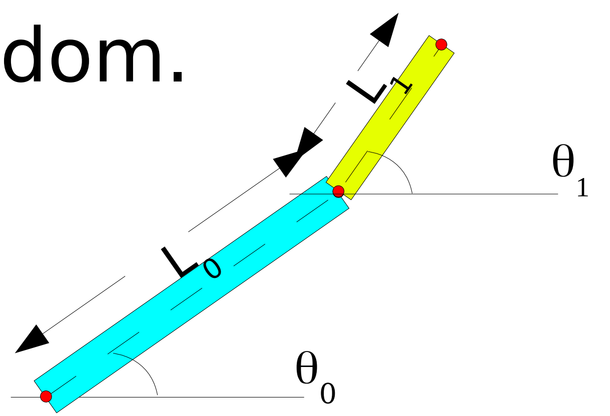
Kinematics Review (3)

- Why do we need a dummy joint between the head reference frame and the camera reference frame in VEX AIM?



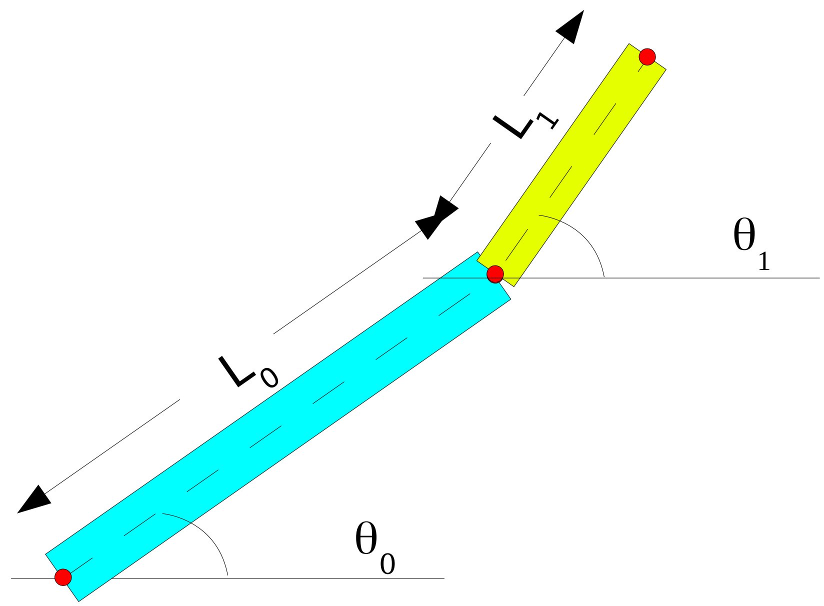
Kinematics Review (3)

- Why do we need a dummy joint between the head reference frame and the camera reference frame in VEX AIM?
  - The four DH parameters for one joint don't provide enough degrees of freedom to let us control both the orientation and the origin of the new reference frame.
  - The dummy joint adds four additional degrees of freedom.



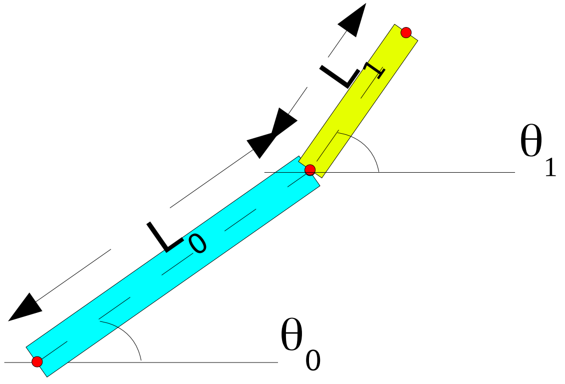
Kinematics Review (4)

- How do we move from the joint *i* reference frame to the link *i* reference frame?



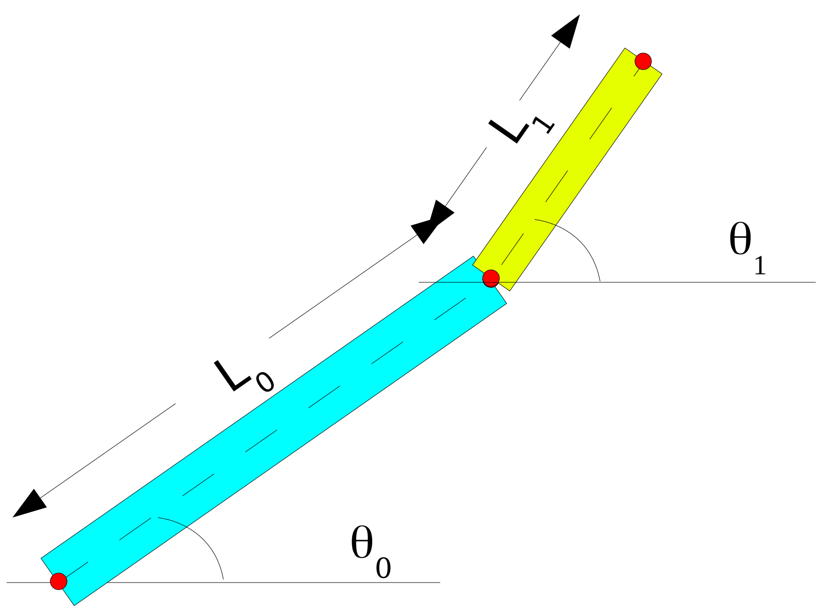
Kinematics Review (4)

- How do we move from the joint *i* reference frame to the link *i* reference frame?
  - The link reference frame rotates with the joint, while the joint reference frame remains fixed.
  - Use joint.apply_q() to apply the rotation. This returns a transformation matrix.



