Path-Velocity Decomposition

Bericht

Inhalt

1 Überblick

Die Wegplanung, um von einem Punkt zu einem anderen zu gelangen, ist für uns Menschen ein alltägliches und intuitiv lösbares Problem. Dabei umgehen wir ohne Schwierigkeiten Hindernisse und weichen anderen bewegten Objekten aus. Für autonome Roboter ist die kollisionsfreie Bewältigung der Wegplanung genauso essentiell wie für uns, um sich in einer dynamischen Umgebung zu bewegen.

Im Gegensatz zu einer statischen Umgebung, in denen sich keine Hindernisse bewegen, ist hier auch die zeitliche Komponente zu berücksichtigen. Beim *Trajectory Planning Problem* (TPP) gilt es eine Bahn zu bestimmen, zu der sich ein Aufenthaltspunkt zu einem gegeben Zeitpunkt angeben lässt.

Ziel ist es, einen Schwarm aus Robotern durch ein Schwarmbetriebssystem zu organisieren und zu verwalten. Eine Kernaufgabe eines Betriebssystems ist das sogenannte Scheduling von verschieden Anwendungen bzw. Prozessen. Im Falle des Schwarmbetriebssystems muss neben dieser klassischen Zeitplanung auch eine Raumplanung durchgeführt werden. Das heißt, die Roboter müssen sich zu einem bestimmten Zeitpunkt an einem bestimmten Ort befinden.

Zur Lösung des TTPs haben (Kant & Zucker, 1986) vorgeschlagen, das Problem in zwei Teile aufzuspalten. Der erste Teil umfasst das statische *Path Planning Problem* (PPP), bei dem bewegte Hindernisse nicht berücksichtigt werden. Der zweite Teil ist das *Velocity Planning Problem* (VPP).

Hierbei werden Kollisionen mit dynamischen Hindernissen durch Anpassung der Geschwindigkeit vermieden.

2 Lösungsansatz

Zur Lösung des TPPs spaltet man das Problem in zwei Teilprobleme auf: Das PPP und das VPP. Beim PPP werden alle zeitlichen Aspekte vernachlässigt. Stattdessen plant man einen Weg zwischen dem Start- und Endpunkt, welcher alle statischen Objekte vermeidet. Dieser Weg ist die Lösung des PPPs.

Die Lösung des VPPs ist die Bestimmung der Geschwindigkeit entlang dieses Weges, und zwar so, dass man sich nie zum selben Zeitpunkt an einer Stelle auf dem Weg befindet, wie ein anderes dynamisches Hindernis[1].

Die Roboter bewegen sich im zweidimensionalen Raum. Einfachheitshalber wird angenommen, dass die Roboter die Größe eines Punktes besitzen. Die Hindernisse werden durch Polygone dargestellt. Bewegte Hindernisse sind ebenfalls Polygone die sich entlang eines Polygonzugs bewegen.

2.1 Path Planning

Im zweidimensionalen Raum ist der kürzeste Pfad zwischen zwei Punkten, welcher keine Polygone kreuzt, zusammengesetzt aus Strecken, die eine Teilmenge der Eckpunkte der Hindernisse verbindet (Wangdahl, Pollock, & Woodward, 1974) (Lozano-Perez, 1980). Der Pfad verläuft also von Eckpunkt zu Eckpunkt (siehe Abbildung 2.1).

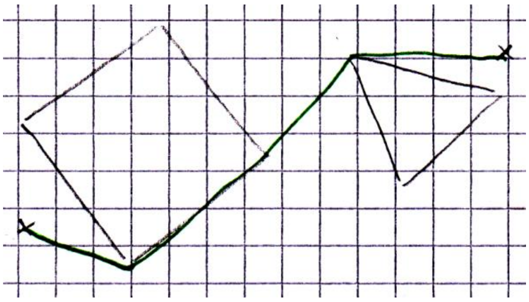

Abbildung 2.1: Ein Pfad entlang der Polygoneckpunkte

Zur Bestimmung dieses Pfades bildet man einen Graph, der den Startpunkt I, den Endpunkt F und die Eckpunkte V der Hindernisse als Knoten enthält. Die Menge der Kanten des Graphs umfasst jene Knoten, die zueinander sichtbar sind (siehe Abbildung 2.2). Das heißt, ihre Sichtlinie wird nicht durch Hindernisse blockiert.

[1] Zu einer Lösung kann es natürlich nur dann kommen, wenn jeder Streckenabschnitt zumindest zeitweise auch freigegeben, also nicht durch ein bewegtes Hindernis blockiert, ist. Eine maximale Geschwindigkeit schränkt die Passierbarkeit noch zusätzlich ein.

