ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA ELEKTROTECHNICKÁ

BAKALÁŘSKÁ PRÁCE

Václav Krtička

Plánování v multi-robotických systémech

Katedra kybernetiky
Vedoucí bakalářské práce: **RNDr. Miroslav Kulich, Ph.D.**

Praha, 2010

Prohlášení

Prohlašuji, že jsem svou diplomovou práci vypracoval samostatně a použil jsem pouze podklady (literaturu, SW, projekty atd.) uvedené v přiloženém seznamu.

V Praze dne 31. 5. 2010 .... Krtička V. ....

podpis

České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra kybernetiky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: Václav Krtička

Studijní program: Elektrotechnika a informatika (bakalářský), strukturovaný

Obor: Kybernetika a měření

Název tématu: Plánování v multi-robotických systémech

Pokyny pro vypracování:

Cílem práce je kompletní dokumentovaná implementace algoritmu "uvolněného" plánování (decoupled planning), který plánuje bezkolizní cestu pro několik mobilních robotů. Tento algoritmus plánuje cestu pro každý robot zvlášť a poté se snaží odstranit kolize jednotlivých cest. Výsledný kód bude použit na reálných robotech katedry kybernetiky.

1. Seznamte se s algoritmy plánování pro více mobilních robotů (viz.[1]).
2. Seznamte se se systémem Player/Stage (viz.[2]).
3. Implementujte algoritmy pro plánování na fixních trajektoriích a sítích cest.
4. Algoritmy experimentálně ověřte na systému Player/Stage a závěry diskutujte.

Seznam odborné literatury:

[1] Lavalle, S.M.; Hutchinson, S.A.: Optimal Motion for Multiple Robots Having Independent Goals. IEEE Trans. on Robotics and Automation, 14(16):912-925, December 1998.
[2] http://playerstage.sourceforge.net)

Vedoucí bakalářské práce: RNDr. Miroslav Kulich, Ph.D.

Platnost zadání: do konce zimního semestru 2010/2011

L.S.

prof. Ing. Vladimír Mařík, DrSc.
vedoucí katedry

doc. Ing. Boris Šimák, CSc.
děkan

V Praze dne 9. 12. 2009

Abstrakt

Bakalářská práce se zabývá problematikou koordinace více mobilních robotů pohybujících se ve společném prostoru a jejím plánováním. Úlohou koordinace je, aby se roboty přemístily v prostoru z počáteční pozice do pozice cílové, aniž by došlo k jakékoliv vzájemné kolizi.

První část je věnována teoretickému rozboru koordinace robotů s pevně stanovenými cestami, kde se koordinace odvíjí pouze od možnosti pohybu a zastavování robotů na jejich fixních předem určených cestách.

V druhé části se lze seznámit s plánováním koordinace robotů na jim určených mapách cest, kde roboty nejsou omezeny pouze na pohyb a zastavování, ale i na možnost vybrat cestu, kterou se dostane z počátku do cíle.

Poslední část prezentuje experimenty popsaných algoritmů koordinace, které byly implementovány, přičemž je ukázána jejich funkčnost, vliv nastavení parametrů a možné problémy, které mohou nastat při uvedení do praxe.

Abstract

The bachelor thesis deals with coordination of multiple mobile robots moving in a common area. The task of coordination is to move the robots in space from an initial position to target position, without mutual collisions.

The first part deals with theoretical analysis, coordination of robots with fixed paths. Coordination depends only on the movement and possibility of stopping the robots at the fixed predetermined paths.

The second part aims planning robots to coordinate their assigned roadmap paths, where robots are not limited to movement and stopping, but also possibility to choose the path that gets from the beginning to the end.

The last section presents the experiments carried out wit coordination of implemented algorithm, which is shown on its functionality, parameters and possible problems that may arise in putting into practice.

Poděkování

Děkuji panu doktoru Miroslavu Kulichovi za trpělivost a ochotu, kterou prokázal při vedení této bakalářské práce.

Obsah

Seznam tabulek

Seznam obrázků

Úvod

V robotice je celá škála problémů, které lze řešit, ať už se týkají fyzických konstrukcí robotů nebo jejich programového vybavení. Jednou z částí robotiky je i řízení více mobilních robotů v prostoru, resp. jejich koordinace mezi sebou tak, aby nedocházelo k jejich kolizím. K této úloze lze přistupovat několika způsoby. Buď jsou roboty samostatnými jednotkami s nezávislou umělou inteligencí a vybaveny senzory, které používají pro orientaci v prostoru a identifikování překážek v aktuálním čase, kde překážkou může být i jiný robot, nebo je řízení centralizováno ve výpočetní jednotce společné všem robotům.

Tato bakalářská práce se zabývá pouze centralizovaným přístupem bezkolizní koordinace mezi roboty. Pro algoritmus, který tuto koordinaci počítá je nezbytná znalost parametrů robotu, jako například jeho rozměry a rychlost, ale také cesta, či cesty, které jsou pro robot již stanoveny. Výstupem je posloupnost pozic robota v jednotlivých časech. Výhodou je, narozdíl od koordinace nezávisle chovajících se robotů, že zvyšuje časovou efektivitu koordinace, protože již předem uvažuje možnost kolizí mezi nimi a tím se může včas rozhodnout jak s těmito informacemi naloží.

V první části se bakalářská práce zabývá koordinací robotů v prostoru, kde každý z nich má pevně stanovenou cestu, po které se pohybuje. Aby se tedy roboty vyhnuly případným kolizím, musejí v některých částech svých cest zastavit a počkat, až jim ostatní roboty uvolní cestu. Problém je detailněji popsán v kapitole 1, kde jsou také uvedeny definice potřebné k popsání algoritmu, kterým se koordinace řeší.

Druhá část bakalářské práce obsahuje popis koordinace robotů v prostoru, které se pohybují po pevně zadaných mapách cest. V těchto mapách si roboty mohou na každém rozcestí vybrat, kterou cestou se budou pohybovat dále až k cíli. Aby se vyhnuly kolizím, mají tedy možnost zastavit a počkat na volnou cestu, nebo zvolit v mapě cestu jinou. Více o této úloze je popsáno v kapitole 2, včetně základních definic a rozebrání příslušného algoritmu řešení.

Ve poslední části bakalářské práce jsou prezentovány některé experimenty, kterými se prokazuje funkčnost implementovaného algoritmu, jeho parametry a nedostatky vycházející z diskretizace problematiky koordinace. Jeden z experimentů je i zobrazen pomocí programu Player/Stage, který simuluje pohyby robotů v prostoru s ohledem na jejich reálné fyzikální vlastnosti.

Cílem práce je implementovat algoritmus, týkající se problematiky popsané v první i druhé části bakalářské práce, v programovacím jazyku Java a odzkoušet jeho funkčnost a parametry různými experimenty uvedenými v kapitolách 3 a 4.

Tato bakalářská práce především vychází z definic a popisů v článku [3] a v knize [4].

Kapitola 1

Koordinace robotů s pevně stanovenými cestami

1.1 Úvod do problematiky

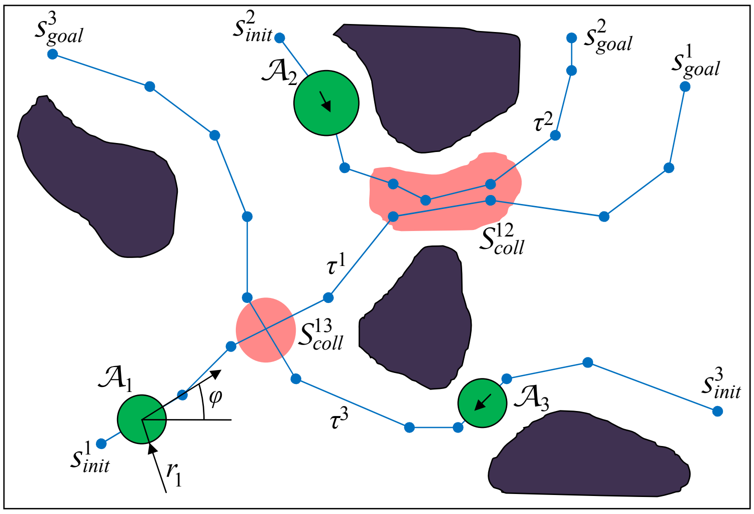
Tato část se zabývá problematikou koordinace více mobilních robotů v prostoru pohybujících se po pevně stanovených cestách. V následujících odstavcích uvedeme hlavní myšlenku koordinačního algoritmu a definice základních pojmů pak budou definovány v kapitole 1.2.

Uvažujme dvourozměrný prostor, ve kterém se pohybuje robot reprezentován kruhem[1] o určitém poloměru. Robot buď stojí na místě, nebo se pohybuje konstantní rychlostí vpřed s možností změny směru pohybu. Robot se pohybuje po fixní, předem stanovené, cestě, která je reprezentována posloupností uzlů, ve kterých robot mění svůj směr. Uzly jsou spojeny hranami obecně různých délek. Cesta samotná se rozdělí na ekvidistantní úseky v závislosti na periodě vzorkování a rychlosti robotu. Čím menší periodu vzorkování zvolíme, tím budou úseky kratší a tím docílíme i vyššího rozlišení.[2] Aktuální stav robotu lze popsat konfigurací, která je dána jeho polohou, resp. souřadnicemi jeho středu. V obecném případě je součástí konfigurace robotu ještě jeho natočení, nicméně pro koordinaci kruhových robotů ho není nutné uvažovat. Aby se přemísťoval od jedné konfigurace k další, musí k tomu dostat příkaz, který je buď "jeď" nebo "stůj". V případě "stůj" robot svou konfiguraci nemění (tj. stojí na místě). Když obdrží příkaz "jeď", přesune se podle definované funkce na další konfiguraci. Každá tato akce se hodnotí určitou cenou. V praxi se může jednat například o spotřebu paliva při přesunu či rozjezdu z klidového stavu robotu, nebo nedodržení časového plánu při zastavení a stání robotu. Jakmile se již robot nachází v cíli, žádné další příkazy již nepřijímá a neplatí žádné ceny.

Nyní uvažujme více robotů se svými cestami, pohybujícími se v tomtéž prostoru. Jejich cesty se mohou navzájem křížit. V tomto případě by mohlo docházet ke kolizím, pokud by se minimálně dva roboty střetly. K tomu nemusí dojít pouze na křižovatce, ale i v případě příliš

[1] Robot libovolného tvaru aproximujeme kruhem.

[2] Platí i pro rychlost. Čím je vyšší, tím delší úseky budou.



Obrázek 1.1: Dvourozměrný prostor, kde se pohybují roboty $\mathcal{A}_i$, s poloměrem r_i a směrem φ, po cestách τ^i s počáteční a koncovou konfigurací s^i_{init} a s^i_{goal}. Množiny kolizních stavů jsou $\mathcal{S}^{ij}_{coll}$.

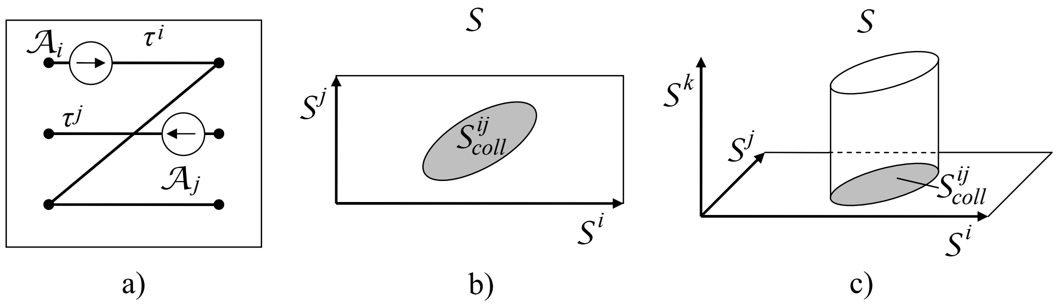
malých vzdáleností částí cest od sebe, viz obr. 1.1. K detekci těchto kolizních stavů slouží právě poloměr kruhu reprezentujícího každý robot. Roboty kolidují, pokud je vzdálenost mezi nimi menší než suma jejich poloměrů.

K nalezení všech kolizních situací je třeba vytvořit prostor kombinací indexů konfigurací všech robotů, který budeme nazývat *koordinační prostor* (někdy též stavový prostor). Tento prostor je vícerozměrný a má tolik rozměrů, kolika se týká robotů. Každý rozměr má velikost odpovídající počtu konfigurací na cestě příslušného robotu. Pro ilustraci si představme pouze dvourozměrný *koordinační prostor* dvou robotů jako obdélník, viz obr. 1.2b. Každý bod uvnitř tohoto obdélníku odkazuje na určitou kombinaci konfigurací robotů na jejich cestách (stav), viz obr. 1.2a. Oblast označená $\mathcal{S}^{ij}_{coll}$ je množina kolizních stavů dvou robotů. Válec tyčící se nad obdélníkem, viz obr. 1.2c, představuje taktéž kolizní stavy robotů, které se mohou promítat i do dalších rozměrů, zde např. rozměr $\mathcal{S}^k$, tzn. v *koordinačním prostoru* více jak dvou robotů, viz obr. 1.3.

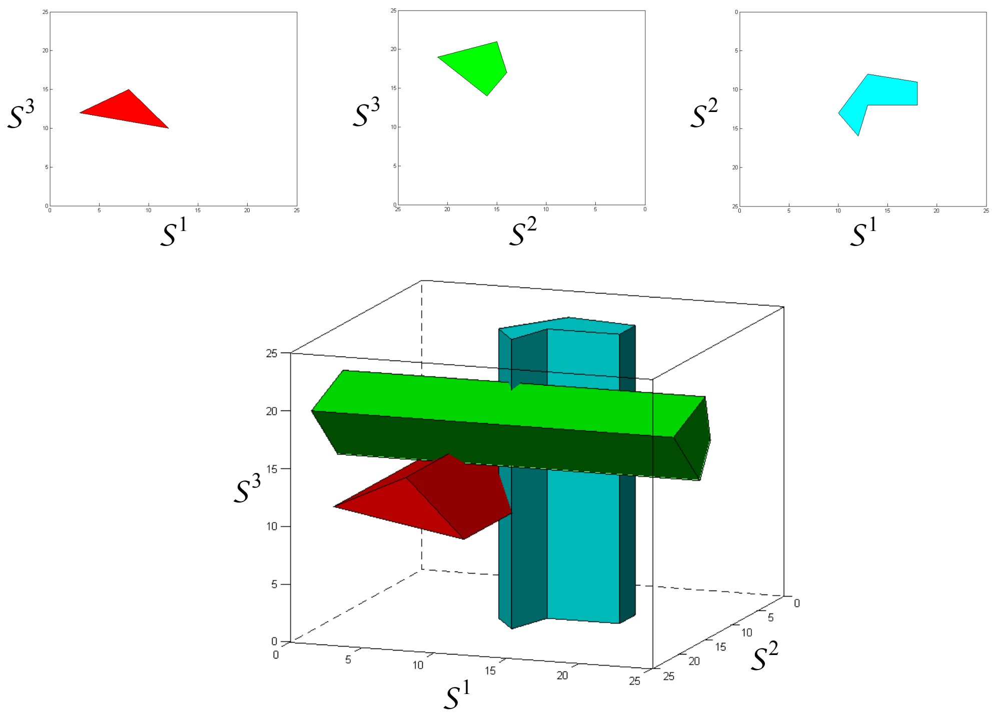
Aby případy kolizí na cestách robotů nenastaly, musí se roboty zkoordinovat, tj. musí se zajistit, aby do sebe nenarazily. To znamená, že jeden robot musí zastavit, aby ten druhý mohl bez úhony jet dál. Jakmile prvnímu již nic nepřekáží, pokračuje ve své cestě. K nalezení optimální koordinace mezi roboty ve výsledku hledáme bezkolizní trajektorii v *koordinačním prostoru*, kterou nazveme *stavová trajektorie*. Ta vede ze stavu, kde konfigurace robotů odpovídají počátečním uzlům jejich cest, do stavu konfigurací robotů v cílových uzlech jejich cest. Pro hledanou *stavovou trajektorii* platí několik základních pravidel:

1. Každý robot se musí dostat do cíle do stanovené doby.

2. *Stavová trajektorie* nesmí vést přes kolizní oblasti v *koordinačním prostoru*.



Obrázek 1.2: Postup tvorby koordinačního prostoru. $\mathcal{A}_i$, $\mathcal{A}_j$ jsou dva roboty, τ^i, τ^j jsou jejich cesty. $\mathcal{S}$ je *koordinační prostor*, $\mathcal{S}^i$, $\mathcal{S}^j$ jsou konfigurační trajektorie obou robotů odpovídajících délek délkám jejich cest a $\mathcal{S}^{ij}_{coll}$ jsou kolizní stavy mezi roboty.



