Thesis Semi-Autonomous Mobile Robot Navigation in Populated Envirments

Kevin Denis

27 september 2016

Doelstelling masterproef

Werkende algoritme maken voor maken van complexe locale paden, door middel van splines. Dit zal een uitbreiding geven aan een huidig algoritme dat enkel circulaire paden toelaat. Deze schiet tekort wanneer de rolstoel naast een deur is.

Week 0

Hierin is het werk gemaakt in de weken voor het begin van de academiejaar gebundeld.

Doel

- Concreet doel voor thesis tegen eind september.
- Literatuur studie over het implementeren van splines in pad planning.
- Leren werken met gitlab.

Bereikt

- Concreet doel voor thesis tegen eind september.

  Eerste visie is neergeschreven. Deze zal tijdens week 1 verfijnt worden.

- Literatuur studie over het implementeren van splines in pad planning.

  Methodes voor het afvlakken van paden die op een andere manier berekend zijn.

  - Bezier Curves : curve gaat door begin en eindpunt. Raaklijn begin en eindpunt hangt af van punt 2 en n-1. Dit is handig, want ik ben voledig vrij in het bepalen van het begin en eind orintatie van de curve. Begin = huidige pose, eind = gewenste pose.
  - Splines : elke controle punt heeft hetzelfde gewicht. Graad bepaald continuteit van de punten. Er zou makkelijk gekozen kunnen worden voor het gebruik van cubische splines. Deze zorgen ervoor dat de functie in elk punt continue is tot de 2de afgeleide (pad is C2 continu). Er is natuurlijk een duidelijk verschil tussen heb gebruik maken van spline methodes of af te vlakken en om te interpoleren.
  - NURBS : cfr. splines maar controle punt heeft een vrij te kiezen gewicht

Opmerkingen voor volgende week

- Benchtest maken om meerdere methodes (Bezier Curves, Splines en NURBS) uit te testen, zodat werking, parameters, vrijheid en rekentijd gekend zijn.
- Leren werken met gitlab.

Wat ik heb gelezen

- „Design and evaluation of a lookup-table based collision-checking approach for fixed sets of mobile robot paths" [5]
- „Manipulaiting b-spline based paths for obstacle avoidance in autonomous ground vehicles" [4]
- „Analysis of a Spline Based, Obstacle Avoiding Path Planning Algorithm" [3]
- „Path Planning Based on Bezier Curve for Autonomous Ground Vehicles" [2]
- „Curvature-continuous trajectory generation with corridor constraint for autonomous ground vehicles" [1]
- „Continuous Path Smoothing for Car-Like Robots Using B-Spline Curves" [6]
- „Implementation of the spline method for mobile robot path control" [7]
- „Optimal Trajectory Generation for Car-type Mobile Robot using Spline Interpolation" [8]

Referenties

[1] J. w Choi, R. E. Curry en G. H. Elkaim. „Curvature-continuous trajectory generation with corridor constraint for autonomous ground vehicles". In: *49th IEEE Conference on Decision and Control (CDC)*. 49th IEEE Conference on Decision and Control (CDC). Dec 2010, p. 7166–7171. DOI: 10.1109/CDC.2010.5718154.

[2] J. w Choi, R. Curry en G. Elkaim. „Path Planning Based on Bezier Curve for Autonomous Ground Vehicles". In: *World Congress on Engineering and Computer Science 2008, WCECS '08. Advances in Electrical and Electronics Engineering - IAENG Special Edition of the*. World Congress on Engineering and Computer Science 2008, WCECS '08. Advances in Electrical and Electronics Engineering - IAENG Special Edition of the. Okt 2008, p. 158–166. DOI: 10.1109/WCECS.2008.27.

[3] John Connors en Gabriel Elkaim. „Analysis of a Spline Based, Obstacle Avoiding Path Planning Algorithm". In: *2007 IEEE 65th Vehicular Technology Conference - VTC2007-Spring*. 2007 IEEE 65th Vehicular Technology Conference - VTC2007-Spring. San Diego, CA, apr 2007, p. 2565–2569. DOI: 10.1109/VETECS.2007.528.

[4] John Connors en Gabriel Elkaim. „Manipulaiting b-spline based paths for obstacle avoidance in autonomous ground vehicles". In: *Proceedings of the 2007 National Technical Meeting of The Institute of Navigation*. Proceedings of the 2007 National Technical Meeting of The Institute of Navigation. San Diego, CA, jan 2007, p. 1081–1088. URL: https://www.ion.org/publications/abstract.cfm?articleID=7204.

[5] Eric Demeester e.a. „Design and evaluation of a lookup-table based collision-checking approach for fixed sets of mobile robot paths". In: *International Symposium on Robotics* (2012).

[6] Mohamed Elbanhawi, Milan Simic en Reza N. Jazar. „Continuous Path Smoothing for Car-Like Robots Using B-Spline Curves". In: *Journal of Intelligent & Robotic Systems* 80.1 (2015), p. 23–56. DOI: 10.1007/s10846-014-0172-0. URL: http://dx.doi.org/10.1007/s10846-014-0172-0.

[7] H. Eren, Chun Che Fung en J. Evans. „Implementation of the spline method for mobile robot path control". In: *Proceedings of the 16th IEEE Instrumentation and Measurement Technology Conference, 1999. IMTC/99.* Proceedings of the 16th IEEE Instrumentation and Measurement Technology Conference, 1999. IMTC/99. Deel 2. 1999, 739–744 vol.2. DOI: 10.1109/IMTC.1999.776966.

[8] Rahee Walambe e.a. „Optimal Trajectory Generation for Car-type Mobile Robot using Spline Interpolation". In: *IFAC-PapersOnLine* 49.1 (2016), p. 601–606. DOI: 10.1016/j.ifacol.2016.03.121. URL: http://www.sciencedirect.com/science/article/pii/S2405896316301215.

Thesis Semi-Autonomous Mobile Robot Navigation in Populated Envirments

Kevin Denis

8 oktober 2016

Doelstelling masterproef

Werkende algoritme maken voor maken van complexe locale paden, door middel van splines. Dit zal een uitbreiding geven aan een huidig algoritme dat enkel circulaire paden toelaat. Deze schiet tekort wanneer de rolstoel naast een deur is.

Vereisten

- Niet 1 pad berekenen maar n paden, die gebruikt worden als hypothesen van navigatie-intentie.
- Snelheid: we willen aan $>$ 5 Hz gedeelde controle beslissingen nemen.
- Nauwkeurigheid: rolstoelen zijn groot in vergelijking met hun omgeving, met 5 à 10 cm speling aan beide kanten bij het rijden door een deur.
- Dynamica: vertraging en versnelling is beperkt omdat er gebruikers in de rolstoel zitten.
- Sterke invloed van castorwielen: kunnen we rekening houden met de stand van de castorwielen bij het genereren van paden?
- Onzekerheid bij het uitvoeren van paden in rekening brengen bij het genereren van paden.

Opmerkingen van Meneer Bruyninckx:

- Welke zijn de "kwaliteitscriteria" waaraan je paden gaan moeten voldoen?
  Deze moeten natuurlijk zo goed moeilijk overeenkom met de intentie van de bestuurder. Maar een andere belangrijke element in mijn ogen is de continuïteit van de paden. Zeker wanneer ze uit meerdere delen bestaan. En ander mogelijke kwaliteitscriteria zou zo laag mogelijke acceleratie hebben. Dit wordt in de context van comfortabel rijden zeker niet geapprecieerd.
- Welke ruimtelijke en temporele nauwkeurigheid hebben je toepassingen nodig? Welke dichtheid van paden?
  Omdat een rolstoel groot is vergeleken met zijn omgeving, heb ik maar nauwe tolerantie om door een deur te kunnen tijden. Ik snap de term "dichtheid van paden"niet, u deze kunnen uitleggen?
- In welke ruimte plan je: 2D, 3D, 2D+tijd, 3D+tijd?
  Voorlopig zal ik enkel in 3D (x,y,theta) rekenen. Later, wanneer ik ook de dynamica van de omgeving zal moeten inbrengen, zal ik de tijd kunnen gebruiken. Hiervoor zal ik dus zeker moeten werken met C-Space. In het geval dat ik enkel werk vet convexe polygonen kan ik het Star algoritme gebruiken.
- Kan je voorspelling gebruik maken van informatie uit het verleden, of begin je telkens helemaal van nul?
  Hier kom ik later terug op in.

- Hoe maak je je geplande paden klaar voor een "shared controlüitvoering?
  Door een set van n mogelijke paden te berekenen kan ik deze vergelijken met een schatting van de intentie van de gebruiker. Het vergelijken van de berekende paden met de intentie van de gebruiker heb ik momenteel nog niet verder verdiept.
- Hoe gaan de eigenschappen van de wielcontrole de probleemstelling van je paden beïnvloeden?
  De robot is differentiaal gestuurd. Alhoewel deze meer vrijheid geeft als b.v. een Ackerman, kan deze nog steeds limitatie hebben van snelheden bij bepaalde situaties. Hiervoor moet ik concrete waardes vinden, zodat ik dat in mijn pad planner kan toevoegen. Zoals eerder vermeld zal ik er ook voor moeten zorgen dat deze pad een bepaalde graad van continuïteit heeft (C-2 of G-2?).
- Hoe modelleer je paden in "populated environments"? Met andere woorden, welke dynamica ga je dan willen/moeten toevoegen aan de zuivere geometrie van splines?
  Hier kom ik later terug op in.
- Welke "horizon"(verleden + toekomst) moet je daarbij best in rekening nemen? Op welke dynamiek in de omgeving moet je reageren, en op welke manier?
  Hier kom ik later terug op in.

Week 1 - 2

Doel

- Opmerkingen van Meneer Bruyninckx toevoegen aan huidig visie doelstelling masterproef
- Een Configuration-Space implementatie in MATLAB uitwerken.
- Verder uitwerking van het stuksgewijs toevoegen van Bézier Curves, en graad van continuiteit garanderen.

Bereikt

- Opmerkingen van Meneer Bruyninckx toevoegen aan huidig visie doelstelling masterproef
- Verder uitwerking van het stuksgewijs toevoegen van Bézier Curves.

Cubic Spline Interpolation

Pro's

- Polynoom gaat door alle gegeven punten.
- Oriëntatie kan op een indirecte manier gegeven worden, door de waarde van de helling van de begin en eindpunt op te geve
- Toevoegen van punt zal enkel voor een lokale verandering zorgen (groot voordeel tegenover bvb Lagrange Interpolatie)

Con's

- Werkt enkel met stijgende waarde van x.
- Spline kan niet evenwijdig met x-as beginnen en evenwijdig met y as eindigen

Mogelijke oplossing voor deze problemen : Door spline in meerdere delen te berekenen en het lokaal assenstelsel te roteren kan dit probleem opgelost worden

Cubic Spline Approximation

Pro's

- Op eerste zicht niet logisch, maar door grote gewichten op begin en eindpunt te zetten zorgt men voor paden die door begin en eindpunt gaan
- Pad zal kleinere krommingen bevatten, comfortabeler voor de passagier
- Door met de interpoleer gewichten variëren creëert men licht verschillende paden.

Con's

- Zelfde opmerkingen als bij Cubic Spline Interpolation

Bézier curves

Pro's

Cubic Bézier Curve lijkt een ideale oplossing, omdat :

- Helling curve (dus oriëntatie robot) wordt bepaald door punt 2 en n-1.
- Bézier curves kunnen ook aan elkaar bevestigd worden. Voorbeeld algoritme: De Casteljau Construction ("Bézier Spline").
- Wel ervoor zorgen dat G-2 en C-2 continue is (zeker wanneer curve uit meerdere Bézier curves opbouw)

OPM

- Indien ik C2 continu Bézier curves aan elkaar toe wil voegen, en nog steeds vrij wil zijn voor positie en oriëntatie van begin en eindpunt, MOET ik 5de orde (4de graad) Bézier curves gebruiken.

Maar, moet dit echt? Zou een controller niet makkelijk deze niet continue versnelling op kunnen vangen?

Controle punten

Tot nu toe heb ik niet het probleem bekeken van waar ik de punten krijg, die ik gebruik als controle punt voor de splines. Deze zouden, onder andere kunnen komen van een globale pad planner. Kleine variaties op deze punten en orientaties zullen er voor zorgen dat ik meerdere paden kan maken
Een andere mogelijkheid zou zijn dat ik enkel de huidige positie en oriëntatie, doel en de free-space heb, en hiermee paden moet genereren. Dit laat veel meer vrijheid toe voor het genereren van paden

Concrete vragen

1. Mijn dilemma is het volgende : moet ik mij enkel in Bézier curves verdiepen, of proberen een twee oplossing te implementeren, dus zowel Cubic Spline Approximation en Bézier curves ?
2. Is C-2 continu eigenlijk een vereiste? Of is G-2 continu genoeg? Dit zou overeenkomen als een "lichte"storing voor de controller; versnelling in dezelfde richting maar niet zelfde amplitude
3. Meneer Bruyninckx, zou u de term "dichtheid van paden"kunnen uitleggen?

