Bacheloropdracht Technische Wiskunde — Universiteit Twente

Hidde Wieringa

29 januari 2016

Begeleider: A.A. Stoorvogel

Inhoudsopgave

Hoofdstuk 1

Inleiding

Slingers kom je overal tegen, van een schommel tot golfen, van een knie tot aan een hijskraan. In dit verslag zal worden uitgelegd hoe je een slinger kan besturen door er kracht op uit te oefenen. We maken het interessanter door een extra element toe te voegen: niet alleen moet de slinger doen wat wij wensen, het moet ook zo snel mogelijk gebeuren.

Om *zo snel mogelijk* te beschrijven is de wiskundige theorie van de optimale regelaars (*optimal control*) nodig. Dit is een tak van wiskunde die beschrijft hoe een besturing van een dynamisch systeem optimaal kan worden gemaakt; iets specifieker de voorwaarden waaraan de besturing moet voldoen om optimaal te zijn. Optimaal kan zo snel mogelijk betekenen, maar ook iets anders.

De basistheorie van optimale regelaars wordt kort toegelicht. Daarmee wordt het duidelijk hoe in de hoofdstukken erna de problemen worden beschreven en opgelost.

Het eerste probleem is dat van de slinger. Een optimale regelaar kan op verschillende manieren worden gevonden. In dit verslag wordt gekeken naar een oplossing die energieën en arbeid in het dynamische systeem gebruikt, een andere oplossing die tijd-optimale besturingen geeft voor alle beginvoorwaarden en ten slotte een oplossing die breder inzetbaar is, maar minder snel. We komen erachter dat een wiskundige oplossing niet altijd betekent dat het probleem is opgelost; dat een besturing is gevonden die bruikbaar is. Naast de oplossingen wordt voor elke methode een simulatie gepresenteerd, met een toelichting hoe de simulatie tot stand is gekomen. Uiteindelijk worden de drie methodes kort vergeleken.

Met de kennis die is opgedaan bij het probleem van de slinger kijken we naar een toepassing: de hijskraan. We laten een karretje met daaraan een gewicht over de *boom* van de kraan rijden. Door het karretje te versnellen of te remmen wordt ook de slinger van het gewicht beïnvloed. Een bestaand model wordt aangepast voor onze doeleinden en twee oplosmethodes worden toegelicht om een besturing te vinden van het karretje, zodanig dat een afstand op de kraan wordt afgelegd en de slinger precies stil hangt bij aankomst.

De twee methodes die hiervoor worden besproken zijn uitbreidingen van die van de slinger. Toch komen er een aantal nieuwe elementen kijken die bij de slinger geen rol spelen omdat het dynamische systeem groter is. Ook voor de kraan worden de oplosmethodes toegelicht met een simulatie, en kort vergeleken.

We zien dat oplosmethodes uit de theorie van optimale regelaars met verschillende voorwaarden en situaties verschillende resultaten opleveren. Een tijd-optimale oplossing zal betere prestaties geven, maar een variatie daarop die breder toepasbaar is kan ook goed werken.

Hoofdstuk 2

Theorie: Optimale regelaars

In dit hoofdstuk zal op een compacte wijze de theorie behandeld worden die hierna wordt gebruikt om de verschillende oplosstrategieën te kunnen toelichten. De theorie in dit hoofdstuk is samengesteld met behulp van [6].

2.1 Dynamische systemen

Een dynamisch systeem is een systeem dat zich gedraagt volgens bepaalde regels, vastgelegd door differentiaalvergelijkingen. Voor een toestand-vector $x(t)$ geldt dan

$$\dot{x} = f(x), \quad x(0) = x_0$$

waarbij $\dot{(\cdot)}$ een tijdsafgeleide voorstelt en $f(x)$ een functie van de variabelen x_i, $1 \leq i \leq n$. Het systeem heeft de beginconditie x_0, hoewel dit ook een conditie op een ander tijdstip dan 0 kan zijn.

Vaak wordt een systeem gelineariseerd rond een punt x_L, waarmee verbanden en berekeningen versimpeld worden. De gelineariseerde vorm van het systeem wordt gegeven door

$$\dot{x} = \left(\frac{\partial f}{\partial x}\Big|_{x=x_L} \right) x, \quad x(0) = x_0,$$

vaak geschreven als $\dot{x} = Ax$ met A een vierkante $n \times n$ matrix.

2.2 Regeling van een systeem

In de theorie voor optimale regelaars wordt er een *invoer* toegevoegd aan het systeem, meestal voorgesteld door $u(t)$, een m-dimensionale tijdsafhankelijke vector. De functie u is vrij te kiezen en wordt gebruikt om invloed te hebben op het systeem, vaak om het systeem naar een gewenste toestand te sturen. Met u toegevoegd ziet het systeem eruit als

$$\dot{x} = f(x,u) \quad \text{of} \quad \dot{x} = Ax + Bu$$

met B een $n \times m$ matrix.

2.3 Optimale regelaar

De theorie van optimale regelaars vertelt ons hoe we u moeten kiezen, zodanig dat het dynamische systeem zich op een optimale manier gaat gedragen tussen de tijdstippen 0 en T. Wat *optimaal* precies inhoudt wordt beschreven door een *kostenfunctie* J, van de vorm

$$J(x,u) = S(x(T)) + \int_0^T L(x,u)\,\mathrm{d}t.$$

Het doel van de optimale regelaar is deze kosten te minimaliseren door u goed te kiezen.

Hiervoor hebben we *costates* nodig. De costates, voorgesteld door $p(t)$ (een n-dimensionale vector), vervullen dezelfde rol als een *Lagrange multiplier* in de minimalisatie van een functie onderhevig aan voorwaarden. Ze helpen ons een expliciete vorm te geven aan het optimalisatie-probleem.

Met de costates kunnen we een *Hamiltoniaan* definiëren. Dit is de functie H gedefinieerd door

$$H(x,p,u) = p^T f(x,u) + L(x,u). \tag{2.1}$$

De Hamiltoniaan geeft ons voorwaarden voor een optimale u, gesteld door de volgende stelling:

Stelling 2.3.1. ***(Pontryagin's Minimum Principle)*** Stel dat $u_*(t) : [0,T] \to \mathbb{R}^m$ een oplossing is van het optimale regelaar probleem, en x_* de resulterende toestand van het systeem. Dan bestaat er een functie $p_*(t) : [0,T] \to \mathbb{R}^n$ zodat

$$\begin{aligned} \dot{x}_*(t) &= \tfrac{\partial H(x_*(t),p_*(t),u_*(t))}{\partial p} & x_*(0) &= x_0 \\ \dot{p}_*(t) &= -\tfrac{\partial H(x_*(t),p_*(t),u_*(t))}{\partial x} & p_*(T) &= \tfrac{\partial S(x_*(T))}{\partial x} \end{aligned}$$

en bij de oplossing $x_*(t), p_*(t)$ de invoer $u_*(t)$ voor elk moment van de tijd de Hamiltoniaan minimaliseert:

$$H(x_*(t),p_*(t),u_*(t)) = \min_{v \in \mathbb{R}^m} H(x_*(t),p_*(t),v(t))$$

voor alle $t \in [0,T]$. □

De variabele T is een gekozen vaste eindtijd van de optimalisatie.

Merk hierbij op dat de stelling aanneemt dat er een oplossing u_* kan worden gevonden. Er wordt geen garantie gegeven dat er een u_* bestaat of dat deze uniek is als hij bestaat. Om te bewijzen dat er een oplossing bestaat voor een optimalisatieprobleem is theorie van Toegepaste Functionele Analyse nodig die buiten de strekking van dit verslag is. Hier zal worden aangenomen dat er een oplossing u_* bestaat. Het bepalen of die oplossing uniek is en hoe hij eruitziet zal uitgebreid worden behandeld.

2.4 Vrije eindtijd

In stelling 2.3.1 wordt een vaste eindtijd T aangenomen. Een vrije eindtijd T_* waarover wordt geoptimaliseerd is ook mogelijk. In de volgende stelling wordt een variatie van stelling 2.3.1 gegeven.

Stelling 2.4.1. Stel dat $u_*(t) : [0, T_*] \to \mathbb{R}^m$ een oplossing is van het optimale regelaar probleem en T_* de optimale eindtijd voor de kostenfunctie J met

$$\dot{x}_* = f(x_*, u_*), \quad x_*(0) = x_0, \quad x_*(T_*) = x_T,$$

en x_* de resulterende toestand van het systeem. Laat

$$H(x, p, u, \lambda) = p^T f(x, u) + \lambda L(x, u).$$

Dan bestaat er een functie $p_*(t) : [0, T_*] \to \mathbb{R}^n$ en een $\lambda \in \{0, 1\}$ zodat

$$\begin{aligned} \dot{x}_*(t) &= f(x_*(t), u_*(t)) & x_*(0) &= x_0 \\ & & x_*(T_*) &= x_T \\ \dot{p}_*(t) &= -\frac{\partial H(x_*(t), p_*(t), u_*(t), \lambda)}{\partial x} & & \end{aligned}$$

en bij de oplossing $x_*(t), p_*(t)$ de invoer $u_*(t)$ voor elk moment van de tijd de Hamiltoniaan minimaliseert:

$$H(x_*(t), p_*(t), u_*(t), \lambda) = \min_{v \in \mathbb{R}^m} H(x_*(t), p_*(t), v(t), \lambda)$$

voor alle $t \in [0, T]$. Ook geldt op de optimale eindtijd

$$H(x_*(T_*), p_*(T_*), u_*(T_*), \lambda) = 0.$$

□

Ook al is er in deze stelling een extra parameter λ toegevoegd aan de Hamiltoniaan kunnen wij voor de strekking van dit verslag veronderstellen dat $\lambda = 1$. Dit geeft ons weer de gebruikelijke Hamiltoniaan uit (2.1).

2.5 Tijd-optimale oplossing

Een tijd-optimalisatie is een optimalisatie waarbij de kostenfunctie

$$J(x, u) = \int_0^{T_*} 1 \mathrm{d}t = T_*$$

wordt gebruikt. De eindtijd T_* wordt hier geminimaliseerd omdat de kosten zo klein mogelijk zijn wanneer T_* zo klein mogelijk is. Het lemma dat hierna volgt beschrijft een eigenschap van een tijd-optimale invoer u die aan een voorwaarde $|u(t)| \leq 1$ voor alle t voldoet.

Lemma 2.5.1. Laat $u_*(t) : [0, T_*] \to \mathbb{R}^m$ een oplossing zijn van het optimale regelaar probleem $\dot{x}_* = Ax_* + Bu_*$ met A een $n \times n$ matrix en B een $n \times m$ matrix, $x_*(0) = x_0$ en $x_*(T_*) = x_T$, met eindtijd T_* en kostenfunctie

$$J(x, u) = \int_0^{T_*} 1 \mathrm{d}t = T_*,$$

met de voorwaarde dat $|u_{*k}(t)| \leq 1$ voor alle $t \in [0, T_*]$ en voor alle $1 \leq k \leq m$.

Dan geldt dat $u(t) \in \{-1, 1\}$ voor alle $t \in [0, T_*]$. □

Een dergelijke optimale regelaar wordt ook wel een *bang-bang* regelaar genoemd. Er wordt van invoer gewisseld op vaste tijdstippen. Het kan voorkomen dat het aantal tijdstippen waarop gewisseld wordt oneindig groot is. Denk aan een situatie als $p(t) = \sin(1/(t - T_*))$, met $u(t) = \mathrm{sgn}(p(t))$. Door de costates te analyseren kan vaak informatie worden afgeleid over het aantal switchpunten. In dit verslag zal bij het bepalen van een tijd-optimale oplossing worden laten zien dat er niet oneindig veel switchtijden kunnen bestaan.

Hoofdstuk 3

Eerste probleem: de slinger

3.1 Inleiding

Hieronder wordt een model gepresenteerd voor de tijd-optimale besturing van een slinger. Dit model wordt gebruikt om drie oplosstrategieën voor een optimale regelaar met verschillende eigenschappen uit te leggen.

De eerste oplosmethode maakt gebruik van energieën in het systeem, en beredeneert op basis daarvan wanneer welke invoer aan het systeem moet worden gegeven. De tweede methode is een bang-bang regelaar die uit de theorie voor optimale regelaars volgt, en voor elke mogelijke begintoestand een aantal eigenschappen en een optimale invoer afleidt. Ten slotte wordt een oplosmethode gepresenteerd die gebruik maakt van een niet tijd-optimale strategie, maar wel aantrekkelijke eigenschappen heeft en algemener toepasbaar is op andere problemen.

Het doel is keer op keer hetzelfde: zo snel mogelijk de slinger stil laten hangen.

3.2 Model

In de volgende secties zal een lineair model voor de slinger worden opgebouwd uit vergelijkingen. Daarna zal een kostenfunctie worden gegeven waarmee we een optimale regelaar kunnen bepalen en beredeneren.

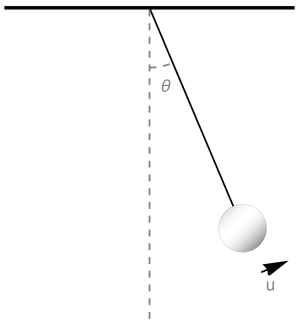
3.2.1 Basisvergelijkingen

Een slinger is een gewicht dat aan hangend aan een kabel aan een vast punt bevestigd is. De slinger kan in een plat vlak heen en weer slingeren onder invloed van een kracht op het gewicht, parallel aan de richting van de snelheid van de slinger.

De slinger wordt gemodelleerd op basis van de vergelijking

$$\ell\ddot{\theta} = -g\sin(\theta) + u$$

die kan worden afgeleid door een krachtenbalans van alle krachten werkend op het gewicht te maken. Hier is θ de hoek die de slinger maakt met de verticaal door het bevestigingspunt, g de gravitatieconstante, ℓ de lengte van de kabel en ten slotte u de (relatieve) invoer kracht, parallel aan de richting van de snelheid van de slinger. De massa m van de slinger komt hier niet in voor. In figuur 4.16 staat een visuele representatie van het model van de slinger.



FIGUUR 3.1: Een visuele representatie van het model met daarin de hoek θ en de kracht u.

In het model is u gedefinieerd als

$$u(t) = u_{\text{real}}(t) \big/ u_{\max},$$

met $u_{\text{real}}(t) \in [-u_{\max}, u_{\max}]$ de daadwerkelijke uitgevoerde kracht. Hierdoor geldt automatisch de voorwaarde dat $|u| \leq 1$ omdat er nooit meer kracht kan worden geleverd dan $u_{\max}$.

Merk op dat alle resultaten die hierna worden afgeleid met de verschillende oplosstrategieën ook toepasbaar is op een systeem met de invoerkracht $u_{\text{real}}(t) \in [u_{\min}, u_{\max}]$, zolang geldt dat $u_{\min} < 0$, wat de regelbaarheid van het systeem garandeert. In hoofdstuk 4, waar het probleem en de toepassing van de hijskraan worden behandeld, moet wel gelden dat $u_{\min} = -u_{\max}$ omdat de symmetrie een belangrijke eigenschap is van één van de oplosmethodes.

3.2.2 Toestanden van het systeem

Laat de toestanden van het systeem van de slinger $x = (x_1, x_2)^T = \left(\theta, \dot{\theta}\right)^T$ zijn. De vergelijkingen worden dan, omgeschreven in de toestand-vorm,

$$\begin{cases} \theta = \dot{x_1} = x_2 \\ \dot{\theta} = \dot{x_2} = -\frac{g}{\ell}\sin(x_1) + \frac{1}{\ell}u. \end{cases}$$

De gelineariseerde vergelijkingen van de slinger rond het punt $(0,0)$ zijn

$$\begin{pmatrix}\dot{x}_1\\ \dot{x}_2\end{pmatrix} = \begin{pmatrix}0 & 1\\ -\frac{g}{\ell} & 0\end{pmatrix}\begin{pmatrix}x_1\\ x_2\end{pmatrix} + \begin{pmatrix}0\\ \frac{1}{\ell}\end{pmatrix} u$$

of simpelweg $\dot{x} = Ax + Bu$. Merk op dat $(0,0)$ een evenwichtspunt is van dit systeem voor $u = 0$. In het bijzonder zijn $\left(\sin^{-1}\left(u/g\right),0\right)$ een evenwichtspunt . Voor het gelineariseerde systeem is dit $(u/g,0)$.

Een linearisatie van het systeem levert geen problemen op. De hoek θ waaronder de slinger beweegt blijft klein, en ook de hoeksnelheid $\dot{\theta}$ blijft klein. In de voorbeelden en simulaties zal gebruik gemaakt worden van een uitwijking van maximaal $30^\circ = \pi/6 \approx 0.52$. In realiteit blijven de hoeken kleiner dan dit. Bij een kleine θ is het verschil tussen $\sin(\theta)$ en θ erg klein (figuur 3.2(a)), en ook de fout bij de linearisering van $\cos(\theta) \approx 1$ wordt niet te groot (figuur 3.2(b)). Hierdoor blijft een simulatie accuraat.

De slinger willen we zo snel mogelijk stil laten hangen. Dit komt neer op het systeem naar de toestand $(0,0)$ sturen, in een zo kort mogelijke tijd.

3.3 Oplossing 1: Energieën

We gaan als eerste oplossing voor het probleem van de slinger een tijd-optimale regelaar vinden met behulp van energieën en verrichte arbeid in het systeem. Het uitoefenen van de kracht u kost een machine, motor of persoon energie en wordt gerekend tot arbeid. De uitgeoefende arbeid verandert de hoeveelheid energie in het systeem. Deze natuurkundige eigenschappen van het systeem geven ons de nodige informatie om voor een bereik aan beginvoorwaarden een optimale regelaar te kunnen vinden.

We gaan uit van een systeem waarbij de totale energie samen met de uitgeoefende arbeid altijd gelijk blijft: er gaat geen energie verloren aan processen die onbekend zijn, zoals uitstoting van warmtestraling of licht.

Voor de vergelijkingen en termen uit de natuurkunde in deze sectie is [7] gebruikt.

3.3.1 Optimale invoer

We zoeken naar een tijd-optimale oplossing voor ons probleem. We nemen dus de kostenfunctie

$$J = \int_0^{T_*} 1 \mathrm{d}t = T_*.$$

Deze integraal wordt geminimaliseerd. Het resultaat is dat het proces met bepaalde beginvoorwaarden in zo'n kort mogelijke tijd naar $(0,0)$ wordt gestuurd.

De Hamiltoniaan wordt dan

$$H(x,p,u) = p^T f + L = x_2 p_1 - \frac{g}{\ell} x_1 p_2 + \frac{u}{\ell} p_2 + 1,$$

waarmee we met stelling 2.4.1 vinden dat

$$u(t) = \underset{v}{\operatorname{argmin}}\, H(x,p,v) = -\operatorname{sgn}(p_2).$$

Dit is een *bang-bang* regelaar volgens lemma 2.5.1, met

$$u(t) \in \{-1, 1\}$$

voor $t \in [0, T_*]$. De invoer is altijd maximaal de ene kant op $(+1)$ of de andere kant op (-1). Dat levert ons een strategie om te beredeneren hoe een systeem in minimale tijd naar $\left(\theta,\dot{\theta}\right) = (0,0)$ kan komen, door te kijken naar de tijden en bijbehorende toestanden waarop er *switches*, dus veranderingen van u, plaatsvinden.

We noemen een toestand $\left(\theta,\dot{\theta}\right)$ waar op een switch plaatsvindt een *switch punt*, en de tijd waarop de switch plaatsvindt de *switchtijd*.

Het is logisch om af te vragen waarom het probleem hiermee nog niet is opgelost. Immers is er een vergelijking voor $u(t)$ gevonden. Het probleem komt door de costates $p(t)$, die onbekend zijn behalve de differentiaalvergelijkingen waaraan ze voldoen. Er zijn alleen geen begin- of eindvoorwaarden bekend: die zijn allemaal gebruikt voor de toestand $x(t)$. Dit geldt voor alle *bang-bang* regelaars die in dit verslag voorkomen.

We zullen dus op basis van de switch punten beredeneren hoe $u(t)$ zich optimaal gedraagt op elk tijdstip.

3.3.2 Energieën en arbeid

De energieën in het systeem zijn op te delen in twee categorieën: hoogte-energie en snelheidsenergie. We kijken daarna naar de verrichte arbeid in het systeem.

Hoogte-energie

De hoogte energie (ook wel potentiële energie) wordt gedefinieerd als

$$E_h = mg\ell(1-\cos(\theta)) = 2mg\ell\sin^2\left(\frac{\theta}{2}\right).$$

In de natuurkunde wordt dit ook wel U genoemd, maar deze letter werkt verwarrend in de context van de optimale regelaars. De expressie kan worden afgeleid uit de normale expressie:

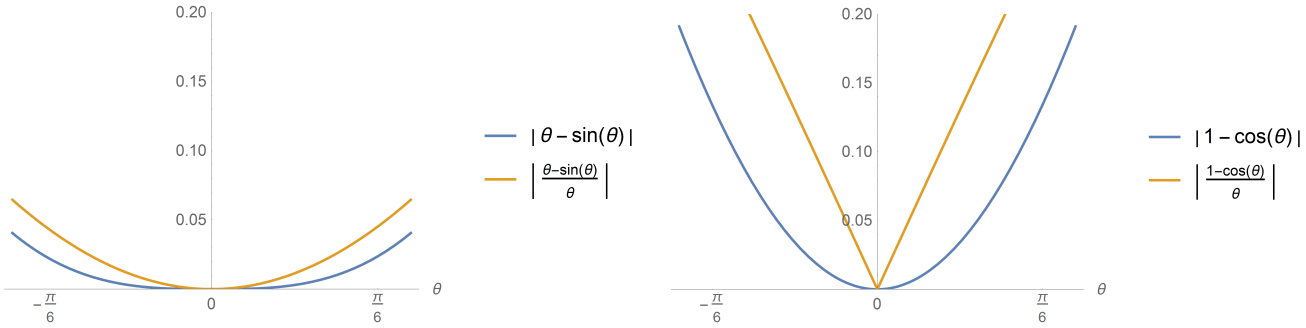
$$E_h = mgh$$

omdat h, de hoogte van de slinger, gelijk is aan

$$\ell(1-\cos(\theta))$$

krijgt men vanzelf bovenstaande expressie.

De tweede versie (met $\sin^2$) kan worden afgeleid met een trigonometrie regel. Afhankelijk of er gelineariseerd wordt en afhankelijk van het aantal termen, kan het nuttig zijn één van de twee vormen te gebruiken. In de hierna volgende tekst zullen we de eerste vorm gebruiken.



(a) Vergelijking van θ en $\sin(\theta)$.

(b) Vergelijking van θ en $\cos(\theta)$.

FIGUUR 3.2: Het absolute en relatieve verschil tussen verschillende functies.

Snelheidsenergie

De snelheidsenergie (ook wel kinetische energie) in een systeem wordt aangegeven met E_v en wordt gedefinieerd als

$$E_v = \frac{1}{2} m \left(\ell \dot{\theta}\right)^2.$$

Deze expressie kan gemakkelijk worden afgeleid uit de normale kinetische energie vergelijking

$$T = \frac{1}{2} m v^2$$

met v omgeschreven van radiale snelheid met

$$v = \ell \dot{\theta}.$$

Ook hier wordt de natuurkundige term T voor kinetische energie niet gebruikt wegens verwarring van betekenis.

Arbeid

De vergelijking die beschrijft wat de verrichte arbeid, aangegeven met W, op een bepaald moment is, wordt gegeven door

$$\mathrm{d}W = F\mathrm{d}s = Fv\mathrm{d}t,$$

met $W(0) = 0$. Hier is F de kracht die wordt uitgeoefend en v de snelheid van de slinger. In dit probleem geldt $F = u$ en $v = l\dot{\theta}$.

De totale verrichte arbeid tussen tijdstip 0 en t is dus

$$W(t) = \int_0^t u(t') \cdot \ell \dot{\theta}(t') \mathrm{d}t'.$$

Voor een constante $u = \hat{u}$ komt dit neer op

$$W(t) = \int_0^t \hat{u} \cdot \ell \dot{\theta}(t') \mathrm{d}t' = \hat{u}\ell(\theta(t) - \theta(0)),$$

wat het uitwerken van integralen gemakkelijker maakt. Merk op dat u een bang-bang regelaar is, dus dat op een aantal discontinuïteiten na $u(t) = \hat{u}$ geldt, voor verschillende constanten $\hat{u}$.

3.3.3 Analyse

We gebruiken de volgende beginvoorwaarden voor het systeem:

$$x_1(0) = \theta_0 > 0$$
$$x_2(0) = \dot{\theta}_0 \leq 0.$$

Voor $\theta_0 < 0$ en $\dot{\theta}_0 \geq 0$ kunnen dezelfde soort argumenten als hierna worden aangedragen worden gebruikt om een optimale regelaar te vinden. Daarbij moet u precies tegenovergesteld zijn vergeleken met de oplossing die hier gepresenteerd wordt.

De beginenergie in het systeem is

$$E_h(0) = mg\ell\left(1 - \cos\left(\theta_0\right)\right) = E_{h_0}$$
$$E_v(0) = \frac{1}{2} m \left(\ell \dot{\theta}_0\right) = E_{v_0}$$

Hiermee is de totale energie E in het systeem

$$E(0) = E_h(0) + E_v(0) = E_{h_0} + E_{v_0}$$

voor $t = 0$.

Wanneer T tijd is verstreken en het systeem in rust is, geldt dat

$$E(T) = E_h(T) + E_v(T) = 0 = \left(E_{h_0} + E_{v_0}\right) + W(T)$$

omdat $E(t)$ gelijk is aan $E(0) + W(t)$ voor alle t en $\theta = 0$ en $\dot{\theta} = 0$ op $t = T$.

