Diff:

Differences between given skeleton and solution
In order to make the sample solution easier to understand, the differences between it and the given skeleton source code were highlighted with the help of the program `diff`.

Legend:

- Gray: unchanged text (only excerpts).
- Green: new lines
- Yellow: changed lines
- Red: deleted lines

Note: Files not listed have not been changed.

This document was created with the help of diff2html erstellt.

```
diff -u ../course10-3d-visualization/exercise/code/functions.py ../course10-3d-visualization/exercise/solution/functions.py
```

../course10-3d-visualization/exercise/code/functions.py		../course10-3d-visualization/exercise/solution/functions.py	
⋮		⋮	
21		21	
22	`# extract trolley position x and pendulum angle phi from the state vector`	22	`# extract trolley position x and pendulum angle phi from the state vector`
23	`# (the velocity components are not needed)`	23	`# (the velocity components are not needed)`
24	`x = y[XXX].`	24	`x = y[0].`
25	`phi = XXX.`	25	`phi = y[1].`
26		26	
27	`# calculate position and orientation of the trolley`	27	`# calculate position and orientation of the trolley`
28	`r_trolley = XXX.`	28	`r_trolley = np.array([x, 0, 0]).`
29	`R_trolley = XXX.`	29	`R_trolley = np.eye(3)  # no orientation change needed (use unit matrix).`
30		30	
31	`# Lage und Orientierung der Last.`	31	`# calculate position and orientation of the load.`
32	`r_load = np.array(XXX).`	32	`r_load = np.array([x+l*sin(phi), 0, -l*cos(phi)]).`
		33	
		34	`# default rotation matrix about y-axis`
		35	`R_load = np.array([[cos(phi), 0, -sin(phi)],`
		36	`[          0, 1, 0           ],`
		37	`[sin(phi), 0,  cos(phi)]])`
33		38	
34	`R_load = XXX`		
35		39	
36	`# task 9:`	40	`# task 9:`
37		41	
38	`# calculate position and orientation of the rope`	42	`# calculate position and orientation of the rope`
39	`# Zylinder muss zusätzlich um 90 Grad um x-Achse in die x-z-Ebene gedreht werden`	43	`# Zylinder muss zusätzlich um 90 Grad um x-Achse in die x-z-Ebene gedreht werden`
40	`# r_rope = None.`	44	`r_rope = np.array([x+l/2*sin(phi),0,-l/2*cos(phi)]).`
41	`# R_rope = None.`		
42		45	
43	`# return (r_trolley, R_trolley, r_load, R_load, r_rope, R_rope).`	46	`# the cylinder that represents the rope has to be rotated also about the x-axis.`
		47	`# this can be achieven by usinge the according default matrix:`
		48	`Rx = np.array([[1,            0,           0],`
		49	`[0,  cos(np.pi/2), sin(np.pi/2)],`
		50	`[0, -sin(np.pi/2), cos(np.pi/2)]])`
		51	
		52	`# the orientation matrix of the rope is now the orientation matrix of the`
		53	`# load (depending on phi) multiplied with that static rotation (Rx)`
		54	`R_rope = R_load@Rx`
44		55	
45	`return (r_trolley, R_trolley, r_load, R_load).`	56	`# return a tuple with 6 elements (brackets are optional).`
		57	`return (r_trolley, R_trolley, r_load, R_load, r_rope, R_rope)`
46		58	
47		59	
48	`def setPokeMatrix(actor, r, R):`	60	`def setPokeMatrix(actor, r, R):`
⋮		⋮	
68	`poke = vtk.vtkMatrix4x4()`	80	`poke = vtk.vtkMatrix4x4()`
69		81	
70	`# stack together the received arguments of the functions`	82	`# stack together the received arguments of the functions`
71	`P = np.column_stack((XXX, XXX)).`	83	`P = np.column_stack((R, r)).`
72	`P = np.row_stack( (XXX, np.array([0, 0, 0, 1])) ).`	84	`P = np.row_stack( (P, np.array([0, 0, 0, 1])) ).`
73		85	
74	`# P is now a numpy array with shape (4, 4); its elements are copied to the above vtk matrix`	86	`# P is now a numpy array with shape (4, 4); its elements are copied to the above vtk matrix`
75	`vtk.vtkMatrix4x4.DeepCopy(XXX, XXX).`	87	`vtk.vtkMatrix4x4.DeepCopy(poke, P.flatten()).`
76		88	
77	`# pass the poke matrix to the actor`	89	`# pass the poke matrix to the actor`
78	`actor.PokeMatrix(poke)`	90	`actor.PokeMatrix(poke)`

Nur in ../course10-3d-visualization/exercise/solution/: __pycache__.

```
diff -u ../course10-3d-visualization/exercise/code/visualization.py ../course10-3d-visualization/exercise/solution/visualization.py
```