Opmerkingen voor volgende week

- Een concrete methode uitwerken voor het genereren van punten als input voor de locale pad planner.
- Invloed stad van castor wielen op pad
- Soccer robots van Eindhoven University of Technology bekijken.
- Planning Algorithms van Steven M. LaValle doornemen
- Doornemen masterproef van Karel Belaen (gebruik van splines voor het afvlakken van paden)

Masterproef Semi-Autonomous Mobile Robot Navigation in Populated Envirments

Kevin Denis

16 oktober 2016

Doelstelling masterproef

Figuur 1a: Een niet te drukke situatie in Brussel Noord, maar nog steeds hoog dynamisch. Eigen foto.

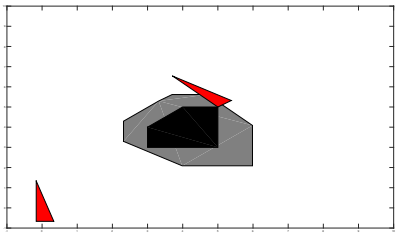
Figuur 1b: Een meer drukke omgeving, deze keer in Brussel Centraal. Door Fabian318 (wikipedia)

De doelstelling van deze masterproef is om een locale pad planner te ontwikkelen die complexe botsingsvrije paden genereert in een hoog dynamische omgeving. Voorbeelden van zulke omgevingen zijn weergeven in fig. 1a en fig. 1b. In beide omgevingen moet de pad planner ook de dynamiek van zijn omgeving kunnen schatten om botsingsvrije pad voor te kunnen stellen. Deze complexe locale paden zijn bestaan uit derde en vierde orde Bézier Curves. Het voorgesteld algoritme is een uitbreiding van een al bestaande pad planner die enkel circulaire paden genegeerde. Deze schiet tekort in sommige situaties, b.v.b. wanneer de rolstoel naast een deur is. Het algoritme dat de dynamiek van de omgeving in rekening brengt is nog niet ontwikkeld. Dit algoritme heeft als hoofddoel om toegepast te kunnen worden voor semi-autonome rolstoelen, welke bijkomende vereisten met zich meebrengt. Er moet niet één mogelijk pad uit dit algoritme komen maar meerdere, dit komt omdat de intentie van de gebruiker constant geschat moet worden en moet vergeleken worden met de mogelijke berekende paden. Het algoritme moet snel werken, (doel : 5 Hz) zodat de gebruiker niet merkt dat er een “latancy” is. Uiteindelijk moet het algoritme op een multi-resolie niveau werken, deze moet op korte afstand de invloed van het draaien van het castor wiel in rekening brengen en de imitatie van de dynamiek van de rolstoel zelf (maximale versnelling/vertraging, maximale kromming van pad, enz.).

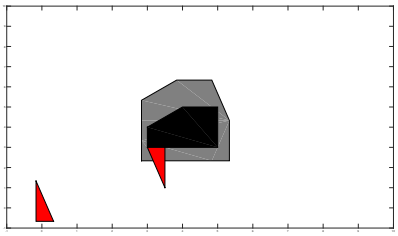
Week 3

Bereikt

- Een Configuration-Space implementatie in MATLAB.
  IN: (convex) polygon robot en (convex) polygon obstakel UIT : mogelijke botsingsvrije "posen" die de robot kan aannemen. Methode toegepast van Computational Geometry: Algorithms and Applications (Minkowski Sum en convhull (matlab functie)). *Was zeker geen vereiste, maar ik vond het gewoon interessant.*
- Verschillende manieren Bézier spline te berekenen met als gedachte zo snel mogelijke berekening, nodig voor implementatie. *Ook vrij vroeg, maar belangrijk in mijn ogen, zodat ik weet op welke verschillende manieren zo'n curve berekend kunnen worden en hun rekentijd.*
- Snelle lectuur van discrete pad planners (LaValle).
- Eerste draft van algoritme doorsturen en bespreken voor de implementatie in MATLAB.



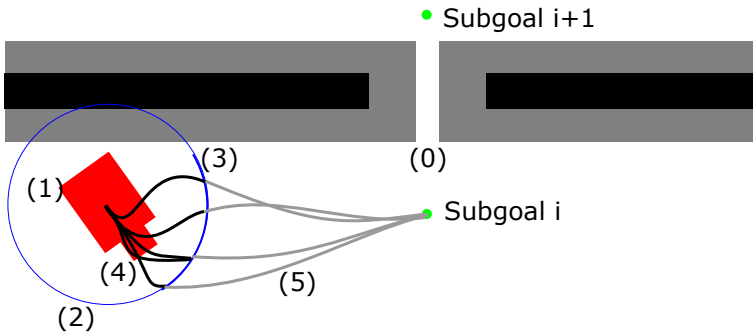
Figuur 2a: Rood: robot. Zwart : Obstakel. Grijs : Obstacle Space, deze is zowel afhankelijk van (x, y, θ)



Figuur 2b: Rood: robot. Zwart : Obstakel. Grijs : Obstacle Space, deze is zowel afhankelijk van (x, y, θ)

Algoritme voor locale pad planner

Overzicht

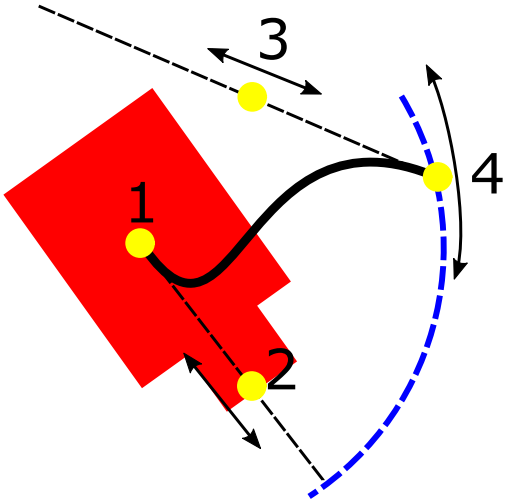


Figuur 3: Overzicht locale pad planner. Rood: robot. Zwart : Obstakel en locale paden. Grijs : 2de Bézier curve (eindput 1ste curve tot subgoal) en Obstacle Space, deze is zowel afhankelijk van (x, y, θ) (deze is op deze figuur niet correct, maar enkel toegevoegd ter illustratie).

0. Het algoritme verwacht de configuration space waarin de robot zich kan verplaatsen (x, y, θ), zodat deze kan nakijken of het gegenereerde paden botsingsvrij is. Er moet ook een subgoal zijn waar de robot zich naar toe moet verplaatsen. Eens dat de robot dicht genoeg is bij deze subgoal, moet deze naar de volgende gaan.
1. Huidige positie en oriëntatie van de robot (x, y, θ).
2. Een cirkel rond de robot legt de maximale afstand die de splines mogen afleggen vergelen met de huidige positie.
3. Enkel een bepaald deel van de cirkel (vet blauw) zal uiteindelijk gebruikt mogen worden waar de eindpunten van de splines zich zullen bevinden.
4. Splines worden gegenereerd met als startpunt de positie van de robot en zijn oriëntatie. Er zijn dus beperkingen op de plaats van de controle punten voor de Bézier curves. Op dit moment word er met cubische Bézier curves, deze bestaan uit 4 controle punten. De eigenschappen van een Bézier curves zijn dat ze door punt 1 en n (=4) gaan, en de oriëntatie (1ste afgeleide) word bepaald door punt 2 en n-1 (=3). De volgende parameters zijn vast :
   (a) Plaats van punt 1 op actuele positie van robot.
   (b) Richtingscoëfficiënt tussen punten 1 en 2 (= actuele oriëntatie robot).
   (c) Richtingscoëfficiënt tussen punt 3 en 4 moet er voor zorgen dat oriëntatie robot op cirkel “meer” wijst naar subgoal als in zijn begin oriëntatie.
   (d) Plaats van punt 4 : op een deel van cirkel.

   De volgende parameters zijn vrij :
   (a) Punt 2 op de rechte bepaald door de oriëntatie en beginpunt van de robot.
   (b) Punt 3 op de rechte bepaald door de eind oriëntatie en punt 4.
   (c) Punt 4 op specifiek deel van de cirkel.
5. De berekende Bézier curves in (4) zullen natuurlijk met een bepaalde costfunctie (dicht bij user-intention, lage kromming, afstand, etc.) met elkaar vergeleken worden. Men zou ook de Bézier cruve kunnen genereren tot de subgoal (in het grijs) en daar informatie van gebruiken voor het overwegen van welke locale pad het beste is (afstand, kromming, etc.).

Motivatie en specifieke uitleg voor positie controle punten



Figuur 4: Uitleg over de plaats van de vrije controle punten. Rood: robot. Zwart : locale pad gemaakt dankzij een Bézier Curve. Geel : ligging van controle punt voor deze specifieke Bézier curve. Stippellijnen duiden aan hoe controle punten 2, 3 en 4 mogelijk van plaats kunnen veranderen. De term resolutie betekend het aantal mogelijke liggingen op de rechte/kromme.

Waarom deze blauwe cirkel ? Het idee van de cirkel kwam door de volgende reden : om een Bézier curve te kunnen maken zijn er controle punten nodig. Om er voor te zorgen dat men niet in het wilde weg controle punten begin te genereren en om het accent tot waar de controle punten zich kunnen bevinden, was de cirkel in mijn ogen een goede oplossing.

Hoe kiest men nu deze deel van de cirkel die gebruikt wordt ? Ik denk dat ik mij hier kan laten inspireren door de Vector Field Histogram (normal, + en *) van Borenstein en Koren, deze hebben een interessante implementatie voor het kiezen van vrije sectoren. Op deze manier zou ik dus vrije sectoren op de blauwe cirkel rond de robot kunnen kiezen.

Waarom nu die 2de Bézier curve ? Ik denk dat er uit deze 2de Bézier curve nuttige informatie kan gehaald worden. Het antwoord in een zekere zin de vraag *"Hoe ver (positie en orientatie) ben ik van het subgoal"*.

En de resolutie ? Er zijn verschillende parameters die een bepaalde resolutie kunnen hebben (de plaatsen van punten 2 en 3 op de rechte bepaald door respectievelijk punten 1 en 4) en de plaats van punt 4 op de cirkel.

Opmerkingen voor volgende week

- Dynamische obstakel ontwijking literatuur, kijken hoe ik de ideeën kan implementeren in deze pad planner.
- Literatuur doornemen die in de opmerkingen van Meneer Demeester staat.
- Verder werken aan draft algoritme pad planner na feedback.
- Bijwerken van referentie bestand voor literatuur !!!

Masterproef Semi-Autonomous Mobile Robot Navigation in Populated Envirments

Kevin Denis

30 oktober 2016

Week 4-5

Bereikt

- Lokale padplanner formuleren als een COP, gegeven begin- en eindpositie en oriëntatie, voor een Bézierkromme van orde $n-1$. n is hier het aantal controlepunten.
- Eerste stappen in gitlab.

Overzicht van padplanner

Illustratie van de padplanner geformuleerd als een COP in MATLAB, met CASADI [1] en OPTISTACK (dit is een front-end/wrapper van CASADI voor MATLAB).

- Parameter
  De parameter die wordt geoptimaliseerd is de positie van de n controlepunten nodig voor het definiëren van de Bézierkromme van orde $n-1$.

$$z = [x_1, y_1, \ldots, x_n, y_n]^T \tag{1}$$

$$B(t) = BézierCurve(x, y, t) \tag{2}$$

  Met :

$$t = 0 : \Delta t : 1, \text{ dimensieloze parameter voor de constructie van de Bézierkromme} \tag{3}$$

- Kostfunctie (minimaliseren) (uit [2] [1])

$$f(z) = \int_0^1 \kappa(t)^2 dt + \alpha \int_0^1 s(t) dt \tag{4}$$

  Met :

$$\kappa(t) = \frac{B'(t) \times B''(t)}{\|B'(t)\|^3}, \text{ kromming} \tag{5}$$

$$s(t) = \int_0^1 \sqrt{B_x'^2 + B_y'^2} dt, \text{ lengte van de Bézierkromme} \tag{6}$$

$$\alpha, \text{ het relatieve gewicht tussen buiging-energie en lengte van de kromme}^2 \tag{7}$$

[1] hier wordt er geïntegreerd over de dimensieloze parameter t. Niet te verwarren met de tijd. In [2] wordt er over ds geïntegreerd, dit heb ik voorlopig nog niet geïmplementeerd.