Obrázek 1.3: Spojitý *koordinační prostor* 3 robotů s množinou kolizních stavů. Tři (menší) 2D obrázky nahoře reprezentují kolizní množiny stavů (polygony) mezi dvěma roboty. 3D obrázek dole (větší) reprezentuje sjednocení všech kolizí (objekty) mezi všemi 3 roboty, viz [4].

3. Ceny (např. za akce robotů, či za porušení prvních dvou pravidel), kterými jsou roboty penalizovány by měly být co nejmenší.

4. *Stavová trajektorie* musí být nejkratší možná s ohledem na předchozí pravidla.

Nesplnění prvních dvou pravidel je penalizováno vysokou cenou (resp. nekonečnou). Pravidla 3 a 4 se snaží ovlivnit vytváření *stavové trajektorie* tak, aby koordinace byla optimální (tj. jsou penalizovány nevýhodné akce).

Cena trajektorie pro jeden robot je vážená suma cen vyplývající z jednotlivých uvedených pravidel.

1.2 Základní definice

Uvažujme 2D prostor $\mathbb{R}^2$, ve kterém se pohybuje N kruhových robotů $\mathcal{A}_i$ se středem (x_i, y_i) a poloměrem r_i, kde index $i \in \langle 1; N \rangle$.[3] Plocha robotu je tedy definována takto

$$\mathcal{A}_i = \left\{ (x, y) | (x - x_i)^2 + (y - y_i)^2 \leq r_i^2 \right\}.$$

Robot $\mathcal{A}_i$ se dokáže pohybovat rychlostí v_i, nebo stát na místě.

Trajektorie pohybu robotu $\mathcal{A}_i$ je dána jeho vlastní, předem stanovenou, fixní cestou τ^i. Ta je tvořena posloupností souřadnic uzlů z počátečního uzlu do cílového uzlu.

1.2.1 Spojitý koordinační prostor

Robot $\mathcal{A}_i$ se v čase t nachází v konfiguraci $s^i(t)$, kde $s^i(t) = (x_i(t), y_i(t))$. Souřadnice $x_i(t)$ a $y_i(t)$ jsou souřadnice středu robotu $\mathcal{A}_i$ v $\mathbb{R}^2$ nacházejícího se na cestě τ^i v čase t. Indexy konfigurací $s^i(t)$ jsou v pořadí zahrnuty do množiny *konfigurační trajektorie* $\mathcal{S}^i$, což definuje zobrazení $s^i : \langle 0; T_i \rangle \to \mathcal{S}^i$, kde T_i je *destinační čas*, ve kterém se robot $\mathcal{A}_i$ nachází v cíli cesty τ^i. Počáteční konfigurace je $s^i_{init} = s^i(0)$ a cílová konfigurace je $s^i_{goal} = s^i(T_i)$. Vzdálenost mezi dvěmi nejbližšími konfiguracemi je

$$\left| s^i(t + \Delta t) - s^i(t) \right| = v_i \Delta t,$$

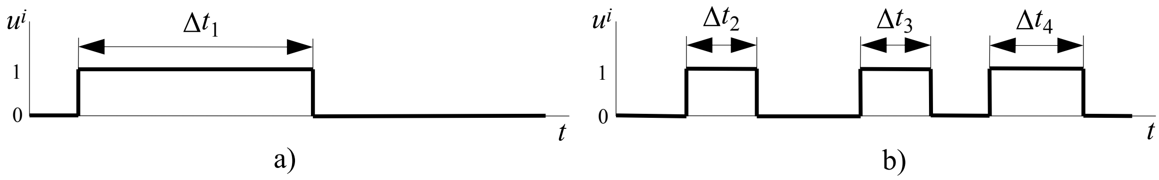
kde Δt je časový rozdíl.

Pohyb robotu (též změna konfigurace) je realizován akcí $u^i(t)$ v čase t. Akce $u^i(t) \in \{0, 1\}$. Když $u^i(t) = 0$, robot $\mathcal{A}_i$ v čase t stojí. Jestliže $u^i(t) = 1$, robot $\mathcal{A}_i$ v čase t jede rychlostí v_i. Formálně platí

$$(s^i(t))' = f^i(s^i(t), u^i(t)), \tag{1.1}$$

kde následující konfigurace stavu $(s^i(t))'$ je dána funkcí f^i, která ze zadané konfigurace $s^i(t)$ a akce $u^i(t)$ vypočte novou konfiguraci $(s^i(t))'$. Ta je při $u^i(t) = 0$ shodná s konfigurací $s^i(t)$, nebo při $u^i(t) = 1$ je následující konfigurace $s^i(t + \Delta t)$.

[3] Algoritmus koordinace robotů nezávisí na počtu rozměrů, ve kterém se robot pohybuje. Tento počet hraje důležitou roli při navrhování samotných cest.



Obrázek 1.4: Akce $u^i(t)$ v čase t, kde $\Delta t_1 = \Delta t_2 + \Delta t_3 + \Delta t_4$.

Akce $u^i(t)$ však nepočítá s fyzickými aspekty skutečného robotu. Uvažuje se totiž, že robot se z klidu uvede do pohybu v nekonečně krátkém čase a naopak se z pohybu taktéž v okamžiku může zastavit.

Pokud chceme uvažovat obecně dobu rozjezdu nebo zastavení, můžeme je ohodnotit úměrně velkou cenou, kterou robot $\mathcal{A}_i$ při každé akci $u^i(t)$ bude penalizován. Nechť cenovou funkcí je $g^i(t, s^i(t), u^i(t))$, která určuje cenu každé akce $u^i(t)$ pro konfiguraci $s^i(t)$ v čase t. Například při přechodu robotu z klidu do pohybu penalizace pro robot bude vyšší, než při plynulé jízdě. Na obr. 1.4 jsou znázorněny dvě různé možnosti akcí $u^i(t)$ v čase t vedoucí ke stejnému stavu, kde $\Delta t_1 = \Delta t_2 + \Delta t_3 + \Delta t_4$, a přitom by součet všech výsledků $g^i(t, s^i(t), u^i(t))$ pro obr. 1.4a byl menší než pro obr. 1.4b.

Dále také platí
$$g^i(t, s^i(t), u^i(t)) = 0 \quad \text{když} \quad s^i(t) = s^i_{goal}. \tag{1.2}$$
Tzn. když je robot $\mathcal{A}_i$ v cílové konfiguraci s^i_{goal}, pak není penalizován žádnou cenou.

Nedorazí-li robot $\mathcal{A}_i$ do cíle s^i_{goal} do *destinačního času* T_i, bude penalizován nekonečně velkou cenou. Formálně
$$q^i(s^i(T_i)) = \begin{cases} 0 & \text{když} \quad s^i(T_i) = s^i_{goal}, \\ \infty & \text{když} \quad s^i(T_i) \neq s^i_{goal}. \end{cases} \tag{1.3}$$

Uvažujme nyní více robotů $\mathcal{A}_i$ v tomtéž prostoru $\mathbb{R}^2$. *Koordinační prostor* $\mathcal{S}$, potažmo stavový prostor, je dán kartézským součinem
$$\mathcal{S} = \mathcal{S}^1 \times \mathcal{S}^2 \times \cdots \times \mathcal{S}^N, \tag{1.4}$$
kde každá *konfigurační trajektorie* $\mathcal{S}^i$ je jedním rozměrem tohoto *koordinačního prostoru* $\mathcal{S}$. Jedná se tedy o N-rozměrný prostor. Stavy *koordinačního prostoru* jsou $s = (s^1, s^2, ..., s^N)$, kde $s \in \mathcal{S}$. Vektor akcí je $u = (u^1, u^2, ..., u^N)$. Vztah 1.1, lze rozšířit pro všechny roboty $\mathcal{A}_i$ v *koordinačním prostoru*
$$(s(t))' = f(s(t), u(t)), \tag{1.5}$$
kde funkce f vypočte ze zadaného stavu $s(t)$ a vektoru akcí $u(t)$ nový stav $(s(t))'$. Ten je shodný s $s(t)$ za předpokladu, že $u(t) = (0, 0, ..., 0)$[4]. Pokud některá $u^i \neq 0$, pak platí $(s(t))' \neq s(t)$ a $(s(t))'$ je některým ze sousedních stavů $s(t)$.

[4]Akční vektor $u(t)$, kde pro $\forall i$ platí $u^i(t) = 0$ se v daném optimálním řešení nevyskytuje. To by totiž znamenalo, že všechny roboty $\mathcal{A}_i$ v čase t stojí.

Definujme podmnožinu kolizních stavů $\mathcal{S}^{ij}_{coll}$ dvou robotů $\mathcal{A}_i$ a $\mathcal{A}_j$ v *koordinačním prostoru* $\mathcal{S}$, tedy $\mathcal{S}^{ij}_{coll} \subset \mathcal{S}$, kde $s^{ij}_{coll} \in \mathcal{S}^{ij}_{coll}$ jsou kolizní stavy mezi dvěma roboty. Nechť $\mathcal{A}^{\circ}_i$ je otevřená množina množiny reprezentující robot $\mathcal{A}_i$ po vyloučení její hranice, pak platí

$$\mathcal{S}^{ij}_{coll} = \left\{ s \ \in \mathcal{S} | \mathcal{A}^{\circ}_i(s^i) \cap \mathcal{A}^{\circ}_j(s^j) \neq 0 \right\}.$$

Indexy $i \neq j$ značí dva různé roboty. Sjednocením

$$\mathcal{S}_{coll} = \bigcup_{i \neq j} \mathcal{S}^{ij}_{coll}$$

získáme právě celou *kolizní podmnožinu* kolizních stavů $\mathcal{S}_{coll} \subset \mathcal{S}$. Rozdíl množin

$$\mathcal{S}_{valid} = \mathcal{S} \backslash \mathcal{S}_{coll}, \tag{1.6}$$

kde $\mathcal{S}_{valid} \subset \mathcal{S}$ je *nekolizní podmnožina* nekolizních stavů. Označme tedy nekolizní stavy $s_{valid} \in \mathcal{S}_{valid}$ a kolizní stavy $s_{coll} \in \mathcal{S}_{coll}$ *koordinačního prostoru* $\mathcal{S}$. Počáteční stav je $s_{init} = (s^1_{init}, s^2_{init}, ..., s^N_{init})$ a cílový stav je $s_{goal} = (s^1_{goal}, s^2_{goal}, ..., s^N_{goal})$.

Definujeme cenu stavu

$$c(s(t)) = \begin{cases} 0 & \text{když} \quad s(t) \in \mathcal{S}_{valid}, \\ \infty & \text{když} \quad s(t) \in \mathcal{S}_{coll}. \end{cases} \tag{1.7}$$

Funkce c vrací nekonečně velkou hodnotu, pokud bude platit, že aktuální stav $s(t)$ v *koordinačním prostoru* $\mathcal{S}$ bude kolizní.

Trajektorie každého robotu $\mathcal{A}_i$ se ohodnotí *funkcí souhrnné ceny*

$$L^i(s_{init}, s_{goal}, u) = \int_0^{T^i} \left[g^i(t, s^i(t), u^i(t)) + c(s(t)) \right] dt + q^i(s^i(T^i)), \tag{1.8}$$

ve které jsou zahrnuty vztahy 1.2, 1.3 a 1.7.

1.2.2 Diskrétní koordinační prostor

Jelikož daný algoritmus je realizován na počítači, je vhodné, aby veškerá příslušná problematika byla uvažována diskrétně.

Pro rozdělení cesty τ^i robotu $\mathcal{A}_i$ na stejně dlouhé úseky definujeme časový interval Δt, neboli *vzorkovací čas*, který je společný všem robotům $\mathcal{A}_i$. Počet úseků cesty τ^i je pak

$$n^i = \left\lceil \frac{length(\tau^i)}{v^i \cdot \Delta t} \right\rceil, \tag{1.9}$$

kde $\lceil \cdot \rceil$ značí zaokrouhlení nahoru.

Jednotlivé konfigurace s^i_k robotu $\mathcal{A}_i$ mezi úseky cesty τ^i mají podobné vlastnosti jako $s^i(t)$. Jsou definovány jako $s^i_k = (x_k, y_k)$. Index k odkazuje na čas $(k-1)\Delta t$, kde $k \in$

$\{1, ..., K\}$. Index $k \in \tilde{\mathcal{S}}^i$ označuje konfiguraci s_k^i, kde $\tilde{\mathcal{S}}^i$ je množina *diskrétní konfigurační trajektorie* robotu $\mathcal{A}_i$. Počet prvků $\tilde{\mathcal{S}}^i$ je

$$K^i = n^i + 1.$$

Počáteční konfigurace je $s_{init}^i = s_1^i$ a cílová konfigurace je $s_{init}^i = s_{K^i}^i$. Vzdálenost mezi dvěma nejbližšími konfiguracemi je

$$\left| s_{k+1}^i - s_k^i \right| = v^i \Delta t.$$

Akce mezi dvěma konfiguracemi s_k^i je u_k^i, kde opět $u_k^i \in \{0, 1\}$. Následující konfigurace je dána vztahem

$$(s_k^i)' = f^i(s_k^i, u_k^i), \tag{1.10}$$

kde funkce f^i ze zadané konfigurace s_k^i a akce u_k^i vypočítá novou konfiguraci $(s_k^i)'$, která bude

$$f^i(s_k^i, u_k^i) = \begin{cases} s_k^i & \text{když} \quad u_k^i = 0, \\ s_{k+1}^i & \text{když} \quad u_k^i = 1. \end{cases}$$

Cenová funkce se uplatní stejná jako ve vztahu 1.2. Ta se využije v dílčí cenové funkci $l_k^i(x_k^i, u_k^i)$, pro kterou platí

$$l_k^i(s_k^i, u_k^i) = \int_{(k-1)\Delta t}^{k\Delta t} g^i(t, s^i(t), u^i(t))dt. \tag{1.11}$$

Penalizační funkce za nedodržení *destinačního času* T^i se chová stejně jako ve vztahu 1.3.

Diskrétní koordinační prostor $\tilde{\mathcal{S}}$ je taktéž kartézským součinem jako ve vztahu 1.4, s tím rozdílem, že $\mathcal{S}^i$ odpovídá $\tilde{\mathcal{S}}^i$. Příslušné stavy *diskrétního koordinačního prostoru* $\tilde{\mathcal{S}}$ jsou $\tilde{s} \in \tilde{\mathcal{S}}$, kde $\tilde{s} = (s^1, s^2, ..., s^N)$.

Diskretizací vztahu 1.5 a dosazením vektoru akcí $u_k = (u_k^1, u_k^2, ..., u_k^N)$ dostaneme

$$\tilde{s}_k' = f(\tilde{s}_k, u_k). \tag{1.12}$$

Množiny $\tilde{\mathcal{S}}_{valid}$ a $\tilde{\mathcal{S}}_{coll}$ se získají z $\mathcal{S}_{valid}$ a $\mathcal{S}_{coll}$, stejně jako jejich prvky, podobně jako $\tilde{\mathcal{S}}$ z $\mathcal{S}$. Záměnou $s(t)$ za $\tilde{s}$ ve vztahu 1.7, dostaneme funkci penalizace za diskrétní kolizní stavy $\tilde{s}_{coll}$.

Aproximujeme 1.8 pro diskrétní čas jako

$$L^i(s_{init}, s_{goal}, u) = \sum_{k=1}^{K} \left\{ l_k^i(s_k^i, u_k^i) + c(\tilde{s}_k) \right\} + q^i(s^i(T^i)),$$

kde se objevují vztahy 1.11, 1.7 a 1.3.

1.2.3 Cesta diskrétním koordinačním prostorem

Úlohou všech robotů $\mathcal{A}_i$ je dostat se z počátečního stavu s_{init} do cílového stavu s_{goal} pomocí cesty *diskrétním koordinačním prostorem* $\tilde{\mathcal{S}}$.

Definujeme pro robot $\mathcal{A}_i$ *strategii robotu* γ^i. Ta reprezentuje možnou volbu funkce akce u^i, která je posloupností jednotlivých akcí u_k^i v krocích k, jako $u_k^i = \gamma^i(\tilde{s}, k)$. *Strategie* v *diskrétním koordinačním prostoru* $\tilde{\mathcal{S}}$ je $\gamma = \left\{\gamma^1, \gamma^2, ..., \gamma^N\right\}$. Nechť $\tilde{\Gamma}$ je množina všech dostupných *strategií*, pak platí $\gamma \in \tilde{\Gamma}$.

Nechť α_γ je *stavová trajektorie*, která vede *diskrétním koordinačním prostorem* $\tilde{\mathcal{S}}$. Je vyjádřena posloupností stavů $\tilde{s}$ a určena pomocí *strategie* γ. Lze ji také popsat zobrazením $\alpha_\gamma : \langle s_{init}; s_{goal}\rangle \to \tilde{\mathcal{S}}$.

Předpokládáme-li, že stavy s_{init} a s_{goal} jsou dány, pak můžeme psát $L^i(\gamma)$ namísto $L^i(s_{init}, s_{goal}, u)$.

