به نام خداوند جان و خرد

برنامه ریزی حرکت روبات

علی خالق زادگان

دانشگاه یزد
زمستان ۱۳۹۳

چشم‌انداز

۱ مقدمه

شکل ۱: ربات انسان نمای آسیمو ساخت شرکت هوندا

"In 2050, a football team composed of fully-autonomous humanoid robots will play against the human winners of the World Cup."

Prof. Alexandre da Sliva Simoes
www.bbc.com

یکی از اهداف نهایی در روباتیک، طراحی روبات های خودکار است.
روبات‌هایی که به آن‌ها بگویید چه کاری را انجام دهند بدون آنکه مجبور باشید چگونگی انجام آن‌کار را برای آن‌ها بیان کنید.

یک روبات خودکار باید قادر باشد حرکت خود را برنامه ریزی کند.
برای این منظور ربات باید برخی اطلاعات از جمله موقعیت موانع موجود در مورد محیط اطراف خود را داشته باشد.

شکل ۲: یک ربات مفصل دار

یک نوع دیگر از روبات ها، روبات مفصل دار است که شامل یک بازوی روباتی است.

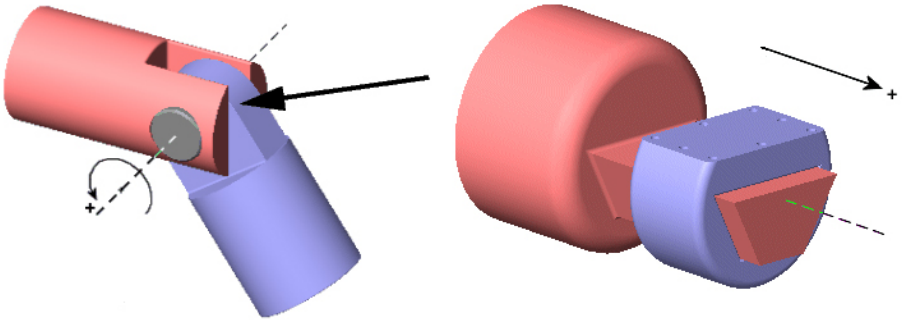
هر بازوی روباتی شامل چندین اتصال است بطوریکه قسمت پایه آن محکم به زمین چسبیده شده و در انتهای دیگر یک دست یا برخی ابزار مورد نیاز دیگر قرار دارد.

این ربات ها بیشتر برای سرهم کردن اجزای یک شی، اسپری کردن و یا جوشکاری به کار می روند. پس باید قادر باشند از یک موقعیت به موقعیت دیگر حرکت کنند بدون آنکه با محیط اطراف یا شیای که روی آن کار میکنند و یا حتی با خودشان برخورد داشته باشند.

انواع مفصل برای ربات های مفصل‌دار:

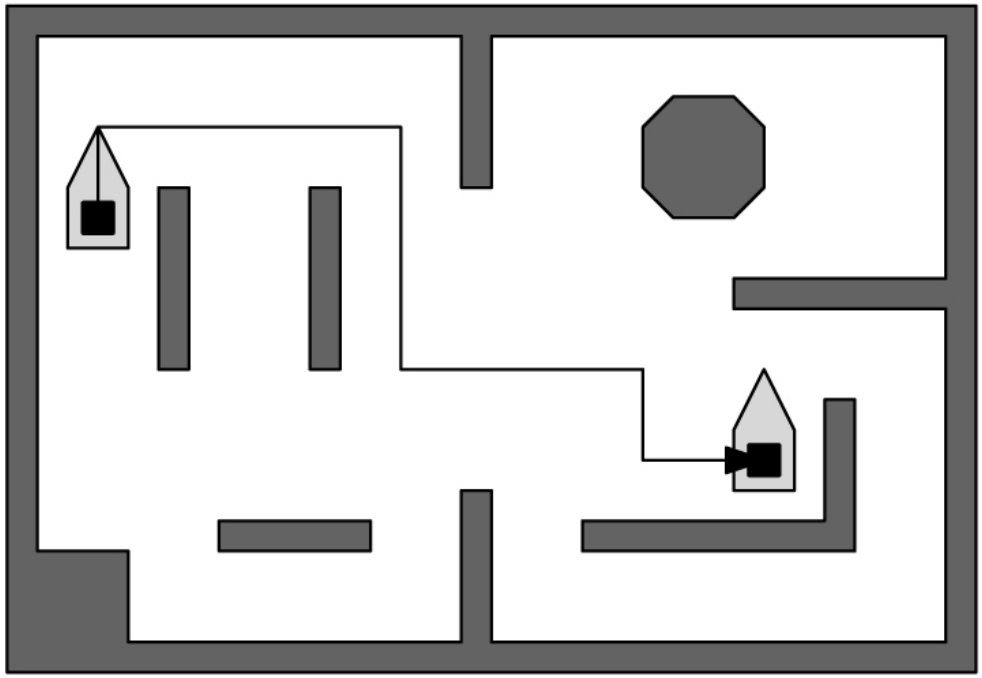
۱- نوع پیچشی: که به اتصال اجازه می‌دهد که حول محور خود گردش کند. بسیار شبیه به آرنج دست انسان