[2] dit is niet helemaal correct, men moet hiervoor $\kappa(t)^2$ integreren over s en niet over t.

- Randvoorwaarden. *Voorlopig enkel geometrisch.*
  - Begin- en eindpunt liggen vast

$$[x_1, y_1] = [x1, y1] \tag{8}$$
$$[x_n, y_n] = [xn, yn] \tag{9}$$

  - Oriëntatie op begin- en eindpunt liggen vast

$$y_2 = \tan(\theta_1)(x_2 - x_1) \tag{10}$$
$$y_{end-1} = \tan(\theta_{end})(x_{end} - x_{end-1}) \tag{11}$$

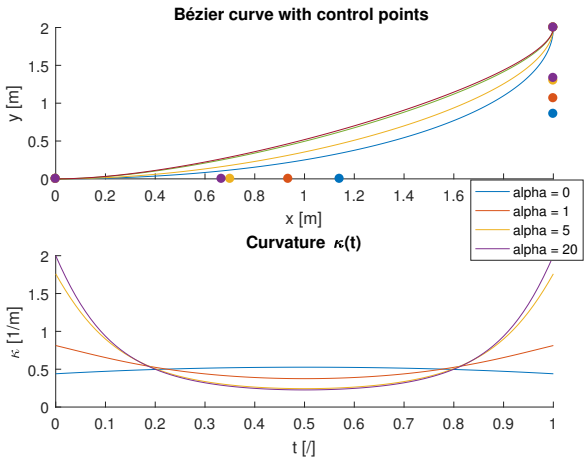
  - Maximale toegelaten kromming

$$\kappa(t) \leqslant \kappa_{max} \tag{12}$$

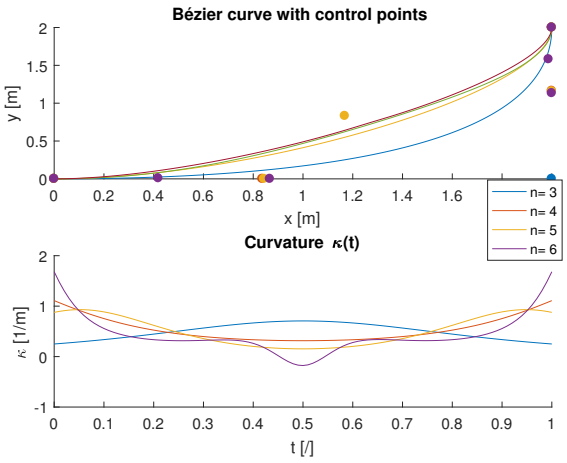
  - controlepunten $2 : n-1$ moeten binnen de rechthoek blijven gedefinieerd door punten 1 en n.

$$[x_1, y_1] <= [x_{2:n-1}, y_{2:n-1}] <= [x_n, y_n] \tag{13}$$

In fig. 1a en fig. 1b kan men figuren vinden van de optimalisatie resultaten met toenemende waarde van α en n respectievelijk.



Figuur 1a: Optimalisatie met verschillende waardes voor α. De kromming neemt toe en de lengte van de kromme daalt met toenemende waarde voor α, totdat deze de maximale waarde $\kappa_{max} = 2$ bereikt.



Figuur 1b: Optimalisatie met verschillende waardes voor n. Zowel de buiging-energie als de lengte van de kromme dalen met het toenemen n. De rekentijd neemt wel toe met toenemende n.

Wat ik heb gelezen

- „Path Planning for Mobile and Hyper-Redundant Robots Using Pythagorean Hodograph Curves" [2]
- „Real-Time Motion Planning in the Presence of Moving Obstacles" [4]
- „Optimal Trajectory Generation for Car-Type Mobile Robot Using Spline Interpolation" [6]
- *A Primer on Bézier Curves* [3]
- *Computer Aided Geometric Design* [5]

Opmerkingen voor volgende week

- Obstakel ontwijking toevoegen [4] en referentie hierin (Hypervlak).
- Gitlab aanvullen met eventueel literatuur, meeting rapport en logboek.
- Update doelstellingen masterproef en 4 concrete meetbare objectieven.

Referenties

[1] Joel Andersson. *A General-Purpose Software Framework for Dynamic Optimization (Een Algemene Softwareomgeving Voor Dynamische Optimalisatie)*. Leuven, 24 okt 2013. 186 p. ISBN: 978-94-6018-750-6. URL: https://lirias.kuleuven.be/handle/123456789/418048 (bezocht op 29-10-2016).

[2] H. Bruyninckx en D. Reynaerts. „Path Planning for Mobile and Hyper-Redundant Robots Using Pythagorean Hodograph Curves". In: *, 8th International Conference on Advanced Robotics, 1997. ICAR '97. Proceedings.* , 8th International Conference on Advanced Robotics, 1997. ICAR '97. Proceedings. Jul 1997, p. 595–600. DOI: 10.1109/ICAR.1997.620243.

[3] Mike Kamermans. *A Primer on Bézier Curves*. 13 jun 2013. URL: http://pomax.github.io/bezierinfo (bezocht op 30-10-2016).

[4] Tim Mercy, Wannes Van Loock en Goele Pipeleers. „Real-Time Motion Planning in the Presence of Moving Obstacles". In: Benelux Meeting on Systems and Control. Soesterberg, Netherlands, 24 mrt 2016.

[5] Thomas W. Sederberg. *Computer Aided Geometric Design*. 28 sep 2016. 288 p. URL: http://tom.cs.byu.edu/~557/text/cagd.pdf (bezocht op 30-10-2016).

[6] Rahee Walambe e.a. „Optimal Trajectory Generation for Car-Type Mobile Robot Using Spline Interpolation". In: *IFAC-PapersOnLine*. 4th IFAC Conference on Advances in Control and Optimization of Dynamical Systems ACODS 2016 49.1 (1 jan 2016), p. 601–606. ISSN: 2405-8963. DOI: 10.1016/j.ifacol.2016.03.121. URL: http://www.sciencedirect.com/science/article/pii/S2405896316301215 (bezocht op 26-09-2016).

Masterproef Semi-Autonomous Mobile Robot Navigation in Populated Envirments

Kevin Denis

14 november 2016

Doelstelling masterproef

Figuur 1a: Een niet te drukke situatie in Brussel Noord, maar nog steeds dynamisch. Eigen foto.

Figuur 1b: Een drukkere omgeving, deze keer in Brussel Centraal. Door Fabian318 (wikipedia)

De doelstelling van deze masterproef is om een lokale padplanner te ontwikkelen die paden genereert in een dynamische omgeving[1]. Voorbeelden van zulke omgevingen zijn weergegeven in fig. 1a en 1b. In beide omgevingen moet de padplanner ook de dynamiek van zijn omgeving kunnen schatten om botsingsvrije pad op te kunnen stellen. Deze complexe lokale paden bestaan uit vierde en vijfde orde Bézierkrommes. Het voorgesteld algoritme is een uitbreiding van een al bestaande padplanner die enkel circulaire paden genereert. Deze schiet tekort in sommige situaties, b.v.b. wanneer de rolstoel naast een deur is. Dit algoritme heeft als hoofddoel om toegepast te kunnen worden op een semi-autonome rolstoelen, welke bijkomende vereisten met zich meebrengt. Er moet niet één mogelijk pad uit dit algoritme komen maar meerdere. Dit, omdat deze gegenereerde paden als hypothesen gebruikt moeten worden voor de navigatie-intentie van de gebruiker. Het algoritme moet snel werken, (doel : 5 Hz) zodat de gebruiker niet merkt dat er vertraging is tussen zijn commando en het uitvoeren daarvan. Uiteindelijk moet het algoritme op een multi-resolutie niveau werken, deze moet op korte afstand de invloed van het draaien van het zwenkwiel in rekening brengen en de dynamische randvoorwaarden van van de rolstoel zelf (maximale versnelling/vertraging, maximale kromming van pad, enz.).

Objectieven

1. Ontwerpen van een lokale padplanner gebaseerd op Bézierkromme, geformuleerd als een COP.
2. Deze padplanner moet dynamische obstakels kunnen ontwijken.
3. Deze padplanner moet de dynamische beperkingen van de rolstoel in rekening brengen.
4. Deze padplanner moet snel genoeg uitgevoerd worden zodat de gebruiken geen last heeft van vertraging. Doel : 5Hz.

[1] Deze paden moeten "veilig" zijn voor de gebruiker en de personen om zich heen. Een mogelijke interpretatie hiervan is *sociaal aanvaardbaar* rij-gedrag. Daarmee wordt er bedoelt dat in een drukke station, het aanvaardbaar is om lichtjes tegen iemand te botsen. De implementatie hiervan is natuurlijk een uitdaging op zich.

Week 6-7

Bereikt

- Lokale padplanner formuleren als een COP, gegeven begin- en eindpositie en oriëntatie, voor een Bézierkromme van orde $n-1$. n is hier het aantal controlepunten. Deze houdt nu rekening met één obstakel. Deze moet ook convex zijn. De methode steunt op de theorema van de scheidende hypervlakken, uitgelegd in [1].
- Update doelstellingen masterproef en 4 concrete meetbare objectieven.

Overzicht van padplanner

Illustratie van de padplanner geformuleerd als een COP in MATLAB vgl. (1). De parameter die wordt geoptimaliseerd is de positie van de n controlepunten nodig voor het definiëren van de Bézierkromme van orde $n-1$.

Voorlopig is de obstakel-ontwijker een zeer dure operatie (van 0.2s naar 1.2s uitvoertijd). Dit kom omdat in het optimaliseringsproces op elke punt van Bézierkromme controleert of deze ver genoeg van het obstakel is. In [3], is er een elegantere methode (en ook sneller) ontwikkeld. Deze methode is nog niet geïmplementeerd.

$$\underset{z}{\text{minimize}} \quad f(z) = \int_0^l \kappa(t)^2 ds + \alpha \int_0^1 s(t) dt \text{ (uit [2])} \tag{1a}$$

$$\text{subject to} \quad [x_1, y_1] = [x1, y1], \tag{1b}$$

$$[x_n, y_n] = [xn, yn], \tag{1c}$$

$$y_2 = \tan(\theta_1)(x_2 - x_1), \tag{1d}$$

$$y_{n-1} = \tan(\theta_n)(x_n - x_{n-1}), \tag{1e}$$

$$\kappa(t)^2 \leqslant \kappa_{max}^2, \tag{1f}$$

$$a^T B(t) \leqslant b \text{ (uit [1], zie ook fig. 2)}, \tag{1g}$$

$$lb_x \leqslant x_i \leqslant ub_x, \tag{1h}$$

$$lb_y \leqslant y_i \leqslant ub_y. \tag{1i}$$

Met :

$t = 0 : \Delta t : 1,$ dimensieloze parameter voor de constructie van de Bézierkromme

$z = [x_1, y_1, \ldots, x_n, y_n]^T$

$B(t) = BezierCurve(x, y, t)$

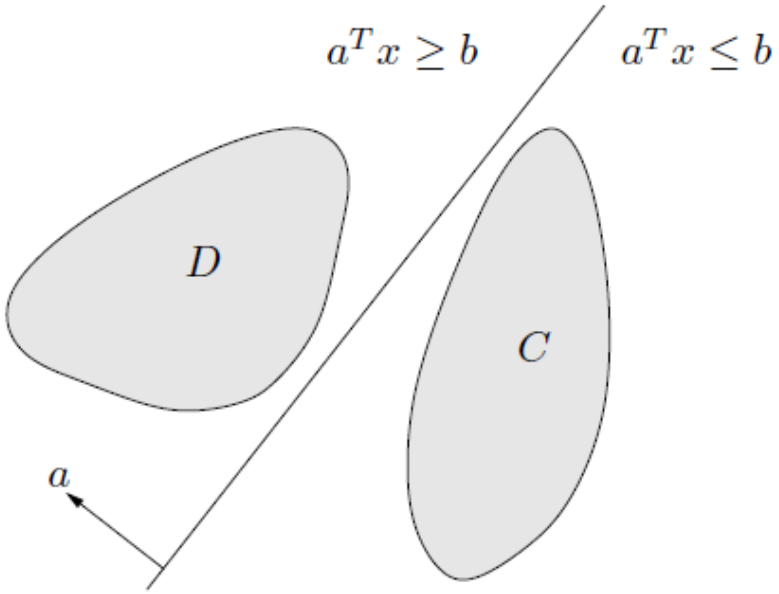
$\kappa(t) = \dfrac{B'(t) \times B''(t)}{\|B'(t)\|^3},$ kromming