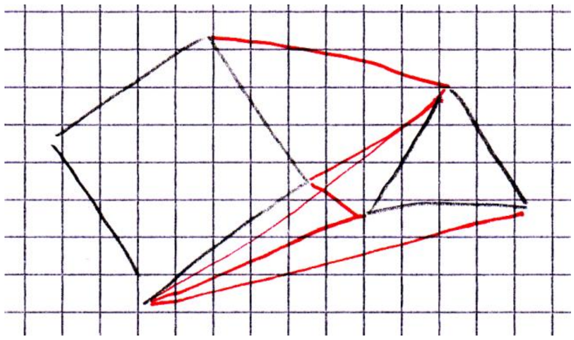

Abbildung 2.2: Sichtlinien zwischen Polygonen

2.1.1 Sichtbarkeit

Die Prüfung auf Sichtbarkeit erfolgt durch Ermittlung von Geradenschnittpunkten. Zwei Punkte P_1 und P_2 sind dann zueinander sichtbar, wenn ihre Strecke $s = \overline{P_1P_2}$ keinen Polygonabschnitt u schneidet.

Die Strecken lassen sich mathematisch folgendermaßen ausdrücken:

$$s: \boldsymbol{x} = \boldsymbol{P_1} + t_s \cdot (\boldsymbol{P_2} - \boldsymbol{P_1})$$
$$u: \boldsymbol{x} = \boldsymbol{Q_1} + t_u \cdot (\boldsymbol{Q_2} - \boldsymbol{Q_1})$$

Wobei Q_1 und Q_2 die Eckpunkte des Streckenabschnitts u sind. Bei t_s und t_u handelt es sich um die Parametervariablen die einen Wert im Intervall $[0,1]$ annehmen können.

Aus diesen Gleichungen lässt sich ein Gleichungssystem konstruieren, dessen Lösung zum Schnittpunkt der Strecken führt:

$$[\boldsymbol{P_2} - \boldsymbol{P_1} \quad \boldsymbol{Q_2} - \boldsymbol{Q_1}] \begin{bmatrix} -t_s \\ t_u \end{bmatrix} = \boldsymbol{P_1} - \boldsymbol{Q_1}$$

Im Falle zweier nicht paralleler Strecken liegt genau dann ein Schnittpunkt vor, wenn sich sowohl t_s als auch t_u im Intervall $[0,1]$ befinden.

Falls die Strecken parallel sind, so lässt sich kein Schnittpunkt bestimmen. Diese Fälle werden an dieser Stelle nicht weiter betrachtet. An dieser Stelle gelten die beiden Punkte P_1 und P_2 zunächst als nicht sichtbar zueinander.

Zu den oben beschriebenen Fällen kommen noch weitere Ausnahmefälle hinzu. Zwei Punkte sind stets zueinander sichtbar, wenn es sich um benachbarte Eckpunkte desselben Polygons handelt. Dabei wird vorausgesetzt, dass keine zwei Polygone sich so überschneiden, dass eine gemeinsame Fläche entsteht (siehe Abbildung 2.3).

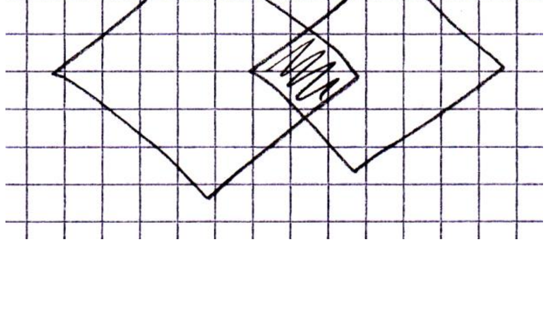

Abbildung 2.3: Verbotene Überschneidung zweier Polygone

Der zweite Ausnahmefall betrifft die zwei anliegenden Kanten des Eckpunkts eines Polygons. Bei der Prüfung auf Sichtbarkeit für solch einen Punkt, dürfen jene Kanten nicht betrachtet werden, da diese stets die Sichtlinie am Eckpunkt berühren.

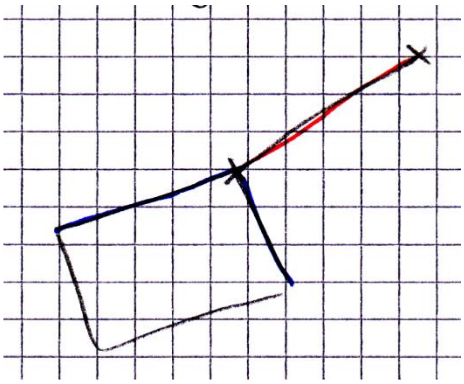

Abbildung 2.4: Anliegende Kanten eines Eckpunkts

Der letzte Ausnahmefall kommt zur Anwendung, wenn es sich bei P_1 und P_2 um Eckpunkte desselben Polygons handelt, die nicht benachbart sind. Im Falle von konvexen Polygonen sind solche nie zueinander sichtbar. Da jedoch auch konkave Polygone zugelassen sind, muss überprüft werden, ob die Sichtlinie durch das innere Gebiet des Polygons verläuft.

