

# VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ  
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY  
DEPARTMENT OF INTELLIGENT SYSTEMS

## PLÁNOVÁNÍ OPTIMÁLNÍ TRAJEKTORIE LETADLA

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE

AUTHOR

PETR JANKŮ

BRNO 2013



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ  
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ  
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY  
DEPARTMENT OF INTELLIGENT SYSTEMS

# PLÁNOVÁNÍ OPTIMÁLNÍ TRAJEKTORIE LETADLA

PATH PLANNING OF AIRPLANE

BAKALÁŘSKÁ PRÁCE  
BACHELOR'S THESIS

AUTOR PRÁCE  
AUTHOR

PETR JANKŮ

VEDOUCÍ PRÁCE  
SUPERVISOR

Ing. JAN SAMEK, Ph.D.

BRNO 2013

## Abstrakt

Tato práce se zabývá hledáním optimální trajektorie letadla, jak ve spojitém, tak v diskretizovaném prostředí. Teoretická část je věnována základním pojmům z problematiky plánování pohybu a popisu vybraných metod plánování pohybu letadla. Praktická část je věnována implementaci vybraných metod a ověření jejich optimálnosti. Z experimentů vyšlo, že menší výpočetní čas potřebuje metoda pracující se spojitým prostředím, ale né vždy najde optimální cestu.

## Abstract

This work deals with finding an optimal trajectory of an aircraft, both in continuous and discretized environment. Theoretical part is devoted to the basic concepts of motion planning problems and description of selected planning methods of aircraft motion. Practical part is devoted to the implementation of selected methods and verification of optimality. Experiments confirmed that the continuous environment method requires shorter computing time; however, may not always find an optimal path.

## Klíčová slova

Dubinsovi křivky, plánování pohybu, diskretizace prostředí.

## Keywords

Dubins curves, motion planning, discretization environment.

## Citace

Petr Janků: Plánování optimální trajektorie letadla, bakalářská práce, Brno, FIT VUT v Brně, 2013

# Plánování optimální trajektorie letadla

## Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Jana Samka, Ph.D.

.....

Petr Janků  
15. května 2013

## Poděkování

Rád bych poděkoval Ing. Janu Samkovi, Ph.D. za jeho ochotu, vstřícnost a neocenitelné rady k této práci.

© Petr Janků, 2013.

*Tato práce vznikla jako školní dílo na Vysokém učení technickém v Brně, Fakultě informačních technologií. Práce je chráněna autorským zákonem a její užití bez udělení oprávnění autorem je nezákonné, s výjimkou zákonem definovaných případů.*

# Obsah

<b>1 Úvod</b>	<b>3</b>
<b>2 Základní znalosti k plánování pohybu</b>	<b>4</b>
2.1 Základní pojmy . . . . .	4
2.1.1 Stavový prostor . . . . .	4
2.1.2 Stupně volnosti . . . . .	4
2.1.3 Holonomní a neholonomní systém . . . . .	5
2.1.4 Dubinsovi křivky . . . . .	6
2.1.5 Klotoida . . . . .	6
2.1.6 Pracovní prostor . . . . .	7
2.1.7 Konfigurační prostor . . . . .	8
2.2 Prohledávací algoritmy . . . . .	8
2.2.1 Neinformované metody . . . . .	9
2.2.2 Informované metody . . . . .	9
<b>3 Metody plánování pro letadlo ve 3D bez překážek</b>	<b>11</b>
3.1 Algoritmus pro 3D UAV - Vytváření cesty a následování . . . . .	11
3.1.1 Vytvoření 2D cesty . . . . .	13
3.1.2 Vytvoření 3D cesty . . . . .	14
3.1.3 Minimální množina parametrů definované cesty . . . . .	15
3.2 Optimální cesta pro Dubinsovo letadlo . . . . .	15
3.2.1 Optimální trajektorie pro letadlo . . . . .	16
3.3 Efektivní dvoufázové 3D plánování pohybu pro malé UAV . . . . .	17
3.3.1 Globální mřížkový plánovač s proveditelnými spojeními . . . . .	17
3.3.2 Lokální runtime plánování podle pohybových primitiv . . . . .	17
<b>4 Zvolené metody a jejich popis implementace</b>	<b>19</b>
4.1 Algoritmus pro 3D UAV - Vytváření cesty a následování . . . . .	19
4.1.1 Implementace Dubinsových křivek . . . . .	20
4.1.2 Implementace počátečního a koncového kruhového oblouku . . . . .	22
4.1.3 Implementace spirály . . . . .	24
4.2 Efektivní dvoufázové 3D plánování pohybu pro malé UAV . . . . .	26
4.2.1 Diskretizace prostředí . . . . .	26
4.2.2 Implementace A* . . . . .	27

<b>5</b>	<b>Testování a vyhodnocení výsledků</b>	<b>28</b>
5.1	Popis aplikace . . . . .	28
5.2	Popis a vyhodnocení provedených experimentů . . . . .	28
5.2.1	Ověření optimálnosti implementovaných spirál . . . . .	29
5.2.2	Optimalizace A* . . . . .	30
5.2.3	Porovnání diskrétní a spojité metody . . . . .	31
<b>6</b>	<b>Závěr</b>	<b>33</b>

# Kapitola 1

## Úvod

Tato práce se zabývá hledáním optimální trajektorie letadla, kde se letadlo pohybuje ve 3D prostoru bez jakýchkoliv překážek. Pojem optimální trajektorie v mé práci znamená nalezení nejkratší cesty z počátečního bodu do cílového bodu za co nejkratší dobu s ohledem na kinematická omezení letadla.

Hledání trajektorie letadla se dá uplatnit v různých aplikacích, konkrétně v počítačových hrách, v letových trenažérech a nebo při plánování trajektorie bezpilotních letounů (někdy nazýváno UAV nebo také dron), které se využívají jak k civilním, tak k vojenským úkolům. Naváděné střely se dají klasifikovat také jako hledání optimální trajektorie letadla.

První část této práce představuje obecný úvod do teorie plánování pohybu. První část kapitoly (2) popisuje základní pojmy potřebné k plánování pohybu a druhá část této kapitoly popisuje prohledávací algoritmy používané k plánování cesty v diskrétním prostředí. Kapitola (3) popisuje některé metody, které lze použít k naplánování trajektorie letadla.

Druhá část této práce se zabývá implementací vybraných metod a jejich testováním. Konkrétně kapitola (4) je věnována implementaci vybraných metod a kapitola (5) je zaměřena na ověření optimálnosti nalezených cest a porovnání vybraných metod.

## Kapitola 2

# Základní znalosti k plánování pohybu

Plánování pohybu je algoritmus, který vytvoří spojitou cestu, která spojí počáteční pozici s cílovou pozicí. Tento algoritmus musí vytvořit takovou cestu, která se vyhne překážkám, které se mohou v modelovaném prostředí vyskytovat.

V této kapitole vysvětlím pojmy, které se vyskytují v problému plánování pohybu.

### 2.1 Základní pojmy

Zde budou uvedeny pojmy, které budou dále použity v textu.

#### 2.1.1 Stavový prostor

V této podkapitole vycházím z práce [6]. Při plánování se každá situace letadla nazývá *stav*. Označuje se jako  $x$  a množina všech možných stavů se nazývá *stavový prostor* a je označen jako  $X$ . Pro diskrétní plánování bude důležité, že tato množina je spočetná a ve většině případů konečná. Každá akce označovaná jako  $u$  aplikovaná na aktuální stav  $x$ , vytvoří nový stav  $x'$ , na který se stav  $x$  změní. Tato změna je vyjádřena *přechodovou funkcí*  $f$ . Rovnice přechodové funkce se nachází níže:

$$x' = f(x, u) \quad (2.1)$$

Množina akcí, kterou lze aplikovat na stav  $x$  nazýváme *akční prostor* a označujeme ho jako  $U(x)$ . Stejná akce může být použitelná v různých stavech, a proto je vhodné definovat množinu  $U$ , která bude obsahovat všechny možné akce všech stavů.

$$U = \bigcup_{x \in X} U(x) \quad (2.2)$$

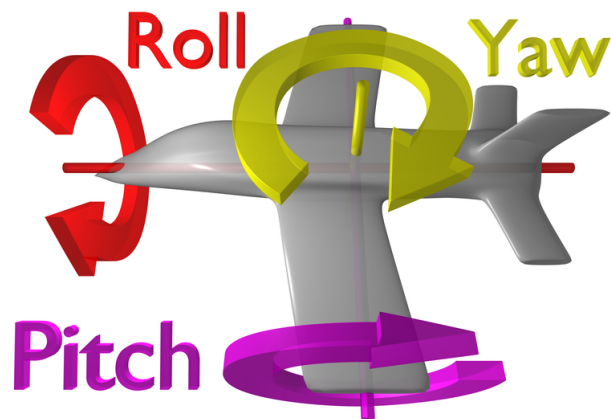
V plánování je také třeba určit množinu *cílových stavů*, která je označena jako  $X_G \subset X$ . Úkolem plánování je najít konečnou posloupnost akcí, která počáteční stav  $x_0$  postupně transformuje na některý ze stavů množiny  $X_G$ .

#### 2.1.2 Stupně volnosti

Stupně volnosti [9] označují základní směry posunu a otáčení, kterými se bod nebo těleso může pohybovat a otáčet. Pohyb letadla v prostoru má šest stupňů volnosti, což je popsáno jako:



- Pohyb nahoru a dolů.
- Pohyb doleva a doprava.
- Pohyb dopředu a dozadu.
- Naklánění dopředu a dozadu.
- Otáčení vlevo a vpravo.
- Naklánění ze strany na stranu.



Obrázek 2.1: Stupně volnosti pro letadlo. Obrázek převzat z [9].

### 2.1.3 Holonomní a neholonomní systém

Tato podkapitola vychází z [6, 5]. Roboty můžeme dělit podle omezení kladená na pohyb holonomní a neholonomní.

#### Holonomní systém

Systém můžeme označit za neholonomní, jestliže všechna jeho omezení jsou holonomní. Takový systém je omezen pouze souřadnicemi systému a časem. Tato omezení lze formulovat do funkce

$$f(x_1, x_2, \dots, x_n, t) = 0 \quad (2.3)$$

Holonomní systém můžeme také určit podle stupňů volnosti. Jestliže je počet říditelných stupňů volnosti roven celkovému počtu stupňů volnosti, pak se jedná o holonomní systém. Příkladem holonomního systému je matematické kyvadlo.

## Neholonomní systém

Neholonomní systém lze určit podobně jako u holonomního systému podle stupňů volnosti. Je-li počet říditelných stupňů volnosti menší jak celkový počet stupňů volnosti, pak můžeme označit systém za neholonomní.

Zda je robot neholonomní lze určit také podle diferenciálních omezení, která jsou na něj kladena a která nejsou plně integrovatelná k odstranění časových derivací stavových proměnných. Příkladem neholonomního robota je například letadlo. Pohyb letadla do strany je určen jeho úhlem natočení a jeho rychlostí.