$s(t) = \displaystyle\int_0^1 \sqrt{B_x'^2 + B_y'^2} dt,$ lengte van de Bézierkromme $s(t=1) = l$

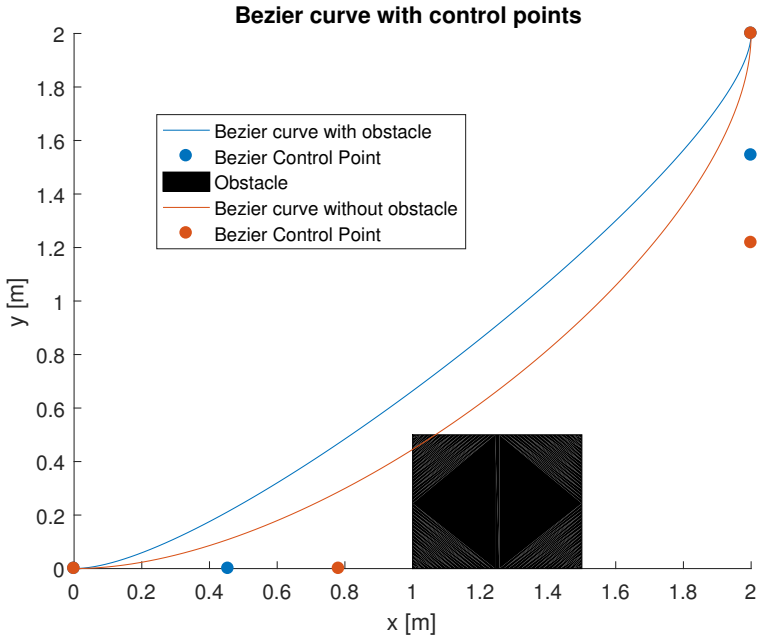
$\alpha,$ het relatieve gewicht tussen buiging-energie en lengte van de kromme

- Randvoorwaarden. *Voorlopig enkel geometrisch.*
  - Begin- en eindpunt liggen vast vgl. (1b) en (1c)
  - Oriëntatie op begin- en eindpunt liggen vast vgl. (1d) en (1e)
  - Maximale toegelaten kromming vgl. (1f)
  - Er moet een hypervlak getekend kunnen worden tussen elk punt van de kromme en het obstakel vgl. (1g).
  - Controlepunten moeten binnen een bepaalde grens blijven vgl. (1h) en (1i)

In fig. 3 kan men de resultaten van het optimaliseringsprobleem vinden met en zonder rekening te houden met een obstakel.



Figuur 2: Tussen twee niet overlappende convexe sets C en D is er altijd een scheidende hypervlak, bepaald voor $a^T x = b$. a is hier de vector bepaald door de twee dichtstbijzijnde punten in C en D. b de offset van deze vlak. In deze toepassing is C de robot en is voorlopig tot een cirkel herleid. D is hier een obstakel gevormd door een convexe polygoon.



Figuur 3: Optimalisatie met en zonder rekening houden van het obstakel, respectievelijk in het blauw en rood.

Wat ik heb gelezen

- *Convex Optimization* [1]
- „Real-Time Motion Planning in the Presence of Moving Obstacles" [3]

Opmerkingen voor volgende week

- Dynamica toevoegen, dus ook "hoe gaat men het pad volgen"(feedforward, feedback, mogelijke inspiratie in [3] en case-studie *Optimization of Mechatronic Systems*).
- Van een cirkelvormige geometrie naar een convexe polynoom geometrie voor het obstakel-ontwijking.
- Afspraak maken met promotors en mentors.
- Gitlab aanvullen met eventueel literatuur, meeting rapport en logboek.

Referenties

[1] Stephen Boyd en Lieven Vandenberghe. *Convex Optimization*. Cambridge: Cambridge university press, 2004. 716 p. ISBN: 0-521-83378-7.

[2] H. Bruyninckx en D. Reynaerts. „Path Planning for Mobile and Hyper-Redundant Robots Using Pythagorean Hodograph Curves". In: *, 8th International Conference on Advanced Robotics, 1997. ICAR '97. Proceedings.* , 8th International Conference on Advanced Robotics, 1997. ICAR '97. Proceedings. Jul 1997, p. 595–600. DOI: 10.1109/ICAR.1997.620243.

[3] Tim Mercy, Wannes Van Loock en Goele Pipeleers. „Real-Time Motion Planning in the Presence of Moving Obstacles". In: Benelux Meeting on Systems and Control. Soesterberg, Netherlands, 24 mrt 2016.

Masterproef Semi-Autonomous Mobile Robot Navigation in Populated Envirments

Kevin Denis

14 november 2016

Doelstelling masterproef

Figuur 1a: Een niet te drukke situatie in Brussel Noord, maar nog steeds dynamisch. Eigen foto.

Figuur 1b: Een drukkere omgeving, deze keer in Brussel Centraal. Door Fabian318 (wikipedia)

De doelstelling van deze masterproef is om een lokale padplanner te ontwikkelen die paden genereert in een dynamische omgeving[1]. Voorbeelden van zulke omgevingen zijn weergegeven in fig. 1a en 1b. In beide omgevingen moet de padplanner ook de dynamiek van zijn omgeving kunnen schatten om botsingsvrije pad op te kunnen stellen. Deze complexe lokale paden bestaan uit vierde en vijfde orde Bézierkrommes. Het voorgesteld algoritme is een uitbreiding van een al bestaande padplanner die enkel circulaire paden genereert. Deze schiet tekort in sommige situaties, b.v.b. wanneer de rolstoel naast een deur is. Dit algoritme heeft als hoofddoel om toegepast te kunnen worden op een semi-autonome rolstoelen, welke bijkomende vereisten met zich meebrengt. Er moet niet één mogelijk pad uit dit algoritme komen maar meerdere. Dit, omdat deze gegenereerde paden als hypothesen gebruikt moeten worden voor de navigatie-intentie van de gebruiker. Het algoritme moet snel werken, (doel : 5 Hz) zodat de gebruiker niet merkt dat er vertraging is tussen zijn commando en het uitvoeren daarvan. Uiteindelijk moet het algoritme op een multi-resolutie niveau werken, deze moet op korte afstand de invloed van het draaien van het zwenkwiel in rekening brengen en de dynamische randvoorwaarden van van de rolstoel zelf (maximale versnelling/vertraging, maximale kromming van pad, enz.).

Objectieven

1. Ontwerpen van een lokale padplanner gebaseerd op Bézierkromme, geformuleerd als een COP.
2. Deze padplanner moet dynamische obstakels kunnen ontwijken.
3. Deze padplanner moet de dynamische beperkingen van de rolstoel in rekening brengen.
4. Deze padplanner moet snel genoeg uitgevoerd worden zodat de gebruiken geen last heeft van vertraging. Doel : 5Hz.

[1] Deze paden moeten "veilig" zijn voor de gebruiker en de personen om zich heen. Een mogelijke interpretatie hiervan is *sociaal aanvaardbaar* rij-gedrag. Daarmee wordt er bedoelt dat in een drukke station, het aanvaardbaar is om lichtjes tegen iemand te botsen. De implementatie hiervan is natuurlijk een uitdaging op zich.

Week 6-7

Bereikt

- Lokale padplanner formuleren als een COP, gegeven begin- en eindpositie en oriëntatie, voor een Bézierkromme van orde $n-1$. n is hier het aantal controlepunten. Deze houdt nu rekening met één obstakel. Deze moet ook convex zijn. De methode steunt op de theorema van de scheidende hypervlakken, uitgelegd in [1].
- Update doelstellingen masterproef en 4 concrete meetbare objectieven.

Overzicht van padplanner

Illustratie van de padplanner geformuleerd als een COP in MATLAB vgl. (1). De parameter die wordt geoptimaliseerd is de positie van de n controlepunten nodig voor het definiëren van de Bézierkromme van orde $n-1$.

Voorlopig is de obstakel-ontwijker een zeer dure operatie (van 0.2s naar 1.2s uitvoertijd). Dit kom omdat in het optimaliseringsproces op elke punt van Bézierkromme controleert of deze ver genoeg van het obstakel is. In [3], is er een elegantere methode (en ook sneller) ontwikkeld. Deze methode is nog niet geïmplementeerd.

$$\underset{z}{\text{minimize}} \quad f(z) = \int_0^l \kappa(t)^2 ds + \alpha \int_0^1 s(t) dt \text{ (uit [2])} \tag{1a}$$

$$\text{subject to} \quad [x_1, y_1] = [x1, y1], \tag{1b}$$

$$[x_n, y_n] = [xn, yn], \tag{1c}$$

$$y_2 = \tan(\theta_1)(x_2 - x_1), \tag{1d}$$

$$y_{n-1} = \tan(\theta_n)(x_n - x_{n-1}), \tag{1e}$$

$$\kappa(t)^2 \leqslant \kappa_{max}^2, \tag{1f}$$

$$a^T B(t) \leqslant b \text{ (uit [1], zie ook fig. 2)}, \tag{1g}$$

$$lb_x \leqslant x_i \leqslant ub_x, \tag{1h}$$

$$lb_y \leqslant y_i \leqslant ub_y. \tag{1i}$$

Met :

$t = 0 : \Delta t : 1,$ dimensieloze parameter voor de constructie van de Bézierkromme

$z = [x_1, y_1, \ldots, x_n, y_n]^T$

$B(t) = BezierCurve(x, y, t)$

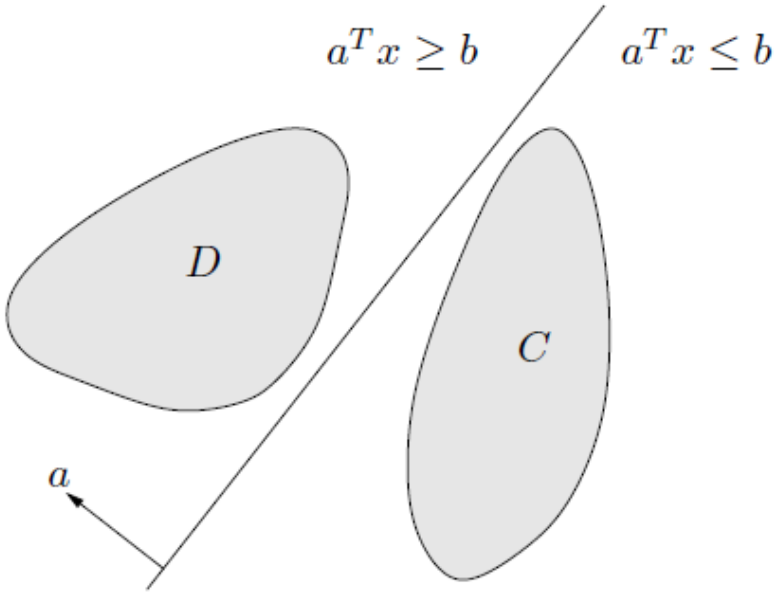
$\kappa(t) = \dfrac{B'(t) \times B''(t)}{\|B'(t)\|^3},$ kromming

$s(t) = \displaystyle\int_0^1 \sqrt{B_x'^2 + B_y'^2} dt,$ lengte van de Bézierkromme $s(t=1) = l$

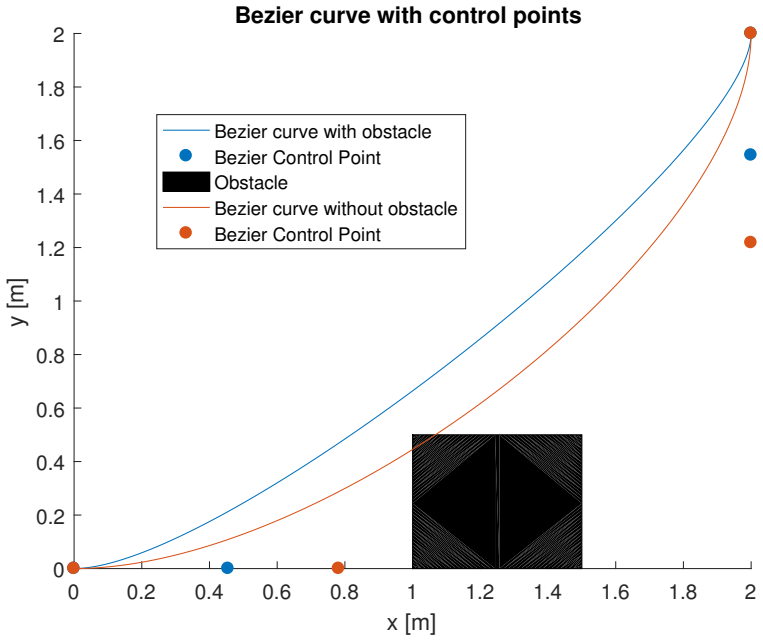
$\alpha,$ het relatieve gewicht tussen buiging-energie en lengte van de kromme

- Randvoorwaarden. *Voorlopig enkel geometrisch.*
  - Begin- en eindpunt liggen vast vgl. (1b) en (1c)
  - Oriëntatie op begin- en eindpunt liggen vast vgl. (1d) en (1e)
  - Maximale toegelaten kromming vgl. (1f)
  - Er moet een hypervlak getekend kunnen worden tussen elk punt van de kromme en het obstakel vgl. (1g).
  - Controlepunten moeten binnen een bepaalde grens blijven vgl. (1h) en (1i)

In fig. 3 kan men de resultaten van het optimaliseringsprobleem vinden met en zonder rekening te houden met een obstakel.