۲- نوع منشوری: که به یکی از دو اتصال درگیر اجازه می‌دهد به صورت کشویی به بیرون و داخل حرکت کند.



در ادامه مفروضات زیر را در نظر می‌گیریم

۱- روبات در یک محیط دو بعدی حرکت می‌کند. پس محیط اطراف، موانع و حتی خود روبات را به صورت چند ضلعی در نظر می‌گیریم.
۲- محیط اطراف روبات استاتیک است. به عبارت دیگر موانع ثابت و بدون حرکت هستند.
۳- روبات در هر جهت دلخواهی می‌تواند حرکت کند اما فقط به صورت انتقالی و مجاز به دوران حول خودش نمی باشد.
۴- موانع را به صورت مجموعه هایی باز در نظر می‌گیریم در این صورت ربات مجاز است موانع را لمس کند اما با توجه به اینکه چنین برخوردهایی ممکن است باعث بروز خطا در سیستم شود، موانع را کمی بزرگتر در نظر می‌گیریم.

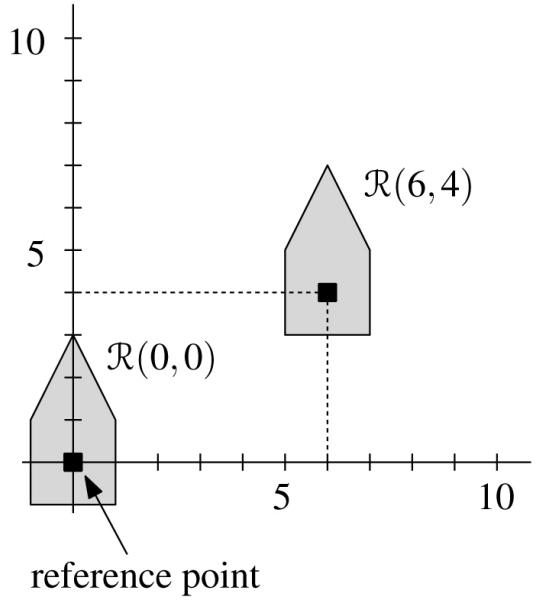


۲ فضاهای کاری و پیکربندی

فضای کاری

فرض کنید R یک ربات به شکل یک چندضلعی در یک فضای دو بعدی باشد و $S = \{p_1, p_2, \ldots, p_n\}$ یک مجموعه از n مانع به شکل چندضلعی باشد.
به محیط واقعی که ربات در آن فعالیت می‌کند فضای کاری می‌گوییم.

باتوجه به اینکه حرکت ربات تنها به صورت انتقالی است پس یک گمارش از ربات با انتقال همه راسهای آن ربات تحت بردار (x, y) بدست می‌آید و با $R(x, y)$ نشان داده می‌شود.می‌توان ربات چندضلعی R را با لیست کردن راس های $R(\circ, \circ)$ به صورت ساعتگرد یا پادساعتگرد مشخص کرد!



