

EECS 367 & ROB 320 Lab

KinEval Setpoints Walkthrough

Administrative

- Assignment #3: Forward Kinematics
 - Due Friday, February 18, 11:59pm

Lab Takeaways

1. KinEval overview
 2. KinEval walkthrough
 3. Finding setpoints demo
- How to start Assignment 4

Dance Controller Overview

Assignment 4: Dance Controller

6

Quaternion joint rotation

2

Interactive base control

2

Pose setpoint controller

2

Dance FSM

KinEval Overview

autorob / **kineval-stencil**

Watch 4

Star 8

Fork 3

<> Code

Issues 1

Pull requests

Actions

Projects

Wiki

Security

Insights

master 1 branch 0 tags

Go to file

Add file

Code

ohseejay Merge pull request #3 from cxt98/master b8f51ea 8 days ago 9 commits

js	initial commit Fall 2018	2 years ago
kineval	initial commit Fall 2018	2 years ago
project_pathplan	Adds refactored stencil files for project 1.	16 days ago
project_pendularm	add refactor of assignment2, tested with CI grader	12 days ago
robots	initial commit Fall 2018	2 years ago
tutorial_heapsort	initial commit Fall 2018	2 years ago
tutorial_js	initial commit Fall 2018	2 years ago
worlds	initial commit Fall 2018	2 years ago
LICENSE	add refactor of assignment2, tested with CI grader	12 days ago
README.md	initial commit Fall 2018	2 years ago
home.html	initial commit Fall 2018	2 years ago

About

Stencil code for KinEval (Kinematic Evaluator) for robot control, kinematics, decision, and dynamics in JavaScript/HTML5

Readme

View license

Releases

No releases published

Packages

No packages published

Contributors 4

All code for assignment 4

KinEval Overview

autorob / **kineval-stencil** Watch 5 Star 9

<> Code Issues 1 Pull requests Actions Projects Wiki Security Insights

master kineval-stencil / kineval / Go to file Add file

zhezhou1993 Factorize kineval stencil for FK problems, fix bugs in previous version 70d8e4b 9 days ago History

..		
kineval.js	initial commit Fall 2018	2 years ago
kineval_collision.js	initial commit Fall 2018	2 years ago
kineval_controls.js	initial commit Fall 2018	2 years ago
kineval_forward_kinematics.js	initial commit Fall 2018	2 years ago
kineval_inverse_kinematics.js	initial commit Fall 2018	2 years ago
kineval_matrix.js	Factorize kineval stencil for FK problems, fix bugs in previous version	9 days ago
kineval_quaternion.js	Factorize kineval stencil for FK problems, fix bugs in previous version	9 days ago
kineval_robot_init.js	Factorize kineval stencil for FK gradining	9 days ago
kineval_robot_init_joints.js	Factorize kineval stencil for FK gradining	9 days ago
kineval_rosbridge.js	initial commit Fall 2018	2 years ago
kineval_rrt_connect.js	initial commit Fall 2018	2 years ago
kineval_servo_control.js	initial commit Fall 2018	2 years ago
kineval_startingpoint.js	initial commit Fall 2018	2 years ago
kineval_threejs.js	initial commit Fall 2018	2 years ago
kineval_userinput.js	initial commit Fall 2018	2 years ago

All code for assignment 4

kineval_forward_kinematics.js Revisited

kineval_forward_kinematics.js

```
18
19  kineval.robotForwardKinematics = function robotForwardKinematics () {
20
21      if (typeof kineval.buildFKTransforms === 'undefined') {
22          textbox.innerHTML = "forward kinematics not implemented";
23          return;
24      }
25
26      // STENCIL: implement kineval.buildFKTransforms();
27
28  }
29
30      // STENCIL: reference code alternates recursive traversal over
31      //   links and joints starting from base, using following functions:
32      //   traverseFKBase
33      //   traverseFKLink
34      //   traverseFKJoint
35      //
```

For each joint, incorporate
.axis and .angle within
forward kinematics. You will
then be able to control joints!