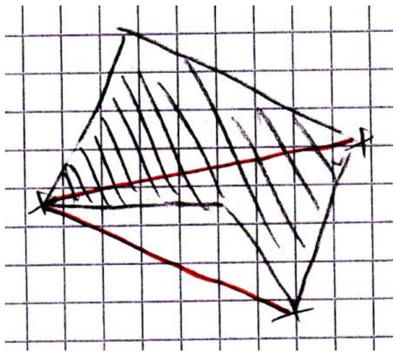

Abbildung 2.5: Punkte desselben Polygons

Da überschlagene Polygone nicht zugelassen sind (siehe 3.2), ist es ausreichend zu prüfen, dass die Sichtlinie auf der Seite des eingeschlossenen Winkels liegt.

2.1.2 Kürzester Pfad

Für alle sichtbaren Punktepaare wird die euklidische Entfernung bestimmt, die das Kantengewicht darstellen. Alle anderen Punktepaare haben keine gemeinsame Kante bzw. besitzen ein unendliches Gewicht. Zu dem so konstruierten Graph wird nun die kürzeste Route zwischen Start- und Endpunkt mittels Dijkstras Algorithmus berechnet.

2.2 Velocity Planning

Nachdem durch die Lösung des PPPs ein Pfad zwischen Start- und Zielpunkt bestimmt wurde, muss nun die Geschwindigkeit entlang dieses Pfades ermittelt werden, sodass es keine Kollisionen mit bewegten Hindernissen gibt. Dies erfolgt in zwei Schritten.

Im ersten Schritt wird bestimmt, zu welchen Zeitintervallen Streckenabschnitte nicht passierbar sind aufgrund passierender Objekte. Das Resultat ist ein $s \times t$-Raum, der sogenannte verbotene Regionen enthält (siehe Abbildung 2.6). Dabei ist s die Bogenlänge des Pfades.

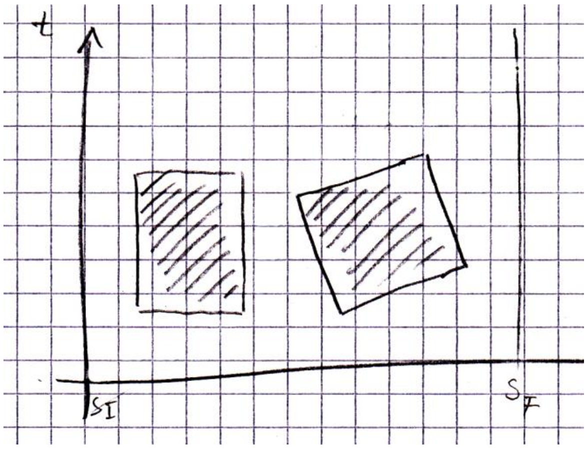

Abbildung 2.6: $s \times t$-Raum mit verbotenen Regionen

Der nachfolgende Schritt bestimmt nun einen neuen Pfad innerhalb dieses Raumes. Dieser legt das Geschwindigkeitsprofil des Roboters fest (siehe Abbildung 2.7). Man kann ihn auf ähnlicher Weise bestimmen wie bereits die Lösung des PPPs. Allerdings muss dabei zusätzlich beachtet werden, dass der Pfad nicht rückwärts in der Zeit verlaufen kann und eine maximale Geschwindigkeit nicht überschritten wird.

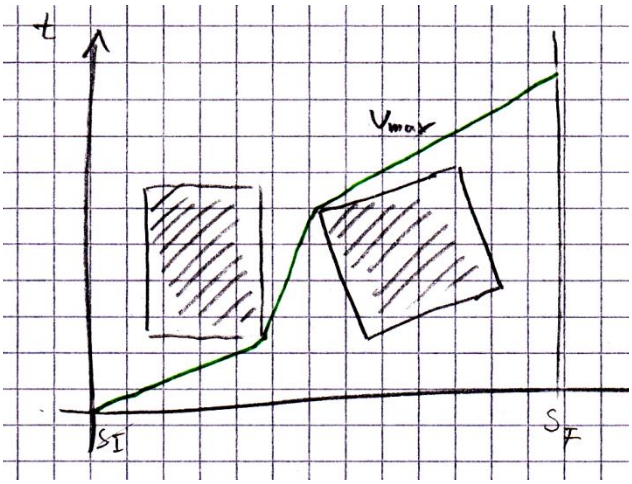

Abbildung 2.7: Geschwindigkeitsprofil entlang des Pfades im $s \times t$-Raum

2.2.1 Verbotene Regionen

Die Lage der verbotenen Regionen wird durch eine Transformation der bewegten Hindernisse aus dem $x \times y \times t$-Raum in den $s \times t$-Raum berechnet. Ein bewegtes Hindernis wird genau wie ein statisches Hindernis durch ein Polygon repräsentiert, dass außerdem einem Bewegungspfad im $x \times y \times t$-Raum folgt.

