

Übung 04b: 3D-Visualisierungen mit Python

Ziel der heutigen Übung ist das Erstellen einer 3D-Animation der Laufkatze (verschiebbarer Körper mit Pendel). Dazu sind drei Python Dateien vorgegeben:

model.py

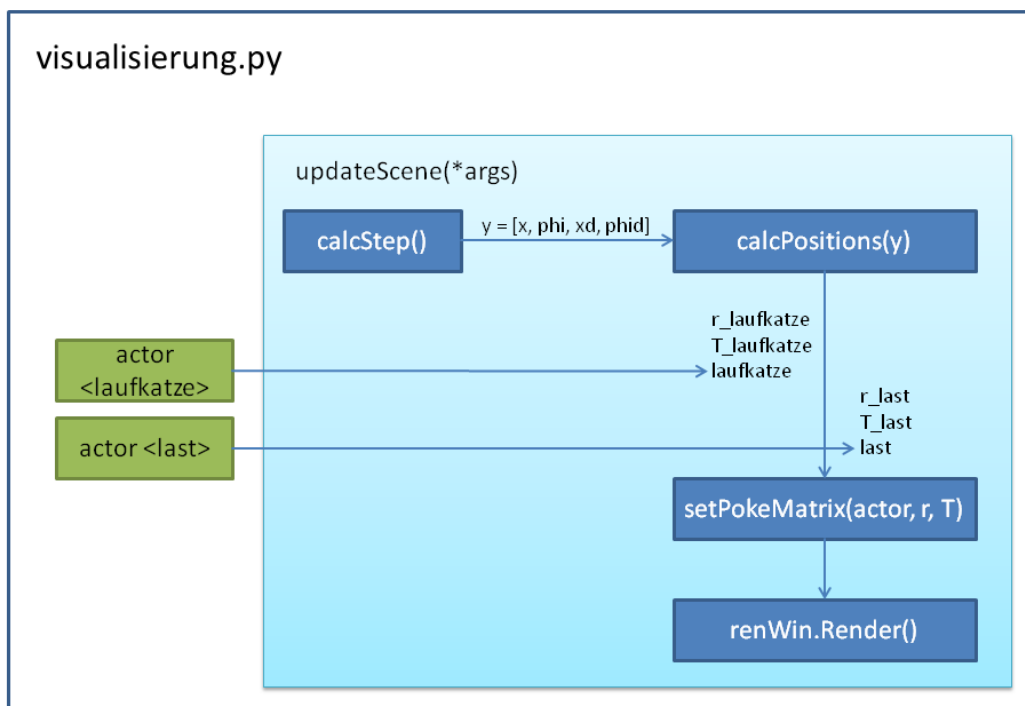
Enthält das Differentialgleichungssystem (Funktion `rhs`) und den Solver sowie eine Funktion zur Berechnung eines einzelnen Integrationsschrittes (`calcStep`).

functions.py

Enthält die Funktion `calcPositions` zur Berechnung der Position und Orientierung der Körper aus dem Simulationsergebnis sowie die Funktion `setPokeMatrix`, die die Lageinformation an die Geometrieobjekte übergibt.

visualisierung.py

Das Hauptskript. Hier werden die Geometrieprimitive angelegt und die Szene erstellt. Dazu ist die Funktion `updateScene` vorgegeben. Um eine Animation zu erhalten, muss sie wiederholt aufgerufen werden. Innerhalb von `updateScene` wird ein Simulationsschritt gerechnet, aus dem Ergebnis die Lage der Körper berechnet, diese wiederum an die Geometrieobjekte (actors) übergeben und schließlich das Bild neu gerendert, siehe Ablaufschema:



Aufgaben

1. Vervollständigen Sie die Funktion `calcPositions(y)` in der Datei `functions.py`. Berechnen Sie die Lage und Orientierung der Laufkatze und der Last aus dem Ergebnis y eines Simulationsschrittes. Da die Laufkatze nur translatorisch verfahren kann, entspricht ihre Orientierung der 3x3 Einheitsmatrix („eye“). Die Position der Last berechnet sich nach: $r = [x + l \sin(\varphi), 0, -l \cos(\varphi)]$. Zur Berechnung der Orientierungsmatrix R_y der Last siehe Folie 10.
Hinweis: Orientieren Sie sich bei der Variablenbezeichnung am Rückgabewert der Funktion (vorgegeben).
2. Vervollständigen Sie die Funktion `setPokeMatrix(actor, r, T)` in der gleichen Datei. Hier ist die Matrix `poke` elementweise mit den Werten aus `r` und `T` zu befüllen (siehe Folie 10).
3. Legen Sie in der Datei `visualisierung.py` einen Quader für die Laufkatze an ($x = 0.3, y = 0.1, z = 0.1$). Fügen Sie einen Mapper und einen Actor für den Quader hinzu
4. Wiederholen Sie Punkt 3 für die Last. Erzeugen Sie aber diesmal einen Würfel mit der Kantenlänge 0.05
5. Fügen sie beide actors zum Renderer hinzu (`ren.AddActor(...)`)
6. Importieren Sie die benötigten Funktionen zur Aktualisierung der Szene (s. Schema)
7. Vervollständigen Sie die Funktion `updateScene`, beachten Sie dabei die Rückgabewerte der einzelnen Funktionen und die Reihenfolge der Argumente!
8. Testen Sie die Animation. Der Befehl `iren.CreateRepeatingTimer(20)` gibt das Aktualisierungsintervall in ms an, also die Geschwindigkeit der Animation.

Zusatzaufgaben

1. Erzeugen Sie einen Zylinder der das Seil darstellt. Der Zylinder erhält die gleiche Orientierung wie die Last, muss jedoch noch entsprechend der Pendellänge verschoben werden (Position r extra berechnen)
2. Ersetzen Sie die Laufkatze durch die beiliegende stl-Datei. Nutzen Sie dazu den `vtkSTLReader` (s. Folie 13). Evtl. ist eine Skalierung von stl oder der Last erforderlich.