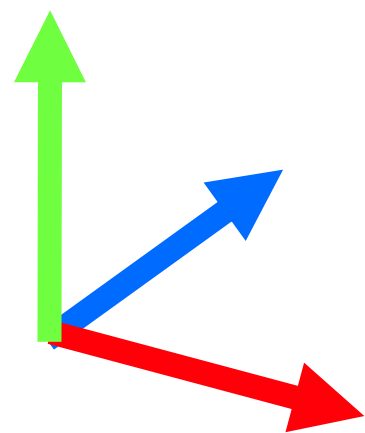
kineval_quaternion.js

kineval_quaternion.js

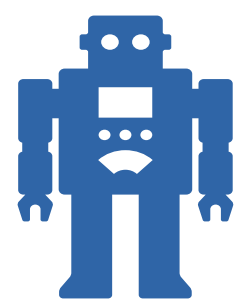
```
1  //////////////////////////////////////
2  /////      QUATERNION TRANSFORM ROUTINES
3  //////////////////////////////////////
4
5  // STENCIL: reference quaternion code has the following functions:
6  //   quaternion_from_axisangle
7  //   quaternion_normalize
8  //   quaternion_to_rotation_matrix
9  //   quaternion_multiply
10
11 // **** Function stencils are provided below, please uncomment and implement them ****//
12
13 // kineval.quaternionFromAxisAngle = function quaternion_from_axisangle(axis,angle) {
```

Define quaternion helper functions
→ Create a joint's rotation matrix
from any axis-angle pair

Joint frame without control

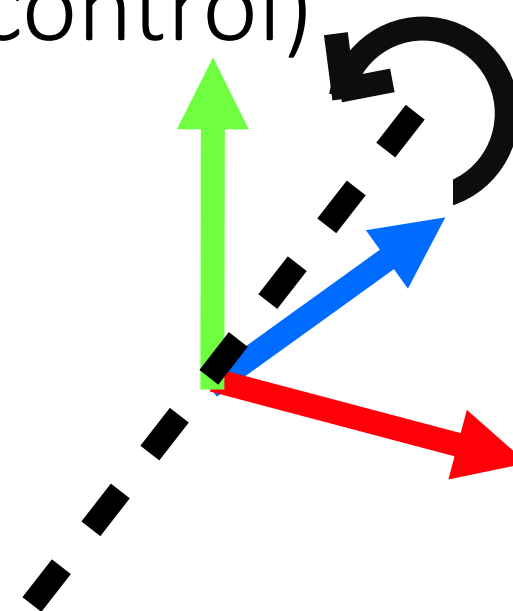


Rest of forward
kinematics

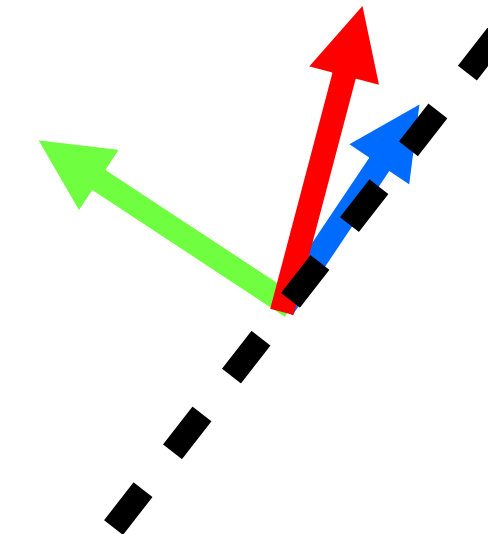


joint.angle
(dynamic w/ control)

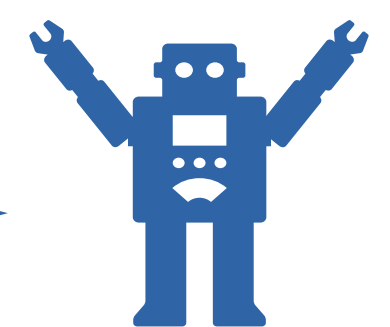
joint.axis
(constant)