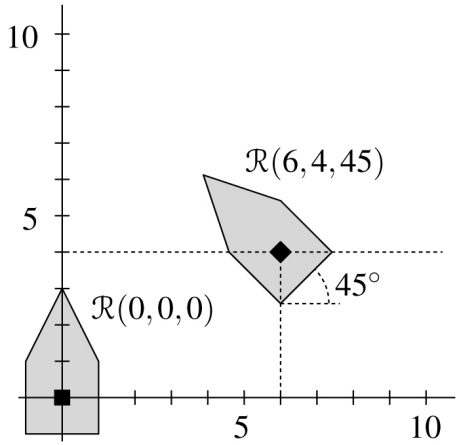
گمارش $R(\circ,\circ)$ از ربات R را در نظر بگیرید
نقطه‌ای از ربات که در مکان $(\circ,\circ)$ قرار می‌گیرد را نقطه مرجع ربات می‌نامیم.
اگر نقطه مرجع خارج از ربات قرار داشته باشد فرض می‌کنیم که توسط خطی نامریی به ربات متصل شده است.

گمارش $R(x,y)$ به این معنا است که
نقطه مرجع ربات چندضلعی R در نقطه (x,y) قرار داشته باشد.

درجه آزادی

فرض کنید ربات چند ضلعی R علاوه بر حرکت انتقالی بتواند جهت حرکتش را با کمک دوران حول نقطه مرجع خود تغییر دهد.
در این صورت به غیر از دو مولفه x و y به یک مولفه اضافه ϕ برای مشخص کردن جهت چرخش روبات نیاز خواهیم داشت. به عبارت دیگر $R(x, y, \phi)$

تعداد پارامترهای لازم برای تعیین یک گمارش از ربات، درجه آزادی ربات نام دارد.



فضای پیکر بندی

فضای پارامتری یک ربات R را فضای پیکربندی ربات می‌گوییم و با $C(R)$ نشان می دهیم.

هر نقطه p در فضای پیکربندی متناظر با یک گمارش خاص $R(p)$ از ربات در فضای کاری است.

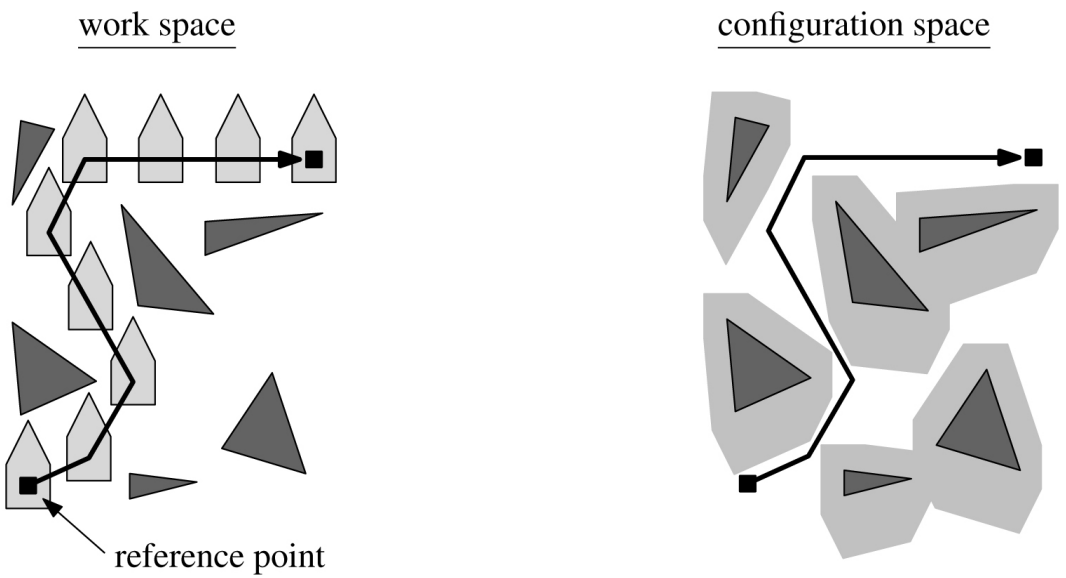
به عنوان مثال برای یک ربات مسطح که حرکت انتقالی و دورانی داشته باشد یک نقطه (x, y, ϕ) در فضای پیکربندی متناظر با گمارش $R(x, y, \phi)$ در فضای کاری است. برای این ربات فضای پیکربندی به صورت $R^2 \times [0 : 360)$ است که توپولوژی آن همانند یک سیلندر است.

