VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ

FACULTY OF MECHANICAL ENGINEERING

ÚSTAV MECHANIKY TĚLES, MECHATRONIKY A BIOMECHANIKY

INSTITUTE OF SOLID MECHANICS, MECHATRONICS AND BIOMECHANICS

OPTIMALIZACE SKLÁDÁNÍ KRABIC NA PŘEPRAVNÍ PALETU

OPTIMIZATION STACKING BOXES ON A PALLET

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE

AUTHOR

BSc Jiří Havlíček

VEDOUCÍ PRÁCE

SUPERVISOR

Ing. Roman Adámek

BRNO 2024

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA STROJNÍHO INŽENÝRSTVÍ

Zadání diplomové práce

Ústav:	Ústav mechaniky těles, mechatroniky a biomechaniky
Student:	**BSc Jiří Havlíček**
Studijní program:	Mechatronika
Studijní obor:	bez specializace
Vedoucí práce:	**Ing. Roman Adámek**
Akademický rok:	2023/24

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma diplomové práce:

Optimalizace skládání krabic na přepravní paletu

Stručná charakteristika problematiky úkolu:

Problematika optimalizace skládání krabic na přepravní paletu je častý problém v oblasti logistiky a v době rozmachu různých přepravních a doručovacích služeb je o to aktuálnější. Diplomová práce se bude zabývat variantou tohoto problému, kdy se budou skládat na paletu krabice s náhodně generovanými rozměry, které budou uloženy v zásobníku. V každém kroku bude možné vybrat z N krabic, které jsou v zásobníku k dispozici. Práce se bude zabývat návrhem samotného algoritmu pro ukládání krabic na paletu a jeho ověřením v simulaci.

Cíle diplomové práce:

1) Proveďte literární rešerši současného stavu poznání v oblasti optimalizačních algoritmů pro skládání krabic na přepravní paletu.

2) Na základě rešerše zvolte nejperspektivnější přístup a ten modifikujte dle specifikace problému v zadání práce. Navrhněte různá kritéria optimalizace skládání s ohledem na zaplněnost palety, její stabilitu atd.

3) Proveďte simulační ověření algoritmu v robotickém simulátoru s ohledem na jeho použitelnost a splnění jednotlivých kritérií, včetně simulace pohybu robotického manipulátoru pro přemísťování krabic.

Seznam doporučené literatury:

[1] SILVA, Elsa, José F. OLIVEIRA a Gerhard WÄSCHER. The pallet loading problem: a review of solution methods and computational experiments. International Transactions in Operational Research [online]. 2016, 23(1-2), 147-172. ISSN 0969-6016. Dostupné z: doi:10.1111/itor.12099

[2] SINGH, Manjeet, Najat ALMASARWAH a Gürsel SÜER. A Two-Phase Algorithm to Solve a 3-Dimensional Pallet Loading Problem. Procedia Manufacturing [online]. 2019, 39, 1474-1481. ISSN 23519789. Dostupné z: doi:10.1016/j.promfg.2020.01.301

[3] SCHUSTER, M, R BORMANN, D STEIDL, S REYNOLDS-HAERTLE a M STILMAN. Stable stacking for the distributor's pallet packing problem. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems [online]. IEEE, 2010, s. 3646-3651. ISBN 978-1-4244-6674-0. Dostupné z: doi:10.1109/IROS.2010.5650217

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2023/24

V Brně, dne

L. S.

prof. Ing. Jindřich Petruška, CSc. ředitel ústavu	doc. Ing. Jiří Hlinka, Ph.D. děkan fakulty

Abstrakt

Tato práce se zaměřuje na Problém skládání palety distributora (DPLP), který je považován za jednu z nejsložitějších úloh optimalizace vyplnění omezeného prostoru. DPLP představuje výzvu z důvodu dynamické povahy a různorodosti rozměrů a hmotností krabic, a zahrnuje požadavky na stabilitu, maximální nosnost a omezené rotace. Je navržena strategie pro výběr pozice, orientace a pořadí dostupných krabic za předpokladu, že je známe jen několik krabic. V rámci práce je navržen algoritmus prohledávání stavového prostoru, implementovaný v MATLABu, který je následně ověřen na základě experimentů. Dále je provedena simulace stability palety pomocí SimScape Multibody a plánování bezkolizní trajektorie pro robotický manipulátor použitím Robotic toolboxu.

Summary

This thesis focuses on the Distributors Pallet Loading Problem (DPLP), considered one of the most challenging problems in optimizing limited space filling. DPLP poses a challenge due to its dynamic nature and the variety of dimensions and weights loaded boxes, including requirements for stability, maximum load capacity, and limited rotations. A strategy is proposed for selecting the position, orientation, and sequence of available boxes under the assumption that only a few boxes are known. Within the scope of the work, a state space search algorithm is designed, implemented in MATLAB, and subsequently validated through experiments. Furthermore, a simulation of pallet stability is conducted using SimScape Multibody and a collision-free trajectory planning for a robotic manipulator is performed using the Robotic Toolbox.

Klíčová slova

Skládání krabic na paletu, Problém skládání palety distributora, DPLP, Optimalizace prostoru, Prohledávání stavového prostoru, MATLAB, Simscape Multibody, Bezkolizní trajektorie, Omezený zásobník

Keywords

Loading boxes on pallet, Distributors pallet loading problem, DPLP, Space optimization, State space search, MATLAB, Simscape Multibody, Collision-free trajectory, Limited Stack

Bibliografická Citace

HAVLÍčEK, J. *Optimalizace skládání krabic na přepravní paletu.* Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2024. 72 s., Vedoucí diplomové práce: Ing. Roman Adámek.

Čestné prohlášení

Prohlašuji, že jsem diplomovou práci zpracoval samostatně, pod vedením Ing. Romana Adámka a že jsem uvedl všechny použité informační zdroje.

Jiří Havlíček

Brno . . . .

Poděkování
Rád bych zde poděkoval všem, kteří mi byli oporou a pomáhali při tvorbě této diplomové práce. Zvláště pak patří díky vedoucímu této práce Ing. Romanu Adámkovi za jeho rady a věcné připomínky.

Jiří Havlíček

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

Efektivní skládání krabic na paletu je klíčovým aspektem pro logistiku a distribuci zboží. Vhodná strategie při uspořádání krabic může zásadně ovlivnit náklady na přepravu, efektivitu skladování a bezpečnost manipulace. Správné skladování nejen maximalizuje využití prostoru, ale také snižuje riziko poškození zboží během přepravy.

Předmětem práce je Problém skládání palety distibutora (Distributors pallet loading problem, DPLP). DPLP je považována za jednu z nejsložitějších problémů z rodiny optimalizačních úloh zabývající se vyplněním omezeného prostoru objekty, kvůli své vysoce dynamické povaze. Předpokládá krabice odlišných rozměrů a hmotností ve variabilním pořadí. Může zahrnovat také požadavky na stabilitu, maximální nosnost, omezené rotace a flexibilitu objednávky.

Zaměříme se na situaci, kdy je předem znám velmi omezený počet krabic, jejichž celkový objem tvoří jen malou část poskládané palety. Tento přístup je relevantní například pro distributory malé elektroniky nebo pro velké automatizované sklady, které musí dynamicky reagovat na měnící se poptávku a skladovou kapacitu, případně při přepravě smíšeného nákladu, kde se složení zásilky může průběžně upravovat podle dostupnosti. Hlavním cílem práce je tedy navrhnout strategii pro výběr pozice, orientace a pořadí dostupných krabic.

Hlavní výzvou při hledání možného řešení DPLP je jeho výpočetní náročnost. Spadá do kategorie NP-úplných problémů (Non-deterministic Polynomial time). To znamená, že zatímco řešení lze ověřit v polynomiálním čase, samotné vyřešení problému v tomto čase není zaručeno. Je proto nezbytné zavést pravidla, která umožní zjednodušení problému a kritéria pro urychlení procesu hledání řešení.

Uvažuje se skládání v 3D prostoru, kdy je znám zásobník N krabic s rozměry $[l, w, h]$ a hmotností m. V rámci této práce se zásobník po každém umístění průběžně doplňuje, přičemž nová krabice je buď náhodně generována, nebo pochází z existujícího datasetu. Každá krabice se předpokládá za ideální kvádr s homogenní distribucí hmotnosti. Nejsou uvažovány křehké krabice a jsou povolené pouze rotace 90°, čímž je omezen nejvyšší počet možných orientací. Paleta je definována rozměry $[L, W, H]$ a maximální nosností M.

V následující části práce jsou nejdříve popsány dřívější poznatky týkající se paletizační problematiky, přičemž je vysvětleno zařazení DPLP mezi podobné optimalizační úlohy a jsou uvedeny různé metody postupu řešení. Poté je podrobně představen návrh algoritmu pro řešení této instance DPLP a jeho implementace v programovacím prostředí MATLAB a jsou prezentovány výsledky pro vybrané datasety.

Další část se zabývá simulací, jejímž hlavním účelem je ověření stability poskládané palety pomocí SimScape Multibody a je v ní shrnuto i plánování bezkolizní trajektorie robotického manipulátoru při přemisťování krabic na paletu.

2 Rešerše

Problém skládání krabic na paletu je specifickým případem obecného problému skládání, který se objevuje v mnoha různých odvětvích průmyslu. Tento problém spočívá v nalezení optimálního způsobu uspořádání menších objektů do většího prostoru tak, aby se minimalizoval nevyužitý prostor. Optimalizace využití prostoru vede k výrazným úsporám nákladů, zejména v kontextu masové produkce, kde i drobné vylepšení může mít velký dopad na celkové náklady.

Různá odvětví průmyslu se setkávají s odlišnými verzemi tohoto problému. V dřevařském, sklářském a papírenském průmyslu jde o problém řezání objektů jednotného tvaru. Oděvní a kožedělný průmysl se naopak potýká s řezáním objektů různých tvarů. Doprava zahrnuje oba tyto typy problémů a navíc může čelit mnoha dalším omezujícím podmínkám, jako jsou hmotnostní limity, stabilita nákladu a další logistické faktory. [1]

S rozvojem počítačových technologií v 60. a 70. letech 20. století začaly firmy hledat způsoby, jak optimalizovat skládání krabic za účelem maximalizace využití prostoru a minimalizace finančních prostředků. První počítačové algoritmy pro paletizaci byly vyvinuty právě v této době a představovaly významný krok vpřed. [2] Tyto rané algoritmy byly poměrně jednoduché a dokázaly řešit základní problémy spojené se skládáním krabic. S rostoucí výpočetní kapacitou počítačů a pokroky v algoritmických technikách se však začaly zdokonalovat.

V 80. a 90. letech 20. století došlo k dalšímu významnému posunu díky rozvoji pokročilých algoritmů a softwarových nástrojů. Optimalizační metody jako dynamické programování, genetické algoritmy a heuristické metody umožnily řešení složitějších problémů skládání krabic na paletu.

V posledních desetiletích se s nástupem pokročilých výpočetních technologií a umělé inteligence otevřely nové možnosti pro optimalizaci skládání krabic. Moderní algoritmy využívají strojové učení a simulace k dosažení ještě efektivnějších řešení.

Dnes je paletizace klíčovou součástí moderní logistiky, a to jak v tradičních průmyslových odvětvích, tak v e-commerce a dalších oblastech.

V následujícím textu budou představena hlavní teoretická východiska práce. Nejprve bude vysvětlena kategorizace problematiky, kde budou zmíněny obecné problémy související s danou tématikou. V další podkapitole bude přiblížen samotný Problém skládání na paletu (PLP) a jeho podkategorie, se zaměřením na DPLP. Dále budou představena některá z možných řešení pro různé instance problému společně s relevantními dílčími algoritmy.

2.1 Kategorizace problematiky

Problém batohu (Knapsack problem, KP)
KP je jedním z klasických optimalizačních problémů v oblasti teorie výpočetní složitosti a kombinatoriky, s prvními zmínkami již z konce 19. století [3]. Smyslem tohoto problému je najít takovou kombinaci objektů z dané množiny, aby byl maximalizován celkový zisk, aniž by byla překročena kapacita omezeného prostoru, typicky představovaná batohem [4].

Problém batohu úzce souvisí se skládáním krabic na paletu, protože oba problémy sdílejí základní myšlenku efektivního využití omezeného prostoru. Podobně jako v případě problému batohu, kde je žádoucí nalézt nejlepší kombinaci objektů pro maximalizaci zisku, je i v problému paletizace důležité nalézt optimální rozmístění krabic tak, aby byl co nejlépe využit dostupný prostor.

Bin Packing Problem (BPP)
Dalším významnou úlohou je BPP. Podstatou tohoto problému je umístit daný počet objektů do co nejmenšího počtu kontejnerů. Tyto kontejnery mohou být stejné nebo různé velikosti, a problém může být řešen v různých dimenzích(1D, 2D i 3DBPP). BPP se často vyskytuje v průmyslových odvětvích, jako je dřevozpracující, sklářský a textilní průmysl, kde je klíčové optimalizovat prostor při balení a skladování [1, 5]

Na rozdíl od KP se BPP zaměřuje na minimalizaci počtu použitých kontejnerů. Tento rozdíl má praktické aplikace například při balení výrobků do co nejmenšího počtu krabic nebo při optimalizaci prostoru ve skladových regálech. V průmyslovém kontextu může efektivní řešení BPP vést k významným úsporám nákladů a zlepšení logistiky.

Container Loading Problem (CLP)
CLP je optimalizační problém spojený s umístěním objektů do kvádrového prostoru daných rozměrů s cílem maximalizovat jeho využití.

Existuje několik variant CLP, uvažující například jen homogenní krabice (všechny krabice stejných rozměrů), slabě heterogenní (několik různých typů krabic) nebo silně heterogenní (všechny krabice různých rozměrů). Řešení CLP často zahrnuje použití různých algoritmů a metod optimalizace, včetně heuristik, metaheuristik a algoritmů řešení lineárního programování. [6, 7, 8]

Pallet Loading Problem (PLP)
PLP sdílí mnoho podobností s CLP, avšak zaměřuje se specificky na umístění krabic na paletu. Často bere v úvahu specifická kritéria jako stabilita nákladu, bezpečnost přepravy, nosnost palety a efektivní skladování. Tyto faktory jsou klíčové pro zajištění bezpečného a efektivního přepravního procesu, což má významný vliv na celkovou logistickou strategii a využití finančních prostředků [1].

Z hlediska uvažovaného nákladu se PLP dělí do dvou hlavních kategorií:

- PLP výrobce (Manufacturers PLP, MPLP):
  MPLP se zaměřuje na efektivní uspořádání krabic stejných rozměrů na paletě, což umožňuje jednodušší algoritmické řešení a minimalizuje potřebu složitých optimalizačních postupů [9].

- PLP distributora (Distributors PLP, DPLP)

2.2 DPLP

DPLP zahrnuje komplexní úlohy, které se zaměřují na umisťování krabic různých velikostí na paletu. Tento problém je považován za složitější než MPLP kvůli různorodosti krabic, které je třeba efektivně uspořádat. Odlišné rozměry krabic přidávají další úroveň složitosti, což vyžaduje pokročilejší algoritmické a optimalizační metody. Je relevantní zejména pro distribuční centra, kde je nutné zohlednit různorodost zboží a zároveň maximalizovat využití palet a minimalizovat náklady na přepravu [10].

2.2.1 Možné strategie paletizace

Řešení PLP je silně závislé na typu úlohy, se kterou se pracuje. Kvůli vysoké výpočetní náročnosti se často přistupuje ke zjednodušení problému. To může zahrnovat například uvažování pouze některých orientací krabic, což redukuje počet možných uspořádání. Další zjednodušení může spočívat v uvažování křehkých krabic, na jejichž horní stěnu nelze žádnou další krabici položit[11].

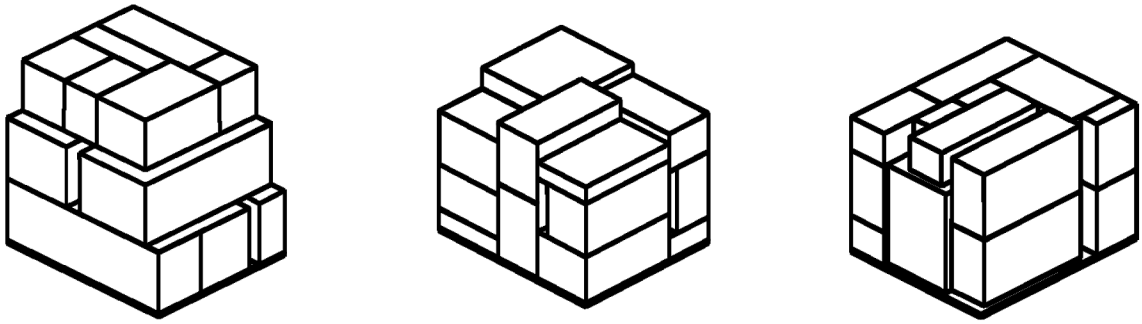
Způsob, jakým jsou krabice umisťovány hraje velmi důležitou roli pro dosažení odlišných cílů paletizace. Je možné rozlišit tři základní strategie pokládání (Obr. 2.1):

Metoda vrstvení
Krabice jsou naloženy po vrstvách, přičemž každá vrstva je tvořena krabicemi stejné výšky, případně se najdou kombinace krabic které společné výšky dosáhne (Obr. 2.1a). Tento proces se opakuje, dokud není dosaženo požadované výšky palety.

Metoda sloupcového stohování
Paleta je skládána do různého počtu sloupců, přičemž každý sloupec je tvořen vertikálně naskládanými krabicemi a výška sloupců nepřesahuje maximální povolenou výšku palety (Obr. 2.1b).

Metoda náhodného stohování
Krabice jakékoli velikosti mohou být naloženy kdekoli na paletě bez konkrétního vzoru (Obr. 2.1c). Tato metoda může dosáhnout lepší konfigurace palety z pohledu její stability.

Vzhledem k tomu, že pro danou úlohu je předem známo pouze několik krabic náhodných rozměrů, nejsou první dvě metody vhodné k použití. V této práci se proto budeme zabývat náhodným stohováním do předem nalezených pozic.



(a) Vrstvení (b) Sloupcové stohování (c) Náhodné stohování

Obrázek 2.1: Ilustrace jednotlivých metod skládání [12]

Práce s datasety je dalším klíčovým aspektem řešení DPLP. Seřazování krabic před a během algoritmu může významně zjednodušit problém a zlepšit efektivitu a kvalitu výsledného uspořádání. Existuje několik běžně používaných metod a strategií seřazování [13].

Seřazení podle **rozměrů** (délka, šířka, výška) může být vzestupné nebo sestupné, což pomáhá lépe využít dostupný prostor a minimalizovat mezery mezi krabicemi. Seřazení podle **objemu** umožňuje nejprve umístit největší krabice a poté doplnit menšími, což usnadňuje stabilní stohování [14]. Těžší krabice se obvykle umisťují do spodních vrstev, aby se předešlo převrácení palety a zajistila se stabilita celého nákladu. Některé krabice mohou mít i vyšší **prioritu** z hlediska důležitosti nebo naléhavosti přepravy, a proto se umisťují na paletu jako první, aby bylo zajištěno jejich včasné doručení [15, 16].

Jednou z pokročilejší metod řešení je tzv. **Layer Based Column Generation Algorithm (LCGA)**. LCGA funguje tak, že nejprve rozdělí úlohu na menší a jednodušší podúlohy. Každá podúloha představuje vrstvu krabic na paletě. Poté se pro každou vrstvu určí optimální uspořádání krabic.

Proces začíná vytvořením počátečního řešení, které může být suboptimální. Následně se postupně generují nové potenciální vrstvy, přičemž se používají různé metody, aby se zlepšila kvalita řešení. Tyto vrstvy jsou přidávány nebo nahrazovány ve stávajícím uspořádání, pokud vedou k lepšímu využití prostoru nebo vyšší stabilitě nákladu.

Každá nová vrstva se hodnotí podle určitého kritéria, například podle toho, jak dobře využívá dostupný prostor nebo jak přispívá ke stabilitě palety jako celku. Pokud nová vrstva zlepšuje celkové řešení, je začleněna do výsledného uspořádání. Tento iterativní proces pokračuje, dokud se nedosáhne optimálního nebo dostatečně dobrého řešení [17].