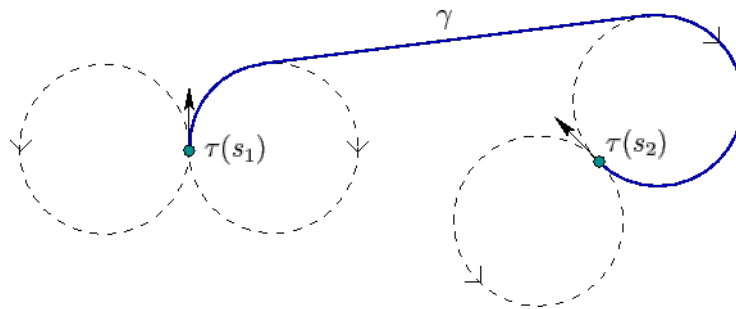
### 2.1.4 Dubinsovi křivky

Dubinsovi křivky představují nejkratší křivky, které spojují dva body ve dvoudimenzionální Euklidovské rovině (např.  $x, y$  rovina) s omezením na křivost cesty a s předepsanou počáteční a koncovou tečnou. Dubinsova křivka se běžně používá v oblasti robotiky a teorii řízení jako způsob, jak plánovat cestu pro kolové roboty, letadla a podvodní vozidla. K dispozici jsou jednoduché geometrické a analytické metody pro výpočet optimální cesty. Například jednoduchý kinematický model vozu je

$$\begin{aligned}x &= V \cos(\theta), \\y &= V \sin(\theta), \\ \theta &= u,\end{aligned}\tag{2.4}$$

kde  $(x, y)$  jsou souřadnice auta v Euklidovském prostoru,  $\theta$  udává směr a auto se pohybuje konstantní rychlostí  $V$  a  $u$  je vybrán z intervalu  $U = [-\tan \phi_{max}, \tan \phi_{max}]$ . V tomto případě maximální velikost otáčení odpovídá minimálnímu poloměru otočení (a ekvivalentně maximálnímu zakřivení). Předepsaná počáteční a koncová tečna odpovídá počátečnímu a koncovému směru. Dubinsova křivka udává nejkratší cestu spojující dva orientované body.

Typ optimální cesty může být popsán použitím analogie s autem, které zatáčí doprava(R), doleva(L) nebo jede rovně(S). Optimální cesta bude vždy alespoň jeden z těchto šesti typů: RSR, RSL, LSR, LSL, RLR, LRL. Více o Dubinsových křivkách lze nalézt [6, 7].



Obrázek 2.2: Ukázka dubinsovi křivky . Obrázek převzat z [6].

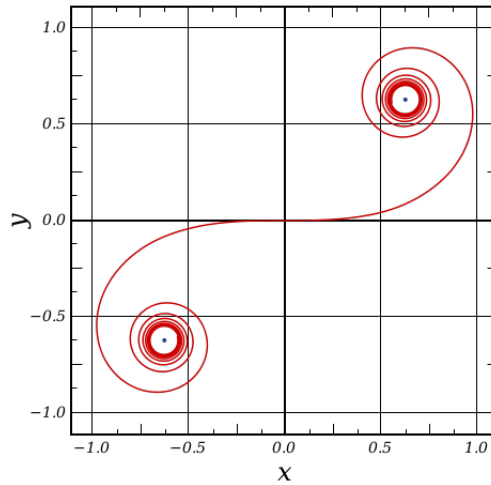
### 2.1.5 Klotoida

Tato podkapitola byla vypracována podle [8, 10]. Klotoida je známá také jako Eulerova spirála nebo také jako Cornova spirála. Je převážně používána k návrhu a aplikaci přechodnic.

Používá se například při návrhu železnice, dálnice a používá se také při návrhu trajektorie pohybu. Je to rovinná a hladká parametrická křivka. Je to jeden z nejjednodušších příkladů oblouku, který může být konstruován z jejího zakřivení, ačkoliv matematicky je poněkud složitá. Eurelova spirála je populární ze dvou důvodů. První důvod je zakřivení, které se mění lineárně podle eulerovi spirály, která usnadňuje neholonomním agentům následovat tuto trajektorii. Druhý důvod je možné použití na přibližné řešení optimální trajektorie. Souřadnice eurelovi spirály mají následující parametrické vyjádření v kartézském souřadném systému:

$$\begin{cases} x(s) = A \int_0^s \cos(1/2ks^2) ds = \frac{A}{(2k)^{1/2}} \int_0^\theta \frac{\cos \theta}{\theta^{1/2}} d\theta, \\ y(s) = A \int_0^s \sin(1/2ks^2) ds = \frac{A}{(2k)^{1/2}} \int_0^\theta \frac{\sin \theta}{\theta^{1/2}} d\theta, \end{cases} \quad (2.5)$$

kde proměnná  $s$  představuje křivočarou úsečku na cestě. Zakřivení se mění lineárně v závislosti na délce zakřivení. Zakřivení udává změna velikosti  $k$  a  $A$  je libovolná kladná konstanta.



Obrázek 2.3: klotoida, kde  $A = \frac{1}{\sqrt{2}}$ . Obrázek převzat z [10].

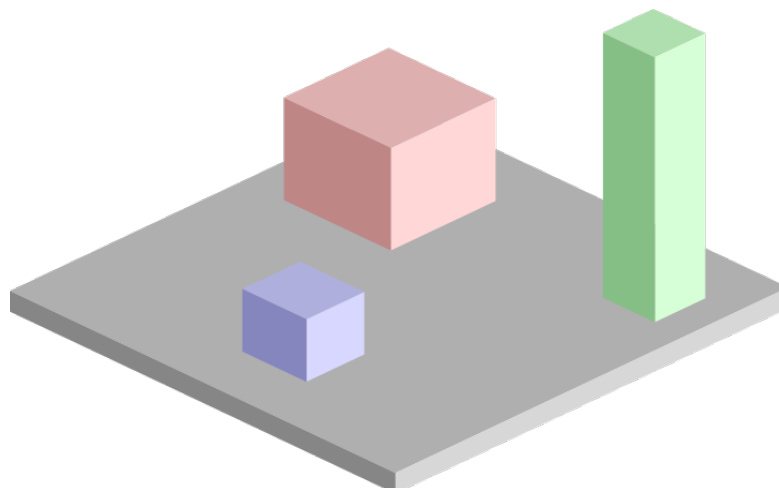
### 2.1.6 Pracovní prostor

Pracovní prostor je prostředí, ve kterém se robot nachází a ve kterém se také pohybuje. Pracovní prostor se značí  $W$  a je to  $N$ -rozměrný euklidovský prostor  $R^N$ . Pracovní prostor bývá buď dvourozměrný  $W = R^2$  a nebo trojrozměrný  $W = R^3$ . Pracovní prostor zahrnuje jak robota, tak taky přepážky, které omezují pohyb robota v tomto prostoru.

Prostředí lze reprezentovat jako spojité nebo jako diskretní [3]:

#### Diskrétní

Diskrétní prostředí je prostor, který je rozdělen buď do buněk nebo do grafu. V případě, že je prostor rozdělen do buněk, je počet buněk závislý na velikosti prostoru a velikosti buněk. Velký počet buněk zvyšuje výpočetní náročnost, ale umožní přesněji modelovat prostředí robota. Buňky mohou mít různé tvary a velikosti. Velikost je omezena velikostí robota tzn., že buňka nesmí být menší než je velikost robota.



Obrázek 2.4: Příklad pracovního prostoru . Obrázek převzat z [11].

## Spojité

Spojité prostředí, n rozdíl od diskrétního prostředí není rozděleno do buněk, protože se jedná o kontinuální prostor a vyznačuje se složitější náročností na prohledávání. Jelikož spojité prostředí je podobné našemu světu, je možné překážky modelovat s větší přesností a pohyb robota není omezen tvarem buňky.

### 2.1.7 Konfigurační prostor

Podle [6] je konfigurační prostor  $C$  určen podle množiny všech možných transformací, které mohou být aplikovány na robota. Má neobvyklou topologickou strukturu, která musí být korektně charakterizována k zajištění správné funkce plánovacích algoritmů. V našem případě bude konfigurační prostor roven stavovému prostoru  $C = X$ . Dimenze konfiguračního prostoru je svázána s počtem stupňů volnosti robota. Dimenze konfiguračního prostoru je počet parametrů nezbytných k určení unikátní konfigurace.

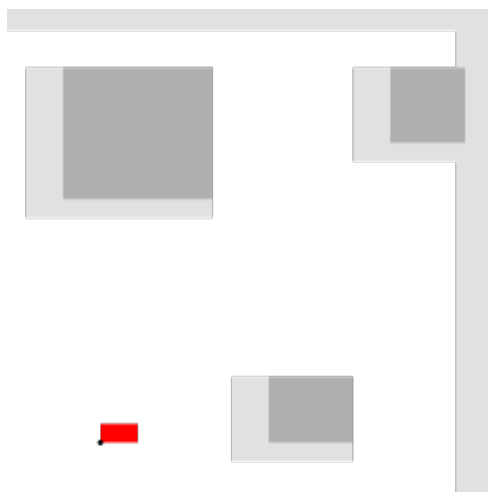
Konfigurace v konfiguračním prostoru se značí  $q \in C$ , kde  $q = (x_t, y_t, \theta)$  nebo  $q = (x_t, y_t, z_t, \theta, \phi, \psi)$ . Počáteční konfigurace se značí  $q_I \in C$  a cílová konfigurace se označuje jako  $q_G \in C$ . Výsledný plán pohybu z konfigurace  $q_I$  do  $q_G$  je potom spojitá cesta v konfiguračním prostoru.

## 2.2 Prohledávací algoritmy

Tato podkapitola vychází z [12]. Prohledávací algoritmy se hodnotí podle čtyř kritérií, kterými jsou:

- úplnost,
- časová náročnost,
- paměťová náročnost,
- optimálnost.

Metody, které prohledávají stavový prostor lze rozdělit do dvou skupin:



Obrázek 2.5: konfigurační prostor. Obrázek převzat z [11].

- neinformované/slepé,
- informované.

### 2.2.1 Neinformované metody

Neinformované metody, jak už z názvu vyplývá, nemají žádnou vhodnou znalost o cílovém stavu a nemají ani žádné prostředky, jak aktuální stavy ohodnotit. Tudíž musí systematicky prohledat stavový prostor. Jednotlivé metody se liší jen tím způsobem, jakým toto systematické procházení provádějí. Mezi nejznámější neinformované metody patří například:

- metoda prohledávání do šířky (BFS - Breadth First Search),
- metoda prohledávání do hloubky (DFS - Depth First Search),
- metoda zpětného navrácení (Backtracking),
- metoda obousměrného prohledávání (BS - Bidirectional BFS search).

### 2.2.2 Informované metody

Informované metody, narozdíl od neinformovaných metod mají k dispozici informaci o cílovém stavu. Mají také prostředky jak aktuální stavy ohodnotit. Heuristickou funkci dodává na základě znalostí člověk a informované metody jsou na ní kriticky závislé. Čím lepší heuristika je k dispozici, tím rychleji a s menším zatížením paměti dojde k nalezení řešení. Vzorec hodnotící funkce je

$$f(n) = g(n) + h(n), \quad (2.6)$$

kde  $g(n)$  je cena cesty z počátečního stavu do uzlu  $n$  a  $h(n)$  je odhad ceny z uzlu  $n$  do cílového stavu. Základními informovanými metodami jsou

- metoda Greedy Search (GS),
- metoda Uniform Cost Search (UCS),

- a metoda A\* Search.

Rozdíl mezi těmito metodami spočívá v ohodnocovací funkci. Metoda A\* používá k určení hodnoty hodnotící funkce  $g(n)$  a  $h(n)$ . Zatímco metody GS a UCS jsou extrémní případy informovaných metod. Metoda GS nepoužívá k určení hodnoty hodnotící funkce  $g(n)$ , z toho vyplývá  $f(n) = g(n) + h(n) = h(n)$ . Naopak metoda UCS nepoužívá k určení hodnoty hodnotící funkce  $h(n)$ , tudíž  $f(n) = g(n) + h(n) = g(n)$ .

## Kapitola 3

# Metody plánování pro letadlo ve 3D bez překážek

V této kapitole popíšeme tři algoritmy, které se popisují jak vytvořit cestu pro letadlo ve 3D prostoru s ohledem na jeho kinematické vlastnosti. Jako model letadla se zde bude používat UAV (Unmanned aerial vehicle), což je bezpilotní letoun. Vycházíme zde z prací [6, 4, 2].

### 3.1 Algoritmus pro 3D UAV - Vytváření cesty a následování

V tomto algoritmu se počáteční pozice bude značit jako  $P_0$  a cílová pozice jako  $P_f$ . Cesta mezi těmito body se bude značit jako  $\Gamma$  a je to přibližná cesta. Tento problém lze zjednodušit zavedením následujících dvou předpokladů.