Für einen geraden Streckenabschnitt des Roboters und des bewegten Hindernisses lässt sich ein Teil einer verbotenen Region wie folgt berechnen:

Aus dem Streckenabschnitt des bewegten Hindernisses lässt sich ein Geschwindigkeitsvektor bestimmen:

$$\boldsymbol{v} = \frac{1}{\Delta t}\begin{bmatrix}\Delta x\\ \Delta y\end{bmatrix}$$

Das Polygon wird an den Startpunkt des Streckenabschnittes gesetzt:

$$\boldsymbol{O}_{\boldsymbol{k}}^{\boldsymbol{m}}(t_1) = \boldsymbol{P}_{\boldsymbol{k}} + \begin{bmatrix}x_1\\ y_1\end{bmatrix}$$

Die Transformation des Polygons in den $s \times t$-Raum erfolgt durch Transformation der Eckpunkte. Zum Verständnis werden nun ein Punkt und ein Streckenabschnitt betrachtet. Der Punkt besitzt eine Geschwindigkeit und wird zu einem bestimmten Zeitpunkt t auf eine bestimmte Stelle s des Streckenabschnitts treffen. Anhand der Abbildung 2.8 ist zu sehen, dass die Stelle anhand des Schnittpunkts des verlängerten Geschwindigkeitsvektors zu ermitteln ist. Um den Zeitpunkt t zu bestimmen, muss zunächst die Entfernung d zwischen dem Startpunkt des Punktes und dem Schnittpunkt bestimmt werden. Teilt man die Entfernung durch die absolute Geschwindigkeit, dann erhält man den Zeitpunkt t.

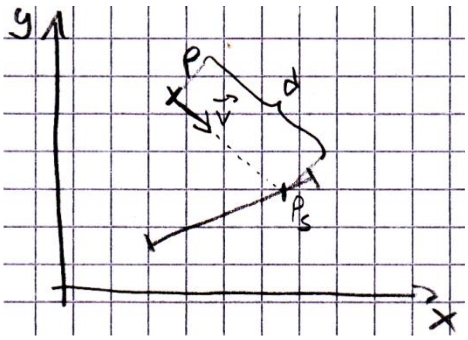

Abbildung 2.8: Bewegter Punkt und Streckenabschnitt

Eine andere Herangehensweise ist, den Streckenabschnitt und den negativen Geschwindigkeitsvektor als eine Basis des $s \times t$-Raumes aufzufassen (siehe Abbildung 2.9).

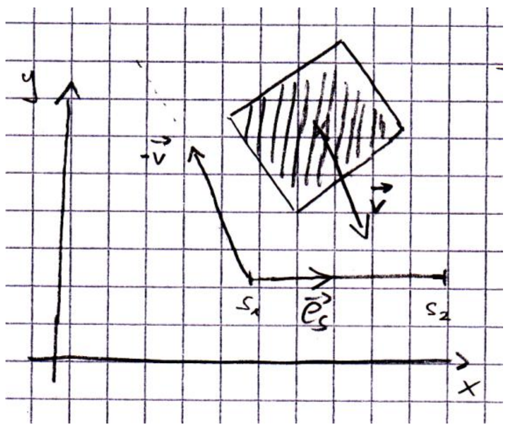

Abbildung 2.9: $s \times \boldsymbol{t}$-Basis

Mit Hilfe dieser Basis lassen sich die Koordinaten (s, t) einfach in (x, y)-Koordinaten umrechnen:

$$\begin{bmatrix}x - x_1\\ y - y_1\end{bmatrix} = (s - s_1) \cdot \boldsymbol{e_s} - (t - t_1) \cdot \boldsymbol{v}$$

Wobei $\boldsymbol{e}_s$ der Einheitsvektor des Streckenabschnitts des Roboters ist. Diese Gleichung lässt sich in das folgende Gleichungssystem umwandeln, dessen Lösung zu dem transformierten Polygon im $s \times t$-Raum führt:

$$\begin{bmatrix} x - x_1 \\ y - y_1 \end{bmatrix} = [\boldsymbol{e}_s \quad -\boldsymbol{v}] \begin{bmatrix} s - s_1 \\ t - t_1 \end{bmatrix}$$

Da jedoch das Polygon die Strecke des Roboters nicht immer in seiner Gänze durchläuft, muss es beschnitten werden (siehe Abbildung 2.10).