Heuristické metody jsou v problematice DPLP velmi rozšířené, neboť poskytují rychlá a často dostatečně kvalitní řešení pro složité úlohy. Tyto metody se liší od exaktních algoritmů tím, že neposkytují záruku nalezení optimálního řešení, ale namísto toho se zaměřují na hledání dostatečně dobrého řešení v co nejkratším čase. [18]

Jedním z příkladů heuristických metod je **Greedy algoritmus**, který pracuje na principu lokální optimalizace. Greedy algoritmus se snaží v každém kroku vybrat nejlepší dostupnou volbu, aniž by bral v úvahu celkovou strukturu problému. Tímto způsobem postupně pokračuje a snaží se dosáhnout co nejlepšího řešení.

Dalším příkladem je **Simulované žíhání** (Simulated Annealing, SA), které je inspirováno procesem chladnutí kovu. Tato metoda umožňuje průzkum prostoru řešení pomocí náhodných změn a postupně se zmenšující pravděpodobností uvažování horších řešení.

Tímto způsobem se algoritmus snaží uniknout lokálním optimům a dosáhnout globálního extrému. [19]

Algoritmus řezů je dalším přístupem, který je avšak využíván zejména v případech, kde je známá množina všech objektů pro umístění. Tento algoritmus rozděluje paletu na menší části (řezy) a hledá optimální uspořádání v těchto částech. Dalšími příklady heuristických metod jsou **Best-fit** a **First-fit decreasing search**, **Nested beam search** [20] nebo **Tabu search** [21].

Každý z těchto přístupů má své výhody a nevýhody, a volba konkrétní metody často závisí na specifických požadavcích úlohy a dostupných výpočetních zdrojích. Mnoho z uvedených metod bere v úvahu znalost celého souboru objektů k položení a jejich použití v této práci tedy není možné. Naopak metody založené na prohledávání a postupném vybírání pořadí krabic se jeví jako vhodní kandidáti pro použití pro instanci DPLP této práce.

Ve většině prací je klíčovým aspektem hledání pozic pro umístění krabic a jejich ohodnocení. To zahrnuje nejen identifikace volných prostor na paletě, ale také určení, které krabice jsou nejvhodnější pro dané pozice z hlediska stability, efektivity využití prostoru, případně dalších logistických požadavků.

2.2.2 Hledání možných umístění

Při řešení úloh DPLP je způsob identifikace možných pozic naprosto klíčový. Metody používané v literatuře zohledňují faktory, jako je dostupný prostor, tvar a rozměry krabic. Při jejich návrhu je zároveň kladen velký důraz na co nejnižší výpočetní náročnost.

Pro reprezentaci umístění bývá nejčastěji uvažován jeden z rohů krabice z několika důležitých důvodů. Použití rohů výrazně zjednodušuje výpočty, neboť jeho pozice je přímo určena souřadnicemi pozic. Naproti tomu použitím těžiště by byly nutné další kroky k určení středu krabice a následné přepočítání polohy vzhledem k ostatním objektům.

Práce s rohy také umožňuje efektivněji uspořádat krabice tak, aby mezi nimi zůstávalo co nejméně volného místa. Pokud bychom používali těžiště, mohlo by dojít k méně efektivnímu využití prostoru, jelikož by bylo obtížnější zajistit, aby se krabice vzájemně dotýkaly a nevznikalo příliš velké množství nevyužitého prostoru.

Některými z nejpoužívanějších metod jsou:

Výšková mapa

Výšková mapa zaznamenává výšky na různých místech palety, což usnadňuje rozhodování o umístění nových krabic. Bývá často implementována jako dvourozměrná matice, kde každý její prvek představuje výšku zaplnění v daném bodě palety. Jejím postupným skenováním ve zvolených směrech jsou následně určeny dostupné pozice. Skenování může být implementováno například hledáním nenulového rozdílu dvou po sobě jdoucích prvků 2D matice ve zvoleném směru.

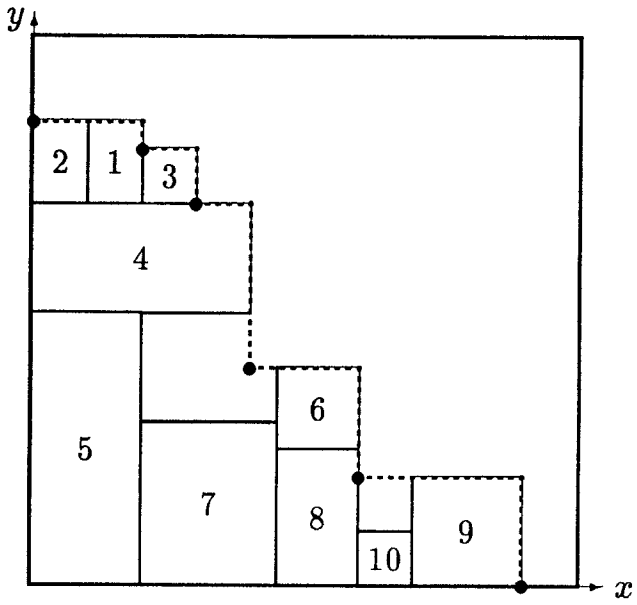
Výhodami této metody je jednoduchost její implementace a díky detailnímu sledování výšek zaplnění je možné nalézt velké množství pozic, vedoucí v efektivnímu využití prostoru.

Jednou z nevýhod této metody je, že s rostoucím vyplnění palety může dojít k výraznému zvětšení 2D matice. To je způsobeno velkou členitostí prostoru, což vede k významnému zvýšení výpočetní náročnosti. Navíc, pokud je i sebemenší plocha krabice nepodepřená, prostor pod krabicí se stává nevyužitelným. Při velké členitosti povrchu může navíc vzniknout příliš velký počet kolizních umístění.

Rohové body

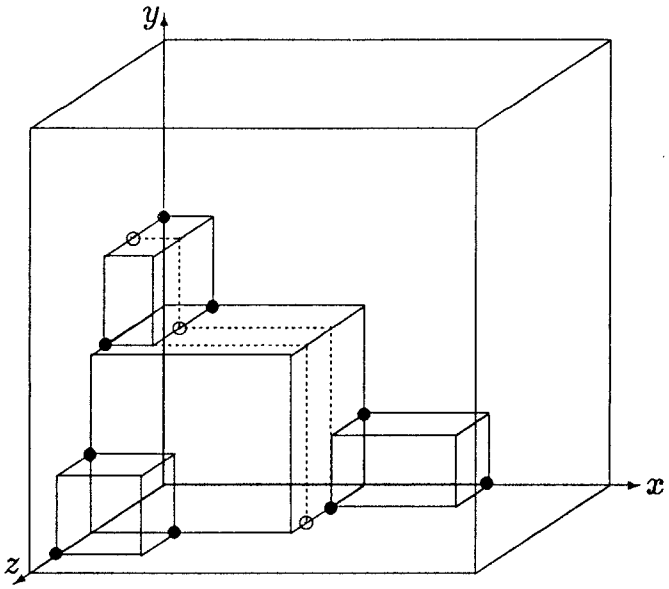
Algoritmus hledání rohových bodů byl navržen v práci od Martello et al. [11], zabývající se řešením 3DBPP. Tento algoritmus lze použít jak pro dvojrozměrné, tak trojrozměrné úlohy. V případě 2D úlohy je založen na nalezení obálky souboru obdélníkových ploch v kartézském souřadnicovém systému. Jeho vstupem je množina I reprezentující pozice a rozměry umístěných obdélníků (Obr. 2.2). Před zahájením je nezbytné množinu seřadit podle koncových bodů obdélníků $[x_i + l_i, y_i + w_i]$, a to sestupně podle souřadnice $y_i + w_i$, případně $x_i + l_i$ v případě shody. Metoda je rozdělena do tří fází:

- Nejprve jsou identifikovány tzv. **Extrémní prvky**, tedy obdélníky jejichž koncový bod se shoduje s bodem, kdy se sklon obálky změní z horizontálního na vertikální.
- Ve druhé fázi jsou nalezeny rohové body jako průsečíky mezi přímkami, rovnoběžnými s osami souřadnicového systému, vycházejících z koncových bodů všech extrémních prvků.
- V poslední fázi je ověřeno, zda při položení nových obdélníků do nalezených bodů nedochází ke kolizi případně vybočení z definovaného prostoru a tyto nevhodné body jsou odstraněny.



Obrázek 2.2: 2D ilustrace výsledku algoritmu hledání rohových bodů [11]

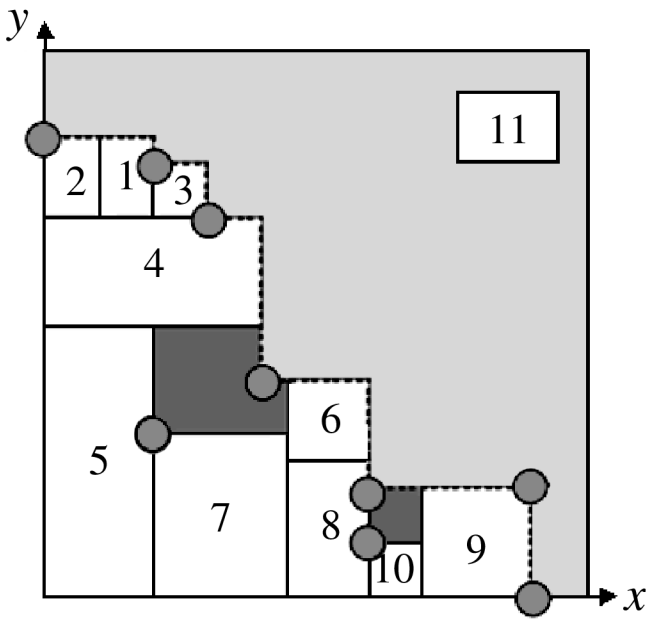
Pro trojrozměrnou úlohu je 2D algoritmus proveden pro každou jedinečnou z souřadnici množiny I. Nevýhodou je však časté nalezení chybných rohů (Obr. 2.3) a je nutné jej tedy rozšířit o tzv. Zkoušku dominance.



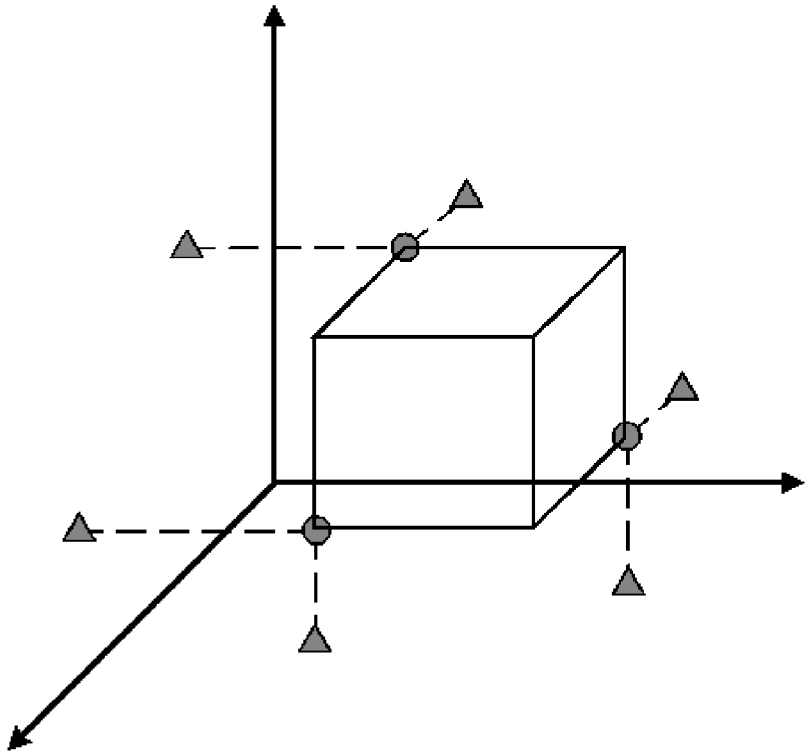
Obrázek 2.3: 3D ilustrace výsledku algoritmu hledání rohových bodů [11]

Extrémní body

Metoda byla představena v práci od Crainic et al. [5], v níž zavádí pojem tzv. **Extrémních bodů**, a jedná se o rozšíření předchozí metody. Základní myšlenkou postupu je, že umístěním krabice s rozměry w_i, d_i a h_i na paletu do pozice (x_i, y_i, z_i), vznikne série nových potenciálních bodů pro umístění. Nové extrémní body jsou generovány projekcí rohových bodů nové krabice $(x_i + w_i, y_i, z_i)$, $(x_i, y_i + d_i, z_i)$ a $(x_i, y_i, z_i + h_i)$ na stěny vymezeného prostoru v ortogonálním směru (Obr. 2.5) a zkoumá se, zda existuje projekce těchto bodů na stěny již umístěných krabic.



Obrázek 2.4: Výsledek metody hledání extrémních bodů [5]



Obrázek 2.5: Projekce vybraných rohových bodů na stěny souřadnicového systému [5]

Na rozdíl od algoritmu hledání rohů, který po každém dalším umístění hledá novou obálku uspořádání krabic, metoda extrémních bodů ponechává zpětně nalezené pozice a při každém umístění aktualizuje seznam bodů přidáním maximálně šesti extrémních bodů. Tento přístup umožňuje dosáhnout nižší výpočetní náročnosti, což je klíčové pro efektivitu a rychlost celého procesu.

Nicméně, tato metoda má své nedostatky. Prvním z nich je, že nedefinuje aktualizaci předchozích pozic po přidání nové krabice. To může vést k neefektivnímu využití prostoru, protože staré pozice zůstávají nezměněny bez ohledu na nově přidané krabice.

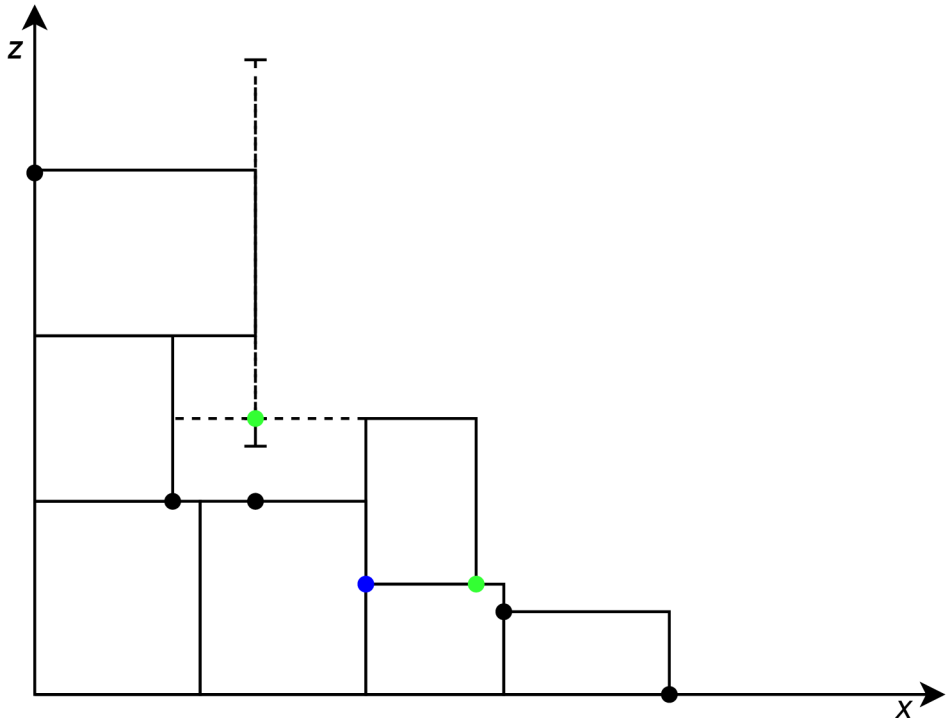
Druhým nedostatkem je, že metoda nemá pevně definovanou metodu pro ověření validní projekce. To znamená, že nemá jasný mechanismus pro ověření, zda je daná projekce možná nebo ne. Špatný návrh projekce tedy může vést k situacím, kdy metoda navrhne nepraktické nebo nemožné uspořádání krabic.

V práci byla použita právě metoda extrémních bodů pro svou nižší výpočetní náročnost. Bylo ovšem nutné definovat vlastní funkce pro projekci a průběžné odstraňování nevhodných pozic.

Návrh pro ověření validní projekce je ilustrován na obrázku 2.6. Projekce je považována za možnou, pokud se osa projekce protíná s úsečkou, vytvořenou prodloužením hrany zkoumaného obdélníku v kolmém směru o polovinu střední hodnoty rozměrů všech umístěných obdélníků. Tím je zaručen vzájemný dotyk objektů, což zvyšuje šanci na efektivní vyplnění prostoru. Pokud je nová krabice umístěna na zvolenou pozici (modrý bod), je zahájen iterační algoritmus, který prochází všechny již umístěné krabice a pro každou z nich zkoumá, zda je projekce v daném směru validní. V uvedeném příkladu je projekce prováděna ve dvou směrech: levý horní roh je promítán ve směru $-x$ a pravý dolní roh ve

směru $-z$. Nalezené extrémní body jsou na obrázku vyznačeny zeleně, zatímco černými body jsou označeny nalezené pozice v předchozích krocích.

V průběhu jsou odstraněny extrémní body ležící uvnitř nově přidané krabice, včetně těch ležících na straně krabice blíže počátku souřadnicového systému.

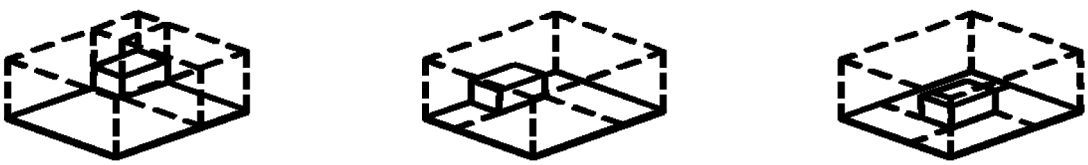


Obrázek 2.6: 2D Ilustrace použité metody projekce

2.2.3 Kritéria hodnocení

Hodnotící kritéria pomáhají určit, které řešení jsou vhodné a které ne, a tím usměrňují algoritmus směrem k hledání optimálního řešení podle cílů dané úlohy. Mezi běžně používaná kritéria hodnocení patří například:

Vytvoření podprostoru (Sub-volume creation) - Při položení krabice do vymezeného prostoru je prostor rozdělen na konečný počet podprostorů. Ohodnocení tedy představuje kolik nových podprostorů přidáním krabice vznikne. [12].



Obrázek 2.7: Ilustrace případů vzniku různého počtů podprostorů

Relativní vyplnění prostoru - Hodnotí, jakou část prostoru krabice vyplňují. Při výpočtu může být uvažován objem celého prostoru palety, nebo objem bounding boxu aktuální konfigurace krabic. [13, 20, 22]

Výška krabice - Zajišťuje, aby krabice byla pokládána na co možno nejnižší pozici. Může být uvažována výška podstavy [20], nebo výška těžiště. Použitím těžiště je docíleno rozdílné ohodnocení pro jednotlivé orientace krabice.

Podepřená plocha (Base of Support) - Posuzuje, jak stabilně jsou krabice umístěny na paletě a zda je zajištěno adekvátní podepření každé krabice. Z hlediska stability je požadováno, aby byla plocha krabice podepřená alespoň ze 70% [17].

Počet podpěrných krabic - Souvisí s vytvářením vazeb vedoucí na stabilnější uspořádání [20].

3 Návrh algoritmu

V této kapitole bude představen návrh algoritmu pro řešení dané instance DPLP. Nejdříve je popsána definice problému, včetně zjednodušujících předpokladů a dalších omezení.

V následující části bude detailně popsán samotný algoritmus řešení DPLP a jeho implementace. Bude uveden výčet klíčových kroků algoritmu a popsány různé strategie, které byly využity k optimalizaci výpočetní náročnosti.