Neem nu aan dat er één switch punt op $t = t_s$, met hoek $\theta_s = \theta(t_s)$, zal plaatsvinden. We willen de energie naar 0 dus zal gelden dat

$$u(t) = \begin{cases} -1 & \theta_s \leq \theta < \theta_0 \\ 1 & \theta_T < \theta < \theta_s \\ 0 & \text{anders} \end{cases},$$

met $\theta_T = \theta(T) = 0$.

We gebruiken het verband tussen $E(0)$, $E(T)$ en $W(T)$, namelijk

$$E(T) - E(0) = W(T)$$

om een vergelijking met daarin θ_s te krijgen:

$$\begin{aligned} 0 - \left(E_{h_0} + E_{v_0}\right) &= \int_0^{t_s} \ell(-1) \cdot \dot{\theta}(t')\mathrm{d}t + \int_{t_s}^{T} \ell(1) \cdot \dot{\theta}(t')\mathrm{d}t \\ &= -\ell\left(\theta_s - \theta_0\right) + \ell\left(\theta_T - \theta_s\right). \end{aligned}$$

Door gebruik te maken van de uitwerking van de integraal voor een vaste $\hat{u}$ krijgen we een vergelijking voor θ_s. Deze kan opgelost worden om de switchhoek θ_s te vinden:

$$\theta_s = \frac{E_{v_0} + E_{h_0} + \ell(\theta_0 + \theta_T)}{2\ell} = \frac{1}{2}\left((\theta_0 + \theta_T) + \frac{E_{v_0} + E_{h_0}}{\ell}\right).$$

3.3.4 Resultaten

Met de parameters $m = 1.0$, $g = 9.81$, $\ell = 60$ en beginvoorwaarden $\theta_0 = 0.08 \approx 4.5°$ en $\dot{\theta}_0 = 0$ is een simulatie uitgevoerd. Een afbeelding van de resultaten daarvan staat in figuur 3.3.

In de eerste afbeelding is de hoek en de hoeksnelheid van de slinger te zien. Verder is de constante hoek θ_s te zien en de invoerkracht $u(t)$. In de tweede afbeelding is de snelheids-, hoogte- en de totale energie in het systeem en de verrichte arbeid te zien.

3.3.5 Toepasbaarheid van oplosmethode

De bovengenoemde methode werkt alleen voor één switchpunt en een hoek die niet van teken mag veranderen: de slinger moet in één keer stil hangen zonder eerst $\theta = 0$ en $\dot{\theta} = 0$ te passeren. Dit beperkt de mogelijke beginwaarden.

Een andere voorwaarde is dat de kracht u genoeg arbeid moet kunnen leveren om de slinger stil te laten hangen. Dat geldt wanneer

$$\theta_s > \theta_0.$$

Deze vergelijking kan worden omgeschreven tot een voorwaarde aan de beginvoorwaarden om

$$\left(\theta_0 - \frac{1}{mg}\right)^2 + \frac{\dot{\theta}_0^2}{g/\ell} > \left(\frac{1}{mg}\right)^2$$

te krijgen. De tegenovergestelde beginvoorwaarden hebben een oplossing. Ze voldoen dus aan

$$\left(\theta_0 - \frac{1}{mg}\right)^2 + \frac{\dot{\theta}_0^2}{g/\ell} \leq \left(\frac{1}{mg}\right)^2$$

voor $\dot{\theta} < 0$. In figuur 3.4 staat een afbeelding van het gebied waarop deze oplosmethode toepasbaar is, inclusief de variant voor $\theta < 0$ en $\dot{\theta} > 0$ met omgekeerde u.

In sectie 3.4 komen we in grote mate dezelfde soort ellipsen weer tegen.

3.4 Oplossing 2: Banen in het (x_1, x_2)-vlak

Net als in de sectie 3.3 zoeken we naar een tijdoptimale besturing van ons systeem. Dit levert ons weer een *bang-bang* regelaar op, met de gebruikelijke problemen dat er geen expliciete vorm voor $u(t)$ kan worden gevonden zonder nadere analyse.

We volgen het idee uit [5] met hulp van [1], waarin oplossingen van het dynamische systeem voor vaste waarden van u worden geanalyseerd. Door te kijken naar banen waarover oplossingen zich in het (x_1, x_2)-vlak bewegen, kan worden beredeneerd wanneer en voor welke (x_1, x_2) geswitcht moet worden van invoer.

Deze methode is inzichtelijk omdat het kan worden weergegeven in een $2D$-vlak. Meerdimensionale problemen kunnen ook worden opgelost maar vereisen meer inleving door de wiskundige die de oplossing beschrijft en de lezer. Ook kan een oplossing of een baan van een oplossing minder makkelijk worden weergeven op een $2D$-vlak. Hier zien we in hoofdstuk 4 een voorbeeld van.

3.4.1 Optimale regelaar

We kijken naar de theorie van de optimale regelaars om extra informatie af te leiden over de voorwaarden waaraan een optimale oplossing moet voldoen.

De Hamiltoniaan wordt net als bij de energieoplosmethode

$$H(x, p, u) = p^T f + L = x_2 p_1 - \frac{g}{\ell} x_1 p_2 + \frac{u}{\ell} p_2 + 1.$$

Aannemende dat er een optimale $u(t)$ bestaat, geldt volgens stelling 2.4.1 dat

$$u(t) = \underset{v}{\operatorname{argmin}}\, H(x, p, v) = -\operatorname{sgn}(p_2).$$

Door naar de costates te kijken zullen we extra informatie over u afleiden. Daarmee kunnen we beredeneren hoe de optimale oplossing zich gedraagt, en specifieker, waar de switch punten plaatsvinden en of het er eindig veel zijn.

De vergelijking waaraan de costates voldoen zijn

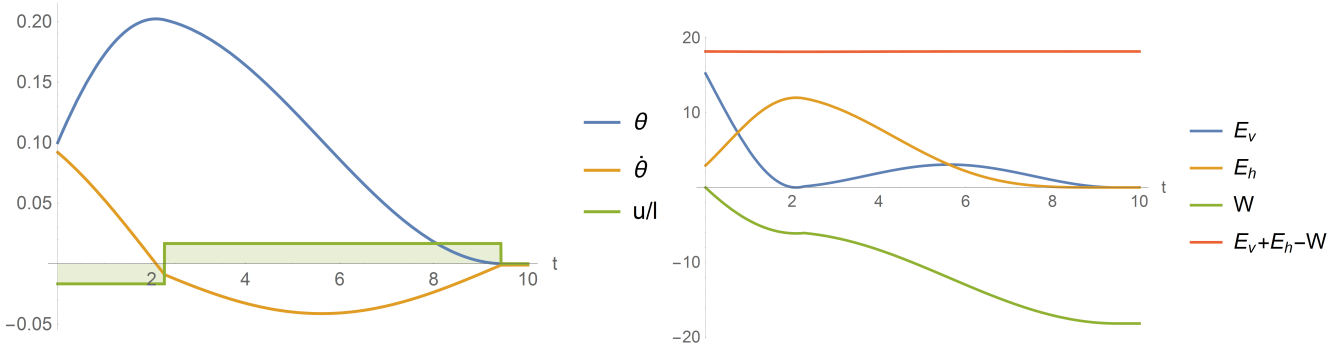
$$\dot{p} = -A^T p,$$

wat de vergelijkingen

$$\begin{aligned} \dot{p}_1 &= -\frac{g}{\ell} p_2 \\ \dot{p}_2 &= p_1 \end{aligned}$$

oplevert. We zijn geïnteresseerd in p_2 omdat $u = -\operatorname{sgn}(p_2)$. De vergelijkingen voldoen aan

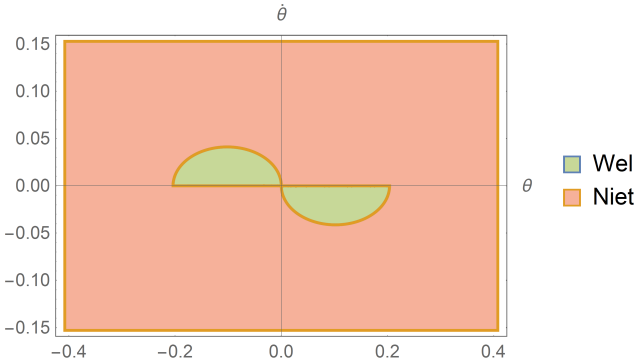
$$p_2 = C_1 \cos\left(\sqrt{\frac{g}{\ell}} t + C_2\right)$$



(a) Uitwijking en hoeksnelheid.

(b) De energieën in het systeem en de verrichte arbeid.

FIGUUR 3.3: Een simulatie van de oplossing met de energie-methode, met switchhoek $\theta_s = 0.056$, switchtijd $t_s = 1.29$ en eindtijd $T = 3.97$.



FIGUUR 3.4: Het gebied van beginvoorwaarden waarvoor de energie-methode wel of niet toepasbaar is.

met vrije parameters C_1 en C_2, dus p_2 heeft een periode van $2\pi\sqrt{\frac{\ell}{g}}$. Elke $\pi\sqrt{\frac{\ell}{g}}$ wisselt het teken van p_2 en daarmee u ook.

Echter kunnen we hiermee voor een bepaalde beginconditie $(\theta_0, \dot{\theta}_0)$ geen expliciete optimale regelaar vinden aangezien er nog altijd twee vrijheidsgraden bestaan in de oplossing. De methode uit [5] wordt hier gebruikt om voor alle mogelijke beginvoorwaarden wel informatie te achterhalen over de optimale oplossing. Daarvoor hebben we voor de verschillende waarden die u kan aannemen informatie nodig over de oplossing.

3.4.2 Vergelijkingen

Neem aan dat voor een bepaalde t_1 geldt dat $u(t_1) = -1$, en voor $t_2 > t_1$ geldt dat $u(t_2) = 1$. De functie $u(t)$ moet dan tussen t_1 en t_2 minimaal eens van waarde zijn veranderd.

Voor de analyse hieronder hernoemen we de variabelen θ in x en $\dot{\theta}$ in y. Zo worden tekeningen in een (x, y)-vlak mogelijk gemaakt en de verbanden inzichtelijk gehouden. De x variabele zal de horizontale as innemen, de y variabele zal de verticale as innemen.

Stel dat $u = 1$. We hebben nu de vergelijkingen

$$\begin{aligned} \dot{x} &= y \\ \dot{y} &= -\frac{g}{\ell}x + \frac{1}{\ell}. \end{aligned}$$

Door $\dot{y}$ te delen door $\dot{x}$ ontstaat

$$\frac{\mathrm{d}y}{\mathrm{d}x} = -\frac{\frac{g}{\ell}x + \frac{1}{\ell}}{y}$$

met een oplossing $\frac{\ell}{g}y^2 = -\left(x^2 - \frac{1}{g}x + C\right)$. Dat geeft de ellips

$$\frac{y^2}{g/\ell} + \left(x - \frac{1}{g}\right)^2 = C.$$

Stel dat $u = -1$. Nu gelden bijna dezelfde vergelijkingen. Oplossen levert

$$\frac{y^2}{g/\ell} + \left(x + \frac{1}{g}\right)^2 = C.$$

Samen levert dit ellipsen op in $\mathbb{R}^2$ met de middelpunten in $(1/g, 0)$ en $(-1/g, 0)$, en alle mogelijke stralen. Voorbeelden van deze cirkels staan in figuur 3.5 in een afbeelding weergegeven ($g = 9.81$ en $\ell = 60$). De ellipsen worden in negatieve richting doorlopen, dus met de klok mee.

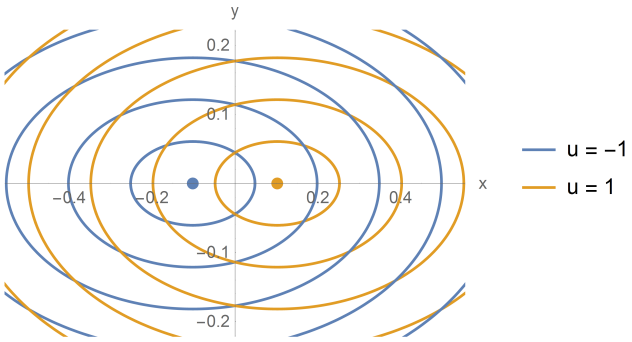
3.4.3 Oplossing

De komende secties zullen vaak de termen ellips, middelpunt en straal bevatten. Een ellips is een verzameling van punten (x, y) die voldoen aan

$$\frac{(x - x_c)^2}{a} + \frac{(y - y_c)^2}{b} = 1$$

voor een vaste x_c, y_c, a en b. Wij zullen hier alleen de vorm

$$(x - x_c)^2 + \frac{(y - y_c)^2}{g/\ell} = r^2$$



FIGUUR 3.5: De ellipsen in het (x, y)-vlak waarover een oplossing zich voor een bepaalde u beweegt.

hanteren omdat dit de ellipsen zijn die voorkomen in de oplossing. Hier is het punt (x_c, y_c) het middelpunt van de ellips en wordt r de straal van de ellips genoemd. Merk op dat in de praktijk de afstand varieert van een punt op de ellips tot het middelpunt van de ellips. In onze situatie is de straal van de ellips is hier alleen gelijk aan op de x-as, wanneer $y = 0$.

Neem aan dat het systeem op $(x_0, y_0) = (\theta_0, \dot{\theta}_0)$ begint. Het doel is naar $(x_e, y_e) = (0,0)$ te komen, op een optimale manier. Dat betekent dat er een *bang-bang* regelaar wordt gebruikt, wat zoals in sectie 3.4.1 is beredeneerd een $u(t) \in \{-1, 1\}$ oplevert.

We weten dat wanneer $u(t) = -1$ of $u(t) = 1$, en $u(t)$ niet verandert, de toestand van het systeem zich in het (x, y)-vlak beweegt in de vorm van een ellips. Door van het eindpunt terug te werken kunnen we een optimale oplossing vinden voor elk beginpunt.

Er zijn twee ellipsen die precies door $(0,0)$ gaan, namelijk die met de middelpunten $(\pm 1/g, 0)$ en straal $1/g$. Eén van de ellipsen $(x \geq 0)$ correspondeert met een u van 1, de andere met een u van -1. We noemen deze twee banen samen Σ_0. Als een oplossing zich op deze baan bevindt, zijn er 0 switches nodig om naar $(0,0)$ te komen. Het doel is dus om een willekeurige baan naar één van die ellipsen toe te sturen.

Om op Σ_0 terecht te komen moet één switch punt plaatsvinden. Er wordt dus veranderd van invoer. Dat betekent dat een (x, y) met nog één switch punt te gaan zich op een andere ellips bevindt dan Σ_0, met een tegenovergesteld middelpunt $(\mp 1/g, 0)$. Het snijpunt van die ellips met Σ_0 is het switch punt (x_1, y_1). We geven alle banen die met één switch punt uitkomen op Σ_0 de naam Σ_1.

Voor een (x, y) buiten Σ_1 herhaalt dit fenomeen zich, totdat (x_0, y_0) zich in Σ_k met $k > 1$ bevindt. In figuur 3.6 is te zien hoe Σ_k eruit zien in het (x, y)-vlak.

De oplettende lezer ziet hier een probleem, namelijk dat snijdende ellipsen (op een aantal specifieke uitzonderingen na) twee snijpunten hebben, en dat bovenstaand gedrag meerdere mogelijke paden heeft door het vlak. Dit wordt opgelost door te kijken naar de costate-vergelijkingen en de resultaten die daar zijn gevonden. Met uitzondering van de eerste en de laatste Σ_k, geldt dat er precies een halve periode, de tijd $\pi\sqrt{\frac{\ell}{g}}$, wordt doorgebracht in Σ_k. Dat betekent dat precies een halve ellips wordt doorlopen (dit geldt wegens de symmetrie van een ellips) wanneer een switch heeft plaatsgevonden en (x, y) zich niet op Σ_0 bevindt.

Het resultaat is dat Σ_0 twee halve ellipsen vormt, de ene met $x > 0, y < 0$, de andere $x < 0, y > 0$. Precies een halve ellips uit Σ_1 verder ontstaat aan de andere kant van de y-as een halve ellips met straal $1/g$, en middelpunt $(\pm 3/g, 0)$. Dit proces herhaalt zich ook voor alle Σ_k, voor beide zijdes van de y-as. Het resultaat is een draai-symmetrisch verdeeld vlak, met twee rijen halve ellipsen met straal $1/g$ boven of onder de x-as. Boven deze ellipsen geldt $u = -1$, onder de ellipsen geldt $u = 1$ voor een optimale regelaar. De grens wordt Γ genoemd. De grafische weergave van Γ in het (x, y)-vlak met bijbehorende u is te zien in figuur 3.7.

Deze verdeling van het (x, y)-vlak geeft ons een unieke u. Voor elk punt (x, y) is gegeven of $u = 1$ en $u = -1$, ook op de grens Γ. Op Γ geldt voor $x > 0$ dat $u = 1$ en voor $x < 0$ dat $u = -1$. Daarmee is ook de laatste ellipsbaan van een oplossing die over Σ_0 loopt (theoretisch) goed gedefinieerd. In de praktijk ziet elke oplossing eruit als een spiraal met 'knikken' erin wanneer Γ wordt gepasseerd, totdat de baan op Σ_0 uitkomt en naar $(0,0)$ loopt.

In tegenstelling tot de theorie die hier gepresenteerd wordt, het domein van (x, y) beperkt moeten worden tot een kleine (x_0, y_0) omdat het systeem en de linearisering geen grote waardes toelaat. Daarom zijn grote k praktisch niet mogelijk, afhankelijk van de beperking.

3.4.4 Simulatie

Om een simulatie te doen is meer informatie nodig dan alleen de definitie van de halve ellipsen waarop $u = 1$ naar $u = -1$ en andersom verandert. Aangezien een simulatie numeriek door een computer wordt gedaan moeten de gebieden waarin u een bepaalde waarde heeft zodanig worden gedefinieerd voor een $(\theta_0, \dot{\theta}_0)$ dat de simulatie een daadwerkelijke grens over gaat bij een wisseling van invoer, en niet zich precies op de grens van twee gebieden beweegt.

Hier volgen een aantal definities die nuttig zijn bij het bepalen van een algoritme om gebieden te bepalen waarin een bepaalde u geldt, gegeven $(\theta_0, \dot{\theta}_0)$. In een aantal functies staan extra haakjes om punten heen. Dit geeft duidelijkheid over de groepering van de argumenten van de functie.

De functie $E((x, y), (x_c, y_c), r)$ geeft aan of een punt (x, y) zich strikt binnen een ellips met centrum (x_c, y_c)

en straal r. Het is gedefinieerd als

$$E\left((x,y),(x_c,y_c),r\right) := (x-x_c)^2 + \frac{(y-y_c)^2}{g/\ell} < r^2.$$

Hiermee wordt een functie $A((x,y))$ gedefinieerd waarmee kan worden bepaald of een punt (x,y) zich boven Γ bevindt. Merk op dat deze functie niet kan worden gebruikt in een simulatie aangezien Σ_0 zich precies op de grens van waar/onwaar bevindt, en numerieke fouten ervoor zorgen dat u ongewenst herhaaldelijk zal wisselen tussen -1 en 1. De functie A wordt als

$$A((x,y)) := \begin{cases} y>0 \vee E\left((x,y),\left(\left(2\left\lfloor\frac{x}{2/g}\right\rfloor+1\right)\frac{1}{g},0\right),\frac{1}{g}\right) & x \geq 0 \\ \neg A((-x,-y)) & x<0 \end{cases}$$

gedefinieerd, met $\lfloor\cdot\rfloor$ als afronden naar beneden. Deze functie kan echter wel goed worden gebruikt om te bepalen in welk vlak het punt (x_0,y_0) zich bevindt.

Verder wordt een functie $S((x,y))$ gedefinieerd die voor een bepaalde (x,y) het volgende snijpunt met Γ bepaalt. Met deze functie kan een serie worden gemaakt die recursief alle snijpunten met Γ vindt. De functie is gedefinieerd als

$$S((x,y)) = \begin{cases} x_s = \frac{r^2-(1/g)^2+((2k+1)^2-1)(1/g)^2}{4(k+1)(1/g)}; & \\ \left(x_s, -\sqrt{\frac{g}{\ell}\left(r^2-\left(x_s+\frac{1}{g}\right)^2\right)}\right) & A((x,y)) \\ -S((-x,-y)) & \neg A((x,y)) \end{cases},$$

waarbij

$$r^2 = \left(x+\frac{1}{g}\right)^2 + \frac{y^2}{g/\ell}$$

en

$$k = \left\lfloor \frac{r-1/g}{2/g} \right\rfloor.$$

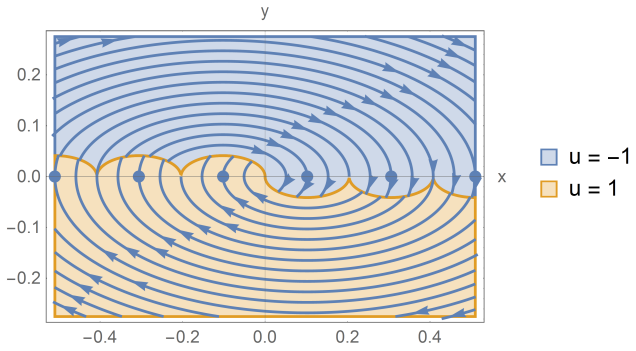
Met behulp van $S((x,y))$ kan recursief een reeks van (x_s,y_s) worden gemaakt: namelijk de snijpunten met Γ. Laat $(x_{s_0},y_{s_0}) = (x_0,y_0)$ en

$$(x_{s_k},y_{s_k}) = S\left((x_{s_{k-1}},y_{s_{k-1}})\right)$$

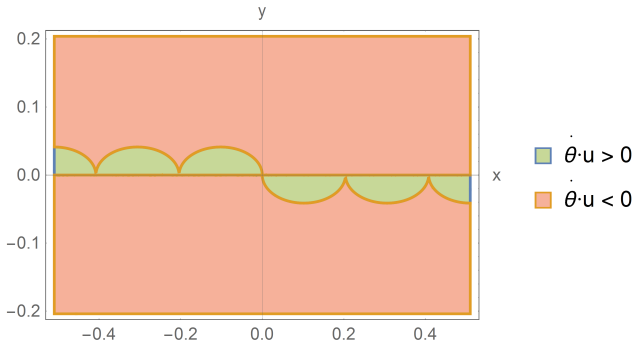
voor $0 < k \leq n$, met n het aantal snijpunten met Γ, oftewel totdat een snijpunt met Σ_0 is gevonden.

Ten slotte wordt de optimale $u(t) = u(x(t),y(t))$ gedefinieerd aan de hand van

$$u(x,y) = \begin{cases} 1 & x<0 \wedge y<0 \\ -1 & x>0 \wedge y>0 \\ -\mathrm{sgn}\left(x_{s_1}\right) & |x|>\left|x_{s_1}\right| \\ & \wedge E\left((x,y),\left(\left(2\mathrm{sgn}\left(x_{s_1}\right)\left\lfloor\frac{x_{s_1}}{2/g}\right\rfloor+1\right)\frac{1}{g},0\right),\frac{1}{g}\right) \\ \vdots & \vdots \\ -\mathrm{sgn}\left(x_{s_n}\right) & |x|>\left|x_{s_n}\right| \\ & \wedge E\left((x,y),\left(\left(2\mathrm{sgn}\left(x_{s_n}\right)\left\lfloor\frac{x_{s_n}}{2/g}\right\rfloor+1\right)\frac{1}{g},0\right),\frac{1}{g}\right) \\ -\mathrm{sgn}(y) & \text{anders} \end{cases}$$



FIGUUR 3.6: De verdeling van banen in het (x,y)-vlak. De scheiding tussen de twee vlakken is Γ.



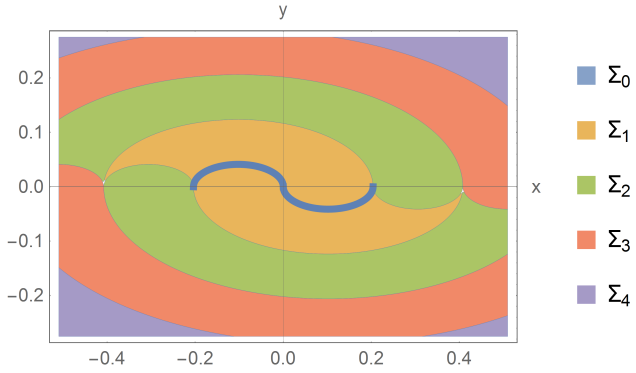
FIGUUR 3.7: De richting van de kracht u ten opzichte van $\dot{\theta}$.

Merk op dat dit alleen een u geeft voor de huidige toestand $(x,y) = (\theta,\dot{\theta})$, en geen afhankelijkheid heeft van t. In sectie 3.4.6 worden de doorlooptijden van elke Σ_k (analytisch) doorgerekend per deel van de ellips.

3.4.5 Convergentie naar (0,0)

Met deze oplossing wordt de numerieke oplossing naar $(0,0)$ gestuurd. Echter zal de simulatie numeriek nooit precies in $(0,0)$ uitkomen, immers is het systeem stabiel en niet asymptotisch stabiel.

Door dicht bij $(0,0)$ te kiezen voor niet-optimale invoer kan een asymptotisch stabiel systeem worden ver-



FIGUUR 3.8: Verschillende representaties van het (x,y)-vlak voor de oplosmethode uit sectie 3.4.

kregen. De volgende eisen worden gesteld voor een $|(x,y)| < \varepsilon_F$:

- $u = F \cdot (x,y)$ en $|u| \leq 1$;
- $A + BF$ heeft eigenwaarden -1.

