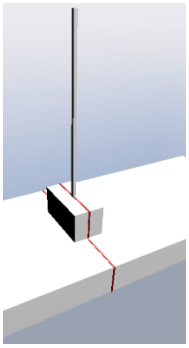
Dynamische systemen - Project 3 - Wiskunde 2.2

LQR

Nathan Isaac – 500899683
Dennis Philipoom – 500904361

Introductie

Voor het derde en laatste project in dynamische systemen hebben we het zogenaamde 'cart inverted pendulum' moeten stabiliseren en de kwadratische kosten van de verplaatsing van de cart zo laag mogelijk moesten krijgen. Het systeem is wederom gesimuleerd met Python, Het systeem kan worden beschreven als twee stokken die verticaal op elkaar zijn gemonteerd, maar nog steeds afzonderlijk van elkaar kunnen bewegen, op een karretje gezet die alleen in de x-richting kan bewegen. Het doel is om beide stokken zo te stabiliseren dat ze rechtop blijven staan terwijl het karretje zelf ook stilstaat op zijn nulpunt. Het streefdoel voor deze stabilisatie is vastgesteld op drie seconden als benchmark. Na het vaststellen van deze benchmark zal er gekeken worden naar hoe de kosten van de verplaatsing zo klein mogelijk gemaakt kunnen worden. De onderzoeksvraag luidt als volgt:



Figuur 11: Ons toegewezen systeem in een gestabiliseerde staat

"Hoever kunnen de toestand kosten van de verplaatsing worden geminimaliseerd terwijl het systeem binnen de grenzen van de benchmark blijft?"

Voor het stabiliseren van het systeem is gebruik gemaakt van een zogenaamde LQR-controller (Least Quadratic Regulator). Bij deze controller vul je een matrix in (ook Q genoemd) zodat hij symmetrisch is. Daarnaast vul je bij een scalar een waarde in die bepaalt hoe zwaar deze matrix meetelt (ook R genoemd). Er moet tijdens het invullen van de matrix rekening gehouden worden dat om het systeem stabiel te krijgen in iedere kolom de grootste waarde op de diagonaal staat.

Analyse

State-space matrix

De state-space representatie werd verkregen door tijdelijk extra sensoren aan het systeem toe te voegen, waardoor de toestand van het systeem gemeten kon worden. Vervolgens werden de constanten in de state-space matrix bepaald door variaties in de toestand. Het systeem dat we modelleren is niet lineair, maar deze methode negeert dit en gaat uit van een lineair systeem. Hierdoor is het model vooral geschikt voor kleine tijdstappen en kleine uitwijkingen. Bij grote uitwijkingen of tijdstappen zou de verkregen matrix niet langer een nauwkeurige beschrijving van het systeem geven. De gebruikte constanten om de state-space matrix te verkrijgen, staan in de bovenstaande tabel. Uit deze constanten zijn de volgende matrices voortgekomen:

Stapgrootte	Tijdstap (s)	Maximale tijd (s)
0,001	0,00001	0,001

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -0{,}18 & 0 & -0{,}18 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 26 & 0 & -25 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & -34 & 0 & 92 & 0 \end{pmatrix}$$

$$B = \begin{pmatrix} 0 \\ 0{,}089 \\ 0 \\ -0{,}22 \\ 0 \\ 0{,}30 \end{pmatrix}$$