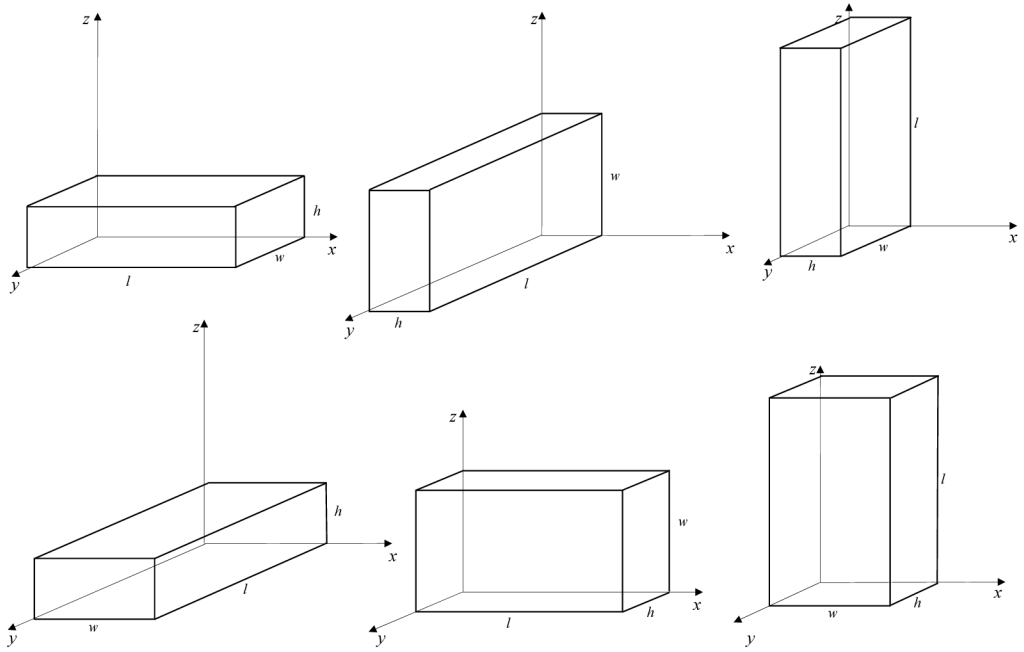
V další podkapitole bude podrobně rozebrána hodnotící funkce, sloužící pro výběr krabice a její umístění. Budou popsány jednotlivé složky této funkce a vysvětleno, jak každá z nich přispívá k celkové optimalizaci. Také budou zmíněny důvody, proč byly určité aspekty hodnoceny přednostně a jaké jsou jejich praktické důsledky při skládání krabic na paletu.

Konečně v závěrečné části kapitoly budou prezentovány výsledky algoritmu při aplikaci na vybrané datasety.

3.1 Definice problému

Paleta je definována rozměry $[L, W, H]$, přičemž obecně neexistuje žádné omezení pro jejich poměr ani velikost. Pro účely práce byly rozměry nastaveny na 800 x 1200 x 2000 mm. První dva rozměry byly zvoleny jako reprezentace europalety EPAL [23]. Paleta je také definována její nosností, tedy maximální hmotností, kterou je paleta schopna snést, aniž by se poškodila. I zde byla konkrétní hodnota inspirována europaletou EPAL s nosností 1000 až 1500 kg v závislosti na rozložení hmotnosti [23].

Krabice jsou definovány rozměry $[w_i, l_i, h_i]$, které nemohou být větší než rozměry palety. Jejich hmotnost m_i je náhodně generována v rozmezí od 0 do 10 kilogramů, za předpokladu že jsou vyplněné, tedy s homogenním rozložením hmotnosti a těžištěm ležícím v geometrickém středu. Kolize krabic je nepřípustná a zároveň nesmí přesáhnout vymezený prostor palety. Jsou povoleny rotace 90°, čímž se omezí maximální počet možných orientací. V krajním případě $l_i \neq w_i \neq h_i$ má krabice tedy jen 6 možných orientací (Obr. 3.1). Dále nejsou uvažovány křehké krabice a z hlediska stability musí být krabice podepřena alespoň ze 70 %.



Obrázek 3.1: Všechny možné orientace krabice [24]

3.2 Hlavní algoritmus

Jak již bylo uvedeno výše, hlavním smyslem algoritmu je z aktuálního zásobníku vybrat příslušnou krabici a umístit ji na jednu z dostupných pozici při dodržení předem stanovených pravidel. Musí být tedy schopný najít dostupné pozice z informací o paletě, identifikovat nepřípustné stavy zahrnující kolize krabic nebo nedostatečné podepření, a využitím relevantních kritérií zvolit následující stav palety.

Část použitých metod již byla diskutována v teoretické části práce, a proto v této sekci budou vysvětleny převážně z hlediska jejich pořadí v navrženém postupu. Větší část kapitoly bude věnována zvolení kritérií pro hodnocení potenciálních stavů a jejich opodstatnění.

Výsledek implementace algoritmu se dá rozdělit do dvou úrovní:

Vyšší úroveň zahrnuje inicializaci palety a definici zásobníku, hlavní cyklus pokládání krabic, probíhající dokud není dosažena ukončovací podmínka, a také vizualizaci, díky které mohly být identifikovány funkcionální chyby v průběhu vypracování a sloužila také jako jedna z metod prezentace výsledků (Obr. 3.3).

Nižší úroveň představuje samotnou strategii výběru krabice, rozdělené do prohledání stavového prostoru do omezené hloubky, ověřování podmínek kolize a podepření, a následným ohodnocením jednotlivých stavů následující výběrem nového.

3.2.1 Inicializace

Pro inicializaci problému skládání palety definujeme strukturu palety a je připraven vstupní dataset s informacemi o krabicích, z něhož je vytvořena první instance zásobníku.

Definice struktury palety

- **Rozměry** palety jsou definovány ve formátu $[L, W, H]$, v práci byly zvoleny rozměry $[800, 1200, 2000]$ mm.
- **Maximální hmotnost** M 1000 kg vzhledem k nehomogennímu rozložení hmotnosti.
- **Množina krabic** I obsahuje informace o krabicích na paletě.
- **Relativní vyplnění** V_r, představuje poměr sumy objemů krabic k celkovému objemu prostoru palety.
- **Ohodnocení** C_k: Numericky hodnotí kvalitu konfigurace (Podkap. 3.2.4)
- **Numerický identifikátor** ID slouží k rozlišení jednotlivých stavů a zpětnému hledání předcházejících uzlů ve stavovém prostoru.
- **Extrémní body** EPs obsahují souřadnice možných umístění, nalezených při expanzi.

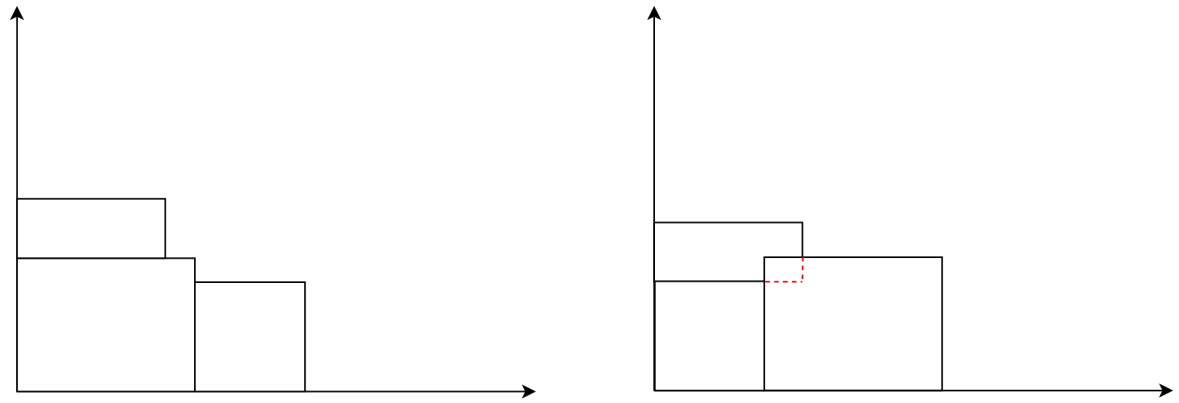
Definice zásobníku

Vstupní dataset je vytvořen dvěma možnými způsoby. V počátku je buď definováno velké množství typů krabic s různými rozměry v určeném intervalu, nebo importovány z existujícího datasetu. Po každém přechodu je nově umístěná krabice nahrazena krabicí náhodně vybranou ze vstupního datasetu. Strukturu krabice definují:

- **Rozměry** $[l_i, w_i, h_i]$ určují délku, šířku a výšku krabice.
- **Pozice** $[x_i, y_i, z_i]$ představují souřadnice rohu krabice nejblíž počátku souřadnicového systému.
- **Hmotnost** m_i je přiřazena náhodně v intervalu od 0 do 10 kg.
- **Numerický identifikátor** id slouží k rozlišení krabic v zásobníku.

Součástí inicializace je kvůli urychlení procesu položení první krabice bez předchozího prohledávání, přičemž je zvolena krabice největšího objemu. Tímto je vyloučeno množství možných kolizních uspořádání (Obr.3.2), což alespoň v počátcích paletizace může zlepšit výsledky optimalizace. Pro prázdnou paletu je zřejmé, že počáteční krabice bude položena do jednoho z rohů palety, reprezentující počátek souřadnicového systému. Vyhneme se tím jednoho cyklu prohledávání. Tato základní inicializace poskytuje výchozí bod pro následné řešení dané instance DPLP.

Po ukončení inicializace je zahájen hlavní cyklus skládání, který pokračuje dokud se nezmění pravdivostní hodnota ukončovacích podmínek. Cyklus může být ukončen více způsoby, a sice překročením maximální nosnosti palety nebo nenalezením žádných nových přípustných stavů.



Obrázek 3.2: Příklad konfigurace vedoucí ke kolizního stavu společně s bezkolizní konfigurací

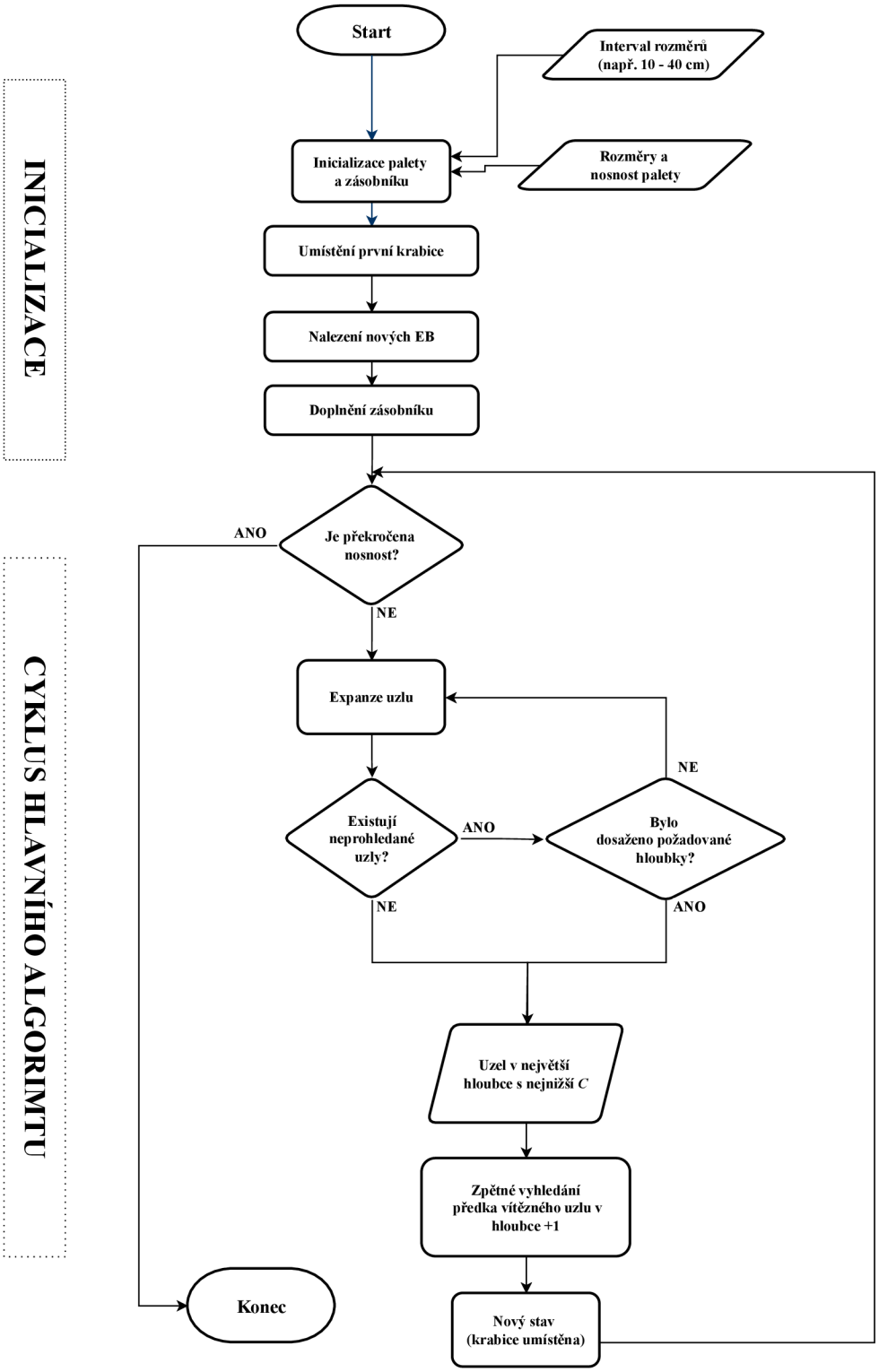
3.2.2 Prohledávání stavového prostoru

Podstatou výběru krabic v průběhu skládání je prohledávání stavového prostoru do předem definované hloubky. V práci byl uvažován zásobník s menším množstvím krabic, hloubka prohledávání byla tedy rovna velikosti zásobníku. V případě většího zásobníku je možné zvolit maximální hloubku prohledávání, nicméně kvůli razantně se zvyšující době výpočtu byla tato možnost velmi omezená.

Postup prohledávání byl implementován následovně:

- Do každého extrémního bodu jsou postupně umístěny všechny známé krabice ve všech možných orientacích.
- Je zkontrolována podmínka, zda je krabice dostatečně podepřená a zároveň je vyloučena kolizní konfigurace.
- Při splnění předchozích podmínek je vytvořen nový uzel, kterému je také přiřazeno ohodnocení, podrobněji diskutováno v 3.2.4.
- Je aktualizována množina extrémních bodů.

Prohledávání je ukončeno nalezením všech možných uzlů stavového stromu po vyčerpání všech krabic ze zásobníku, případně dosažením zvolené hloubky. V pozdějších fázích paletizace, například při polovičním zaplněním prostoru, může být počet nalezených uzlů natolik velký, že je nezbytné zavést eliminační kritéria, která přispějí k snížení výpočetní náročnosti. Při nesplnění podmínek se daný uzel neuvažuje a není uložen do paměti.

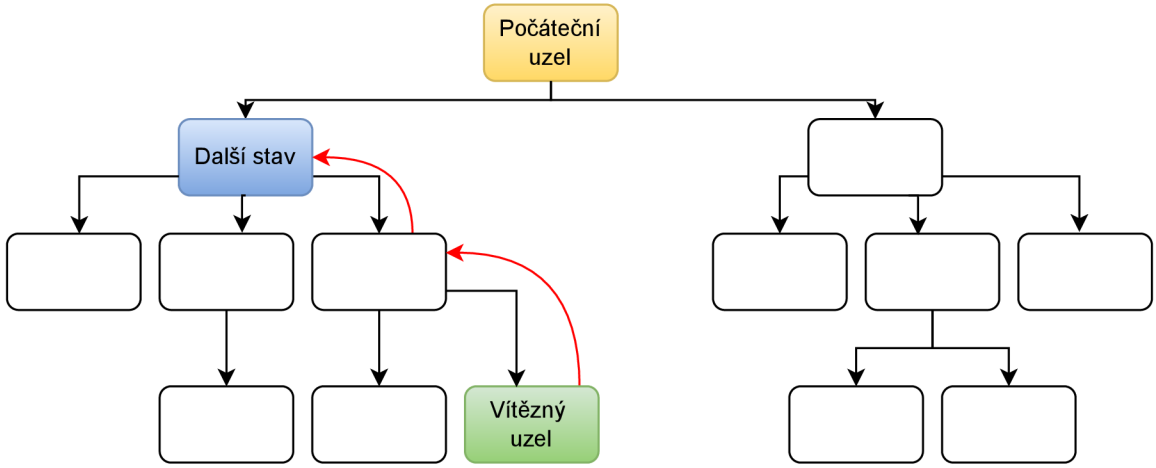


Obrázek 3.3: Flowchart popisující hlavní kroky algoritmu

3.2.3 Volba vítězného uzlu

Vítězný uzel je vybrán z uzlů nejvyšší dosažené hloubky na základě číselného ohodnocení, představující kumulativní cenu přechodů. Jakmile je vítězný uzel identifikován, je provedeno zpětné vyhledání toho předka, který leží v hloubce o jednu vyšší než je hloubka počátečního uzlu (Obr. 3.4). Ten se následně stane dalším počátečním stavem pro další iteraci hlavní smyčky. Tento postup zajišťuje, že je vybrán nejlepší možný stav pro další expanzi.

Tento přístup zároveň zohledňuje fakt, že v další iteraci nabude zásobník nové podoby představením nové krabice, čímž vzniknou nové kombinace pořadí krabic a jejich umístění.



Obrázek 3.4: Schéma větvení stavového stromu s ilustrací výběru nového stavu palety

3.2.4 Návrh hodnotící funkce

V návrhu hodnotící funkce jsou specifikovány kritéria pro ohodnocení jednotlivých uzlů stavového stromu. Hodnocení je založeno na výpočtu kumulované ceny přechodů mezi jednotlivými uzly v rámci jednoho cyklu prohledávání. Vlastní hodnocení uzlu je vypočítáno jako vážená suma normalizovaných cen, která zahrnuje různé aspekty krabic a jejich umístění na paletě.

$$C = \sum_{i=1}^{n} W_i c_i \tag{3.1}$$

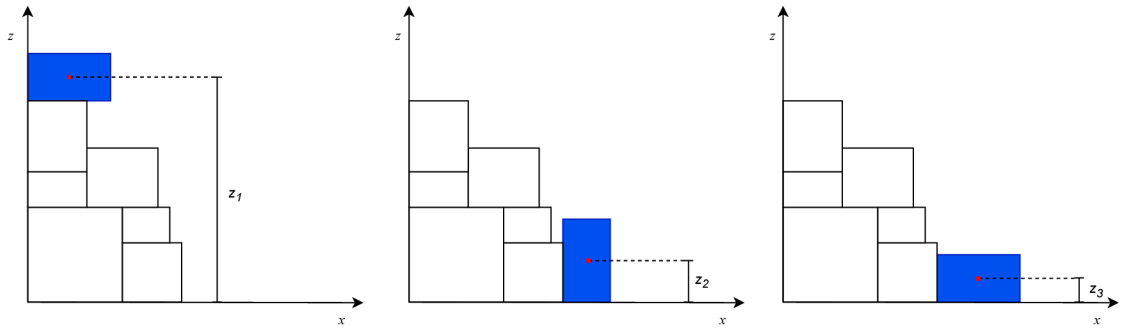
Vybraná kritéria

Poloha těžiště krabice z_T

Výška těžiště krabice v daném umístění hraje klíčovou roli při hodnocení. Zaručuje, že algoritmus upřednostní umístění krabic do nižších pozic a zároveň lépe zhodnotí stabilnější orientaci krabice v téže pozici. To vede k rovnoměrnějšímu vyplňování a může zabránit časté tvorbě nestabilních sloupců.

Normalizovaná cena je stanovena jako podíl souřadnice těžiště krabice a maximální možné výšky na paletě.

$$c_H = \frac{z_T}{H - h_i/2} \tag{3.2}$$



Obrázek 3.5: Ilustrace možných umístění pro jednu krabici s různou výškou těžiště. V této situaci je upřednostěná možnost vpravo.