Joint frame with control



Rest of forward
kinematics



kineval_controls.js

kineval_controls.js

```
19 kineval.applyControls = function robot_apply_controls(curRobot) {  
20     // apply robot controls to robot kinematics transforms and joint angles, then zero controls  
21     // includes update of camera position based on base movement  
22  
23     // update robot configuration from controls  
24     for (x in curRobot.joints) {  
25  
26         // update joint angles  
27         if ( (typeof curRobot.joints[x].type !== 'undefined')  
28             || (typeof curRobot.joints[x].type !== 'fixed') ) {  
29  
30             if (isNaN(curRobot.joints[x].control))  
31                 console.warn("kineval: control value for " + x + " is a nan");  
32  
33             curRobot.joints[x].angle += curRobot.joints[x].control;  
34         }  
35  
36         // STENCIL: enforce joint limits for prismatic and revolute joints  
37  
38  
39         // clear controls back to zero for next timestep  
40         curRobot.joints[x].control = 0;  
41     }
```

Control is already applied to
all joint.angles and
robot.origin for you

Grad section will need to
enforce joint limits

kineval_servo_control.js

kineval_servo_control.js

```
19 kineval.setpointDanceSequence = function execute_setpoints() {
20
21     // if update not requested, exit routine
22     if (!kineval.params.update_pd_dance) return;
23
24     // STENCIL: implement FSM to cycle through dance pose setpoints
25 }
26
27 kineval.setpointClockMovement = function execute_clock() {
28
29     // if update not requested, exit routine
30     if (!kineval.params.update_pd_clock) return;
31
32     var curdate = new Date();
33     for (x in robot.joints) {
34         kineval.params.setpoint_target[x] = curdate.getSeconds()/60*2*Math.PI;
35     }
36 }
37
38
39 kineval.robotArmControllerSetpoint = function robot_pd_control () {
40
41     // if update not requested, exit routine
42     if ((!kineval.params.update_pd)&&(!kineval.params.persist_pd)) return;
43
44     kineval.params.update_pd = false; // if update requested, clear request and process setpoint control
45
46     // STENCIL: implement P servo controller over joints
47 }
```

Implement a Finite State Machine for setpoint dance routine

Implement P controller for joint control to setpoints

kineval_servo_control.js

kineval_servo_control.js

```
19 kineval.setpointDanceSequence = function execute_setpoints() {  
20  
21     // if update not requested, exit routine  
22     if (!kineval.params.update_pd_dance) return;  
23  
24     // STENCIL: implement FSM to cycle through dance pose setpoints  
25 }  
26  
27 kineval.setpointClockMovement = function execute_clock() {  
28
```

Implement a Finite State Machine for setpoint dance routine

Thought experiment:

1. Why are we only asking for a P controller?
2. What would control look like with a PID controller?
3. What about a PD controller?

```
38  
39 kineval.robotArmControllerSetpoint = function robot_pd_control () {  
40  
41     // if update not requested, exit routine  
42     if ((!kineval.params.update_pd)&&(!kineval.params.persist_pd)) return;  
43  
44     kineval.params.update_pd = false; // if update requested, clear request and process setpoint control  
45  
46     // STENCIL: implement P servo controller over joints  
47 }
```

Implement P controller for joint control to setpoints

home.html

Create a cool dance routine by defining a sequence of joint angle setpoints to be used by the FSM implementation

```
home.html
131  //////////////////////////////////////
132  /////      MAIN FUNCTION CALLS
133  //////////////////////////////////////
134
135  // start KinEval execution once the page and its resources are loaded
136  //window.onload = kineval.start;
137  document.body.onload = kineval.start;
138
139  // STUDENT: my_animate is where your robot's controls and movement are updated over time
140  function my_init() {
141
142      kineval.startingPlaceholderInit(); // a quick and dirty JavaScript tutorial
143      Initialize kineval.setpoints and
144  }      kineval.params.dance_sequence_index here
```

Poses for servo can be set and stored
interactively in KinEval using [0-9]
keys and Shift+[0-9]

`JSON.stringify(kineval.setpoints)`
will output the currently available servo
setpoints to the console as a string

Demo