توجه توجه

تنها برای یک ربات مسطح با حرکت انتقالی فضای پیکر بندی با فضای کاری یکسان است و به صورت یک فضای اقلیدسی دو بعدی است.
اما نباید این دو دو فضا را با همدیگر اشتباه گرفت:
فضای کاری همان محیط واقعی است که ربات در آن فعالیت می‌کند در حالیکه فضای پیکربندی فضای پارامتری ربات است.

نتیجه

یک ربات چندضلعی در فضای کاری معادل با یک نقطه در فضای پیکربندی است و برعکس هر نقطه در فضای پیکربندی معادل با یک گمارش از ربات در محیط واقعی است.



شکل ۳: یک مسیر در فضای کاری و منحنی متناظر آن در فضای کاری

فضای پیکربندی ممنوعه

بعضی از نقاط در فضای پیکربندی متناظر با یک گمارش از ربات هستند که در آن محل ربات با موانع S برخورد می‌کند و بنابراین رخ دادن آن‌ها غیر ممکن است.
قسمتی از فضای پیکربندی حاوی مجموعه نقاط غیرممکن را فضای پیکربندی ممنوعه یا به اختصار فضای ممنوعه می‌نامیم و با $C_{frob}(R,S)$ نشان می‌دهیم.

فضای پیکربندی آزاد

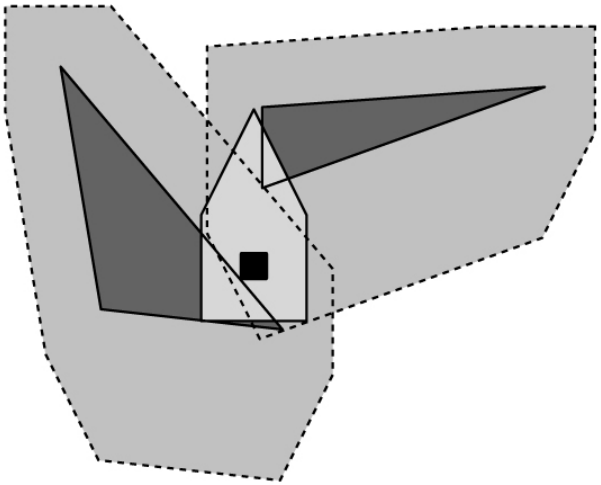
قسمت با قیمانده فضای پیکربندی که متناظر با گمارش های آزاد یا ممکن ربات است را فضای پیکربندی آزاد یا به اختصار فضای آزاد می‌نامیم و با $C_{free}(R,S)$ نشان می‌دهیم.
به عبارت دیگر $C_{free}(R,S) = C(R) - C_{frob}(R,S)$

فضای پیکربندی مانع

با توجه به اینکه همه نقاط در فضای پیکربندی ممکن نیستند
ما یک مانع O در فضای کاری را به مجموعه‌ای از نقاط مانند p در فضای پیکربندی که در آن‌ها گمارش $R(p)$ ربات با مانع O برخورد دارد، نگاشت می‌کنیم
و به صورت خلاصه با $C_{obstacle}$ نمایش می‌دهیم.

توجه

ممکن است موانع $C_{obstacle}$ با همدیگر همپوشانی داشته باشند حتی اگر در فضای کاری موانع مجزا باشند و این حالت زمانی رخ می‌دهد که در یک زمان ربات با بیش از یک مانع برخورد کند.