Vzdálenost těžiště od počátku d_o

Vzdálenost těžiště krabice od počátku palety má význam v rámci uvažování pokládání krabic manipulátorem s pevně fixovanou základnou. Smyslem použití je předejít některým možným kolizním stavům s ramenem manipulátoru snahou skládat krabice rovnoměrně v oblasti počátku.

$$d_o = \sqrt{(x_i + \frac{l_i}{2})^2 + (y_i + \frac{w_i}{2})^2 + (z_i + \frac{h_i}{2})^2} \tag{3.3}$$

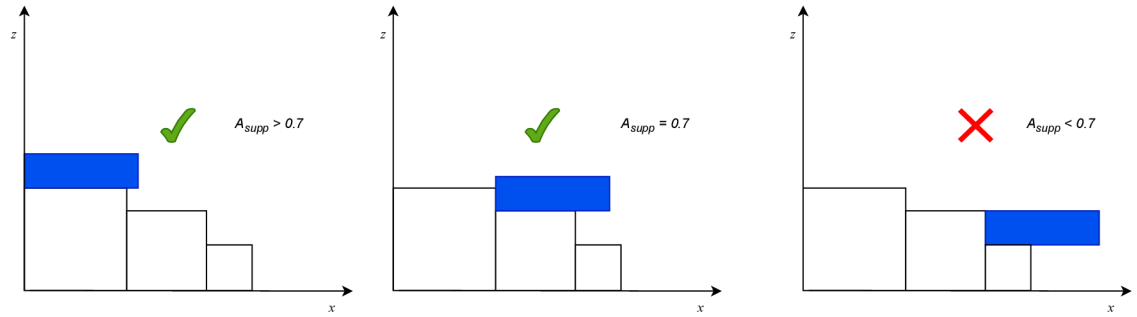
Normalizovaná cena je určena jako rozdíl vzdálenosti těžiště krabice v aktuální orientaci a minimální vzdálenosti od počátku podělený rozdílem minimální a maximální možné vzdálenosti.

$$c_{d_o} = \frac{d_o - d_{min}}{d_{max} - d_{min}} \tag{3.4}$$

Podepřená plocha $A_{supp} \in \langle 0.7, 1 \rangle$

Podepřená plocha krabice, je významnou metrikou pro stabilitu a bezpečnost na paletě. Normalizovaná cena roste se snižujícím procentuálním podepřením. To znamená, že čím větší je podepřená plocha, tím nižší je hodnota této metriky. Podepření může nabývat hodnot pouze v rozmezí od 0.7 do 1, normalizace má proto následující podobu.

$$c_{A_{supp}} = -\frac{10}{3}(A_{supp} - 1) \tag{3.5}$$



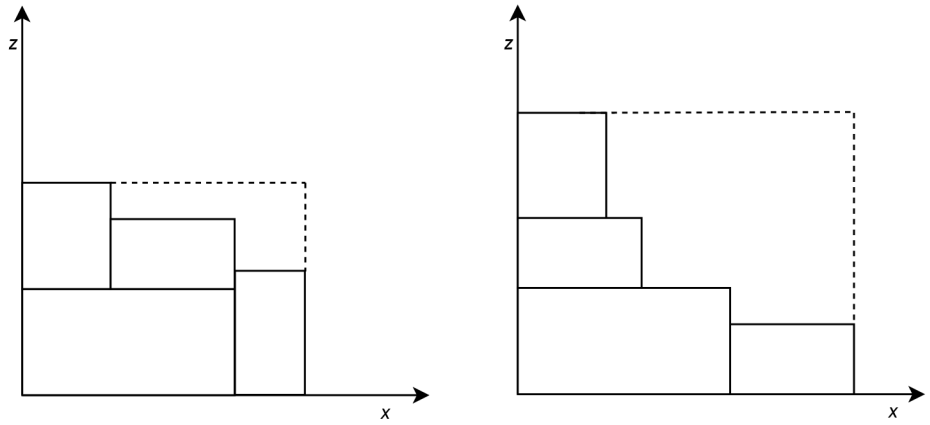
Obrázek 3.6: Ilustrace možných umístění pro jednu krabici s různým podepřením

Počet krabic podpírající novou krabici N_{supp}
Z hlediska stability je toto kritérium naprosto zásadní. Smyslem použití je totiž upřednostnění tvorby vazeb mezi jednotlivými krabicemi, podobně jako při stavbě cihlové zdi. Normalizovanou cenu prakticky nelze spolehlivě stanovit, jelikož není znám maximální možný počet krabic, které novou krabici je schopno podepřít. Tento člen bude tedy více závislý na vhodném zvolení váhové konstanty. Cena je určena jako převrácená hodnota počtu krabic, což znamená, že čím méně krabic podpírá novou krabici, tím vyšší je hodnota této dílčí ceny. V případě, že krabice leží na zemi, normalizovaná cena nabude hodnoty -1.

$$c_{N_{supp}} = \frac{1}{N_{supp}} \tag{3.6}$$

Relativní vyplnění prostoru
Vyjadřuje poměr celkového objemu krabic na paletě k objemu bounding boxu obklopující současnou konfiguraci krabic. Oproti předchozím kritériím je hodnota lokálního zaplnění zahrnuta pouze pro uzly s nejvyšší dosaženou hloubkou v aktuálním procesu prohledávání. Normalizovaná cena klesá s vyšším procentuálním zaplněním objemového prostoru. Smyslem kritéria je snaha dosáhnout co možno nejkompaktnějšího uspořádání a efektivního využití prostoru.

$$V_{\text{rel}} = \frac{\sum_{i=1}^{n} V_i}{V_{bb}} \tag{3.7}$$



Obrázek 3.7: Ilustrace dvou případů možných umístění stejných krabic s odlišným relativním vyplněním

Normalizovanou cenu získáme rozdílem 1 a hodnoty relativního objemového vyplnění.

$$c_{V_{rel}} = 1 - V_{rel} \tag{3.8}$$

Cenová funkce přechodu z jednoho stavu do druhého tedy nabývá podoby:

$$C = W_1 c_z + W_2 c_{d_0} + W_3 c_{A_{supp}} + W_4 c_{N_{supp}} \tag{3.9}$$

Pro uzel v největší možné hloubce v aktuálním výběru krabice poté platí:

$$C = W_1 c_z + W_2 c_{d_0} + W_3 c_{A_{supp}} + W_4 c_{N_{supp}} + W_5 c_{V_{rel}} \tag{3.10}$$

Konečné ohodnocení uzlů má podobu kumulované ceny přechodů v rámci všech předcházejících stavů.

$$C_{k+1} = C_k + C \tag{3.11}$$

Vhodné váhové konstanty jsou laděny empiricky pro dosažení požadovaného chování algoritmu. Nelze však zaručit, že povedou na chtěný výsledek pro jakoukoliv kombinaci vstupní typů krabic. Je prakticky nemožné zvolit váhové konstanty tak, aby byl schopný dosáhnout kýženého výsledku ve všech situacích. Uvedená rovnice 3.12 znázorňuje pouze podobu cenové funkce s vahami, které dosáhli poměrně slušných výsledků na použité datové sady.

$$C = 5c_z + 0.1c_{d_0} + c_{A_{supp}} + c_{N_{supp}} + c_{V_{rel}} \tag{3.12}$$

3.3 Výsledky

Pro vyhodnocení výsledků budou použita následující kritéria: procentuelní vyplnění prostoru, rychlost výpočtu, počet krabic na paletě a poloha těžiště.

Vyplnění prostoru bude vypočítáno jako poměr součtu objemů všech krabic na paletě a objemu celkového vymezeného prostoru palety.

Z hlediska obecného hodnocení není **počet krabic** příliš relevantní vzhledem k tomu, že počítáme s náhodným výběrem krabic a nemáme dostatek informací o konečné podobě výsledné palety. Nicméně poslouží jako porovnání konečných konfigurací v rámci stejných intervalů rozměrů, případně pro různé velikosti zásobníku.

Poloha těžiště je jedním z klíčových ukazatelů celkové stability palety. Jelikož pracujeme s různými rozměry a hmotnostmi, těžiště spočítáme jako vážený průměr jednotlivých souřadnic těžiště všech krabic (Rovnice 3.13, 3.14, 3.15). Za ideální těžiště je považován geometrický střed palety, tedy $[40, 60, 100]$ cm.

$$x_t = \frac{\sum_{i=1}^{n} x_i m_i}{\sum_{i=1}^{n} m_i} \tag{3.13}$$

$$y_t = \frac{\sum_{i=1}^{n} y_i m_i}{\sum_{i=1}^{n} m_i} \tag{3.14}$$

$$z_t = \frac{\sum_{i=1}^{n} z_i m_i}{\sum_{i=1}^{n} m_i} \tag{3.15}$$

Rychlostí algoritmu se rozumí jednoduše čas, který algoritmus potřebuje na získání výsledku, tedy čas potřebný pro složení celé palety. Během výpočtu je zakázáno používání jakéhokoliv procesu, který je časově náročný a není nezbytný pro funkci algoritmu, jako je například hudební přehrávač nebo internetový prohlížeč . Algoritmus nebude využívat žádné přebytečné časově náročné funkce, jako je například výpis do příkazového panelu. Do rychlosti algoritmu nejsou započítány časově náročné operace, jako je vykreslení průběžného stavu palety, ani čas pro výpočet kritérií hodnocení algoritmu.

Následující část je rozdělena do dvou částí: výsledky dosažené pro vytvořené sady krabic s náhodnými rozměry a poté pro datasety převzaté z [12],[25].

V případě generace vlastních sad krabic jsou prezentovány výsledky pro rozměry ve variabilních intervalech v rozmezí od 10 do 50 cm. Tím ověříme chování algoritmu v závislosti na různorodosti zásobníku a také potvrdíme důležitost poměru rozměrů krabic k rozměrům palety. Porovnáme také výsledky v závislosti na velikosti zásobníku, přičemž předpokládáme, že při větším zásobníku, neboli při větší hloubce prohledávání, bychom měli dosáhnout lepších výsledků. Souhrn dosažených výsledků je vyobrazen v tabulce 3.2 pro vlastní sady krabic a 3.1 pro ostatní datasety.

V každé instanci je vygenerována sada 100 krabic, z kterých je náhodným výběrem zásobník doplňován.

3.3.1 Vlastní datasety

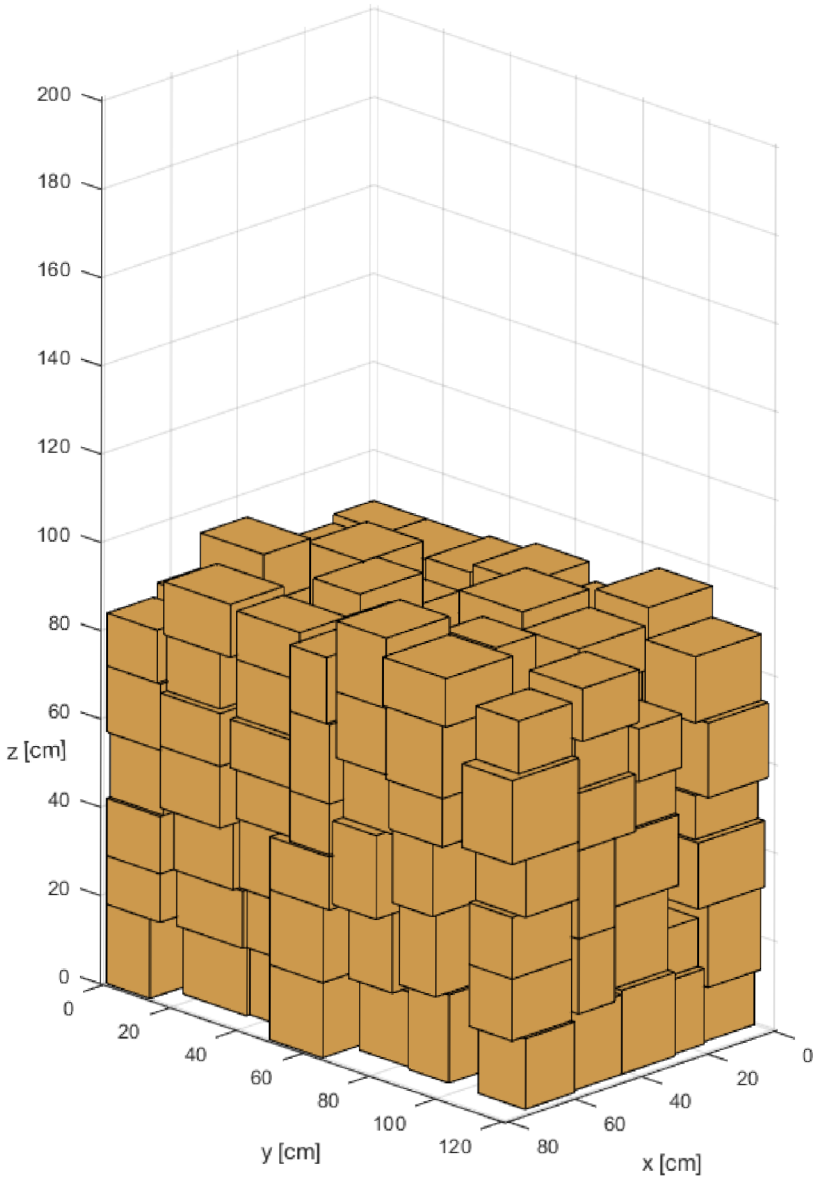
Během experimentace byly rozměry krabic generovány náhodně v rozmezí mezi 10 a 50 cm a byly zohledněny pouze celočíselné hodnoty. Tímto přístupem jsme zavedli určitou toleranci, která reflektuje možné odchylky ve skutečných podmínkách, kdy mohou být krabice deformované nebo může docházet k chybám měření. Tímto způsobem jsme se snažili modelovat reálné podmínky a zohlednit možné nepřesnosti měřící metody.

Velmi malé krabice
Z obrázku 3.8 je patrné, že byla vyplněna přibližně jen polovina palety, a to kvůli dosažení maximální nosnosti. Při průměrné hmotnosti 5 kg je paleta schopná pojmout jen kolem 200 krabic, přičemž na této paletě jich bylo položeno 206. Je možné si také povšimnout velmi výrazné členitosti povrchu. Díky velké různorodosti rozměrů je velmi obtížné za daných podmínek umístit krabice tak, aby jejich horní stěny měly stejnou výšku a tím společně vytvořily souvislou plochu pro vytváření vazeb. To má bohužel za následek, že se krabice skládají do vysokých sloupců.

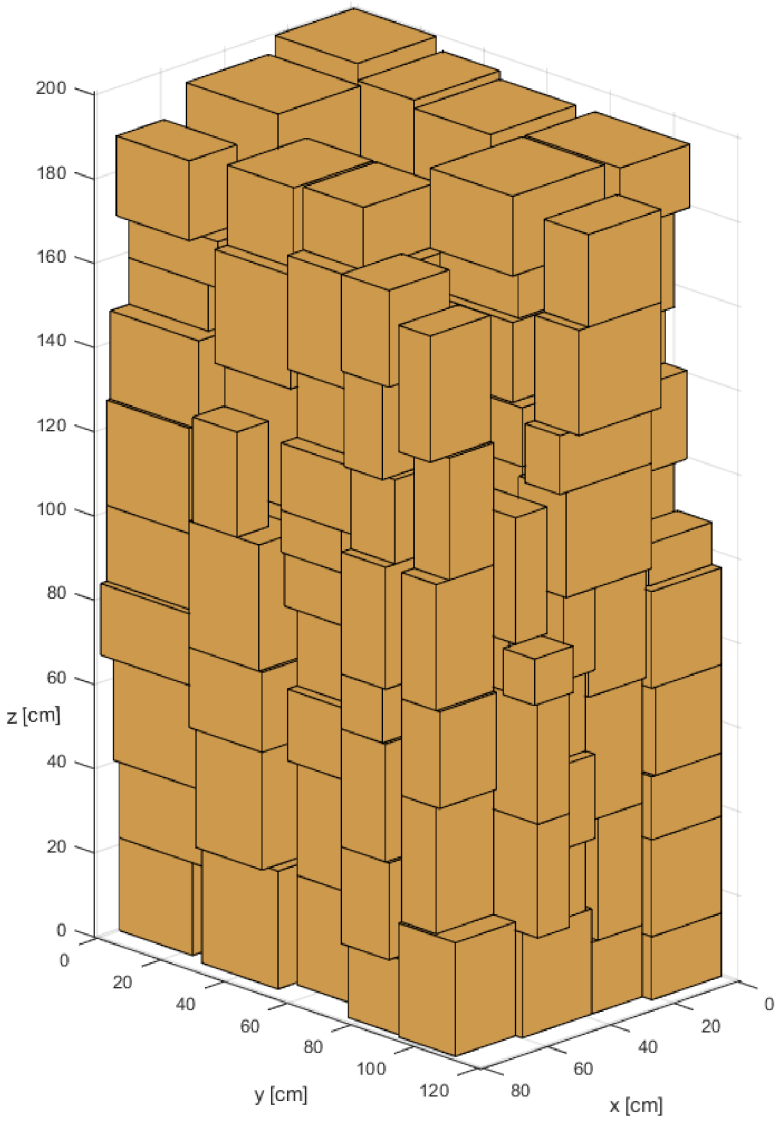
Mírným rozšířením intervalu rozměrů již podmínka nosnosti dosažena nebyla a proces skládání byl ukončen v okamžiku, kdy každá další přidaná krabice vybočovala z definovaného prostoru (Obr.3.9).

Kombinace velkých a malých krabic (Obr.3.10)
Kombinace větších a menších krabic umožnila dosáhnout lepšího vyplnění prostoru ve srovnání s předchozím případem. To bylo možné díky tomu, že menší krabice často vyplňovaly mezery vzniklé mezi velkými krabicemi. Nicméně při nízké hloubce prohledávání nebo malém zásobníku docházelo k velmi nízkému využití prostoru a podobně jako v předchozím případě nedocházelo k dostatečnému vytvoření vazeb.

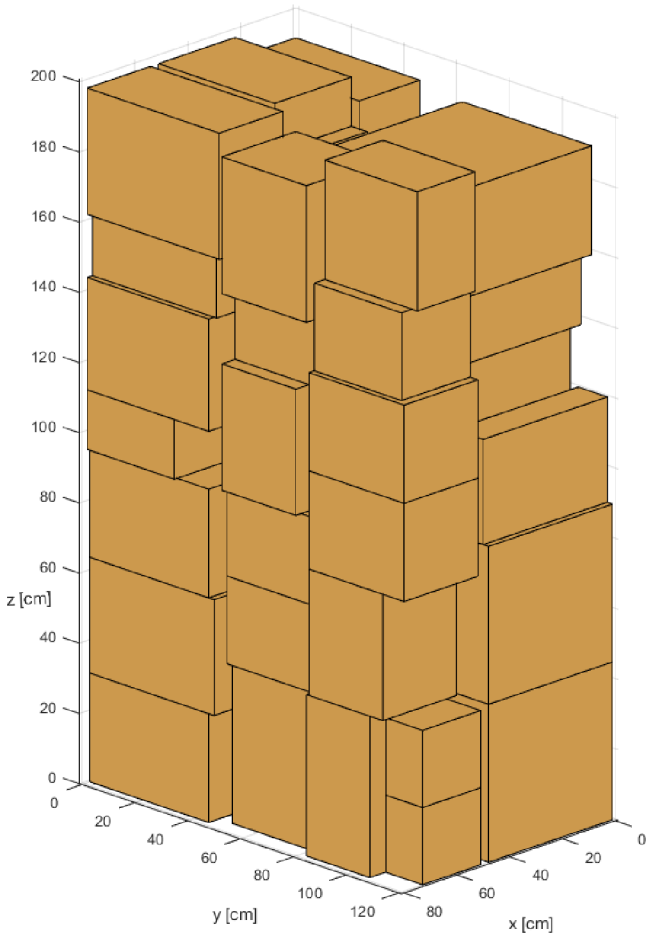
Velké krabice (Obr.3.11)
Při překročení určitého poměru rozměrů krabic vůči rozměrům palety již nevzniká dostatečné množství kombinací a vytrácí se závislost mezi vyplněním prostoru a velikostí zásobníku či hloubkou prohledávání. V takovém případě je možné pouze stohování do sloupců.



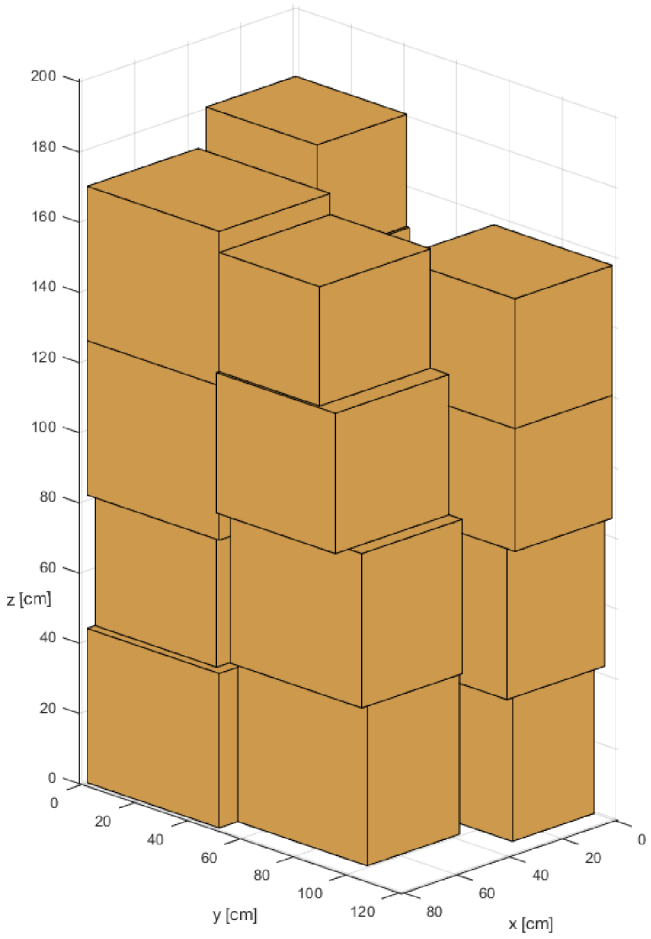
Obrázek 3.8: Výsledná paleta krabic rozměrů 10-20 cm, paletizace ukončená podmínkou nosnosti



Obrázek 3.9: Výsledná paleta krabic rozměrů 10 - 30 cm



Obrázek 3.10: Výsledná paleta krabic rozměrů 10 - 50 cm



Obrázek 3.11: Výsledná paleta krabic rozměrů 30 - 50 cm

3.3.2 Testované datasety

Dataset z [12]

Prvním použitým datasetem je sada krabic použitá v práci od Abdou et al. [12], jejichž rozměry jsou vypsány v tabulce 3.2. Je patrné, že existuje spousta konfigurací, které vedou na vytvoření souvislých ploch, jelikož pro jejich rozměry jsme schopni najít hned několik společných dělitelů. Očekáváme tedy častější tvoření vazeb a celkově stabilnější uspořádání, což potvrzuje obrázek 3.12.

