Aufgabe 7

Die Aufgabe wurde mittels des Matlab-Skripts „Aufgabe7/sieben\_Rob\_Simulink.mlx“ und der Simulik-Datei „Augabe7/sl\_robreg1\_CR\_MM\_dezPD.slx“ umgesetzt.

PD-Regler (PD = Proportional-Derivativ) sind einfach in der Implementierung und im Tuning. Selbst ohne ein komplexes dynamisches Modell kann durch die dezentrale Berechnung eine Bahn im Gelenkraum ermittelt werden. Außerdem sorgen PD-Regler für gut gedämpfte Bewegungen mit minimalen Oszillationen. Damit sinkt auch der Rechenaufwand, da ein Regler nur Positions- und Geschwindigkeitsdifferenzen, aber keine Kopplungseffekte benötigt.

Die Regelung erfolgt durch die Formel:

Der Nachteil besteht darin, dass nichtlineare Effekte wie die Corioliskräfte, Trägheiten und Gravitation nicht berücksichtigt werden. Außerdem können schnelle Bewegungen und hohe Lasten zu ungenauerer Bahnverfolgung führen. Somit muss der Regler bei neuen Lasten oder anderen Geschwindigkeiten neu abgestimmt werden. In Abbildung 1 ist das zugehörige Simulink-Model sichtbar.

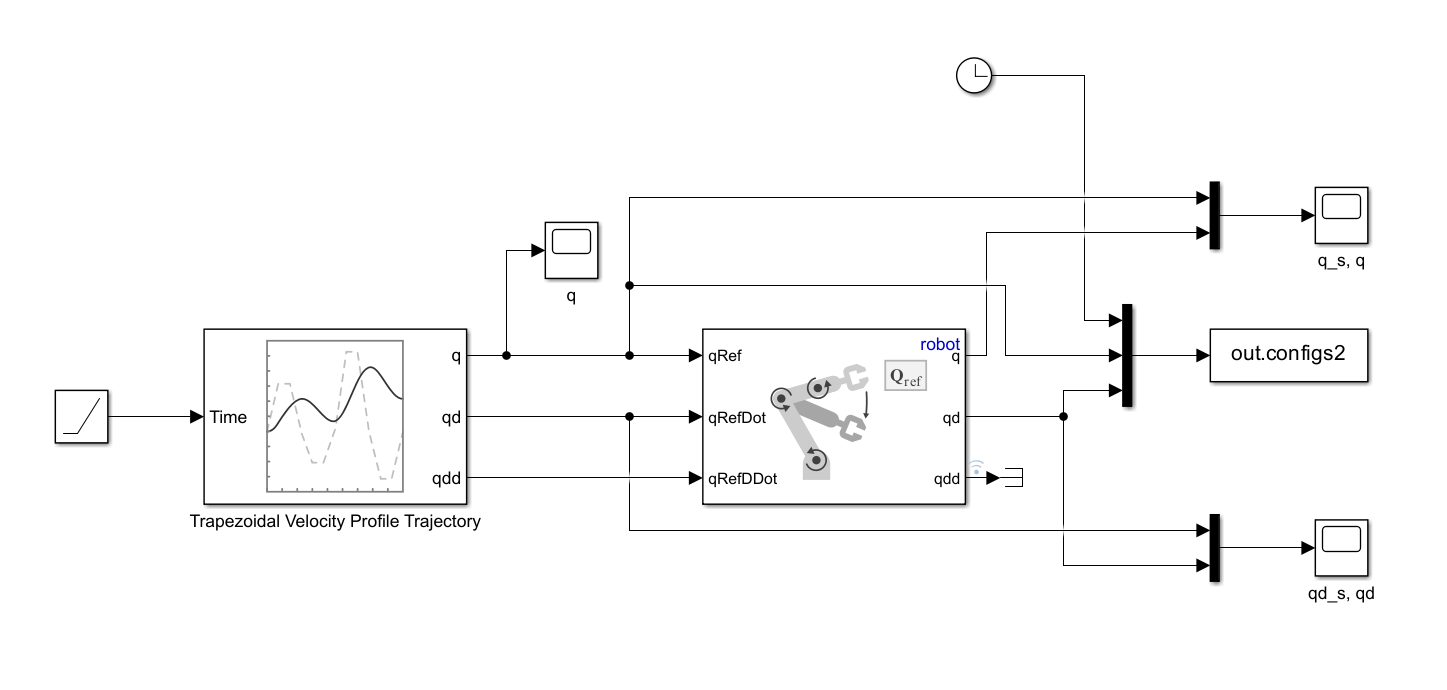


Abbildung 1: Simulink-Model

Das Skript beginnt mit dem Laden des Roboter-Modells und der Spezifikation der Bahnpunkte. Hier wurden die Bahnpunkte aus der vorherigen Aufgabe verwendet.