۳ ربات نقطه ای

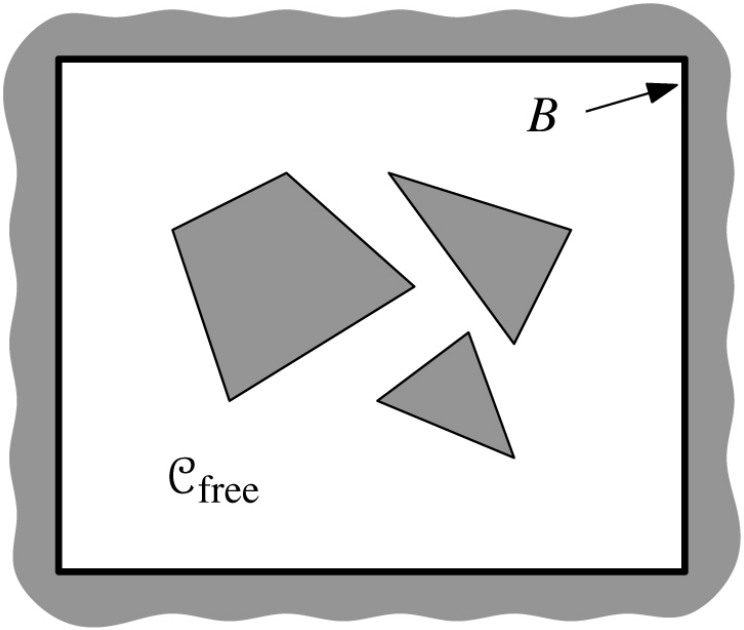
در ادامه حرکت یک ربات نقطه‌ای را برنامه‌ریزی می‌کنیم.

مفروضات

۱- ربات R به همراه موانع $p_1, p_2, \ldots, p_t$ که چند ضلعی هایی مجزا هستند داده شده است به عبارت دیگر داخل موانع با همدیگر برخورد ندارد.
۲- حرکت ربات داخل یک جعبه محدود کننده به اندازه بزرگ B که حاوی همه موانع است انجام می‌گیرد.
۳- تعداد کل راس های موانع برابر n است.

توجه کنید برای یک ربات نقطه‌ای فضای کاری و فضای پیکر بندی یکسان است.

در ادامه داده ساختاری را برای نمایش فضای پیکربندی آزاد ربات $(C_{free} = B - \cup p_i)$ محاسبه خواهیم کرد و به کمک آن یک مسیر بین هر دو نقطه را بدست می آوریم.
اما ابتدا همه موانع را داخل یک جعبه محدود کننده قرار می‌دهیم.



روشی که در ادامه بحث خواهد شد برای زمانی مفید است که فضای کاری ربات به صورت استاتیک بوده و هدف محاسبه مسیرهای زیادی بین هر دو نقطه در فضای کاری باشد.

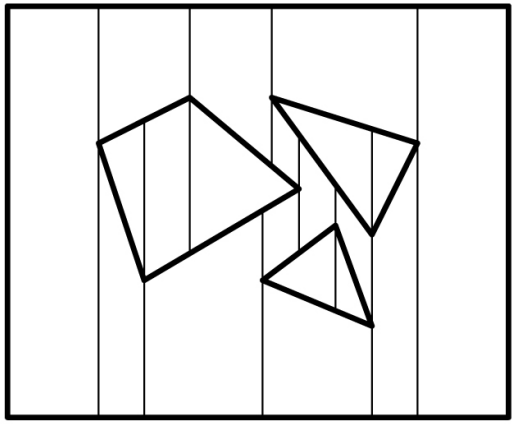
الگوریتم زیر با کمک الگوریتم ارایه شده در فصل ۶، نقشه ذوزنقه‌ای فضای پیکربندی آزاد روبات را که با $T(C_{free})$ نشان می‌دهیم، محاسبه می‌کند.