Přestože docházelo k častější tvorbě vazeb, stále se krabice primárně pokládaly do nestabilních sloupců. Hlavní příčinou je omezená velikost zásobníku, kdy v daný moment nebyla k dispozici ta krabice, která by dokázala vytvořit požadované kompaktní uspořádání. Větší počet krabic v zásobníku by mohl vést na lepší uspořádání, nicméně obecně to zaručit nelze.

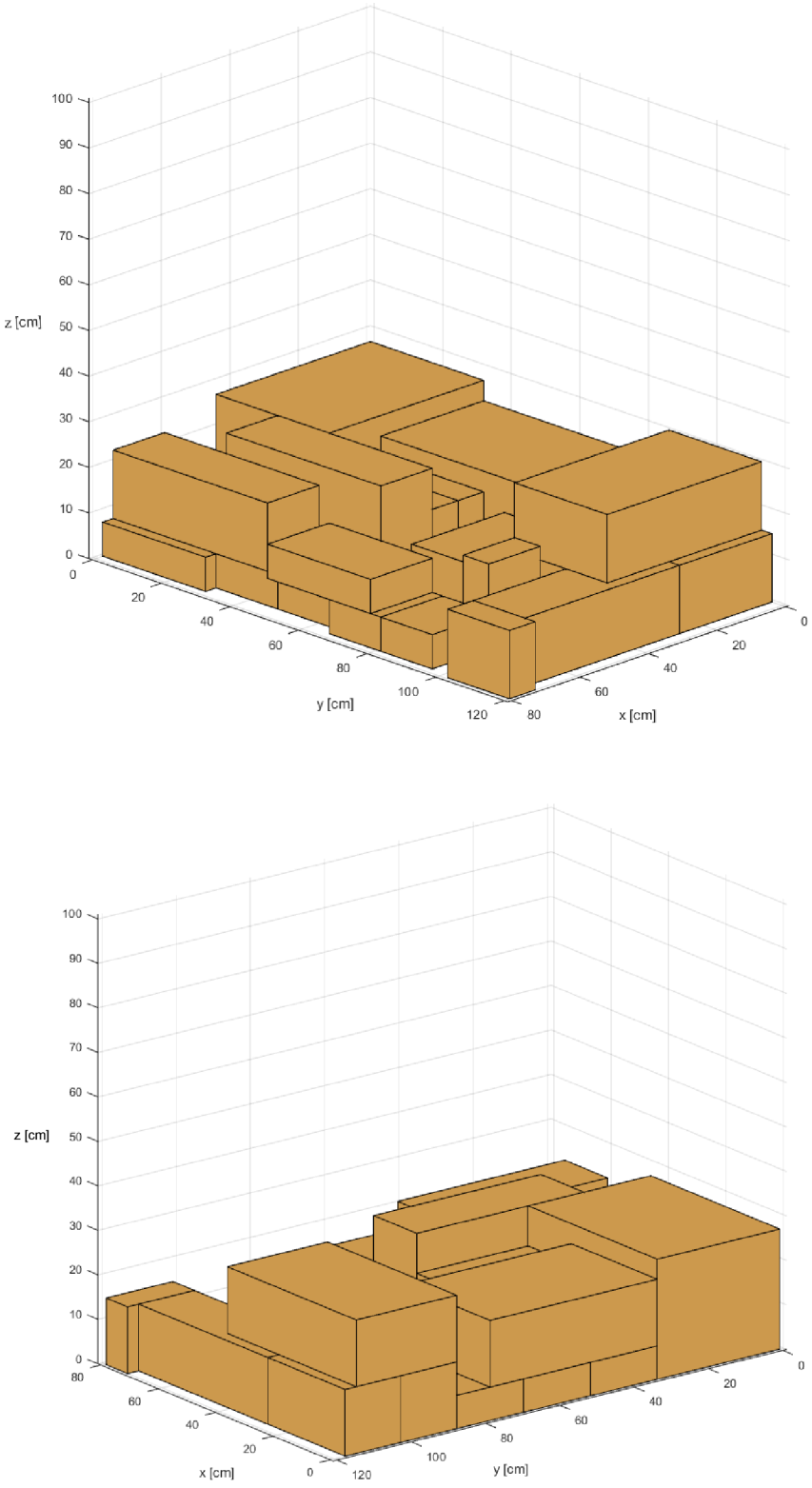
-	1	2	3	4	5	6	7	8
$\mathbf{l}_i$	12	12	18	18	30	30	25	22
$\mathbf{w}_i$	10	20	10	30	10	20	20	30
$\mathbf{h}_i$	5	5	10	10	10	10	15	18

Tabulka 3.2: Dataset poměrných rozměrů slabě heterogenních krabic [12]

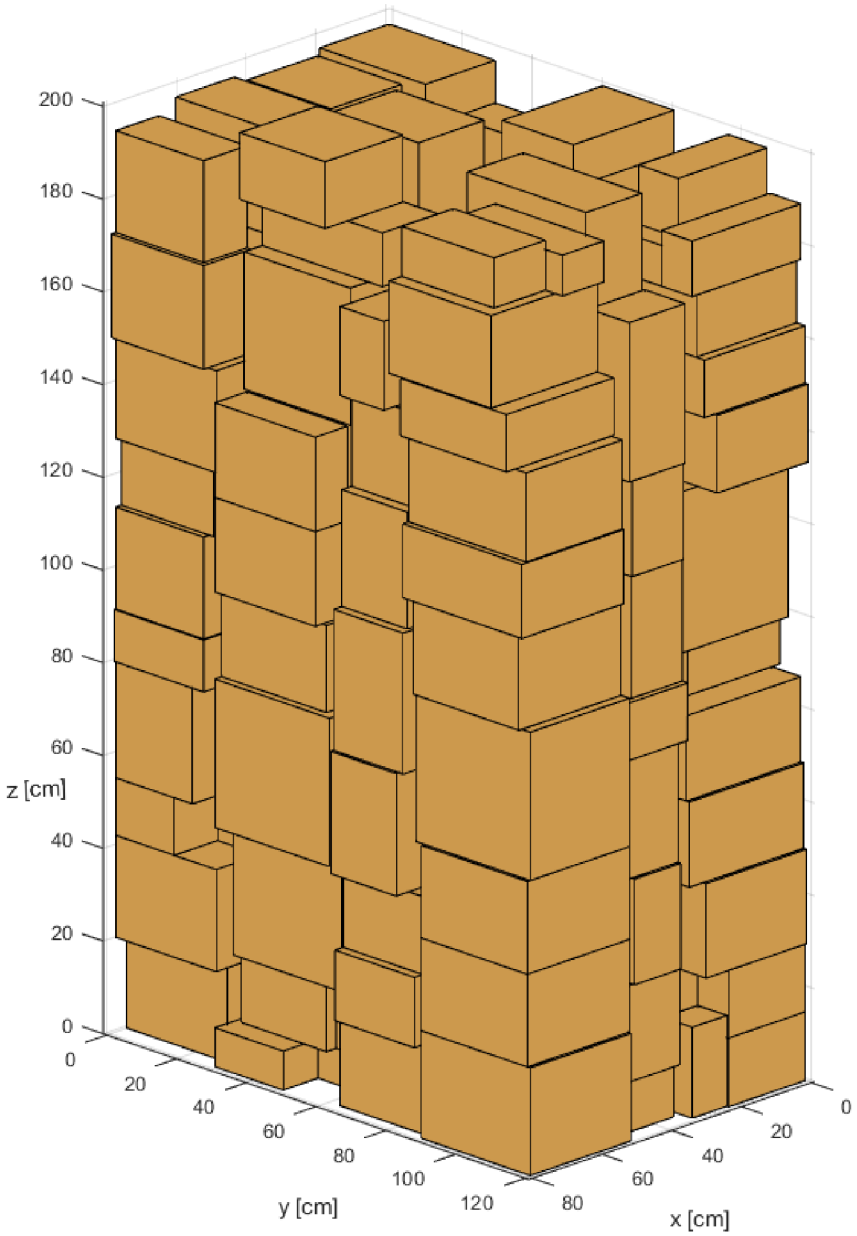
Dataset z Operational Research (OR) knihovny [25] (Obr. 3.13)

OR knihovna je rozsáhlá databáze datasetů pro rozmanité optimalizační úlohy, včetně datasetů krabic pro CLP. Byly použity 2 balíčky z tohoto datasetu a rozměry krabic byly vhodně naškálovány, aby byly přizpůsobeny rozměrům paletizačního prostoru.

Z hlediska vyplnění i stabilních požadavků bylo dosaženo podobných výsledků jako v předchozích případech.



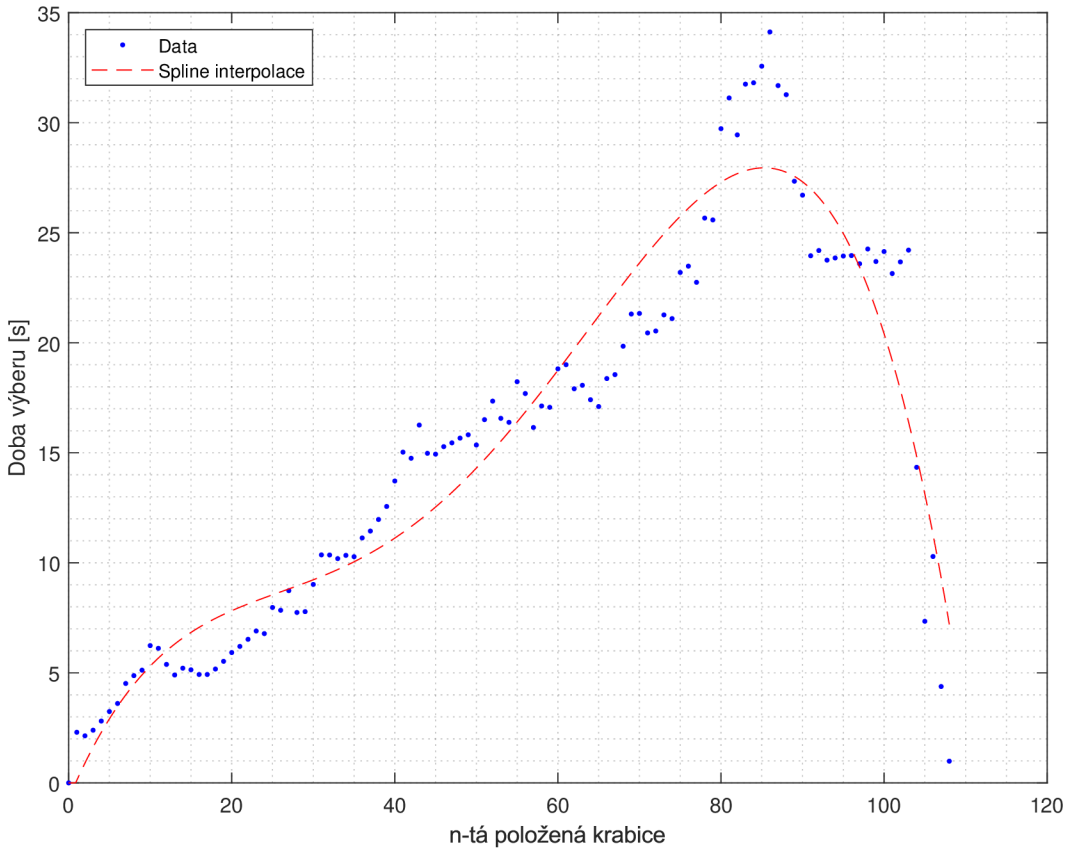
Obrázek 3.12: Ukázka větší tvorby vazeb pro převzatý dataset z [12]



Obrázek 3.13: Výsledná paleta pro krabice z OR knihovny

Velkým problémem současné metody je výrazné prodlužování času potřebného pro výběr následující krabice s rostoucím počtem krabic na paletě. Tento problém je způsoben především implementací několika dílčích funkcí algoritmu pomocí for smyček, které procházejí všechny již položené krabice. Toto se týká například funkce pro detekci kolizí, hledání nových pozic nebo výpočtu podepřené plochy.

V průběhu paletizace se počet možných pozic pro umístění dalších krabic postupně zvětšuje, což vede k exponenciálnímu nárůstu času potřebného k výběru následující krabice. Typický průběh tohoto nárůstu je znázorněn na obrázku 3.14.



Obrázek 3.14: Typický průběh doby výběru krabice při paletizaci

Rozměry krabic [cm]	*Počet krabic v zásob-níku*	*Celkový počet krabic*	*Vyplnění [%]*	*Doba paletizace [s]*	*Poloha těžiště* $[x_t, y_t, z_t]$
10 - 20	1	205	35.43	118	[36.04, 55.26, 42.56]
	2	199	37.69	379	[38.73, 54.07, 46.08]
	3	205	35.93	1658	[36.04, 55.26, 42.56]
	4	199	38.35	3061	[38.73, 54.07, 46.08]
10 - 30	1	155	61.28	43	[35.10, 58.65, 78.01]
	2	157	62.26	191	[38.69, 62.44, 81.31]
	3	169	72.21	605	[37.31, 62.17, 86.43]
	4	189	72.04	1546	[36.89, 60.21, 87.20]
10 - 40	1	93	68.77	5	[35.64, 63.20, 68.93]
	2	92	66.37	16	[35.91, 56.40, 87.30]
	3	90	63.93	31	[33.94, 57.00, 83.78]
	4	78	76.3	56	[39.82, 55.89, 93.11]
10 - 50	1	35	39.55	2	[37.96, 57.64, 56.22]
	2	44	55.13	17	[42.02, 59.73, 80.14]
	3	53	72.72	48	[42.08, 59.77, 88.25]
	4	57	77.96	125	[39.76, 53.28, 82.50]
20 - 30	1	85	68.86	8	[36.59, 55.01, 85.66]
	2	91	71.79	48	[37.16, 53.52, 89.51]
	3	97	75.66	86	[39.18, 59.86, 93.24]
	4	84	66.39	156	[36.60, 63.33, 85.95]
20 - 40	1	44	64.26	1.5	[31.76, 49.80, 84.40]
	2	50	68.79	9	[37.75, 64.24, 90.59]
	3	55	74.60	12	[38.08, 64.71, 104.79]
	4	49	69.07	33	[33.70, 58.48, 97.02]
20 - 50	1	26	59.63	1	[37.86, 51.29, 85.81]
	2	28	70.91	2	[36.47, 61.83, 80.32]
	3	35	74.33	13	[41.72, 57.23, 91.23]
	4	31	64.12	30	[36.10, 59.68, 79.28]
30 - 40	1	27	64.78	1	[40.10, 53.17, 81.57]
	2	35	75.50	5	[39.18, 58.89, 98.14]
	3	34	72.79	34	[38.14, 48.23, 98.53]
	4	35	75.55	102	[36.75, 45.77, 95.44]
30 - 50	1	20	67.1	0.5	[39.36, 54.84, 81.39]
	2	19	56.7	1	[39.33, 54.32, 57.34]
	3	22	68.48	5	[41.61, 55.88, 86.39]
	4	21	72.00	24	[36.01, 66.78, 90.26]

Tabulka 3.1: Tabulka dosažených výsledků pro vygenerované datasety

Rozměry krabic [cm]	*Velikost zásob-níku*	*Celkový počet krabic*	*Vyplnění [%]*	*Doba paletizace [s]*	*Poloha těžiště* $[x_t, y_t, z_t]$
Dataset z [12]	1	82	67.71	6	[38.10, 58.38, 87.60]
	2	85	72.31	40	[39.10, 57.34, 92.40]
	3	92	75.47	69	[37.04, 59.26, 92.56]
	4	95	76.14	114	[38.73, 57.07, 96.08]
Dataset OR 1 [25]	1	84	65.26	5	[39.10, 58.45, 89.78]
	2	98	73.45	36	[39.16, 58.34, 93.41]
	3	101	75.94	112	[37.45, 59.96, 91.76]
	4	96	74.12	259	[39.03, 58.46, 97.51]
Dataset OR 2 [25]	1	95	67.73	8	[38.78, 59.08, 91.63]
	2	85	69.23	56	[39.69, 58.44, 95.47]
	3	92	70.03	87	[37.04, 59.26, 91.58]
	4	95	73.23	301	[39.53, 58.23, 97.68]

Tabulka 3.3: Tabulka dosažených výsledků pro převzaté datasety

4 Simulace

V této kapitole se zaměříme na simulační ověření vyvinutého algoritmu pro paletizaci s ohledem na jeho praktickou použitelnost a splnění stanovených kritérií. Kapitola je rozdělena do dvou hlavních částí.

Nejprve je simulačně ověřena stabilita palety v průběhu paletizace (kap. 4.1). Tato část se zaměřuje na zajištění, že paleta během stavby nezkolabuje a že všechny krabice setrvají na svém místě bez rizika převrácení.

V následující části provedeme simulaci v robotickém simulátoru, který umožňuje realistické modelování a analýzu chování robotického manipulátoru při přemísťování krabic (kap. 4.2). Cílem této části je ověřit, že robotický manipulátor dokáže bezkolizně postavit paletu, tedy že během celého procesu nedojde k žádným kolizím mezi manipulátorem a krabicemi nebo mezi krabicemi samotnými.

Implementačně je simulace provedena až po nalezení konfigurací krabic a tedy vstupními daty je nalezená množina krabic v pevně daném pořadí, s konečnými pozicemi a orientacemi. Vzhledem k vysoké výpočetní náročnosti simulace zapříčiněné velkým počtem objektů navzájem v kontaktu je simulace provedena pouze pro několik vybraných konfigurací. I přesto simulace poskytne cenné informace pro další zdokonalování algoritmu a jeho případné použití v reálných průmyslových aplikacích.

4.1 Ověření stability

Pro účely simulačního ověření stability palety v průběhu paletizace byl implementován skript v MATLABu pro vytvoření Simulink modelu s využitím předdefinovaných bločků Simscape Multibody. Tento model zahrnuje všechny potřebné funkční bloky pro realistické modelování dynamického chování krabic při jejich umisťování. Simulační proces je kódem automatizován, což umožňuje snadné opakování a úpravy modelu dle potřeby.

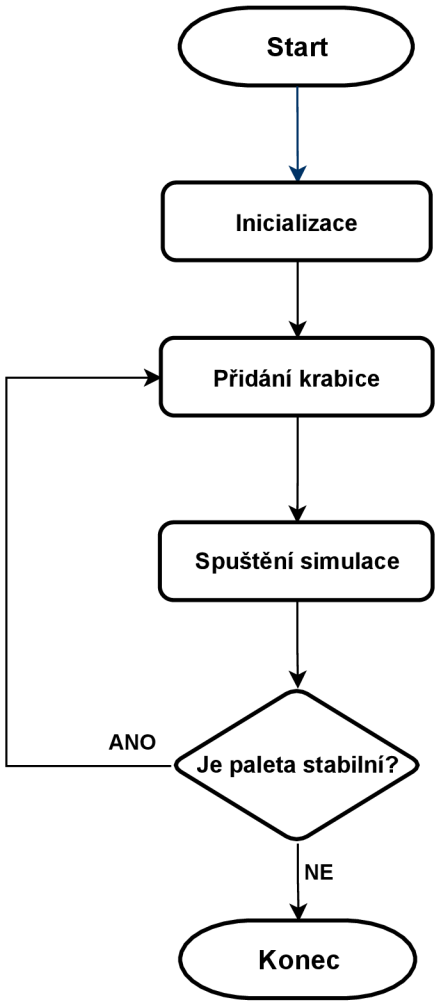
Po každém umístění krabice je simulace spuštěna a zkoumá se, zda paleta zůstane stabilní nebo zda dojde k výrazné změně polohy jedné či více krabic (Obr. 4.1).