1. Předpokládá se, že neexistují překážky, které je třeba obcházet.
2. Předpokládá se, že dynamické a strukturální omezení UAV může být vzata v úvahu jednoduchými geometrickými omezeními na cestě  $\Gamma$ .

V kartézském souřadnicovém systému  $(x, y, z)$ , který je fixovaný se zemí může být cesta  $\Gamma$  reprezentována podle následujících rovnic

$$\frac{dx(s)}{ds} = \cos \psi(s) \cos \gamma(s), \quad (3.1)$$

$$\frac{dy(s)}{ds} = \sin \psi(s) \cos \gamma(s), \quad (3.2)$$

$$\frac{dz(s)}{ds} = \sin \gamma(s), \quad (3.3)$$

kde závislá proměnná  $s$  představuje křivočarou úsečku na cestě  $\Gamma$ . Daný bod  $P(s) = (x(s), y(s), z(s)) \in \Gamma$ . Úhel mezi osou  $x$  a projekcí tečného vektoru ke  $\Gamma(dP(s)/ds)$  do roviny  $xy$  je dán podle  $\psi(s) \in [-180^\circ, 180^\circ]$  (úhel kursu), zatímco úhel mezi  $dP(s)/ds$  a osou  $z$  je dán podle  $90^\circ - \gamma(s)$  ( $\gamma(s) \in [-90^\circ, 90^\circ]$ ) je úhel dráhy letu). V dalším pokračování bude užitečné si zavést další dvě rovnice pro derivaci kursu a úhlu dráhy letu

$$\frac{d\psi(s)}{ds} = \eta, \quad \frac{d\gamma(s)}{ds} = \mu. \quad (3.4)$$

Proměnné  $\eta$  a  $\mu$  budou hrát roli řídicích vstupů v problému optimálního řízení, který budeme definovat. Počáteční podmínky pro rovnice (3.1) - (3.4) jsou dány vektorem  $q_0 =$

$(x_0, y_0, z_0, \psi_0, \gamma_0)$ . První tři souřadnice vektoru představují počáteční pozici, zatímco poslední dvě počáteční směr UAV. Předpokládejme, že byla přidělena také poslední podmínka  $q_f = (x_f, y_f, z_f, \psi_f, \gamma_f)$ . Zakřivení cesty  $\Gamma$  je dáno podle

$$c(s) = \sqrt{\left(\frac{d\psi(s)}{ds}\right)^2 \cos^2 \gamma(s) + \left(\frac{d\gamma(s)}{ds}\right)^2} = \sqrt{\eta^2(s) \cos^2 \gamma(s) + \mu^2(s)}. \quad (3.5)$$

Nechť

$$\begin{aligned} q(s) &= (x(s) \quad y(s) \quad z(s) \quad \psi(s) \quad \gamma(s))^T \\ u(s) &= (\eta(s) \quad \mu(s))^T s \end{aligned}$$

rovnice (3.1) - (3.4) mohou být přepsány společně s danou počáteční a koncovou podmínkou jako

$$\frac{dq(s)}{ds} = f(q(s), u(s)), \quad q(0) = q_0, \quad q(s_{0f}) = q_f. \quad (3.6)$$

Problém u vytvoření dráhy je řídicí funkce  $u(s)$ , která musí být určena takový způsobem, aby bylo zaručeno splnění omezení

$$-c_{max} \leq c(s) \leq c_{max}, \quad (3.7)$$

$$-\gamma_{min} \leq \gamma(s) \leq \gamma_{max}, \quad (3.8)$$

kde  $c_{max}$  je velikost maximálního zakřivení (minimální poloměr),  $\gamma_{min}$  je minimální úhel klesání a  $\gamma_{max}$  je maximální úhel stoupání. Zároveň by měla minimalizovat délku cesty  $\Gamma$

$$J(q_0, q_f, u) = \int_0^{s_f} ds. \quad (3.9)$$

Bohužel tento problém optimálního řízení nepřipouští analytické řešení, což je příliš náročné, aby bylo řešeno numerickým přístupem. Z tohoto důvodu se použije suboptimální přístup, který je přijatelný z výpočetního pohledu. Toto řešení je založeno na následujících empirických úvahách.

- Jedním z problémů optimálního řízení, které je definováno rovnicemi (3.6)-(3.9), je splnění omezení (3.7), které zahrnuje obě řídicí proměnné ( $\eta$  a  $\mu$ ) ve stejný čas. Tento problém by mohl být jednodušší, pokud by cesta z  $P_0$  do  $P_f$  měla konstantní dráhu letu  $\gamma$ . S  $\mu = 0$  a konstatou  $\gamma$  bude omezení (3.7) obsahovat pouze řídicí proměnnou  $\eta$ .
- Na základě předchozího bodu je jasné, že pro zkrácení délky  $\Gamma$  je vybrána hodnota  $\gamma$  taková, která by měla být největší (v absolutní hodnotě) a kompatibilní s omezením (3.8). Výjimkou je, když výškový rozdíl  $|z_f - z_0|$  je malý.
- V důsledku toho, že úhel dráhy letu zůstává konstantní, lze dráhu  $\Gamma$  převést do roviny. Z toho vyplývá, že se problém sníží z 3D na 2D. To je výhodné, protože pro 2D řešení existuje atraktivní numerické řešení založené na Dubinsových křivkách.

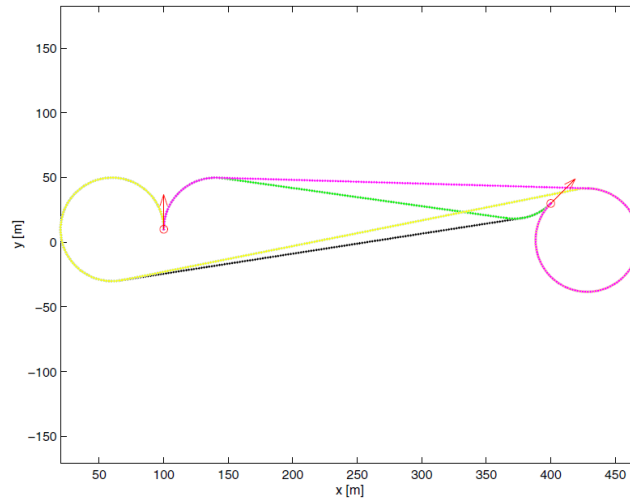
Předtím než navrhne 3D algoritmus pro vytvoření cesty, popíšeme nejdříve 2D řešení.



### 3.1.1 Vytvoření 2D cesty

Při vytváření 2D cesty musíme udělat pár zjednodušení a to  $z_f = z_0$  a  $\gamma_f = \gamma_0 = 0$ . V tomto případě je nejkratší cesta složená z dvou kružnicových oblouků a spojující úsečky, která je tečnou obou těchto oblouků. Předpokládáme, že vzdálenost v rovině  $xy$  mezi  $P_0$  a  $P_f$  je větší než  $2/c_{max}$ , jinak vytvořená dráha může být pouze suboptimální. Oba kružnicové oblouky mají poloměr  $r_D$  daný podle inverze maximálního zakřivení  $c_{max}$ . Optimální cesta může být získána z následující geometrické konstrukce.

1. Zvažme dva obvody kružnice  $\sum_{0A}$  a  $\sum_{0B}$  s poloměrem  $r_D$  obsahující výchozí bod  $P_0$  a mající v tomto bodě tečnu jejíž směr je určen podle úhlu  $\psi_0$ . Navíc zvažme obdobné obvody kružnic  $\sum_{fA}$  a  $\sum_{fB}$  obsahující konečný bod  $P_f$ .
2. Dostaneme páry  $(\sum_{0A}, \sum_{fA}), (\sum_{0A}, \sum_{fB}), (\sum_{0B}, \sum_{fA}), (\sum_{0B}, \sum_{fB})$ , které získáme kombinací obvodu kružnice obsahující počáteční bod a obvodu kružnice obsahující koncový bod. Vyberem například pár  $(\sum_{0A}, \sum_{fA})$ . K dispozici jsou čtyři přímky, které jsou tečné k oběma kružnicím  $\sum_{0A}$  a  $\sum_{fA}$ .
3. Nyní můžeme uvažovat čtyři cesty spojující  $P_0$  a  $P_f$ . Každá cesta je tvořena počátečním kružnicovým obloukem  $\sum_{0A}$ , koncovým kružnicovým obloukem  $\sum_{fA}$  a jednou ze čtyř přímek, které jsou tečné k oběma obloukům. Pouze jedna z těchto cest je optimální k počátečnímu a koncovému směru UAV. Každou z těchto vzniklých cest pojmenujeme  $\Gamma_{AA}, \Gamma_{AB}, \Gamma_{BA}$  a  $\Gamma_{BB}$ .
4. Optimální cesta je ta nejkratší z  $\Gamma_{AA}, \Gamma_{AB}, \Gamma_{BA}$  a  $\Gamma_{BB}$ .



Obrázek 3.1: Čtyři možné cesty vytvořené pomocí Dubinsových křivek. Obrázek je převzat z [1].

Použijeme Dubins algoritmus k získání prvního odhadu 3D cesty následujícím způsobem.

1. Počáteční a koncová podmínka na  $\gamma$  jsou zanedbány. Optimální 2D cesta  $\Gamma_{xy}$  v rovině  $xy$  je vytvořena tak, aby vyhovovala počáteční a koncové podmínce  $x, y$  a  $\psi$ . Musí také vyhovovat omezení  $c(s) = \eta(s) \leq c_{max}$ .

2. Pomocí  $d_{xy}$ , který je roven délce  $\Gamma_{xy}$ , určíme úhel

$$\gamma_{xy} = \arctan\left(\frac{z_f - z_0}{d_{xy}}\right), \quad (3.10)$$

který umožňuje přechod z výšky  $z_0$  na výšku  $z_f$ .

3. Při získání úhlu dráhy letu  $\gamma$ , vzniknout dvě možnosti

- (a) splňuje požadavky omezení (3.8),
- (b) nesplňuje omezení (3.8).

Když nastane první případ, opustíme tuto fázi algoritmu. V druhém případě je třeba zvýšit  $d_{xy}$ . Toho lze dosáhnout výběrem jiné cesty v rovině  $xy$  (Pamatujme, že 2D Dubins circle algoritmus má čtyři kandidáty na cestu) nebo návrat do prvního kroku s nižší hodnotou pro  $c_{max}$ .

Nyní se nacházíme v bodě, kdy máme 2D cestu začínající v  $P_0$  a končící v  $P_f$ . Tato cesta ovšem nesplňuje počáteční a koncovou podmínku na úhel dráhy letu  $\gamma$ . V následující kapitole ukážeme, jak toto odstanit.

### 3.1.2 Vytvoření 3D cesty

Rozdělíme 3D cestu  $\Gamma$  do tří podcest  $\Gamma_1, \Gamma_2$  a  $\Gamma_3$ .

1. Podcesta  $\Gamma_1$  je oblouk kružnice takový, že úhel kursu  $\psi$  je konstantní a rovná se  $\psi_0$  (počáteční úhel kursu) a úhel dráhy letu  $\gamma$  se mění lineárně s křivostí úsečky začínající od počáteční hodnoty  $\gamma_0$  a končící v hodnotě  $\bar{\gamma}$ , která bude stále konstantní v následné podcestě  $\Gamma_2$ . Poznamenejme, že výchozí bod cesty  $\Gamma_1$  splňuje počáteční podmínku na  $\Gamma$ .
2. Podcesta  $\Gamma_2$  je obsažena v rovině, kde úhel dráhy letu je konstantní (a rovná se  $\bar{\gamma}$ ), zatímco úhel směru  $\psi$  se mění mezi  $\psi_0$  a  $\psi_f$ . Tato cesta je vytvořena použitím optimálního algoritmu popsáným v sekci (3.1.1).
3. Podcesta  $\Gamma_3$  je znovu oblouk kružnice, kde úhel kursu je konstantní a končí v hodnotě  $\psi_f$ . Zatímco úhel dráhy letu  $\gamma$  se mění mezi  $\bar{\gamma}$  a  $\gamma_f$ . Koncový bod  $\Gamma_3$  splňuje koncovou podmínku.