Algorithm COMPUTEFREESPACE(S)
Input. A set S of disjoint polygons.
Output. A trapezoidal map of $\mathcal{C}_{\text{free}}(\mathcal{R}, S)$ for a point robot $\mathcal{R}$.
1. Let E be the set of edges of the polygons in S.
2. Compute the trapezoidal map $\mathcal{T}(E)$ with algorithm TRAPEZOIDALMAP described in Chapter 6.
3. Remove the trapezoids that lie inside one of the polygons from $\mathcal{T}(E)$ and return the resulting subdivision.

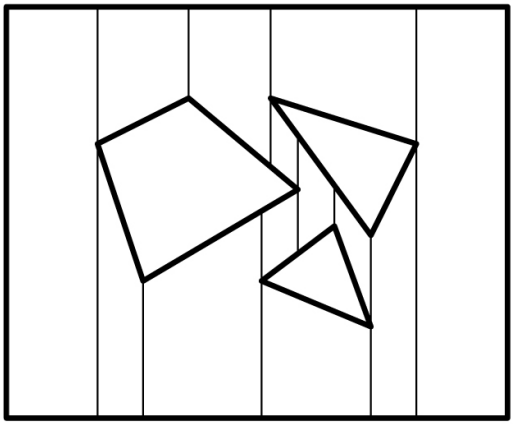
پرسش

چگونه می‌توان ذوزنقه‌های داخل موانع را بدست آورد؟

(a)



(b)



پاسخ

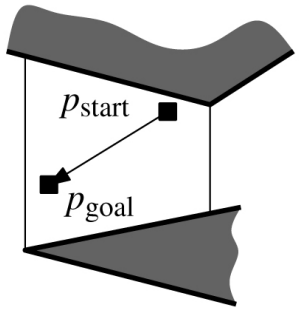
در صورتی که برای هر ذوزنقه Δ دو شرط زیر برقرار باشد آن ذوزنقه حذف نخواهد شد:
۱- می‌دانیم فیلد $top(\Delta)$ به یکی از اضلاع موانع اشاره می‌کند.
با حرکت ساعتگرد روی آن ضلع، مانع در یک سمت و ذوزنقه در سمت دیگر باشد.
۲- می‌دانیم فیلد $bottom(\Delta)$ به یکی از اضلاع موانع اشاره می‌کند.
با حرکت ساعتگرد روی آن ضلع، مانع در یک سمت و ذوزنقه در سمت دیگر باشد.
پس کافی است همه ذوزنقه ها را یکبار بررسی کنیم!

لم ۱۳.۱

نقشه ذوزنقه‌ای فضای پیکربندی آزاد یک روبات نقطه‌ای که در بین مجموعه‌ای از موانع با مجموع n یال، حرکت می‌کند با کمک یک الگوریتم تصادفی در زمان موردانتظار $O(nlogn)$ قابل محاسبه است.

روند محاسبه یک مسیر (در صورت وجود)
از نقطه شروع P_{start} به نقطه پایان P_{goal} با استفاده از $T(C_{free})$

اگر هر دو نقطه شروع و پایان در یک ذوزنقه قرار داشته باشند
کافی است ربات در یک خط مستقیم به نقطه پایان حرکت کند.
اما در غیر این صورت
مسیر موردنظر از تعدادی ذوزنقه عبور خواهد کرد.
در این حالت با ساخت یک گراف مسطح جاسازی شده در فضای پیکربندی آزاد که آنرا نقشه راه می‌نامیم و با G_{road} نشان می‌دهیم، حرکت روبات را برنامه‌ریزی می‌کنیم.