V první fázi je vytvořen základní Simulink soubor, který bude sloužit jako výchozí bod pro přidávání dalších bloků a konfiguraci simulace.

Simulační prostředí je definováno přidáním nezbytných bloků (Obr. 4.2), jimiž jsou:

Blok **World Frame** představuje fixní referenční bod v simulaci, který slouží jako základní souřadnicový systém pro všechny ostatní objekty. V rámci simulace v prostředí Simscape Multibody tento blok definuje globální souřadnice, vůči kterým se měří polohy a orientace všech těles.

Blok **Solver Configuration** slouží pro nastavení parametrů, který Simscape používá



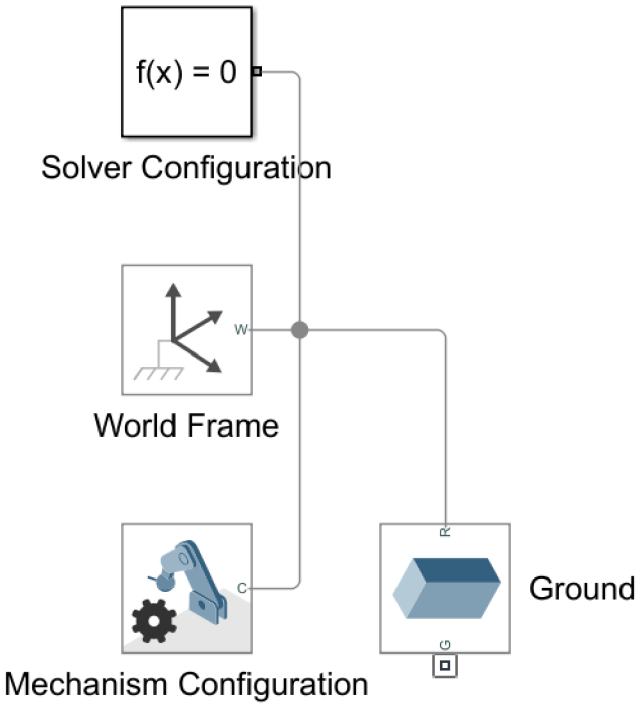
Obrázek 4.1: Flow chart implementace simulace v Simscape multibody

k numerickému řešení fyzikálních rovnic. Zde je určen typ řešiče, časové kroky a tolerance pro numerické chyby. Správná konfigurace řešiče je zásadní pro stabilní a přesnou simulaci, obzvlášť když simulace obsahuje početné interakce mezi tělesy, jako jsou kolize a vzájemné kontakty mezi krabicemi.

Blok **Mechanism Configuration** se využívá pro nastavení globálních parametrů mechanického systému. Tento funkční blok umožňuje definovat gravitační sílu a směr, poskytuje možnosti pro ladění parametrů, jako jsou simulace zpětné vazby a diagnostika. Tento blok tedy zajišťuje, že mechanický systém je správně nakonfigurován a všechny komponenty interagují dodržením fyzikálních zákonů.

Blok **Brick Solid** představuje základní stavební prvek, který při inicializaci simuluje podlahu, respektive paletu s definovanými fyzikálními vlastnostmi. Tento blok umožňuje specifikovat rozměry (délka, šířka, výška), materiálové vlastnosti (hustota, pružnost), a další parametry, jako je počáteční poloha a orientace. V simulaci se tento blok chová jako pevné těleso, které může interagovat s jinými tělesy prostřednictvím kontaktů a kolizí. V

následující fázi je tento blok použit pro reprezentaci jednotlivých krabic.



Obrázek 4.2: Inicializační bloky v Simscape Multibody

Pro správnou reprezentaci procesu paletizace byl implementován iterativní algoritmus. Tento algoritmus v každé iteraci přidává sadu bloků potřebných pro simulaci další krabice. Při každém opakování cyklu jsou do modelu přidány následující bloky:

Blok **Rigid Transform** slouží k definování pevné (rigidní) transformace mezi dvěma souřadnicovými systémy nebo referenčními rámci. Tento blok umožňuje specifikovat jak translaci (posun), tak rotaci (otáčení) mezi dvěma body, což je zásadní pro přesné umístění a orientaci jednotlivých objektů v systému. Při modelování polohy krabice na paletě blok Rigid Transform určuje přesnou pozici a orientaci krabice vzhledem k referenčnímu rámci. Rozdíl oproti implementaci optimalizačního algoritmu je ten, že zde je poloha krabice reprezentována polohou těžiště, nikoliv jednoho ze svých rohů, proto je zde nutná drobná úprava vstupních informací.

Blok **Spatial Contact Force** modeluje síly a momenty vznikající při kontaktu mezi dvěma tělesy. Tento blok je klíčový pro simulace, kde dochází ke kolizím nebo kontaktům mezi různými objekty. Spatial Contact Force blok vypočítává síly působící mezi kontaktními povrchy na základě jejich fyzikálních vlastností, jako je tuhost, tlumení, tření, a geometrie kontaktu. Tento blok umožňuje simulovat realistické chování kontaktů, včetně odrazu, sklouznutí a stlačení, což je nezbytné pro analýzu stability paletizovaných krabic. Pro správnou reprezentaci situace je nutné definovat kontakty mezi všemi předchozími krabicemi i zemí, což bohužel výrazně zvyšuje počet vypočítávaných interakcí. S přibývajícím počtem krabic na paletě počet kontaktních bloků velmi rychle narůstá a řídí se

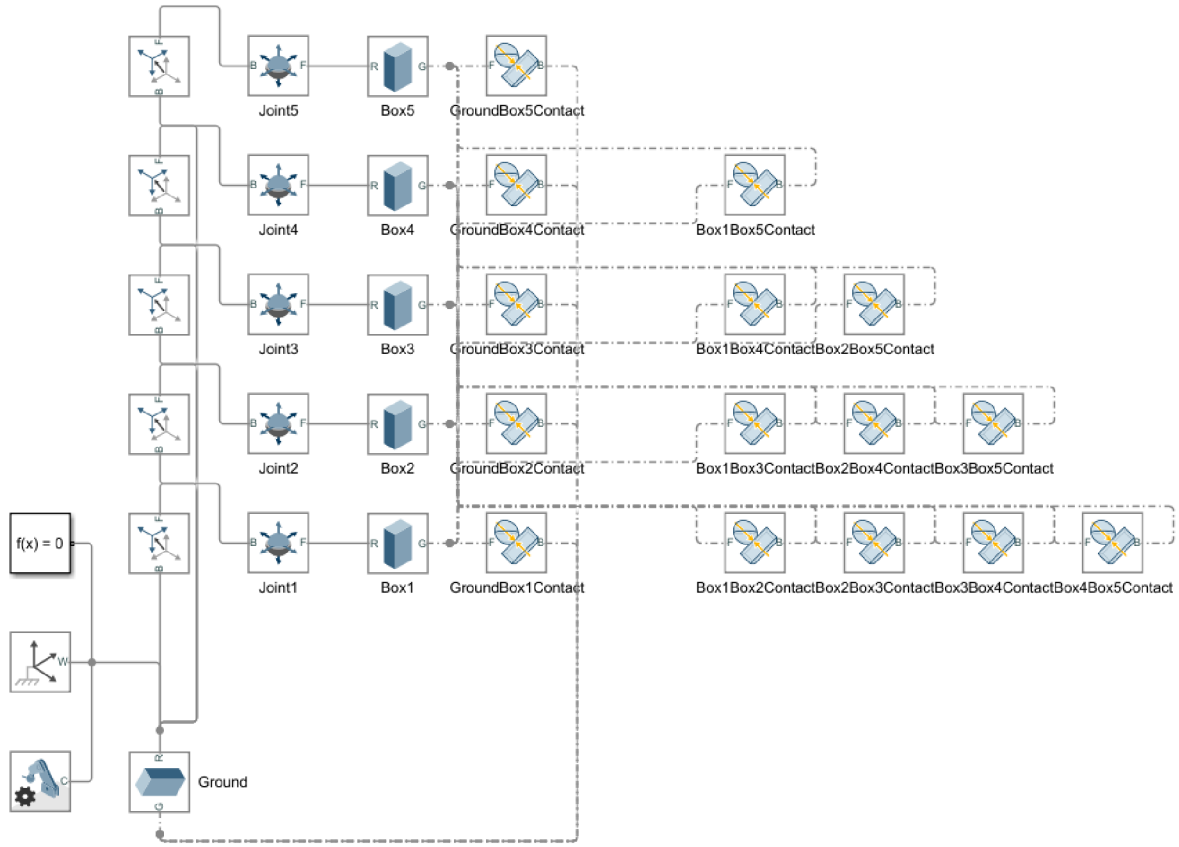
vztahem:

$$K = \frac{k(k+1)}{2} \tag{4.1}$$

Kde K je celkový počet kontaktních bloku a k počet krabic na paletě.

Blok **6DOF Joint** (Kloub s 6 stupni volnosti), jak již název napovídá, umožňuje volný pohyb tělesa ve všech šesti stupních volnosti, tedy tři translace a tři rotace. Reprezentuje modelování volného pohybu krabic nebo jiných těles, které nejsou pevně spojeny s jinými částmi systému. 6DOF Joint blok poskytuje flexibilitu při simulaci pohybů a interakcí, umožňující tělesům volně se pohybovat a reagovat na síly a momenty působící v systému. Celkový počet tohoto bloku se rovná počtu krabic na paletě.

Blok **Brick Solid** pro reprezentaci krabice již byl vysvětlen v předešlé části textu.



Obrázek 4.3: Blokové schéma pro prvních 5 krabic

Před samotným spuštěním simulace je zapotřebí zvolit parametry, aby byl výpočet dostatečně rychlý a současně byla zachována rozumná přesnost. Parametry byly zvoleny následovně:

- MaxStep - 0.1

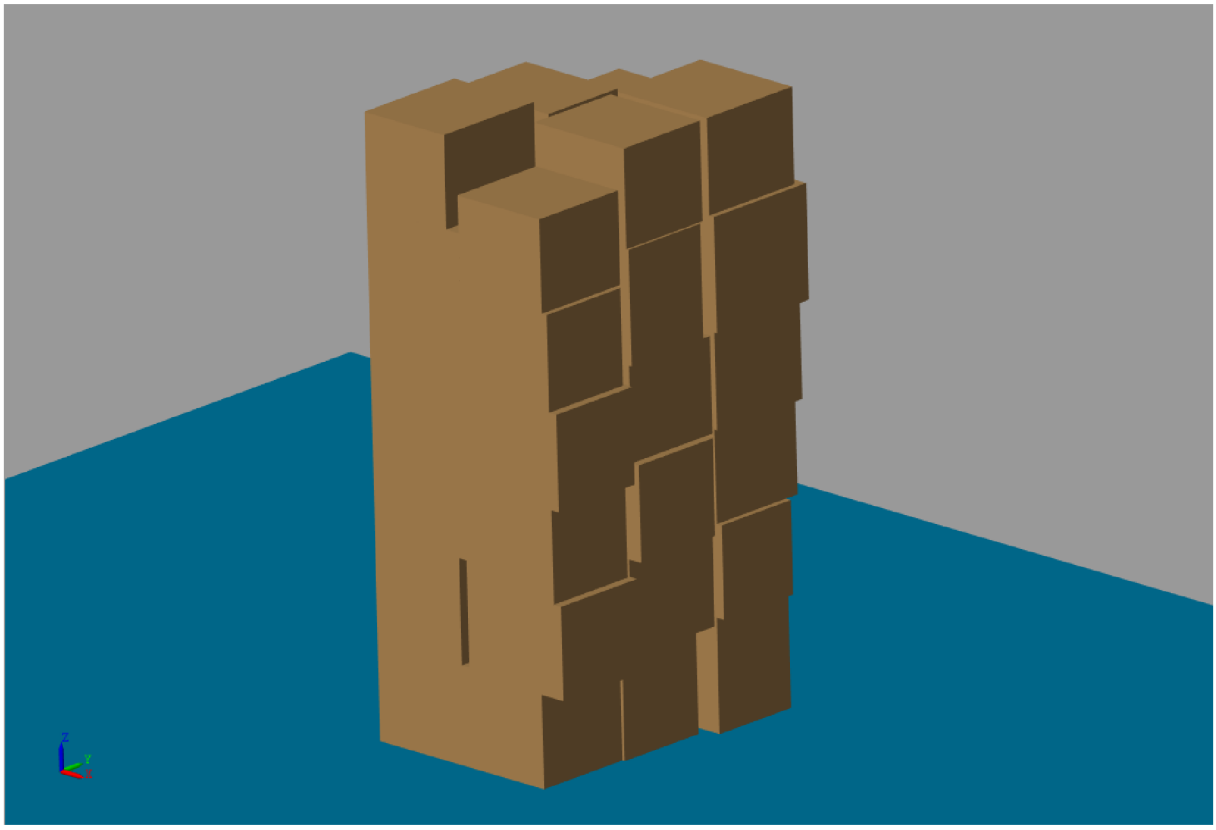
- MinStep - 0.0001
- StopTime - 0.5
- RelTol - 0.02

Parametr **MaxStep** určuje maximální délku časového kroku, který může solver použít během simulace. Hodnota 0.1 sekund byla zvolena jako kompromis mezi rychlostí a přesností simulace. Toto nastavení zajišťuje dostatečně jemné kroky, aby byly detekovány důležité interakce mezi krabicemi, aniž by simulace trvala příliš dlouho.

Parametr **MinStep** určuje minimální délku časového kroku, který může solver během simulace použít. Hodnota 0.0001 sekund zajistí, že solver může použít velmi malé kroky, pokud je to nutné pro přesné řešení rychle se měnících stavů systému. Toto je obzvláště důležité při simulaci kolizí a kontaktů. Nicméně i tak malý krok mnohdy nestačil a téměř vždy nebyla dodržena minimální chybová tolerance během výpočtu.

Parametr **StopTime** určuje čas, kdy simulace skončí. Hodnota 0.5 sekund byla zvolena tak, aby poskytla dostatečný čas na pozorování klíčových událostí a chování systému během paletizace, aniž by simulace byla zbytečně dlouhá.

Parametr **RelTol** určuje relativní toleranci, kterou solver používá k řízení přesnosti řešení. Hodnota 0.02 představuje kompromis mezi přesností a výpočetní náročností. Relativní tolerance 2 % je často dostatečně přesná pro inženýrské aplikace, přičemž stále umožňuje rychlý výpočet.



Obrázek 4.4: Vizualizace palety v Mechanics Explorer

Výsledky simulace pro vybrané konfigurace je shrnuta v tabulce 4.1. Bylo dosaženo poměrně pozitivních výsledku, jelikož většina ze zkoumaných palet dokázala vydržet až do konce paletizace bez převrhnutí.

Zřejmým nedostatkem použité metody je předpoklad krabic jako ideálních kvádrů. V reálných podmínkách má každá krabice určitou deformaci, což výrazně ovlivní jejich konečné uspořádání a v krajním případě může mít zásadní dopad na stabilitu palety jako celku.

Simulace palety byla také provedena pouze za statických podmínek a nebylo uvažováno působení vnějších sil a momentů, čímž nebylo zhodnoceno chování palety při jejím přemisťování.

Sada krabic	*Počet krabic*	*Stabilita*	*Kolaps*
10 - 20 $[cm]$	206	Stabilní	-
10 - 30 $[cm]$	169	Nestabilní	73. krabice
10 - 40 $[cm]$	78	Nestabilní	46. krabice
10 - 50 $[cm]$	57	Stabilní	-
20 - 30 $[cm]$	97	Stabilní	-
20 - 40 $[cm]$	55	Nestabilní	35. krabice
20 - 50 $[cm]$	35	Stabilní	-
30 - 40 $[cm]$	35	Stabilní	-
30 - 50 $[cm]$	21	Stabilní	-
Dataset z [12]	95	Stabilní	-
Dataset OR 1	206	Stabilní	-
Dataset OR 2	206	Stabilní	-

Tabulka 4.1: Tabulka výsledků simulace stability

4.2 Bezkolizní trajektorie

Implementace algoritmu bezkolizní trajektorie byla provedena opět v MATLABu s použitím Robotics System a Optimization toolboxů. Důvodem byla snadná návaznost a jednoduchost použití výsledků z algoritmu paletizace (kap. 3) a přítomnost rozsáhlé dokumentace.

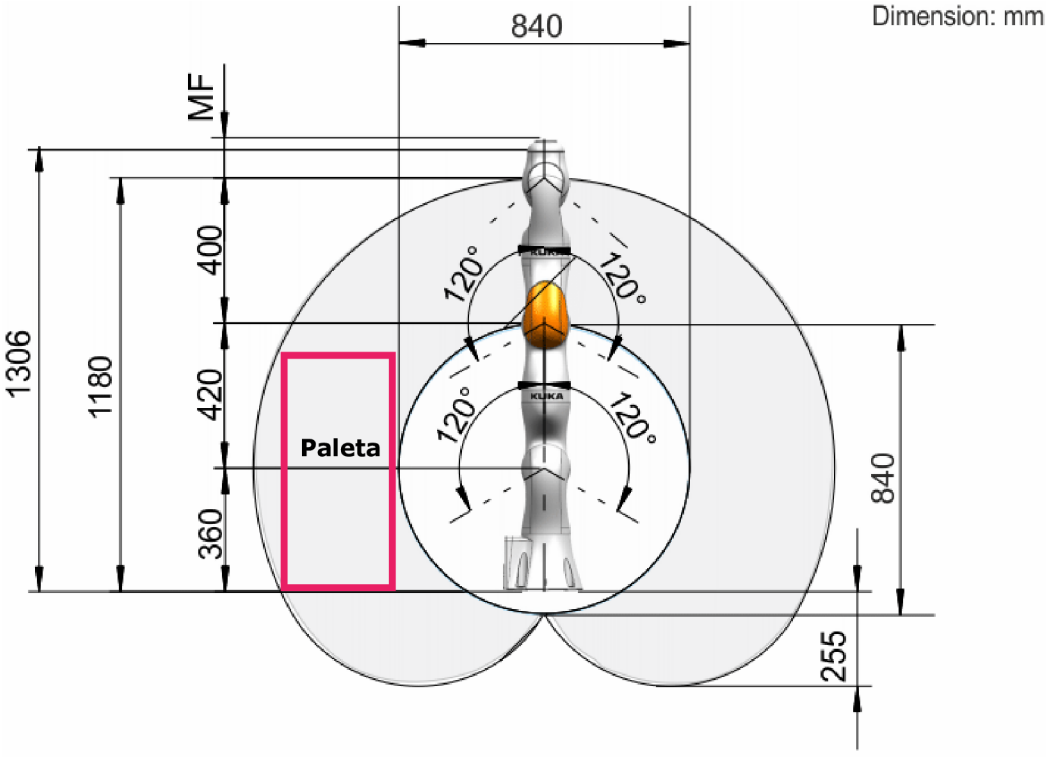
Algoritmus byl inspirován modelovým příkladem z knihovny Mathworks [26], zabývající se plánováním bezkolizní trajektorie manipulátorů kolem definovaných kolizních objektů. Implementace zde tedy nebude podrobně vysvětlena a budou zmíněny jen hlavní rozdíly a konkrétní úpravy související s problematikou této práce.

Výhodou použití Robotics toolboxu je, že obsahuje několik předdefinovaných modelů robotických manipulátorů a potřebných funkčních bloků, čímž je implementace výrazně ulehčena. V simulaci paletizace byl použit robotický manipulátor *KUKA LBR iiwa 14* (Obr. 4.5), jelikož poskytuje největší dosah mezi dostupnými roboty. Průmyslové roboty od společnosti KUKA se velmi často používají pro precizní manipulaci v široké škále aplikací, včetně výroby, montáže a logistiky. Jejich vysoká přesnost a flexibilita z nich činí dobré kandidáty pro různé úlohy, kde je vyžadována jemná manipulace s objekty.

Vzhledem k těmto vlastnostem není vyloučeno, že by podobný manipulátor našel uplatnění právě při paletizaci. Použití tohoto manipulátoru v simulaci poskytuje propojení simulace s reálnou aplikací.