Pro úplnou charakterizaci všech tří podcest, musíme určit úhel  $\bar{\gamma}$ . Za tímto účelem vyhodnotíme počáteční odhad. Nechť

$$\gamma_{est} = \begin{cases} \gamma_{min} & \text{jestliže } \arctan\left(\frac{z_f - z_0}{\bar{d}}\right) \leq \gamma_{min} \\ \gamma_{max} & \text{jestliže } \arctan\left(\frac{z_f - z_0}{\bar{d}}\right) \geq \gamma_{max} \\ \arctan\left(\frac{z_f - z_0}{\bar{d}}\right) & \text{jinak} \end{cases}$$

kde

$$\bar{d} = \sqrt{(x_f - x_0)^2 + (y_f - y_0)^2}.$$

Cesta  $\Gamma_1$  generuje  $\bar{\gamma} = \gamma_{est}$ . Cesta  $\Gamma_2$  je vytvořena použitím algoritmu popsáného v sekci (3.1.1) a začíná z koncového bodu  $\Gamma_1$ . Může se objevit i nespojistost na  $\gamma$  v důsledku

rozdílu mezi  $\gamma_{est}$  a  $\gamma_{xy}$ , který je získán z rovnice (3.10). Tento rozdíl může být snížen vztahem  $\bar{\gamma} = \gamma_{xy}$  a přepočítáním cest  $\Gamma_1$  a  $\Gamma_2$ . Podle zkušeností se po jedné iteraci stane nespojitost na  $\gamma$  zanedbatelná. Podobný postup může být přijat pro výpočet  $\Gamma_3$  s cílem získat hladký přechod mezi  $\Gamma_2$  a  $\Gamma_3$ . Nakonec se výsledná 3D cesta získá z rovnice  $\Gamma = \Gamma_1 \cup \Gamma_2 \cup \Gamma_3$ .

Tento popsaný algoritmus je efektivní (tj. není daleko od optimální cesty), pakliže je vzdálenost mezi  $P_0$  a  $P_f$  v rovině  $xy$  je velká. V opačném případě nemusí být vytvořena cesta optimální. Aby tento problém byl odstraněn je třeba modifikovat algoritmus, aby se vytvořila podcesta, která bude mít tvar válcové spirály. Bohužel podrobnosti k této modifikaci nejsou v práci [1] uvedeny.

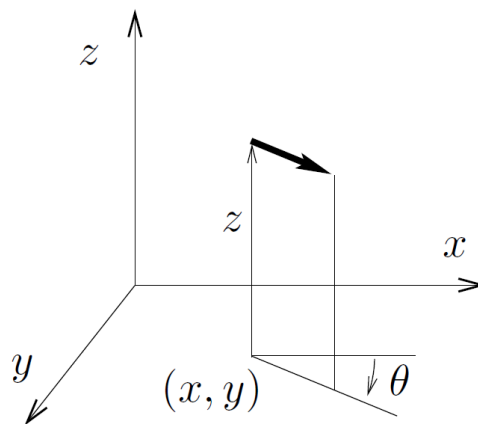
### 3.1.3 Minimální množina parametrů definované cesty

Jednou z charakteristik vytvořených cest podle navrhovaného algoritmu je, že se skládají z přímků a kružnic/oblouků konstantních poloměrů. Každá z těchto částí je zcela určena

- souřadnicemi počátečního bodu,
- hodnota zakřivení úsečky odpovídající tomuto počátečnímu bodu,
- hodnota  $\psi$  v počátečním bodě,
- konstantní hodnota derivace  $\psi$ ,
- hodnota  $\gamma$  v počátečním bodě,
- konstantní hodnota derivace  $\gamma$ .

## 3.2 Optimální cesta pro Dubinsovo letadlo

Dubinsovo letadlo rozšiřuje model Dubinsova auta, jenž má konfigurační prostor  $(x, y, \theta) \in \mathbb{R}^2 \times \mathbb{S}^1$ , o přidanou konfigurační proměnnou pro výšku  $z$ . Z toho vyplývá, že Dubinsovo letadlo je čtyřdimensionální systém s konfigurační proměnnou definovanou jako  $q = (x, y, z, \theta) \in C = \mathbb{R}^3 \times \mathbb{S}^1$ , kde  $x, y$  a  $z$  jsou souřadnice letadla ve třidimenzionálním Euklidovském prostoru a  $\theta \in (0, \pi)$  je úhel mezi osou  $x$  a místní podélnou osou v rovině  $xy$ , jak je ukázáno na obrázku 3.2. Takto popsaný model lze nazvat jako jednoduchý model letadla. Tento systém



Obrázek 3.2: Model Dubinsova letadla. Obrázek převzat z [2].

má nezávislou vymezenou kontrolu nad rychlostí stoupání  $u_z$ , stejně tak jako nad rychlostí zatáčení  $u_\theta$ . Jinými slovy lze systém popsat jako

$$\dot{q} = f(q, u) = f_0(q) + u_z f_1(q) + u_\theta f_2(q), \quad (3.11)$$

kde  $f_0$ ,  $f_1$  a  $f_2$  jsou vektorová pole v konfiguračním prostoru. Předpokládejme, že minimální poloměr otáčení a maximální rychlost stoupání jsou 1. V tom případě  $f_0$ ,  $f_1$  a  $f_2$  jsou

$$f_0 = \begin{pmatrix} \cos \theta \\ \sin \theta \\ 0 \\ 0 \end{pmatrix}, f_1 = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}, f_2 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}. \quad (3.12)$$

Díky využití Pontrjagova principu maxima lze charakterizovat časově optimální cestu pro popsany model letadla. Výsledná cesta se skládá z kruhových oblouků o minimálním poloměru, rovných úseček a z kusů rovinné elastica. Více o Pontrjagovu principu maxima lze nalézt v práci [2].

### 3.2.1 Optimální trajektorie pro letadlo

Bez ztráty obecnosti můžeme předpokládat, že počáteční konfigurace systému je  $q_i = (0, 0, 0, 0) \in C$  a cílová konfigurace systému je  $q_g = (x_g, y_g, z_g, \theta_g)$ . Zavedeme pojem Dubinsova vzdálenost  $\Delta$ , která značí dobu trvání nebo délku nejkratší Dubinsovi křivky od  $q_i$  do  $q_g$ .

Cílová výška hraje důležitou roli při určení optimální trajektorie. Rozlišujeme tři případy výšek.

#### Nízká výška

Výšku nazveme nízkou pokud platí, že  $|z_g| \leq \Delta$ . Pro nízkou cílovou výšku se optimální trajektorie (3.11) sestává z nejkratší Dubinsovi křivky a s rychlosti stoupání  $u_z = \frac{z_g}{\Delta}$ . Doba trvání takové trajektorie je  $\Delta$ .

#### Vysoká výška

Výšku nazveme vysokou pokud platí, že  $|z_g| \geq \Delta + 2\pi$ , kde  $2\pi$  znamená délku spirály za předpokladu, že minimální poloměr otáčení je 1. Pro vysokou cílovou výšku je optimální trajektorie (3.11) složená ze dvou částí. Pro obě části platí  $u_z = \text{sgn}(z_g)$ . Projekce první části do prostoru  $(x, y, \theta)$  je nejkratší Dubinsova křivka pro  $(x_g, y_g, \theta_g)$ . Druhá část je spirála, pro kterou platí  $u_\theta = \frac{2\pi}{|z_g| - \Delta}$ . Systém tedy první provede nejkratší Dubinsovu křivku s plnou rychlostí stoupání a potom provede spirálu, tj. celý kruh s plnou rychlostí stoupání. Délka takové trajektorie je  $|z_g|$ .

#### Střední výška

Zde existují dva případy. Buď cesta z počáteční konfigurace do cílové konfigurace je délky  $|z_g|$ , potom optimální trajektorie odpovídá  $u_z = 1$  nebo  $u_z = -1$ . Nebo cesta z počáteční konfigurace do cílové konfigurace není délky  $|z_g|$ , potom optimální trajektorie musí odpovídat  $u_z \in \langle -1, 1 \rangle$ .

### 3.3 Efektivní dvoufázové 3D plánování pohybu pro malé UAV

V této práci je plánování pohybu rozděleno na globálního plánování a lokálního plánování.

#### 3.3.1 Globální mřížkový plánovač s proveditelnými spojeními

Používá se diskretizované prostředí pomocí mřížky reprezentující prostředí. Globální plánovač pro UAV tedy pracuje na 3D mřížce, kde uzel je voxel. Proveditelná cesta vznikne propojením dvou uzlů, které spolu sousedí v závislosti na rodiči.

##### Definice uzlu a propojení uzlu

Uzel pro globální plánování je definován svou pozicí na 3D mřížce a směrem od svého rodiče (rodičovské propojení). Existuje 26 možných způsobů propojení s rodičovským uzlem. Takové propojení lze kódovat úhlem vychýlení ( $\phi$ ) a úhlem stoupání ( $\theta$ ) letadla. Z toho vyplývá, že uzel odpovídá bodu v diskretizovaném prostoru  $C = (x, y, z, \phi, \theta)$ . Úhel náklonu letadla ( $\psi$ ) se nenachází v definovaném prostoru  $C$ , protože neexistuje žádný způsob, jak jej zde zohlednit.

Propojení uzlu lze definovat libovolně, ale může být zpracován tak, aby odrážel kinematické omezení letounu. Horizontální propojení je definováno jako množina propojení, které obsahuje úhly s rodičovským uzlem menším než  $90^\circ$  a vzniklé propojení je vodorovné. Toto propojení je fyzicky přiměřené, neboť letadlo se pohybuje s minimální změnou rychlosti. Pro propojení uzlů vertikálním způsobem existuje několik možností. Volba závisí na vertikálním manévrovatelnosti letounu.

##### Nastavení velikosti buňky mřížky

Předpokládejme, že v globálním plánovači jsou hlavními kinematickými omezeními na UAV minimální horizontální poloměr otáčení ( $R_{h,min}$ ) a maximální úhel stoupání ( $\mu_{max}$ ). Opakováním zatáčení na jednu stranu v horizontální poloze se vytváří kruh, jehož minimální poloměr je jeden a půl krát velikost mřížky. Proto můžeme nastavit velikost mřížky na  $x(S_x)$  a  $y(S_y)$  na dvě třetiny  $R_{h,min}$ .

Výška buňky  $z(S_z)$  se rozhoduje podle maximálního úhlu stoupání  $\mu_{max}$  a výběrem vertikálního uzlu.

##### Předvýpočet optimální cesty hodnocené podle heuristiky

Ve vyhledávacím grafu globálního plánovače nelze Euclidovu vzdálenost považovat za dobrou heuristiku, protože uzel má směr a pozici ve 3D mřížce.

Proto lze vypočítat optimální hodnotu cesty přes nějakou oblast kolem původu (například mřížka  $9 \times 9 \times 9$ ). Když je heuristika potřebná v aktuálním uzlu, oblast předvýpočtu se přesune do aktuálního uzlu. V případě, že cíl je mimo tuto oblast, tak se nejprve nalezne uzel, který je nejbližší k cíli, se stejným rodičovským uzlem v této oblasti. Potom se vypočítá Euclidova vzdálenost od cíle k uzlu k předvýpočtu ceny tohoto uzlu. Předpočítání heuristiky pomáhá snížit počet navštívených uzlů.