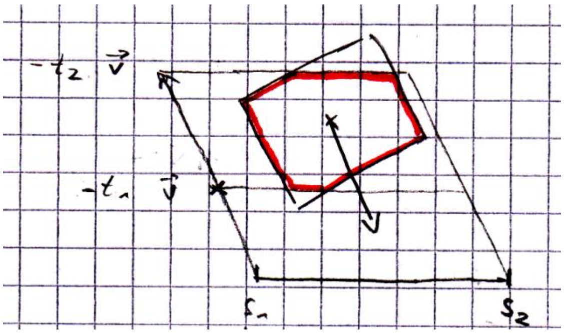

Abbildung 2.10: Beschneidung des Polygons im $x \times y$-Raum

Zum einen wird der Schnittbereich durch die Strecke des Roboters begrenzt. Sie legt das zulässige s-Intervall fest. Zum anderen wird das Polygon durch das Zeitintervall $[t_1, t_2]$ beschnitten, welches durch den Pfad des Hindernisses angegeben wird.

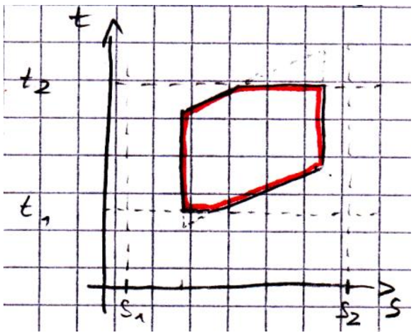

Abbildung 2.11: Beschneidung des Polygons im $s \times t$-Raum

Eine Beschneidung kann bereits vor der Transformation erfolgen.

In jedem Zeitintervall wird für jedes Streckenstück des Roboters das bewegte Hindernis in den $s \times t$-Raum transformiert. Dadurch entstehen in einem Raster die Kacheln der verbotenen Regionen, wie in Abbildung 2.12 zu sehen.

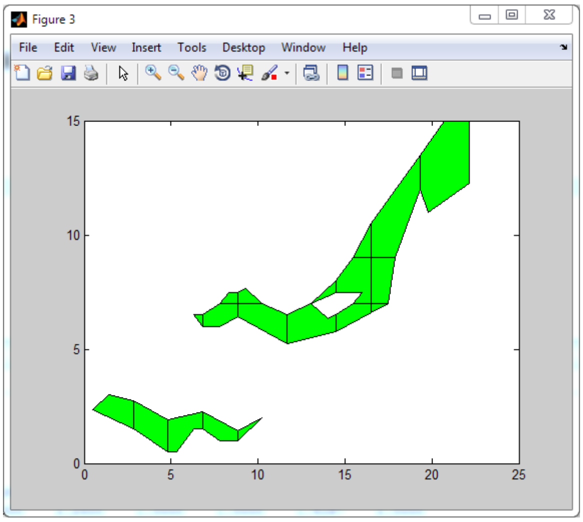

Abbildung 2.12: Gekachelte verbotene Regionen im $s \times t$-Raum

Die Kacheln müssen nun noch miteinander verbunden werden. Andernfalls könnte durch einer der „Fugen“ navigiert werden. Das Ergebnis ist in Abbildung 2.13 dargestellt.

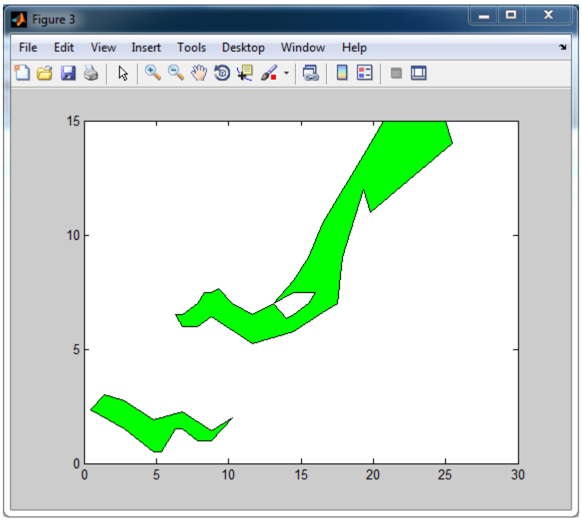

Abbildung 2.13: Fugenlose verbotene Regionen im $s \times t$-Raum

2.2.2 Geschwindigkeitsprofil

Nach der Bestimmung der verbotenen Regionen muss nun noch ein Pfad, das Geschwindigkeitsprofil, durch den $s \times t$-Raum ermittelt werden, welcher keiner dieser Regionen durchläuft. Dieser Pfad muss den Start- und Endpunkt des Roboterpfades verbinden. Im $s \times x$-Raum muss dieser also von $s_{min} = 0$ nach $s_{max} = L$ gelangen[2]. Die Zeitkomponenten der Start- und Endpunkte sind $t_{min} = 0$ und t_{max}. Der Wert von t_{max} entweder vorbestimmt sein oder es handelt sich die minimale Zeit aller möglichen Pfade von s_{min} nach s_{max}.