Obrázek 4.5: Robotický manipulátor *Kuka LBR iiwa 14* [27]

Aby byl manipulátor schopný poskládat nalezené konfigurace krabic, bylo nutné rozměry krabic naškálovat tak, aby se plně poskládaná paleta vešla do jeho pracovního prostoru. (Obr. 4.6)



Obrázek 4.6: Schéma pracovního prostoru manipulátoru s vyznačeným prostorem palety [27]

Před zahájením hledání trajektorie bylo nejprve definováno simulační prostředí (viz Obr. 4.7). Manipulátor je inicializován v počáteční nesingulární orientaci s jeho základnou umístěnou na podlaze. V jeho pracovním prostoru je umístěn kolizní kvádr představující paletu, na kterou budou krabice pokládány. Aby se zabránilo pohybu manipulátoru do negativních Z souřadnic, bylo také nutné definovat podlahu, reprezentovanou tenkým kolizním válcem. Tento válec v obrázku vyznačuje pracovní pole manipulátoru.

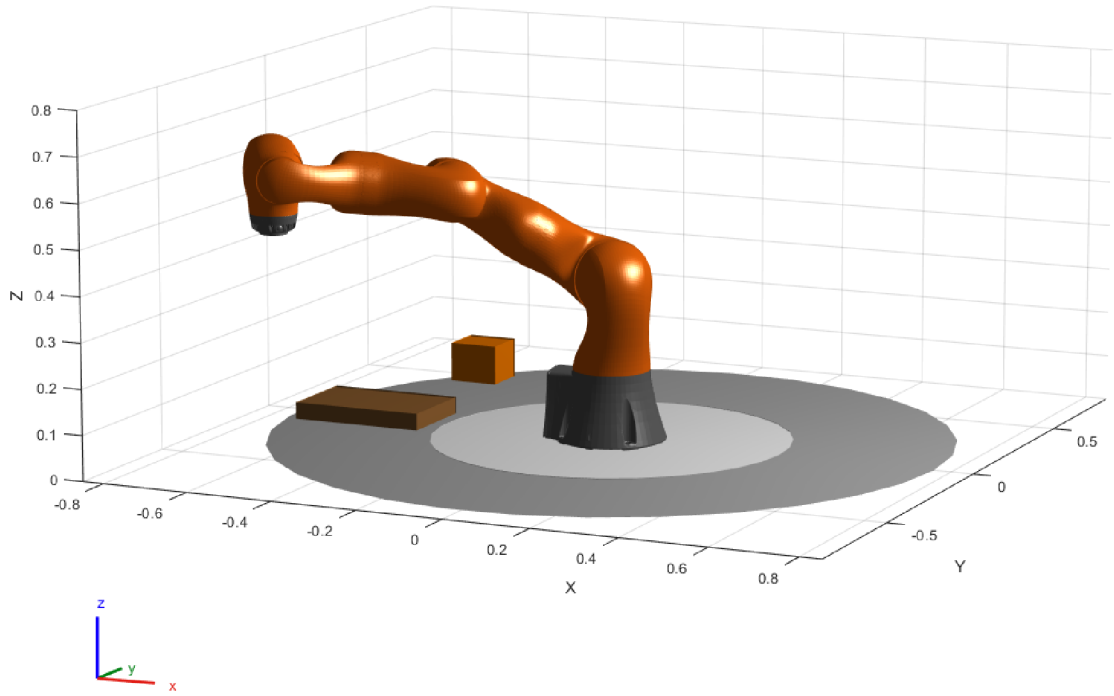
Pro jednoduchost je v každém kroku vygenerována krabice na předem definované pozici. Algortimus je implementován pomocí cyklu, kdy maniplátor postupně pokládá krabici na paletu jednu po druhé. V průběhu výpočtu tedy vykonává manipulátor následující cyklus operací:

1. Přesune se na místo, kde je vygenerována krabice, a připraví se k jejímu uchopení ze shora. Je nutné poznamenat, že zvolená fixní poloha krabice představuje polohu jejího těžiště. To znamená, že pro každou krabici se bude poloha koncového efektoru při uchopení lišit vertikální souřadnicí. (Obr. 4.8)

2. Uchopí krabici, což je implementováno přidáním kolizního objektu do rigidního stromu robota. Prakticky to znamená, že krabice je považována za koncový efektor manipulátoru.

3. Přemístí krabici na paletu. Jakmile je krabice na požadované pozici na paletě, odstraní se objekt krabice z rigidního stromu manipulátoru, jenž se posléze začne připravovat na uchopení další krabice.

Tento sled událostí pokračuje, dokud nejsou všechny krabice umístěny na paletu, nebo dokud je algoritmus schopen najít bezkolizní trajektorii.



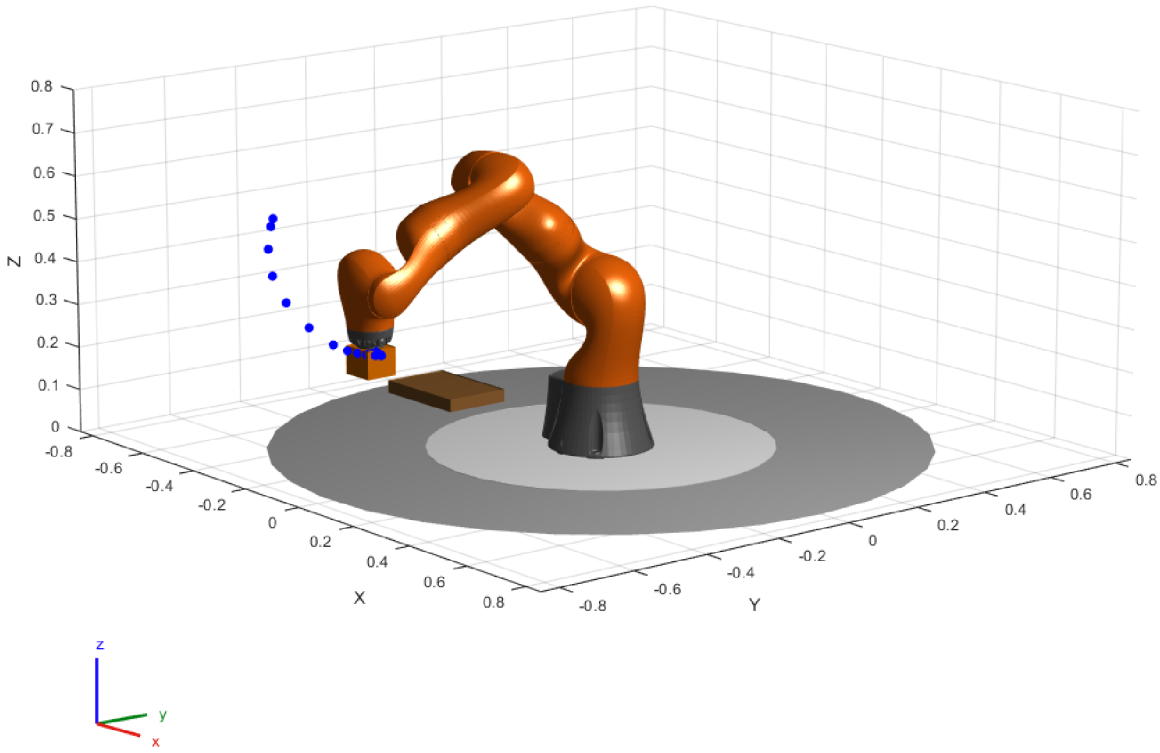
Obrázek 4.7: Inicializované prostředí simulace robotického manipulátoru

Samotné plánování trajektorie robota je realizováno pomocí nelineárního prediktivního řízení (NLMPC) a iterativního algoritmu pro generování trajektorie. Nejprve se vypočítá aktuální konfigurace kloubů robota pomocí inverzní kinematiky. Poté jsou definovány počáteční a cílové pozice robota, přičemž cílová pozice představuje pozici pro koncový efektor.

Je vytvořen objekt pro nelineární prediktivní řízení (NMPC), který definuje požadavky na sledování trajektorie. Plánování trajektorie probíhá iterativně. V každém kroku algoritmus optimalizuje řídicí stavy tak, aby robot dosáhl cílové pozice. To zahrnuje optimalizaci a aktualizaci stavu robota a časového kroku. Po každém kroku je kontrolováno, zda byla dosažena cílová pozice. Pokud ano, algoritmus končí.

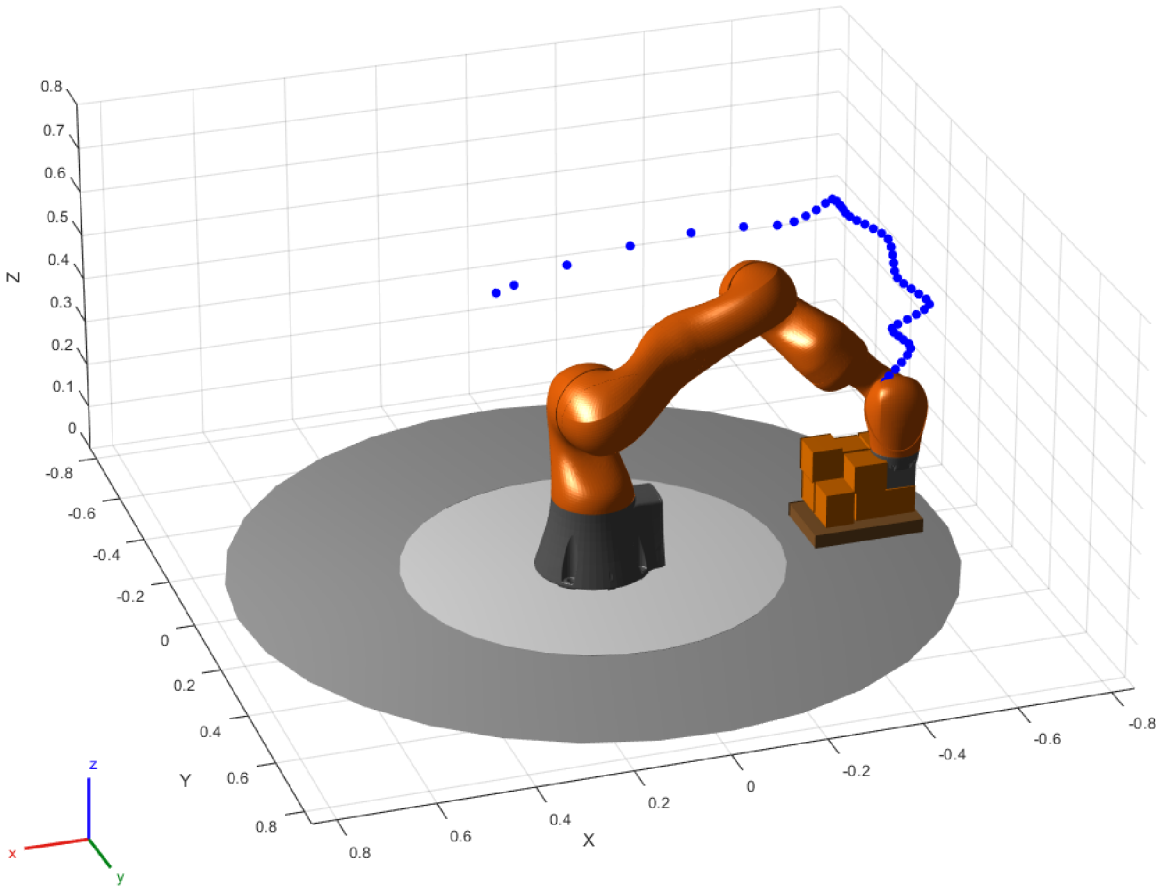
Algoritmus dosáhl poměrně vysoké úspěšnosti, jelikož téměř vždy dokázal najít bezkolizní trajektorii. Vysoká úspěšnost se ovšem dá vysvětlit tím , že při výpočtu zohledňoval pouze kolize koncového efektoru s ostatními objekty a ignoroval možné kolize s jinými rameny manipulátoru.

Nižší úspěšnost se objevila převážně u datasetů s širokým intervalem rozměrů krabic (např. 10-50 cm, 20-50 cm). To bylo často způsobeno snahou umístit malou krabici na pozici obklopenou většími. Dalším velkým nedostatkem byla rostoucí výpočetní náročnost při zvyšujícím naplnění palety. Více kolizních objektů vedlo k prodloužení výpočtu, což



Obrázek 4.8: Trajektorie přesunu manipulátoru pro uchopení krabice

omezilo opakovatelnost experimentů.



Obrázek 4.9: Trajektorie přesunu manipulátoru při umístění krabice na paletu

5 Závěr

V rámci této diplomové práce byla zkoumána problematika algoritmů řešení Problému skládání palet distributora (DPLP). Bylo zjištěno, že literatura zabývající se případem znalosti velmi omezeného počtu krabic předem je velmi omezená, což ztížilo přejímání metod z předešlých studií věnujících se DPLP. To vedlo k potřebě vyvinout částečně vlastní postup řešení.

Hlavní část práce se soustředila na navržení algoritmu prohledávání stavového prostoru do předem definované hloubky za účelem výběru následujícího stavu palety v závislosti na ohodnocení pomocí navržené cenové funkce. Jejím smyslem bylo přiřadit numerickou hodnotu přechodu z jednoho stavu do druhého, přičemž finální ohodnocení jednotlivých stavů mělo podobu sumy cen všech předcházejících přechodů. Cenová funkce byla reprezentována jako vážená suma relevantních parametrů týkajících se vyplnění prostoru, umístění krabic a stability.

Tento algoritmus byl následně implementován pomocí MATLAB skriptu, který zahrnoval také vizualizaci palety pro ověření správnosti jeho dílčích funkcí. Během experimentů však bylo odhaleno několik nedostatků. S rostoucí velikostí zásobníku exponenciálně rostla výpočetní náročnost kvůli narůstajícímu větvení stavového stromu a přibývajícím možným umístěním krabic, což výrazně omezilo rozsah experimentace. Dosažení optimálního výsledku nebylo možné zajistit vzhledem k nepředvídatelné sekvenci krabic v zásobníku. Dalším problémem bylo volení vah jednotlivých parametrů cenové funkce, což bylo komplikované kvůli velké variabilitě vstupních sad krabic.

Výsledky experimentů hledání konfigurace palety nebyly příliš uspokojivé, neboť jejich vyplnění dosahovaly v průměru jen kolem 70 %, což je podstatně méně než bylo dosaženo v jiných studiích. Důvodem je pravděpodobně nedostatečná velikost zásobníku, jelikož s vyšším výběrem krabic v dílčích krocích roste šance na přítomnost krabice vedoucí k lepšímu vyplnění prostoru. Lepších výsledků bylo dosaženo pro datové sady s užším rozmezím rozměrů krabic (např. 30-40 cm, 10-20 cm), nicméně pro tyto rozmezí docházelo mezi krabicemi k mnohem menší tvorbě vazeb. Dalším důležitým faktorem byl poměr rozměrů krabic k paletizačnímu prostoru. Pro dosažení lepších výsledků je nutné, aby na každou osu bylo možné umístit alespoň tři krabice.

Jedním z možných vylepšení metody by mohlo být přidání odkládacího zásobníku. Tento zásobník by umožnil vzít v úvahu situace, kdy každá z nabízených krabic způsobí výraznou odchylku některých kritérií, například relativního vyplnění prostoru. Dalším vylepšením by mohla být integrace funkcionality pro dodatečné posouvání krabic tam, kde je to možné, což by mohlo vést k lepší eliminaci nevyužitého prostoru.

Následující část práce se věnovala implementaci skriptu pro vytvoření blokového schématu v Simscape Multibody za účelem ověření stability palety během paletizace. Během této fáze byly identifikovány nedostatky spojené s volbou simulačních parametrů. Často docházelo k velmi dlouhému trvání výpočtů nebo dokonce k nekonvergenci řešení,

zejména při práci s velkým množstvím krabic. Navzdory těmto problémům poskytla tato část poměrně uspokojivé výsledky, jelikož pouze 3 z 12 zkoumaných konfigurací skončily kolapsem. Stabilita krabic byla ovšem testována jen ve statických podmínkách. Dalším krokem simulačního ověření by tedy mohlo být zkoumání, jak se paleta bude chovat při převozu a jiné manipulaci.

Poslední část práce se zaměřila na algoritmus hledání bezkolizní trajektorie pro robotický manipulátor. I přes časté nalezení existujících bezkolizních trajektorií, tento algoritmus měl omezenou schopnost, neboť při výpočtu zohledňoval pouze kolize koncového efektoru s ostatními objekty a ignoroval možné kolize s ostatními rameny manipulátoru.

Seznam použitých zkratek

PLP	Pallet Loading Problem
DPLP	Distributors Pallet Loading Problem
3DBPP	3D Bin Packing Problem
CLP	Container Loading Problem
KP	Knapsack Problem
LCGA	Layer based Column Generation Algorithm
OR	Operations Research
SA	Simulated Annealing
NLMPC	Non-Linear Model Predictive Control

Bibliografie

[1] VARGAS-OSORIO Saúl; Zúñiga, Catya. A Literature Review on the Pallet Loading Problem. *Lámpsakos, Issue 15, p69*. 2016, roč. 20. Dostupné z DOI: 10.21501/21454086.1790.

[2] A. POPPLE Columbia Machine, Inc. *Science of Palletizing* [online]. [cit. 2024-05-21]. Dostupné z: https://palletizing.com/wp-content/uploads/sites/3/2017/11/Science-of-Palletizing.pdf.

[3] MATHEWS, G. B. On the Partition of Numbers. *Proceedings of the London Mathematical Society*. 1896, roč. s1-28, č. 1, s. 486–490. Dostupné z DOI: https://doi.org/10.1112/plms/s1-28.1.486.

[4] PISINGER, D. *Algorithms for Knapsack Problems*. 1995. Dis. pr. University of Copenhagen, Dept. of Computer Science.

[5] CRAINIC, Teodor Gabriel; PERBOLI, Guido; TADEI, Roberto. Extreme Point-Based Heuristics for Three-Dimensional Bin Packing. *INFORMS Journal on Computing*. 2008, roč. 20, s. 368–384. Dostupné z DOI: 10.1287/ijoc.1070.0250.

[6] HO, Ziao-Fung; LEE, Lai Soon; ABDUL MAJID, Zanariah; SEOW, Hsin-Vonn. An Improved GRM(OD) Heuristic for Container Loading Problem. *AIP Conference Proceedings*. 2013, roč. 1557, s. 439–443. Dostupné z DOI: 10.1063/1.4823952.

[7] SHAMSUDIN, Maryam; ISMAIL, Fatimah Sham; SELAMAT, Hazlina; KHAMIS, Nurulaqilla. Analysis of Space Optimization of Three-Dimensional Container Loading Problem. In: 2022, s. 451–464. Dostupné z DOI: 10.1007/978-981-19-3923-5_39.

[8] HUANG, Wenqi; HE, Kun. An Efficient Algorithm for Solving the Container Loading Problem. In: 2007, s. 396–407. Dostupné z DOI: 10.1007/978-3-540-74450-4_36.

[9] SILVA, Elsa; OLIVEIRA, José; WÄSCHER, Gerhard. The pallet loading problem: A review of solution methods and computational experiments. *International Transactions in Operational Research*. 2014, roč. 23. Dostupné z DOI: 10.1111/itor.12099.

[10] SCHEITHAUER, Guntram; TERNO, Johannes. A Heuristic Approach for Solving the Multi-Pallet Packing Problem. 1999.

[11] MARTELLO, Silvano; PISINGER, David; VIGO, Daniele. The Three-Dimensional Bin Packing Problem. *Operations Research*. 1998, roč. 48. Dostupné z DOI: 10.1287/opre.48.2.256.12386.

[12] ABDOU, George; ELMASRY, M.I. 3D random stacking of weakly heterogeneous palletization problems. *International Journal of Production Research*. 2010, roč. 37, s. 1505–1524. Dostupné z DOI: 10.1080/002075499191102.

[13] BISCHOFF, E.E.; RATCLIFF, M.S.W. Issues in the development of approaches to container loading. *Omega*. 1995, roč. 23, č. 4, s. 377–390. Dostupné z DOI: https://doi.org/10.1016/0305-0483(95)00015-G.

[14] PISINGER, David. Heuristics for the container loading problem. *European Journal of Operational Research*. 2002, roč. 141, č. 2, s. 382–392. Dostupné z DOI: https://doi.org/10.1016/S0377-2217(02)00132-7.

[15] BORTFELDT, Andreas; MACK, Daniel. A heuristic for the three-dimensional strip packing problem. *European Journal of Operational Research*. 2007, roč. 183, č. 3, s. 1267–1279. Dostupné z DOI: https://doi.org/10.1016/j.ejor.2005.07.031.

[16] HOPPER, E; TURTON, B.C.H. An empirical investigation of meta-heuristic and heuristic algorithms for a 2D packing problem. *European Journal of Operational Research*. 2001, roč. 128, č. 1, s. 34–57. Dostupné z DOI: https://doi.org/10.1016/S0377-2217(99)00357-4.