Definujeme ekvivalenci $\sim_{L^i}$ mezi různými dvěma strategiemi γ^i a $\gamma^{i'}$ robotu $\mathcal{A}_i$. Píšeme $\gamma^i \sim_{L^i} \gamma^{i'}$, pokud platí $L^i(\gamma^i) = L^i(\gamma^{i'})$ pro jeden robot $\mathcal{A}_i$.

Ekvivalenci mezi strategiemi γ a γ' všech robotů $\mathcal{A}_i$ budeme označovat symbolem $\sim_L$ a nazveme ho *vztah ekvivalence* všech párů strategií $\gamma \in \tilde{\Gamma}$. Když bude platit $L^i(\gamma) = L^i(\gamma')$ pro každý robot $\mathcal{A}_i$, pak platí $\gamma \sim_L \gamma'$, což znamená, že γ je ekvivalentní s γ'. Obecně bude existovat mnoho strategií $\gamma \in \tilde{\Gamma}$ s ekvivalentními cenami.

Definujeme *kvocient strategie* $[\gamma]_L$, který zahrnuje, podobně jako množina, všechny strategie γ, které jsou ekvivalentní podle *vztahu ekvivalence* $\sim_L$.

Nechť $L^i(\gamma) \leq L^i(\gamma')$ pro každý robot $\mathcal{A}_i$, kde $[\gamma]_L, [\gamma']_L \in \Gamma/\!\sim$, pak platí $[\gamma]_L \preceq [\gamma']_L$, kde $\preceq$ je symbol, který nazveme *parciálním řazením*. Když navíc pro nějaký robot $\mathcal{A}_j$ platí $L^j(\gamma) < L^j(\gamma')$, můžeme říci, že $[\gamma]_L$ je *lepší* než $[\gamma\ ]_L$. Pokud však pro nějaké dva roboty $\mathcal{A}_i$ a $\mathcal{A}_j$, kde $i \neq j$, platí $L^i(\gamma) < L^i(\gamma')$ a zároveň $L^j(\gamma) > L^j(\gamma')$, pak $[\gamma]_L$ a $[\gamma']_L$ považujeme za *neporovnatelné*.

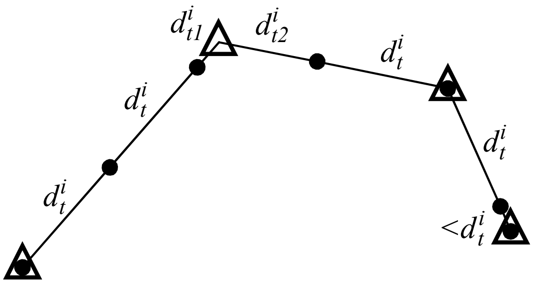
Symbol γ^* značí *strategii*, pro kterou platí, že *kvocient strategie* $[\gamma^*]_L$ je *minimální*, když pro všechny $[\gamma]_L \neq [\gamma^*]_L$ a zároveň $[\gamma]_L$ a $[\gamma^*]_L$ nejsou *neporovnatelné*, pak $[\gamma^*]_L \preceq [\gamma]_L$.

1.3 Algoritmus koordinace

Uvažujme *diskrétní koordinační prostor* popsaný v kapitole 1.2.2. Vstupem algoritmu jsou seznamy souřadnic uzlů cest τ^i pro každý definovaný robot $\mathcal{A}_i$. Výstupem jsou všechny *strategie* γ^* a jejich *stavové trajektorie* α_{γ^*}, tak jak byly definovány v kapitole 1.2.3.

1.3.1 Diskrétní konfigurační trajektorie

Uvažujme jeden robot $\mathcal{A}_i$ s rychlostí v_i a poloměrem r_i. Poloměr r_i může být i větší, než je skutečný rozměr robotu $\mathcal{A}_i$, což sníží riziko případných kolizí robotů, které mohou nastat v důsledku diskrétního uvažování koordinace. Cesta τ^i je načtena jako posloupnost souřadnic uzlů (`x1 y1 x2 y2 ... xn yn`). Zadán je i *vzorkovací čas* Δt.



Obrázek 1.5: Cesta τ^i vzorkovaná dvěma způsoby a) a b).

Účelem je postupně nalézt všechny konfigurace na cestě τ^i, které jsou od sebe vzdáleny maximálně o vzdálenost danou součinem $v_i \Delta t$, a zařadit jejich indexy do *diskrétní konfigurační trajektorie* $\tilde{\mathcal{S}}^i$. Postupuje se od prvního uzlu cesty τ^i do posledního.

Vzorkování cesty

Definujme *vzorkovací vzdálenost* robotu $\mathcal{A}_i$

$$d^i_t = v_i \Delta t. \tag{1.13}$$

Cestu τ^i lze vzorkovat několika metodami. Dvě z nich jsou ukázány na obr. 1.5. Malé trojúhelníky značí uzly cesty τ^i, černé body jsou konfigurace s^i_k.

Metoda 1: Algoritmus navzorkuje cestu τ^i *vzorkovací vzdáleností* d^i_t na ekvidistantní úseky tak, jak je to vidět na obr. 1.5a, kde v okolí uzlů mohou být úseky d^i_{t1} a d^i_{t2} kratší než d^i_t, ale přitom platí jejich součet

$$d^i_t = d^i_{t1} + d^i_{t2}. \tag{1.14}$$

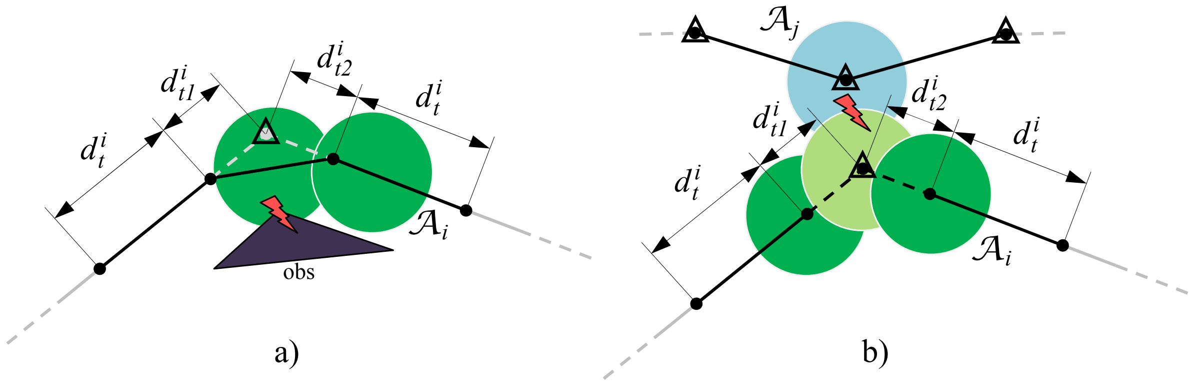
V tomto případě může a nemusí být uzel cesty τ^i zároveň konfigurací s^i_k. To závisí na tom, zda vzdálenost od počátečního uzlu cesty τ^i k danému uzlu je celým násobkem *vzorkovací vzdálenosti* d^i_t. Poslední úsek cesty značený $< d^i_t$ může být kratší než d^i_t, ale tato skutečnost je zanedbatelná.

Metoda 2: Tato metoda navzorkuje cestu τ^i *vzorkovací vzdáleností* d^i_t, která opět v okolí uzlu nemusí být dodržena s tím rozdílem, že každý uzel se stává i konfigurací s^i_k, viz obr. 1.5b. Oproti *metodě 1* se zorkuje vždy pouze hrana mezi uzly zvlášť od ostatních.

Metoda 1

Ukažme si, co se může stát, kdybychom vzorkovali cestu τ^i tak, jak je to na obr. 1.5a. Máme dvě možnosti.

První je, že se robot $\mathcal{A}_i$ nedrží přesně své cesty τ^i, pak by mohla nastat situace na obr. 1.6a, kde robot označen $\mathcal{A}_i$, reprezentován kruhem, by se držel pouze cesty vytvořené



Obrázek 1.6: Ukázky možných problémů při vzorkování cesty τ^i *Metodou 1*.

úseky mezi jednotlivými konfiguracemi vzniklými vzorkováním cesty τ^i. Toto vzorkování by zachovávalo ekvidistantní délky úseků na této cestě τ^i (až na poslední úsek), o čemž svědčí i vztah 1.14. Předpokladem tohoto tvrzení je, že robot $\mathcal{A}_i$ je naprogramován tak, aby při $u_k^i = 1$ se přemístil ze své aktuální konfigurace do následující konfigurace, podle vztahu 1.10, nejkratší přímou cestou. Důsledkem tohoto jednání by však mohla být kolize s možnou překážkou (viz obr. 1.6a útvar označený "obs").

Druhou možností je, že by se robot $\mathcal{A}_i$ držel své cesty τ^i a bylo by opět dodrženo vzorkování ekvidistantních úseků na této cestě (až na poslední úsek). Předpokládá se, že se robot $\mathcal{A}_i$ při $u_k^i = 1$ přemístí ze své aktuální konfigurace do následující konfigurace, podle vztahu 1.10, dodržujíc trajektorii cesty τ^i. V tomto případě platí obr. 1.6b. Zde je znázorněna možná kolize robotu $\mathcal{A}_i$ s jiným robotem $\mathcal{A}_j$. Její příčinou je fakt, že při vytváření *koordinačního prostoru* $\tilde{\mathcal{S}}$ není počítáno s uzlem cesty τ^i (na obr. 1.6b bod mezi úseky délek d_{t1}^i a d_{t2}^i), jako s platným stavem s_k^i, kde $k \in \tilde{\mathcal{S}}^i$.

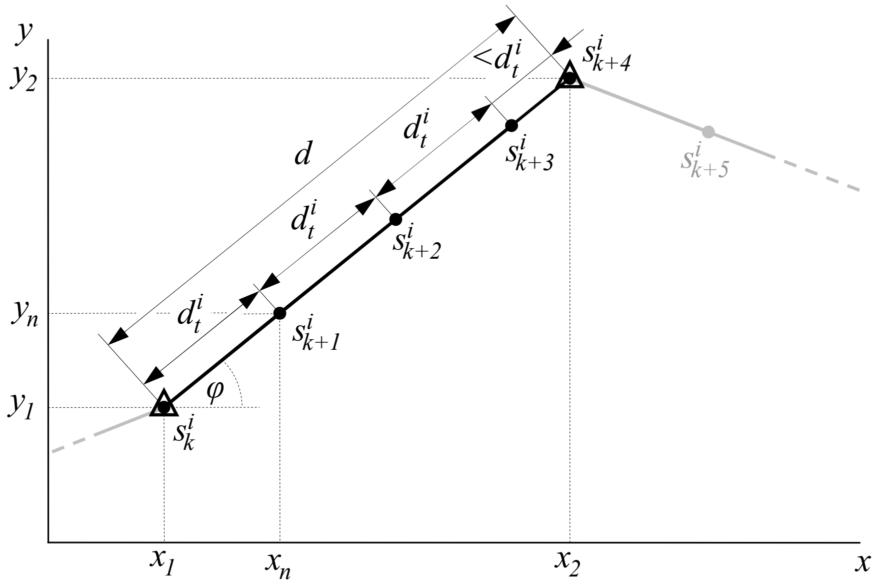
Tyto dva popsané principy, resp. samotnou *Metodu 1* považujeme tedy za nevhodné, jelikož je u nich vysoké riziko neodhalitelných kolizí.

Metoda 2

Vzorkujeme-li cestu τ^i podle obr. 1.5b, vzorkujeme každou hranu cesty τ^i (mezi dvěma uzly) zvlášť bez ohledu na ostatní hrany. Podívejme se na obr. 1.7, který znázorňuje část cesty τ^i. Pohybujeme se v souřadnicovém systému prostoru $\mathbb{R}^2$ o hlavní ose x a na ni kolmé ose y, kde bereme osu x jako referenční směr. Body označené jako $s_k^i, ..., s_{k+5}^i$ jsou konfiguracemi robotu $\mathcal{A}_i$. Hrana v tomto prostoru je úsečkou s počátečním bodem (x_1, y_1), koncovým bodem (x_2, y_2), které jsou zároveň uzly cesty τ^i, a úhlem natočení φ. Délka d této úsečky, jakožto hrany, se vypočítá jako euklidovská vzdálenost

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}. \tag{1.15}$$

Pro nalezení souřadnic všech konfigurací mezi uzly (x_1, y_1), (x_2, y_2) cesty τ^i využijeme matematické analýzy. Známe tedy body uzlů (x_1, y_1), (x_2, y_2) a *vzorkovací vzdálenost* d_t^i



Obrázek 1.7: Vzorkovaná hrana cesty τ^i.

vypočítanou podle vztahu 1.13. Chceme vypočítat souřadnice bodu (x_n, y_n), jakožto konfigurace s^i_{k+n}. S výhodou využijeme goniometrických funkcí

$$\begin{aligned} \cos\varphi &= \tfrac{x_2 - x_1}{d}, \\ \sin\varphi &= \tfrac{y_2 - y_1}{d}. \end{aligned}$$

Tyto dále uplatníme pro zbytek výpočtu

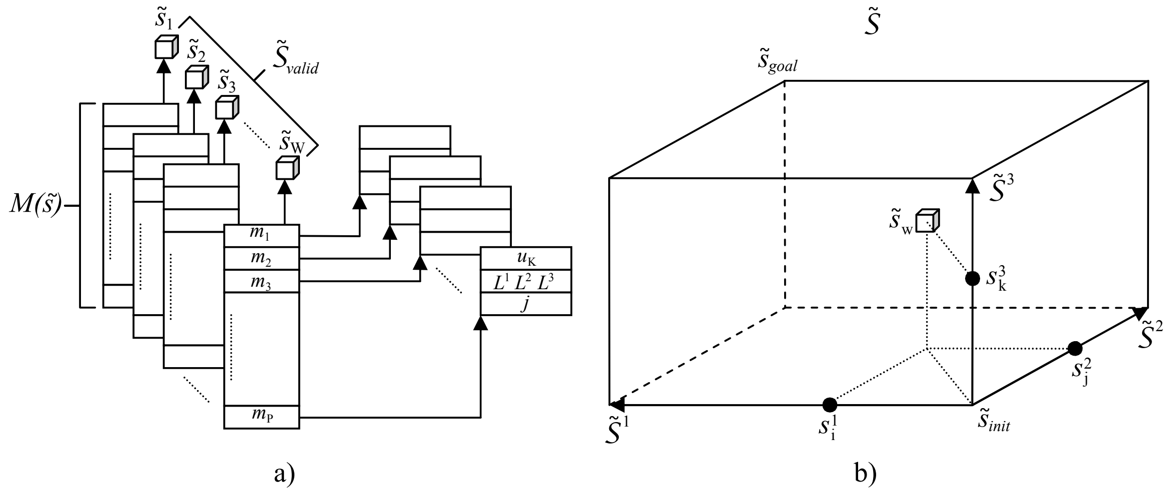
$$\begin{aligned} x_n &= n d^i_t \cos\varphi + x_1, \\ y_n &= n d^i_t \sin\varphi + y_1, \end{aligned}$$

kde n je počet úseků d^i_t od bodu (x_1, y_1), o které je vzdálena daná konfigurace s^i_{k+n}. Číslo $n \in \langle 1; N \rangle$ a zároveň $n \in \mathbb{N}$. Platí

$$N = \left\lfloor \frac{d}{d^i_t} \right\rfloor ,$$

kde $\lfloor \cdot \rfloor$ značí zaokrouhlení dolů.

Tato metoda však obsahuje také své nedostatky vyplývající z diskretizace cesty τ^i, resp. jejích hran. Poslední úsek hrany na obr. 1.7 označen jako $< d^i_t$ je kratší. Těchto hran s jedním úsekem kratším $< d^i_t$ může existovat více po celé cestě τ^i. To už vnáší značné nedostatky do optimálnosti úlohy. Znamená to, že robot $\mathcal{A}_i$ na těchto kratších úsecích $< d^i_t$ neurazí celou *vzorkovací vzdálenost* d^i_t a musí dřív zastavit. Pokud by tedy $\mathcal{A}_i$ měl naplánovanou strategii γ^i bez zastávek, přesto jeho jízda nebude plynulá. Příčinou tohoto



Obrázek 1.8: Obsah stavů *koordinačního prostoru* a), *koordinační prostor* 3 robotů b).

principu je, že robot $\mathcal{A}_i$ se musí držet své cesty τ^i, kde do konfigurací s_k^i se započítávají postupně uzly cesty τ^i a jednotlivé vzorkované konfigurace hran mezi nimi.

Řešením tohoto problému může být zvýšení rozlišení vzorkování, resp. zkrácení *vzorkovacího času* Δt, nebo rozměrové přizpůsobení načítané cesty τ^i požadavkům optimálního vzorkování. Částečně kompenzovat tento problém lze i zavedením ceny penalizace robota úměrné vzdálenosti mezi jednotlivými konfiguracemi do vztahu dílčí cenové funkce 1.11 jako součást cenové funkce $g^i(t, s^i(t), u^i(t))$.