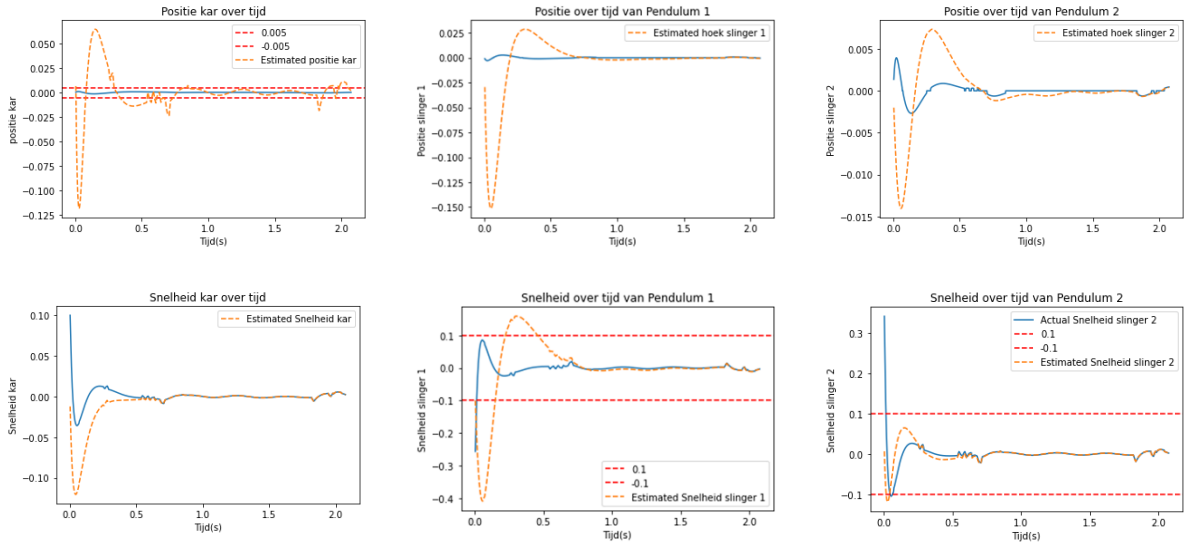
Deze matrices vormen de lineaire representatie van het systeem, hoewel het systeem zelf niet-lineair is. Het is belangrijk op te merken dat voor nauwkeurige resultaten, dit model het beste werkt bij kleine afwijkingen en tijdstappen. Bij grotere afwijkingen moeten mogelijk aanpassingen worden gemaakt om de niet-lineariteit van het systeem in overweging te nemen.

Systeemeigenschappen

Doormiddel van pythoncode zijn de stabiliteit, controleerbaarheid en stabiliseerbaarheid bepaalt. Het systeem is volgens onze code volledig controleerbaar wat betekent dat je van elke toestand naar een andere toestand kan komen zolang je genoeg energie erin steekt. Ons systeem is ook stabiliseerbaar wat betekend dat ons systeem minstens één stabiele toestand heeft. Ons systeem zal stabiel zijn als alle eigenwaardes van matrix A negatief zijn.

Sensorplaatsing

De opdracht was om zo min mogelijk sensoren te gebruiken om een schatting te maken van de posities en snelheden. Met één sensor bleek het al snel onmogelijk om een goede schatting van het systeem maken. Zoiezo was hij met een sensor alleen maar observeerbaar als de sensor op de positie van de kar, snelheid van pendulum 1, positie van pendulum 2 of snelheid van pendulum 2 stond. Daarnaast was te zien dat als je één sensor op de werkende posities zette de observaties compleet fout waren zelfs na een aantal seconden. Door een tweede sensor te plaatsen bleken de schattingen al na 1 seconde het daadwerkelijke systeem te volgen. Bij het plaatsen van meer sensoren hierna bleek het niet te helpen om een snellere schatting te krijgen. Hierom hebben we het gebruik van de schatting als onmogelijk verklaard. In de grafieken hieronder is te zien dat de schatting telkens ongeveer tussen de 0,5 en 1 seconden erover doet om gelijk te komen met de daadwerkelijke simulatie. In deze tijd schiet de kar al zodanig ver weg uit de lineaire waarden die we aannemen als waarheid dat deze niet meer gebruikt kan worden. Hierom gaan we verder met de daadwerkelijke observatie van de simulatie en niet die van de berekende schatting.



Oplossing

Wij hebben gekozen om de LQR-systeem te gebruiken i.p.v. pole-placement voor zowel het behalen van de benchmark als de verdiepende opdracht, omdat wij met een LQR veel meer controle hebben over ons systeem en zo makkelijker kunnen verdiepen in de opdracht. De LQR bestaat uit twee matrices die het systeem beïnvloeden; een Q-matrix en een R-matrix. De R-matrix is maar 1 bij 1 groot, de Q-matrix is zes bij zes groot. De zes parameters op de diagonaal van de Q-matrix geven respectievelijk aan hoeveel waarde het systeem geeft aan het stabiliseren van de positie van de kar, snelheid van de kar, hoek van de laagste slinger, hoeksnelheid van de laagste slinger hoek van de hoogste slinger en hoeksnelheid van de hoogste slinger. De scalar in de R-matrix geeft aan hoe veel waarde er wordt gegeven aan het maken van kwadratische kosten.