Aan deze voorwaarden kan gemakkelijk worden voldaan door een F te kiezen met $|F| \leq 1/\varepsilon_F$. Er moet nog een F worden gekozen met de correcte eigenwaarden maar dit is een systeem met twee onbekenden en twee vrijheidsgraden. Aangezien ε_F vrij is kan deze kleiner worden gekozen wanneer $|F|$ een te grote waarde heeft.

Voor $g = 9.81$ en $\ell = 60$ geldt bijvoorbeeld dat $F = (-50.2, -120.0)$.

3.4.6 Doorlooptijden oplossen

Het is nuttig informatie te hebben over de de doorlooptijden van een ellipsbaan, aangegeven met T_k voor een ellipsbaan tot het $k$de switchpunt, en T_n de laatste ellipsbaan tot $(0,0)$. Merk op dat de indices k precies andersom worden doorlopen als het gebied Σ_k waarin een ellipsbaan zich bevindt.

Voor een beginpunt (x_0, y_0) kunnen alle snijpunten met Γ worden berekend door middel van $S((x,y))$ herhaaldelijk toe te passen.

Gegeven een paar opvolgende punten met (x_0, y_0) en $(0,0)$ meegerekend (noem het paar $\{(x_k, y_k), (x_{k+1}, y_{k+1})\}$) kan nu de analytische oplossing worden berekend voor de baan die de slinger aflegt, gegeven u.

Immers, de vergelijking voor θ voldoet aan

$$\ell\ddot{\theta} = -g\theta + u.$$

De oplossing van deze vergelijking heeft twee vrijheidsgraden die ingevuld kunnen worden door de uitwijking en hoeksnelheid van het eerste punt van het paar. Daarmee kan het tijdstip worden bepaald waar het tweede punt van het paar de oplossing snijdt. Hier moet goed rekening worden gehouden dat het voor kan komen dat pas de tweede keer dat de slinger een bepaalde hoek passeert wordt geswitcht: dit komt door het teken van $\dot{\theta}$.

Numeriek kan dit worden uitgewerkt naar een viertal vaste punten, waarop een korte analyse wordt uitgevoerd. Deze punten zijn bepaald als de vier oplossingen voor t het dichtst bij $t = 0$ van de vergelijking

$$\xi(t) = C$$

waarbij $\xi(t)$ de oplossing is van

$$\ell\ddot{\xi} = -g\xi + u, \quad \xi(0) = x_0, \quad \dot{\xi}(0) = y_0$$

en gegeven een x_0, y_0, u en C zodat er oplossingen bestaan. De vorm van de expliciete formule voor deze vier oplossingen is te zien in figuur 3.9. Sommige daarvan kunnen kleiner dan 0 zijn. In dat geval moet er $2\pi\sqrt{\frac{\ell}{g}}$ bij worden opgeteld: één periode van de ellipsbaan.

Van deze waarden wordt het minimum genomen, onder voorwaarde dat het imaginaire deel nul is. De wortel kan namelijk imaginaire oplossingen geven. We geven de doorlooptijd van de ellipsbaan voor het paar $\{(x_k, y_k), (x_{k+1}, y_{k+1})\}$ de naam T_k. De som van deze tijden,

$$\sum_k T_k$$

is gelijk aan de totale tijd T om van het punt (x_0, y_0) naar het punt $(0,0)$ te komen. Voor alle paren waar (x_0, y_0) en $(0,0)$ niet in voorkomen geldt $T_k = \pi\sqrt{\frac{\ell}{g}}$. Immers geldt dat behalve voor het eerste en laatste stuk van de baan van de oplossing steeds een halve periode van een ellips wordt doorlopen.

Hierbij moet de kanttekening worden geplaatst dat de totaaltijd T in de simulatie niet precies klopt. Binnen een straal ε_F rond de oorsprong het systeem verandert naar een asymptotisch stabiel systeem met $u = F \cdot (x,y)$. Daardoor wijkt T_n in de praktijk af van de analytische waarde, en daarom T ook.

In figuur 3.10 staat een plot van de minimale totale tijd die het kost om naar $(0,0)$ te komen vanuit een beginpositie. De vormen van de ellipsen boven en onder de x-as zijn duidelijk te zien.

3.4.7 Voorbeelden

In deze voorbeelden nemen we $g = 9.81$ en $\ell = 60$ aan. Voor ε_F wordt de waarde $2.5 \cdot 10^{-5}$ gebruikt. De simulatie wordt afgebroken wanneer $|(x,y)| < 10^{-5}\varepsilon_F$.

In figuur 3.11 staat een simulatie met $(x_0, y_0) = (0.1, 0.03)$. Er is goed te zien dat de theoretische ellipsen goed overeenkomen met de simulatie. Er vindt één switch punt plaats.

In figuur 3.12 staat een simulatie met $(x_0, y_0) = (-0.2, 0.2)$. Hier vinden twee switch punten plaats.

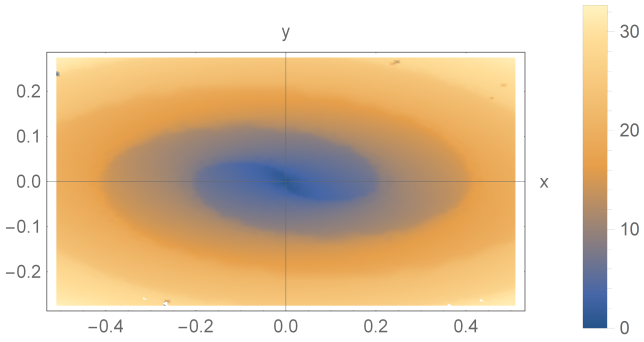
3.5 Oplossing 3: Kosten op invoer en eindtoestand

Bij deze oplosstrategie passen we de kostenfunctie aan door extra kosten toe te voegen op de invoerkracht. Er zal kort worden uitgelegd wat dit betekent voor de kostenfunctie en de optimale regelaar, waarna een expliciete optimale oplossing voor $u(t)$ wordt bepaald. Ook wordt uitgelegd hoe de optimale T kan worden bepaald.

Ten slotte wordt een korte discussie gevoerd over het kiezen van de goede waarden voor de parameters die worden geïntroduceerd voor de oplossing.

$$T_k := \pm\sqrt{\frac{\ell}{g}}\cos^{-1}\Bigg(\Big(\big(g^2x_kx_{k+1} - gux_k - gux_{k+1} + u^2\big) \pm \sqrt{g^3\ell x_k^2y_k^2 - g^3\ell x_{k+1}^2y_k^2 + g^2\ell^2y_k^4 - 2g^2u\ell x_ky_k^2 + 2g^2u\ell x_{k+1}y_k^2}\Big) \Big/ \big(g^2x_k^2 + g\ell y_k^2 - 2gux_k + u^2\big)\Bigg)$$

FIGUUR 3.9: De vorm van de doorlooptijden van het stuk van de ellips van het paar $\{(x_k, y_k), (x_{k+1}, y_{k+1})\}$, gegeven u. Elke $\pm$ geeft een mogelijke oplossing, wat er in totaal vier oplevert.



FIGUUR 3.10: De minimale totale tijd die het kost om vanuit een willekeurig beginpunt naar het punt $(0,0)^T$ te komen met een optimale u. De afwijkende vlekjes zijn numerieke fouten bij het bepalen of een punt binnen een ellips ligt.

3.5.1 Optimale regelaar

Laat $\varepsilon > 0$ en $s > 0$. We kiezen nu de kostenfunctie als

$$J = sx(T)^2 + \int_0^T 1 + \varepsilon u(t)^2 \mathrm{d}t$$

voor een gekozen vaste T. Merk op dat een oplossing die hiermee wordt gevonden wordt voor deze kostenfunctie optimaal is, maar nooit tijd-optimaal voor het slingerprobleem. Een kleine ε geeft een benadering voor het originele tijd-optimale probleem. Door een grote waarde te kiezen voor s wordt de eindtoestand $x(T)$ harder richting $(0,0)^T$ gestuurd.

Het toevoegen van een ε en s in de kostenfunctie geeft twee gevolgen. Ten eerste is het nu mogelijk meer optimalisatietechnieken te gebruiken om een optimale u te bepalen. Er kan worden namelijk worden afgeleid dat dit een een 'gladde' $u(t)$ garandeert die Lipschitz continu is.

Ten tweede geeft een kleine ε een realistische kostenfunctie. Het toevoegen van kosten op de invoer u zorgt ervoor dat het iets 'kost' om invoer te leveren. Daardoor zal een switch nooit spontaan van -1 naar 1 gaan, maar vloeiend. Het voordeel dat dit geeft is een $u(t)$ die continu differentieerbaar is en bovendien een afgeleide heeft in $t = 0$. Daarmee kan er wel gelineariseerd worden in tegenstelling tot een stapfunctie.

We kunnen nu niet meer stelling 2.4.1 toepassen omdat de kostenfunctie is veranderd, maar vallen terug op de stelling 2.3.1. Dit levert ons de Hamiltoniaan

$$H = p^T f + L = \left(p_1x_2 - \frac{g}{\ell}x_2p_2 + \frac{u}{\ell}p_2\right) + \left(1 + \varepsilon u^2\right)$$

We vinden de optimale $u(t)$ door voor alle t de u te kiezen die H minimaliseert, oftewel

$$\begin{aligned} u(t) &= \operatorname*{argmin}_v \left(p_1x_2 - \frac{g}{\ell}x_2p_2 + \frac{v}{\ell}p_2\right) + \left(1 + \varepsilon v^2\right) \\ &= \operatorname*{argmin}_v \frac{v}{\ell}p_2 + \varepsilon v^2. \end{aligned}$$

Aangezien $\ell > 0$ moet gelden dat

$$u = -\frac{1}{2\ell\varepsilon}p_2.$$

Definieer $R = 2\varepsilon$, twee keer de kosten die we op de invoer hebben gezet in de kostenfunctie J. Dit is een gebruikelijke naamgeving binnen de theorie van optimale regelaars voor een gelineariseerd dynamisch systeem. We hebben nu

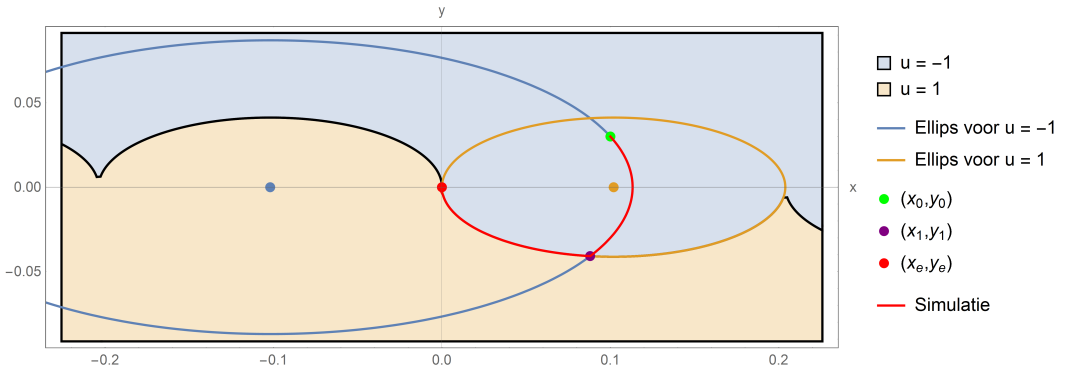
$$u = -R^{-1}B^Tp = -\frac{1}{2\ell\varepsilon}p_2,$$

een resultaat dat ook direct is af te leiden uit een optimale regelaar probleem met willekeurige matrices A, B en R (en een optionele Q die in ons probleem niet voorkomt).

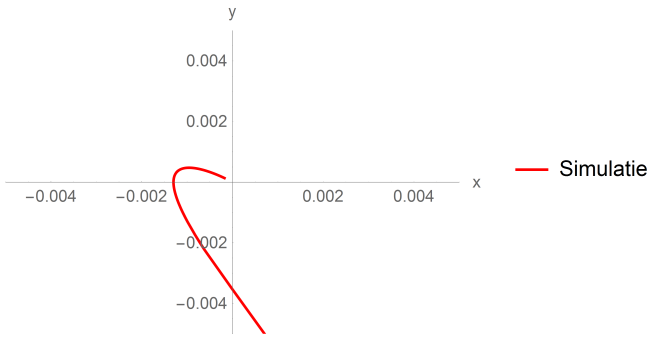
Merk op dat de voorwaarde dat $|u(t)| \leq 1$ voor alle t hier wordt genegeerd. Als bijvoorbeeld $p_2 > 2\ell\varepsilon$ op een bepaald tijdstip is $u > 1$. In sectie 3.5.3 wordt uiteengezet hoe aan de voorwaarde voor $u(t)$ kan worden voldaan door de waarde van T te variëren om daarmee een geldige $u(t)$ en een oplossing voor het optimalisatieprobleem te vinden.

3.5.2 Simulatie

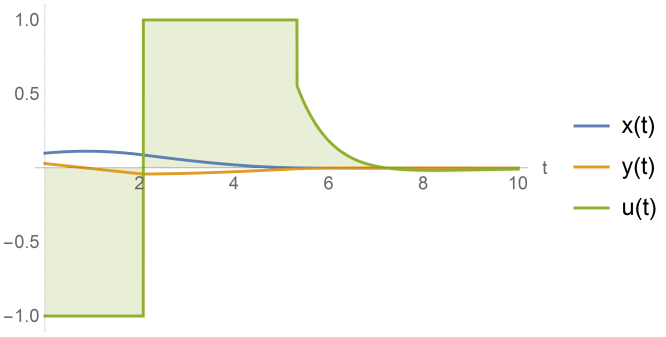
Een simulatie kan worden uitgevoerd door het systeem voor $(x_1, x_2)^T$ te combineren met dat van $(p_1, p_2)^T$, om



(a) De banen van de slinger, de ellipsen en de switch punten.

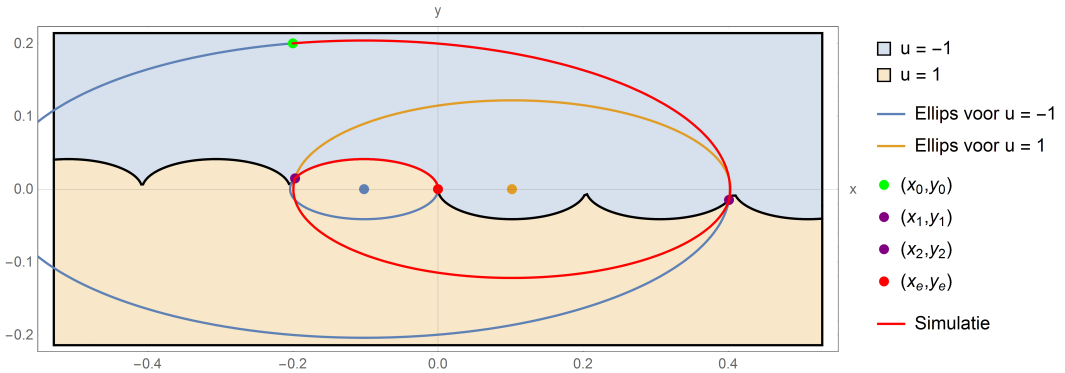


(b) De simulatie dicht bij het punt $(0,0)$, met $u = Fx$.

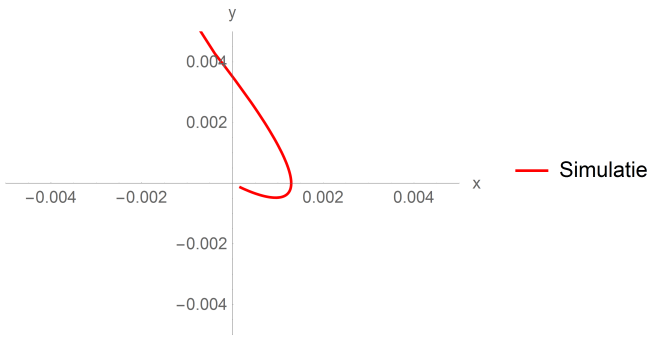


(c) De uitwijking x, hoeksnelheid y en invoerkracht u van de slinger.

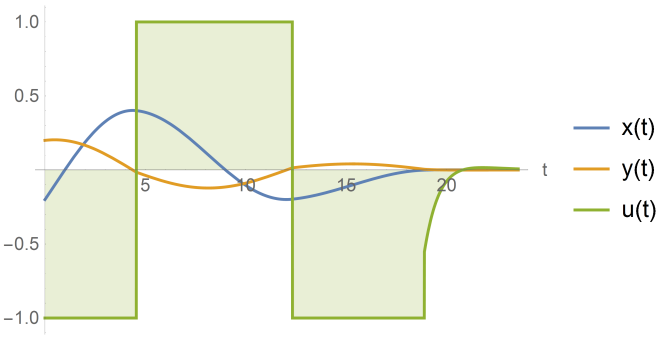
FIGUUR 3.11: Een simulatie van een oplossing met de Hautus-methode met een enkel switch punt.



(a) De banen van de slinger, de ellipsen en de switch punten.



(b) De simulatie dicht bij het punt $(0,0)$, met $u = Fx$.



(c) De uitwijking x, hoeksnelheid y en invoerkracht u van de slinger.

FIGUUR 3.12: Een simulatie van een oplossing met de Hautus-methode met meerdere switch punten.

zo één groot systeem met de variabelen $(x_1, x_2, p_1, p_2)^T$ te krijgen, onder invloed van een matrix H.

We hebben

$$\dot{x} = Ax + Bu, \quad \dot{p} = -A^T p$$

en krijgen als gecombineerd systeem

$$(\dot{x}_1, \dot{x}_2, \dot{p}_1, \dot{p}_2)^T = H \cdot (x_1, x_2, p_1, p_2)^T$$

met

$$\begin{aligned} H &= \begin{pmatrix} A & -BR^{-1}B^T \\ Q & -A^T \end{pmatrix} \\ &= \begin{pmatrix} 0 & 1 & 0 & 0 \\ -g/\ell & 0 & 0 & -1/(2\ell^2\varepsilon) \\ 0 & 0 & 0 & g/\ell \\ 0 & 0 & -1 & 0 \end{pmatrix}. \end{aligned}$$

Voor dit proces gelden de begin- en eindcondities $x(0) = x_0$ en $p(T) = sx(T)$, in totaal n voorwaarden die het systeem goed gedefinieerd maken.

3.5.3 Binary search naar T

Deze simulatie kan numeriek (soms zelfs analytisch) gemakkelijk worden opgelost. Echter moeten er waardes voor s, ε en T worden gekozen.

We kiezen voor deze oplossing de waarden van s en ε vast. Echter is er dan nog een vrije parameter T over, de eindtijd van de simulatie. Deze is vrij te kiezen: het dynamische systeem minimaliseert de kostenfunctie J tussen de tijdstippen 0 en T.

Dat kan veroorzaken dat $|u|$ groter wordt dan 1 wanneer de waarde voor T klein wordt gekozen, iets wat niet mag volgens de voorwaarden van het systeem. Het is dus de taak om een zo klein mogelijke eindtijd T te vinden waarvoor $|u(t)| \leq 1$ voor alle t, zodat de oplossing is toegestaan en het systeem in zo'n kort mogelijke tijd in $(0,0)$ uitkomt.

Dit doen we met een *binary search* die bestaat uit de volgende stappen:

- Kies een $T_{\min}$ en een $T_{\max}$, met $T_{\min} < T_{\max}$. Kies ook een $\varepsilon_B > 0$ die bepaalt wanneer de binary search stopt.
- Bepaal
$$u_{m,T} = \max_{t \in [0,T]} |u(t)|,$$
de maximale $|u|$ voor een simulatie met $T = (T_{\min} + T_{\max})/2$.
  - Als $u_{m,T} > 1$, zet dan $T_{\min} = (T_{\min} + T_{\max})/2$.
  - Als $u_{m,T} < 1$, zet dan $T_{\max} = (T_{\min} + T_{\max})/2$.
- Herhaal bovenstaande stappen totdat $T_{\max} - T_{\min} < \varepsilon_B$.

Merk op dat een dergelijk algoritme alleen werkt als $u_{m,T_1} > u_{m,T_2}$ dan en slechts dan als $T_1 < T_2$, oftewel $u_{m,T}$ moet een strikt dalende functie van T zijn. Echter is het algoritme triviaal aan te passen om te werken voor een strikt stijgende functie.

In het geval van ons probleem is $u_{m,T}$ zoals aangenomen strikt dalend voor T.

3.5.4 Voorbeeld

Met de oplosmethode die in deze sectie is toegelicht is een simulatie gedaan ter voorbeeld. De gebruikte beginvoorwaarden en parameters voor het dynamische systeem zijn respectievelijk $x_1(0) = 0.2$, $x_2(0) = -0.2$ en $\ell = 60$, $g = 9.81$, $s = 10^4$, $\varepsilon = 1$. Om een optimale eindtijd T te vinden zijn voor de binary search de parameters $T_{\min} = 0$, $T_{\max} = 40$ en $\varepsilon_B = 10^{-2}$. In figuur 3.13 is het resultaat van de simulatie te zien.

3.5.5 Observaties

De verhouding tussen s en ε stelt de verhouding voor van de kosten op de eindtoestand $x(T)$ van het systeem en de kosten van de invoer. Er geldt namelijk

$$\begin{aligned} J &= sx(T)^T x(T) + \int_0^T 1 + \varepsilon u(t)^2 \mathrm{d}t \\ &= s\left(x(T)^T x(T)\right) + \varepsilon \left(\int_0^T u(t)^2 \mathrm{d}t \right) + T. \end{aligned}$$

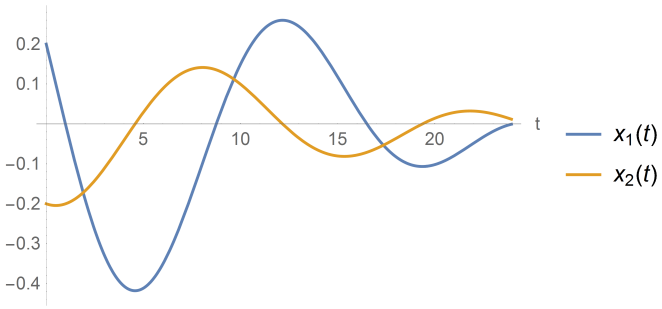
Hier zijn voor een vaste T altijd de kosten T vast, en bepaalt de verhouding s/ε hoe de minimale kosten over $x(T)^T x(T)$ en de integraal over de invoer worden verdeeld. Het liefst willen we deze verhouding zo groot mogelijk hebben: weinig kosten op de invoer zijn geen groot probleem, en we willen de eindtoestand zo klein mogelijk krijgen.

Het zou interessant zijn om de limit van de verhouding s/ε naar 0 te nemen, aangezien we op dat punt uitkomen bij de eindvoorwaarden

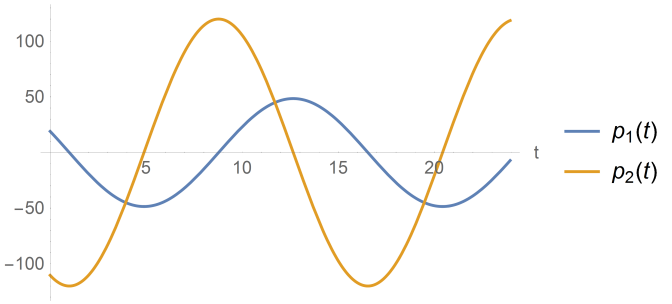
$$p(T) = 0x(T) = 0,$$

en de kosten $sx(T)^T x(T)$ de eindtoestand naar 0 dwingen voor een bepaalde T. We benaderen dan het tijdoptimale probleem. Echter schuilt er een numeriek probleem dat ontstaat wanneer de matrix H wordt gemaakt en opgelost. Hoe kleiner de gekozen ε, hoe instabieler de matrix is. Wiskundig gezien gaat het conditiegetal van de matrix naar oneindig (∞), wat een numerieke fout in het begin van de simulatie oneindig groot laat worden wanneer de simulatie klaar is.

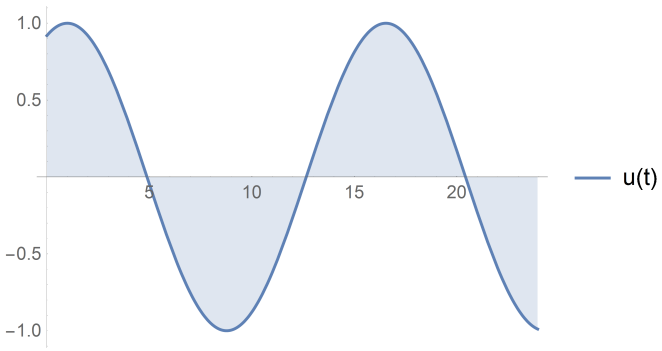
Men zou kunnen stellen dat een ε die niet te klein is en een hele grote s dezelfde verhouding geeft. Echter geeft dit dezelfde numerieke problemen. Niet in de matrix H, maar tijdens het numeriek integreren van de oplossingen voor x en p. Er moet dan namelijk een



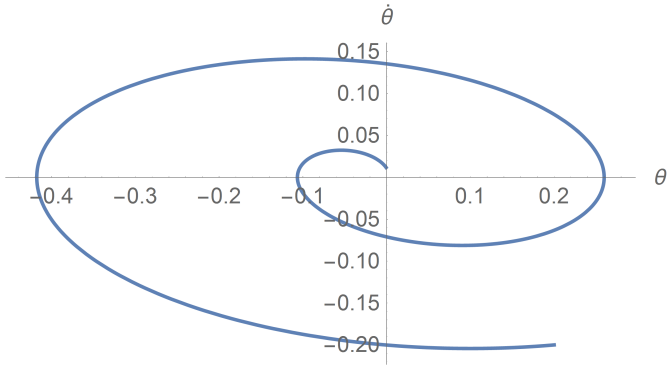
(a) De hoek en hoeksnelheid van de slinger



(b) De costates



(c) De invoer



(d) De beweging van de simulatie in het $(\theta,\dot{\theta})$-vlak

FIGUUR 3.13: Simulatie van de slinger met de kosten-methode, met parameters $\ell = 60$, $g = 9.81$, $s = 10^4$, $\varepsilon = 1$ en $\varepsilon_F = 10^{-5}$. De gevonden T is 23.94 en $|x(T)| = 0.012$.

grote verhouding in begin- en eindtoestanden van de verschillende variabelen worden overwonnen en berekend.