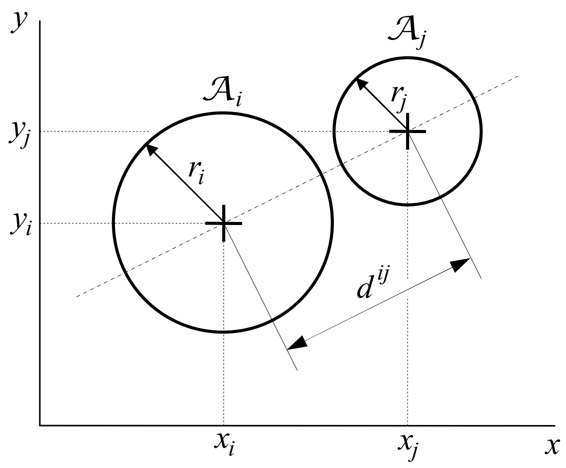
1.3.2 Diskrétní koordinační prostor

Máme již vytvořené *diskrétní konfigurační trajektorie* $\tilde{\mathcal{S}}^i$. Z nich sestavíme *diskrétní koordinační prostor* $\tilde{\mathcal{S}}$ pomocí vztahu 1.4. Tím vzniknou stavy $\tilde{s}$ jako kombinace všech konfigurací s_k^i.

Pro ilustraci budeme problém vysvětlovat na *diskrétním koordinačním prostoru* $\tilde{\mathcal{S}}$ 3 robotů dle obr. 1.8b, jelikož znázornění více jak 3-rozměrného prostoru by bylo příliš komplikované a nepřehledné. Jak na obrázku vidíme, každý rozměr $\tilde{\mathcal{S}}$ je reprezentován *diskrétní konfigurační trajektorií* $\tilde{\mathcal{S}}^i$ orientovaných ve směru šipek. Stejně tak v počátku začíná stav $\tilde{s}_{init}$ a na druhém konci $\tilde{\mathcal{S}}$ je stav $\tilde{s}_{goal}$. Stav $\tilde{s}_w$ má své souřadnice (i, j, k) dané pořadími příslušných konfigurací s_i^1, s_j^2, s_k^3 v jejich *konfiguračních prostorech* $\tilde{\mathcal{S}}^1, \tilde{\mathcal{S}}^2, \tilde{\mathcal{S}}^3$. Index $w \in \langle 1; W \rangle$, a zároveň $w \in \mathbb{N}$, kde W je počet všech stavů $\tilde{s}$ v *koordinačním prostoru* $\tilde{\mathcal{S}}$, neboli součinem počtů konfigurací všech zúčastněných *konfiguračních trajektorií.*

Kolizní stavy

Nalezení kolizních stavů $\tilde{s} \in \tilde{\mathcal{S}}_{coll}$ v *koordinačním prostoru* $\tilde{\mathcal{S}}$ se provede tak, jak je to popsáno v kapitole 1.2.2, resp. 1.2.1. Prakticky se vezme každá dvojice konfigurací s_k^i a



Obrázek 1.9: Analýza kolize mezi roboty $\mathcal{A}_i$ a $\mathcal{A}_j$ ve stavu $\tilde{s}$.

s_k^j příslušných pro stav $\tilde{s}$, které náleží robotům $\mathcal{A}_i$ a $\mathcal{A}_j$, kde $i \neq j$. Z těchto konfigurací se vyjádří body (x_i, y_i) a (x_j, y_j) (viz obr. 1.9), vypočítá se jejich vzdálenost od sebe d^{ij} pomocí vztahu euklidovské vzdálenosti 1.15. Se znalostí průměrů obou robotů r_i a r_j se zjistí, zda se kružnice $\mathcal{A}_i$ a $\mathcal{A}_j$ překrývají a došlo ke kolizi, či naopak. Platí tedy $\tilde{s} \in \mathcal{S}_{coll}$, když existuje nějaké $d^{ij} < (r^i + r^j)$. Takto se ověří každý stav $\tilde{s} \in \tilde{\mathcal{S}}$. Množina nekolizních stavů $\tilde{\mathcal{S}}_{valid}$ se z $\tilde{\mathcal{S}}_{coll}$ získá pomocí vztahu 1.6. *Diskrétní koordinační prostor* je pak binární N-rozměrný kvádr.

Okolí stavu

Definujeme podmnožinu $\mathcal{N}(\tilde{s}) \subset \tilde{S}$, kterou nazveme *okolí* stavu $\tilde{s}$. Je to množina všech sousedních stavů $\tilde{s}' \in \tilde{\mathcal{S}}_{valid}$, které mohou být ze stavu $\tilde{s}$ dosaženy v jednom kroku směrem, ve kterém je euklidovská vzdálenost mezi $\tilde{s}'$ a $\tilde{s}_{goal}$ menší než mezi $\tilde{s}$ a $\tilde{s}_{goal}$. Formálně

$$\mathcal{N}(\tilde{s}_k) = \left\{\ \tilde{s}' = f(\tilde{s}, u_k) \quad | \quad u_k \in U \quad \text{a} \quad f(\tilde{s}, u_k) \in \tilde{\mathcal{S}}_{valid}\ \right\} \tag{1.16}$$

kde U je množina všech možných akčních vektorů a funkce f je popsána ve vztahu 1.12.

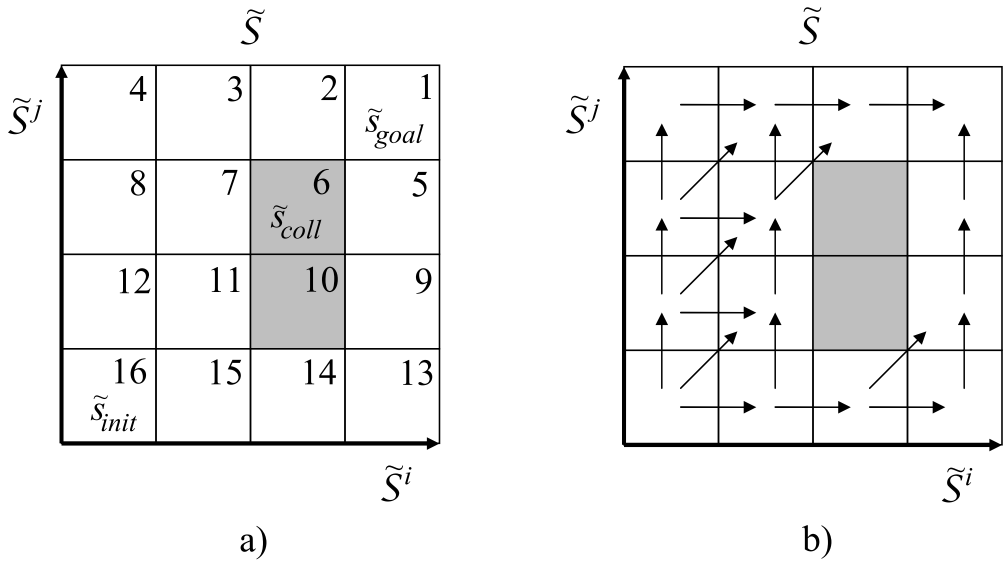
Minimální strategie

Definujme *množinu minimálních strategií* $M(\tilde{s})$. Dále definujme element $M(\tilde{s})$, nazvaný jako *minimální strategie* $m \in M(\tilde{s})$, která je definována vztahem

$$m = \langle u_k, [L^1 L^2 \cdots L^N], m' \rangle. \tag{1.17}$$

Zde u_k je vektor akcí robotů $\mathcal{A}_i$, L^i reprezentují souhrnné ceny všech $\mathcal{A}_i$ obdržené během vytváření *minimálních strategií* m. Symbol m', sloužící jako odkaz, značí následující *minimální strategii* $m' \in M(\tilde{s}')$, která, pokud $m' \notin M(\tilde{s}_{goal})$ ukazuje na další m'', atd.

Množiny $M(\tilde{s})$ lze vytvořit a připojit každému nekoliznímu stavu $\tilde{s}_w \in \tilde{\mathcal{S}}_{valid}$ tak, jak to vidíme na obr. 1.8a.



Obrázek 1.10: Ukázka 2D *diskrétního koordinačního prostoru*, kde a) je číslovaný postup uvažování jednotlivých stavů $\tilde{s}$ pro expanzi *minimálních strategií* m a b) šipkami znázorněné ukazatele *minimálních strategií* m do sousedních stavů $\tilde{s}'$, resp. jejich příslušných *minimálních strategií* m'. Oba obrázky značí totožný *diskrétní koordinační prostor.*

Expanze minimálních strategií

Mějme *koordinační prostor* $\tilde{\mathcal{S}}$ s vytvořenými stavy $\tilde{s}$, které nemají vytvořené množiny $M(\tilde{s})$. Je známa množina $\tilde{\mathcal{S}}_{valid}$. Cyklicky se budou vytvářet v každém stavu $\tilde{s} \in \tilde{\mathcal{S}}_{valid}$ množiny $M(\tilde{s})$ se svými *minimálními strategiemi* m danými vztahem 1.17 od stavu $\tilde{s}_{goal}$. Tento postup je znázorněn pseudokódem ve **scénáři** 1 a zároveň detailně popsán v následujícím odstavci.

Popis algoritmu **Scénář** 1*:*

V prvních 3 krocích se inicializují proměnné algoritmu a především ve stavu $\tilde{s}_{goal}$ se založí $M(\tilde{s})$, ve které se vytvoří $m = \langle \emptyset, [0\ 0\ ...\ 0], \emptyset \rangle$.

V cyklech reprezentovaných kroky 4-7 se budou uvažovat postupně všechny kombinace indexů reprezentující stavy $\tilde{s} \in \tilde{\mathcal{S}}$ tak, že když si pro jednoduchost představíme 2D *diskrétní koordinační prostor* jako horizontálně vertikální mřížku stavů $\tilde{s}$ (stavy v řádcích a sloupcích), viz obr. 1.10a, čísla ukazují postup uvažování stavů $\tilde{s}$.

V kroku 8 se podle indexů k^i adresuje příslušný stav $\tilde{s}$, který zatím nemá vytvořenou množinu $M(\tilde{s})$ (to je zaručeno cyklickým postupem 4-7), kde platí pro všechna $\tilde{s}' \in \mathcal{N}(\tilde{s})$, že mají vytvořenou množinu $M(\tilde{s}')$.

Krok 9-12 se provede jen tehdy, když platí $\tilde{s} \in \tilde{\mathcal{S}}_{valid}$.

Kroky 10-11 reprezentují založení $M(\tilde{s})$ v $\tilde{s}$. Dále se postupně se uvažují všechny m

Scénář 1: Pseudokód vytvoření všech minimálních strategií v *diskrtétním koordinačním prostoru* $\tilde{S}$. Přeloženo a upraveno z [3].

1: Nechť $k^i \in \{1, ..., K^i\}$, kde K^i je poč. prvků $\tilde{\mathcal{S}}^i$
2: Nechť N je počet robotů $\mathcal{A}_i$
3: Nechť $M(\tilde{s}_{goal}) = \{\langle \emptyset, [0\ 0\ ...\ 0], \emptyset \rangle\}$, všechny ostatní $M(\tilde{s}) = \emptyset$
4: **Pro každý** k^1 od K^1 do 1 **dělej**
5:     **Pro každý** k^2 od K^2 do 1 **dělej**
6:         $\ddots$
7:             **Pro každý** k^N od K^N do 1 **dělej**
8:                 Nechť $\tilde{s} = (k^1, k^2, ..., k^N)$
9:                 **Když** $\tilde{s} \in \tilde{\mathcal{S}}_{valid}$ **pak**
10:                     Nechť M_u je sjednocením všech $M(\tilde{s}')$ pro každý $\tilde{s}' \in \mathcal{N}(\tilde{s})$
11:                     Vytvoří se množina $M(\tilde{s})$ rozšířením minimálních strategií v M_u
12:                 **Konec: když**
13:             **Konec: pro každý**
14:         $\cdot^{\cdot^{\cdot}}$
15:     **Konec: pro každý**
16: **Konec: pro každý**
17: **Vrať** $M(\tilde{s}_{init})$

v každé $M(\tilde{s}')$ náležící každému $\tilde{s}'$, viz obr. 1.10b, kde šipky v každé $\tilde{s}$, reprezentovaném nejmenším čtvercem, ukazují na všechny tyto sousední stavy $\tilde{s}' \in \mathcal{N}(\tilde{s})$. Pro aktuální uvažovanou $m \in M(\tilde{s}')$ se vytvoří $m \in M(\tilde{s})$, do které se uloží u_k takové, aby platil vztah $\tilde{s}' = f(\tilde{s}, u_k)$ jako ve vztahu 1.12. Dále se převezme z $m \in M(\tilde{s}')$ vektor souhrnných cen $[L^{1'} L^{2'} \cdots L^{N'}]$, kde pro $m \in M(\tilde{s})$ platí u každé $L^i = L^{i'} + l_k^i(\tilde{s}^i, u_k^i)$, kde $i \in \{1, ..., N\}$ a $l_k^i(\tilde{s}^i, u_k^i)$ odpovídá vztahu 1.11, za předpokladu $\tilde{s}^i \neq s_{goal}^i$. V opačném případě platí $L^i = 0$. Nakonec se symbolu m' v $m \in M(\tilde{s})$ přiřadí odkaz právě na použitou $m \in M(\tilde{s}')$. Pokud již byly uvažovány všechny m každé $M(\tilde{s}')$ náležící každému $\tilde{s}'$, pokračuje se opět v cyklech v krocích 4-7.

Po ukončení všech cyklů se přejde ke kroku 17, kde se vrátí množina počátečního stavu $M(\tilde{s}_{init})$, ve kterém se obecně nachází velké množství *minimálních strategií m*. Pokud je však množina $M(\tilde{s}_{init})$ prázdná, pak je úloha koordinace neřešitelná, tzn., že neexistuje možnost, jak bezkolizně zkoordinovat roboty $\mathcal{A}_i$ na jejich fixních cestách τ^i.

Výsledkem tohoto postupu jsou celé strategie γ reprezentovány *minimálními strategiemi m* v množině $M(\tilde{s}_{init})$ počátečního stavu $\tilde{s}_{init}$ a jejich následníky danými odkazovým symbolem m'. Mezi nimi se dají vybrat takové γ^*, které mají *kvocient strategie* $[\gamma^*]_L$ *minimální*. Ty by již byly použitelné pro optimální koordinaci. Nevýhodou však je vysoká časová i paměťová náročnost.

Redukce minimálních strategií

Algoritmus může vytvářet velký počet *minimálních strategií* m v každé příslušné množině $M(\tilde{s})$. Z důvodů časové a paměťové náročnosti, jak již bylo uvedeno v předchozí sekci, je někdy vhodné tyto *minimální strategie* m redukovat již v průběhu jejich vytváření, viz **scénář** 1.

Proto je zapotřebí selekce *minimálních strategií* m v každé množině $M(\tilde{s})$ každého stavu $\tilde{s}$ již v průběhu vytváření. Jednou z metod je součet všech souhrnných cen L^i vektoru souhrnných cen každé *minimální strategie* m v daném expandovaném stavu $\tilde{s}$, jejich porovnání a výběr *minimální strategie* m s nejnižším součtem souhrnných cen L^i. Těch však může být i víc. Redukovat je lze navíc náhodným výběrem jedné nebo více *minimálních strategií* m. Předpokladem tohoto uvažování je uniformita cen placených jako souhrnné ceny L^i. Dále lze zvýhodnit postupně každý robot $\mathcal{A}_i$ zvlášť vybráním *minimálních strategií* m s nejnižší souhrnnou cenou L^i robotu $\mathcal{A}_i$. Pokud jich bude víc, redukují se například součtem všech ostatních souhrnných cen L^j, kde $i \neq j$, a opětovným vybráním m s nejnižším součtem, ev. navíc náhodným výběrem, při zůstatku více takových m.

Destinační čas

Cyklický postup vytváření *minimálních strategií* v sekci **Expanze minimálních strategií** nepočítá s funkcí nedodržení *destinačního času* T^i každého robotu $\mathcal{A}_i$ popsané ve vztahu 1.3. Pro uplatnění této funkce by *minimální strategie* m musela navíc obsahovat informace o časech (označme si je jako t^i_m), které trvají při přechodech právě z aktuální m po jejích následnících až do každého cílových stavů s^i_{goal}. Resp. by stačilo, aby byly známy počty těchto přechodů. Ty by se pak vynásobily *vzorkovacím časem* Δt. Při znalosti zmíněných časů t^i_m pro aktuálně vytvářenou m by pak stačilo porovnat s T^i. Pokud čas $t^i_m > T^i$ pro nějaký index i, dojde k zániku této m a tím i jejím následníkům.