محاسبه نقشه راه

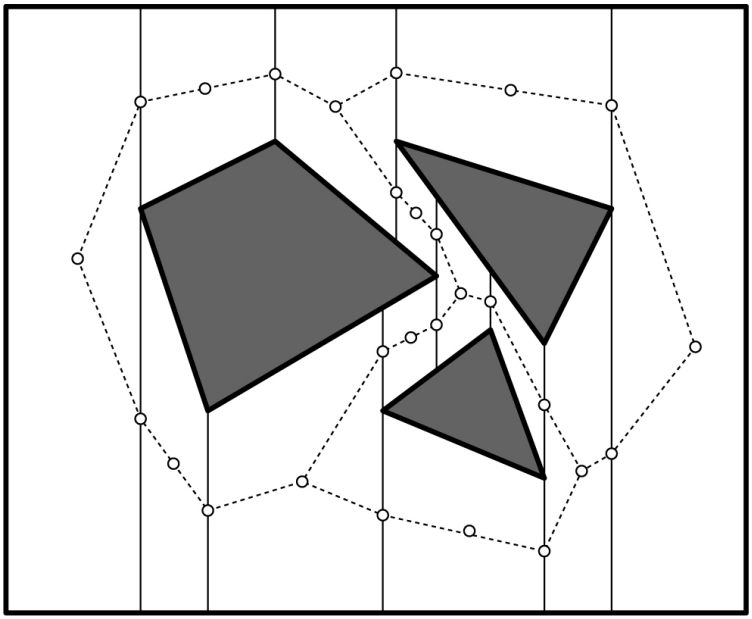
گره‌های G_{road}

یک گره در مرکز هر ذوزنقه قرار می‌دهیم.
یک گره در وسط هر گسترش عمودی قرار می‌دهیم.

یال‌های G_{road}

بین هر دو گره یک یال به صورت خطی مستقیم قرار می‌دهیم اگفگر یکی از گره‌ها در مرکز یک ذوزنقه و دیگری روی مزر همان ذوزنقه باشد.

نقشه راه را می‌توان در زمان $O(n)$ با پیمایش همه ذوزنقه‌های ساختار $DCEL$ متناظر با $T(C_{free})$ محاسبه کرد.



شکل ۴: نقشه راه متناظر با نقشه ذوزنقه‌ای فضای آزاد

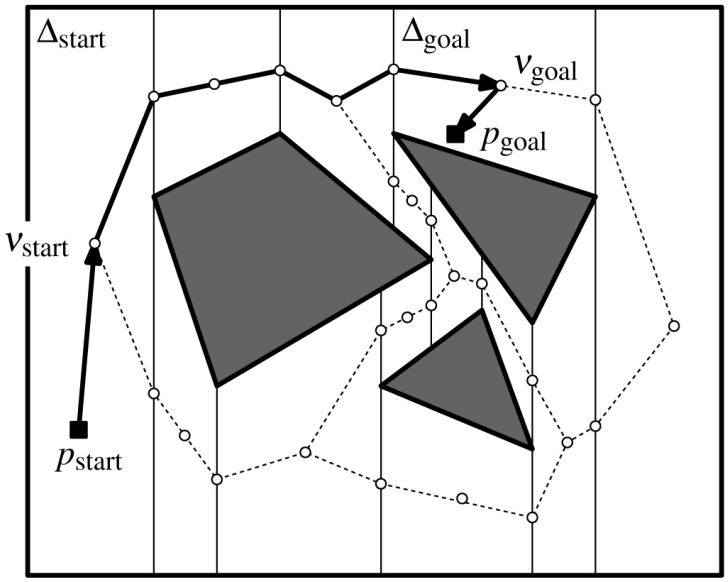
الگوریتم زیر با توجه به نقشه راه، یک مسیر بدون برخورد اما نه لزوما یک کوتاهترین میسر را از نقطه شروع به نقطه پایان محاسبه می‌کند.

نمادهای تعریف شده

۱- Δ_{start} ذوزنقه‌ی حاوی نقطه شروع p_{start} و Δ_{goal} ذوزنقه حاوی نقطه پایان p_{goal} است.
۲- v_{start} و v_{goal} راس‌هایی از نقشه راه هستند که به ترتیب در مرکز ذوزنقه‌های Δ_{start} و Δ_{goal} قرار دارند.

Algorithm COMPUTEPATH($\mathcal{T}(\mathcal{C}_{\text{free}}), \mathcal{G}_{\text{road}}, p_{\text{start}}, p_{\text{goal}}$)
Input. The trapezoidal map $\mathcal{T}(\mathcal{C}_{\text{free}})$ of the free space, the road map $\mathcal{G}_{\text{road}}$, a start position p_{start}, and goal position p_{goal}.
Output. A path from p_{start} to p_{goal} if it exists. If a path does not exist, this fact is reported.