#### 3.3.2 Lokální runtime plánování podle pohybových primitiv

Lokální plánování má upřesnit globální cestu lokálně v případě, když po vytvoření cesty globálním plánovačem dojde ke změně prostředí (přidání překážky). Lokální plánování propojí

dvě konfigurace jemným rozlišením. K určení nejlepší cesty z husté množiny pohybových primitiv se použije jako heuristika Dubinsovi křivky, které dostatečně napodobují kinematické vlastnosti letadla. Prostor  $C$  je stejný jako v případě globálního plánování  $C = (x, y, z, \phi, \theta)$ . Úhel náklonu letadla je stejně jak v předchozím případě ignorován.

## Kapitola 4

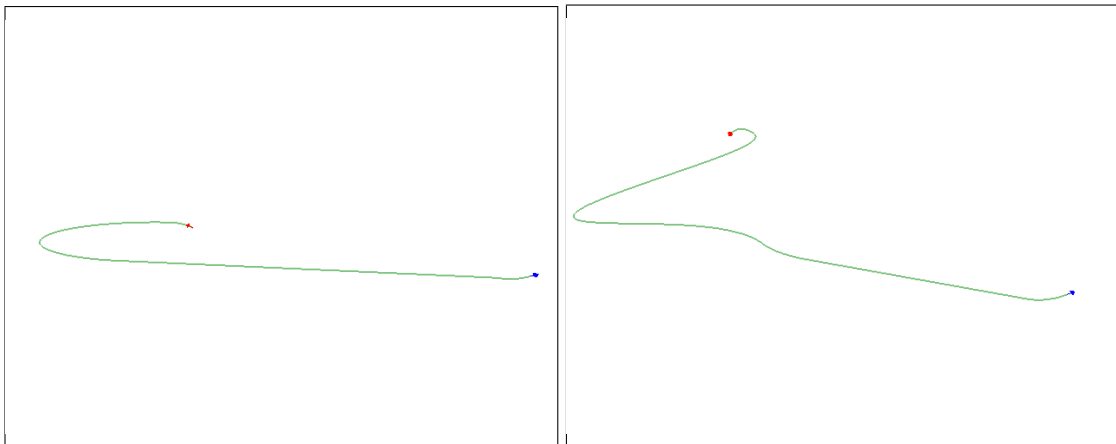
# Zvolené metody a jejich popis implementace

V této kapitole bude popsáno, jak byly implementovány jednotlivé části vybraných metod popsané v kapitole (3) a jejich další rozšíření.

K zobrazení implementovaných metod používám grafický 3D engine Irrlicht<sup>1</sup>. Irrlicht je open source a je kompletně multiplatformní. Pro vykreslení používá D3D, OpenGL a vlastní softwarový render. Má aktivní komunitu a je používán v mnoha projektech.

### 4.1 Algoritmus pro 3D UAV - Vytváření cesty a následování

V následujících sekcích bude vysvětleno a na obrázcích ukázáno, jak byly implementovány jednotlivé části této metody. Jedná se o implementaci počátečního oblouku, koncového oblouku, Dubinových křivek a spirály.



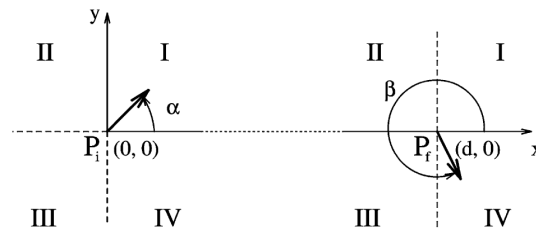
Obrázek 4.1: Ukázka cesty vytvořené popisovanou metodou. Obrázek nalevo ukazuje cestu bez spirály a obrázek vpravo se spirálou.

---

<sup>1</sup>Viz <http://irrlicht.sourceforge.net>

### 4.1.1 Implementace Dubinsových křivek

Následující sekce vychází z práce [7]. Pro implementaci Dubinsových křivek je potřeba mít alespoň dva body v rovině, tedy počáteční a koncový bod. Počáteční bod označíme jako  $P_i = (x, y)$  a koncový bod si označíme jako  $P_f = (x, y)$ . Jak počáteční tak i koncový bod má směr pohybu, které si označíme  $\alpha$  a  $\beta$ . Z těchto dvou informací získáme počáteční konfiguraci  $(P_i, \alpha)$  a cílovou konfiguraci  $(P_f, \beta)$ . Nechť Euklidova vzdálenost mezi body  $P_i$  a



Obrázek 4.2: Znázornění počáteční konfigurace  $(P_i, \alpha)$  a cílové konfigurace  $(P_f, \beta)$  v souřadnicovém systému. Obrázek je převzat z [7].

$P_f$  bude značena jako  $d$ . Jak je znázorněno na obrázku 4.1 má počáteční bod  $P_i$  souřadnice  $(0,0)$  a cílový bod  $P_f$  má souřadnice  $(d,0)$ . Bez ztráty obecnosti můžeme předpokládat, že minimální poloměr otáčení  $\rho = 1$ . V případě, že  $\rho \neq 1$ , je zapotřebí přepočítat délku  $d$  tak, aby platilo  $\rho = 1$ . Toho lze dosáhnout pomocí rovnice

$$d = D/\rho, \quad (4.1)$$

kde  $D$  je aktuální Euklidova vzdálenost mezi body  $P_i$  a  $P_f$ .

#### Převedení souřadnic

Protože v další podkapitole budeme vycházet, že počáteční bod má souřadnice  $P_i = (0,0)$  a cílový bod má souřadnice  $P_f = (d,0)$ , kde  $d$  se vypočítá podle rovnice (4.1) a  $\rho = 1$ , je potřeba upravit souřadnice. Je třeba získat úhel, který svírá spojnice mezi počátečním bodem  $P_i$  a cílovým bodem  $P_f$  a osou  $x$ . Ten si označíme jako  $\gamma$  a získáme ho z rovnice

$$\gamma = \arctan \frac{y_i - y_f}{x_i - x_f}. \quad (4.2)$$

Vypočítaný úhel  $\gamma$ , pak odečteme od počátečního směru pohybu  $\alpha$  a cílového směru pohybu  $\beta$ .

Na závěr je potřeba upravit získanou křivku podle zadaného minimálního poloměru otáčení  $\rho_v$  a počáteční pozice  $P_i = (x_i, y_i)$ . Toho dosáhneme použitím jednoduchých rovnic

$$x(t) = x(t)\rho_v + x_i, \quad (4.3)$$

$$y(t) = y(t)\rho_v + y_i, \quad (4.4)$$

kde  $t$  je čas. Jak získat konfigurace pojednává následující podkapitola.

#### Proveditelné cesty a jejich specifikace

K určení proveditelné cesty zavedeme tři elementární pohyby. Otočení doleva, otočení do prava (oba podél kruhu  $C$  o poloměru 1) a rovný úsek  $S$ . Budem také potřebovat tři



odpovídající operace:  $L_v$  (pro otočení doleva),  $R_v$  (pro otečení doprava),  $S_v$  (pro cestu rovně), které transformují libovolný bod  $(x, y, \theta) \in \mathbb{R}^3$  do odpovídajícího myšleného bodu v  $\mathbb{R}^3$

$$\begin{aligned} L_v(x, y, \theta) &= (x + \sin(\theta + v) - \sin \theta, y - \cos(\theta + v) + \cos \theta, \theta + v), \\ R_v(x, y, \theta) &= (x - \sin(\theta - v) + \sin \theta, y + \cos(\theta - v) - \cos \theta, \theta - v), \\ S_v(x, y, \theta) &= (x + v \cos \theta, y + v \sin \theta, \theta), \end{aligned} \quad (4.5)$$

kde  $v$  značí pohyb podél ( $C$  nebo  $S$ ) segmentu délky  $v$ . S těmito elementárními transformacemi může být vyjádřena libovolná cesta z Dubinsovi množiny  $\mathcal{D} = LSL, RSR, RSL, LSR$  ( $LRL$  a  $RLR$  jsou zde vynechány, protože nejsou popsány v (3.1.1)) podle příslušných rovnic. Ve vybraném souřadnicovém systému je počáteční konfigurace každé cesty  $(0, 0, \alpha)$  a cílová konfigurace  $(d, 0, \beta)$ . Například cesta, která začíná v bodě  $(0, 0, \alpha)$  a která je tvořena segmenty  $L, R$  a  $L$ , jejichž délky jsou  $t, p, q$ , musí končit v  $L_q(R_p(L_t(0, 0, \alpha))) = (d, 0, \beta)$ . Délka cesty  $L$  může být definována jako součet délek  $t, p$  a  $q$ .

$$\mathcal{L} = t + p + q \quad (4.6)$$

Následuje popis jak vypočítat délku jednotlivých cest z množiny

$$1. L_q(S_p(L_t(0, 0, \alpha))) = (d, 0, \beta)$$

$$\begin{aligned} t_{lsl} &= -\alpha + \arctan \frac{\cos \beta - \cos \alpha}{d + \sin \alpha - \sin \beta}, \\ p_{lsl} &= \sqrt{2 + d^2 - 2 \cos(\alpha - \beta) + 2d(\sin \alpha - \sin \beta)}, \\ q_{lsl} &= \beta - \arctan \frac{\cos \beta - \cos \alpha}{d + \sin \alpha - \sin \beta}. \end{aligned} \quad (4.7)$$

Použití definice (4.6) získáme délku cesty  $LSL$  jako funkci

$$\mathcal{L}_{lsl} = t_{lsl} + p_{lsl} + q_{lsl}. \quad (4.8)$$

$$2. R_q(S_p(R_t(0, 0, \alpha))) = (d, 0, \beta)$$

$$\begin{aligned} t_{rsr} &= \alpha - \arctan \frac{\cos \alpha - \cos \beta}{d - \sin \alpha + \sin \beta}, \\ p_{rsr} &= \sqrt{2 + d^2 - 2 \cos(\alpha - \beta) + 2d(\sin \beta - \sin \alpha)}, \\ q_{rsr} &= -\beta + \arctan \frac{\cos \alpha - \cos \beta}{d - \sin \alpha + \sin \beta}, \end{aligned} \quad (4.9)$$

a délka cesty je dána podle

$$\mathcal{L}_{rsr} = t_{rsr} + p_{rsr} + q_{rsr}. \quad (4.10)$$

$$3. R_q(S_p(L_t(0, 0, \alpha))) = (d, 0, \beta)$$

$$\begin{aligned} t_{rsl} &= -\alpha + \arctan \frac{-\cos \alpha - \cos \beta}{d + \sin \alpha + \sin \beta} - \arctan \frac{-2}{p_{rsl}}, \\ p_{rsl} &= \sqrt{-2 + d^2 + 2 \cos(\alpha - \beta) + 2d(\sin \alpha + \sin \beta)}, \\ q_{rsl} &= -\beta + \arctan \frac{-\cos \alpha - \cos \beta}{d + \sin \alpha + \sin \beta} - \arctan \frac{-2}{p_{rsl}}, \end{aligned} \quad (4.11)$$

a délka cesty je dána podle

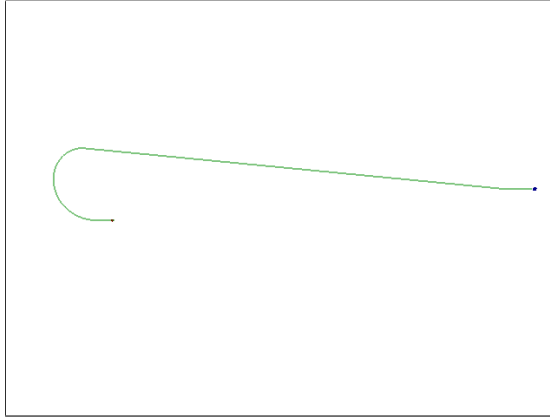
$$\mathcal{L}_{rsl} = t_{rsl} + p_{rsl} + q_{rsl}. \quad (4.12)$$

$$4. L_q(S_p(R_t(0, 0, \alpha))) = (d, 0, \beta)$$

$$\begin{aligned} t_{lsr} &= \alpha - \arctan \frac{\cos \alpha + \cos \beta}{d - \sin \alpha - \sin \beta} + \arctan \frac{2}{p_{lsr}}, \\ p_{lsr} &= \sqrt{-2 + d^2 + 2 \cos(\alpha - \beta) - 2d(\sin \alpha + \sin \beta)}, \\ q_{lsr} &= \beta - \arctan \frac{\cos \alpha + \cos \beta}{d - \sin \alpha - \sin \beta} + \arctan \frac{2}{p_{lsr}}, \end{aligned} \quad (4.13)$$

a délka cesty je dána podle

$$\mathcal{L}_{lsr} = t_{lsr} + p_{lsr} + q_{lsr}. \quad (4.14)$$



Obrázek 4.3: Ukázka Dubinsovi křivky.