Předběžné zjištění, zda nějaký čas t^i_m s indexem i pro m někde v průběhu vytváření ještě nevytvořených předchůdců m určitě dosáhne *destinačního času* T^i lze pomocí správně zvolené heuristické funkce, například euklidovské vzdálenosti, směřované k počátečnímu stavu $\tilde{s}_{init}$.

1.3.3 Alternativní hledání stavové trajektorie

Může se stát, že hardware, na kterém je počítán koordinační algoritmus, není schopen v rozumném časovém horizontu, nebo z důvodů paměťové náročnosti, nalézt řešení koordinace. Pak se tedy musí přistoupit k alternativním metodám výpočtu.

Definujme cenu L^i robota $\mathcal{A}_i$ vztahem

$$L^i = \begin{cases} l^i_k(\tilde{s}^i, u^i_k) & \text{když} \quad \tilde{s}^i \neq s^i_{goal}, \\ 0 & \text{když} \quad \tilde{s}^i = s^i_{goal}. \end{cases} \tag{1.18}$$

Funkce $l^i_k(\tilde{s}^i, u^i_k)$ odpovídá vztahu 1.11. Dále ve vztahu 1.17 pro *minimální strategii* změňme odkaz na následující m' na odkaz na následující stav $\tilde{s}'$ (což může být v souvislosti s přítomným akčním vektorem u_k redundantní informací).

Expanze minimálních strategií

Narozdíl od sekce **Expanze minimálních strategií** v kapitole 1.3.2, bude postup zde značně zjednodušen.

Mějme *koordinační prostor* $\tilde{\mathcal{S}}$ s vytvořenými stavy $\tilde{s}$. Je známa množina $\tilde{\mathcal{S}}_{valid}$, kde každý stav $\tilde{s} \in \tilde{\mathcal{S}}_{valid}$ již má vytvořenou množinu $M(\tilde{s})$. Pro každou tuto množinu $M(\tilde{s})$ v každém stavu $\tilde{s} \in \tilde{\mathcal{S}}_{valid}$, kromě $\tilde{s}_{goal}$, v jehož $M(\tilde{s}_{goal})$ bude $m = \langle \emptyset, [0\ 0\ \ldots\ 0], \emptyset \rangle$, se vytvoří všechny *minimální strategie* $m \in M(\tilde{s})$ takové, aby každá z nich odkazovala na jeden z následujících stavů $\tilde{s}' \in \mathcal{N}(\tilde{s})$. Tedy do každé *minimální strategie* $m \in M(\tilde{s})$ se uloží u_k takové, aby platil vztah $\tilde{s}' = f(\tilde{s}, u_k)$ jako ve vztahu 1.12. Dále pro každou cenu L^i platí vztah 1.18.

Výsledkem této expanze však vznikne pouze stavový prostor pospojovaný *minimálními strategiemi* m se známou cenou každého přechodu mezi stavy a možným směrem prohledávání. To však stačí k použití některého z algoritmů prohledávání stavového prostoru, jako jsou například BFS, viz [2], nebo A*, viz [5], které najdou optimální *strategii* γ^*.

Jejich výhodou, především algoritmu A*, který využívá heuristických funkcí, je značené ušetření času a paměťového prostoru pro výpočet narozdíl od metody popsané v kapitole 1.3.2.

Kapitola 2

Koordinace robotů s nezávislými mapami cest

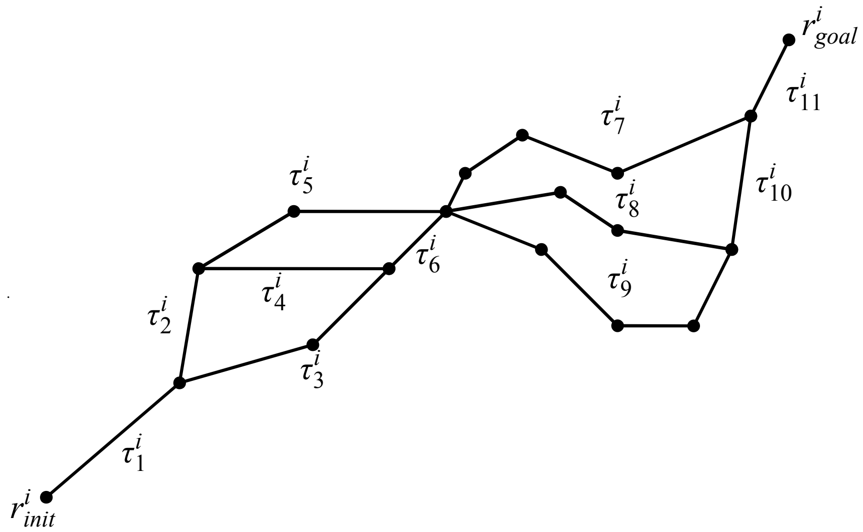
2.1 Úvod do problematiky

Tato kapitola se podobně jako kapitola 1 zabývá koordinací více mobilních robotů v prostoru s tím rozdílem, že se roboty pohybují po nezávislých *mapách cest*. Z počáteční pozice do cíle se roboty mohou dostat různými, jim definovanými cestami, které jsou vzájemně propojeny. Tato problematika je s problematikou v kapitole 1 v mnoha ohledech podobná a využívá často i podobných definic a algoritmů, proto popíšeme pouze rozdíly mezi nimi.

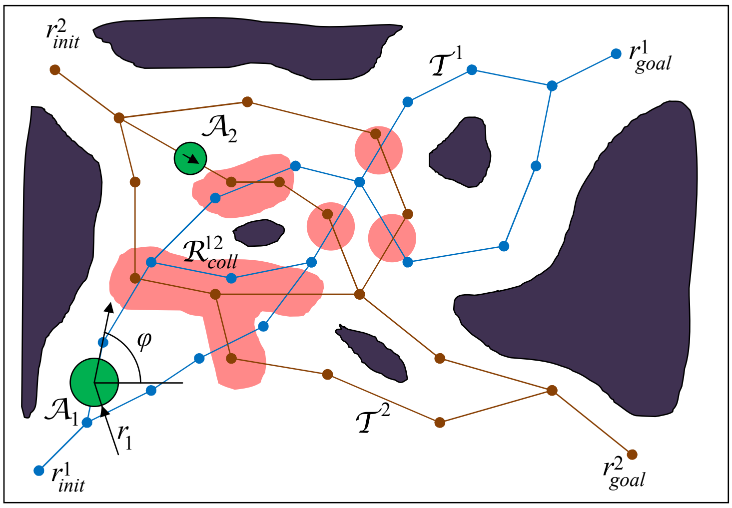
Jak již bylo zmíněno, robot se pohybuje po *mapě cest*, viz obr. 2.1, která, jak z názvu vyplývá, je reprezentována množinou vzájemně propojených dílčích cest, které odpovídají cestám definovaných v kapitole 1. Minimálně mohou být v jednom uzlu propojeny tři cesty, jelikož propojení pouze dvou cest by dávalo opět v součtu cestu jednu. Tento spojovací uzel označme *rozcestí*. Rozdělení cest na konfigurace robotů je taktéž stejné jako v kapitole 1. K množině příkazů robotu "stůj", "jeď" je nutné navíc definovat příkaz "jeď definovanou cestou", který je platný na *rozcestí*. Ten definuje robotu, jakou následnou cestou se má robot vydat.

Uvažujeme-li více robotů ve stejném prostoru s vlastními *mapami cest*, pak počítáme s tím, že se navzájem mohou křížit a vytvořit tak opět možnosti kolizí, viz obr. 2.2. Obdobně jako v kapitole 1 se vytvoří *koordinační prostor map*, jehož topologie je mnohem komplikovanější než v případě *koordinačního prostoru* robotů s fixními cestami. Množiny indexů konfigurací *map cest*, jejichž kartézským součinem se získá právě *koordinační prostor map*, musejí být napřed sestaveny algoritmem zvaným *vlnoplocha*, který bude popsán později.

V případě *map cest* mají roboty k dispozici víc možností, jak se vyhnout kolizím. Pokud kolize hrozí, mohou nejen zastavit a počkat na volnou cestu, ale i zvolit možnost změnit cestu, což může být někdy výhodnější než možnost první. Hledání *stavové trajektorie* je, i přes značnou komplikovanost *koordinačního prostoru map*, velmi podobné hledání *stavové trajektorii koordinačního prostoru* robotů s fixními cestami, za podmínky, že před sestave-



Obrázek 2.1: *Mapa cest* $\mathcal{T}^i$ jednoho robotu rozdělená na cesty τ^i_k.



Obrázek 2.2: Dvourozměrný prostor, kde se pohybují roboty $\mathcal{A}_i$, s poloměrem r_i a směrem φ, po *mapách cest* $\mathcal{T}^i$ s počáteční a koncovou konfigurací r^i_{init} a r^i_{goal}. Množiny kolizních stavů jsou $\mathcal{R}^{ij}_{coll}$.

ním *koordinačního prostoru map* byl použit algoritmus *vlnoplocha*. Pro *stavovou trajektorii koordinačního prostoru map* platí analogicky stejná pravidla popsaná v kapitole 1.1.

2.2 Základní definice

V kapitole 1 je stanovena většina definic, které platí i zde. Uvedeme si pouze nové nebo rozdíly od těch původních.

Předpokládejme, že robot $\mathcal{A}_i$ se pohybuje po hranách orientovaného grafu, který nazveme *mapa cest* $\mathcal{T}^i$, přičemž pro různé roboty se tento graf může lišit.

Tato *mapa cest* $\mathcal{T}^i$ je tvořena dílčími úseky τ_j^i, kde index $j \in \langle 1; M \rangle$ a zároveň $j \in \mathbb{N}$. Číslo M udává počet těchto obsažených úseků τ_j^i v $\mathcal{T}^i$. Platí tedy $\tau_j^i \in \mathcal{T}^i$.

Definujme *rozcestí* J^i, jako uzel společný více než jedné cestě $\tau_j^i \in \mathcal{T}^i$. Tento uzel je koncovým pro každou znich. Tzn., že nemůže být cesta τ_j^i, v jejímž průběhu je takový uzel, který není ani na jednom jejím konci a zároveň je *rozcestím*. *Rozcestí* J^i obsahuje množinu odkazů vstupujících cest I^i do tohoto *rozcestí* a množinu odkazů vystupujících cest O^i z tohoto *rozcestí*. Platí $J^i = \{I^i, O^i\}$.

2.2.1 Koordinační prostor map

Přistupujme k tomuto problému přímo pomocí diskrétních veličin.

Konfiguraci s_k^i robotu $\mathcal{A}_i$ zde budeme značit jako r_k^i a bude mít stejné parametry. Každá konfigurace r_k^i je výsledkem vzorkování všech cest $\tau_j^i \in \mathcal{T}^i$.

Nechť $\mathcal{R}^i$ je množina všech indexů k konfigurací r_k^i. Formálně $k \in \mathcal{R}^i$. Ta je podobná *diskrétní konfigurační trajektorii* $\mathcal{S}^i$. Indexy k konfigurací r_k^i jsou do $\mathcal{R}^i$ řazeny pomocí algoritmu zvaného *vlnoplocha*, který bude vysvětlen později. Počáteční konfigurace je r_{init}^i a cílová je r_{init}^i, které jsou opět analogickým odvozením konfigurací s_{init}^i a s_{goal}^i.

Definujme *koordinační prostor map* $\mathcal{R}$, který je kartézským součinem

$$\mathcal{R} = \mathcal{R}^1 \times \mathcal{R}^2 \times \cdots \times \mathcal{R}^N. \tag{2.1}$$

Stavy *koordinačního prostoru map* $\mathcal{R}$ jsou $r = (r^1, r^2, ..., r^N)$.

Akce u_k^i je nutno rozšířit na $u_k^i \in \{0, ..., Z\}$, kde $Z = 1$, pokud se robot $\mathcal{A}_i$ nachází v konfiguraci r_k^i, která je na cestě τ_j^i a zároveň není *rozcestím* J^i. Pokud se robot $\mathcal{A}_i$ nachází na zmíněném *rozcestí* J^i, pak Z je rovno počtu cest τ_j^i z *rozcestí* vystupujících, neboli počtu odkazů v O^i. Ve výsledku pokud $u_k^i = 0$, znamená to pro robot $\mathcal{A}_i$ "stůj". Pokud $u_k^i \neq 0$, znamená to "jeď definovanou cestou" na kterou odkazuje O^i. Vektor akcí je stejný $u_k = (u_k^1, u_k^2, ..., u_k^N)$.

Konfigurace $(r_k^i)'$ je následující konfigurace, která je definována, podobně jako ve vztahu 1.10, vztahem

$$(r_k^i)' = f^i(r_k^i, u_k^i), \tag{2.2}$$

kde funkce f^i ze zadané konfigurace r_k^i a akce u_k^i vypočítá novou konfiguraci $(r_k^i)'$. Platí

$$f^i(r_k^i, u_k^i) = \begin{cases} r_k^i & \text{když} \quad u_k^i = 0, \\ r_z^i & \text{když} \quad u_k^i \in \{1, ..., Z\}, \end{cases}$$

kde r_z^i je následující (nejbližší) konfigurace na cestě τ_j^i společné i konfiguraci r_k^i, nebo na cestě τ_j^i jejíž odkaz patří do množiny vystupujících cest O^i z *rozcestí* J^i, pokud r_k^i je *rozcestí* J^i. Číslo udané akcí u_k^i je pak indexem odkazu v množině odkazů vystupujících cest O^i, patřících tomuto *rozcestí* J^i.

Následující stav r' je dán analogicky, podobně jako $\tilde{s}'_k$ ve vztahu 1.12 a zároveň podle vztahu 2.2 vztahem

$$\tilde{r}' = f(r_k, u_k). \tag{2.3}$$

Kolizní množina $\mathcal{R}_{coll}$ a její doplněk nekolizní množina $\mathcal{R}_{valid}$ jsou získány obdobně jako $\tilde{\mathcal{S}}_{coll}$ a $\tilde{\mathcal{S}}_{valid}$ v kapitole 1.2.2.

2.3 Algoritmus koordinace

Uvažujme *koordinační prostor map* popsaný v kapitole 2.2.1. Vstupem algoritmu jsou číslované seznamy souřadnic uzlů cest τ^i, a jejich spojení pro každý definovaný robot $\mathcal{A}_i$. Výstupem jsou *strategie* γ^* a jejich *stavové trajektorie* α_{γ^*}.

2.3.1 Mapa cest

Mapa cest $\mathcal{T}^i$ je načtena jako seznam očíslovaných souřadnic uzlů (`1 x1 y1 2 x2 y2 ... n xn yn`), tedy, že každý uzel má své číslo. Dále je načten seznam spojení jednotlivých uzlů (resp. hran) jako dvojice čísel, jejichž pořadí udává následnost dvou uzlů s těmito čísly např. (`1 2 2 3 2 4 3 5 ...`), tedy že uzel 1 je spojen s uzlem 2, uzel 2 je spojen s uzlem 3 a zároveň s uzlem 4, uzel 3 je spojen s uzlem 5, atd.

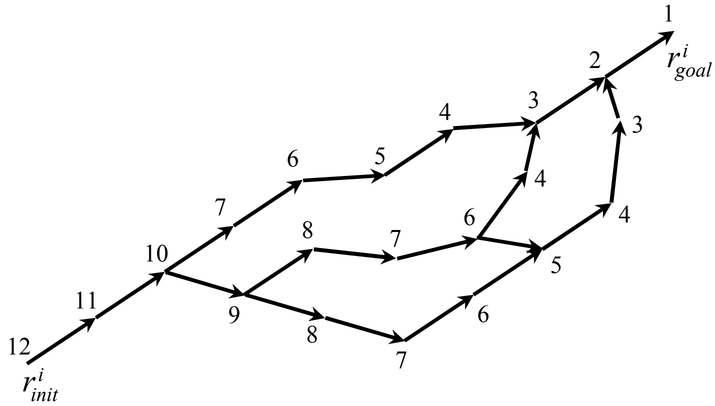
Vzorkování mapy cest

Účelem je nalézt všechny konfigurace r_k^i v *mapě cest* $\mathcal{T}^i$, které jsou od sebe vzdáleny maximálně o vzdálenost danou součinem $v_i \Delta t$, a zařadit jejich indexy do množiny $\mathcal{R}^i$. Stejně jako v sekci **Vzorkování cesty** kapitoly 1.3.1 použijeme ***Metodu 2*** ke vzorkování každé hrany mezi dvěma uzly v *mapě cest* $\mathcal{T}^i$. Získáme tak všechny konfigurace r_k^i. Jelikož tedy platí ***Metoda 2*** vzorkování každé hrany, ve výsledku nerozlišujeme úseky τ_j^i v *mapě cest* $\mathcal{T}^i$ ale pouze již zmíněné jednotlivé hrany.