Figuur 2: Tussen twee niet overlappende convexe sets C en D is er altijd een scheidende hypervlak, bepaald voor $a^T x = b$. a is hier de vector bepaald door de twee dichtstbijzijnde punten in C en D. b de offset van deze vlak. In deze toepassing is C de robot en is voorlopig tot een cirkel herleid. D is hier een obstakel gevormd door een convexe polygoon.



Figuur 3: Optimalisatie met en zonder rekening houden van het obstakel, respectievelijk in het blauw en rood.

Wat ik heb gelezen

- *Convex Optimization* [1]
- „Real-Time Motion Planning in the Presence of Moving Obstacles" [3]

Opmerkingen voor volgende week

- Dynamica toevoegen, dus ook "hoe gaat men het pad volgen"(feedforward, feedback, mogelijke inspiratie in [3] en case-studie *Optimization of Mechatronic Systems*).
- Van een cirkelvormige geometrie naar een convexe polynoom geometrie voor het obstakel-ontwijking.
- Afspraak maken met promotors en mentors.
- Gitlab aanvullen met eventueel literatuur, meeting rapport en logboek.

Referenties

[1] Stephen Boyd en Lieven Vandenberghe. *Convex Optimization*. Cambridge: Cambridge university press, 2004. 716 p. ISBN: 0-521-83378-7.

[2] H. Bruyninckx en D. Reynaerts. „Path Planning for Mobile and Hyper-Redundant Robots Using Pythagorean Hodograph Curves". In: *, 8th International Conference on Advanced Robotics, 1997. ICAR '97. Proceedings.* , 8th International Conference on Advanced Robotics, 1997. ICAR '97. Proceedings. Jul 1997, p. 595–600. DOI: 10.1109/ICAR.1997.620243.

[3] Tim Mercy, Wannes Van Loock en Goele Pipeleers. „Real-Time Motion Planning in the Presence of Moving Obstacles". In: Benelux Meeting on Systems and Control. Soesterberg, Netherlands, 24 mrt 2016.

Masterproef Semi-Autonomous Mobile Robot Navigation in Populated Envirments

Kevin Denis

28 november 2016

Week 8-9

Bereikt

- Lokale padplanner formuleren als een COP, gegeven begin positie en oriëntatie, voor een Bézierkromme van orde $n-1$. n is hier het aantal controlepunten. Deze houdt nu rekening met één obstakel. Deze moet ook convex zijn. De methode steunt op de theorema van de scheidende hypervlakken en een lineaire clasificatie methode, uitgelegd in [1]. Belangrijke toevoegingen vergeleken met vorige log :
  - Eind positie en oriëntatie zijn geen randvoorwaarden meer. Deze wordt wel gebruikt in de doelfunctie (zie vgl. (1a))
  - Aanpassing van vgl. (1a), de verschillende termen zijn nu geschaald hun maximale waarde, dit maakt het wegen van de afzonderlijke termen makkelijker, zoals in [3].
  - Na een gesprek met de Assistenten van Numerische Optimalisatie maak ik terug gebruik van Casadi. Deze vonden dat de fmincon (niet-lineaire optimalisatie) functie van MATLAB onbetrouwbaar was. De formulering van mijn COP van MATLAB ondersteunde solver naar de Casadi solver heeft veel tijd genomen.
  - Obstakel ontwijker steunt nu op een conservatievere maar efficiëntere methode.
- Gitlab repo heeft nu een betere structuur. Daar zullen nu ook consequent updates staan van mijn logboek.

Overzicht van padplanner

Padplanner geformuleerd als een COP in vgl. (1). De parameter die wordt geoptimaliseerd is de positie van de n controlepunten nodig voor het definiëren van de Bézierkromme van orde $n-1$. De doelfunctie komt deels uit [2, 3].

Voorlopig is de obstakel-ontwijker zeer conservatief. Er moet gelden dat het convexe omhulsel, gevormd door de n controlepunten[1] en de obstakel gescheiden kan worden door een hypervlak (zoals in [5]), dit zorgt er echter wel voor een snellere rekentijd.

$$\underset{x,y,a,b}{\text{minimize}} \quad f(x,y) = \alpha_{curv}J_{curv} + \alpha_{length}J_{length} + \alpha_{dist}J_{dist} + \alpha_{angle}J_{angle} \tag{1a}$$

$$\text{subject to} \quad [x_1, y_1] = [x1, y1], \tag{1b}$$

$$y_2 = \tan(\theta 1)(x_2 - x_1) + y_1, \tag{1c}$$

$$\kappa(t)^2 \leqslant \kappa_{max}^2, \tag{1d}$$

$$a^T B(t) \leqslant b, \tag{1e}$$

$$a^T obs \geqslant b, \tag{1f}$$

$$lb_x \leqslant x \leqslant ub_x, \tag{1g}$$

$$lb_y \leqslant y \leqslant ub_y. \tag{1h}$$

[1] Voor een Bézierkromme geld dat deze zich altijd binnen het convexe vlak bevindt, beschreven door zijn controlepunten.

Met :

$t = 0 : \Delta t : 1,$ dimensieloze parameter voor de constructie van de Bézierkromme

$B(t) = BezierCurve(x, y, t)$

$$J_{curv} = \frac{\int_0^1 \kappa(t)^2 dt}{\kappa_{max}^2}$$

$$J_{length} = \frac{s(1)}{\|P_{goal} - B(0)\|_1}$$

$$J_{dist} = \frac{\|P_{goal} - B(1)\|_2}{\|P_{goal} - B(0)\|_1}$$

$$J_{angle} = \frac{\theta_{goal} - \theta_n}{\pi}$$

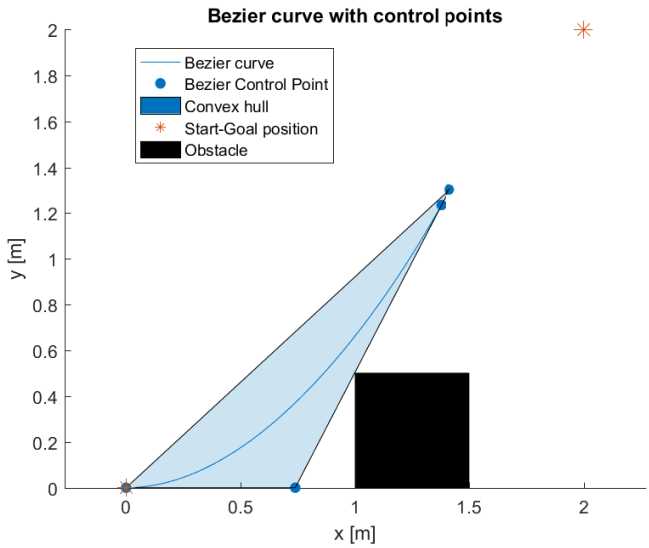
$$\kappa(t) = \frac{B'(t) \times B''(t)}{\|B'(t)\|^3},$$ kromming

$$s(t) = \int_0^1 \sqrt{B_x'^2 + B_y'^2} dt,$$ lengte van de Bézierkromme $s(t = 1) = l$

$\alpha_i,$ gewichten voor buiging-energie, totale lengte, verschil afstand en oriëntatie vgl met doel

- Randvoorwaarden. *Voorlopig enkel geometrisch*.
  - Begin positie en oriëntatie liggen vast vgl. (1b) en (1c)
  - Maximale toegelaten kromming vgl. (1d)
  - Er moet een hypervlak getekend kunnen worden tussen elk punt van de kromme en het obstakel vgl. (1e) en (1f) (uit [1]).
  - Controlepunten moeten binnen een bepaalde grens blijven vgl. (1g) en (1h)

In fig. 1 kan men de resultaten van het optimaliseringsprobleem vinden.



Figuur 1: Conservatie obstakel ontwijker steunend op het convex omhulsel gedefinieerd door de controlepunten van de Bézierkromme .

Wat ik heb gelezen

- „Real-Time Motion Planning in the Presence of Moving Obstacles" [5]
- *Convex Optimization* [1]
- „Fine Motion Planning for Shared Wheelchair Control: Requirements and Preliminary Experiments" [3]
- „Any-Angle Path Planning" [6]
- „Generating State Lattice Motion Primitives for Differentially Constrained Motion Planning" [7]

Opmerkingen voor volgende week

- Obstakel ontwijker minder conservatief maken. Mogelijke oplossing: gebruik maken van de Casteljau algoritme (1 Bézierkromme in 2 splitsen), en dan het convex omhulsel van deze twee curves gebruiken. Andere methode beschreven in [4] het convex omhulsel kleiner maken ("tight boxing of Bézier curves").
- Afspraak voorbereiden van donderdag 1 december met promotors en mentors.
- Gitlab aanvullen met eventueel literatuur en meeting rapport.

Referenties

[1] Stephen Boyd en Lieven Vandenberghe. *Convex Optimization*. Cambridge: Cambridge university press, 2004. 716 p. ISBN: 0-521-83378-7.

[2] H. Bruyninckx en D. Reynaerts. „Path Planning for Mobile and Hyper-Redundant Robots Using Pythagorean Hodograph Curves". In: *, 8th International Conference on Advanced Robotics, 1997. ICAR '97. Proceedings.* , 8th International Conference on Advanced Robotics, 1997. ICAR '97. Proceedings. Jul 1997, p. 595–600. DOI: 10.1109/ICAR.1997.620243.

[3] Eric Demeester, Marnix Nuttin en Hendrik Van Brussel. „Fine Motion Planning for Shared Wheelchair Control: Requirements and Preliminary Experiments". In: *Proceedings of the 11th International Conference on Advanced Robotics*. Coimbra, Portugal, jun 2003.

[4] Mike Kamermans. *A Primer on Bézier Curves*. 13 jun 2013. URL: http://pomax.github.io/bezierinfo (bezocht op 30-10-2016).

[5] Tim Mercy, Wannes Van Loock en Goele Pipeleers. „Real-Time Motion Planning in the Presence of Moving Obstacles". In: Benelux Meeting on Systems and Control. Soesterberg, Netherlands, 24 mrt 2016.

[6] Alex Nash en Sven Koenig. „Any-Angle Path Planning". In: *AI Magazine* 34.4 (Winter 2013), p. 85–107. ISSN: 07384602. URL: http://search.proquest.com.kuleuven.ezproxy.kuleuven.be/docview/1490901652/abstract/ECC7C534D5BD4F8DPQ/1 (bezocht op 28-11-2016).

[7] Mihail Pivtoraiko en Alonzo Kelly. „Generating State Lattice Motion Primitives for Differentially Constrained Motion Planning". In: International Conference on Intelligent Robots and Systems. Algarve, Portugal, 7 okt 2012, p. 101–108.

