AutoRob

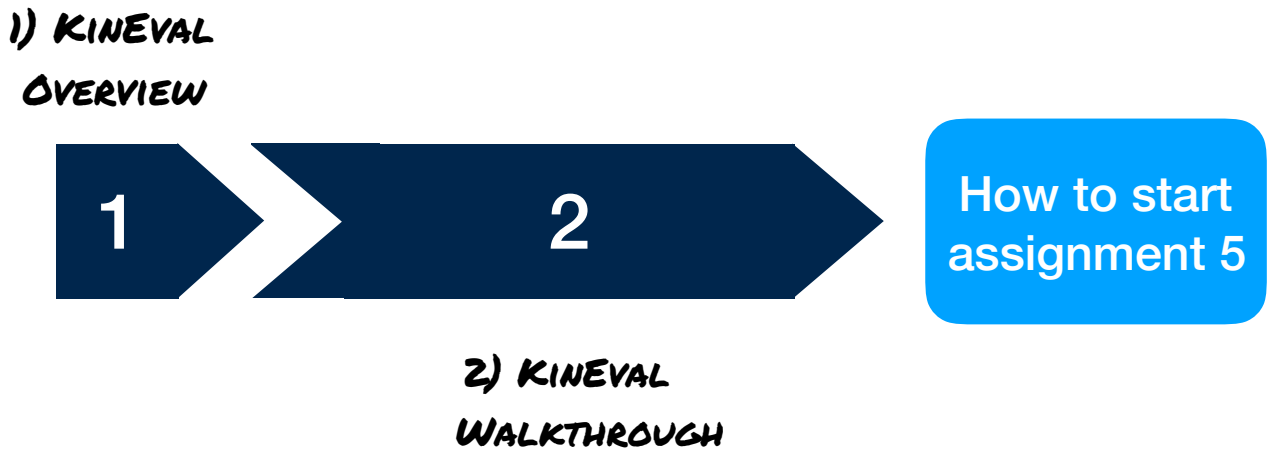
Introduction to Autonomous Robotics
Michigan EECS 367

Robot Kinematics and Dynamics
Michigan ME 567 EECS 567 ROB 510

Fall 2019

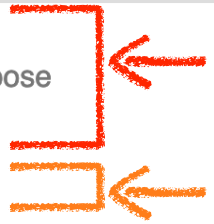
EECS 367 Lab:
KinEval IK Control Flow and Parameters