../course10-3d-visualization/exercise/code/visualization.py	../course10-3d-visualization/exercise/solution/visualization.py
⋮	⋮
2 from vtk.util.colors import *	2 from vtk.util.colors import *
3	3
4	4
5 # insert imports	5 from functions import calcPositions, setPokeMatrix
6 from functions import XXX	6 from model import calcStep
7 from XXX import XXX	
8	7
9	8
10 # ----------	9 # ----------
11	10
12 # cuboid for the trolley with dimension x=0.3, etc.	11 # cuboid for the trolley with dimension x=0.3, etc.
13 # trolley_source = XXX	
14 # trolley_source.SetXLength(XXX)	
15 # ...	
16	12
17 trolley_mapper = XXX	13 # trolley_source = vtk.vtkCubeSource()
18 # ..	14 # trolley_source.SetXLength(0.3)
	15 # trolley_source.SetYLength(0.1)
	16 # trolley_source.SetZLength(0.1)
19	17
20 trolley_actor = XXX	
21 #	
22	18
	19 # taks 10:
	20 # load geometry from stl file
	21 trolley_source = vtk.vtkSTLReader()
	22 trolley_source.SetFileName('../data/trolley.stl')
23	23
	24 trolley_mapper = vtk.vtkPolyDataMapper()
	25 trolley_mapper.SetInputConnection(trolley_source.GetOutputPort())
	26
	27 trolley_actor = vtk.vtkLODActor()
	28 trolley_actor.SetMapper(trolley_mapper)
	29
	30
	31 # task 10:
	32 hook_source = vtk.vtkSTLReader()
	33 hook_source.SetFileName('../data/hook.stl')
	34
	35 hook_mapper = vtk.vtkPolyDataMapper()
	36 hook_mapper.SetInputConnection(hook_source.GetOutputPort())
	37
	38 hook_actor = vtk.vtkLODActor()
	39 hook_actor.SetMapper(hook_mapper)
24	40
25 # ----------	41 # ----------
26	42
27 # cuboid for the load	43 # cuboid for the load
28 load_source	44 load_source = vtk.vtkCubeSource()
	45 load_source.SetXLength(0.1)
	46 load_source.SetYLength(0.1)
	47 load_source.SetZLength(0.1)
29	48
30 # mapper, actor	49 load_mapper = vtk.vtkPolyDataMapper()

```
                                                                                load_mapper.SetInputConnection(load_source.GetOutputPort())
                                                                                
                                                                                load_actor = vtk.vtkLODActor()
                                                                                load_actor.SetMapper(load_mapper)
                                                                                
-# ------------------------------------------------------------------------------
                                                                                
                                                                                # ------------------------------------------------------------------------------
                                                                                
# task 9:                                                                       # task 9:
# cylinder for the rope                                                         # cylinder for the rope
rope_source = XXX                                                               rope_source = vtk.vtkCylinderSource()
XXX.SetRadius(XXX)                                                              rope_source.SetRadius(0.01)
XXX.SetHeight(XXX)                                                              rope_source.SetHeight(0.5)
                                                                                
                                                                                
                                                                                rope_mapper = vtk.vtkPolyDataMapper()
                                                                                rope_mapper.SetInputConnection(rope_source.GetOutputPort())
                                                                                
# mapper, actor                                                                 rope_actor = vtk.vtkLODActor()
                                                                                rope_actor.SetMapper(rope_mapper)
                                                                                
                                                                                
# ------------------------------------------------------------------------------ # ------------------------------------------------------------------------------
⋮                                                                               ⋮
iren.SetRenderWindow(renWin)                                                    iren.SetRenderWindow(renWin)
                                                                                
# add actors to renderer                                                        # add actors to renderer
ren.AddActor(XXX)                                                               ren.AddActor(trolley_actor)
# ...                                                                           ren.AddActor(hook_actor)
                                                                                ren.AddActor(load_actor)
                                                                                ren.AddActor(rope_actor)
                                                                                
# background color and window size                                              # background color and window size
ren.SetBackground(0.1, 0.2, 0.4)                                                ren.SetBackground(0.1, 0.2, 0.4)
⋮                                                                               ⋮
    """                                                                             """
                                                                                
    # Calculate a simulation step                                               # Calculate a simulation step
    XXX, XXX = calcStep()                                                       t, y = calcStep()
                                                                                
    # Calculate positions and orientations of the bodies                        # Calculate positions and orientations of the bodies
    XXX1, XXX2, XXX3, XXX4 = calcPositions(y)                                   r_trolley, R_trolley, r_last, R_last, r_seil, R_seil = calcPositions(y)
                                                                                
    # update bodies                                                             # update bodies
    setPokeMatrix(XXX, XXX, XXX)                                                setPokeMatrix(trolley_actor, r_trolley, R_trolley)
    # ...                                                                       setPokeMatrix(hook_actor, r_trolley, R_trolley)
                                                                                setPokeMatrix(load_actor, r_last, R_last)
                                                                                setPokeMatrix(rope_actor, r_seil, R_seil)
                                                                                
    # re-render the image                                                       # re-render the image
    renWin.Render()                                                             renWin.Render()
```