Nachfolgend wird eine Trajektorienplanung im Gelenkraum (Joint Space) durchgeführt. Die Gelenkwinkel werden dabei mittels Inverser Kinematik aus den Bahnpunkten berechnet und in C1 bis C7 gespeichert. Aus diesen Gelenkkonfigurationen wird mittels trapveltraj() eine Bahn im Gelenkraum mit jeweils linearer Beschleunigung, konstanter Geschwindigkeit und linearer Verzögerung erzeugt. Dieses trapezförmige Geschwindigkeitsprofil sorgt für eine glatte Bewegung ohne Sprünge in der Geschwindigkeit unter Einhaltung der kinematischen und dynamischen Beschränkungen des Roboters.

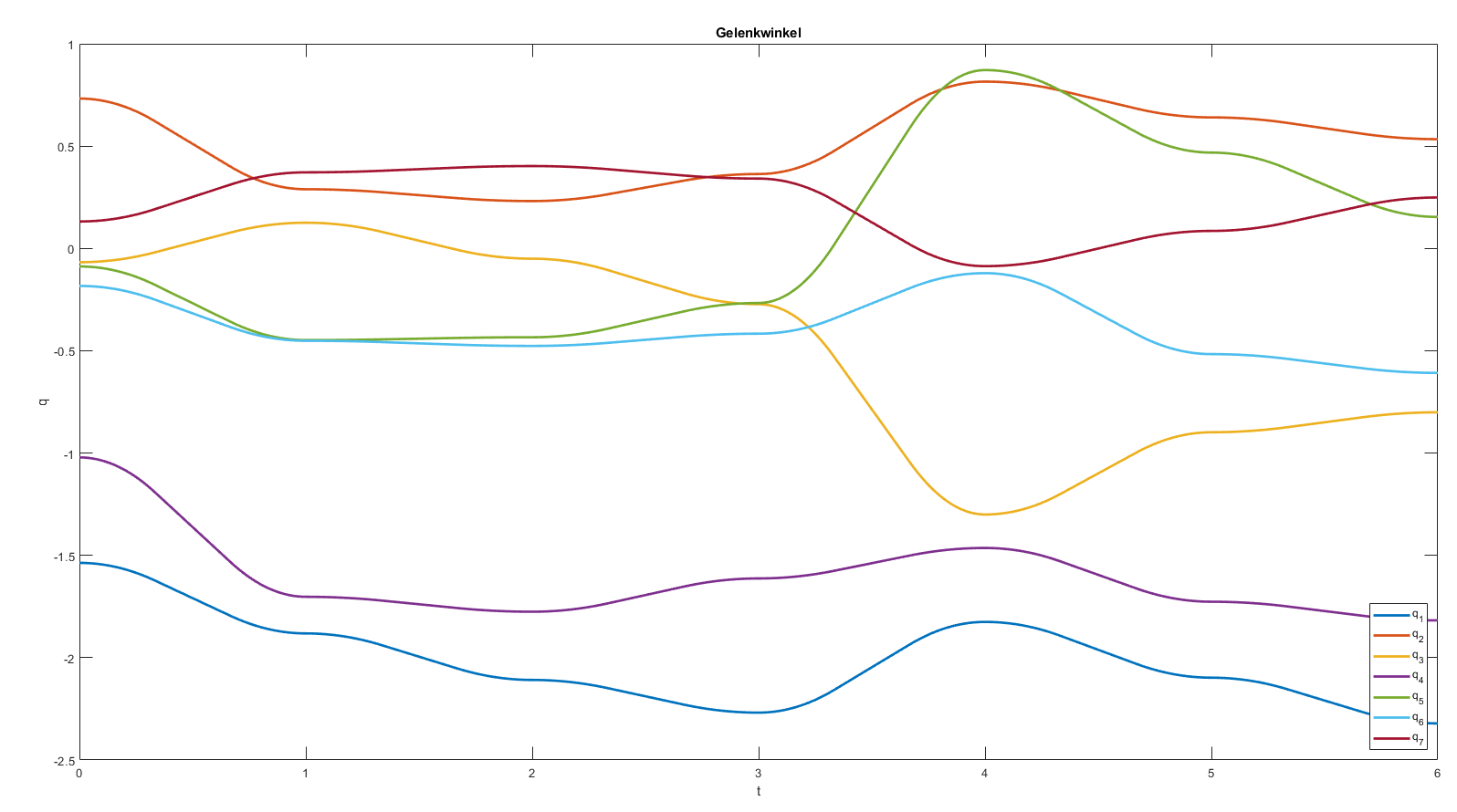


Abbildung 2: Gelenkwinkel

Dabei sind „q“ (Abbildung 1) die berechneten Gelenkwinkel, „qd“ (Abbildung 2) die Gelenkwinkelgeschwindigkeiten und „qdd“ die Gelenkwinkelbeschleunigungen. Die daraus folgende Bewegung wird schlussendlich in einem Live-Plot simuliert.