Masterproef Semi-Autonomous Mobile Robot Navigation in Populated Envirments

Kevin Denis

5 december 2016

Week 10

Bereikt

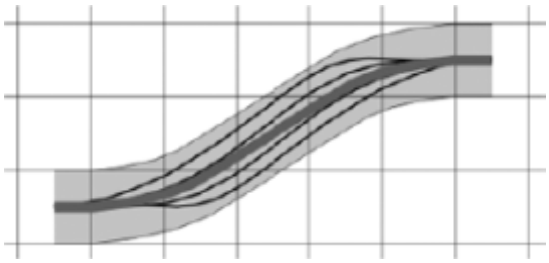
- Inzicht van State Lattice padplanner
- Link met COP padplanner die ik tot nu toe heb gebruikt

Overzicht State Lattice padplanner

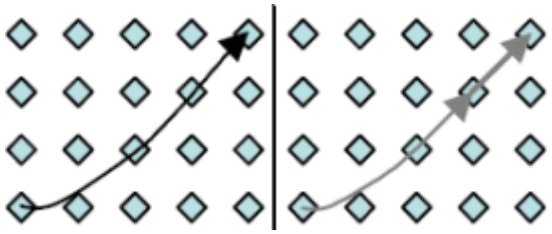
Een state lattice padplanner werkt in een zekere zin zoals een discrete grid padplanner : deze zal een pad plannen met een set van discrete beslissingsmogelijkheden.
In tegenstelling tot de klassieke grid padplanner, maakt de State Lattice padplanner gebruik van paden die op zichzelf al de randvoorwaarden (continuiteit in orientatie, maximale kromming en zelfs niet holonomiteit) in rekening brengen, deze worden bewegingsprimitieven (*E: motion primitives)* genoemt. Deze zijn op voorhand berekend en vormen *mogelijke set* van verbindingen tussen de discrete latice nodes. Er moet dus geen "post-processing" gedaan worden, eenmaal een optimaal pad (aaneensluiting bewegingsprimitieven van de State Lattice) gevonden is, dankzij een globale padplanner, omdat men al op voorhand de randvoorwaarden hebben opgelegd aan deze discrete set van bewegingen.
Deze set van bewegingsprimitieven is zo opgesteld dat :

- Paden met zelfde begin en eindpunt die een bepaalde gelijkenis vertonen (verschil $< \delta_s$) worden beschouwen als één zelfde pad. Ter illustratie fig. 1a.
- Paden moeten niet opgebouwd kunnen worden door de vorige berekende paden, (hun verschil moet dus weer groter als δ_s). Ter illustratie fig. 1b.

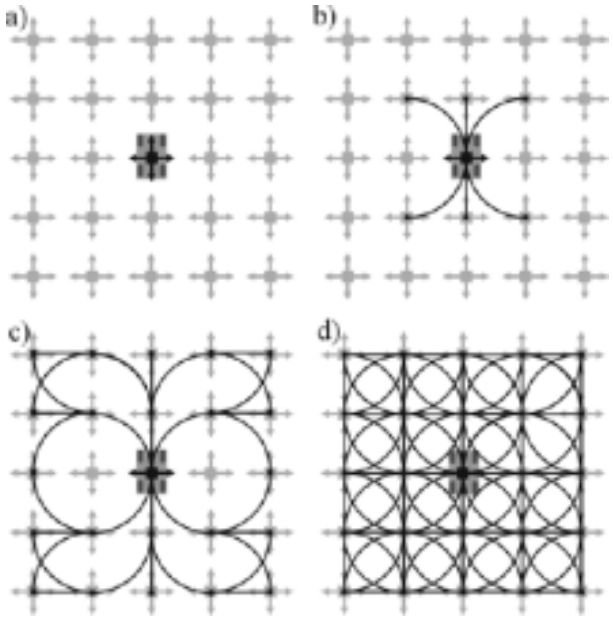
Deze twee punten zorgen er dus voor dat men werkt met een eindige, onafhankelijke set van bewegingsprimitieven, wat er voor gaat zorgen dat de globale padplanner sneller gaat werken. Ter illustratie fig. 2.



Figuur 1a: Paden tussen twee configuraties (dunne zwarte lijnen) binnen de omhullende (licht grijs), worden hervat tot één equivalent pad (dikke donkergrijze lijn) [2].



Figuur 1b: Een pad dat (ongeveer) kan worden opgebouwd door eerder bepaalde primitieven kan niet beschouwd worden als een bewegingsprimitief [2].



Figuur 2: Opstellen van een de lattice voor de een auto. a) Definitie van de discretisatie van de grid (x,y en orientatie). b) Men bepaald uitvoerbare paden van de oorsprong naar de 8 dichtstbijzijnde knopen c) Zelfde, maar nu naar de 24 dichtstbijzijnde knopen (enkele paden getoond) [2].

Mogelijke uitbreidingen en aandachtspunten van State Lattice padplanner

Multi-resolutie van discrete latice knopen, afhankelijk van [5]:

- Aafstand (om niet holonomiteit van zwenkwiel in rekening te brengen)
- Taak (dokken van rolstoel aan een tafel).
- Omgeving (nauwheid van gang). Hier zou men trouwens ook het gedrag zelfs, omgevingsafhankelijk maken :(bvb. als de rolstoel zich in een nauwe gang bevindt en deze is vast doordat er iemand de weg blokkeert, sta stil totdat deze persoon weg is (zoek geen andere alternatief, er zijn er geen!

Wat ik heb gelezen

- „Generating near Minimal Spanning Control Sets for Constrained Motion Planning in Discrete State Spaces" [2].
- „Efficient Constrained Path Planning Via Search In State Lattices" [3].
- „Generating State Lattice Motion Primitives for Differentially Constrained Motion Planning" [4].
- „Smooth Path Planning in Constrained Environments" [5].

Opmerkingen voor volgende week

- Gebruik maken van het tot nu toe ontworpen COP padplanner voor de bewegingsprimitieven.
- Verwoording (*voor tussentijdse masterproef presentatie !*) van hoe mijn werk binnen het werk van Meneer Demeester past [1].

Referenties

[1] Eric Demeester e.a. „Design and Evaluation of a Lookup-Table Based Collision-Checking Approach for Fixed Sets of Mobile Robot Paths". In: *International Symposium on Robotics* (2012).

[2] M. Pivtoraiko en A. Kelly. „Generating near Minimal Spanning Control Sets for Constrained Motion Planning in Discrete State Spaces". In: *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems. Aug 2005, p. 3231–3237. DOI: 10.1109/IROS.2005.1545046.

[3] Mihail Pivtoraiko en Alonzo Kelly. „Efficient Constrained Path Planning Via Search In State Lattices". In: *Proc. of 'The 8th International Symposium on Artifical Intelligence, Robotics and Automation in Space*. International Symposium on Artifical Intelligence, Robotics and Automation in Space. Munich, Germany., 5–8 sep 2005.

[4] Mihail Pivtoraiko en Alonzo Kelly. „Generating State Lattice Motion Primitives for Differentially Constrained Motion Planning". In: International Conference on Intelligent Robots and Systems. Algarve, Portugal, 7 okt 2012, p. 101–108.

[5] M. Rufli, D. Ferguson en R. Siegwart. „Smooth Path Planning in Constrained Environments". In: *2009 IEEE International Conference on Robotics and Automation*. 2009 IEEE International Conference on Robotics and Automation. Mei 2009, p. 3780–3785. DOI: 10.1109/ROBOT.2009.5152506.

Masterproef Semi-Autonomous Mobile Robot Navigation in Populated Envirments

Kevin Denis

26 februari 2017

Semester 2, Week 1 - 2

Bereikt

- Gebruik maken van clothoïde als bewegingsprimitiven i.p.v. cubische Bézier Curves.
- State Lattice padplanner dat zo snel mogelijk kiest voor rechtlijnige beweging.
- Conceptuele "Backup-plan" indien State Lattice slecht schaalt voor fijne beweging.

Nieuwe bewegingsprimitief : de clothoïde

De belangrijkste eigenschap van de clothoïde is dat deze een lineair variërende kromming heeft i.f.v. zijn lengte. Rechte en cirkels zijn dus bijzondere waar de kromming constant is.

$$\kappa(s) = \kappa_0 + \kappa' s \tag{1}$$

Met :

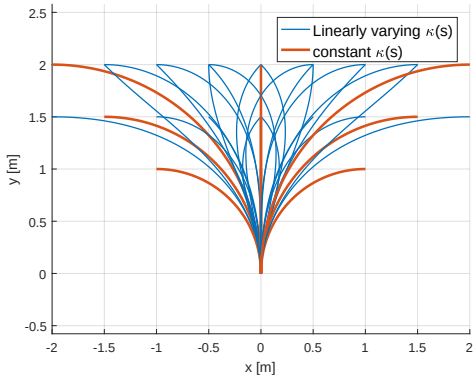
$\kappa(s)$, kromming

κ_0, initiële kromming

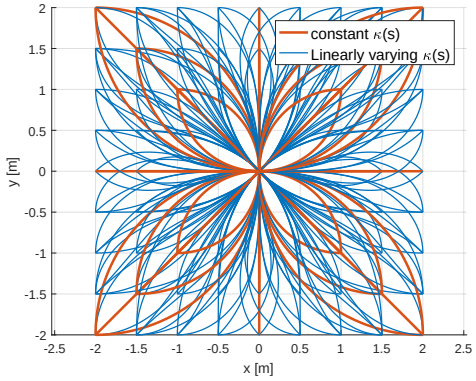
κ', toenemende kromming per eenheid van lengte

$s = 0 : \Delta s : L$, lengte, L, totale lengte curve

Fig. 1a toont de mogelijke bewegingen uit de oorsprong met een specifieke oriëntatie. Fig. 1b toont de bewegingen vanuit de oorsprong ongeacht de initiële oriëntatie. Dit zal de bewegingsprimitiven zijn voor de State Lattice. N.B. enkel een deel van de mogelijke bewegingen zijn hier getoond, de werkelijke discretisatie van de bewegingsprimitive is getoond in tabel 1.



Figuur 1a: Mogelijke bewegingen uit $[0, 0, \pi/2]$



Figuur 1b: Rotatie en spiegeling van fig. 1a.

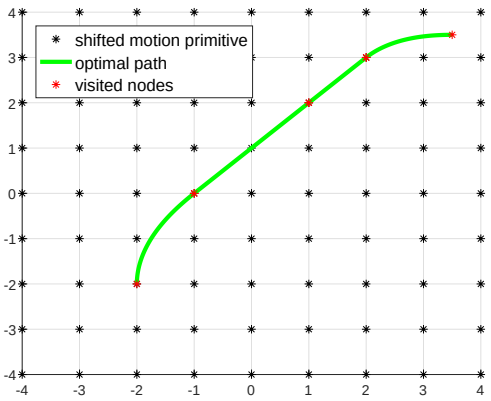
Tabel 1: Overzicht parameters voor bewegingsprimitieven

parameter	waarde
κ_{max}	1 m^{-1}
x_{max}, y_{max}	2 m
dx, dy	0.25 m
$d\theta$	$\pi/8$ rad

State Lattice padplanner

Door de set van bewegingsprimitiven op verschillende discretisatiepunten te plaatsen kan men een (ijle) verbindings-matrix generen. Voorlopig heb ik gebruik gemakt van de standaard graphshortestpath functie van MATLAB, die Dijkstra's methode gebruikt om de set van paden van start naar eindpunt te vinden met de laagste kost. Voorlopig hebben de takken (*E: edges*) tussen de mogelijke knopen (x, y, θ) (*E: vertices*) de volgende kost :

$$edge = \int ds + \frac{1}{3} \int \kappa^2(s) ds \tag{2}$$



Figuur 2: Optimale pad van knoop $[-2, -2, \pi/2]$ naar $[3.5, 3.5, 0]$.

Tabel 2: Overzicht parameters voor State Lattice

parameter	waarde
x_{max}	4 m
y_{max}	4 m
dx	1 m
dy	1 m

De volgende stap is nu om een concrete situatie te implementeren, waar men obstakels toevoegt aan de omgeving. Paden waar de robot tegen een obstakel zou bosten zullen niet meer in de (ijle) verbindings-matrix zijn. Hiervoor zal ik gebruik maken van [2].

"Back-up plan" indien State Lattice slecht schaalt voor fijne beweging

Indien ik merk dat voor fijne bewegingen de State Lattice oplossingsmethode te groot wordt kan ik de volgende oplossing kunnen hanteren :

1. Ik begin terug bij het genereren van bewegingsprimitieven

2. In plaats van één enkele clothoïde kan ik er 2 aan elkaar zetten, dit om te zorgen dat ik een grotere variëteit van beweging kan hebben als fig. 1a. Dit zal als een COP geformuleerd worden.

3. In plaats van enkel een maximale afstand van 2m te gebruiken van de bewegingsprimitieven kan ik deze verhogen naar 4m.

4. Omdat mijn bewegingsprimitieven nu over heel mijn berijk van 4m loopt en een grotere variëteit van beweging hebben, moet ik niet meer gebruik maken van de State Lattice.

Wat ik heb gelezen

- „G1 Fitting with Clothoids" [1].
- „Clothoidal Interpolation — A New Tool for High-Speed Continuous Path Control" [3].
- „The Use of Cornu Spirals in Drawing Planar Curves of Controlled Curvature" [4].
- „G1 Interpolation with a Single Cornu Spiral Segment" [5].

Opmerkingen voor volgende week

- Bespreking en implementeren van "fast collision-checking" met Meneer Demeester [2].
- Conceptuele implementatie van back-up plan.