### Transformace do 3D prostoru

Implementace Dubinsových křivek uvedená výše je pouze v 2D prostoru a je proto třeba ji převést do 3D prostoru. K tomu potřebujeme úhel  $\gamma_{xy}$ , který získáme z rovnice (3.10) a který je konstantní podle podsekcce (3.1.2). Použití transformace (4.5) a úhlu  $\gamma_{xy}$  získáme výslednou transformaci do 3D prostoru.

$$\begin{aligned} L_v(x, y, z, \theta, \gamma) &= (x + \sin(\theta + v) - \sin \theta, y - \cos(\theta + v) + \cos \theta, v \tan \gamma, \theta + v, \gamma), \\ R_v(x, y, z, \theta, \gamma) &= (x - \sin(\theta - v) + \sin \theta, y + \cos(\theta - v) - \cos \theta, v \tan \gamma, \theta - v, \gamma), \\ S_v(x, y, z, \theta, \gamma) &= (x + v \cos \theta, y + v \sin \theta, v \tan \gamma, \theta, \gamma), \end{aligned} \quad (4.15)$$

Ukázka, jak vypadá transformace Dubinsovi křivky do 3D prostoru, je na obrázku 4.1

#### 4.1.2 Implementace počátečního a koncového kruhového oblouku

Kruhový oblouk je část kružnice o velikosti úhlu  $\alpha$ . K sestavení kruhového oblouku potřebujeme buď tři body (počáteční, upřesňující a koncový) nebo některou jinou charakteristiku (např. poloměr kružnice, velikost úhlu  $\alpha$  apod.). V našem případě známe počáteční bod, poloměr kružnice a velikost úhlu  $\alpha$  si můžeme dopočítat.

Jak počáteční, tak koncový kruhový oblouk byl implementován podle podkapitoly (3.1.2). Jeho jednotlivé body v čase  $t$  získáme z rovnic (3.1 - 3.3), což jsou rovnice pro výpočet koule.

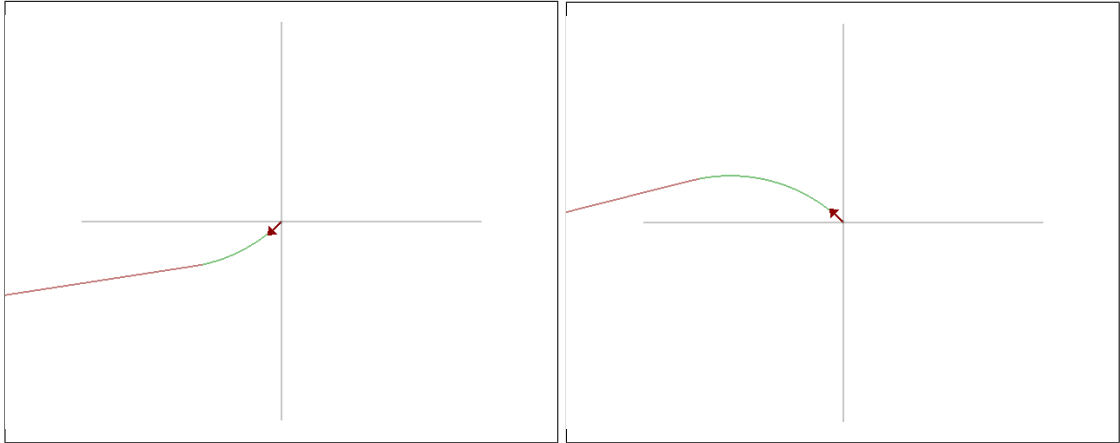
Jelikož úhel  $\psi$  je konstantní a mění se pouze úhel  $\gamma$ , získáme kruhový oblouk. Nesmíme zapomenout upravit úhel dráhy letu  $90^\circ - \gamma(s)$ , jak bylo zmíněno zde (3.1).

Otázkou zůstává, jak vypočítat velikost úhlu  $\alpha$ . Ten získáme z jednoduché rovnice

$$\alpha = \gamma_0 - \gamma_{xy}. \quad (4.16)$$

Tato rovnice platí i pro koncový kruhový oblouk.

Úhel  $\gamma_0$  se může nacházet buď v prvním kvadrantu (úhel  $\gamma_0 > 0$ ) a nebo ve čtvrtém kvadrantu (úhel  $\gamma_0 < 0$ ), jak je ukázáno na obrázku 4.4. Z toho vyplývá že vztah  $90^\circ - \gamma(s)$  nemůže platit pro situaci, kdy je  $\gamma_0 < 0$ . Proto je třeba upravit tento vztah, aby platil i pro čtvrtý kvadrant  $\gamma_0 = -(90 + \gamma_0)$ . Jak rozhodnout, kdy použít jaký vztah je popsáno níže.



Obrázek 4.4: Ukázka počátečního kruhového oblouku, kde kruhový oblouk je zeleně zvýrazněn. Obrázek nalevo znázorňuje situaci, kde je  $\gamma_0 < 0$ .

Z předchozích odstavců v této sekci lze sestavit jednoduchý algoritmus pro vytvoření jak počátečního kruhového oblouku, tak koncového kruhového oblouku. Následující popis algoritmu je pro počáteční kruhový oblouk.

1. Nejprve musíme upravit úhel  $\psi$  a posunout ho o  $180^\circ$ .
2. Na základě znaménka úhlu  $\alpha$  lze určit v jakém kvadrantu se oblouk nachází. Podle toho v jakém se kruhový oblouk kvadrantu nachází, můžeme vhodně upravit úhel  $\gamma_0$  (v případě koncového kruhového oblouku úhel  $\gamma_f$ ).

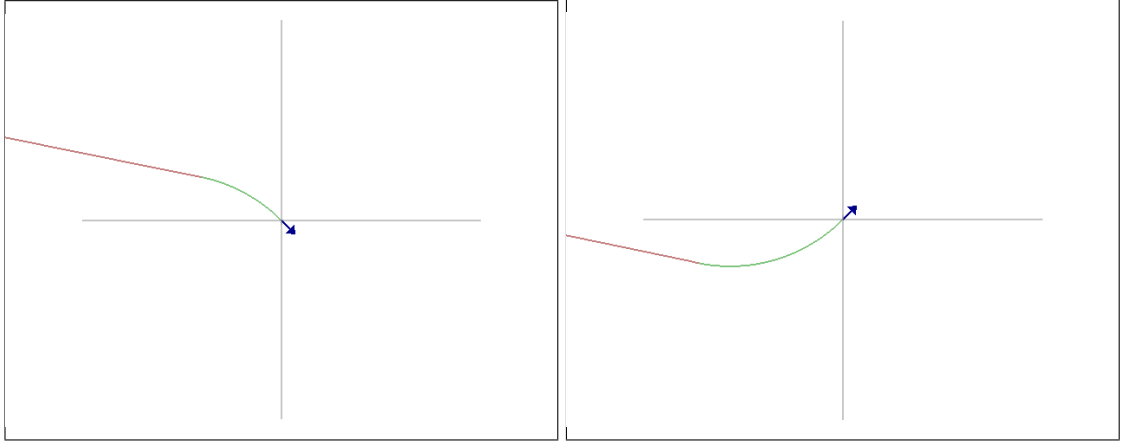
- (a) Je-li úhel  $\alpha$  záporný, tak úhel  $\gamma_0$  musíme upravit podle následující rovnice

$$\gamma_0 = -(90 + \gamma_0). \quad (4.17)$$

- (b) Je-li úhel  $\alpha$  kladný, pak použijeme rovnici

$$\gamma_0 = 90 - \gamma_0. \quad (4.18)$$

Z obrázku 4.5 je patrné, že úhel  $\gamma_f$  se nachází v opačných kvadrantech než úhel  $\gamma_0$ . Proto je třeba upravit výše uvedený algoritmus pro koncový kruhový oblouk. Toho lze dosáhnout jednoduchou úpravou, kde se přesune krok 2b před krok 2a.



Obrázek 4.5: Ukázka koncového kruhového oblouku, kde kruhový oblouk je zeleně zvýrazněn. Obrázek nalevo znázorňuje situaci, kde je  $\gamma_f < 0$ .

### 4.1.3 Implementace spirály

Spirála je rovinná křivka, která se otáčí kolem pevného bodu konstantní rychlostí ve vzdálenosti  $\rho$ . Vzdálenost  $\rho$  je funkcí velikosti úhlu  $\alpha$ , tedy  $\rho = f(\alpha)$ . Předtím než použijeme spirálu v následujícím odstavci, stanovíme, že  $\rho$  je konstantní.

K implementaci spirály budeme tedy potřebovat úhel  $\psi$ , který se bude měnit v čase z  $\psi_0$  na  $\psi_0 + 2\pi$ , konstantní úhel  $\gamma$  a počáteční bod  $P_0$ . Ještě budeme potřebovat velikost minimálního poloměru otáčení, což je v našem případě  $\rho$ . Jednotlivé body dostaneme z následujících rovnic

$$x = \cos(\pi + t)\rho + x_0, \quad (4.19)$$

$$y = \sin(\pi + t)\rho + y_0, \quad (4.20)$$

$$z = \tan(\gamma)t\rho + z_0, \quad (4.21)$$

kde  $t$  značí pohyb po spirále.

Z výše uvedených odstavců musíme upravit algoritmus pro cestu  $\Gamma_2$ , který byl popsán v podkapitole (3.1.2). Upravený algoritmus můžeme popsat v následujících bodech.

1. První krok algoritmu je stejný jak v podkapitole (3.1.1).
2. Určíme si proměnnou  $c_{spiral}$ , která bude vyjadřovat počet užitých spirál a nastavíme jí na  $c_{spiral} = 0$ .
3. Určíme délku spirály  $d_{spiral} = 2\pi\rho$  a celkovou délku  $d_{total}$ , kterou dostaneme z

$$d_{total} = d_{dubins} + c_{spiral}d_{spiral}, \quad (4.22)$$

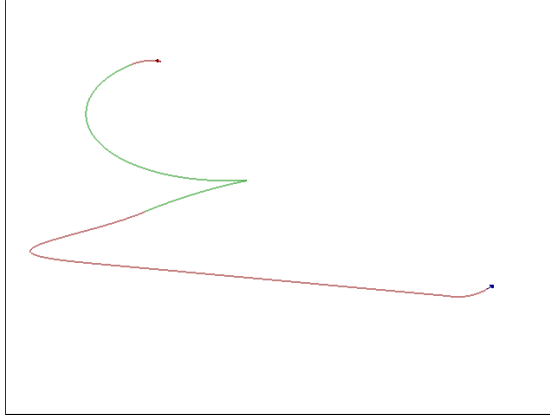
kde  $d_{dubins}$  představuje délku zvolené Dubinsovi křivky.

4. Pomocí  $d_{total}$ , který je roven délce  $\Gamma_2$ , určíme úhel

$$\gamma_{xy} = \arctan\left(\frac{z_f - z_0}{d_{total}}\right), \quad (4.23)$$

který umožňuje přechod z výšky  $z_0$  na výšku  $z_f$ .