Lab Takeaways



Forward Kinematics Overview

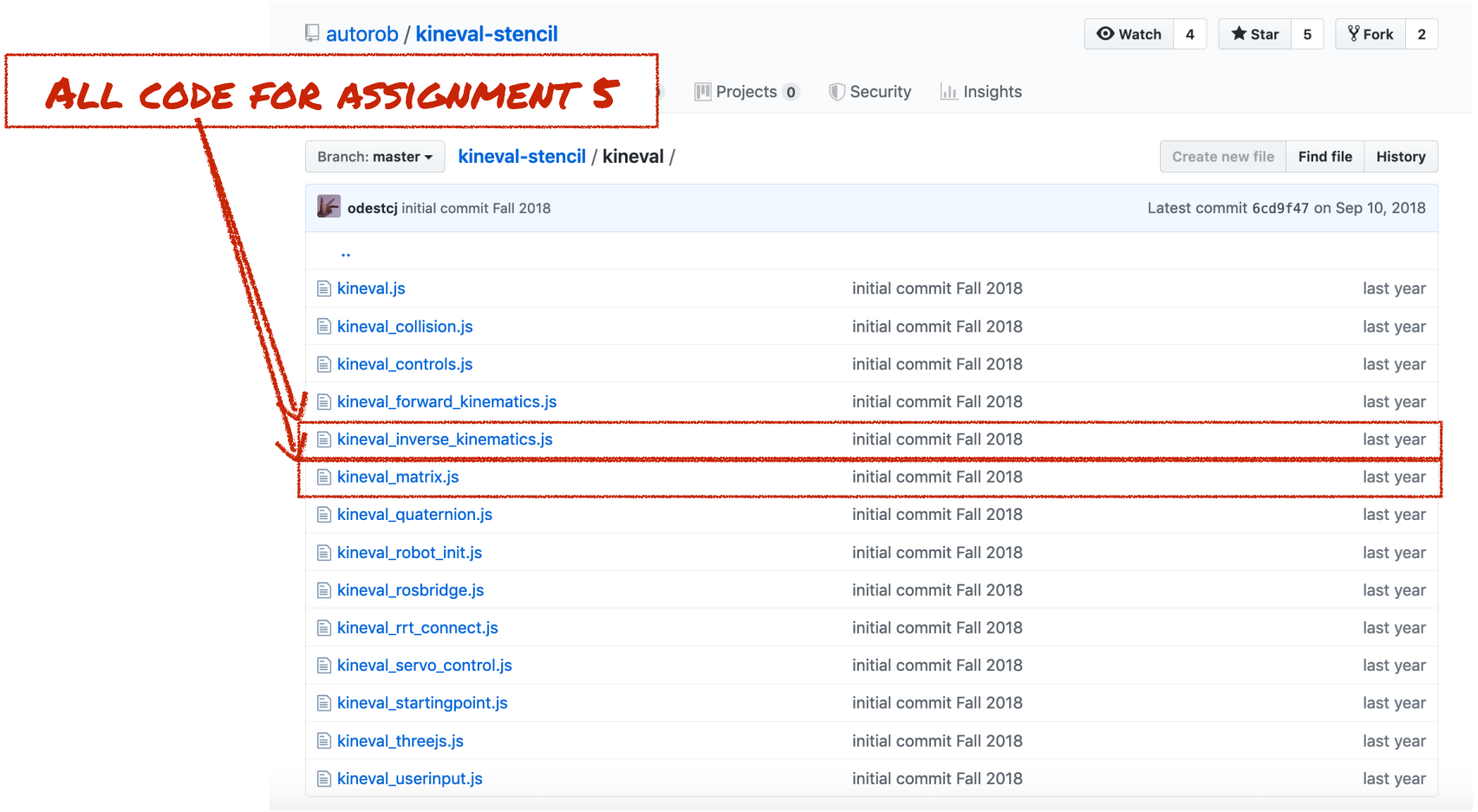
		Assignment 5: Inverse Kinematics
6	All	Manipulator Jacobian
3	All	Gradient descent with Jacobian transpose
3	All	Jacobian pseudoinverse
6	Grad	Euler angle conversion



Features assigned to all sections

Features assigned to graduate sections

KinEval Overview



kineval_inverse_kinematics.js

kineval_inverse_kinematics.js

```
19  kineval.robotInverseKinematics = function robot_inverse_kinematics(endeffector_target_world, endeffector_joint,
20
21      // compute joint angle controls to move location on specified link to Cartesian location
22      if ((kineval.params.update_ik)||(kineval.params.persist_ik)) {
23          // if update requested, call ik iterator and show endeffector and target
24          kineval.iterateIK(endeffector_target_world, endeffector_joint, endeffector_position_local);
25          if (kineval.params.trial_ik_random.execute)
26              kineval.randomizeIKtrial();
27          else // KE: this use of start time assumes IK is invoked before trial
28              kineval.params.trial_ik_random.start = new Date();
29      }
30
31      kineval.params.update_ik = false; // clear IK request for next iteration
32  }
33
34  kineval.randomizeIKtrial = function randomIKtrial () {
35
36      // update time from start of trial
37      cur_time = new Date();
38      kineval.params.trial_ik_random.time = cur_time.getTime()-kineval.params.trial_ik_random.start.getTime();
39
40      // STENCIL: see instructor for random time trial code
41  }
42
43  kineval.iterateIK = function iterate_inverse_kinematics(endeffector_target_world,
44                                                          endeffector_joint,
45                                                          endeffector_position_local) {
46
47      // STENCIL: implement inverse kinematics iteration
48  }
49
50
```