Referenties

[1] Enrico Bertolazzi en Marco Frego. „G1 Fitting with Clothoids". In: *Math. Meth. Appl. Sci.* 38.5 (30 mrt 2015), p. 881–897. ISSN: 1099-1476. DOI: 10.1002/mma.3114. URL: http://onlinelibrary.wiley.com.kuleuven.ezproxy.kuleuven.be/doi/10.1002/mma.3114/abstract (bezocht op 25-02-2017).

[2] Eric Demeester e.a. „Design and Evaluation of a Lookup-Table Based Collision-Checking Approach for Fixed Sets of Mobile Robot Paths". In: *International Symposium on Robotics* (2012).

[3] H. Makino. „Clothoidal Interpolation — A New Tool for High-Speed Continuous Path Control". In: *CIRP Annals - Manufacturing Technology* 37.1 (1988), p. 25–28. ISSN: 0007-8506. DOI: 10.1016/S0007-8506(07)61578-9. URL: http://www.sciencedirect.com/science/article/pii/S0007850607615789 (bezocht op 25-02-2017).

[4] D. S. Meek en D. J. Walton. „The Use of Cornu Spirals in Drawing Planar Curves of Controlled Curvature". In: *Journal of Computational and Applied Mathematics* 25.1 (jan 1989), p. 69–78. ISSN: 0377-0427. DOI: 10.1016/0377-0427(89)90076-9. URL: http://www.sciencedirect.com/science/article/pii/0377042789900769 (bezocht op 25-02-2017).

[5] D. J. Walton en D. S. Meek. „G1 Interpolation with a Single Cornu Spiral Segment". In: *Journal of Computational and Applied Mathematics* 223.1 (1 jan 2009), p. 86–96. ISSN: 0377-0427. DOI: 10.1016/j.cam.2007.12.022. URL: http://www.sciencedirect.com/science/article/pii/S037704270700670X (bezocht op 25-02-2017).

Masterproef Semi-Autonomous Mobile Robot Navigation in Populated Envirments

Kevin Denis

20 maart 2017

Semester 2, Week 3 - 4 - 5

Bereikt

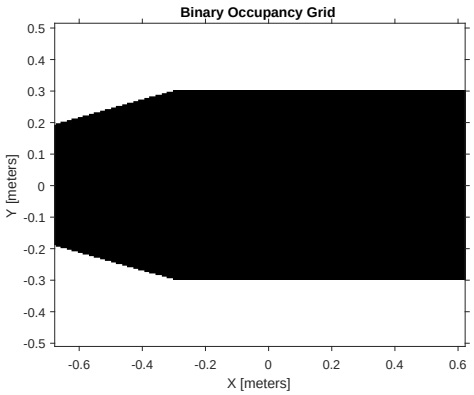
- De plaats die de robot gaat innemen voor elke bewegingsprimitive.
- Bepalen of een pad wel of niet bostingsvrij is.
- Bostingsvrije pad van begin naar eind pose

Kort overzicht

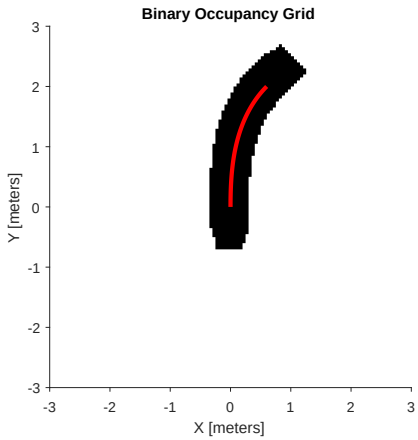
1. Berekenen van bewegingsprimitiven.
2. Berekenen van occupancy grid voor elke pad.
3. Berekenen van de State Lattice (= transleren van bewegingsprimitive + Occupancy grid).
4. Bepalen of een pad wel of niet botsingsvrij is.
5. Bepalen van beste set van vrije paden, gegeven begin en eind-punt.

Occupancy grid bewegingsprimitive

Door de gegeven geometrie van de robot "over" de berekende set van paden uit de bewegingsprimitiven te laten gaan kan men de plaats berekenen die de robot inneemt. fig. 1a en fig. 1b .



Figuur 1a: Occupancy grid van rolstoel op $[0, 0, 0]$

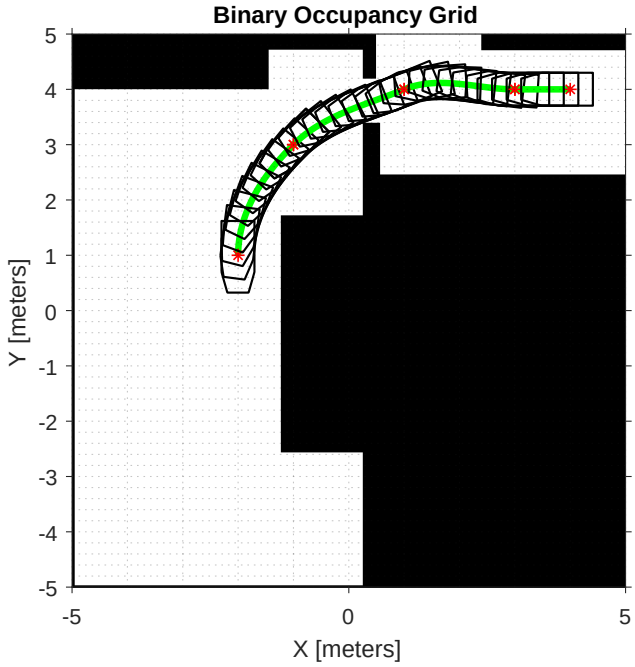


Figuur 1b: Occupancy grid pad + rolstoel

Padplanner

Door de set van bewegingsprimitiven op verschillende discrete punten te plaatsen en de occupancy grid van elke pad te vergelijken met de omgeving kan men een (ijle) botsingsvrije-verbindings-matrix generen. Dit is weergeven in fig. 2.
Voorlopig heb ik gebruik gemakt van de standaard graphshortestpath functie van MATLAB, die Dijkstra's methode gebruikt om de set van paden van start naar eindpunt te vinden met de laagste kost. Voorlopig hebben de takken (*E: edges*) tussen de mogelijke knopen (x, y, θ) (*E: vertices*) de volgende kost :

$$edge = \int ds + \frac{1}{3} \int \kappa^2(s) ds \tag{1}$$



Figuur 2: Optimale pad van knoop $[-2, 1, \pi/2]$ naar $[4, 4, 0]$.

Opmerkingen voor volgende week

- Methodes aanpassen zodat ze beter overeenstemmen met de methodologie die meneer Demeester heeft uitgelegd.
- Doornemen van de documenten die meneer Demeester heeft doorgestuurd.
- Een extra kost toevoegen in functie van de afstand tot een obstacel, om te voorkomen dat deze te dicht komt bij obstakels, zoals in fig. 2.
- Concrete benchmark, waar men ziet dat deze methode een pad vindt, terwijl de methode die enkel steunt op circulaire paden geen pad vindt.

Masterproef Semi-Autonomous Mobile Robot Navigation in Populated Envirments

Kevin Denis

4 april 2017

Semester 2, Week 6 - 7

Bereikt

- State Lattice padplanner aanpassen zodat deze beter overeenstemt met de methodologie die meneer Demeester gebruikt (Local Path Template).
- Sterke vermindering van het aantal gebruikte paden, door het probleem "lokaal" te formuleren. Paden die nu berekend worden zijn altijd direct verbonden met begin pose [0 0 0] van de robot.
- Obstacle-table tabel generatie volgens [2].
- Gebruik van een "multi-size" grid (fijn dichtbij de robot, ruw ver weg)

Kort overzicht Lokale State Lattice padplanner

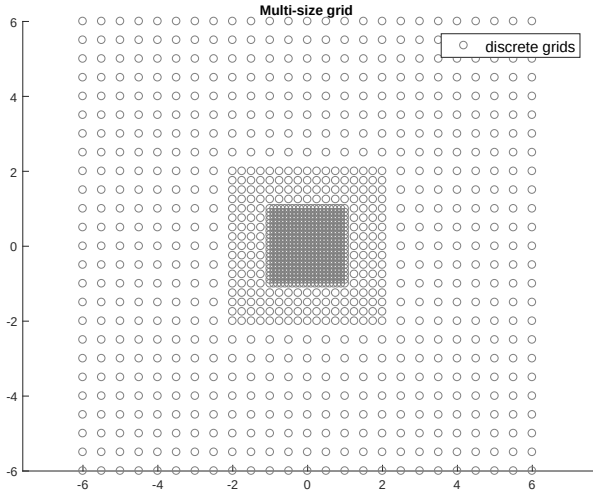
De padplanner die hier beschreven wordt is een synergie tussen [2], [1] en [3].
Deze methode kan gezien worden als een *Lokale State Lattice* padplanner. Deze combineerdt het lokaal (en efficient) karracter van [1] met een grotere vrijheidsgraad (complexere vormen) dankzij [3]. Hieronder een korte beschrijving van de stappen die gevolgd worden door de Lokale State Lattice padplanner.

1. Generen van een multi-size grid. Deze is zeer fijn dicht bij de oorsprong en ruwer ver weg.
2. Clothoïden worden berekend tussen de oorsprong en gridcellen die zich in de Zone Van Interesse (*E: Region Of Interest, ROI*) rond de robot bevinden. Indien deze voldoen aan de randvoorwaarden (kromming kleiner als 1 m^{-1}), worden deze clothoïden toegevoegd aan de set van bewegingsprimitieven.
3. Om een grotere flexibiliteit (en variëteit) van paden te verkrijgen worden er op discrete gridcellen opnieuw een set van bewegingsprimitieven berekend. Dit zorgt ervoor dat bepaalde paden uit twee opeenvolgende clothoïden bestaan.
4. Berekenen van occupancy grid voor elke pad.
5. Bepalen of een pad wel of niet botsingsvrij is.
6. Bepalen van beste set van vrije paden, gegeven gewenste richting / pad van rolstoel gebruiker.

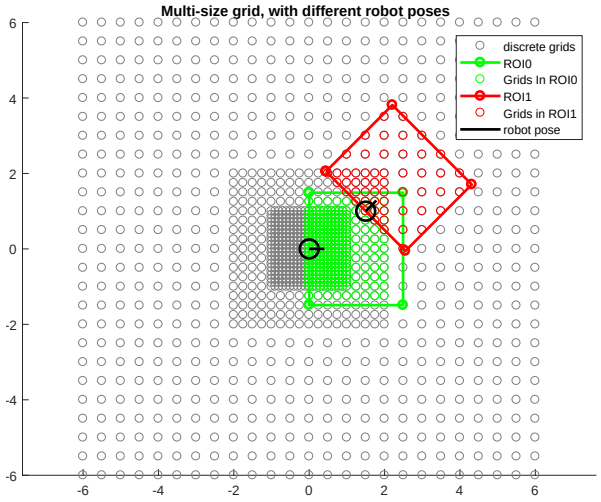
Stap 1 t/m 4 gebeuren offline. 5-6 online.

Multi-size grid en Zone Van Interesse

De eerste stap van de padplanner is het genereren van een discrete grid. Deze bestaat voorlopig uit 3 verschillende groottes : "fijn", "gemiddeld" en "ruw", zie fig. 1. Hoekdiscretistatie ($d\theta$) is voorlopig nog niet positie afhankelijk. Deze zou bvb ook ruwer kunnen worden verder weg van de oorsprong.
Een Zone Van Interesse (ROI) bepaald welke gridcellen verbonden zullen worden met de huidige pose van de robot, zie fig. 2. Aangezien dit een lokale padplanner is, zijn alle stappen berekend volgens "robot coördinaten frame" (= de robot start altijd op pose [0 0 0]). De groene ROI in fig. 2 toont de gridcellen die gebruikt worden voor de bewegingsprimitieven uit [0 0 0]. De rode ROI uit fig. 2 toont de gridcellen die gebruikt worden voor de bewegingsprimitieven uit [1.5 1 45°].
Zie tabel 1 voor concrete waardes van de grid en de ROI.