Experimenten laten zien dat een verhouding van s/ε van tussen de 10^4 en 10^6 goede resultaten geeft. In figuur 3.14 staat een tabel met alle gevonden resultaten. Hoewel het lijkt dat de fout van $x(T)$ naar 0 convergeert geeft Mathematica (waarmee de simulaties worden uitgevoerd) convergentiefouten weer bij de tweede helft van de tabel. Het is daarom niet duidelijk of de laatste helft van de resultaten betrouwbaar genoeg zijn, al volgen ze wel de trend van de waarden ervoor.

3.6 Vergelijking Oplosmethodes

We zullen voor vier beginvoorwaarden een vergelijking maken tussen de drie methodes. Alle methodes zijn zo gebruikt dat de gemeten tijd de tijd is van $(\theta_0,\dot{\theta}_0)$ tot de oplossing binnen een afstand $5\cdot10^{-3}$ van $(0,0)$ komt. Voor de derde oplosmethode zijn daarvoor $s = 10^5$, $\varepsilon = 1$ en $\varepsilon_B = 10^{-2}$ gebruikt. De laatste twee vergelijkingen bestaan alleen uit methode 2 en 3 omdat methode 1 niet toepasbaar is wegens de beginvoorwaarden.

De resultaten van de vergelijking staan in figuur 3.15. Tevens staan afbeeldingen van de banen die worden afgelegd voor de vier begincondities in figuur 3.16. Hoewel de baan van methode 2 bijna volledig achter de baan van methode 1 verschuilt, is hij op het eind van

s	ε	s/ε	Conditiegetal H	$\lvert x(T)\rvert$
$10^{1/2}$	$10^{-1/2}$	10^1	$6.1\cdot10^0$	$2.8\cdot10^{-1}$
10^1	10^{-1}	10^2	$6.2\cdot10^0$	$2.8\cdot10^{-1}$
$10^{3/2}$	$10^{-3/2}$	10^3	$6.3\cdot10^0$	$6.4\cdot10^{-2}$
10^2	10^{-2}	10^4	$6.6\cdot10^0$	$1.2\cdot10^{-2}$
$10^{5/2}$	$10^{-5/2}$	10^5	$7.8\cdot10^0$	$1.2\cdot10^{-3}$
10^3	10^{-3}	10^6	$1.1\cdot10^1$	$1.2\cdot10^{-4}$
$10^{7/2}$	$10^{-7/2}$	10^7	$2.3\cdot10^1$	$1.2\cdot10^{-5}$
10^4	10^{-4}	10^8	$9.0\cdot10^1$	$1.3\cdot10^{-6}$
$10^{9/2}$	$10^{-9/2}$	10^9	$7.7\cdot10^2$	$2.1\cdot10^{-7}$
10^5	10^{-5}	10^{10}	$7.4\cdot10^3$	$1.0\cdot10^{-7}$
$10^{11/2}$	$10^{-11/2}$	10^{11}	$7.3\cdot10^4$	$8.4\cdot10^{-8}$
10^6	10^{-6}	10^{12}	$7.2\cdot10^5$	$8.5\cdot10^{-8}$

FIGUUR 3.14: Resultaten van het testen met verschillende verhoudingen tussen s en ε voor het simuleren met de matrix H. Onder de stippellijn bevatten de berekeningen significante numerieke fouten.

		Oplosmethode		
θ_0	$\dot{\theta}_0$	1	2	3
-0.05	0.01	3.87	2.59	3.36
0.15	0.00	5.41	5.27	6.25
0.10	0.02	–	4.89	5.99
0.30	0.05	–	12.7	17.4

FIGUUR 3.15: Vergelijking van oplostijden T van de drie oplosmethodes voor verschillende begincondities.

zijn baan te zien wanneer wordt overgegaan op $u = Fx$.

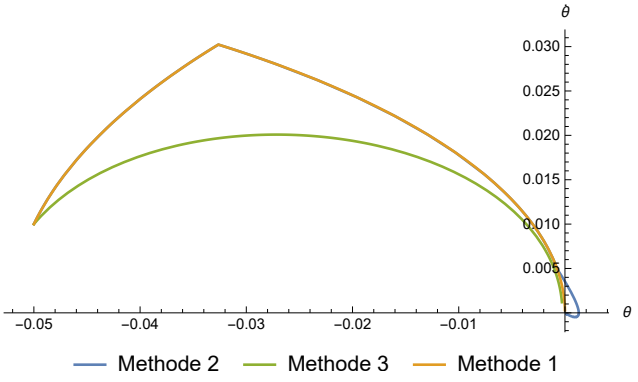
Er kunnen een aantal opmerkingen worden gemaakt als de oplostijden worden vergeleken. Ten eerste werkt methode 2 beter dan methode 1, en methode 1 beter dan methode 3. Echter maken methode 1 en 2 letterlijk dezelfde baan dus zal dit met afrondfouten van de simulatie (en eindtijden) te maken hebben. Het is logisch dat de tijd-optimale oplosmethodes snellere oplossingen geven dan methode 3, aangezien methode 3 niet tijd-optimaal is opgezet.

Ten tweede zijn de eerste twee begincondities voor vergelijkingen zo gekozen dat alle drie de methodes uitgevoerd kunnen worden. Meerdere switchpunten zijn niet mogelijk voor methode 1, evenals beginnen met $\theta_0 > 0$ en $\dot{\theta}_0 > 0$ of $\theta_0 < 0$ en $\dot{\theta}_0 < 0$.

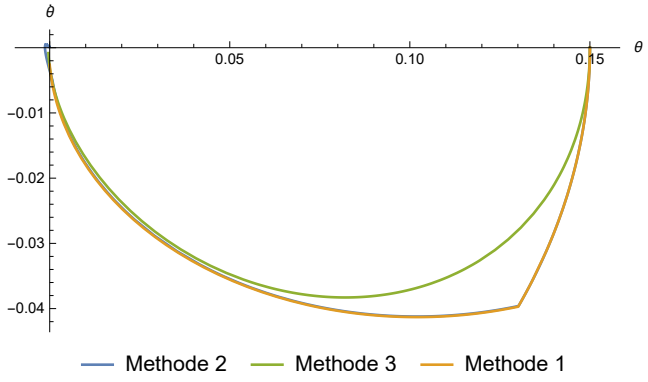
3.7 Conclusie

We hebben in de secties hierboven drie manieren gezien om een optimale regelaar te vinden voor het probleem van de slinger. Met de methode van de energie kan alleen één switchpunt worden beredeneerd, maar de methode is wel erg intuïtief. Met de oplosmethode met banen in het $(\theta, \dot{\theta})$-vlak kan voor elke beginvoorwaarde een unieke en optimale bang-bang regelaar worden gevonden. Deze methode werkt goed, maar is erg specifiek voor het slingerprobleem. Ten derde hebben we een oplosmethode gezien met kosten op de invoer en eindtoestand van het systeem waardoor de tijd-optimaliteit werd losgelaten. De methode is erg algemeen en is daarom breder inzetbaar dan alleen het probleem van de slinger.

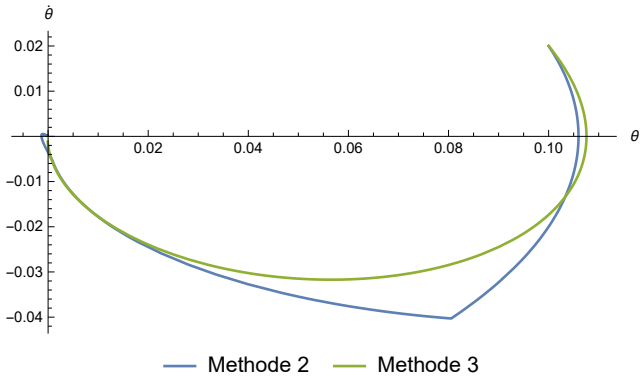
Bij een vergelijking zien we dat de tweede oplosmethode zoals verwacht beter presteert dan de andere twee methodes.



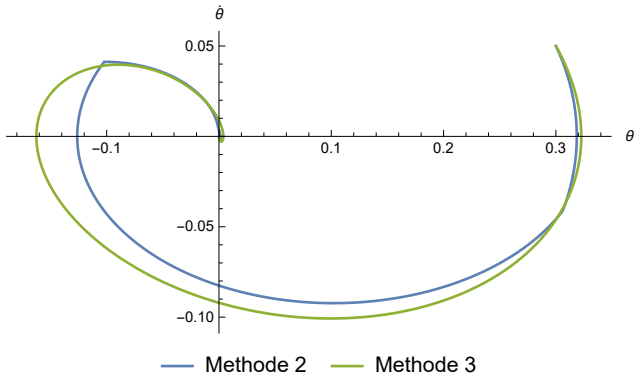
(a) Begincondities $\theta_0 = -0.05, \dot{\theta}_0 = -0.01$.



(b) Begincondities $\theta_0 = 0.15, \dot{\theta}_0 = 0.00$.



(c) Begincondities $\theta_0 = 0.10, \dot{\theta}_0 = 0.02$.



(d) Begincondities $\theta_0 = 0.3, \dot{\theta}_0 = 0.05$.

FIGUUR 3.16: Vergelijking van afgelegde banen van de drie oplosmethodes voor verschillende begincondities. Methode 1 staat achteraan omdat deze anders niet is te zien achter methode 2.

Hoofdstuk 4

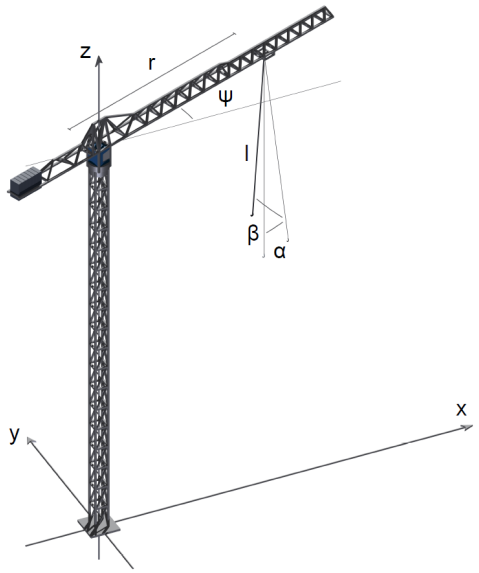
Tweede probleem: een hijskraan

4.1 Inleiding

In hoofdstuk 3 hebben we gezien hoe de optimale regelaar van een slinger kan worden gevonden, door te redeneren over de switchpunten en switchtijden van de invoer en de gevolgen die dat heeft voor het verloop van het dynamische systeem.

Deze theorie en oplosmethodes gaan we toepassen in een situatie die minder simpel is dan de slinger, maar er wel erg op lijkt. We gaan kijken naar het optimale regelaar probleem van de kraan. We stellen ons een kraan voor als een lange arm (de *boom*), waarop een karretje rijdt met daaraan een gewicht, hangend aan een kabel van vaste lengte. Het doel is het verplaatsen van het het karretje van één positie naar een andere positie op de boom, in minimale tijd, zonder slingering van het gewicht bij aankomst.

Eerst wordt het model van de kraan gepresenteerd en daarna twee oplosmethodes van het optimale regelaar probleem.



FIGUUR 4.1: Een visuele representatie van het basismodel.

4.2 Model

In [3] wordt een model van een kraan gepresenteerd dat op zijn beurt is gebaseerd op onderzoek uit [2]. In het onderzoek van [3] wordt een regelaar ontworpen voor een kraan om het karretje naar een gewenste referentiesnelheid te sturen, met minimale slingering van het gewicht.

De auteurs van [3] hebben het probleem dat er geen begincondities zijn voor de costate. Ze lossen dat probleem op door van twee kanten met een heel scala aan begin- en eind condities van de toestand x en de costate p naar elkaar toe te werken. De middelste toestanden en costates die van twee kanten worden benaderd worden opgeslagen met de bijbehorende begin- en eind condities. Dit is de wiskundige vorm van een *meet in the middle* aanval die kan worden gebruikt om encryptie te kraken.

Hieronder wordt kort het bestaande model toegelicht, waarna de aanpassingen aan het model worden uitgelegd om het model toepasbaar te maken voor onze doeleinden.

4.2.1 Basismodel

In [3] worden niet-lineaire vergelijkingen gegeven voor $\ddot{r}, \ddot{\psi}, \ddot{\alpha}, \ddot{\beta}$ en $\ddot{l}$. Deze variabelen stellen de versnelling van het karretje, de hoek van de kraanarm, de slingering in radiale en tangentiële richting van het gewicht en ten slotte de lengte van de kabel waaraan het gewicht hangt voor. Een visuele representatie van het model is te zien in figuur 4.1. De vergelijking in het ba-

sismodel zijn

$$
\begin{aligned}
\ddot{r} &= r\dot{\psi}^2 + \frac{F_c}{m_t}\sin(\alpha)\cos(\beta) - \frac{F_t}{m_t} \\
\ddot{\psi} &= \frac{1}{I_j + m_t r^2}\left(rF_c\sin(\beta) - 2m_t r\dot{r}\dot{\psi} + n_m T_m\right) \\
\ddot{\alpha} &= \frac{1}{\ell\cos(\beta)}\Big(\left(\ell\ddot{\psi} + 2\ell\dot{\psi}\right)\cos(\alpha)\sin(\beta) - g\sin(\alpha) \\
&\quad - 2\dot{\ell}\dot{\alpha}\cos(\beta) + \left(\ell\dot{\psi}^2\sin(\alpha) + 2\ell\dot{\beta}\dot{\psi}\right)\cos(\alpha)\cos(\beta) \\
&\quad - \left(\ddot{r} - r\dot{\psi}^2\right)\cos(\alpha) + 2\ell\dot{\alpha}\dot{\beta}\sin(\beta)\Big) \\
\ddot{\beta} &= \frac{1}{\ell}\Big(-\left(\ell\ddot{\psi} + 2\ell\dot{\psi}\right)\sin(\alpha) - g\cos(\alpha)\sin(\beta) - 2\dot{\ell}\dot{\beta} \\
&\quad + \left(\ell\dot{\psi}^2\cos(\alpha)\sin(\beta) - 2\ell\dot{\alpha}\dot{\psi}\cos(\beta)\right)\cos(\alpha)\cos(\beta) \\
&\quad - \ell\dot{\alpha}^2\sin(\beta)\cos(\beta) - \left(2\dot{r}\dot{\psi} + r\ddot{\psi}\right)\cos(\beta) \\
&\quad + \left(\ddot{r} - r\dot{\psi}^2\right)\sin(\alpha)\sin(\beta)\Big) \\
\ddot{l} &= \ell\dot{\alpha}^2\cos(\beta)^2 + g\cos(\alpha)\cos(\beta) + 2\ell\dot{\beta}\dot{\psi}\sin(\alpha) \\
&\quad - \left(\ddot{r} - r\dot{\psi}^2\right)\sin(\alpha)\cos(\beta) - \left(2\dot{r}\dot{\psi} + r\ddot{\psi}\right)\sin(\beta) \\
&\quad + \ell\left(\dot{\beta}^2 + \dot{\psi}^2\right) - 2\ell\dot{\alpha}\dot{\psi}\cos(\alpha)\sin(\beta)\cos(\beta) \\
&\quad - \ell\dot{\psi}^2\cos(\alpha)^2\cos(\beta)^2 - \frac{F_c}{m_l}
\end{aligned}
$$

Deze vergelijkingen zijn gebaseerd op een krachtenbalans. Elke vergelijking wordt geschreven in een vorm met een niet-lineaire som van krachten, en een invoerkracht. De invoerkracht wordt opgesplitst in twee losse krachten: een reactiekracht en een gebruikers-invoerkracht. De reactiekracht wordt gelijkgesteld aan het tegengestelde van de niet-lineaire som van krachten, zodat de niet-lineaire component van de vergelijking wegvalt en alleen de gebruikers-invoerkracht overblijft. We krijgen daarmee simpele vergelijkingen voor $\ddot{r}$, $\ddot{\psi}$ en $\ddot{\ell}$, namelijk

$$
\begin{aligned}
\ddot{r} &= -\frac{F_{t,i}}{m_t} \\
\ddot{\psi} &= \frac{n_m T_{m,i}}{I_j + m_t r^2} \\
\ddot{\ell} &= -\frac{F_{c,i}}{m_l},
\end{aligned}
$$

met $F_{t,i}$ de gebruikers-invoerkracht op het karretje en m_t de massa van het karretje (de t staat hier voor *trolley*), $F_{c,i}$ de gebruikers-invoerkracht op de kabel, m_l de massa van de kabel, $T_{m,i}$ de koppel van de motor om de kraan te draaien, I_j het traagheidsmoment van de boom en n_m een versnellingsconstante.

Verder maken de auteurs van [3] de opmerking dat zolang er geen verbuiging van de kraanboom optreedt, de slingeringen α en β onafhankelijk van elkaar beschouwd kunnen worden. Dit is nuttig aangezien wij niet geïnteresseerd zijn in β, ψ en ℓ als veranderende grootheden.

Met de definities $\omega = \sqrt{g/\ell}$ (de natuurlijke frequentie van de slinger), $\ddot{r}_{\max}$ (de maximale versnelling van het karretje) en $k = \ddot{r}_{\max}/\ell$ krijgen we de vergelijking voor de slinger in radiale richting

$$\ddot{\alpha} = -\omega^2\sin(\alpha) - \ddot{r}\cos(\alpha)k/\ddot{r}_{\max}.$$

De constante k is hier ingevoerd om de soortgelijke vergelijking voor $\ddot{\beta}$ in een gelijke vorm te krijgen als voor $\ddot{\alpha}$. Merk op dat deze vergelijking ook gemakkelijk kan worden afgeleid uit het normale model door, volgens onze aannames, $\psi(t) = 0$, $\beta(t) = 0$ en $\ell(t) = 0$ te stellen.

Onder andere deze vergelijking wordt omgeschreven in toestand-vorm met toestanden x_1, x_2 en x_3 en invoer u, zodat we

$$x_1 = \alpha, \quad x_2 = \dot{\alpha}, \quad x_3 = \dot{r}, \quad u = \ddot{r}/\ddot{r}_{\max}$$

krijgen, met de systeemvergelijkingen

$$
\begin{cases}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= -\omega^2\sin(x_1) - k\cos(x_1)\,u \\
\dot{x}_3 &= k\ell\,u
\end{cases}
$$

met $|u| \leq 1$.

De correctheid van dit model is door de auteurs van [3] succesvol geverifieerd. Zij zien een goede overeenstemming tussen een fysieke kraan en de verwachte theoretische uitkomsten.

4.2.2 Aanpassingen aan model

Met [3] als basis is het model aangepast om toepasselijker te zijn voor onze doeleinden. Wij zullen alleen naar de vergelijkingen voor r en α kijken.

In het bestaande systeem is alleen een toestand aanwezig voor de snelheid van het karretje, $\dot{r}$. Dit is onvoldoende aangezien wij ook de positie willen sturen. Er wordt daarom een extra systeemtoestand $x_4 = r$ toegevoegd met $\dot{x}_4 = x_3$.

Ook wordt een aantal variabelen en constanten aangepast om het systeem beter te laten aansluiten op dat van hoofdstuk 3. We noemen α, de uitwijking van de slinger, opnieuw θ. Ook wordt de vergelijking van de slinger aangepast zodat de gravitatieconstante g en de lengte van de slinger l erin voorkomen. Dat maakt het model algemener en breder toepasbaar. Ook wordt de constante k uit het model gehaald omdat dit de vergelijkingen ingewikkelder maakt en niet het voordeel biedt dat de auteurs van [3] hiervan ondervonden.

4.2.3 Resulterend model

We definiëren de toestand van het systeem als

$$x = (x_1, x_2, x_3, x_4)^T = \left(\theta, \dot{\theta}, r, \dot{r}\right)^T.$$

Het systeem vertoont de niet-lineaire dynamiek $\dot{x} = f(x,u)$, gelijk aan:

$$\begin{cases} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -\frac{g}{\ell}\sin(x_1) - \frac{\ddot{r}_{\max}}{\ell}\cos(x_1)u \\ \dot{x}_3 &= x_4 \\ \dot{x}_4 &= \ddot{r}_{\max}u \end{cases}$$

Het probleem is niet lineair omdat er hoeken in voorkomen, wat een sinus en cosinus in de krachten oplevert. We lineariseren $f(x,u)$ om een werkbaarder probleem te krijgen. Dezelfde argumenten als genoemd in vorig hoofdstuk zijn toepasbaar om aan te tonen dat dit geen problemen oplevert op het gebied van numerieke fouten.

In de toepassing van een hijskraan wordt deze aanname nog aannemelijker: een te grote uitwijking kan niet voorkomen door de grote gewichten aan de kraan de kracht die daardoor nodig is om het karretje zodanig te versnellen dat de uitwijking extreem wordt. Wanneer men over zulke acties met een hijskraan praat komen er andere (externe) effecten in het spel die hier niet behandeld worden.

Het nieuwe (gelineariseerde) systeem is

$$\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{pmatrix} = \left(\begin{array}{cc|cc} 0 & 1 & 0 & 0 \\ -\frac{g}{\ell} & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{array}\right) \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} + \begin{pmatrix} 0 \\ -\frac{\ddot{r}_{\max}}{\ell} \\ 0 \\ \ddot{r}_{\max} \end{pmatrix} u,$$

oftewel $\dot{x} = Ax + Bu$. Merk op dat de systemen onafhankelijk zijn: de matrix A is op te delen in vier submatrices door

$$A = \begin{pmatrix} A_1 & 0 \\ 0 & A_2 \end{pmatrix},$$

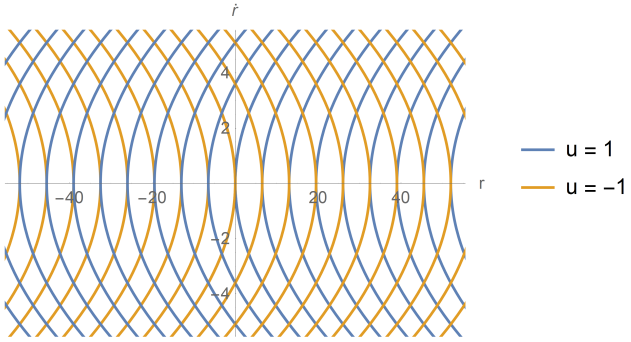
Deze eigenschap wordt gebruikt om het grote systeem in twee losse kleine systemen op te splitsen, beide op te lossen en weer samen te voegen tot een grote oplossing, waarbij rekening wordt gehouden met de voorwaarden van het grote dynamische systeem. Hierbij wordt rekening gehouden dat de invoer u wel hetzelfde is voor beide subsystemen.

Het systeem heeft hiermee een ander evenwicht gekregen vergeleken met de slinger, namelijk $\left(-\ddot{r}_{\max}u/g, 0, 0, 0\right)$.

Net als het probleem van de slinger zullen we een tijd-optimale oplossing zoeken en daarna de kostenfunctie aanpassen om een niet-tijd-optimale oplossing te zoeken. We zullen steeds het probleem beschouwen met de beginvoorwaarde $(0,0,r_0,0)$ met $r_0 < 0$ en de eindtoestand $(0,0,0,0)$ waarbij het systeem in rest is.

4.3 Oplossing 1: Bang-bang regelaar

Voor de eerste oplosmethode zoeken we een tijd-optimale oplossing. Met behulp van de theorie van optimale regelaars worden eigenschappen van een optimale (*bang-bang*) regelaar bepaald. Daarna wordt voor een vast aantal switchtijden, namelijk één, twee, drie en vijf, een oplossing geconstrueerd. Hieruit volgt een algemene oplossing voor de optimale regelaar voor het kraanprobleem.



FIGUUR 4.2: Het $(r,\dot{r})$-vlak waarin het karretje zich beweegt over de blauwe parabolen voor $u = 1$ en de oranje parabolen voor $u = -1$.

4.3.1 Dynamiek van $(r,\dot{r})$

In het voorgaande hoofdstuk is de dynamiek van de slinger uitgebreid bestudeerd. Ditzelfde gaan we doen voor het karretje, gekarakteriseerd door r als zijn positie en $\dot{r}$ als zijn snelheid.

De vergelijkingen waar r aan voldoet is

$$\ddot{r} = \ddot{r}_{\max}u,$$

oftewel $r(t) = \frac{1}{2}\ddot{r}_{\max}ut^2 + at + b$, een parabool waarbij $\dot{r}(0) = a$ en $r(0) = b$.