V tomto případě považujme každý r_k^i za *rozcestí* $J_k^i = \{I_k^i, O_k^i\}$ s předchůdci I_k^i a následníky O_k^i, kterými jsou okolní r_k^i.

Vlnoplocha

Algoritmus *vlnoplocha* (z anglického *wavefront*) zaručí, že indexy k konfigurací r_k^i se seřadí v množině $\mathcal{R}^i$ v pořadí vhodném pro další zpracování. Jak z názvu vyplývá, můžeme jej přirovnat vlně, která prochází určeným prostorem a postupně obsáhne všechny prvky v něm v pořadí začínajícím u zdroje této vlny.



Obrázek 2.3: Navzorkovaná *mapa cest* $\mathcal{T}^i$ ilustrující algoritmus *vlnoplochy*.

Postup je znázorněn na obr. 2.3. Zde vidíme navzorkovanou *mapu cest* $\mathcal{T}^i$ s počáteční konfigurací r^i_{init} a cílovou r^i_{goal}. Čísla udávají pořadí kroků, ve kterých se indexy k konfigurací r^i_k vkládají do množiny $\mathcal{R}^i$. Ekvivalentní hodnoty čísel značí, že indexy daných konfigurací r^i_k se vkládají do $\mathcal{R}^i$ ve stejnou dobu, ale ve výsledku nezáleží na jejich pořadí. Začíná se od cílové konfigurace r^i_{goal} po krocích směrem zpět k r^i_{init} a to tak, že každá konfigurace r^i_k, která má více jak jednoho následníka O^i_k, musí čekat na zařazení svého indexu k do $\mathcal{R}^i$, dokud nebudou právě indexy těchto následníků již vloženy do $\mathcal{R}^i$.

2.3.2 Koordinační prostor map

Když jsou již vytvořeny množiny $\mathcal{R}^i$ všech robotů $\mathcal{A}_i$, sestavíme z nich *koordinační prostor map* $\mathcal{R}$ podle vztahu kartézského součinu 2.1. Vzniknou tak stavy r jako kombinace všech konfigurací r^i_k.

Nalezení kolizních a nekolizních množin stavů $\mathcal{R}_{coll}$ a $\mathcal{R}_{valid}$ je analogicky popsáno s hledáním $\tilde{\mathcal{S}}_{coll}$ a $\tilde{\mathcal{S}}_{valid}$ v sekci **Kolizní stavy** kapitoly 1.3.2.

Okolí $\mathcal{N}(r_k)$ stavu r_k je opět analogicky popsáno úpravou vztahu 1.16 pomocí vztahu 2.3.

Minimální strategie m je stejná jako ve vztahu 1.17 a jejich množina $M(r)$ je podobná množině $M(\tilde{s})$. Tedy $m \in M(r)$.

Expanzi minimálních strategií v *koordinačním prostoru map* $\mathcal{R}$ můžeme provést podobně tak, jak je to popsáno v sekci **Expanze minimálních strategií** kapitoly 1.3.2 pro stavy r. Především lze s výhodou využít upraveného pseudokódu pro $\mathcal{R}$ ve **scénáři** 1, který značně ušetří čas hledání správné r. Tento postup je zaručen správným seřazením indexů v množinách $\mathcal{R}^i$ všech robotů $\mathcal{A}_i$ použitím algoritmu *vlnoplocha* popsaného v kapitole 2.3.1.

Redukce minimálních strategií $m \in M(r)$ pomocí souhrnných cen L^i a *destinačních časů* T^i je obdobná jako v kapitole 1.3.2.

Výsledkem jsou celé strategie γ^* s *minimálním kvocientem strategie* $[\gamma^*]_L$ reprezentovány *minimálními strategiemi* m v množině $M(r_{init})$ počátečního stavu r_{init} a jejich následníky danými odkazovým symbolem m'.

Kapitola 3

Experimenty koordinace robotů s pevně stanovenými cestami

Tato kapitola se zabývá experimenty za účelem ověření správné funkce implementovaného algoritmu koordinace robotů s pevně stanovenými cestami, viz kapitola 1, a ukázkou jeho nedostatků vzhledem k diskretizaci problému.

V kapitolách 3.1 a **??** ověříme ovlivnitelnost času výpočtu koordinace množstvím robotů v prostoru a velikostí jejich poloměrů. Kapitola 3.3 ukazuje nedostatky algoritmu závislé na velikosti periody vzorkování Δt a jejich možné řešení.

Experimenty probíhaly na notebooku Asus K50AB-SX031 jehož relevantní parametry jsou:

Procesor:	AMD Athlon 64 X2, 2100MHz
Operační paměť:	DDR2, 800MHz, 3072MB
Pevný disk:	SATA, 5400 ot./min
Operační systém:	Windows 7

Algoritmus byl implementován v programovacím jazyku Java verze 1.6.0 od firmy Sun [1]. Nastavení virtuální paměti je dáno parametricky: `-Xms1024m -Xmx1024m`.

V popisu experimentů algoritmu koordiance robotů s pevně stanovenými cestami jsou všechny vzdálenosti v metrech a čas v sekundách. Stanovili jsme neměnné parametry robotů $\mathcal{A}_i$, jako jejich rychlost $v_i = 1ms^{-1}$ a *destinační čas* T_i neomezený, tzn. nemá na experimenty vliv.

Uveďme symboly, které se u experimentů budou uplatňovat a definujme některé nové v následující tabulce:

Veličina	Jednotka	Popis
Δt	s	Perioda vzorkování.
N	$-$	Počet robotů.
r_i	m	Poloměr robota $\mathcal{A}_i$.
t_{total}	s	Doba trvání výpočtu celé koordinace.
$t_{\tilde{\mathcal{S}}}$	s	Doba trvání sestavení *koordinačního prostoru* (nalezení kolizních stavů $\tilde{s} \in \tilde{\mathcal{S}}_{coll}$).
$t_{\forall M(\tilde{s})}$	s	Doba trvání vytvoření všech *minimálních strategií* (resp. jejich množin $M(\tilde{s})$).
$K_{\tilde{\mathcal{S}}}$	$-$	Celkový počet stavů $\tilde{s} \in \tilde{\mathcal{S}}$.
$K_{\tilde{\mathcal{S}}_{coll}}$	$-$	Počet všech kolizních stavů.
$K_{M(\tilde{s})}$	$-$	Počet všech minimálních strategií.
K_{γ^*}	$-$	Počet nalezených *strategií*.

3.1 Vliv množství robotů na čas výpočtu koordinace

Experimentem zjistíme, jak ovlivní počet robotů ve stejném prostoru, jejichž cesty se kříží, čas výpočtu koordinace mezi nimi.

3.1.1 Realizace

K dispozici máme konfiguraci cest robotů jako mřížku, viz obr. 3.1, kde se každý robot musí dostat z pozice počáteční (na obr. znázorněnou kružnicí, která robota reprezentuje), přímou cestou na druhý konec cesty do pozice cílové. Experiment se provádí zvlášť pro různé počty robotů N od 1 do 16, které se pro každý test zvlášť přidávají dle pořadí napsaném v téže obrázku. Zadán je stejný poloměr pro všechny roboty $r_i = 4m$ a perioda vzorkování $\Delta t = 100s$. Nejdůležitějším měřeným parametrem bude celkový uplynulý čas výpočtu koordinace $t_{total}[s]$.

3.1.2 Výsledky

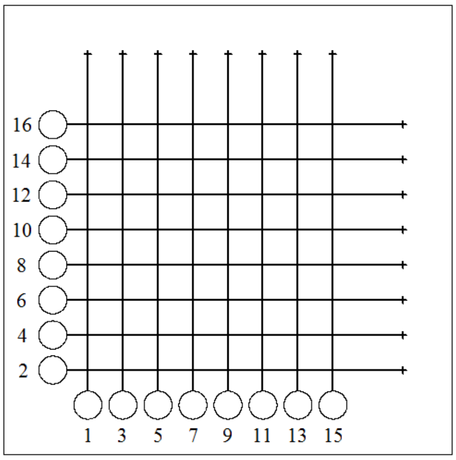
V tab. 3.1 jsou zobrazeny výsledky měření. Ačkoliv se předpokládalo, že experiment bude proveden postupně pro všech 1 až 16 robotů, trvání výpočtu algoritmu t_{total} rostlo exponenciálně se stoupajícím počtem robotů N, což je zřejmé i z grafu na obr. 3.2, kde osa času t, resp. t_{total}, je logaritmická. Za těchto okolností již při počtu robotů $N = 11$ dosahoval čas běhu testu v řádu hodin. Výpočet pro 12 a více robotů by trval přibližně přes 3 hodiny a dále by exponenciálně rostl, proto jsme experiment předčasně ukončili.

3.1.3 Závěr

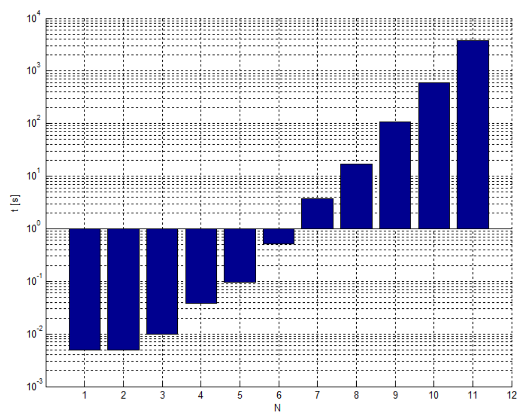
S rostoucím počtem robotů stoupají i exponenciálně nároky na výpočetní proces. V praxi to znamená, že čím více robotů chceme koordinovat, tím výkonnější počítač potřebujeme,

N	$t_{total}[s]$	$t_{\tilde{\mathcal{S}}}[s]$	$t_{\forall M(\tilde{s})}[s]$	$K_{\tilde{\mathcal{S}}}$
1	0.005	0.003	0.002	3
2	0.005	0.003	0.002	9
3	0.010	0.005	0.005	27
4	0.038	0.005	0.033	81
5	0.097	0.007	0.090	243
6	0.516	0.007	0.509	729
7	3.766	0.017	3.749	2187
8	16.918	0.033	16.885	6561
9	105.998	0.123	105.875	19683
10	604.514	0.485	604.029	59049
11	3774.004	1.638	3772.366	177147

Tabulka 3.1: Výsledky experimentu závislosti časové náročnosti výpočtu koordinace na počtu robotů N. Vzorkovací perioda je $\Delta t = 100s$, počet kolizních stavů a výsledných strategií je $\forall N(K_{\tilde{\mathcal{S}}_{coll}} = 0, K_{\gamma^*} = 1)$, celkový počet *minimálních strategií* je $K_{M(\tilde{s})} = K_{\tilde{\mathcal{S}}}$.



Obrázek 3.1: Konfigurace cest 16 robotů, pro které se počítá koordinace s postupným přidáváním robotů v pořadí dle přidělených čísel.



Obrázek 3.2: Graf experimentu ukazující závislost velikosti doby t_{total} (logaritmické měřítko) potřebné k výpočtu koordinace na počtu robotů N v prostoru.

abychom prováděli výpočty v reálném čase, což je obvyklým požadavkem zpětnovazebného řízení robotů.

3.2 Vliv poloměrů robotů na výpočet koordinace

V další sadě experimentů byl zkoumán vliv, jaký mají různé poloměry robotů na výpočet koordinace, resp. její parametry jako čas výpočtu či množství kolizních stavů.

3.2.1 Realizace

Uvažujme konfiguraci cest 3 robotů znázorněnou na obr. 3.3. Pro každé jedno měření se bude brát v úvahu jeden poloměr r, různý od ostatních měření, společný všem robotům. Účelem je hledání všech hodnot, které již byly uvedeny v kapitole 3.1 v závislosti na zmíněném poloměru r. Zadán je počet robotů $N = 3$, který je neměnný, perioda vzorkování je $\Delta t = 5s$.

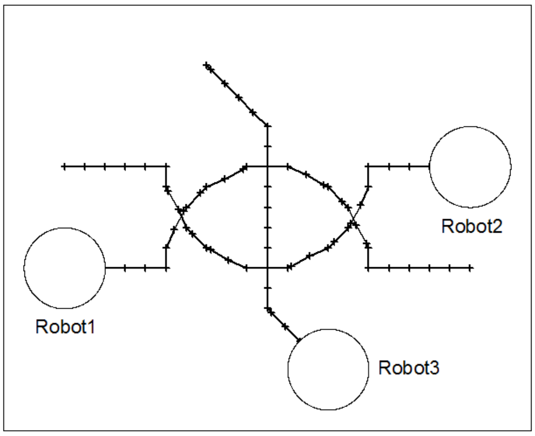
3.2.2 Výsledky

V tab. 3.2 jsou naměřené hodnoty pro tento experiment. V grafu na obr. 3.7 (vlevo) je vidět, že se stoupajícím poloměrem robotů r klesá i čas výpočtu t_{total}. Je to způsobeno především klesajícím množstvím *minimálních strategií*, jak je vidět na obr. 3.7 (vpravo). Obojí je důsledkem stoupajícího množství kolizních stavů v *koordinačním prostoru*, který zapříčiňuje úbytek nekolizních stavů $\tilde{s} \in \tilde{\mathcal{S}}_{valid}$, ve kterých je možné vytvářet zmíněné *minimální strategie*, resp. v příslušných množinách $M(\tilde{s})$.

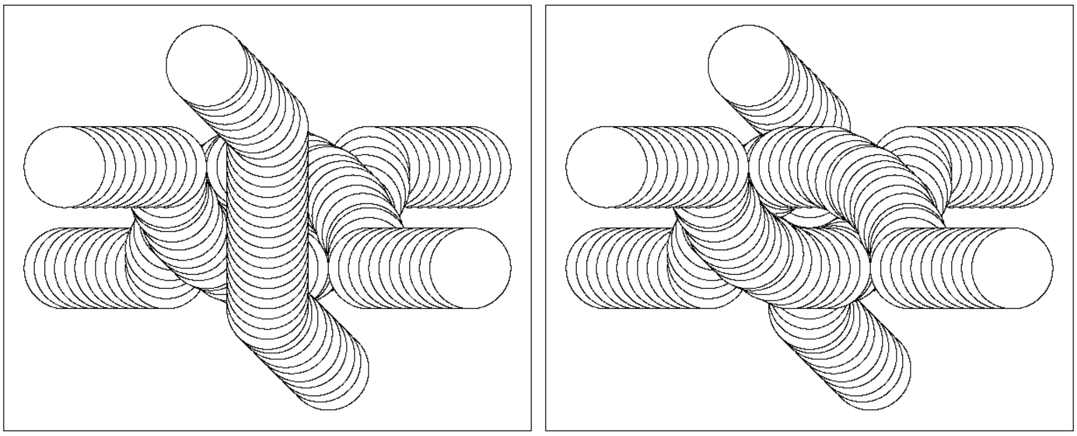
Na obr. 3.4 jsou vidět ukázky scénářů dvou různých *strategií* γ^* (viz obr. 3.5), kde lze intuitivně rozpoznat, v jakém pořadí projely roboty své cesty. Na levém z těchto dvou obrázků je zřejmé, že robot 3 mající prostřední cestu od spoda nahoru, projel prostorem až poslední po obou ostatních robotech, zatímco v pravém obrázku je tomu naopak, tzn., že tentýž robot projel prostorem jako první. Na obr. 3.6 jsou ukázky *koordinačního prostoru* téže konfigurace cest, viz obr. 3.3. Tyto *koordinační prostory* obecně korespondují s 3D *koordinačním prostorem* na obr. 1.3 s tím rozdílem, že jsou diskrétní, o čemž svědčí i jednotlivé kolizní stavy zobrazené jako krychle. Je zřejmé, že pokud je *koordinační prostor* zahlcen velkým množstvím kolizních stavů, pak řešení koordinace neexistuje, tzn. neexistuje žádná *stavová trajektorie* α_γ, viz obr. 3.6 zcela dole.