5. Pokud úhel  $\gamma_{xy}$ , splňuje požadavky omezení (3.8), ukončíme algoritmus. Jinak pokračuj.
6. Vybereme další nejkratší křivku (máme čtyři kandidáty na cestu) a vrátíme se do třetího kroku. Pokud jsme vyzkoušeli všechny cesty a žádná nesplňuje omezení (3.8), tak pokračuj na další krok.
7. Zvětši, o jedna proměnnou  $c_{spiral} = c_{spiral} + 1$  a vrať se na krok tři.



Obrázek 4.6: Ukázka spirály, kde spirála je zeleně znázorněna.

Jiný algoritmus pro spirálu vychází jak z podkapitoly (3.2.1), tak také z podkapitoly (3.1.2). Výsledný algoritmus je

1. První krok algoritmu je stejný jak v podkapitole (3.1.1).
2. Určíme si proměnnou  $d_{best}$ , která bude obsahovat délku nejlepší Dubinsovi křivky.
3. Určíme celkovou délku  $d_{total}$ , kterou dostaneme z rovnice

$$d_{total} = d_{best} + 2\pi\rho, \quad (4.24)$$

kde  $2\pi\rho$ , znamená délku nejmenší možné spirály.

4. Pomocí  $d_{xy}$ , určíme úhel, který je roven délce  $\Gamma_2$ , určíme úhel

$$\gamma_{xy} = \arctan\left(\frac{z_f - z_0}{d_{xy}}\right), \quad (4.25)$$

který umožňuje přechod z výšky  $z_0$  na výšku  $z_f$ .

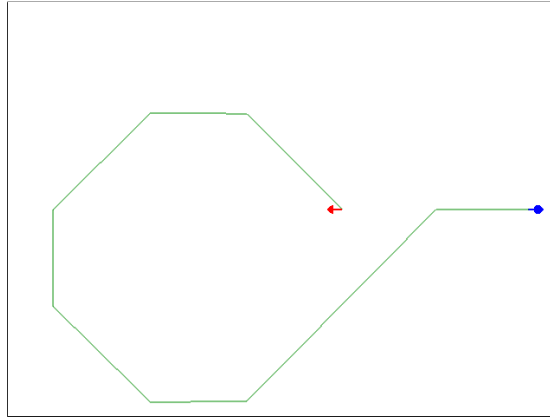
5. Pokud úhel  $\gamma_{xy}$ , splňuje požadavky omezení (3.8), ukončíme algoritmus. Jinak pokračuj.
6. Vybereme další nejkratší křivku (máme čtyři kandidáty na cestu), jejíž délka je menší jak  $d_{total}$  a vrátíme se do čtvrtého kroku. Pokud jsme vyzkoušeli všechny cesty, které splňovaly podmínku, tak pokračuj na další krok.
7. Nastavíme úhel  $\gamma = \gamma_{max}$  (nebo  $\gamma = \gamma_{min}$ ) a vypočítáme délku spirály

$$d_{spiral} = \tan(\gamma_{max})(z_f - z_0) - d_{best}. \quad (4.26)$$

8. Výsledná křivka je potom tvořená z vypočítané spirály a nejkratší Dubinsovi křivky.

## 4.2 Efektivní dvoufázové 3D plánování pohybu pro malé UAV

V této kapitole bude popsáno jak bylo diskretizováno prostředí a jak bude implementována vyhledávací metody.





Obrázek 4.7: Ukázka nalezené cesty diskretní metodou. Cesta je znázorněna zelenou barvou.

Jak je vidět z obrázku 4.7 první krok z počáteční pozice (červená šipka) nevede ve směru šipky ale vede hned k cíli (modrá šipka). Je to dáno tím, že špička šipky neukazuje na počáteční pozici, ale spodek šipky.

### 4.2.1 Diskretizace prostředí

V podkapitole (3.3.1) je prostředí diskretizováno do 3D mřížky, kde jeden uzel mřížky se nazývá Voxel. Diskretizace prostředí do mřížky je jednoduchá a dobře se s tímto prostředím pracuje. Proto bývá často využíváno ve 2D prostředí. Pro určení pozice voxelu v 3D mřížce, lze využít kartézský souřadnicový systém  $(x, y, z)$ . Konfigurace je podle podkapitoly (3.3.1) určena souřadnicemi  $(x, y, z)$  a úhly  $(\psi, \theta)$ , tedy  $C = (x, y, z, \phi, \theta)$ . Na obrázku 4.8 je ukázán pohyb letadla, jak v horizontální, tak ve vertikální rovině. Letadlo se pohybuje ve směru osy  $x$ .

$-1, 1, 0, 135^\circ, 0^\circ$	$0, 1, 0, 90^\circ, 0^\circ$	$1, 1, 0, 45^\circ, 0^\circ$			$1, 0, 1, 0^\circ, 45^\circ$
$-1, 0, 0, 180^\circ, 0^\circ$		$1, 0, 0, 0^\circ, 0^\circ$			$1, 0, 0, 0^\circ, 0^\circ$
$-1, -1, 0, 225^\circ, 0^\circ$	$0, -1, 0, 270^\circ, 0^\circ$	$1, -1, 0, 315^\circ, 0^\circ$			$1, 0, -1, 0^\circ, -45^\circ$

Obrázek 4.8: Ukázka horizontálního (vlevo) a vertikálního (vpravo) pohybu letadla.

## Diskretizace cílové konfigurace

Jak bylo uvedeno v podkapitole (3.3.1) je délka kroku ve směru osy  $x$  a osy  $y$  rovna dvěma třetinám minimálního poloměru otáčení  $R_{h,min}$ . Délka kroku ve směru osy  $z$  závisí na úhlu stoupání  $\theta$  a na délce kroku ve směru osy  $x$  nebo osy  $y$ . Jinými slovy  $z = \tan(\theta)x$ . Když známe délky kroků v jednotlivých směrech, můžeme transformovat cílovou pozici do odpovídajícího voxelu v diskretizovaném prostředí. Transformační rovnice jsou následující

$$\begin{aligned}x_f &= x_0 + \text{round}(\text{abs}(x_f - x_0)/dx)dx, \\y_f &= y_0 + \text{round}(\text{abs}(y_f - y_0)/dy)dy, \\z_f &= z_0 + \text{round}(\text{abs}(z_f - z_0)/dz)dz,\end{aligned}$$

kde **abs** je funkce, která vrací absolutní hodnotu zadaného čísla, funkce **round** zaokrouhlí zadané číslo a  $dx, dy, dz$  jsou délky kroků v jednotlivých osách. Najdeme nejkratší cestu a poté nahradíme poslední konfiguraci zadanou cílovou konfigurací. Jelikož cílová konfigurace nemusí mít stejné souřadnice, jako poslední konfigurace, dochází k mírnému zakřivení výsledné cesty u cíle.

### 4.2.2 Implementace A\*

Tato podkapitola vychází z práce [12]. Jak bylo vysvětleno v podkapitole (2.2.2) je A\* informovaná metoda a je založena na výběru nejlépe ohodnoceného stavu (Best First Search). Je to metoda úplná a optimální. Aby byla metoda A\* co nejméně časově a prostorově náročná, musí být vybrána vhodná heuristika s dobrým spodním odhadem skutečné ceny (odhad musí být menší, než skutečná cena). Je třeba zvážit složitost heuristické funkce, protože se významně promítá do celkové doby výpočtu.

Kvůli výše zmíněným vlastnostem je metoda A\* jedna z nejpoužívanějších metod pro hledání cesty ve 2D prostoru. Její algoritmus je

1. Sestrojte seznam OPEN (bude obsahovat všechny uzly určené k expanzi) a umístěte do něj počáteční uzel včetně jeho ohodnocení.
2. Je-li seznam OPEN prázdný, pak úloha nemá řešení a ukončete proto prohledávání jako neúspěšné. Jinak pokračujte.
3. Vyberte ze seznamu OPEN uzel s nejlepším (nejnižším) ohodnocením.
4. Je-li vybraný uzel uzlem cílovým, ukončete prohledávání jako úspěšné a vraťte cestu od kořenového uzlu k uzlu cílovému (vrací se posloupnost stavů, nebo operátorů). Jinak pokračujte.
5. Vybraný uzel expandujte. Všechny jeho bezprostřední následníky, kteří nejsou jeho předky, umístěte do seznamu OPEN včetně jejich ohodnocení. Z uzlů, které se v seznamu OPEN vyskytují vícekrát, ponechte pouze uzel s nejlepším ohodnocením, ostatní se seznamu OPEN vyškrkněte a vraťte se na bod 2.

K ověření, zda generovaný následník není předkem expandovaného uzlu, se může použít seznam CLOSED.

## Kapitola 5

# Testování a vyhodnocení výsledků

V této kapitole budou provedeny experimenty nad vybranými metodami z kapitoly 4 a dále bude popsána aplikace pro vizualizaci cest.

### 5.1 Popis aplikace

Pro testování a vizualizaci implementovaných metod byla v jazyce *C++* vytvořena aplikace. Tato aplikace dokáže zobrazit počet generovaných uzlů přidaných do seznamu OPEN, délku cesty a vizualizovat výslednou cestu. Měření času potřebného k nalezení cesty je bohužel prováděno ručně použitím unixového programu `time`.

K zadání podmínek počáteční a cílové konfigurace slouží textový soubor `configuration.txt`. Na první řádek tohoto souboru se zadá počáteční konfigurace, na další řádek cílová konfigurace a na poslední řádek se zadají podmínky. Jednotlivé hodnoty se musí oddělit mezerou. U konfigurací se jednotlivé parametry zadávají v pořadí:  $x, y, z, \psi, \gamma$ , kde  $\psi$  a  $\gamma$  se zadávají ve stupních. U podmínek je to: minimální poloměr otáčení,  $\gamma_{max}$ .

Výsledná trajektorie je zobrazená zeleně. Počáteční konfigurace je znázorněná červenou šipkou a cílová konfigurace je znázorněná modrou šipkou. K přesunutí kamery na počáteční konfiguraci slouží klávesnicové tlačítko  $s$  a na cílovou konfiguraci tlačítko  $t$ . K zobrazení souřadnicových os slouží klávesnicové tlačítko  $a$ . Zobrazené osy jsou barevně odlišeny kvůli lepšímu rozlišení. Osa  $x$  má červenou barvu, osa  $y$  má modrou barvu a osa  $z$  má barvu bílou. Klávesnicové tlačítko `esc` vypíná aplikaci. Klávesnicovými tlačítky  $c$  (spojitá metoda) a  $d$  (diskrétní metoda) se přepíná mezi jednotlivými metodami. Pravým tlačítkem myši se mění cílový bod kamery, kolem kterého se stiskem levého tlačítka myši lze otáčet. Stisknutím obou tlačítek myši zároveň se zvolený bod oddaluje a přibližuje.

Jak již bylo zmíněno, čas se měří použitím unixového programu `time`. K měření času jednotlivých metod byl software ručně upravený tak, aby měřil pouze inicializaci metody a následné vyhledání cesty podle zvolené metody.

### 5.2 Popis a vyhodnocení provedených experimentů

Následující experimenty mají za cíl ověřit a vybrat nejlepší způsob plánování optimální trajektorie.



### 5.2.1 Ověření optimálnosti implementovaných spirál

Cílem tohoto experimentu bylo vybrat lepší z algoritmů popsanych v podkapitole (4.1.3). Při tomto experimentu byla ověřována časová náročnost a délka nalezené cesty.

#### Popis experimentu

Experiment byl zkoušen ve třech různých výškách, tak aby byla, co možná nejspolehlivěji, ověřena optimálnost nalezené cesty. Tabulka (5.1) ukazuje konfigurace, které byly použity k testování. Konfigurace  $q_{I_1}$  -  $q_{I_3}$  jsou startovní konfigurace a  $q_f$  je cílová konfigurace. Podmínky použité v experimentu byly  $\gamma_{min} = -15^\circ$ ,  $\gamma_{max} = 20^\circ$  a  $c_{max} = 4$ . Každé měření

	$x$	$y$	$z$	$\psi$	$\gamma$
$q_{i_1}$	-20	-1	50	$45^\circ$	$20^\circ$
$q_{i_2}$	-20	-1	100	$45^\circ$	$20^\circ$
$q_{i_3}$	-20	-1	31	$45^\circ$	$20^\circ$
$q_f$	20	2	10	$0^\circ$	$20^\circ$

Tabulka 5.1: Konfigurace, které byly použity k testování.

potřebného času k nalezení cesty bylo provedeno desetkrát. Do tabulky 5.2 byla zanesena průměrná hodnota  $t$  z naměřených časů a celková vzdálenost  $d$ .