Wat ons opviel is dat dit systeem veel hogere kwadratische kosten had dan vorige systemen die wij hebben moeten stabiliseren dit blok. D.m.v. trial en error hadden wij uiteindelijk besloten dat we eerst de kar waarop de slingers staan te stabiliseren en daarna te focussen op de slingers. Nadat de kar binnen de benchmark stabiel was hebben wij de kwadratische kosten geoptimaliseerd door enkel de eerste twee parameters van de Q-matrix te variëren. Het werd al snel duidelijk dat als de kar stabiel is de slingers ook al gestabiliseerd zijn, zelfs als hier geen kwadratische kosten voor worden gevraagd. Nadat wij eerst de parameter in de R-matrix verhoogd hadden zodat het systeem zo min mogelijk kwadratische kosten verbruikte. Zijn wij de vier andere parameters gaan veranderen zodat de slingers ook zo min mogelijk kosten maakten. Hier hebben wij een stuk kleinere parameters gebruikt voor het optimaliseren als de eerste twee parameters op de diagonaal. Omdat meer niet nodig was voor het minimaliseren van de kosten. Ten slotte hebben wij de kosten nogmaals geminimaliseerd door de R-matrix te variëren.

Voor onze verdieping moesten wij de toestand kosten zo laag mogelijk krijgen door de Q en R matrices aan te passen. Omdat dit bij het behalen van de benchmark zo goed werkte, zijn wij ook bij de verdieping begonnen met de eerste twee parameters van de diagonaal te optimaliseren. We hebben hier wel extra stappen gezet om het optimalisatieproces een stuk grondiger te laten verlopen. We hebben nadat we de parameters geoptimaliseerd hadden zijn we teruggegaan naar de eerste parameter en deze opnieuw gestabiliseerd. Dit hebben wij de meerdere keren herhaalt totdat dit niet meer beter kon met een nauwkeurigheid van twee significante cijfers. We hebben de toestand kosten nog lager gekregen door ook bij de verdieping de kosten van de slingers achteraf aan te passen. Dit had wel effect op het verlagen van de toestand kosten, maar wel een zeer klein effect. We hebben nog geprobeerd de R-matrix aan te passen maar ook dit had geen effect meer op de toestand kosten.

Conclusie

Wij zijn tot de conclusie gekomen dat het gegeven systeem (cart inverted pendulum) het best te controleren is als de positie en de snelheid van de kar de prioriteit krijgen. De slingers op de kar hadden een stuk minder moeite met stabiliseren als de kar eenmaal gestabiliseerd was. Dit geldt voor het optimaliseren voor zowel de kwadratische als de toestand kosten van het systeem. De laagste toestand kosten die wij hebben kunnen produceren uit ons systeem is: 2,69. Dit is met de volgende parameters (op volgorde): $5.0 \cdot 10^6$, $5.7 \cdot 10^6$, 1.9, 0.0, 1.9, 0.0. Er is een extra bijlage waarin alle metingen staan gedocumenteerd. Wij hadden on systeem nog beter kunnen optimaliseren door op

meer significante cijfers nauwkeurig de parameters te variëren. Daarentegen hebben wij wel veel metingen gedaan waardoor wij wel zeker zijn dat wij wel de minimale toestand kosten hebben bereikt met onze significantie.

Advies

Wij raden aan om voorzichtig te zijn met het streven naar minimale toestandskosten in je verplaatsing. Hoewel het verminderen van de toestandskosten van groot belang kan zijn, kunnen de extra kwadratische kosten die hieruit voortkomen aanzienlijk zijn. Het lijkt niet nuttig om vijf keer zoveel kwadratische kosten te genereren puur om de toestandskosten te verminderen. We zien momenteel geen enkele geldige reden om deze benadering te volgen. Het is raadzaam om een evenwicht proberen te vinden tussen het minimaliseren van toestandskosten en het voorkomen van onnodige toename van kwadratische kosten.

Verantwoording

In dit verslag is geen gebruik gemaakt van generatieve AI.