IMPLEMENT `iterateIK()` SUCH THAT EACH JOINT ALONG THE ENDEFFECTOR PATH RESULTS WITH AN UPDATE TO ITS `.control` TERM

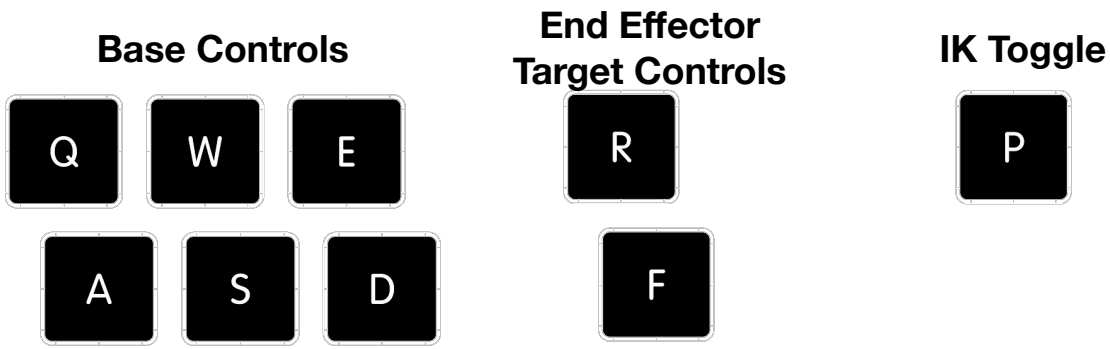
KinEval IK Parameters

- `iterate_inverse_kinematics(...)`
  - `endeffector_target_world`
    - Target pose of end effector for IK, `.position` and `.orientation`
  - `endeffector_joint`
    - String name of joint connected to end effector
  - `endeffector_position_local`
    - Position of end effector with respect to local frame

- `kineval.params.ik_steplength`
  - Size of step to take along configuration gradient when updating control
- `kineval.params.ik_pseudoinverse`
  - Boolean flag denoting which method to use (Jacobian transpose or pseudo inverse)

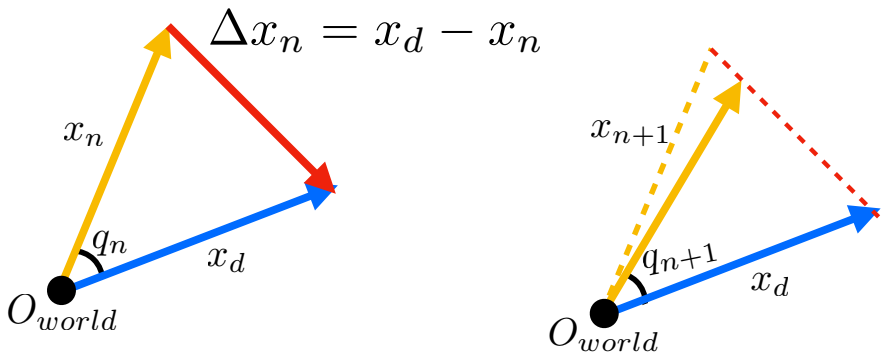
Performance Validation

- `kineval.randomizeIKtrial()`
  - Source code will be provided in assignment slack channel
  - Grad-extension points for reaching at least 100 targets in 60 seconds
- Inverse Kinematics will react in realtime
  - IK will account for manual adjustments to robot base or joint angles
  - Also for any modification to end effector target



kineval_quaternion.js

State



Update

$$\Delta q_n = J(q_n)^{-1} \Delta x_n$$

$$q_{n+1} = q_n + \gamma \Delta q_n$$

KinEval Variables

x_d ~ `endeffector_target_world`

q_n ~ `robot.joints`

p^{x_n} ~ `endeffector_position_local`

$x_n = T^O_{x_n} p^{x_n}$

γ ~ `kineval.params.ik_steplength`