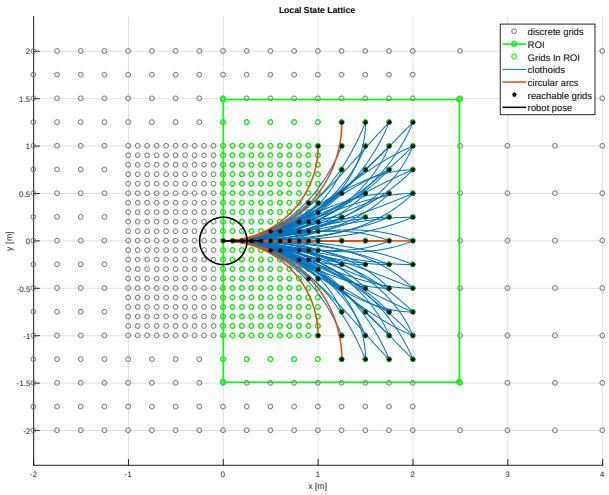
Figuur 1: Multi-size grid



Figuur 2: ROI op positie [0 0 0] en [1.5 1 45°]

Bewegingsprimitieven

Eenmaal de discrete grid en ROI berekend is, tracht de padplanner de oorsprong (=huidige positie van de robot) te verbinden met de gridcellen in de ROI. Indien deze clothoïden voldoen aan de radvoorwaardes worden deze toegevoegd aan de set van bewegingsprimitieven. Dit is geïllustreerd in fig. 3.



Figuur 3: Bewegingsprimitieven uit de oorsprong, gegeven een grid en ROI.

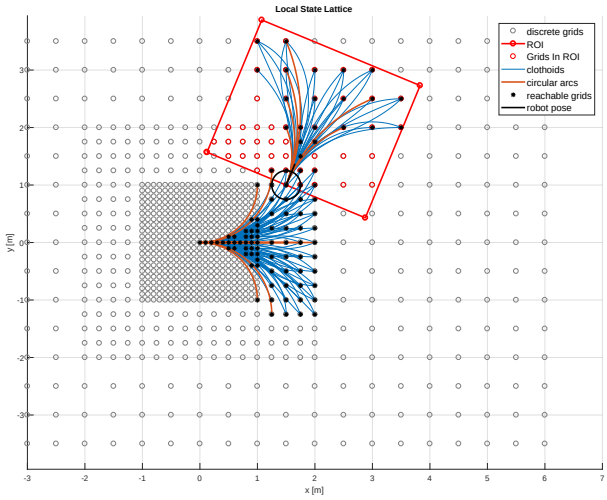
Tabel 1: Waardes van de verschillende parameters met korte beschrijving

par.	waarde	beschrijving
$dx1$	0.1m	Afstand tussen twee gridcellen in de "fijne" grid.
$x1_{max}$	1m	Maximale afstand (L_∞) tot de oorsprong van de "fijne" grid.
$dx2$	0.25m	Afstand tussen twee gridcellen in de "gemiddelde" grid.
$x2_{max}$	2m	Maximale afstand (L_∞) tot de oorsprong van de "gemiddelde" grid.
$dx3$	0.5m	Afstand tussen twee gridcellen in de "ruwe" grid.
$dx3_{max}$	6m	Maximale afstand (L_∞) tot de oorsprong van de "ruwe" grid.
$d\theta$	$\pi/8$ rad	Hoek discretisatie, voorlopig nog niet positie afhankelijk.
dx_{SL}	0.5m	State lattice positie als afstand (L_1) van gridcel veelvoud van dx_{SL} is.
x_{ROI}	2.49m	Afstand in de x-richting voor de Zone Van Interesse, enkel +.
y_{ROI}	1.49m	Afstand in de y-richting voor de Zone Van Interesse, + en -.
κ_{max}	1 m^{-1}	Maximale kromming clothoïde.

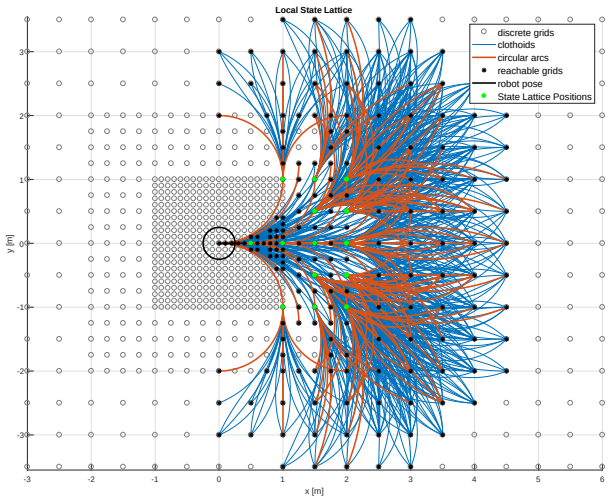
De waardes van x_{ROI} en y_{ROI} zijn zo gekozen dat de randen *net geen* state lattice positie zijn (net geen veelvoud van dx_{SL}), zie fig. 3 voor alle duidelijkheid. Dit zodat er minder paden (te) ver van de oorsprong gelegen zijn.

Lokale State Lattice

Om een grotere variëteit van paden te generen, gaat de planner op verschillende, discrete eindposities (= "State Lattice" posities) van de set van bewegingsprimitieve uit de oorsprong, opnieuw een set van bewegingsprimitieven berekenen, zie fig. 4. N.B. dankzij de multi-size grid zullen bewegingsprimitieve State Lattice posities ver van de oorsprong minder paden generen. Dit process word herhaald voor elke State Lattice posities, zie fig. 5. Deze paden vormen de volledige set van paden die de Lokale State Lattice padplanner tot zijn beschikking heeft.



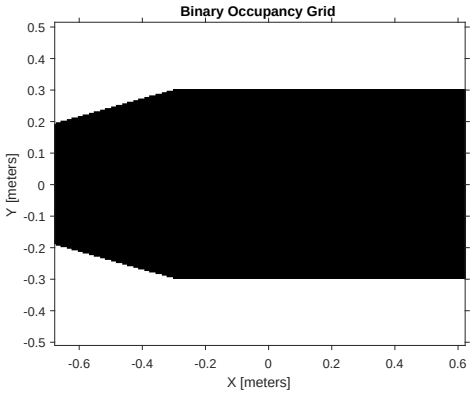
Figuur 4: Bewegingsprimitieven uit [1.5 1 45°], gegeven een grid en ROI.



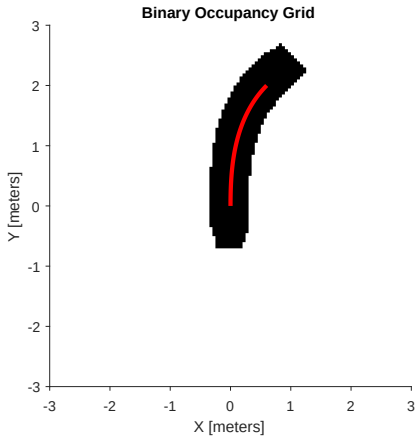
Figuur 5: Volledige set van paden uit de oorsprong.

Occupancy grid bewegingsprimitive en selectie van botsingsvrije paden

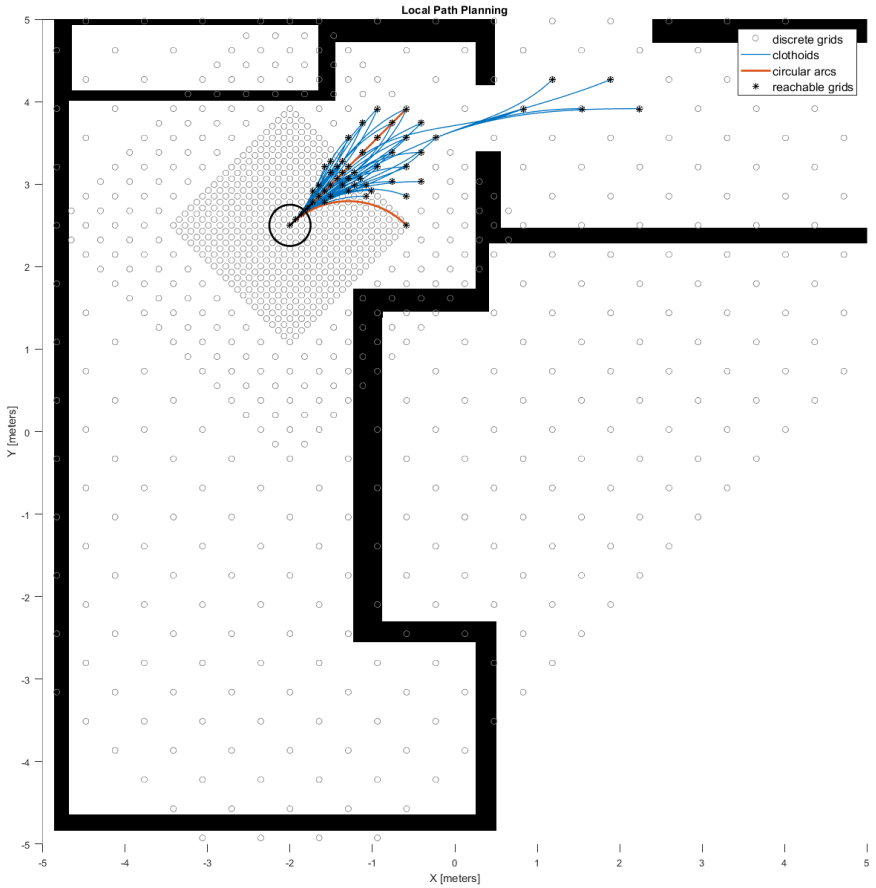
Door de gegeven geometrie van de robot "over" de berekende set van paden te laten gaan kan men de plaats berekenen die de robot inneemt. fig. 6a en fig. 6b .
De stappen die tot nu toe uitgevoerd zijn kunnen offline gebeuren, zodat men enkel online moet bepalen of een pad wel of niet botsingsvrij is. Gegeven de omgeving (geformuleerd volgens de robot coordinaten frame !), kan de padplanner de botsingsvrije paden selecteren, zie fig. 7.



Figuur 6a: Occupancy grid van rolstoel op $[0, 0, 0]$



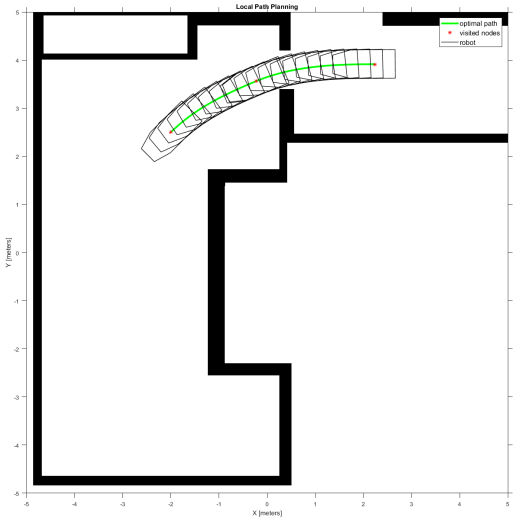
Figuur 6b: Occupancy grid pad + rolstoel



Figuur 7: Vrije paden vanuit huidige positie robot.

Padplanner

De laatste stap is het bepalen van de set van paden (max 2 !) die het beste overeenkomen met het pad of richting die de gebruiker van de rolstoel wilt nemen. Als voorbeeld wordt er hier getracht door een nauwe deur te gaan, zie fig. 8.
Deze plattegrond zou het robotlabo van werktuigkunde moeten voorstellen. Dankzij de plannen die Johan mij vorig weekend heeft gestuurd zal ik in staat zijn om een iets betere plattegrond te maken in een volgende "benchmark".



Figuur 8: Optimale pad van knoop $[-2, 1, \pi/2]$ naar $[4, 4, 0]$.

Opmerkingen voor volgende week

- Literatuur doornemen over Human-aware robot navigation, Proxemics, Navigating a Robotic Wheelchair in a Railway Station during Rush Hour.
- Methode om efficient een tabel/template te maken, voor het bepalen van de kleinste afstand van een obstakel tot een pad, en dit gebruiken als een extra kost.
- Concrete benchmark, waar men ziet dat deze methode een pad vindt, terwijl de methode die enkel steunt op circulaire paden geen pad vindt.

Referenties

[1] Eric Demeester, Marnix Nuttin en Hendrik Van Brussel. „Fine Motion Planning for Shared Wheelchair Control: Requirements and Preliminary Experiments". In: *Proceedings of the 11th International Conference on Advanced Robotics*. Coimbra, Portugal, jun 2003.

[2] Eric Demeester e.a. „Design and Evaluation of a Lookup-Table Based Collision-Checking Approach for Fixed Sets of Mobile Robot Paths". In: *International Symposium on Robotics* (2012).

[3] Mihail Pivtoraiko en Alonzo Kelly. „Efficient Constrained Path Planning Via Search In State Lattices". In: *Proc. of 'The 8th International Symposium on Artifical Intelligence, Robotics and Automation in Space*. International Symposium on Artifical Intelligence, Robotics and Automation in Space. Munich, Germany., —5–8 sep 2005.