Das Geschwindigkeitsprofil wird ähnlich wie bereits beim Path Planning unter 2.1 berechnet. Allerdings müssen die Regeln für die Sichtbarkeit erweitert werden. Da es sich bei einer Dimension im $s \times t$-Raum um die Zeit handelt, werden folgende Regeln hinzugefügt:

[2] L ist die Gesamtlänge des Roboterpfades.

1. Der Roboter kann sich nur in der Zeit vorwärts bewegen. Darum sind zwei Knoten des Graphs nur dann sichtbar, wenn die Zeit des ersten vor dem des zweiten liegt.
2. Der Roboter kann sich nicht unendlich schnell bewegen, sondern besitzt eine Höchstgeschwindigkeit.

Formal sehen die Regeln so aus:

1. $t_i < t_j$
2. $\left|\frac{s_j - s_i}{t_j - t_i}\right| \leq v_{max}$

Wobei $n_i = (s_i, t_i)$ der erste und $n_j = (s_j, t_j)$ der zweite Knoten ist.

Bei der Gewichtung der Kanten kann man unterschiedliche Ansätze verfolgen. Entweder man nimmt ein konstantes Gewicht, sodass der Pfad mit den wenigsten „Kurven" gewählt. Die Zeit oder Länge der Kanten kann auch als Gewicht dienen. Im Falle der vorbestimmten Zeit t_{max} ist eine Zeitgewichtung jedoch nicht sinnvoll. Ein Längengewicht ist nur für Fälle interessant, bei denen Roboter Stücke des Weges mehrmals befahren (also rückwärtsfahren).

Nach diesen Regeln wird der Graph bestimmt, der beispielsweise wie in Abbildung 2.14 aussehen kann.

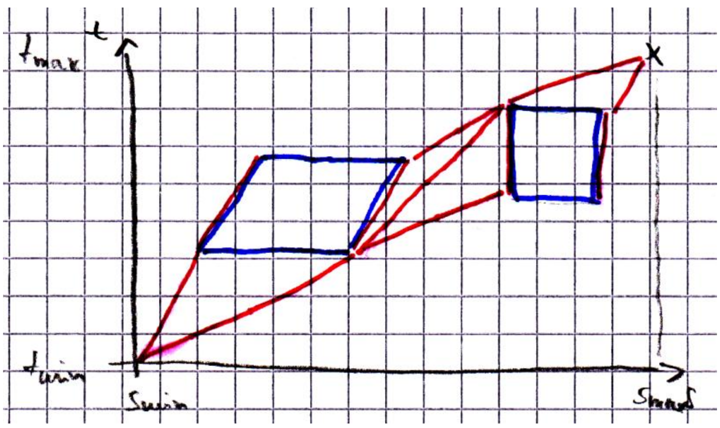

Abbildung 2.14: Graph im $s \times t$-Raum

Durch Anwendung des Dijkstra-Algorithmus erhält man auch hier einen Pfad von Start- zu Endpunkt (siehe Abbildung 2.15).

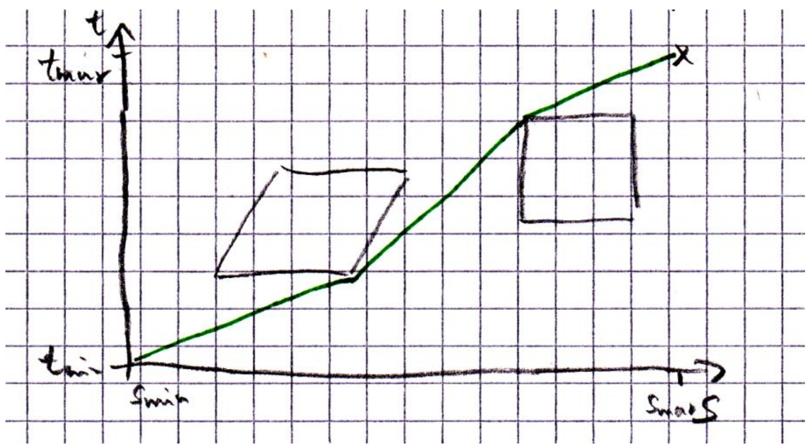

Abbildung 2.15: Pfad im $s \times t$-Raum

2.3 $x \times y \times t$-Pfad

Aus dem Pfad des Path Plannings und dem Geschwindigkeitsprofil des Velocity Plannings ($s \times t$-Pfad) lässt sich nun ein dreidimensionaler Pfad im $x \times y \times t$-Raum berechnen. Dazu werden die Daten des Geschwindigkeitsprofils genutzt um Zeitwerte für die Eckpunkte des $x \times y$-Pfades zu bestimmen. In gleicher Weise werden Ortspunkte (x, y) zu den Eckpunkten des Geschwindigkeitsprofils anhand der Daten des $x \times y$-Pfades bestimmt.