We leiden ook het verband af tussen r en $\dot{r}$. Dit wordt namelijk gebruikt om te kunnen redeneren over de baan van $(r,\dot{r})$ voor een vaste u. Er geldt

$$\dot{x}_3 = x_4$$
$$\dot{x}_4 = \ddot{r}_{\max}\hat{u}.$$

Door $\dot{x}_4$ door $\dot{x}_3$ te delen krijgen we

$$\frac{\mathrm{d}x_4}{\mathrm{d}x_3} = \frac{\ddot{r}_{\max}\hat{u}}{x_4}$$

met een oplossing

$$x_4^2 = 2\ddot{r}_{\max}\hat{u} + C$$

voor een vaste $\hat{u}$ en een vrije constante C. Dit zijn ook parabolen in het $(r,\dot{r})$-vlak. In figuur 4.2 zijn deze parabolen te zien.

4.3.2 Optimale regelaar

Met de kostenfunctie

$$J(x,u) = \int_0^{T_*} 1\,\mathrm{d}t$$

beredeneren we een optimale invoer. Dit wordt gegeven door de Hamiltoniaan

$$\begin{aligned} H(x,p,u) &= p^T f + L \\ &= \left(x_2 p_1 + \left(-\frac{g}{\ell} - \frac{\ddot{r}_{\max}}{\ell} u\right) p_2 + x_4 p_3 + \ddot{r}_{\max} u p_4\right) + 1 \end{aligned}$$

op elk punt te minimaliseren volgens stelling 2.4.1:

$$\begin{aligned} u(t) &= \operatorname*{argmin}_v H(x,p,v) \\ &= \operatorname*{argmin}_v \left(\ddot{r}_{\max} p_4 - \frac{\ddot{r}_{\max}}{\ell} p_2\right) v \end{aligned}$$

Hiermee kunnen conclusies worden getrokken voor $u(t)$. Als

$$\ddot{r}_{\max} p_4 > \frac{\ddot{r}_{\max}}{\ell} p_2$$

dan moet u zo negatief mogelijk, dus $u = -1$. Andersom: als

$$\ddot{r}_{\max} p_4 < \frac{\ddot{r}_{\max}}{\ell} p_2$$

dan moet $u = 1$. Er tussenin geldt $u = 0$. Kortom:

$$u(t) = -\operatorname{sgn}\left(\ddot{r}_{\max} p_4 - \frac{\ddot{r}_{\max}}{\ell} p_2\right).$$

Deze functie moeten we even laten rusten voor we hier iets nuttigs over kunnen zeggen. Daarvoor moeten we eerst de dynamiek bekijken van $p(t)$, de costates.

Voor de costates geldt het systeem

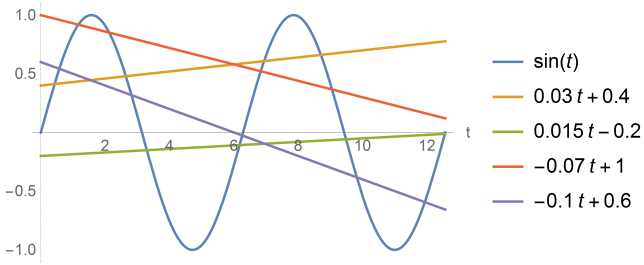
$$\dot{p} = -A^T p = \left(\begin{array}{cc|cc} 0 & \frac{g}{\ell} & 0 & 0 \\ -1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \end{array}\right) p.$$

Omdat de matrix is op te delen kan apart voor (p_1, p_2) en (p_3, p_4) worden opgelost. We kijken eerst naar p_4, één van de variabelen in onze optimale invoer. Er geldt $\dot{p}_4 = -p_3$ en $\dot{p}_3 = 0$, dus $p_3 = C_1$ en $p_4 = -C_1 t + C_2$ voor constanten C_1 en C_2. Voor p_2 geldt een gelijke redenatie. Er moet gelden dat

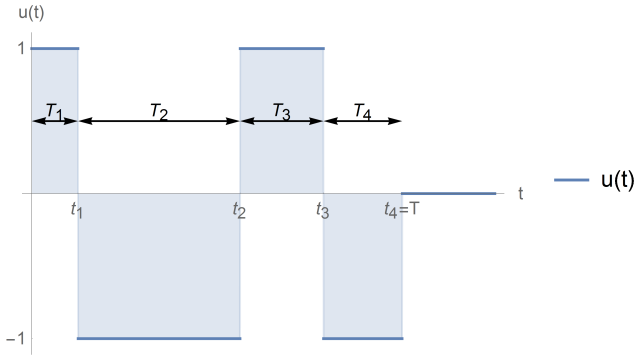
$$\ddot{p}_2 = -\frac{g}{\ell} p_2$$

dus $p_2 = C_3 \sin\left(\sqrt{\frac{g}{\ell}} t + C_4\right)$ met $0 \le C_4 < 2\pi$.

We hebben de snijpunten van p_4 en p_2 nodig om de switchtijden te achterhalen. Een voorbeeld van $p_2 = \sin(t)$ en $p_4 = at + b$ met a en b vrij staat in figuur 4.3. Hier zijn twee van de vier vrijheidsgraden van p_2 en p_4 afgebeeld. Het is duidelijk dat hieruit niet veel concrete informatie over de daadwerkelijke snijpunten van de functies kan worden afgeleid om een expliciete oplossing voor $u(t)$ te vinden.



FIGUUR 4.3: Visualisatie van de snijpunten van $\sin(t)$ met de $at + b$ met a en b vrij te kiezen.



FIGUUR 4.4: De switchtijden t_k en doorlooptijden T_k.

4.3.3 Het bepalen van de optimale invoer

Helaas kunnen we aan de hand van bovenstaande beschrijving voor $u(t)$ geen expliciete oplossing afleiden omdat p_2 en p_4 te onbekend zijn. Ook de eigenschap dat $H(x,p,u) = 0$ op $t = T_*$ geeft ons geen hulpmiddelen. We redeneren aan de hand van de eigenschappen van de twee losse systemen (de slinger en het karretje) hoe $u(t)$ eruit moet zien om geldig en optimaal te zijn.

We gaan bij deze oplosmethode veel gebruikmaken van switchtijden en doorlooptijden. Er geldt dat de switchtijd t_k de tijd is wanneer voor de $k$de keer wordt gewisseld van invoer. De doorlooptijd T_k is de tijd die tussen twee switchtijden wordt doorgebracht met een bepaalde u, dus

$$T_k = t_k - t_{k-1}.$$

De totale tijd is

$$T = \sum_k T_k.$$

Een afbeelding hiervan staat in figuur 4.4.

Verder noemen we de punten waarop wordt geswitcht r_i, met i de index van de switch, oftewel

$$r_i = r(t_i).$$

Zo is bijvoorbeeld r_2 de locatie van het karretje op $t = t_2$.

Tijdens het oplossen van het probleem met een verschillend aantal switchtijden zal symmetrie vaak naar voren gebracht als argument. Dit heeft het vermoeden doen opkomen dat voor een geldige oplossing geldt dat

$p_4(T/2) = 0$ en ook $p_2(T/2) = 0$, wat direct een snijpunt van p_4 en p_2 op $t = T/2$ impliceert en een aantal vrijheidsgraden van de costates wegneemt. Het is echter niet gelukt om dit vermoeden te bewijzen.

4.3.4 Eén switchtijd

Om de redenering logisch te maken, beginnen we met een specifiek geval, namelijk

$$r_0 = -\ddot{r}_{\max}(2\pi)^2 \frac{\ell}{g}.$$

De redenatie achter deze waarde is als volgt:

- De slinger bevindt zich in $(0,0)$, het karretje in $(r_0,0)$, met r_0 een vrij te kiezen getal.
- Wanneer de slinger een volledige periode van een ellips aflegt met $u = 1$, is de slinger terug in $(0,0)$. Het karretje is nu echter in $(r_1,\dot{r}_1)$.
- Wanneer de slinger een volledige periode van een ellips aflegt met $u = -1$, is de slinger terug in $(0,0)$. Het karretje is nu echter in $(r_2,0)$.
- Wanneer we r_0 zo kiezen dat $r_2 = 0$ (precies de waarde van r_0 die hierboven staat genoemd), dan eindigt het karretje in $(0,0)$, precies het doel van de optimalisatie.

De gevonden $u(t)$ is in dit specifieke geval dus

$$u(t) = \begin{cases} 1 & 0 \le t < P \\ -1 & P \le t < 2P \\ 0 & 2P \le t \end{cases},$$

met P de periode van één ellips, de tijd $2\pi\sqrt{\frac{\ell}{g}}$. Een afbeelding van deze oplossing staat in figuur 4.5. We gaan nu deze manier van oplossen voor $u(t)$ uitbreiden, om voor elke r_0 een bijbehorende optimale $u(t)$ te vinden.

Hiervoor maken we de observatie dat de tijd die het systeem met $u = 1$ doorbrengt gelijk moet zijn aan de tijd die het systeem in $u = -1$ doorbrengt. Immers, we willen dat $\dot{r}(0) = 0$ en $\dot{r}(T) = 0$ met $\ddot{r} = \ddot{r}_{\max} u$, dus

$$\begin{aligned} \dot{r}(T) - \dot{r}(0) &= \int_0^T \ddot{r}\,\mathrm{d}t \\ &= \sum_{k,u=1} (1)\,T_k \ddot{r}_{\max}\mathrm{d}t + \sum_{k,u=-1} (-1)\,T_k \ddot{r}_{\max}\mathrm{d}t \\ &= 0 \end{aligned}$$

dan en slechts dan als

$$\sum_{k,u=1} T_k = \sum_{k,u=-1} T_k.$$

Dit betekent dat er voor de slinger n_{+1} en n_{-1} ($\in \mathbb{N}$) volledige ellipsen kunnen worden doorgebracht met respectievelijk $u = 1$ en $u = -1$. Wanneer $n_{+1} = n_{-1}$ is $\dot{r}(T) = \dot{r}(0) = 0$.

We nemen $n_{+1} = n_{-1} = n$ in dit voorbeeld. Er worden eerst n ellipsen met $u = 1$ doorlopen en daarna n ellipsen voor $u = -1$. We weten dat $r(0) = r_0$ en $\dot{r}(0) = 0$. Hiermee kunnen we volgens de vergelijkingen van $r(t)$ en $\dot{r}(t)$ afleiden dat voor een $t_1 > 0$, na T_1 tijd met $u = 1$ geldt

$$\begin{aligned} r(t_1) &= r_0 + \frac{1}{2}\ddot{r}_{\max} T_1^2 \\ \dot{r}(t_1) &= 0 + \ddot{r}_{\max} T_1. \end{aligned}$$

Op dezelfde manier kunnen we de waarden op $t = t_2$ bepalen, maar dan met $u = -1$ voor T_2 tijd. We vinden

$$r(t_2) = r_0 + \frac{1}{2}\ddot{r}_{\max} T_1^2 - \frac{1}{2}\ddot{r}_{\max} T_2^2 + \ddot{r}_{\max} T_1 T_2 \qquad (4.1)$$

$$\dot{r}(t_2) = 0 + \ddot{r}_{\max} T_1 - \ddot{r}_{\max} T_2. \qquad (4.2)$$

Omdat $T_1 = T_2 = 2n\pi\sqrt{\frac{\ell}{g}}$ en $t_2 = T$ houden we over dat $\dot{r}(T) = 0$ en

$$r(T) = r_0 + \ddot{r}_{\max}(2n\pi)^2 \frac{\ell}{g}$$

Het resultaat hiervan is dat we een r_0 kunnen kiezen die gelijk is aan

$$-\ddot{r}_{\max}(2n\pi)^2 \frac{\ell}{g},$$

voor een $n \in \mathbb{N}$ zodat $r(T) = 0$. In figuur 4.6 staat hiervan een voorbeeld met $n = 3$.

Hoewel dit resultaat een goed begin is, zijn er maar slechts een aantal waarden voor r_0 waarvoor een oplossing bestaat voor $u(t)$. We willen voor elke r_0 een oplossing, dus breiden we het aantal switchtijden uit naar meer dan 1.

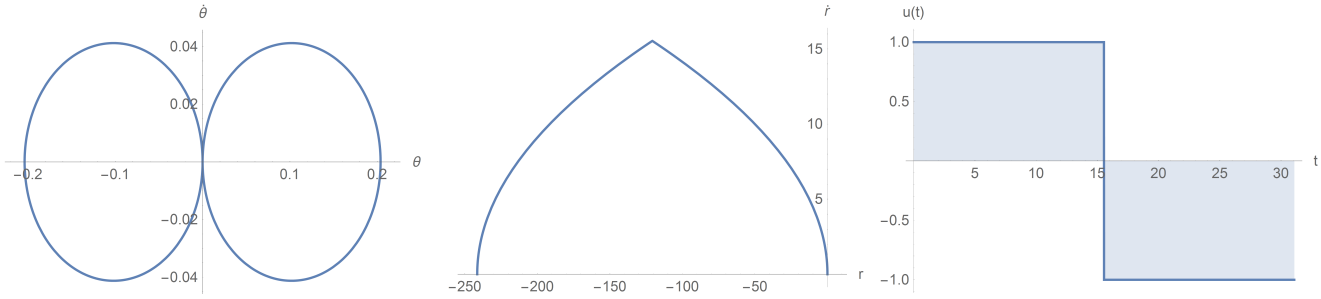
4.3.5 Uitbreiding naar 2 switchtijden

Wanneer we twee switchtijden om een oplossing te vinden komen we in een onmogelijke situatie.

Stel er wordt T_1 tijd met $u = 1$ wordt doorgebracht. Daarna brengen we T_2 tijd met $u = -1$ door, waarna er $T_3 = T_2 - T_1$ tijd met $u = 1$ wordt doorgebracht omdat de totale tijd met $u = 1$ gelijk moet zijn aan de totale tijd met $u = -1$. We hebben dus twee vrije doorlooptijden.

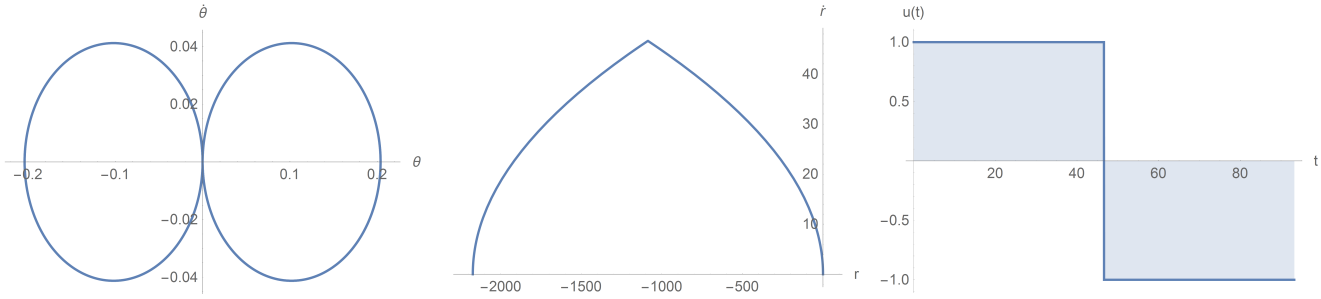
Echter is dat schijn. De waarde van T_2 moet precies zo gekozen worden dat $T_2 \ge T_1$. Ook moet het punt $(\theta_2,\dot{\theta}_2)$ precies op de ellips met middelpunt $(\ddot{r}_{\max}/g, 0)$ en straal $\ddot{r}_{\max}/g$ liggen omdat na $T_2 - T_1$ tijd met $u = 1$ de slinger weer in $(0,0)$ moet zijn.

De tijd T_2 ligt dus vast: het is de tijd tussen het punt $(\theta_1,\dot{\theta}_1)$ en het volgende snijpunt met de ellips: $(\theta_1,-\dot{\theta}_1)$. Met deze informatie kijken we welke T_1 zorgt voor $(\theta(T),\dot{\theta}(T)) = (0,0)$. De waarde van $(\theta(T),\dot{\theta}(T))$ voor $0 \le T_1 \le P/2$ is te zien in figuur 4.7. We vinden dat $T_1 = 0$ of $T_1 = P/2$.



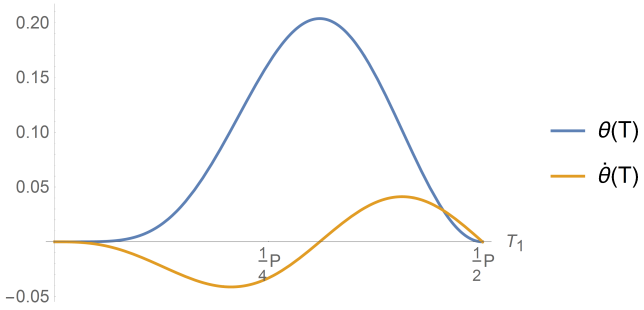
(a) De beweging door het $(\theta,\dot{\theta})$-vlak (b) De beweging door het $(r,\dot{r})$-vlak (c) De invoer u

FIGUUR 4.5: Een karretje met slinger met $r_0 = -\ddot{r}_{\max}(2n\pi)^2\frac{\ell}{g}$ opgelost met één switchtijd en $n = 1$.



(a) De beweging door het $(\theta,\dot{\theta})$-vlak (b) De beweging door het $(r,\dot{r})$-vlak (c) De invoer u

FIGUUR 4.6: Een karretje met slinger met $r_0 = -\ddot{r}_{\max}(2n\pi)^2\frac{\ell}{g}$ opgelost met één switchtijd en $n = 3$.



FIGUUR 4.7: De waarde van $\theta(T)$ en $\dot{\theta}(T)$ voor een gegeven T_1 bij twee switchtijden.

Beide waarden voor T_1 geven geen oplossing. Voor $T_1 = 0$ beweegt het karretje niet, en voor $T_1 = P/2$ (en bijbehorende $T_2 = P$) rijdt het karretje heen en weer om weer in hetzelfde punt uit op de boom te komen. Immers, $\dot{r}$ is precies even lang positief als negatief.

Voor $T_1 > P/2$ zijn dezelfde soort argumenten geldig. Er zijn geen r_0 anders dan 0 waarvoor een oplossing kan worden gevonden met twee switchtijden. We breiden de theorie dus verder uit naar drie switchtijden.

4.3.6 Uitbreiding naar 3 switchtijden

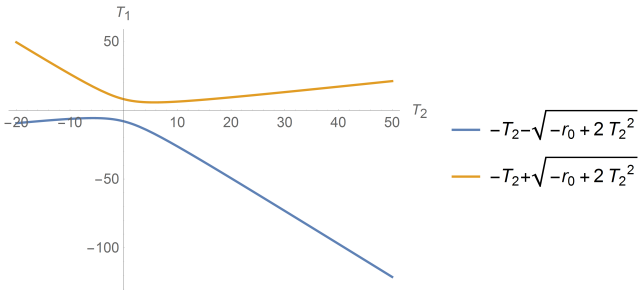
We nemen een weer tijdstip $t_1 > 0$ voor de eerste switchtijd. Op dit tijdstip switchen we van $u = 1$ naar $u = -1$. Net zo kiezen we t_2 voor de volgende switchtijd, en t_3 voor de laatste switchtijd.

Wanneer er een switch op tijdstip t_1 plaatsvindt, moet er op t_2 een volgend switchpunt plaatsvinden. Dit is precies op $T/2$ aangezien daarna nogmaals T_2 tijd wordt doorgebracht met $u = 1$ en ten slotte nogmaals T_1 tijd wordt doorgebracht met $u = -1$, wat het totaal correct maakt.

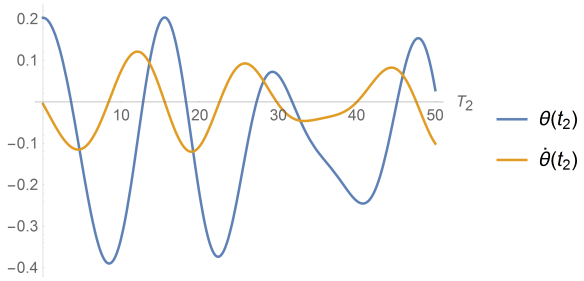
Omdat $\left(\theta(T),\dot{\theta}(T)\right) = (0,0)$ moet gelden kunnen we nog een voorwaarde voor θ_2 vinden. Deze moet namelijk precies samenvallen met de $\dot{\theta}$-as, oftewel $\theta_2 = 0$. De reden dat deze voorwaarde ontstaat is symmetrie: op het moment dat de baan niet gespiegeld kan worden in de $\dot{\theta}$-as, geven dezelfde doorlooptijden andere banen, en is komt de baan op $t = T$ niet in $(0,0)$ uit (op een aantal triviale oplossingen na die geen nuttige bijdrage leveren aan een oplossing voor een willekeurige r_0).

Hierbij moet een opmerking worden geplaatst. Wanneer t_1 zo gekozen wordt dat $T_1 < T_2$ zal de snelheid van het karretje voor een korte tijd negatief worden. Hierover wordt een korte discussie gevoerd in sectie 4.3.8.

Het resultaat is dat we voor alle t_1 nu een oplossing kunnen vinden met een bijbehorende r_0. Om een expliciete T_1 en T_2 voor een r_0 te bepalen kijken we eerst



FIGUUR 4.8: De oplossingen voor T_1 als functie van T_2 voor drie switchpunten. Hier is $r_0 = -80$ gebruikt.



FIGUUR 4.9: De waarde van $\theta(t_2)$ en $\dot{\theta}(t_2)$ voor een gegeven T_2 bij drie switchtijden. Hier is $r_0 = -80$ gebruikt.

naar het karretje. Volgens (4.1) geldt dat

$$r(t_2) = r_0 + \frac{1}{2}\ddot{r}_{\max} T_1^2 - \frac{1}{2}\ddot{r}_{\max} T_2^2 + \ddot{r}_{\max} T_1 T_2,$$

deze keer met vrije T_1 en T_2. De enige voorwaarde is dat $r(t_2) = r_0/2$, omdat erna $T_3 = T_2$ en $T_4 = T_1$ zodat de totalen kloppend worden. Hieruit kunnen we afleiden dat

$$T_1 = -T_2 + \sqrt{-r_0 + 2T_2^2}.$$

De oplettende lezer ziet dat er nog een oplossing is voor T_1, namelijk met een min-teken voor de wortel. Echter geeft deze altijd negatieve T_1 dus zal niet verder worden gebruikt. In figuur 4.8 zijn beide oplossingen voor T_1 als functie van T_2 te zien.

Voor een gegeven T_2 (en bijbehorende T_1) kan gemakkelijk het punt $\left(\theta(t_2), \dot{\theta}(t_2)\right)$ worden bepaald. Voor een bereik aan T_2 is de functie voor θ te zien in figuur 4.9. Wij willen de T_2 vinden, waarvoor $\theta(t_2)$ voor het eerst gelijk aan 0 is. Omdat de functie van T_2 geen analytische oplossingen heeft, gebeurt dit numeriek.

Merk op dat wanneer $T_1 > P$ er eerst minimaal één volledige ellips met $u = 1$ wordt doorlopen voordat er voor het eerst geswitcht wordt. Er wordt op de 'terugweg' hetzelfde aantal ellipsen met $u = -1$ doorlopen (en een minimaal eens $(0,0)$ gepasseerd), alvorens definitief op tijdstip T in $(0,0)$ uit te komen.

Voorbeelden van de oplossing met drie switchpunten zijn gesimuleerd voor de waarden $r_0 = -65$ (figuur 4.10) en $r_0 = -10$ (figuur 4.11). We zien dat het karretje bij de oplossing voor $r_0 = -10$ een korte tijd $\dot{r} < 0$ heeft.

4.3.7 Uitbreiding naar 5 switchtijden

Na oplossingen voor drie switchtijden kijken we naar een mogelijke verbetering door nog meer switchtijden toe te voegen.

Hier wordt alleen gekeken naar een situatie met vijf switchtijden, want net als bij twee switchtijden zou vier switchtijden een asymmetrische oplossing geven wat altijd een niet geldige, triviale of niet-optimale oplossing oplevert.

We passen dezelfde strategie toe als bij één en drie switchtijden. Twee switchpunten vinden plaats met $\theta > 0$, en twee met $\theta < 0$. Stel dat er van invoer gewisseld wordt van $u = 1$ naar $u = -1$ op $t_1 = T_1$. Daarna vindt nog een switch plaats, waarna de $\dot{\theta}$-as wordt bereikt met $\theta(t_3) = 0$. Daarmee vinden we dat op zijn vroegst de tweede keer gewisseld kan worden als we op dezelfde ellips als tussen 0 en t_1 zijn aanbeland. Een vroegere switch geeft een straal kleiner dan $\ddot{r}_{\max}/g$, waardoor de ellips nooit de $\dot{\theta}$-as kruist.

Ook kunnen we beredeneren dat t_2 minimaal moet zijn, dus wanneer de ellips wordt gesneden met straal $\ddot{r}_{\max}/g$. We willen namelijk dat de tijd dat $u = -1$, wanneer het karretje wordt afgeremd, wordt geminimaliseerd.

Hiermee kunnen we alle switchtijden vinden. Eerst wordt er T_1 tijd met $u = 1$ doorgebracht, dan T_2 tijd met $u = -1$ totdat de eerste ellips weer wordt gesneden. Dan weer T_1 tijd met $u = 1$ tot het punt $(0,0)$. Daarna worden dezelfde doorlooptijden voor T_4, T_5 en T_6 gebruikt met omgekeerde u.