3.2.3 Závěr

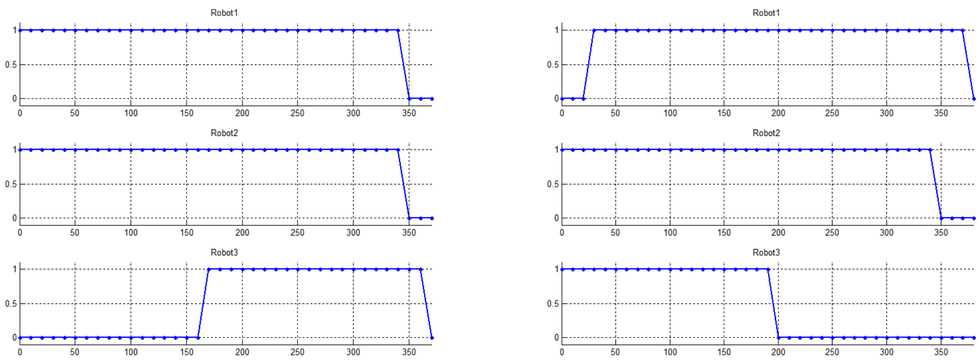
Vrátíme-li se opět k obr. 3.7 (vlevo), vidíme zde dvě zřetelné špičky vymykající se trendu křivky. Ty jsou způsobeny nesprávným měřením časů t_{total}, který byl získáván jako rozdíl



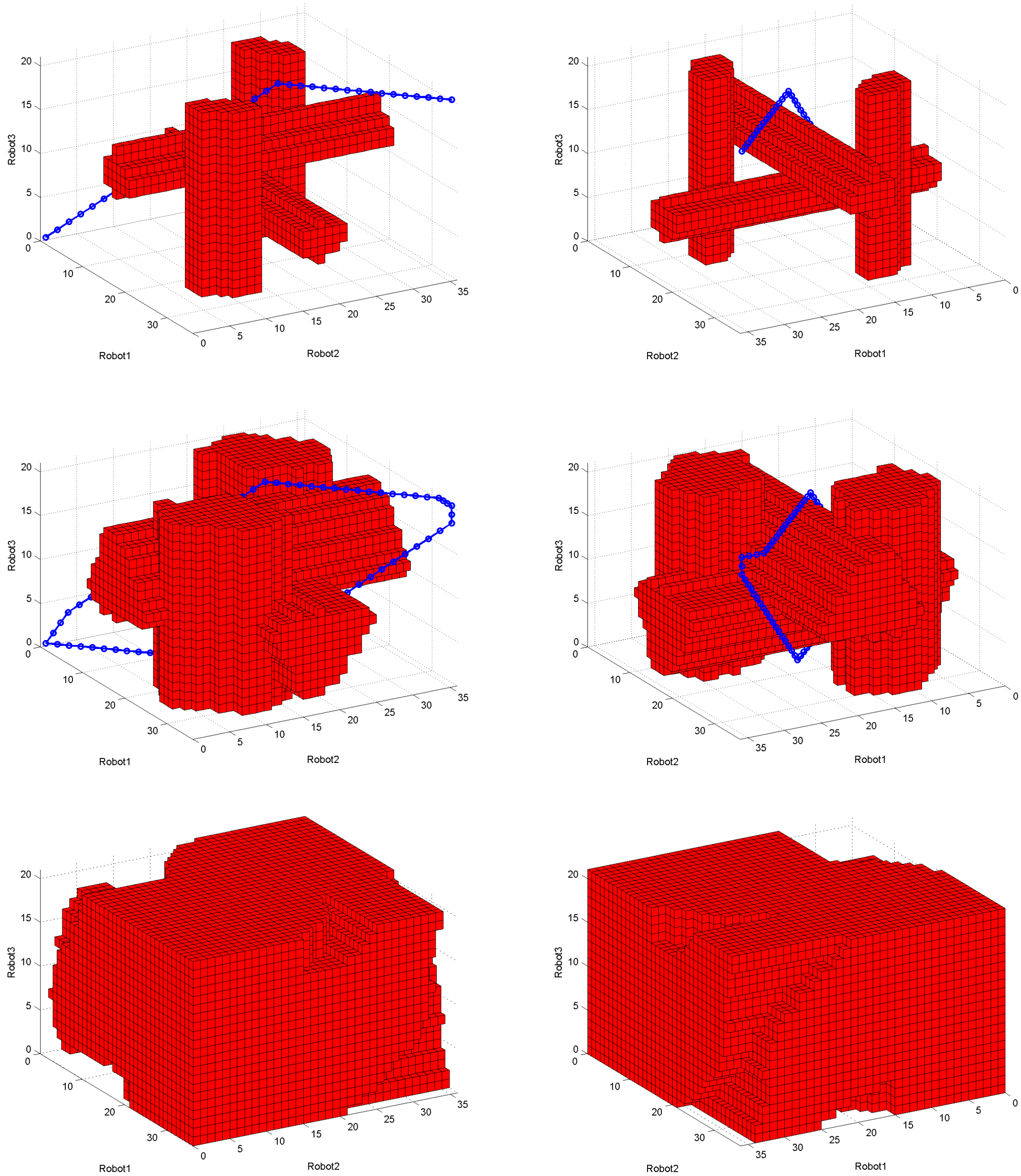
Obrázek 3.3: Konfigurace cest 3 robotů s periodou vzorkování $\Delta t = 10s$.



Obrázek 3.4: Ukázky scénářů dvou různých *strategií* γ^* 3 robotů o poloměrech $r_i = 20$ pro jednu konfiguraci cest, viz obr. 3.3.



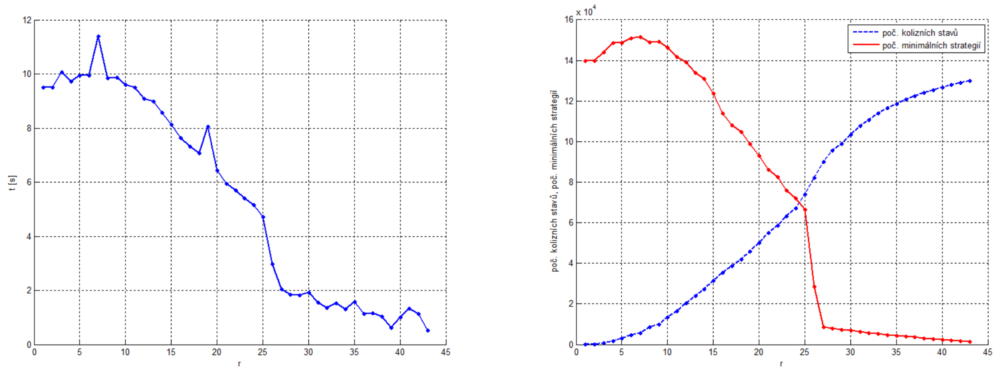
Obrázek 3.5: Dvě různé *strategie* γ^* 3 robotů v čase příslušných scénářů na obr. 3.4.



Obrázek 3.6: Ukázka tří různých *koordinačních prostorů* $\tilde{\mathcal{S}}$ daných různými poloměry robotů $r_i = \{10, 20, 40\}\, m$ seřazených pod sebou, patří stejné konfiguraci cest pro 3 roboty, viz obr. 3.3. Každá dvojice obrázků vedle sebe odpovídá stejnému *koordinačnímu prostoru* $\tilde{\mathcal{S}}$, který je zobrazen ze dvou různých úhlů pohledu. Jednotlivé krychle značí kolizní stavy $\tilde{s} \in \tilde{\mathcal{S}}_{coll}$, čáry z jednoho rohu do protějšího rohu 3D *koordinačního prostoru* zobrazené odlišnou barvou s vyznačenými body značí *stavovové trajektorie* α_γ. Perioda vzorkování je zde $\Delta t = 10s$.

$t_{total}[s]$	$t_{\tilde{\mathcal{S}}}[s]$	$t_{\forall M(\tilde{s})}[s]$	$K_{\tilde{\mathcal{S}}_{coll}}$	$K_{M(\tilde{s})}$	K_{γ^*}	$r[m]$
0.525	0.300	0.225	130189	1601	0	43
1.135	0.850	0.285	129158	1870	0	42
1.335	0.270	1.065	128055	2242	0	41
1.017	0.287	0.730	126908	2535	0	40
0.615	0.315	0.300	125487	2889	0	39
1.047	0.722	0.325	124187	3216	0	38
1.163	0.768	0.395	122400	3706	0	37
1.140	0.755	0.385	120895	3982	0	36
1.573	0.712	0.861	118703	4528	0	35
1.318	0.303	1.015	116591	4853	0	34
1.545	0.288	1.257	114185	5329	0	33
1.350	0.613	0.737	110612	5889	0	32
1.552	0.325	1.227	107735	6332	0	31
1.933	0.820	1.113	103685	7089	0	30
1.828	0.310	1.518	98938	7521	0	29
1.852	0.332	1.520	95662	8074	0	28
2.048	0.413	1.635	90230	8783	0	27
2.970	0.350	2.620	82356	28555	0	26
4.720	0.828	3.892	74107	66686	3	25
5.173	0.895	4.278	67135	72230	2	24
5.406	0.460	4.946	63160	75942	2	23
5.700	0.542	5.158	58657	82688	2	22
5.948	0.355	5.593	55109	86152	2	21
6.452	0.462	5.990	50227	93121	2	20
8.057	0.382	7.675	46047	98910	2	19
7.080	0.385	6.695	41914	104956	2	18
7.323	0.905	6.418	38834	108082	2	17
7.647	0.432	7.215	35467	114060	2	16
8.123	0.390	7.733	31540	123943	2	15
8.575	0.395	8.180	27433	130997	2	14
8.985	0.987	7.998	23937	133996	2	13
9.088	0.953	8.135	20400	139165	2	12
9.517	0.990	8.527	16400	141903	2	11
9.608	0.400	9.208	13592	146349	2	10
9.870	0.460	9.410	9972	149204	1	9
9.848	0.418	9.430	8580	149109	1	8
11.408	1.003	10.405	5672	151679	1	7
9.963	0.497	9.466	4704	151052	1	6
9.957	1.012	8.945	2972	148785	1	5
9.735	0.603	9.132	1916	148856	1	4
10.081	0.565	9.516	980	143975	1	3
9.515	0.425	9.090	272	139938	1	2
9.500	0.535	8.965	196	140016	1	1

Tabulka 3.2: Naměřené hodnoty bez počtu stavů *koordinačního prostoru* $K_{\tilde{\mathcal{S}}}$, který byl společný pro všechna měření a měl hodnotu 136800 stavů $\tilde{s}$ pro periodu vzorkování $\Delta t = 5s$.



Obrázek 3.7: Grafy závislostí celkových časů výpočtu koordinace t_{total} na poloměru robotů r (vlevo) a závislostí počtu kolizních stavů $K_{\tilde{S}_{coll}}$ a počtech *minimálních strategií* $K_{M(\tilde{s})}$ na poloměru robotů r (vpravo).

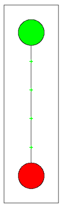
dvou časových stop odebíraných v době těsně před počátkem vytváření *koordinačního prostoru* včetně *minimálních strategií* a těsně po jejich ukončení. Při měření se opomněl fakt, že operační systém, na kterém byly experimenty měřeny, neprovádí své úlohy nepřetržitě a tím dochází k nepravidelným zpožděním, které jsou právě příčinou zmíněných špiček v grafu.

Dále v téže obr. 3.7 (vlevo) je vidět pokles času t_{total} v závislosti na poloměrech robotů r až do poloměru přibližně $r = 32m$, kde se pokles téměř zastavil. Stejně tak tomu je i co se týká počtu vytvořených minimálních strategií na obr. 3.7 (vpravo), kde před tímto poloměrem $r = 32m$ je jejich pokles strmý a dále pak s jeho růstem je změna minimální. V této fázi je množství kolizních stavů již tak velké, že počet nekolizních stavů v *koordinačním prostoru* okolo $\tilde{s}_{goal}$ je uzavřen a v důsledku toho algoritmus relativně brzy zjistí, že koordinace již není možná, tzn. že v nekolizních stavech všude jinde v *koordinačním prostoru* se již minimální strategie nevytvářejí, protože nemají žádné okolí stavu definované v kapitole 1.3.2.

Pokles doby výpočtu koordinace s rostoucími poloměry robotů je důsledkem klesání počtu nekolizních stavů a tím i omezení doby potřebné k vytvoření všech minimálních strategií v těchto stavech. V praxi analogicky platí, že čím víc má robot omezen prostor pro svůj pohyb, tím méně informací o prostoru má ke zpracování a v důsledku toho poklesne i doba výpočtu.

3.3 Vliv velikosti vzorkovací periody na koordinaci

Ukážeme vliv velikosti vzorkovací periody Δt na koordinaci robotů, resp. chyby, ke kterým může dojít při nesprávné volbě této periody vzorkování Δt.



Obrázek 3.8: Shodné cesty 2 robotů, které se překrývají a vedou vzájemně opačnými směry.

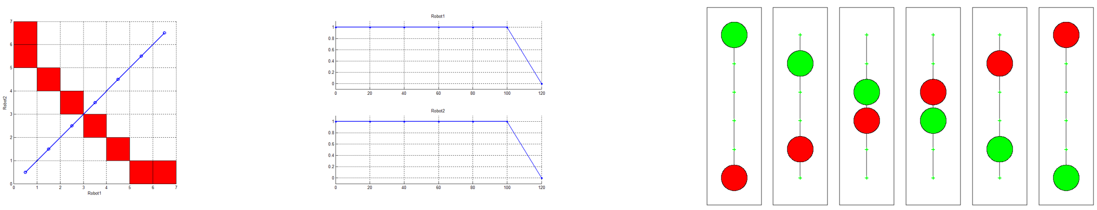
3.3.1 Realizace

Mějme 2 roboty mířící stejnou fixní cestou proti sobě s cílem vyměnit si navzájem své pozice, viz obr. 3.8. V praxi je zřejmé, že se musí zákonitě srazit, pokud by měly v úmyslu své úlohy splnit. Ukážeme si však, že algoritmus při zadání některých velikostí hodnot periody vzorkování Δt nalezne způsob koordinace tohoto v reálu neřešitelného problému a může tak způsobit chybu, která by v praxi vedla ke kolizi mezi roboty.

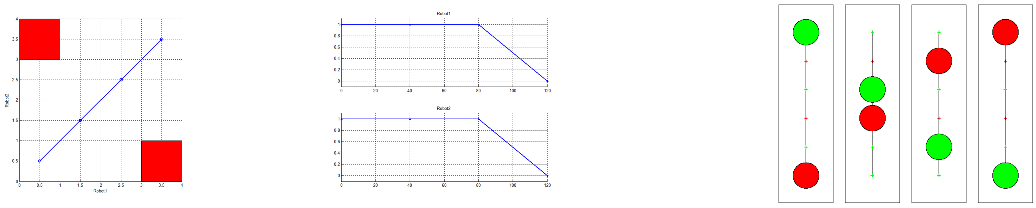
Volíme různé periody vzorkování Δt. Cesty τ^i obou robotů jsou stejně dlouhé a měří $length(\tau^i) = 100m$, poloměry robotů jsou $r_i = 9m$.

3.3.2 Výsledky

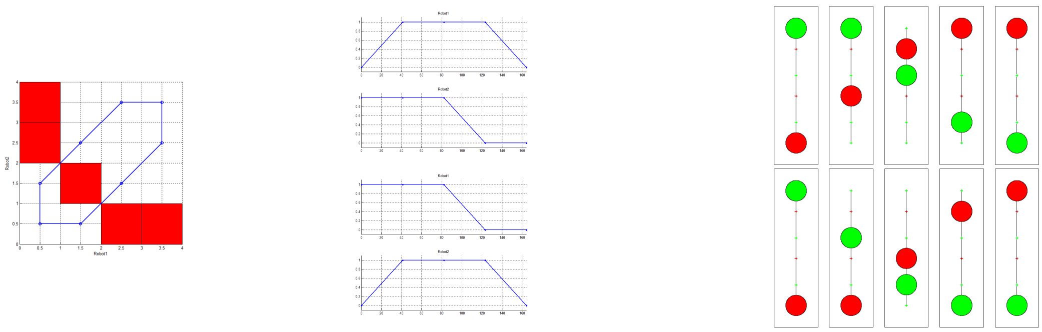
Měřením bylo zjištěno, že pro periodu vzorkování $\Delta t < 20s$ již výpočet koordinace nemá řešení, resp. nebyly nalezeny žádné *stavové trajektorie* α_{γ^*}, což je správně. Pro periodu vzorkování vyšší vycházelo, že koordinace je možná. To jsou právě hledané případy nesprávné koordinace. Příčinou je diskretizace koordinačního problému. Na obr. 3.12 jsou vidět ukázky příslušného *koordinačního prostoru* s různou periodou vzorkování Δt, která se zvyšuje s jednotlivými obrázky směrem doprava. Je vidět, že pro *koordinační prostor* na prvním obrázku nebyly nalezeny žádné *stavové trajektorie* α_{γ^*} díky vysoké hustotě kolizních stavů $\tilde{s} \in \tilde{\mathcal{S}}_{coll}$. Pro ostatní již nějaké α_{γ^*} nalezeny byly. Ukázky příslušných nalezených *strategií* γ^* v čase jsou k uvedeným *koordinačním prostorům* zobrazeny na obr. 3.9, 3.10 a 3.11 včetně sekvencí postupů obou robotů v čase. Z těchto sekvencí je zřejmé, jak se roboty minou vzájemným přeskočením, či okamžitou výměnou pozic, což, jak již bylo zmíněno, je nesmysl.