Das Ergebnis ist ein $x \times y \times t$-Pfad, welcher die Eckpunkte des $x \times y$- und $s \times t$-Pfades sortiert nach ihren s-Werten enthält.

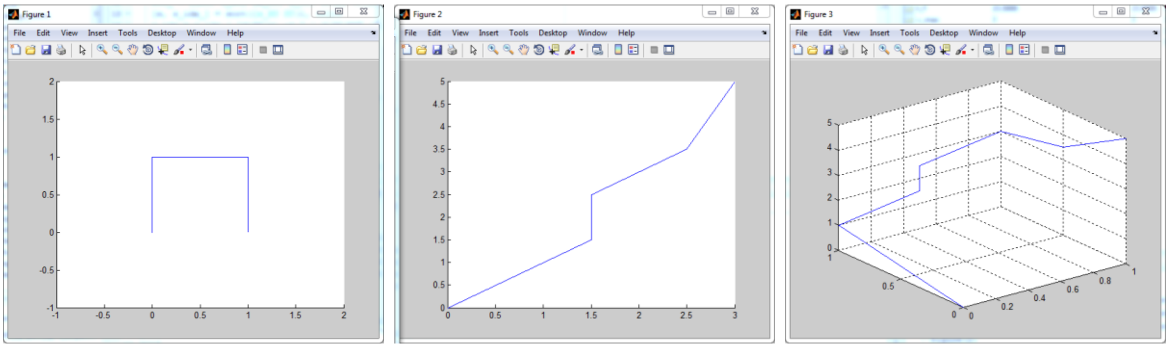

Abbildung 2.16: Erzeugung des $x \times y \times t$-Pfades (von links nach rechts: $x \times y$, $s \times t$ und $x \times y \times t$)

3 Definitionen

3.1 Roboter

Beim Roboter handelt es sich um ein Punktobjekt[3], dem zu jedem Zeitpunkt t ein Ort (x, y) zugeordnet ist. Dabei ist zu beachten, dass der Roboter nur stetige Bewegungen durchführen kann und eine Höchstgeschwindigkeit nicht überschreitet.

3.2 Statisches Hindernis

Ein Hindernis O_k^s wird durch ein Polygon beschrieben. Das Polygon grenzt dabei die zusammenhängende Fläche des Hindernisses ein. Der Rand dieser Fläche ist jedoch nicht Bestandteil des Hindernisses. Das Polygon selbst wird durch eine Serie von Punkten beschrieben. Zwei aufeinanderfolgende Punkte bilden einen Streckenabschnitt des Polygons. Der letzte und der erste Punkt bilden ebenfalls einen Streckenabschnitt. Diese Strecken sind gerichtet. Konventionsgemäß befindet sich auf der linken Seite der Strecke immer die Innenfläche des Polygons, sodass sie entgegen des Uhrzeigersinns von den Strecken umschlossen wird (siehe Abbildung 3.1).

[3] Für eine praktische Anwendung, in der der Roboter eine Fläche einnimmt, kann das Problem übersetzt werden. Dazu werden die Hindernisse in ihrer Fläche vergrößert, und der Roboter auf Punktgröße verkleinert. Die Hindernisse erhalten also eine Randzone, um den Roboter auf den nötigen Abstand zu halten.

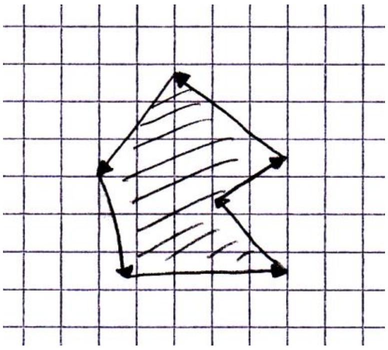

Abbildung 3.1: Polygon

Die Streckenabschnitte dürfen keine anderen Streckenabschnitte schneiden oder berühren. Davon ausgenommen sind die gemeinsamen Eckpunkte benachbarter Strecken. [4]

3.3 Dynamisches Hindernis

Ein bewegtes Hindernis O_k^m wird ebenso wie ein statisches Hindernis durch ein Polygon beschrieben. Da dieses sich bewegt, wird außerdem noch ein Pfad im $x \times y \times t$-Raum angegeben (siehe Abbildung 3.2).