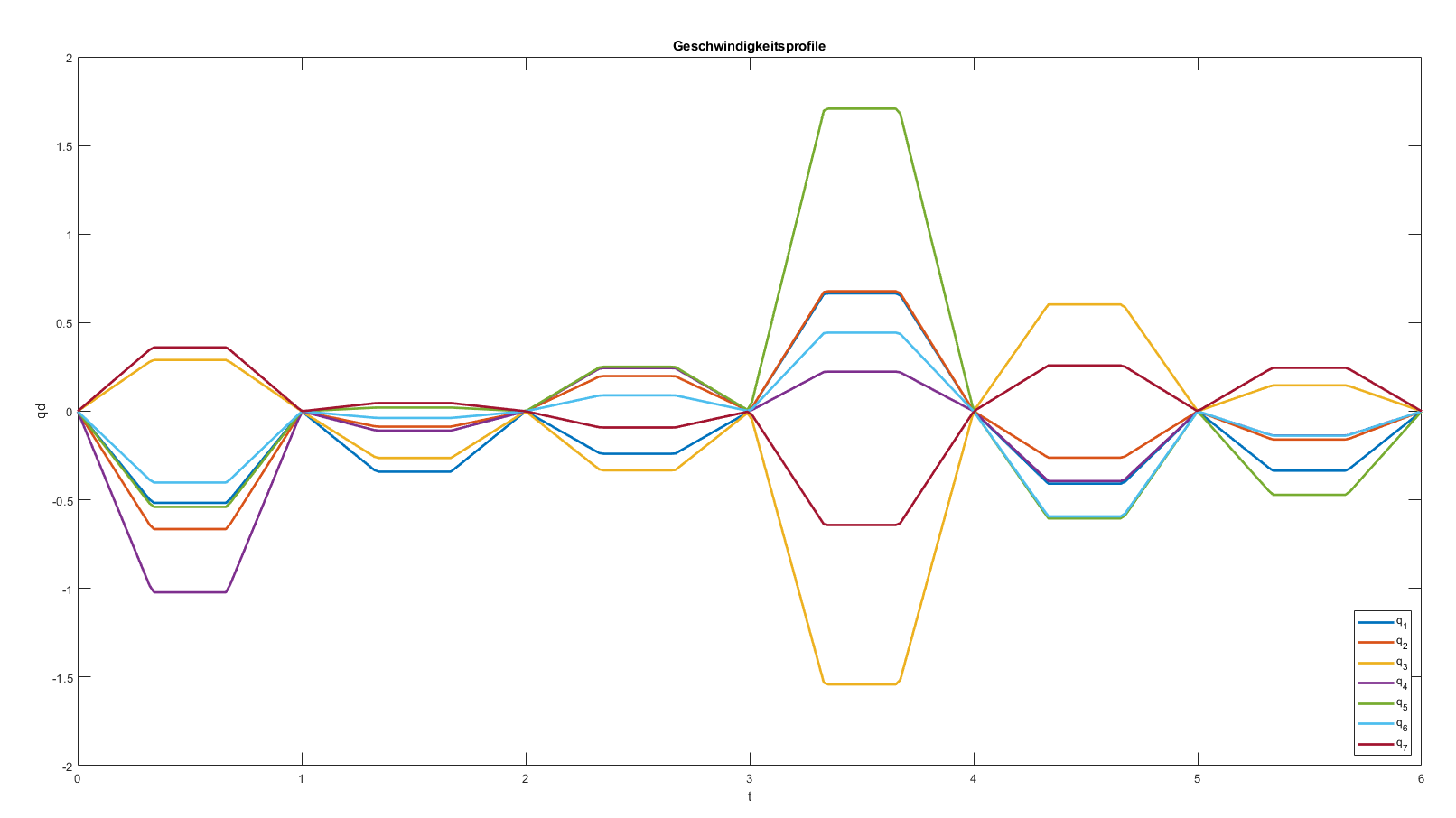


Abbildung 3: Geschwindigkeitsprofil

**Auswertung**

Dank geringer Geschwindigkeit und keiner zusätzlicher Last verfolgt der Roboterarm die vorgegebene Bahn sehr präzise. In Abbildung 3 ist beispielhaft ein Einschwingvorgang des Reglers erkennbar. Verglichen wird hier die vorgegebene Geschwindigkeit des Gelenk 3 mit der tatsächlichen. Es kommt kaum zu Abweichungen.