	<i>algm1</i>		<i>algm2</i>	
	$t$ [s]	$d$	$t$ [s]	$d$
1.	0.0077	166.183	0.0072	154.952
2.	0.0074	342.138	0.0076	341.555
3.	0.0093	90.7753	0.0087	84.0432

Tabulka 5.2: Naměřené hodnoty z prováděných experimentů, kde *algm1* představuje algoritmus popsany jako první v podkapitole (4.1.3) a *algm2* představuje algoritmus popsany jako druhý v již zmíněné kapitole.

#### Vyhodnocení měření

Naměřené časy obou metod jsou přibližně stejné. Rozdíl je ve velikostech nalezených cest. Tento rozdíl je dán tím, že algoritmus *algm2* využívá nejkratší Dubinsovu křivku a pouze jednu spirálu, které se s přibývajícím výškou zvětšuje poloměr. Kdežto druhá metoda má přesně danou velikost spirály a v případě potřeby přidá další spirály. Navíc algoritmus *algm1* projde všechny křivky a potom, když žádná nevyhovuje zadaným podmínkám, přidá spirálu. To nemusí vést k optimálnímu řešení, jak je vidět z třetího experimentu.

Při střední výšce (tj. výška, kdy nejde použít nejkratší Dubinsova křivka a nelze použít ani spirálu) nemusí obě metody nalézt optimální cestu. To je způsobeno tím, že následující nejkratší Dubinsova křivka nemusí nalézt optimální cestu.

## 5.2.2 Optimalizace A\*

Tento experiment má za cíl zjistit, zda je optimálnější použít seznam CLOSE nebo zjišťovat, zda bezprostřední následník není předkem v seznamu předků expandovaného uzlu. V druhé části experimentu bude zkoušeno, která heuristika (Euklidova vzdálenost, Manhattanova vzdálenost) je optimálnější. Při tomto experimentu byla ověřována časová náročnost, počet vygenerovaných uzlů do seznamu OPEN a v druhé části i délka nalezené cesty.

### Popis experimentu

V první části experimentu byla použita jako heuristika Euklidova vzdálenost. Cena cesty  $g(n)$  se rovněž počítá jako Euklidova vzdálenost. Tabulka (5.3) ukazuje konfigurace, které byly použity k testování. Délka kroku v ose  $x$  a  $y$  je jedna a úhel stoupání  $45^\circ$ . Každé měření

	$x$	$y$	$z$	$\psi$	$\gamma$
$q_{i_1}$	-10	3	0	$0^\circ$	$45^\circ$
$q_{f_1}$	0	0	5	$180^\circ$	$-45^\circ$
$q_{i_2}$	-6	-10	0	$135^\circ$	$0^\circ$
$q_{f_2}$	0	0	10	$180^\circ$	$0^\circ$
$q_{i_3}$	0	0	0	$135^\circ$	$0^\circ$
$q_{f_3}$	0	0	0	$180^\circ$	$0^\circ$

Tabulka 5.3: Konfigurace, které byly použity k testování.

potřebného času k nalezení cesty bylo provedeno desetkrát. Do tabulky 5.4 byla zanesena průměrná hodnota  $t$  z naměřených časů a počet přidanych uzlů do seznamu OPEN  $n$ .

	<i>S CLOSE</i>		<i>Bez CLOSE</i>	
	$t[s]$	$n$	$t[s]$	$n$
1.	27.8614	24813	68.3404	59458
2.	1.0015	8129	1.7325	12690
3.	0.0752	2234	0.062	2344

Tabulka 5.4: Naměřené hodnoty z provedených experimentů.

V druhé části, jak již bylo zmíněno, se bude zkoušet použitá heuristika. Je zde použit seznam CLOSE k ověření, zda generovaný následník není předkem expandovaného uzlu. Tabulka (5.3) ukazuje konfigurace, které byly použity k testování. Délka kroku ve všech osách je stejná jak v předchozím případě a stejně tak cena cesty  $g(n)$  se rovná Euklidově vzdálenosti. Také i při tomto experimentu bylo každé měření provedeno desetkrát. Výsledky měření ukazuje tabulka 5.5.

### Vyhodnocení experimentů

Tabulka 5.4 jasně dokazuje, že lepší je používat seznam CLOSE. Seznam CLOSE obsahuje všechny expandované uzly narozdíl od druhého způsobu. Nevýhoda použití seznamu CLOSE je při větším počtu kroků, protože jeho velikost roste s každým krokem, což způsobuje pomalejší prohledávání tohoto seznamu. Nadruhou stranu rapidně snižuje počet vygenerovaných uzlů, což tuto nevýhodu vyrovná.

	<i>Euklid</i>			<i>Manhattan</i>		
	$t[s]$	$n$	$d$	$t[s]$	$n$	$d$
1.	27.8614	24813	18.6315	4.1782	13350	18.6315
2.	1.0015	8129	18.6848	0.0492	1452	18.7812
3.	0.0752	2234	9.65685	0.0394	1395	9.65685

Tabulka 5.5: Naměřené hodnoty z provedených experimentů, kde  $n$  je počet vygenerovaných uzlů a  $d$  je délka cesty.

Druhá část experimentu dokázala, že při použití heuristiky Manhattan se vygeneruje menší počet uzlů a zároveň i čas, který je potřebný k nalezení cesty je menší. V tabulce 5.5 si lze však povšimnout, že v druhém experimentu je cesta, která byla nalezena pomocí Euklidovi vzdálenosti, menší než při použití Manhattan.

### 5.2.3 Porovnání diskrétní a spojitě metody

V tomto experimentu budu porovnávat diskrétní a spojitou metodu, které jsem implementoval. Porovnání těchto dvou metod není moc dobře proveditelné, protože tyto metody pracují na různých principech. Proto nelze porovnat délku nalezené cesty, ale pouze čas za který je nalezena cesta.

#### Popis experimentu

Experiment byl zkoušen v několika bodech, aby byly ověřeny vlastnosti zvolených metod. Tabulka 5.6 ukazuje konfigurace, které byly použity k testování. U diskrétní metody byla

	$x$	$y$	$z$	$\psi$	$\gamma$
$q_{i_1}$	0	0	0	180°	0°
$q_{f_1}$	3	0	2	0°	0°
$q_{i_2}$	-10	2	2	180°	0°
$q_{f_2}$	10	-6	32	0°	0°
$q_{i_3}$	0	2	2	180°	0°
$q_{f_3}$	10	-6	42	0°	0°

Tabulka 5.6: Konfigurace, které byly použity k testování.

použita heuristika Manhattan a cena cesty  $g(n)$  se rovná Euklidově vzdálenosti. Podmínky použité v experimentu byly  $\gamma_{max} = 45^\circ$  a minimální poloměr otáčení měl velikost tři. Každé měření potřebného času k nalezení cesty bylo provedeno desetkrát. Do tabulky 5.2 byla zanesena průměrná hodnota  $t$  z naměřených časů.

#### Vyhodnocení experimentů

Tabulka 5.7 dokazuje, že rychlejší časy má spojitá metoda. Nicméně pro velmi malé vzdálenosti nenajde spojitá metoda vždy neoptimálnější řešení. Naopak diskrétní metoda je na malých vzdálenostech rychlá a vždy najde optimální řešení. U třetího experimentu, kdy jsou od sebe body relativně výškově daleko, je mnohem rychlejší spojitá metoda a v tomto případě najde téměř optimální řešení. To samé platí i u druhého experimentu.

	<i>Diskrétní metoda</i>	<i>Spojité metoda</i>
	$t[s]$	$t[s]$
1.	0.1432	0.0073
2.	1.4989	0.0075
3.	44.5346	0.0076

Tabulka 5.7: Naměřené hodnoty z provedených experimentů.

# Kapitola 6

## Závěr

Cílem této práce bylo najít a popsat přístupy plánování letadla v neomezeném, statickém prostoru bez překážek a dále implementovat a provést srovnávací experimenty u vybraných metod.

Z výsledků experimentů vyplývá, že metoda založená na Dubinových křivkách dosahuje nejlepšího výpočetního času potřebného k nalezení cílové cesty. Problém této metody ovšem spočívá v tom, že si nedokáže poradit s velmi malými vzdálenostmi. Pro tyto vzdálenosti nalezne neoptimální cestu. Dále je třeba doladit i výpočet trajektorie pro střední výšku, kde nemusí být vybrána neoptimálnější trajektorie. Možné řešení by mohlo spočívat v provedení nejmenší možné spirály a poté dopočítat přes Dubinovi křivky cestu k cíli.

Co se týče optimálnosti cesty, tak je na tom lépe metoda založená na diskretizaci prostoru. Ta si díky implementaci A\* dokáže poradit i s malými vzdálenostmi. Problém této metody, i přes provedené optimalizace, je ve výpočetním času, který je potřebný k nalezení cesty. Může za to zvolená heuristika, která si neumí poradit s cílovým a počátečním nakloněním letadla. To by se dalo odstranit předvýpočtem cílové a počáteční konfigurace, tak aby se eliminovalo počáteční a cílové naklonění.

Z hlediska pokračování vývoje je významější metoda založená na diskretizaci prostředí, protože lze ji upravit o vyhýbání se překážek.

# Literatura

- [1] Ambrosino, G.; aj.: *Algorithms for 3D UAV Path Generation and Tracking*. 45th IEEE Conference on Decision and Control, 2006.
- [2] Chitsaz, H.; LaValle, S. M.: *Time-optimal paths for a Dubins airplane*. 46th IEEE Conference on Decision and Control, 2007.
- [3] Gross, T.: *Plánování cesty mobilního robota*. Diplomová práce, FSI VUT v Brně, Brno, 2007.
- [4] Hwangbo, M.; Kuffner, J.; Kanade, T.: *Efficient Two-phase 3D Motion Planning for Small Fixed-wing UAVs*. IEEE International Conference on Robotics and Automation, 2007.
- [5] Krček, P.: *Plánování cesty autonomního lokomočního robota na základě strojového učení*. Diplomová práce, FSI VUT v Brně, Brno, 2010.
- [6] LaValle, S. M.: *Planning Algorithms*. Cambridge University Press, 2006.
- [7] Shkel, A. M.; Lumelsky, V.: *Classification of the Dubins set*. Robotics and Autonomous Systems 34, 2001.
- [8] Wan, T.; Tang, W.; Chen, H.: *A real-time 3D motion planning and simulation scheme for nonholonomic systems*. Simulation Modelling Practice and Theory 19, 2011.
- [9] Wikipedie: Degrees of freedom (mechanics) — Wikipedia, The Free Encyclopedia. [http://en.wikipedia.org/wiki/Degrees\\_of\\_freedom\\_%28mechanics%29](http://en.wikipedia.org/wiki/Degrees_of_freedom_%28mechanics%29), [Online; navštíveno 30. 04. 2013].
- [10] Wikipedie: Klotoida — Wikipedie: Otevřená encyklopedie. <http://cs.wikipedia.org/wiki/Klotoida>, [Online; navštíveno 30. 04. 2013].
- [11] Wikipedie: Motion planning — Wikipedia, The Free Encyclopedia. [http://en.wikipedia.org/wiki/Motion\\_planning](http://en.wikipedia.org/wiki/Motion_planning), [Online; navštíveno 30. 04. 2013].
- [12] Zbořil, F. V.; Zbořil, F.: *Základy umělé inteligence IZU*. FIT VUT v Brně, 2012.