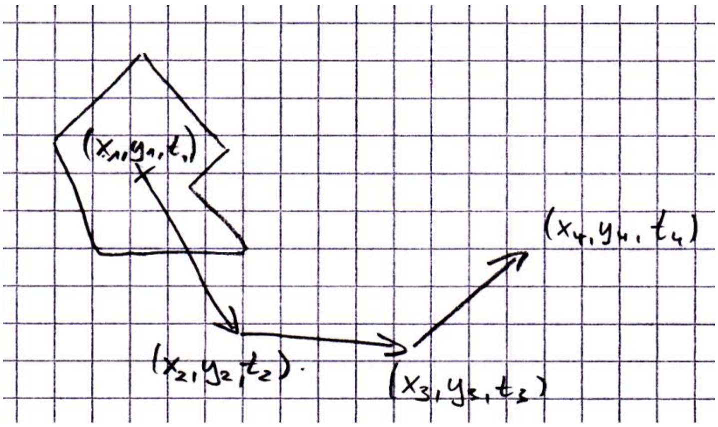

Abbildung 3.2: Bewegungspfad

Ein Punkt (x_i, y_i, t_i) des Pfades gibt den Aufenthaltspunkt des Polygons zu einer Zeit an. Die Punkte des Polygons werden entsprechend verschoben.

$$\boldsymbol{O_k^m}(t_i) = \boldsymbol{P_k} + \begin{bmatrix} x_i \\ y_i \end{bmatrix}$$

Zwischen den Punkten führt das Polygon eine lineare Bewegung zum nächsten Punkt aus. Das Hindernis existiert nur während des Zeitintervalls $[t_1, t_n]$. Das heißt vor dem ersten und nach dem letzten Pfadpunkt existiert das Hindernis nicht. Außerdem gilt zu beachten, dass sich ein Hindernis nur vorwärts durch die Zeit bewegen darf. Instantane Bewegungen sind ebenfalls unzulässig.

3.4 Verbotene Region

Eine verbotene Region ist ein Gebiet im $s \times t$-Raum, welcher nicht vom Roboter betreten werden darf. Sie geben an zu welchen Zeitpunkten welche Stellen des $x \times y$-Pfades von einem dynamischen Hindernis blockiert sind.

[4] Grund dafür ist die Vereinfachung von Funktionen, wie das Zerschneiden von Polygonen.

Bei den Regionen handelt es sich stets um Flächen, die wie Hindernisse durch den Rand von Polygonen eingegrenzt sind. Wie auch bei Hindernissen gilt der Rand selbst nicht als Teil der Fläche.

Im Gegensatz zu Hindernissen, sind hier auch Polygonformen erlaubt, bei denen Polygonstrecken gemeinsame Punkte haben, so lang die eingegrenzte Fläche zusammenhängend ist (siehe Abbildung 3.3).[5] Nach wie vor sind überschlagene Polygone nicht erlaubt, da diese getrennte Flächen bilden (siehe Abbildung 3.4).

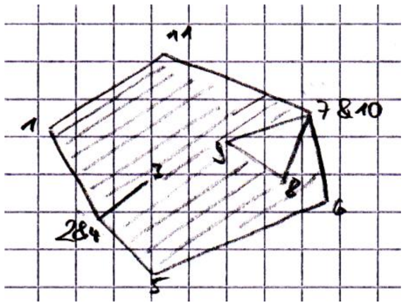

Abbildung 3.3: Fugen innerhalb einer verbotenen Region

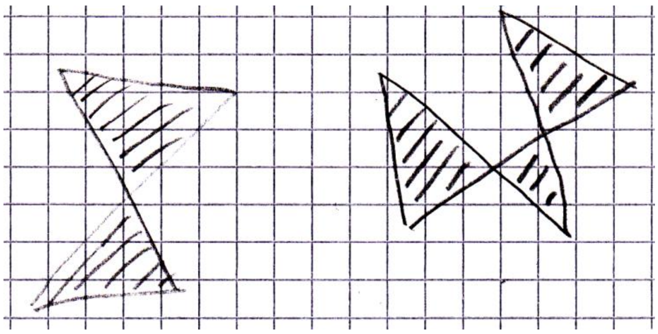

Abbildung 3.4: Überschlagene Polygone sind nicht erlaubt

[5] Derartige Polygone sind erlaubt, da durch die Transformation der Hindernisse in den $s \times t$-Raum derartige Polygone entstehen können. Weiterhin werden diese Polygone nicht zerschnitten, weshalb dies hier zu keinen weiteren Problemen führt.