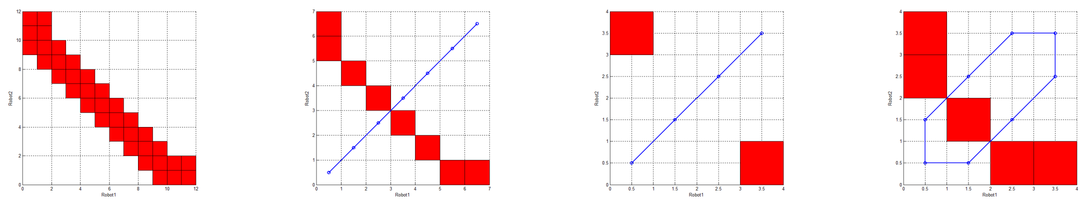
Obrázek 3.9: Situace 2 robotů s periodou vzorkování $\Delta t = 20s$. *Koordinační prostor* s 1 nalezenou *strategií* γ^* (vlevo), *strategie* γ^* (uprostřed), sekvence postupu 2 robotů v čase (vpravo) pro nalezenou *strategii* γ^*.



Obrázek 3.10: Situace 2 robotů s periodou vzorkování $\Delta t = 40s$. *Koordinační prostor* s 1 nalezenou *strategií* γ^* (vlevo), *strategie* γ^* (uprostřed), sekvence postupu 2 robotů v čase (vpravo) pro nalezenou *strategii* γ^*.



Obrázek 3.11: Situace 2 robotů s periodou vzorkování $\Delta t = 41s$. *Koordinační prostor* se 2 nalezenými *strategiemi* γ^* (vlevo), *strategie* γ^* (uprostřed), sekvence postupu 2 robotů v čase (vpravo) pro dvě různé *strategie* γ^*.



Obrázek 3.12: Různé *koordinační prostory* robotů konfigurace cest popsané v kapitole 3.3, které se liší periodou vzorkování Δt, která je zleva doprava $10s$, $20s$, $40s$, $41s$.

3.3.3 Závěr

Problém v této kapitole popsaný se analogicky týká i jiných situací, než té, která je zde popsána. Aby se zabránilo nezjistitelným kolizím, je vhodné zvolit periodu vzorkování Δt co nejnižší možnou s respektováním délky doby trvání výpočtu koordinace t_{total}, která s klesající periodou vzorkování Δt naopak stoupá.

3.4 Shrnutí

Experimenty ve všech případech proběhly dle očekávání. Algoritmus byl naimplementován správně. Změřené parametry mohou posloužit k eventuelnímu rozhodování, zda program s koordinačním algoritmem, tak jak byl implementován, je vhodný k nějakému dalšímu použití řešení koordinačních problémů.

Kapitola 4

Experimenty koordinace robotů s nezávislými mapami cest

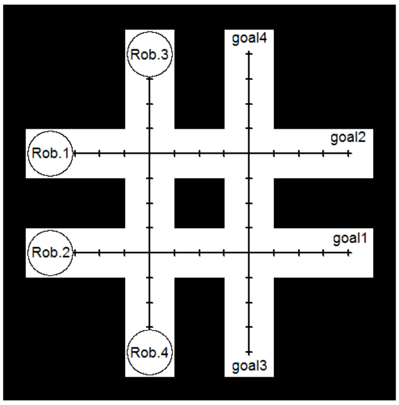
V této kapitole se ověřuje funkčnost implementace algoritmu koordinace robotů s nezávislými mapami cest, jehož popis je uveden v kapitole 2.

Algoritmus je opět implementován v Javě a počítán stejným systémem, jak je popsáno v kapitole 3. Prezentace výsledků provádění jednotlivých testů v téže kapitole není podstatná, jelikož implementace algoritmu koordinace robotů s nezávislými mapami cest vychází právě z implementace algoritmu popsané v kapitole 3 a zároveň je jejím rozšířením. Ověřuje se pouze schopnost algoritmu vybrat optimální cestu z mapy cest každého robota a nepoužívat pouze cestu jednu jedinou.

V následujících podkapitolách jsou zobrazeny ukázky jednoho příkladu map cest 4 robotů nejprve obecně pro všechny nalezené strategie v kapitole 4.1 a pak v kapitole 4.2 pomocí sekvencí obrázků z programu Player/Stage.

4.1 Příklad mapy cest 4 robotů

Mapa cest je zobrazena na obr. 4.1, kde je vidět její konfigurace 4 robotů umístěných na počátečních pozicích a jejich cíle na opačných koncích mapy. Na obr. 4.2 jsou vidět 4 různé strategie zobrazené na každém řádku zvlášť, které jsou výsledkem příslušného algoritmu pro danou konfiguraci map cest robotů. Tyto strategie jsou prezentovány různými cestami jednotlivých robotů včetně posloupností příkazů v čase. Je zřejmé, že řádky 1,3 a řádky 2,4 mají shodně zobrazené scénáře, které se však liší posloupností příkazů v čase. Je to výsledkem toho, že pro každou strategii na řádcích byl upřednostněn vždy jeden robot (ve stejném pořadí jak jdou řádky od shora), aby se dostal do cíle první.



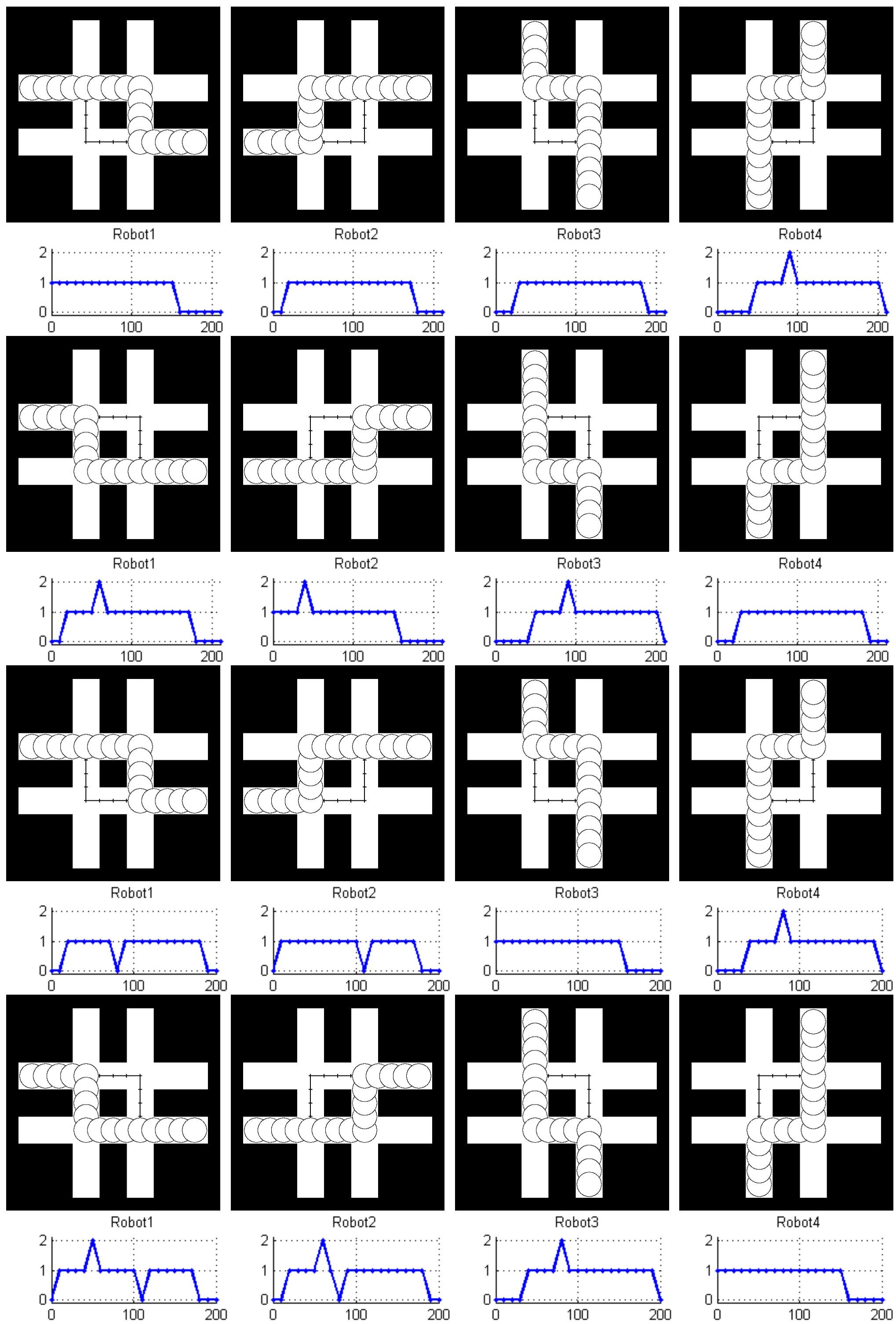
Obrázek 4.1: Mapa cest 4 robotů, kde jsou též vyznačeny jejich cíle.

4.2 Player/Stage

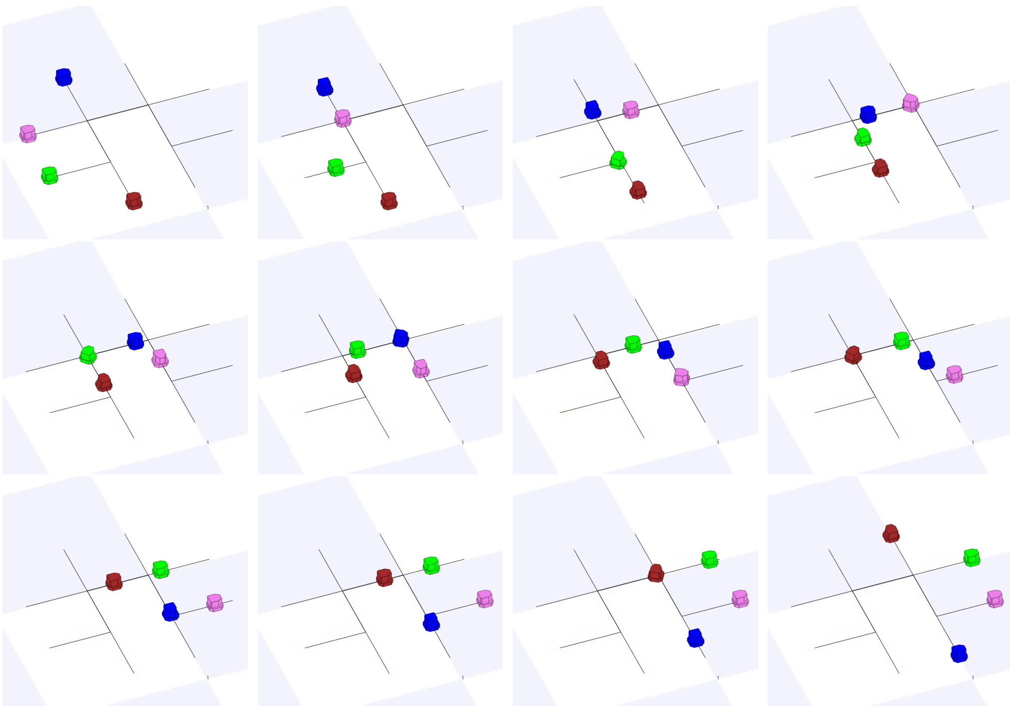
Algoritmus byl testován i na systému Player/Stage, který pracuje na principu server/-klient. Pomocí klienta se serveru Player posílají jednotlivé konfigurace robotů v čase a roboty simulované serverem Player se pohybují pomocí pozicového regulátoru k souřadnicím reprezentovanými těmito konfiguracemi. Zobrazení celé situace je zprostředkováno simulátorem 2D prostředí Stage. Narozdíl od algoritmu, popsaného v kapitole 2, se roboty chovají dle fyzikálních zákonů dynamiky, tzn. že jejich pohyb není skokový, ale plynulý včetně jejich rozjezdu i zastavení.

Výsledkem byly nesouvislé pohyby robotů, které způsobilo čekání ostatních robotů na roboty, kteří se ještě v daném čase nedostavili na pozici, na kterou jim zaslaná konfigurace ukazovala. Příčinou bylo, že robotu, který se měl přesunout mezi konfiguracemi přes zatáčku na cestě trval pohyb déle, než robotům, které se nacházely na přímé cestě, jelikož se musel v zatáčce otočit požadovaným směrem, což v praxi znamená zdržení s nímž algoritmus koordinace nepočítal. Řešením bylo navzorkovat všechny cesty jemněji kratší periodou vzorkování.

Na obr. 4.3 je zobrazen příklad vycházející z kapitoly 4.1, který ukazuje příklad první strategie upřednostňující robot 1, viz obr. 4.1 v sekvencích obrázků tak jak se roboty pohybovaly v čase.



Obrázek 4.2: Scénáře jednotlivých robotů včetně jejich posloupností příkazů (strategií) v čase, kde každý řádek odpovídá jedné strategii a každý sloupec odpovídá jednomu robotu.



Obrázek 4.3: Sekvence scénáře všech robotů pro jednu strategii exportované v obrázcích z programu Player/Stage.

Kapitola 5

Závěr

V této bakalářské práci jsme se zaměřili na teorii koordinace více mobilních robotů v prostoru, kterou jsme si rozdělili na řešení dvou úzce spojených problémů. První z nich je algoritmus koordinace robotů s fixními cestami, kterou jsme popsali v kapitole 1. Ta se zabývá koordinací robotů pomocí jejich schopností pohybu a zastavení, kde v případě možné kolize jeden robot zastaví a počká až se mu uvolní cesta, která byla blokována jiným robotem. Druhý problém se týkal algoritmu koordinace robotů s mapami cest, kde si roboty na rozdíl od prvního problému mohly navíc vybrat volbu změny cesty v místech rozcestí a vyhnout se tak řešení koordinace zastavením a počkáním jako u prvního problému. Teorie k této části je obsažena v kapitole 2.

Oba algorimy byly implementovány v programovacím jazyce Java v pořadí uvedeném v předchozím odstavci, což zabíralo nejvíce času práce na této bakalářské práci. Ty jsme následně experimentálně ověřili testy a simulacemi popsanými v kapitolách 3 a 4. Bylo ukázáno, že algorimy fungují správně dle očekávání, které vycházelo z popsané teorie. Ve výsledku při řešení obou zmíněných problémů stačí používat pouze druhý algoritmus koordinace robotů s mapami cest, který ve všech ohledech umí zastoupit první implementovaný algoritmus koordinace robotů s fixními cestami, jehož je rozšířením.

Použití implementovaných algoritmů v praxi tak jak jsou vytvořeny však není vhodné vzhledem k nerespektování fyzikálních vlastností robotů. Nicméně byly odzkoušeny na simulacích systému Player/Stage, který částečně nahrazuje reálné chování robotů, kde jejich pohyb sice nebyl zcela plynulý, ale po provedených úpravách popsaných v kapitole 4.2, se plynulost pohybu robotů částečně zlepšila.

Bakalářská práce by mohla pokračovat úpravou algoritmů koordinace tak, aby byly zohledněny zmíněné fyzikální zákony, kterými jsou roboty ovlivněny. Měly by brát v potaz, že robot se nepohybuje konstantní rychlostí a jeho uvedení do pohybu či zastavení je plynulé nikoliv skokové. Dále by měly být co nejméně časově náročné, aby dokázaly provádět koordinační výpočty v reálném čase. Algoritmy by se pak vyzkoušely na reálných modelech robotů v laboratoři, kde by se ověřila jejich schopnost řízení v praxi a provedly se eventuelní korekce.

Práce na této bakalářské práci byla pro mne zajímavým poznáním rozsáhlé problematiky robotiky alespoň prostřednictvím její malé části, která mi otevřela vnímání robotiky z pohledu programátora.

Literatura

[1] Oracle, Sun Developer Network (SDN), http://java.sun.com/. [online], Naposledy navštíveno 24. 5. 2010.

[2] Wikipedia, Breadth First Search, http://en.wikipedia.org/wiki/Breadth-first_search. [online], Naposledy navštíveno 24. 5. 2010.

[3] S. M. LaValle; S. A. Hutchinson. *Optimal Motion Planning for Multiple Robots Having Independent Goals.* IEEE Trans. on Robotics and Automation, 1998.

[4] S. M. LaValle. *Planning Algorithms.* Cambridge University Press, Cambridge, U.K., 2006.

[5] P. E. Hart; N. J. Nilsson; B. Raphael. *A Formal Basis for the Heuristic Determination of Minimum Cost Paths.* IEEE Trans. on Systems Science and Cybernetics, 1968.

Obsah CD

Přiložené CD obsahuje zdrojové kódy pro algoritmy koordinace robotů, text diplomové práce ve formátu PDF a zdrojové kódy celého textu pro systém LaTeX. V následující tabulce je popsána struktura CD.

Adresář	Popis
`src`	zdrojové kódy knihovny
`doc`	zdrojové kódy textu diplomové práce
`thesis.pdf`	text diplomové práce

Tabulka 1: Adresářová struktura na CD