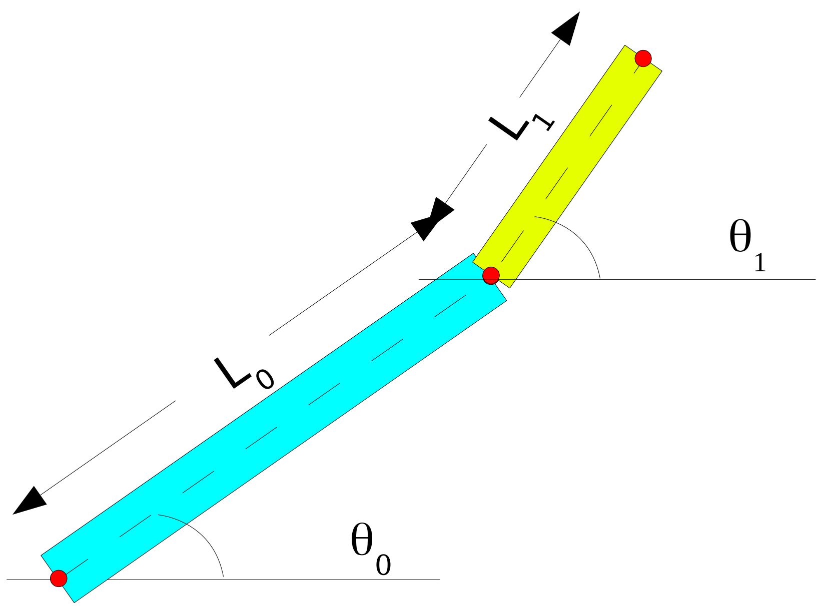
Kinematics Review (5)

- How do we move from the link *i* reference frame to the joint *i+1* reference frame?



Kinematics Review (5)

- How do we move from the link *i* reference frame to the joint *i+1* reference frame?
  - Apply the (constant) transformation matrix described by the DH parameters.



How To Build A World Map

- SLAM: Simultaneous Localization and Mapping algorithm.
- Each particle stores:
  - a hypothesis about the robot's location
  - a hypothesis about the map, e.g., a set of landmark identities and locations.
- Particles score well if:
  - Landmark locations match the sensor values predicted by the robot's location.
- Robot location is jittered by the motion model. This jitters the landmark locations.

First SLAM Video

- SLAM works well even when landmarks are ambiguous, such as identical markers.
- Reason: updating the particle weights based on sensor readings after movement applies strong constraints on possible robot locations.

Brenner's Particle Filter Course

- Part A: introduce robot, odometry, laser scanner as distance sensor.
- Part B: using laser sensor data to estimate landmark positions.
- Part C: Bayes filter: predict (motion model) and correct (sensor model).
- Part D: Kalman filter (Bayes with gaussian noise model) and Extended Kalman Filter (arbitrary noise model; approximate with Taylor series). Error ellipses.

Brenner’s Particle Filter Course

- Part E: particle filters (non-parametric alternative to EKF; arbitrary distributions including multi-modal).

SLAM:

- Part F: EKF SLAM: use EKF for both position and landmarks.
- Part G: Particle SLAM: use particle filter for position and EKF for landmarks.

The vex-aim-tools Particle Filter

- Defined in aim_fsm/particle.py
  - Versions with and without SLAM
  - Default is SLAMParticleFilter
  - Uses ArUco markers or walls defined by ArUco markers as landmarks, but you can control this.

```
robot.particle_filter

p0 = robot.particle_filter.particles[0]

p0.landmarks
```

Representation of a Landmark

Assume the robot is seeing Wall 1.

wall1 = p0.landmarks['Wall-1']

- wall1[0] is a column vector $[x,y]^T$ giving the position of the landmark on the map.
- wall1[1] is the landmark's orientation, theta.
- wall1[2] is the covariance matrix Σ used in the EKF update equation.

How Do We Display the Map?

- Every particle has a weight.
- Use the map from the most highly weighted particle.
- This means the map will sometimes “jump” to a new configuration if the highest weighted particle changes.

FSM Debugging Strategies

- What is my state machine doing?
  - simple_cli: “show active”
  - tracefsm(1) or tracefsm(4)
- What event did I receive?
  - print(event)
  - print(dir(event))
- Can I pause here and examine stuff?
  - `MyNode1() =TM=> MyNode2()`

PythonDebugging

```
import pdb


def foo(x):
    print(f’x = {x}’)
    breakpoint()
    print(f’x is now {x}’)
```

Use “continue” to continue from a breakpoint.

When to Call super().start()

When setting parameters (e.g., turn angle), they must be set before super().start(event)

When posting a completion or failure event, you must do so after having called super().start(event) so that the outgoing transitions are activated and listening.

Getting this wrong leads to common bugs.