1. Find the trapezoid Δ_{start} containing p_{start} and the trapezoid Δ_{goal} containing p_{goal}.
2. **if** Δ_{start} or Δ_{goal} does not exist
3. **then** Report that the start or goal position is in the forbidden space.
4. **else** Let v_{start} be the node of $\mathcal{G}_{\text{road}}$ in the center of Δ_{start}.
5. Let v_{goal} be the node of $\mathcal{G}_{\text{road}}$ in the center of Δ_{goal}.
6. Compute a path in $\mathcal{G}_{\text{road}}$ from v_{start} to v_{goal} using breadth-first search.
7. **if** there is no such path
8. **then** Report that there is no path from p_{start} to p_{goal}.
9. **else** Report the path consisting of a straight-line motion from p_{start} to v_{start}, the path found in $\mathcal{G}_{\text{road}}$, and a straight-line motion from v_{goal} to p_{goal}.



شکل ۵: یک مسیر محاسبه شده از نقشه راه

اثبات درستی

۱- مسیری که توسط الگوریتم گزارش می‌شود خالی از برخورد است (چرا؟)
ذوزنقه‌هایی که براساس آن‌ها نقشه راه بدست آمده است در فضای آزاد قرار دارند و یال‌های نقشه راه به صورت پاره‌خط‌هایی داخل ذوزنقه‌هاست.

۲- در صورت وجود مسیری خالی از برخورد الگوریتم آن مسیر را حتما پیدا می‌کند(چرا؟)
فرض کنید یک مسیر خالی از برخورد از p_{start} به p_{goal} وجود داشته باشد.
کافی است ثابت کنیم یک مسیر از v_{start} به v_{goal} وجود دارد. فرض کنید $\Delta_1 = \Delta_{start}, \Delta_2, \ldots, \Delta_k = \Delta_{goal}$ دنباله‌ای از ذوزنقه‌هایی باشد که مسیر از آن‌ها عبور می‌کند.
فرض کنید v_i گره‌ای از G_{road} باشد که در مرکز ذوزنقه Δ_i قرار دارد.
اگر مسیر از Δ_i به Δ_{i+1} حرکت کند در این صورت این دو ذوزنقه با یکدیگر همسایه هستند پس حتما یک گسترش عمودی مشترک بین آن‌ها وجود دارد، اما در نقشه راه، گره‌های معادل چنین ذوزنقه‌هایی از طریق گره‌ای که روی مرز مشترک آن‌ها قرار دارد به هم متصل شده‌اند، پس مسیری از v_i به v_{i+1} در نقشه راه وجود دارد.
درنتیجه حتما از v_1 به v_k مسیری وجود دارد!

تحلیل زمان اجرا

با توجه به اینکه فضای آزاد به صورت نقشه ذوزنقه‌ای است پس پیدا کردن Δ_{start} و Δ_{goal} در زمان $O(logn)$ امکان‌پذیر است.

با توجه به اینکه پیچیدگی گراف G_{road} از مرتبه $O(n)$ است پس انجام جست‌وجوی اول-سطح در زمان $O(n)$ امکان‌پذیر است.

قضیه ۱۳.۲

فرض کنید R یک ربات نقطه‌ای باشد که در بین یک مجموعه S از موانع چند ضلعی با تعداد کل n راس حرکت می‌کند. می‌توان S را در زمان مورد انتظار $O(nlogn)$ به گونه‌ای پیش‌پردازش کرد که در زمان خطی $O(n)$ یک مسیر بدون برخورد (درصورت وجود) بین هر نقطه شروع و پایان قابل محاسبه باشد.

یکی درد و یکی درمان پسندد
یکی وصل و یکی هجران پسندد

من از درمان و درد و وصل و هجران
پسندم آنچه را جانان پسندد
باباطاهر