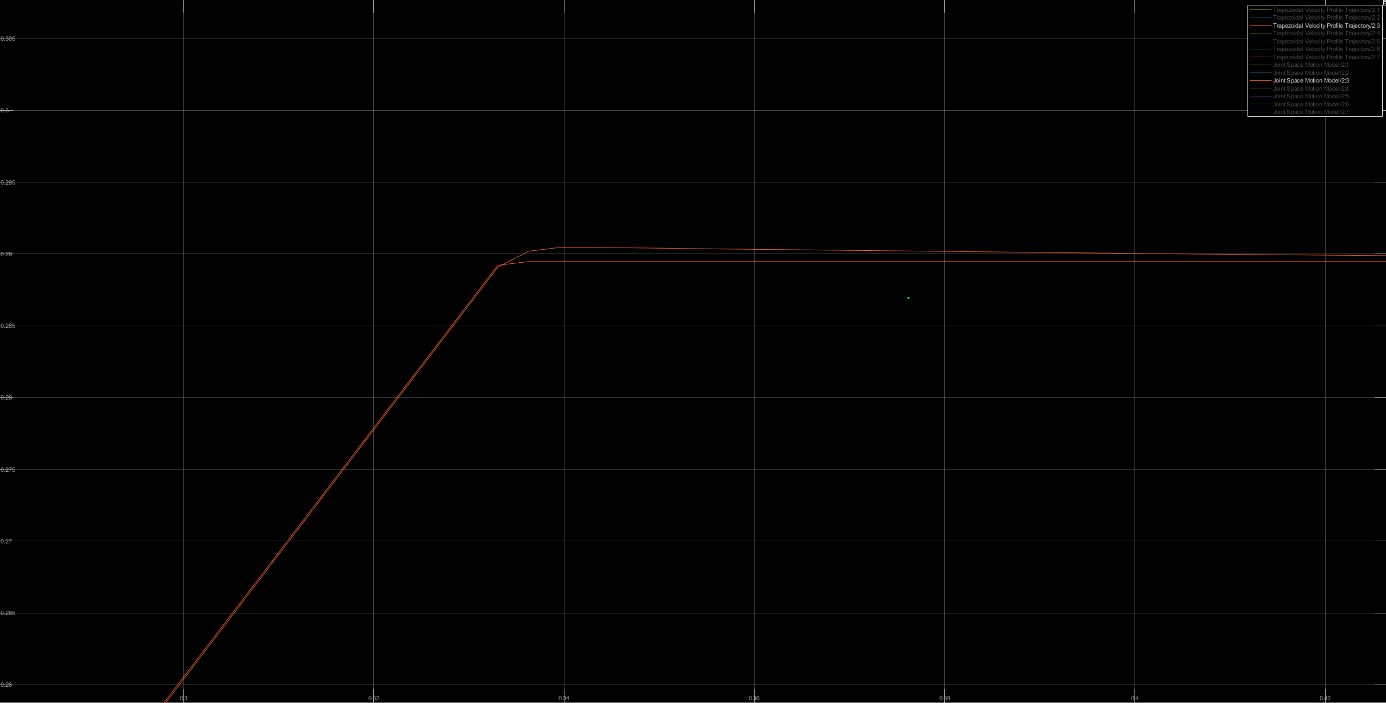


Abbildung 4: Einschwingverhalten

Vergleich mit dem Beispiel „Plan and Execute Task- and Joint-Space Trajectories Using KINOVA Gen3 Manipulator“ aus der RSTB-Dokumentation:

|  |  |  |
| --- | --- | --- |
| **Kriterium** | **Beispiel RSTB-Doku** | **unsere Umsetzung** |
| Trajektorienraum | Kombination Joint-Space und Task-Space | nur Joint-Space |
| IK-Verwendung | Zuerst im Task-Space, dann Umwandlung in Joint-Space | direkte Nutzung der IK für Wegpunkte |
| Trajektorienerzeugung | transformtraj -> Task Space  cubicpolytraj -> Joint-Space | trapveltraj -> glatte Bewegung im Joint-Space |
| Simulationsmethode | ODE-Solver -> berücksichtigt Dynamik | direkte Animation + Simulink -> keine Dynamik |
| Geschwindigkeitsprofil | automatische Berechnung der Zeit auf Basis der gewünschten Endeffektor-Geschwindigkeit | trapezförmiges Geschwindigkeitsprofil |