Net als bij 3 switchtijden kunnen we een verband afleiden voor T_1 en T_2, wanneer wordt gekeken naar de afstand die het karretje tussen 0 en t_3 aflegt. De positie van het karretje is op $t = t_3$ gelijk aan

$$r(r_3) = r0 + 2T_1^2 - \frac{1}{2}T_2^2 = \frac{1}{2}r_0.$$

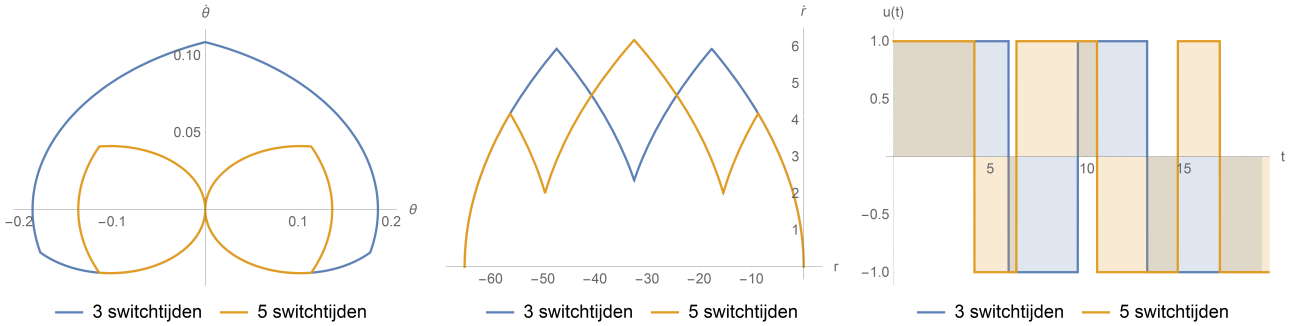
Hiermee kunnen we afleiden dat

$$T_1 = \frac{1}{2}\sqrt{-r_0 + T_2^2},$$

waarbij we de oplossing met het minteken negeren omdat dit een negatieve T_1 geeft.

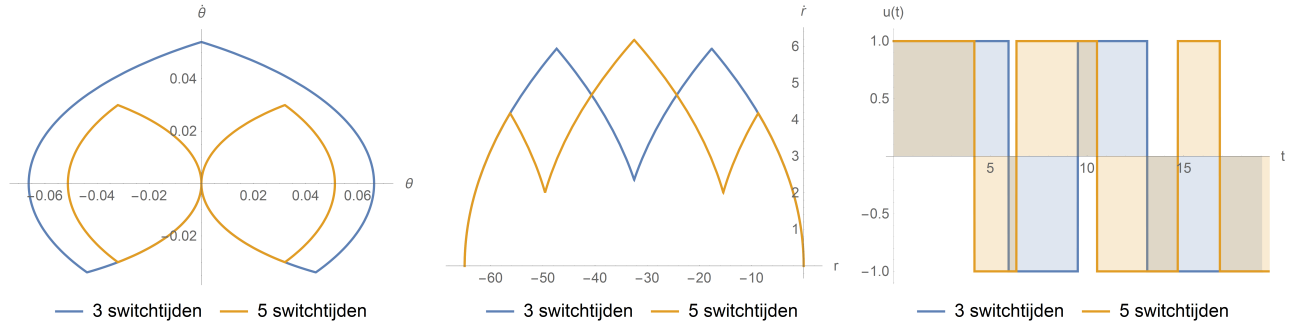
Nu moet $\left(\theta, \dot{\theta}\right)$ precies in $(0,0)$ uitkomen op $t = t_3$. Dat geeft ons weer een vergelijking die alleen numeriek opgelost kan worden. Een afbeelding van $\theta(t_3)$ en $\dot{\theta}(t_3)$ als functie van T_2 staat in figuur 4.12.

Met deze oplossingen zijn simulaties uitgevoerd voor $r_0 = -65$ (figuur 4.10) en $r_0 = -10$ (figuur 4.11).



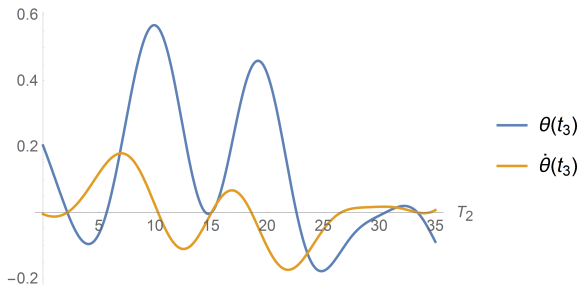
(a) De beweging door het $(\theta, \dot{\theta})$-vlak (b) De beweging door het $(r, \dot{r})$-vlak (c) De invoer u

FIGUUR 4.10: Een karretje met slinger met $r_0 = -65$ opgelost met drie en vijf switchtijden.



(a) De beweging door het $(\theta, \dot{\theta})$-vlak (b) De beweging door het $(r, \dot{r})$-vlak (c) De invoer u

FIGUUR 4.11: Een karretje met slinger met $r_0 = -10$ opgelost met drie en vijf switchtijden.



FIGUUR 4.12: De waarde van $\theta(t_3)$ en $\dot{\theta}(t_3)$ voor een gegeven T_2 bij 5 switchtijden. Hier is $r_0 = -65$ gebruikt.

4.3.8 Discussie en optimaliteit

De parameters die hier zijn gebruikt zijn $g = 9.81$, $\ell = 60$ en $\ddot{r}_{\max} = 1.0$.

Voor een bereik aan waarden van r_0 tussen -100 en -10 zijn de optimale oplossingen vergeleken met drie en vijf switchtijden. De resultaten staan in figuur 4.13. Hier zijn de punten waar één switchpunt een oplossing geeft niet meegenomen, aangezien deze pas beginnen bij $r_0 \approx -250$ en niet relevant zijn voor deze discussie.

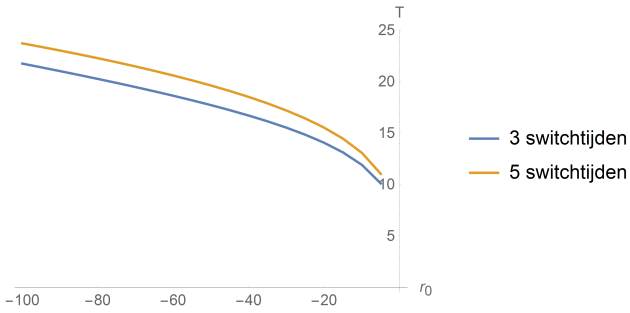
We kunnen hieruit concluderen dat drie switchtijden altijd een snellere en dus betere oplossing geeft. Dit is opmerkelijk, aangezien de tijd die wordt doorgebracht met $\dot{r} < 0$ waarbij het karretje achteruit rijdt niet altijd kleiner is bij de oplossing met vijf switchtijden.

Ook is in figuur 4.13 mooi het kwadratische verband te zien tussen de afstand die het karretje aflegt en de eindtijd T van de oplossing. De afstand r is een som van parabolische functies dus zal altijd een kwadratische functie van t (en dus ook T) zijn.

Er moet een opmerking worden gemaakt over en gevonden oplossing door deze methode. Voor een beginconditie $(0,0,r_0,0)^T$ kan voor elke r_0 een $u(t)$ worden gevonden zodanig dat het systeem tijd-optimaal naar $(0,0,0,0)^T$ zal gaan. Echter kan er geen correctie worden uitgevoerd tijdens de loop van het systeem want de toestand zal over het algemeen niet voldoen aan de vorm $(0,0,r,0)^T$. In vergelijking met de oplossing voor de slinger, die voor elke beginvoorwaarde een optimale regelaar kan geven, is deze oplossing dus gevoeliger voor fouten die tijdens het verloop van het systeem optreden.

4.4 Oplossing 2: Kosten op invoer

We voegen voor deze oplosmethode weer kosten toe op de eindtoestand en de invoer u, met de factor $\varepsilon > 0$ en $s > 0$. Dit geeft ons dezelfde kostenfunctie als vorig



FIGUUR 4.13: Een vergelijking tussen de eindtijd met drie of vijf switchpunten voor verschillende waarden voor r_0.

hoofdstuk,

$$J = sx(T)^T x(T) + \int_0^T 1 + \varepsilon u(t)^2 \mathrm{d}t$$

en de Hamiltoniaan

$$H(x,p,u) = p^T f + L = \left(x_2 p_1 + \left(-\frac{g}{\ell} - \frac{\ddot{r}_{\max}}{\ell} u\right) p_2 + x_4 p_3 + \ddot{r}_{\max} u p_4\right) + \left(1 + \varepsilon u^2\right).$$

Dit garandeert ons een 'gladde' $u(t)$ die in ieder geval Lipschitz continu is. De optimale regelaar wordt volgens stelling 2.3.1 gegeven door

$$\begin{aligned} u(t) &= \operatorname*{argmin}_v H(x,p,v) \\ &= \operatorname*{argmin}_v \left(\ddot{r}_{\max} p_4 - \frac{\ddot{r}_{\max}}{\ell} p_2\right) v + \varepsilon v^2 \\ &= \frac{1}{2\varepsilon}\left(\frac{\ddot{r}_{\max}}{\ell} p_2 - \ddot{r}_{\max} p_4\right) \end{aligned}$$

Voor een gelineariseerd systeem geldt dan

$$u(t) = \left(0, \frac{\ddot{r}_{\max}}{2\ell\varepsilon}, 0, -\frac{\ddot{r}_{\max}}{2\varepsilon}\right)\left(p_1, p_2, p_3, p_4\right)^T,$$

oftewel $u = -R^{-1}B^T p$ met $R = 2\varepsilon$. Hiermee kan een simulatie kan worden uitgevoerd.

We maken hiervoor weer een gecombineerd systeem

$$\left(\dot{x}_1, \dot{x}_2, \dot{x}_3, \dot{x}_4, \dot{p}_1, \dot{p}_2, \dot{p}_3, \dot{p}_4\right)^T = H \cdot \left(x_1, x_2, x_3, x_4, p_1, p_2, p_3, p_4\right)^T$$

met

$$H = \begin{pmatrix} A & -BR^{-1}B^T \\ Q & -A^T \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\frac{g}{\ell} & 0 & 0 & 0 & 0 & -\frac{\ddot{r}_{\max}^2}{2\ell^2\varepsilon} & 0 & \frac{\ddot{r}_{\max}^2}{2\ell\varepsilon} \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{\ddot{r}_{\max}^2}{2\ell\varepsilon} & 0 & -\frac{\ddot{r}_{\max}^2}{2\varepsilon} \\ 0 & 0 & 0 & 0 & 0 & \frac{g}{\ell} & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \end{pmatrix}$$

en begin- en eind voorwaarden $x(0) = x_0$ en $p(T) = sx(T)$, wat $2n$ voorwaarden oplevert voor $2n$ variabelen wat het systeem goed gedefinieerd maakt.

Dit (grote) dynamische systeem kan gesimuleerd worden voor een gegeven T. Daarna wordt met behulp van een binary search de kleinste T bepaald waarvoor $|u(t)| \leq 1$ voor alle t, net als in sectie 3.5.3.

4.4.1 Voorbeeld

Een voorbeeld van een simulatie met het verkregen dynamische systeem staat in figuur 4.14. De gebruikte parameters zijn $g = 9.81$, $\ell = 60$, $\ddot{r}_{\max} = 1.0$, $s = 10^6$ en $\varepsilon = 1$.

4.5 Vergelijking tussen methodes

Hier zullen beide oplosmethodes voor het probleem van de kraan worden vergeleken. Voor de oplosmethode met de bang-bang regelaar zal de oplossing met drie switchtijden worden gebruikt.

In figuur 4.15 zijn een aantal beginvoorwaarden te zien waarvoor de methodes zijn vergeleken. Verder zijn de banen in het $(\theta, \dot{\theta})$-vlak en in het $(r, \dot{r})$-vlak weergegeven.

Hoewel de eerste oplosmethode die gebruik maakt van de bang-bang regelaar natuurlijk sneller is omdat deze tijd-optimaal wordt geconstrueerd, convergeren de eindtijden T van beide oplossingen naar elkaar toe voor grote grote $|x_0|$ en verschillen de prestaties niet veel van elkaar. Ook is de tweede methode toepasbaar voor alle beginvoorwaarden.

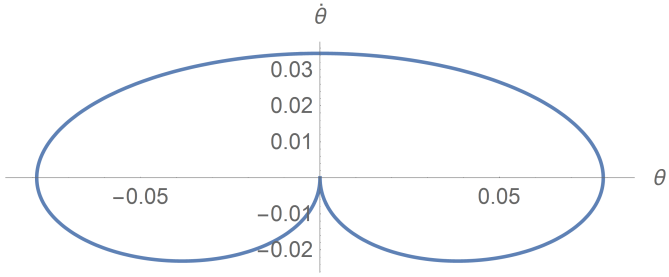
4.6 Conclusie

We hebben in dit hoofdstuk gezien hoe op twee manieren een optimale regelaar kan worden gevonden voor het probleem van de slinger.

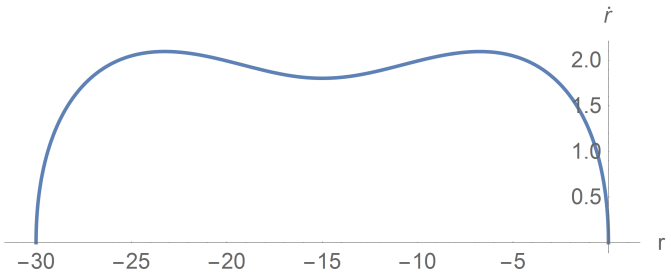
De eerste oplosmethode gebruikt een redenatie over de tijd-optimale regelaar met een vast aantal switchtijden van de invoer. Door deze redenaties kan worden afgeleid dat drie switchtijden in alle situaties de optimale regelaar geeft.

De tweede oplosmethode is een directe uitbreiding van de derde oplosmethode van de slinger, en kan worden toegepast om met kosten op de invoerkracht het karretje en de slinger naar de toestand $(0,0,0,0)^T$ te sturen.

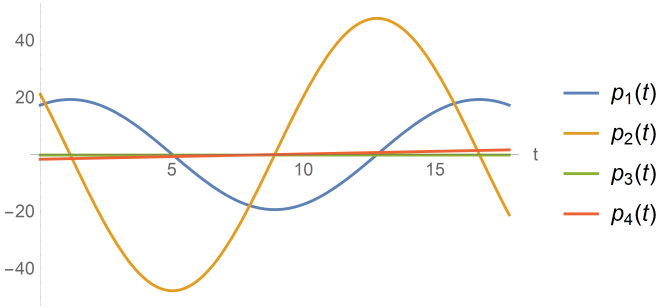
Bij een vergelijking blijkt dat de eerste methode beter is, maar dat het verschil niet groot is. De eerste methode is erg specifiek voor deze situatie, terwijl de tweede methode breder inzetbaar is en dus gemakkelijker aan een ander model aangepast kan worden.



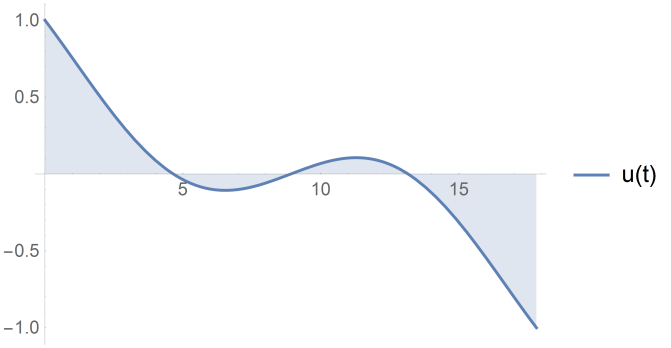
(a) De beweging van de simulatie door het $(\theta, \dot{\theta})$-vlak



(b) De beweging van de simulatie door het $(r, \dot{r})$-vlak

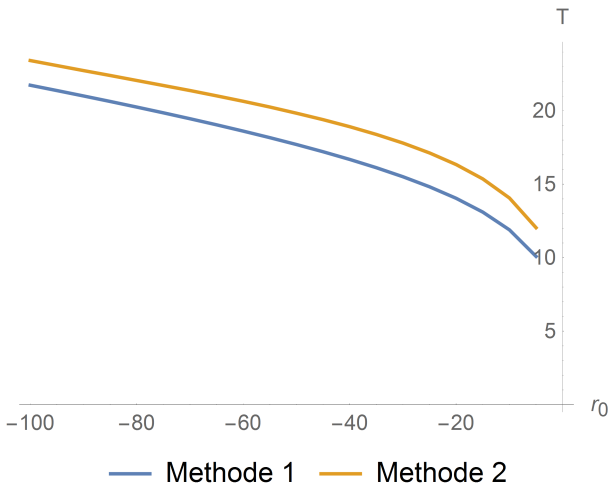


(c) De costates van het systeem

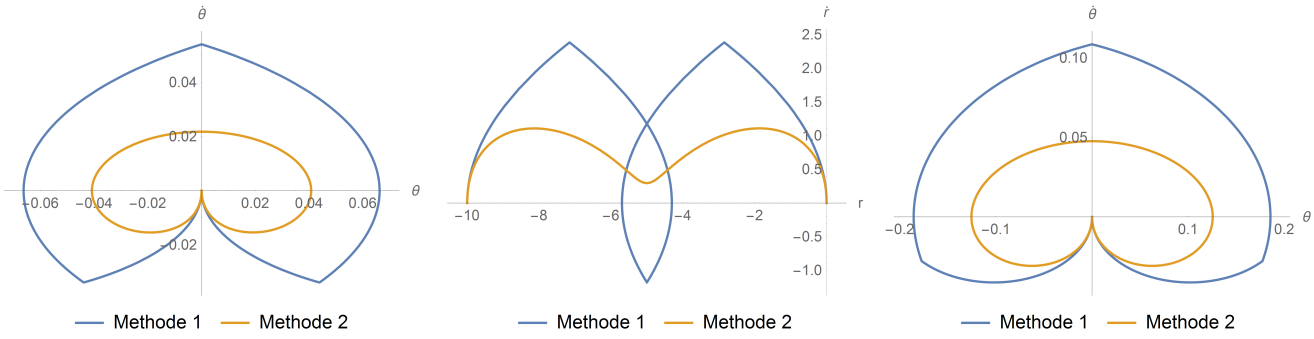


(d) De invoerkracht

FIGUUR 4.14: Een simulatie van $\dot{x} = Hx$ met $x_0 = (0, 0, -30, 0)^T$ en parameters $\ell = 60$, $g = 9.81$, $\ddot{r}_{\max} = 1.0$, $s = 10^6$ en $\varepsilon = 1$. De gevonden eindtijd is $T = 17.77$, met $|x(T)| = 0.0026$.



FIGUUR 4.15: Vergelijking van oplostijden T van de twee oplosmethodes voor verschillende begincondities r_0 tussen -100 en -10.



(a) Het $(\theta, \dot{\theta})$-vlak met $r_0 = -10$

(b) Het $(r, \dot{r})$-vlak met $r_0 = -10$

(c) Het $(\theta, \dot{\theta})$-vlak met $r_0 = -65$

(d) Het $(r, \dot{r})$-vlak met $r_0 = -65$

(e) Het $(\theta, \dot{\theta})$-vlak met $r_0 = -100$

(f) Het $(r, \dot{r})$-vlak met $r_0 = -100$

FIGUUR 4.16: Vergelijking van afgelegde banen van de twee oplosmethodes voor verschillende begincondities.

Hoofdstuk 5

Conclusie

In de voorgaande hoofdstukken hebben we gezien hoe voor twee verschillende dynamische systemen, de slinger en de kraan, een optimale regelaar kunnen bepalen.

Voor de slinger is er een simpel model gepresenteerd, dat op twee manieren is opgelost, met of zonder 'kosten' op de invoerkracht. Voor het probleem met kosten op de invoerkracht zijn twee oplossingen gepresenteerd, waarvan er één gebruikmaakt van de energie in het systeem en de andere van banen die oplossingen afleggen in het $2D$-vlak. De derde oplossing gebruikt een directe oplossing van de een niet tijd-optimale regelaar. Met een binary search kon ook daar een optimale invoer gevonden worden. Bij een vergelijking van de oplosmethodes van het slingerprobleem zien we dat de tweede methode het beste presteert.

De hijskraan is een toepassing van de slinger. De slinger kan bestuurd worden door het karretje om de boom van de kraan te versnellen of vertragen. Door een bestaand wiskundig model aan te passen zijn hiermee twee oplosmethodes gevonden voor het optimale regelaar probleem, net als bij de slinger met of zonder 'kosten' op de invoerkracht. Zonder kosten kon het probleem worden opgesplitst in twee gekoppelde systemen. Door middel van redeneren over de banen die de oplossing in twee $2D$-vlakken aflegt, kan dan een oplossing worden gevonden voor verschillende afstanden die het karretje moest afleggen. Ten slotte is er met de methode met kosten op de invoer een soortgelijke oplossing gevonden als voor de slinger, weliswaar met een groter systeem maar toepasbaar voor alle beginvoorwaarden in tegenstelling tot de eerste oplosmethode.

5.1 Toepasbaarheid

De oplostechnieken die hier zijn bediscussieerd zijn toepasbaar voor verschillende doeleinden.

Een slinger komt op veel verschillende plekken voor. Een voorbeeld waar onze oplosmethodes toepasbaar zijn is een deur die automatisch wordt dichtgedaan. Normaal wordt hier een kracht met demping uitgevoerd totdat de deur dichtvalt. Door gebruik te maken van één van de oplosmethodes kan een kracht worden uitgevoerd totdat de deur precies dichtvalt, zonder een grote klap.

Een toepassing in een kraan kan beperkt worden gebruikt voor een echte hijskraan omdat het model niet erg uitgebreid is. Echter kunnen de oplossingen wel op een horizontale kraan waarover gewichten aan een kabel worden verplaatst worden toegepast. Bijvoorbeeld in een haven waar containers op een schip of trein worden gezet.

Hierbij moet rekening worden gehouden dat de oplossingen die in dit onderzoek worden gepresenteerd grotendeels theoretisch zijn. Op fouten in metingen wordt bijvoorbeeld niet gelet: hierover staat in de volgende sectie meer.

5.2 Verder onderzoek

Dit onderzoek kan op meerdere manieren worden vervolgd. Per probleem worden een aantal verdere onderzoeksthema's toegelicht.

5.2.1 Slinger

In de eerste oplossing voor het probleem van de slinger is een methode gebruikt die energieën in het systeem gebruikt. Die methode is op een beperkt bereik aan situaties toepasbaar. Mogelijk kan met een betere redenatie over de oplossing uit de optimale regelaar meer switchpunten worden gevonden waardoor de methode breder inzetbaar is.

5.2.2 Kraan

De kraan heeft een aantal punten waarop verder onderzoek uitgevoerd kan worden.

Ten eerste is het gebruikte model gesimplificeerd. In dit onderzoek wordt alleen gekeken naar het karretje en de slingerhoek en -snelheid. Echter zou het interessant zijn om ook de hoek van de kraan en de lengte van de kabel te variëren. Wanneer de lengte van de kabel gevarieerd wordt levert dit een erg verschillend probleem op omdat het dynamische systeem andere eigenschappen krijgt.

Ook zijn er vergelijkingen bekend over buigingen van de kraan. Dit komt alleen voor bij grote krachten, maar kan fouten opleveren in de berekeningen zoals ze nu worden gedaan. Door verbuigingen mee te nemen worden verschillende elementen van het systeem afhankelijk van elkaar wat het oplossen moeilijker maakt.

Verder is het wiskundig interessant en nuttig om verdere beperkingen mee te nemen dan alleen die op de invoerkracht. Zo kan bijvoorbeeld de algemene uitwijking van de slinger, de uitwijking buiten het gebied onder de baan van het karretje en de snelheid van het karretje worden beperkt.

Ten slotte kunnen verstoringen in de metingen en van het systeem worden meegenomen in het model. De positie en hoek van het gewicht aan de kraan kan niet altijd perfect worden gemeten. Ook kan er wind aanwezig zijn bij een hijsoperatie, en kunnen er trillingen door de kraan lopen. Al deze dingen geven fouten tijdens het berekenen van een oplossing, en zouden geanalyseerd kunnen worden op hun invloed op de oplossing.

5.3 Ter afsluiting

Tijdens de afgelopen drie maanden is een onderzoek verricht dat tot veel inzichten heeft geleid. Niet alleen wiskundig zijn er interessante dingen ontdekt, ook onderzoekstechnisch is er veel geleerd.

Op wiskundig vlak heeft de theorie achter optimale regelaars een andere betekenis gekregen. Zoals de theorie werkt is een wiskundige oplossing een oplossing van het probleem. Niks is minder waar. In een van de simpelste denkbare systemen, de slinger, geeft een oplossing geen uitsluitsel over hoe deze te gebruiken is in bijvoorbeeld een simulatie.

Op onderzoeksvlak zijn er andere dingen ontdekt. Twee voorbeelden zijn de voorspelbaarheid van onderzoek en de ideeën die nodig zijn om een oplossing te vinden. Drie maanden geleden lag er een plan op tafel om volstrekt andere dingen te onderzoeken aan hijskranen. Het resultaat is anders, maar des te interessanter geworden. Ten tweede is het vaak niet mogelijk een probleem op te lossen door er lang naar te kijken en dingen te proberen. De ideeën die daadwerkelijk problemen oplossen ontstaan meestal pas als even iets heel anders wordt gedaan.

De afgelopen tijd is interessant, vernieuwend, frustrerend en vooral leerzaam geweest, en heeft een mooi resultaat opgeleverd.

Bijlage A

Code

Alle Mathematica code die gebruikt is om simulaties te draaien en eigenschappen af te leiden staat hieronder. De versie van Mathematica die gebruikt is, is *Mathematica* 10, Student Edition, 10.3.0.0 op Windows 10, 64 bits.

De code is hier weergegeven in *InputForm*. Hiermee kan de code rechtstreeks in Mathematica worden geplakt om uitgevoerd te worden. Een mooiere weergave kan worden bereikt door het om te zetten naar *StandardForm* in Mathematica.

Constanten

De definitie van constanten.

```
g = 9.81;
l = 60;
m = 1;
uMax = 1;
P = 2 \[Pi] Sqrt[l/g];
figureExport = FileNameJoin[{NotebookDirectory[], "..", "Report", "figures"}];
animationExport = FileNameJoin[{NotebookDirectory[], "..", "Presentation", "animations"
    }];
```

Ellipsen

De functie *withinEllipse* bepaalt of het punt p_1 binnen een ellips met centrum c en straal r valt.

```
withinEllipse[p1_, c_, r_] := (First@p1 - First@c)^2 + (Last@p1 - Last@c)^2/(g/(m l)) <
    r^2;
```

De functie *above*Γ bepaalt of een punt (x, y) boven de lijn Γ ligt.

```
above\[CapitalGamma][x_, y_] := If[x >= 0,
   y > 0 || withinEllipse[{x, y}, {(2 Floor[x/(2/g)] + 1) 1/g, 0}, 1/g],
   ! above\[CapitalGamma][-x, -y]
   ];
```

De functie *firstCuttingPoint* bepaalt het eerste snijpunt van de ellips met Γ voor een punt (x, y).

```
firstCuttingPoint[x_, y_] := Module[{k, r, xs},
   If[above\[CapitalGamma][x, y] || {0, 0} == {x, y},
    r = Sqrt[(x + 1/g)^2 + y^2/(g/(m l))];
    k = Floor[(r - 1/g)/(2/g)];
    xs = (r^2 - (1/g)^2 + ((2 k + 1)^2 - 1) (1/g)^2)/(4 (k + 1) (1/g));
    {xs, -Sqrt[g/(m l) (r^2 - (xs + 1/g)^2)]},
    -firstCuttingPoint[-x, -y]
    ]
   ];
```

De functie *findCuttingPoints* bepaalt het alle volgende snijpunten met Γ tot $(0,0)$ wordt bereikt, met tenminste één snijpunt.

```
findCuttingPoints[x_, y_] := First@Last@Reap[With[{cp = firstCuttingPoint[x, y]},
      Sow[cp]; findRemainingCuttingPoints @@ cp
      ]];
```

De functie *findCuttingPoints* bepaalt het alle volgende snijpunten met Γ tot $(0,0)$ wordt bereikt.

```
findRemainingCuttingPoints[x_, y_] := If[! (-(2/g) <= x <= 2/g),
   With[{cp = firstCuttingPoint[x, y]},
    If[(x - First@cp)^2 + (y - Last@cp)^2 > 0.01,
     Sow[cp]; findRemainingCuttingPoints @@ cp]
    ]
   ];
```

Hautus oplosmethode

De functie *solveHautus* vindt een oplossing en simuleert de oplossing voor de oplosmethode van de slinger voor een (x,y) en een ε_F waarbinnen wordt overgegaan op de invoer $u = Fx$.

```
solveHautus[xx0_, yy0_, threshold_] := Module[{
    A = ( {
       {0, 1},
       {-g/(l), 0}
      } ),
    B = ( {
       {0},
       {1/l}
      } ),
    F,
    x10 = xx0, x20 = yy0,
    x0 = xx0, y0 = yy0,
    T = 35,
    r, cp, points, stopThreshold, u, count, tt, te, sol, events, ts
    },
   r = Sqrt[(If[above\[CapitalGamma][x0, y0], x0 + 1/g, x0 - 1/g])^2 + (y0)^2/(g/(m l))
       ];
   cp = findCuttingPoints[x0, y0];
   points = Join[{{x0, y0}}, cp, {{0, 0}}];

   F = ( {
       {f11, f12}
      } ) /. First@Solve[Eigenvalues[A + B.( {
             {f11, f12}
            } )] == {-1, -1}];
   stopThreshold = threshold/100000;

   u[\[Theta]_, \[Theta]d_] = Piecewise[Join[{
       {(F.( {
             {\[Theta]},
             {\[Theta]d}
            } ))[[1, 1]], Sqrt[\[Theta]^2 + \[Theta]d^2] < threshold}, {1, \[Theta] < 0
              && \[Theta]d < 0}, {-1, \[Theta] > 0 && \[Theta]d > 0}},
      ({-Sign[First[#]], Abs[\[Theta]] > Abs[First[#]] && withinEllipse[{\[Theta], \[
         Theta]d}, {(Sign[First[#]] (2 Floor[Abs[First[#]]/(2/g)] + 1) (1/g)), 0}, 1/g
         ]}) & /@ cp
      ], -Sign[\[Theta]d]];

   count = 0;
   tt = T;
   te = T;
   {sol, events} = Reap[NDSolve[
      {( {
          {x1'[t]},
          {x2'[t]}
         } ) == A.( {
```

```
            {x1[t]},
            {x2[t]}
           } ) + B u[x1[t], x2[t]], x1[0] == x10, x2[0] == x20},
      {x1, x2}, {t, 0, T},
      PrecisionGoal -> 6,
      Method -> {"EventLocator", "Event" -> {u[x1[t], x2[t]], Sqrt[x1[t]^2 + x2[t]^2] <
          threshold, x1[t]^2 + x2[t]^2 < stopThreshold},
        "EventAction" :> {count++; Sow[Chop@{t, x1[t], x2[t], u[x1[t], x2[t]]}],
          tt = t; Print[StringForm["Final time: `` ", t]],
          Throw[te = t, "StopIntegration"]}
        }
      ]];
   tt = Min[tt, te];
   Print[StringForm["Interval: `` ", First@InterpolatingFunctionDomain[First[x1 /. sol
       ]]]];

   {sol, u, te, r, cp, ts, points, tt}
   ];
```

De functie *hautus* gebruikt een oplossing van de methode van Hautus om een serie grafieken te maken voor een punt (x, y) en een optionele ε_F.

```
Options[hautus] = {
   Threshold -> 0.005
   };
hautus[xx0_, yy0_, opts : OptionsPattern[]] :=
  Module[{
    x0 = xx0, y0 = yy0,
    threshold = OptionValue[Threshold],
    sol, r, u, te, cp, ts, points, tt
    },
   {sol, u, te, r, cp, ts, points, tt} = solveHautus[xx0, yy0, threshold];

   {Show[
     Quiet@RegionPlot[
       {above\[CapitalGamma][x, y], Not@above\[CapitalGamma][x, y]},
       {x, -1.05 r, 1.05 r}, {y, -1.05 r/Sqrt[((m l)/g)], 1.05 r/Sqrt[((m l)/g)]},
       MaxRecursion -> 6,
       PlotLegends -> {"u = -1", "u = 1"},
       ImageSize -> 600,
       AspectRatio -> Automatic,
       PlotRange -> Automatic,
       AxesOrigin -> {0, 0},
       AxesLabel -> {"x", "y"},
       PlotStyle -> Opacity[0.25],
       BoundaryStyle -> Black
       ],
     ParametricPlot[Evaluate@Join[
        {{If[above\[CapitalGamma][x0, y0], -1, 1] 1/g + r Cos[t], Sqrt[g/(m l)] r Sin[t
            ]}},
        With[{rr = \[Sqrt]((First@# + If[above\[CapitalGamma] @@ #, 1, -1] 1/g)^2 + (
            Last@#)^2/(g/(m l)))},
           {If[above\[CapitalGamma] @@ #, -1, 1] 1/g + rr Cos[t], Sqrt[g/(m l)] rr Sin[t
               ]}
           ] & /@ cp
        ],
      {t, 0, 2 \[Pi]},
      PlotStyle -> Take[ColorData[97, "ColorList"], 2],
      PlotLegends -> {"Ellips voor u = -1", "Ellips voor u = 1"}
      ],
     (*ParametricPlot[
     Join[Table[{(2k+1)(m l)/g+(m l)/gCos[t+\[Pi]],Sqrt[g/(m l)](m l)/gSin[t+\[Pi]]},{k
         ,0,2}],Table[{-(2k+1)(m l)/g+(m l)/g Cos[\[Pi]-t],Sqrt[g/(m l)](m l)/g Sin[\[Pi
         ]-t]},{k,0,2}]],
     {t,0,\[Pi]}
     ],*)
```

```
      ListPlot[{{{-(1/g), 0}}, {{1/g, 0}}}, PlotStyle -> PointSize[Large]],
      ListPlot[{#} & /@ points, PlotStyle -> Join[{{PointSize[Large], Green}}, {PointSize
          [Large], Purple} & /@ cp, {{PointSize[Large], Red}}],
       PlotLegends -> Join[{"(\!\(\*SubscriptBox[\(x\),␣\(0\)]\),\!\(\*SubscriptBox[\(y\)
          ,␣\(0\)]\))"}, Table[StringForm["(\!\(\*SubscriptBox[\(x\),␣\(``\)]\),\!\(\*
          SubscriptBox[\(y\),␣\(``\)]\))", kk, kk], {kk, 1, Length[cp]}], {"(\!\(\*
          SubscriptBox[\(x\),␣\(e\)]\),\!\(\*SubscriptBox[\(y\),␣\(e\)]\))"}]],
      ParametricPlot[
       Evaluate[{x1[t], x2[t]} /. sol],
       {t, 0, te},
       PlotStyle -> Red,
       PlotLegends -> {"Simulatie"}
       ],
      Axes -> True,
      AxesLabel -> {"x", "y"}
      ],
     ParametricPlot[
      Evaluate[{x1[t], x2[t]} /. sol],
      {t, 0, te},
      PlotStyle -> Red,
      PlotLegends -> Placed[{"Simulatie"}, Right],
      PlotRange -> {{-0.005, 0.005}, {-0.005, 0.005}},
      PlotRange -> Automatic,
      AxesOrigin -> {0, 0},
      AxesLabel -> {"x", "y"},
      ImageSize -> {300, 200},
      AspectRatio -> 2/3
      ],
     Plot[
      Evaluate[{x1[t], x2[t], u[x1[t], x2[t]]} /. sol],
      {t, 0, te},
      AxesLabel -> {"t"},
      PlotRange -> All,
      PlotLegends -> Placed[{"x(t)", "y(t)", "u(t)"}, Right],
      Filling -> {3 -> Axis},
      ImageSize -> {300, 200},
      AspectRatio -> 2/3
      ],
     FindRoot[Sqrt[(x1[t] /. First@sol)^2 + (x2[t] /. First@sol)^2] == threshold, {t, te
         /2, 0, te}]
     }
    ];
```

De functie *energy* gebruikt de oplosmethode met energieën om een serie grafieken te maken voor een punt $(x_{1,0}, x_{2,0})$ en een optionele eindtijd T.

```
Options[energy] = {
   T -> 5.5
   };
energy[x10_, x20_, opts : OptionsPattern[]] := Block[{
    A = ( {
       {0, 1},
       {-g/(m l), 0}
      } ),
    B = ( {
       {0},
       {1/(m l)}
      } ),
    T = OptionValue[T],
    sol, H, Eh, Ev, u, xx1, xx2, q, \[Theta]s, \[Theta]E, ts, QQ, te, QQn, QQp, W
    },
   Ev[\[Theta]_, \[Theta]d_] := 1/2 m (l \[Theta]d)^2;
   Eh[\[Theta]_, \[Theta]d_] := m g l (1 - Cos[\[Theta]]);(*m g l Sin[(\[Theta]/2)]^2*)

   \[Theta]s = If[x10 >= 0,
     1/2 (x10 + (Ev[x10, x20] + Eh[x10, x20])/l),
```

```
      1/2 (x10 - (Ev[x10, x20] + Eh[x10, x20])/l)
      ];
    Print[StringForm["\!\(\*SubscriptBox[\(\[Theta]\),␣\(s\)]\)␣=␣``", \[Theta]s]];

    u[\[Theta]_, \[Theta]d_] := Which[x10 >= 0,
      \!\(\*
TagBox[GridBox[{
{"\[Piecewise]", GridBox[{
{
RowBox[{"-", "1"}],
RowBox[{
RowBox[{"\[Theta]d", ">", "0"}], "||",
RowBox[{"\[Theta]s", "<", "\[Theta]"}]}]},
{"1",
RowBox[{"0", "<", "\[Theta]", "<=", "\[Theta]s"}]},
{"0",
RowBox[{"\[Theta]", "<=", "0"}]}
},
AllowedDimensions->{2, Automatic},
Editable->True,
GridBoxAlignment->{"Columns" -> {{Left}}, "ColumnsIndexed" -> {}, "Rows" -> {{Baseline
    }}, "RowsIndexed" -> {}, "Items" -> {}, "ItemsIndexed" -> {}},
GridBoxItemSize->{"Columns" -> {{Automatic}}, "ColumnsIndexed" -> {}, "Rows" -> {{1.}},
    "RowsIndexed" -> {}, "Items" -> {}, "ItemsIndexed" -> {}},
GridBoxSpacings->{"Columns" -> {
Offset[0.27999999999999997`], {
Offset[0.84]},
Offset[0.27999999999999997`]}, "ColumnsIndexed" -> {}, "Rows" -> {
Offset[0.2], {
Offset[0.4]},
Offset[0.2]}, "RowsIndexed" -> {}, "Items" -> {}, "ItemsIndexed" -> {}},
Selectable->True]}
},
GridBoxAlignment->{"Columns" -> {{Left}}, "ColumnsIndexed" -> {}, "Rows" -> {{Baseline
    }}, "RowsIndexed" -> {}, "Items" -> {}, "ItemsIndexed" -> {}},
GridBoxItemSize->{"Columns" -> {{Automatic}}, "ColumnsIndexed" -> {}, "Rows" -> {{1.}},
    "RowsIndexed" -> {}, "Items" -> {}, "ItemsIndexed" -> {}},
GridBoxSpacings->{"Columns" -> {
Offset[0.27999999999999997`], {
Offset[0.35]},
Offset[0.27999999999999997`]}, "ColumnsIndexed" -> {}, "Rows" -> {
Offset[0.2], {
Offset[0.4]},
Offset[0.2]}, "RowsIndexed" -> {}, "Items" -> {}, "ItemsIndexed" -> {}}],
"Piecewise",
DeleteWithContents->True,
Editable->False,
SelectWithContents->True,
Selectable->False]\) ,
      x10 < 0,
      \!\(\*
TagBox[GridBox[{
{"\[Piecewise]", GridBox[{
{"1",
RowBox[{
RowBox[{"\[Theta]d", "<", "0"}], "||",
RowBox[{"\[Theta]s", ">", "\[Theta]"}]}]},
{
RowBox[{"-", "1"}],
RowBox[{"\[Theta]s", "<=", "\[Theta]", "<", "0"}]},
{"0",
RowBox[{"\[Theta]", ">=", "0"}]}
},
AllowedDimensions->{2, Automatic},
Editable->True,
```

```
GridBoxAlignment->{"Columns" -> {{Left}}, "ColumnsIndexed" -> {}, "Rows" -> {{Baseline
    }}, "RowsIndexed" -> {}, "Items" -> {}, "ItemsIndexed" -> {}},
GridBoxItemSize->{"Columns" -> {{Automatic}}, "ColumnsIndexed" -> {}, "Rows" -> {{1.}},
    "RowsIndexed" -> {}, "Items" -> {}, "ItemsIndexed" -> {}},
GridBoxSpacings->{"Columns" -> {
Offset[0.27999999999999997`], {
Offset[0.84]},
Offset[0.27999999999999997`]}, "ColumnsIndexed" -> {}, "Rows" -> {
Offset[0.2], {
Offset[0.4]},
Offset[0.2]}, "RowsIndexed" -> {}, "Items" -> {}, "ItemsIndexed" -> {}},
Selectable->True]}
},
GridBoxAlignment->{"Columns" -> {{Left}}, "ColumnsIndexed" -> {}, "Rows" -> {{Baseline
    }}, "RowsIndexed" -> {}, "Items" -> {}, "ItemsIndexed" -> {}},
GridBoxItemSize->{"Columns" -> {{Automatic}}, "ColumnsIndexed" -> {}, "Rows" -> {{1.}},
    "RowsIndexed" -> {}, "Items" -> {}, "ItemsIndexed" -> {}},
GridBoxSpacings->{"Columns" -> {
Offset[0.27999999999999997`], {
Offset[0.35]},
Offset[0.27999999999999997`]}, "ColumnsIndexed" -> {}, "Rows" -> {
Offset[0.2], {
Offset[0.4]},
Offset[0.2]}, "RowsIndexed" -> {}, "Items" -> {}, "ItemsIndexed" -> {}}],
"Piecewise",
DeleteWithContents->True,
Editable->False,
SelectWithContents->True,
Selectable->False]\)
     ];

   (*sol=NDSolve[{x1'[t]\[Equal]x2[t],x2'[t]\[Equal]-(g/l)Sin[x2[t]]+u[t],x1[0]\[Equal]
       x10,x2[0]\[Equal]x20},{x1,x2},{t,0,T}];*)
   sol = First@NDSolve[{
       ( {
          {x1'[t]},
          {x2'[t]}
         } ) == A.( {
            {x1[t]},
            {x2[t]}
           } ) + B u[x1[t], x2[t]], x1[0] == x10, x2[0] == x20},
      {x1, x2}, {t, 0, T}
      ];
   xx1[t_] = x1[t] /. sol;
   xx2[t_] = x2[t] /. sol;

   ts = t /. FindRoot[xx1[t] == \[Theta]s, {t, T/2, 0, T}];
   te = t /. FindRoot[xx1[t] == 0, {t, T/2, 0, T}];
   Print[StringForm["Final time: ``", te]];
   Print[StringForm["Switch time: ``", ts]];

   W[t_] = WW[t] /. First@NDSolve[
       WW'[t] == -(-u[xx1[t], xx2[t]]) (l xx2[t]) &&
        WW[0] == 0
       , WW[t], {t, 0, T}
       ];

   {Plot[Evaluate[{x1[t], x2[t], u[x1[t], x2[t]]/(m l)} /. sol],
     {t, 0, T},
     PlotLegends -> {"\[Theta]", "\!\(\*OverscriptBox[\(\[Theta]\), \(.\)]\)", "u/l"},
     Filling -> {3 -> Axis},
     AxesLabel -> {"t"},
     PlotRange -> All,
     ImageSize -> {300, 200},
     AspectRatio -> 2/3
```

```
142       ],
143      ParametricPlot[{x1[t], x2[t]} /. sol, {t, 0, T}],
144      Plot[Evaluate[{
145         Ev[x1[t], x2[t]] /. sol,
146         Eh[x1[t], x2[t]] /. sol,
147         W[t],
148         Ev[x1[t] /. sol, x2[t] /. sol] + Eh[x1[t] /. sol, x2[t] /. sol] - W[t]
149         }], {t, 0, T},
150       PlotLegends -> {"\!\(\*SubscriptBox[\(E\),␣\(v\)]\)", "\!\(\*SubscriptBox[\(E\),␣\(
              h\)]\)", "W", "\!\(\*SubscriptBox[\(E\),␣\(v\)]\)+\!\(\*SubscriptBox[\(E\),␣\(h
              \)]\)-W"},
151       PlotRange -> All,
152       AxesLabel -> {"t"},
153       ImageSize -> {300, 200},
154       AspectRatio -> 2/3
155       ],
156      sol, u, te
157      }
158     ];
```

Kosten op de invoer

De functie *epsilonOC* zoekt een optimal control met kosten ε op de invoer, n variabelen, matrix A en B, kosten s op de eindtoestand, beginconditie $x_{0,\text{temp}}$ en het bereik van de binary search als $T_{\text{min,temp}}, T_{\text{max,temp}}$.

```
1  epsilonOC[n_, A_, B_, s_, eps_, x0temp_, {TminTemp_, TmaxTemp_}, \[Epsilon]_] := Module
       [{
2      Q = ConstantArray[0, {n, n}],
3      S = s IdentityMatrix[n],
4      sol, H, u, xx2, q, ts, \[Theta]s,
5      variables,
6      x0,
7      R,
8      xs, ps, xps, T, it,
9      maxIt = 25,
10     Tmin, Tmax
11     },
12    R = 2 eps;
13
14    x0 = {#} & /@ x0temp;
15    H = Join[Join[A, Q], Join[-B.(R^-1 B\[Transpose]), -A\[Transpose]], 2];
16    SetPrecision[H, 25];
17
18    xs = Table[{Subscript[x, i][t]}, {i, n}];
19    ps = Table[{Subscript[p, i][t]}, {i, n}];
20    xps = Join[xs, ps];
21    variables = Flatten@xps;
22
23    Tmin = TminTemp;
24    Tmax = TmaxTemp;
25    it = 0;
26    While[
27     Tmax - Tmin > \[Epsilon] && it < maxIt,
28     T = (Tmin + Tmax)/2;
29     sol = First@NDSolve[{
30         D[#, t] & /@ xps == H.xps,
31         (xs /. t -> 0) == x0,
32         (ps /. t -> T) == (S.xs /. t -> T)
33         }, variables, t
34        ];
35     u[t_] = Evaluate[(-(R^-1 B\[Transpose]).ps) /. sol];
36     If[
37      (Max[First@FindMaximum[{Abs[First@First@u[t]], 0 <= t <= T}, {t, #}] & /@ {T/100, T
           /2, 99/100 T}]) < 1,
```

```
38        Tmax = T,
39        Tmin = T
40        ];
41       it++;
42       ];
43
44      T = N@((Tmax + Tmin)/2);
45      sol = First@NDSolve[{
46          D[#, t] & /@ xps == H.xps,
47          (xs /. t -> 0) == x0,
48          (ps /. t -> T) == (S.xs /. t -> T)
49          }, variables, t
50         ];
51      u[t_] = Evaluate[(-(R^-1 B\[Transpose]).ps) /. sol];
52      Print[{s, eps, s/eps, LinearAlgebra`MatrixConditionNumber[H], Sqrt[Subscript[x, 1][t
            ]^2 + Subscript[x, 2][t]^2] /. sol /. t -> T}];
53
54      {Sqrt[Subscript[x, 1][t]^2 + Subscript[x, 2][t]^2] /. sol /. t -> T,
55       Plot[Evaluate[xs /. sol], {t, 0, T}, PlotLegends -> Flatten@xs, AxesLabel -> {"t"},
56        ImageSize -> 300],
57       Plot[Evaluate[ps /. sol], {t, 0, T}, PlotLegends -> Flatten@ps, PlotRange -> All,
             AxesLabel -> {"t"},
58        ImageSize -> 300],
59       ParametricPlot[
60        {Subscript[x, 1][t], Subscript[x, 2][t]} /. sol,
61        {t, 0, T},
62        AxesLabel -> {"\[Theta]", "\!\(\*OverscriptBox[\(\[Theta]\),␣\(.\)]\)"},
63        ImageSize -> 300
64        ],
65       If[n >= 4,
66        ParametricPlot[
67         {Subscript[x, 3][t], Subscript[x, 4][t]} /. sol,
68         {t, 0, T},
69         AxesLabel -> {"r", "\!\(\*OverscriptBox[\(r\),␣\(.\)]\)"},
70         AspectRatio -> 1/3,
71         ImageSize -> 300
72         ]
73        ],
74       Plot[u[t] /. sol, {t, 0, T}, PlotLegends -> {"u(t)"}, ImageSize -> {300, 200},
             AspectRatio -> 2/3, Filling -> Axis],
75       T,
76       sol,
77       u
78       }
79      ];
```

Tijd van een oplossing

De functie *time* zoekt de tijd die een oplossing erover doet om van x_0 naar *target* te komen met invoer u en een optie *precision* die aangeeft wanneer twee punten gelijk zijn. Hierin staan de vier vaste punten die een mogelijke oplossing geven gehardcoded.

```
1  Options[time] = {
2     Precision -> 10^-2
3     };
4  time[x0_, target_, u_, opts : OptionsPattern[]] := Block[{solves, sol, eps = OptionValue
       [Precision], sols, q, ret, x},
5     (*solves=DSolve[{m l x''[t]\[Equal]-g x[t]+u,x[0]\[Equal]First@x0,x'[0]\[Equal]
          Last@x0},x[t],t];*)
6     x[t_] := sinDistance[First@x0, Last@x0, u, t];
7     (*sol=First@solves;*)
8     (*sols=t/.Quiet[Solve[x[t]\[Equal]First[target],t]];*)
9     sols = {-(1/Sqrt[g]) Sqrt[l] Sqrt[m] ArcCos[(u^2 - g u First[target] - g u First[x0]
          + g^2 First[target] First[x0] - \[Sqrt](2 g^2 l m u First[target] Last[x0]^2 - g^3
```

```
          l m First[target]^2 Last[x0]^2 - 2 g^2 l m u First[x0] Last[x0]^2 + g^3 l m First
         [x0]^2 Last[x0]^2 + g^2 l^2 m^2 Last[x0]^4))/(u^2 - 2 g u First[x0] + g^2 First[x0
         ]^2 + g l m Last[x0]^2)],
10       1/Sqrt[g] Sqrt[l] Sqrt[m] ArcCos[(u^2 - g u First[target] - g u First[x0] + g^2
           First[target] First[x0] - \[Sqrt](2 g^2 l m u First[target] Last[x0]^2 - g^3 l m
            First[target]^2 Last[x0]^2 - 2 g^2 l m u First[x0] Last[x0]^2 + g^3 l m First[
           x0]^2 Last[x0]^2 + g^2 l^2 m^2 Last[x0]^4))/(u^2 - 2 g u First[x0] + g^2 First[
           x0]^2 + g l m Last[x0]^2)], -(1/Sqrt[g]) Sqrt[l] Sqrt[m]
11         ArcCos[(u^2 - g u First[target] - g u First[x0] + g^2 First[target] First[x0] +
             \[Sqrt](2 g^2 l m u First[target] Last[x0]^2 - g^3 l m First[target]^2 Last[x0
             ]^2 - 2 g^2 l m u First[x0] Last[x0]^2 + g^3 l m First[x0]^2 Last[x0]^2 + g^2
             l^2 m^2 Last[x0]^4))/(u^2 - 2 g u First[x0] + g^2 First[x0]^2 + g l m Last[x0
             ]^2)],
12       1/Sqrt[g] Sqrt[l] Sqrt[m] ArcCos[(u^2 - g u First[target] - g u First[x0] + g^2
           First[target] First[x0] + \[Sqrt](2 g^2 l m u First[target] Last[x0]^2 - g^3 l m
            First[target]^2 Last[x0]^2 - 2 g^2 l m u First[x0] Last[x0]^2 + g^3 l m First[
           x0]^2 Last[x0]^2 + g^2 l^2 m^2 Last[x0]^4))/(u^2 - 2 g u First[x0] + g^2 First[
           x0]^2 + g l m Last[x0]^2)]};
13     sols = Chop[sols];
14     sols = Re[Select[sols, Im[#] < eps &]];
15     sols = Sort[If[# <= 0, 2 \[Pi] Sqrt[g/(m l)] + #, #] & /@ sols];
16     ret = Select[sols,
17       Abs[x[#] - First@target] < eps
18         && Abs[x'[#] - Last@target] < eps &
19       ];
20
21     If[Length[ret] > 0,
22      First@ret
23      ]
24     ];
```

De functie *totalTime* zoekt de totale tijd om met de Hautus oplosmethode naar het punt $(0,0)$ te komen voor een punt (x_0, y_0) een een optie *precision*.

```
1  Options[totalTime] = {
2     Precision -> 10^-2
3     };
4  totalTime[x0_, y0_, opts : OptionsPattern[]] := Module[{cp, points, u},
5     cp = findCuttingPoints[x0, y0];
6     points = Join[{{x0, y0}}, cp, {{0, 0}}];
7     u = If[above\[CapitalGamma][x0, y0], -1, 1];
8     (*Print[cp,points,u];*)
9     Total[
10     (time[points[[#]], points[[# + 1]], u*If[OddQ[#], 1, -1], opts]) &
11      /@ Range[Length[points] - 1]
12     ]
13    ];
```

De functie *uFromSwitchTimes* geeft een invoer u gegeven de doorlooptijden T_k.

```
1  uFromSwitchTimes[Ts_] := Block[{u = 1, acc = Accumulate[Ts]},
2     Piecewise[Join[
3       {{u, 0 <= t < acc[[1]]}},
4       (u = -u; {u, acc[[#]] <= t < acc[[# + 1]]}) & /@ Range[Length[Ts] - 1],
5       {{0, True}}
6       ]]
7     ];
```

Afstanden

De functie *paraDistance* geeft de afstand die wordt afgelegd door een parabool $r(t)$ met $r(0) = x_0$, $r'(0) = y_0$, invoer u en tijd t.

```
1  paraDistance[r0_, rd0_, u_, t_] := 1/2 u uMax t^2 + rd0 t + r0;
```

De functie *sinDistance* geeft de afstand die wordt afgelegd door een sinusoide $r(t)$ met $r(0) = x_0$, $r'(0) = y_0$, invoer u en tijd t.

```
sinDistance[x0_, y0_, u_, t_] := 1/g (u - u Cos[(Sqrt[g] t)/(Sqrt[l] Sqrt[m])] + g x0
    Cos[(Sqrt[g] t)/(Sqrt[l] Sqrt[m])] + Sqrt[g] Sqrt[l] Sqrt[m] y0 Sin[(Sqrt[g] t)/(Sqrt
    [l] Sqrt[m])]);
```

De functie *dSinDistance* geeft de afstand die wordt afgelegd door de afgeleide van een sinusoide met $r(t)$ met $r(0) = x_0$, $r'(0) = y_0$, invoer u en tijd t.

```
dSinDistance[x0_, y0_, u_, t_] := (Sqrt[g] Sqrt[l] Sqrt[m] y0 Cos[(Sqrt[g] t)/(Sqrt[l]
    Sqrt[m])] + (u - g x0) Sin[(Sqrt[g] t)/(Sqrt[l] Sqrt[m])])/(Sqrt[g] Sqrt[l] Sqrt[m]);
```

Simulatie van kraan

De functie *finalOc* simuleert een kraan met beginconditie x_0, eindtijd T, invoer u en optionele *PlotLegends*.

```
Options[finalOc] = {
   PlotLegends -> None
   };
finalOc[x0temp_, T_, u_, opts : OptionsPattern[]] := Module[{
    A = ( {
       {0, 1, 0, 0},
       {-g/l, 0, 0, 0},
       {0, 0, 0, 1},
       {0, 0, 0, 0}
      } ),
    B = ( {
       {0},
       {-uMax/l},
       {0},
       {uMax}
      } ),
    n = 4,
    sol, q,
    variables,
    x0, xs
    },
   x0 = {#} & /@ x0temp;

   xs = Table[{Subscript[x, i][t]}, {i, n}];
   variables = Flatten@xs;

   sol = First@NDSolve[{
       D[#, t] & /@ xs == A.xs + B*u[t],
       (xs /. t -> 0) == x0
       }, variables, {t, 0, T}
      ];
   {
    StringForm["Errors:␣``,␣T:␣``,␣\!\(\*SubscriptBox[\(r\),␣\(0\)]\):␣``", First@Sqrt[
        Total[#^2 & /@ (xs /. sol /. t -> T)]], T, x0[[3]]],
    Plot[Evaluate[xs /. sol], {t, 0, T}, PlotLegends -> Flatten@xs,
     ImageSize -> 300, AxesLabel -> {"t"}],
    Plot[Evaluate[xs /. sol], {t, 99/100 T, T}, PlotLegends -> Flatten@xs,
     ImageSize -> 300, AxesLabel -> {"t"}],
    ParametricPlot[
     {Subscript[x, 1][t], Subscript[x, 2][t]} /. sol,
     {t, 0, T},
     PlotRange -> All,
     AxesLabel -> {"\[Theta]", "\!\(\*OverscriptBox[\(\[Theta]\),␣\(.\)]\)"},
     AspectRatio -> 2/3,
     AxesOrigin -> {0, 0},
     ImageSize -> {300, 200},
     PlotLegends -> OptionValue[PlotLegends]
```

```
      ],
     If[n >= 4,
      ParametricPlot[
       {Subscript[x, 3][t], Subscript[x, 4][t]} /. sol,
       {t, 0, T},
       PlotRange -> All,
       AxesLabel -> {"r", "\!\(\*OverscriptBox[\(r\), \(.\)]\)"},
       AxesOrigin -> {0, 0},
       AspectRatio -> 2/3,
       PlotLegends -> OptionValue[PlotLegends],
       ImageSize -> {300, 200}
       ]
      ],
     Plot[u[t] /. sol, {t, 0, T}, PlotLegends -> OptionValue[PlotLegends],
      ImageSize -> {300, 200}, AspectRatio -> 2/3, AxesLabel -> {"t", "u(t)"}, Filling ->
          Axis],
     sol
     }
    ];
```

Exporteren en animaties

De functie *export* exporteert *content* naar PDF met de naam *name*, wanneer *ex* waar is. Er wordt automatisch over lijsten gelopen.

Het is nuttig om een optionele exporteer functie te hebben die gemakkelijk van waar naar onwaar kan worden gezet wanneer alleen de *content* berekend moet worden.

```
export[content_List, name_, ex_] := MapIndexed[export[#1, name <> "-" <> ToString[First@
    #2], ex] &, content];

export[content_, name_, ex_] := Block[{},
   If[ex, Export[
     FileNameJoin[{figureExport, name <> ".pdf"}], content
     ]];
   content
   ];
```

De functie *exportRaster* exporteert *content* naar PNG met de naam *name*, wanneer *ex* waar is. Er wordt automatisch over lijsten gelopen.

```
exportRaster[content_List, name_, ex_] := MapIndexed[exportRaster[#1, name <> "-" <>
    ToString[First@#2], ex] &, content];
exportRaster[content_, name_, ex_] := Block[{},
   If[ex, Export[
     FileNameJoin[{figureExport, name <> ".png"}], content, ImageResolution -> 600
     ]];
   content
   ];
```

De functie *exportAnimation* exporteert *content* naar een Flash bestand met de naam *name* voor $t_{\min} \leq t \leq t_{\max}$ met stappen *step*, wanneer *ex* waar is. Als *ex* niet waar is wordt een *Animate* teruggegeven voor hetzelfde tijdsbereik.

```
exportAnimation[content_, name_, {tMin_, tMax_, step_: 1}, ex_] := If[ex,
   (*Export[
   FileNameJoin[{animationExport,name<>".mov"}],
   Table[Evaluate[content@t],{t,tMin,tMax,step}],
   "VideoEncoding"\[Rule]"Cinepak"
   ],*)
   {Export[
     FileNameJoin[{animationExport, name <> ".png"}],
     Evaluate[content[tMin + step/100]]
     ],
    Export[
```

```
12       FileNameJoin[{animationExport, name <> ".swf"}],
13       ParallelTable[Evaluate[content[t]], {t, tMin, tMax, step}]
14       ]},
15     Animate[Evaluate[content[t]], {t, tMin, tMax}, AnimationRunning -> False]
16     ];
```

Animatie

De functie *swingBall* maakt een grafische bal op centrum c.

```
1  swingBall[c_] := Module[{coord, gradient, img, rastimg, shiny},
2     gradient = Image[DensityPlot[(x - y)^2, {x, -.8, .2}, {y, -1, 0}, PlotRangePadding ->
           0, Frame -> None, ColorFunction -> (GrayLevel[1 - #/2] &), ColorFunctionScaling
          -> True]];
3     coord = CirclePoints[100];
4     {Texture@gradient, EdgeForm@None, Polygon[c + # & /@ coord, VertexTextureCoordinates
          -> (coord/2 + .5)], Gray, Thickness[0.0001], Circle[c]}];
```

De functie *cartAnimation* maakt een grafische weergave van de situatie voor de toestand $(\theta, \dot{\theta}, r, \dot{r})$ met invoer u, een bereik voor r van $[r_{\min}, r_{\max}]$, en of het karretje bestuurd wordt of de slinger. Als opties kunnen de variabelen worden geprint, de kabellengte worden gevarieerd, de breedte van de afbeelding worden gewijzigd en ten slotte de waarden van θ en r worden weergegeven.

```
1  Options[cartAnimation] = {
2     CableLength -> 60,
3     Width -> 1024,
4     Variables -> False,
5     Values -> True
6     };
7  cartAnimation[{\[Theta]_, \[Theta]d_, r_, rd_}, u_, {rMin_, rMax_}, trolleyControl_,
       opts : OptionsPattern[]] := With[{
8      R = OptionValue[CableLength],
9      w = 1,
10     h = .5,
11     v = OptionValue[Variables]
12     },
13    Graphics[{
14      Line[{{r, 0}, {r + R Sin[\[Theta]], -R Cos[\[Theta]]}}],
15      {Thick, Line[{{rMin - w, H}, {rMax + w, H}}]} /. H -> If[trolleyControl, -8/5 h,
            0],
16      If[trolleyControl,
17       {Red, EdgeForm[Thickness[Medium]], Rectangle[{r - w, -h}, {r + w, h}], LightGray,
             Rectangle[{r - w, -h}, {r + w, h}]}
18       ],
19      If[trolleyControl,
20       {GrayLevel[.45], Disk[{r - 8/10 w, -h}, 3/5 h], Disk[{r + 8/10 w, -h}, 3/5 h]}
21       ],
22      swingBall[{r + R Sin[\[Theta]], -R Cos[\[Theta]]}],
23      If[u != 0,
24       If[trolleyControl,
25        {Which[u rd > 0, Green, u rd < 0, Red], Arrow[{{r, 0}, {r + (rMax - rMin)/6 u,
              0}}]},
26        {Which[u \[Theta]d > 0, Green, u \[Theta]d < 0, Red], Arrowheads[.05], Arrow[{{r
              + (R + 2) Sin[\[Theta]], -(R + 2) Cos[\[Theta]]}, {r + (R + 2) Sin[\[Theta]] +
               (rMax - rMin)/8 u Cos[\[Theta]], -(R + 2) Cos[\[Theta]] + (rMax - rMin)/8 u
              Sin[\[Theta]]}}]}
27       ]
28       ],
29      If[v && u != 0,
30       If[trolleyControl,
31        {Gray, Text[If[OptionValue[Values], StringForm["u=``", Round[u, 10^-1]], "u"], {r
               + (rMax - rMin)/6 u + If[u > 0, .5, -.5], 0}, {If[u > 0, -1, 1], 0},
              Background -> White]},
```

```
32          {Gray, Text[If[OptionValue[Values], StringForm["u=``", Round[u, 10^-1]], "u"], {r
                + (R + 3) Sin[\[Theta]], -(R + 3) Cos[\[Theta]]}, {0, -1}, Background ->
               White]}
33          ]
34         ],
35        If[v,
36         {Dashed, Gray, Line[{{r, -R - 3}, {r, 0}}], Circle[{r, 0}, R/4, {-(\[Pi]/2), -(\[
              Pi]/2) + \[Theta]}], Text[If[OptionValue[Values], StringForm["\[Theta]=``", \[
              Theta]], "\[Theta]"], {r + (R/3) Sin[\[Theta]/2], -(R/3) Cos[\[Theta]/2]}, {0,
              0}, Background -> White]}
37         ],
38        If[v && trolleyControl,
39         {Gray, Text[If[OptionValue[Values], StringForm["r=``", r], "r"], {r, 3 h}, {0, 1},
               Background -> White]}
40         ]
41        },
42       PlotRangePadding -> 0,
43       PlotRange -> {{Min[rMin - 5, -R/Sqrt[3] - 5], Max[rMax + 5, R/Sqrt[3] + 5]}, {-R -
           3, 3}},
44       ImageSize -> OptionValue[Width],
45       Axes -> If[trolleyControl, {False, True}, False],
46       Ticks -> None
47       ]];
```

Drie switchtijden voor de kraan

De functie *find3Times* vindt een oplossing voor de kraan met r_0 met drie switchtijden, en maakt een serie grafieken.

```
1  find3Times[r0_] := Module[{r1, r2, rd0, rd1, rd2, x0, x1, x2, y0, y1, y2, s1, s2, q, P,
       root},
2     rd0 = 0;
3     r1[T1_] = paraDistance[r0, rd0, 1, T1];
4     rd1[T1_] = rd0 + uMax T1;
5     r2[T1_, T2_] = paraDistance[r1[T1], rd1[T1], -1, T2];
6     rd2[T1_, T2_] = rd1[T1] - uMax T2;
7
8     x0 = 0;
9     y0 = 0;
10    x1[T1_] = sinDistance[x0, y0, 1, T1];
11    y1[T1_] = dSinDistance[x0, y0, 1, T1];
12    x2[T1_, T2_] = sinDistance[x1[T1], y1[T1], -1, T2];
13    y2[T1_, T2_] = dSinDistance[x1[T1], y1[T1], -1, T2];
14
15    (*Print[Solve[rr0+1/2T1^2+ T1 T2-1/2T2^2\[Equal]rr0/2,{T1}]];*)
16    s1[T2_] := -T2 - Sqrt[-r0 + 2 T2^2];
17    s2[T2_] := -T2 + Sqrt[-r0 + 2 T2^2];
18    (*Print[Solve[-T2-Sqrt[-rr0+2 T2^2]\[Equal]T2,rr0]];*)
19
20    P = 2 \[Pi] Sqrt[(m l)/g];
21    root = FindRoot[x2[s2[TT2], TT2], {TT2, P/8, 0, P/4}];
22    If[Length[root] > 0,
23     q = {T1 -> s2[TT2], T2 -> TT2} /. root;
24     {
25      StringForm["\!\(\*SubscriptBox[\(r\),␣\(0\)]\)␣=␣``", r0],
26      Show[
27       ListPlot[{{x0, y0}, {x1[T1], y1[T1]}, {x2[T1, T2], y2[T1, T2]}} /. q, PlotRange ->
            {{-0.4, 0.4}, {-0.3, 0.3}}, PlotStyle -> PointSize[Large]],
28       ParametricPlot[{sinDistance[x0, y0, 1, t], dSinDistance[x0, y0, 1, t]} /. q, {t,
           0, T1 /. q}],
29       ParametricPlot[{sinDistance[x1[T1], y1[T1], -1, t], dSinDistance[x1[T1], y1[T1],
           -1, t]} /. q, {t, 0, T2 /. q}]
30       ],
31      Show[
```

```
32        ListPlot[{{r0, rd0}, {r1[T1], rd1[T1]}, {r2[T1, T2], rd2[T1, T2]}} /. q, PlotStyle
              -> PointSize[Large], AxesOrigin -> {0, 0}],
33        ParametricPlot[{r0 + rd0 t + 1/2 uMax t^2, rd0 + uMax t} /. q, {t, 0, T1 /. q}],
34        ParametricPlot[{r1[T1] + rd1[T1] t - 1/2 uMax t^2, rd1[T1] - uMax t} /. q, {t, 0,
              T2 /. q}]
35        ],
36       q,
37       Plot[{x2[s2[T2], T2], y2[s2[T2], T2]}, {T2, 0, 50}, PlotLegends -> {"\[Theta
             ](\!\(\*SubscriptBox[\(t\),␣\(2\)]\))", "\!\(\*OverscriptBox[\(\[Theta]\),␣\(.\)
             ]\)(\!\(\*SubscriptBox[\(t\),␣\(2\)]\))"}, AxesLabel -> {"\!\(\*SubscriptBox[\(T
             \),␣\(2\)]\)"}, ImageSize -> {300, 200},
38        AspectRatio -> 2/3],
39       Plot[
40        {s1[T2], s2[T2]},
41        {T2, -20, 50}, AxesLabel -> {"\!\(\*SubscriptBox[\(T\),␣\(2\)]\)", "\!\(\*
              SubscriptBox[\(T\),␣\(1\)]\)"}, PlotLegends -> {"-\!\(\*SubscriptBox[\(T\),␣
              \(2\)]\)-\!\(\*SqrtBox[\(\(-\*SubscriptBox[\(r\),␣\(0\)]\)␣+␣2\\␣\*
              SuperscriptBox[SubscriptBox[\(T\),␣\(2\)],␣\(2\)]\)]\)", "-\!\(\*SubscriptBox
              [\(T\),␣\(2\)]\)+\!\(\*SqrtBox[\(\(-\*SubscriptBox[\(r\),␣\(0\)]\)␣+␣2\\␣\*
              SuperscriptBox[SubscriptBox[\(T\),␣\(2\)],␣\(2\)]\)]\)"},
42        ImageSize -> {300, 200},
43        AspectRatio -> 2/3],
44       2 (s2[T2] + T2) /. q
45       }
46      ]
47     ];
```

De functie *find5Times* vindt een oplossing voor de kraan met r_0 met vijf switchtijden, en maakt een serie grafieken.

```
1  Options[find5Times] = {
2     PlotLegends -> None
3     };
4  find5Times[rr0_, opts : OptionsPattern[]] := Module[{r1, r2, r3, s1, s2, x0, x1, x2, x3,
       sol, q},
5     r1 = paraDistance[r0, 0, 1, T1];
6     r2 = paraDistance[r1, uMax T1, -1, T2];
7     r3 = paraDistance[r2, uMax (T1 - T2), 1, T1];
8     (*Print[FullSimplify@r3];*)
9     (*Solve[r3\[Equal]r0/2,T1];*)
10    s1[T2_] := -(1/2) Sqrt[-r0 + T2^2];
11    s2[T2_] := 1/2 Sqrt[-r0 + T2^2];
12    (*Plot[Evaluate[{s1[T2],s2[T2]}/.r0\[Rule]-30],{T2,0,10}]*)
13    x0 = {0, 0};
14    x1[T2_] := # @@ {First@x0, Last@x0, 1, s2[T2]} & /@ {sinDistance, dSinDistance};
15    x2[T2_] := # @@ {First@x1[T2], Last@x1[T2], -1, T2} & /@ {sinDistance, dSinDistance};
16    x3[T2_] := Chop@FullSimplify[# @@ {First@x2[T2], Last@x2[T2], 1, s2[T2]} & /@ {
          sinDistance, dSinDistance}];
17    (*Plot[Evaluate[x3[T2]/.r0\[Rule]rr0],{T2,0,10}]*)
18    sol = FindRoot[First[x3[T2]] /. r0 -> rr0, {T2, 1, 0, 100}];
19    q = {T1 -> s2[T2]} /. r0 -> rr0;
20    Join[Most@finalOc[{0, 0, rr0, 0}, 4 T1 + 2 T2 /. q /. sol, Function[t,
21        uFromSwitchTimes[{T1, T2, T1, T1, T2, T1} /. q /. sol]
22        ], PlotLegends -> OptionValue[PlotLegends]],
23     {Plot[Evaluate[x3[T2] /. r0 -> rr0], {T2, 0, 35}, PlotLegends -> {"\[Theta](\!\(\*
           SubscriptBox[\(t\),␣\(3\)]\))", "\!\(\*OverscriptBox[\(\[Theta]\),␣\(.\)]\)
           (\!\(\*SubscriptBox[\(t\),␣\(3\)]\))"}, AxesLabel -> {"\!\(\*SubscriptBox[\(T\),␣
           \(2\)]\)"}, AspectRatio -> 2/3, ImageSize -> {300, 200}], 4 T1 + 2 T2 /. q /. sol
           }
24     ]
25    ];
```

Vergelijken van eindtijden

De functie *compareTrajectories* vergelijkt drie en vijf switchtijden voor een r_0 en maakt een serie grafieken.

```
compareTrajectories[r0_] := Block[{f1, f11, f2},
   f1 = find3Times[r0];
   f11 = Most@finalOc[{0, 0, r0, 0}, f1[[7]], Function[t,
       uFromSwitchTimes[{T1, T2, T2, T1} /. f1[[4]]]
       ], PlotLegends -> Placed[{"3 switchtijden"}, Below]];
   f2 = find5Times[r0, PlotLegends -> Placed[{"5 switchtijden"}, Below]];
   {Show[f11[[4]] /. _?ColorQ -> ColorData[97, "ColorList"][[1]], f2[[4]] /. {_?ColorQ
       -> ColorData[97, "ColorList"][[2]]}],
    Show[f11[[5]] /. _?ColorQ -> ColorData[97, "ColorList"][[1]], f2[[5]] /. {_?ColorQ
        -> ColorData[97, "ColorList"][[2]]}],
    Show[f11[[6]] /. _?ColorQ -> ColorData[97, "ColorList"][[1]], f2[[6]] /. {_?ColorQ
        -> ColorData[97, "ColorList"][[2]]}]
    }
   ];
```

De functie *plotSolveTimes* maakt een grafiek van de eindtijden T voor drie of vijf switchtijden voor een bereik van r_0.

```
plotSolveTimes[] := Block[{r0s = Range[-100, -5, 5]},
   ListLinePlot[{{#, find3Times[#][[7]]} & /@ r0s, {#, find5Times[#][[8]]} & /@ r0s},
       PlotLegends -> {"3 switchtijden", "5 switchtijden"}, AxesLabel -> { "\!\(\*
       SubscriptBox[\(r\), \(0\)]\)", "T"}, ImageSize -> {300, 200}, AspectRatio -> 2/3]
   ];
```

De functie *plotSolveTimes2* maakt een grafiek van de eindtijden T voor de oplosmethoden voor het kraanprobleem voor een bereik van r_0.

```
plotSolveTimes2[] := Block[{r0s = Range[-100, -5, 5]},
   ListLinePlot[{ParallelMap[{#, find3Times[#][[7]]} &, r0s], ParallelMap[{#, epsilonOC
       [4, ( {
            {0, 1, 0, 0},
            {-g/l, 0, 0, 0},
            {0, 0, 0, 1},
            {0, 0, 0, 0}
           } ), ( {
            {0},
            {-uMax/(m l)},
            {0},
            {uMax}
           } ), 10^6, 1, {0, 0, #, 0}, {0, 40}, 10^-2][[7]]} &, r0s]},
    PlotLegends -> Placed[{"Methode 1", "Methode 2"}, Below], AxesLabel -> { "\!\(\*
        SubscriptBox[\(r\), \(0\)]\)", "T"}, ImageSize -> {300, 200}, AspectRatio -> 2/3]
   ];
```

Bibliografie

[1] V.G. Boltyanskii. *Mathematical methods of optimal control.* Holt, Rinehart & Winston of Canada Ltd, juni 1971.

[2] W. Devesse. Slew control methods for tower cranes. Master of science thesis, Stockholm University, juni 2012.

[3] W. Devesse. A real-time optimal control method for swing-free tower crane motions. pages 336–341, Madison, WI, Verenigde Staten, augustus 2013. Automation Science and Engineering (CASE). doi: 10.1109/CoASE.2013.6653933.

[4] L. Markus E.B. Lee. *Foundations of Optimal Control Theory.* Krieger Pub Co, september 1986.

[5] M.L.J. Hautus. *Tijdoptimale besturing van lineaire systemen.* Stichting Mathematisch Centrum, 1992. Vakantiecursus van Centrum voor Wiskunde en Informatica.

[6] G. Meinsma. *Dictaat van Optimal Control.* University of Twente, januari 2015.

[7] S.T. Thornton. *Classical Dynamics of Particles and Systems.* Brooks Cole, 5th edition, juli 2003.