[17] GZARA, Fatma; ELHEDHLI, Samir; YILDIZ, Burak C. The Pallet Loading Problem: Three-dimensional bin packing with practical constraints. *European Journal of Operational Research*. 2020, roč. 287, č. 3, s. 1062–1074. Dostupné z DOI: 10.1016/j.ejor.2020.04.05.

[18] FAROE, Oluf; PISINGER, David; ZACHARIASEN, Martin. Guided Local Search for the Three-Dimensional Bin Packing Problem. *INFORMS Journal on Computing*. 2000, roč. 15. Dostupné z DOI: 10.1287/ijoc.15.3.267.16080.

[19] HU, Yuchuan; ZUO, Yi; SUN, Zhuo. Combination of Simulated Annealing Algorithm and Minimum Horizontal Line Algorithm to Solve Two-Dimensional Pallet Loading Problem. In: *2022 Winter Simulation Conference (WSC)*. 2022, s. 1956–1966. Dostupné z DOI: 10.1109/WSC57314.2022.10015349.

[20] SCHUSTER, M.; BORMANN, Richard; STEIDL, D.; REYNOLDS-HAERTLE, S.; STILMAN, Mike. Stable stacking for the distributor's pallet packing problem. In: 2010, s. 3646–3651. Dostupné z DOI: 10.1109/IROS.2010.5650217.

[21] ALVAREZ-VALDES, R.; PARREÑO, F.; TAMARIT, J. M. A tabu search algorithm for the pallet loading problem. *OR Spectrum*. 2005, roč. 27. Dostupné z DOI: 10.1007/s00291-004-0183-5.

[22] SINGH, Manjeet; ALMASARWAH, Najat; SÜER, Gürsel. A Two-Phase Algorithm to Solve a 3-Dimensional Pallet Loading Problem. *Procedia Manufacturing*. 2019, roč. 39, s. 1474–1481. Dostupné z DOI: https://doi.org/10.1016/j.promfg.2020.01.301.

[23] CZECH, EPAL National Comitee; Z.S., Slovak republics. *EUROPALETA EPAL* [online]. [cit. 2024-05-22]. Dostupné z: https://cz.epal-pallets.org/nosice-nakladu/europaleta-epal#c4208.

[24] JOZEFOWSKA, Joanna; PAWLAK, Grzegorz; PESCH, Erwin; MORZE, Michał; KOWALSKI, Dawid. Fast truck-packing of 3D boxes. *Engineering Management in Production and Services*. 2018, roč. 10, s. 29–40. Dostupné z DOI: 10.2478/emj-2018-0009.

[25] BEASLEY, J E. *OR Library* [online]. [cit. 2024-05-21]. Dostupné z: http://people.brunel.ac.uk/~mastjjb/jeb/info.html.

[26] THE MATHWORKS, Inc. *Generate Code to Plan and Execute Collision-Free Trajectories Using KINOVA Gen3 Manipulator* [online]. [cit. 2024-05-21]. Dostupné z: https://www.mathworks.com/help/mpc/ug/generate-code-for-manipulator-trajectory-planning-and-execution.html.

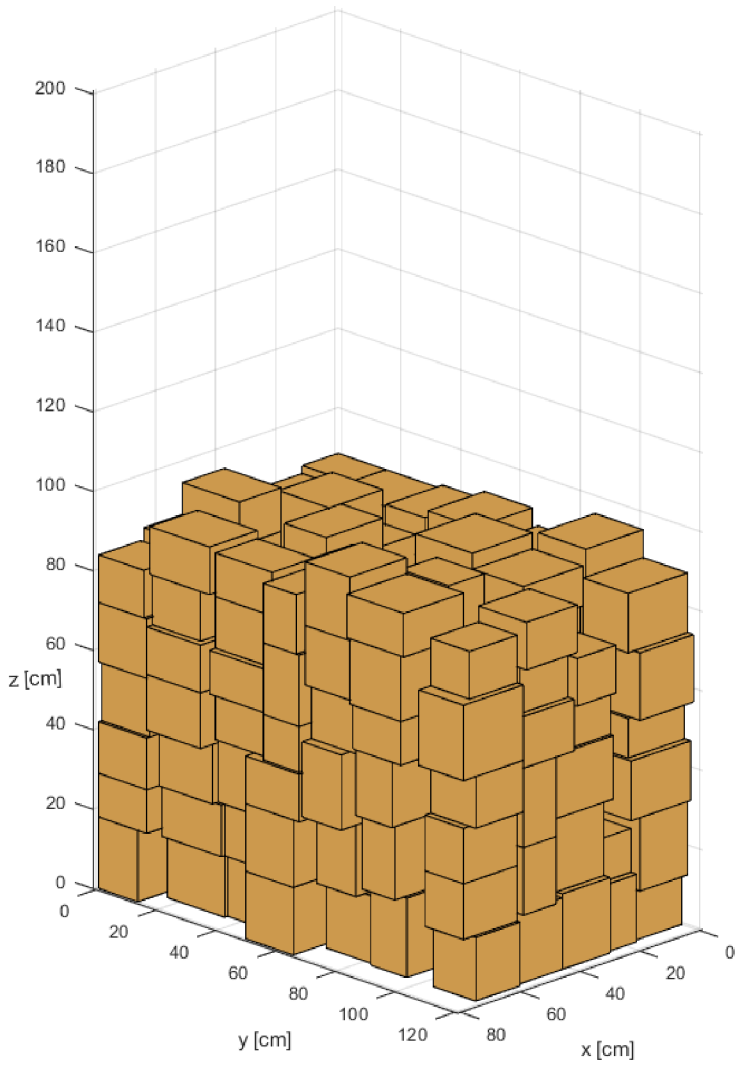
[27] AG, KUKA. *LBR iiwa* [online]. [cit. 2024-05-22]. Dostupné z: https://www.kuka.com/cs-cz/produkty,-slu%C5%BEby/robotick%C3%A9-syst%C3%A9my/pr%C5%AFmyslov%C3%A9-roboty/lbr%C2%A0iiwa.

Seznam příloh

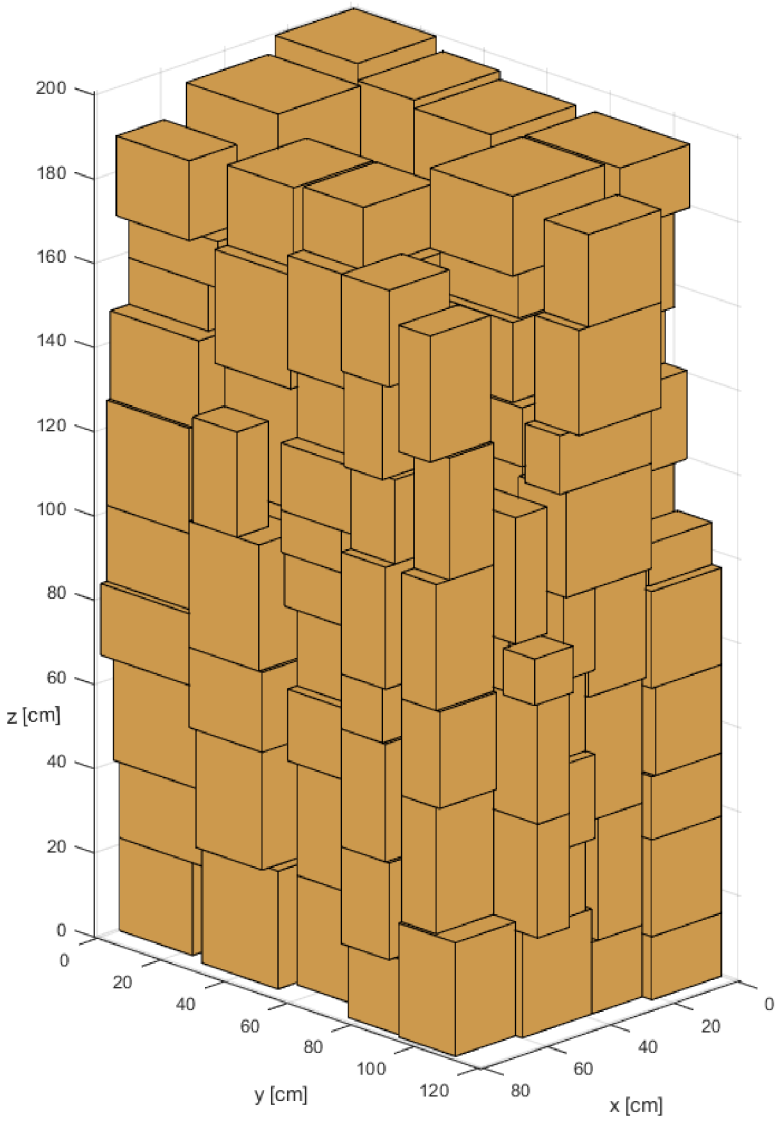
A. Externí přílohy

K práci je přiložena složka, obsahující zdrojové kódy pro hlavní algoritmus prohledávání, simulaci v Simcape Multibody a hledání bezkolizní trajektorie manipulátoru navržené v rámci této diplomové práce.

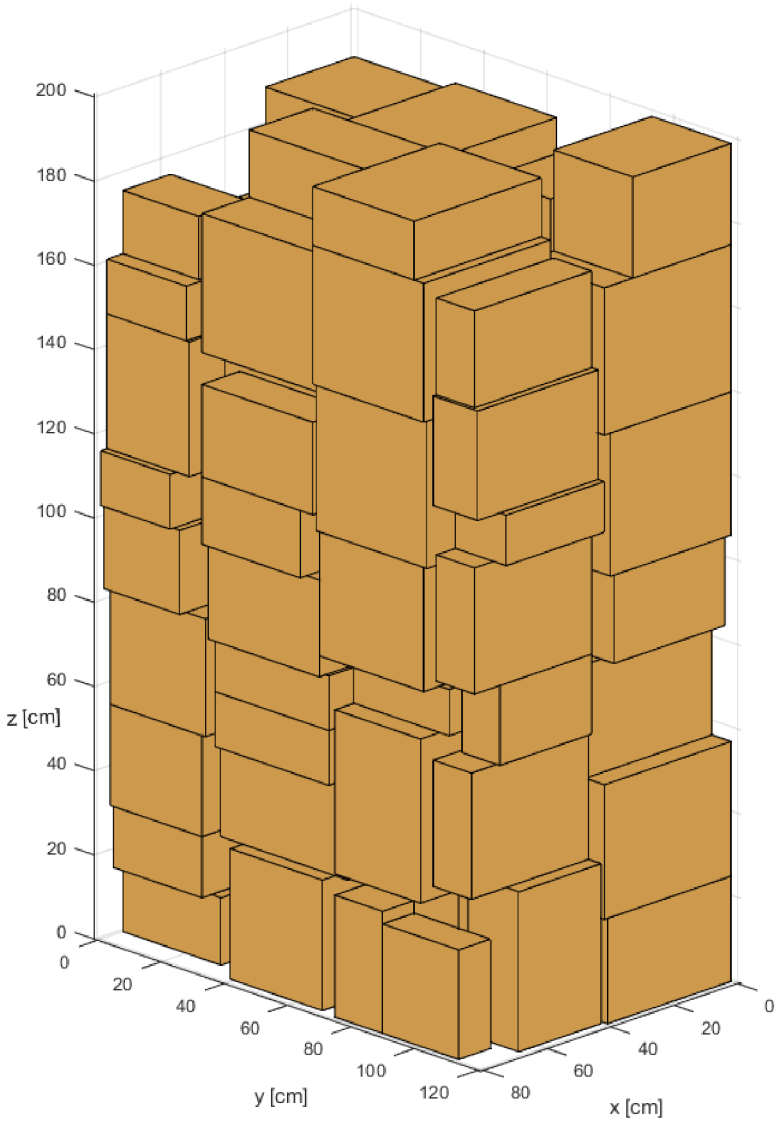
B. Výsledné palety



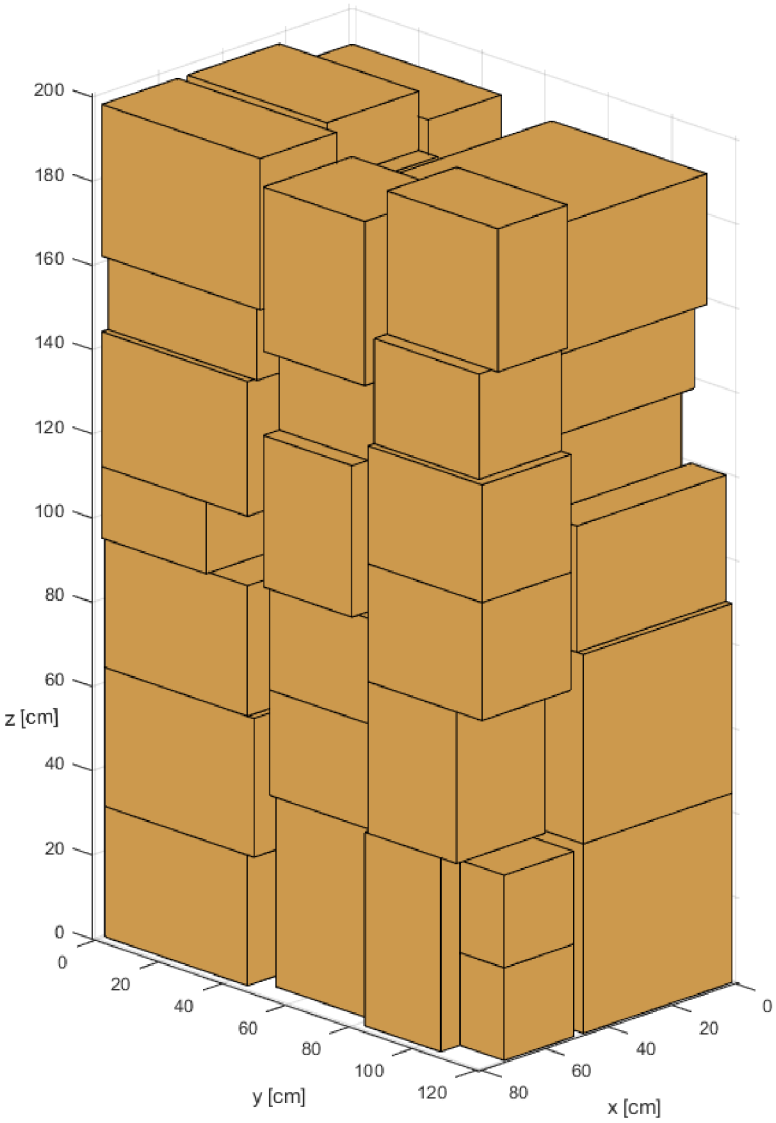
Výsledná paleta krabic rozměrů 10-20 cm, paletizace ukončená podmínkou nosnosti



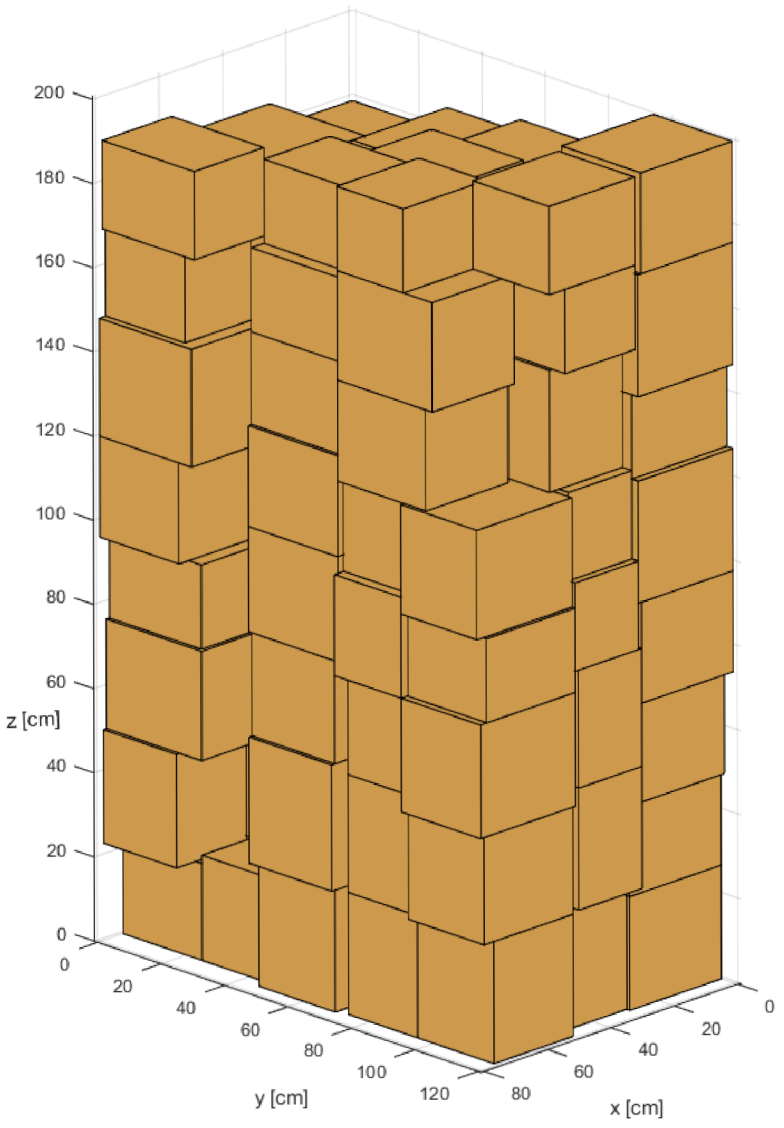
Výsledná paleta krabic rozměrů 10 - 30 cm



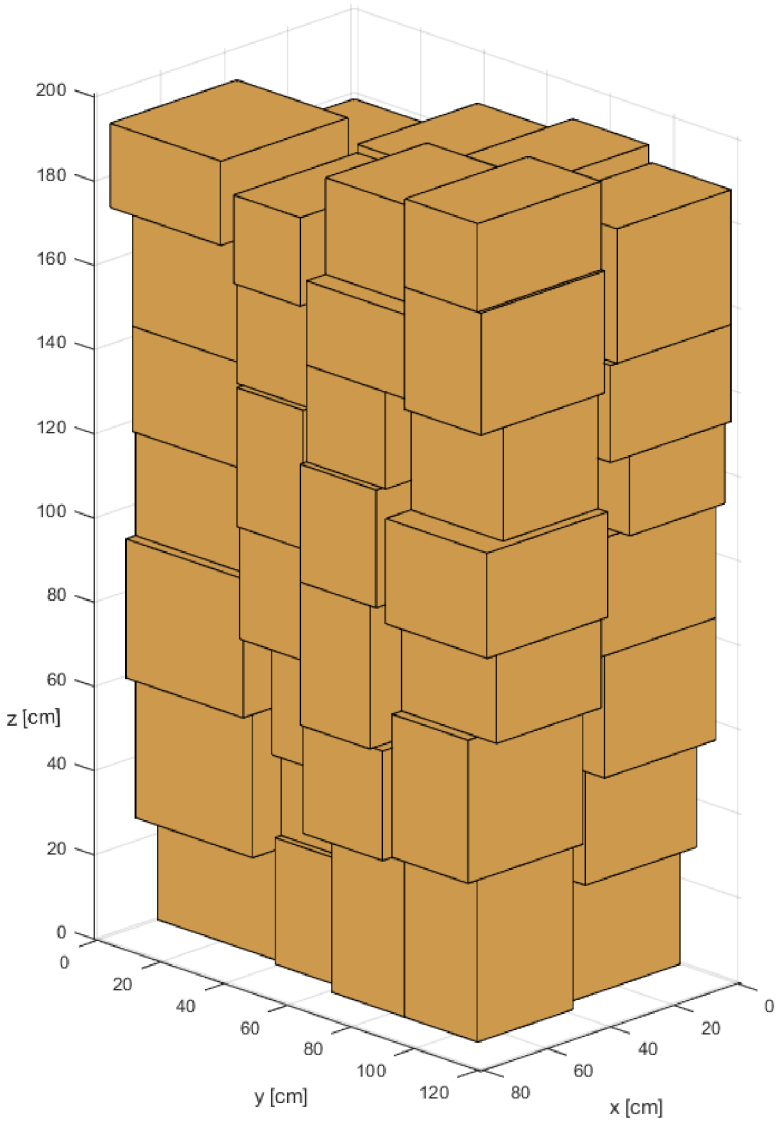
Výsledná paleta krabic rozměrů 10 - 40 cm



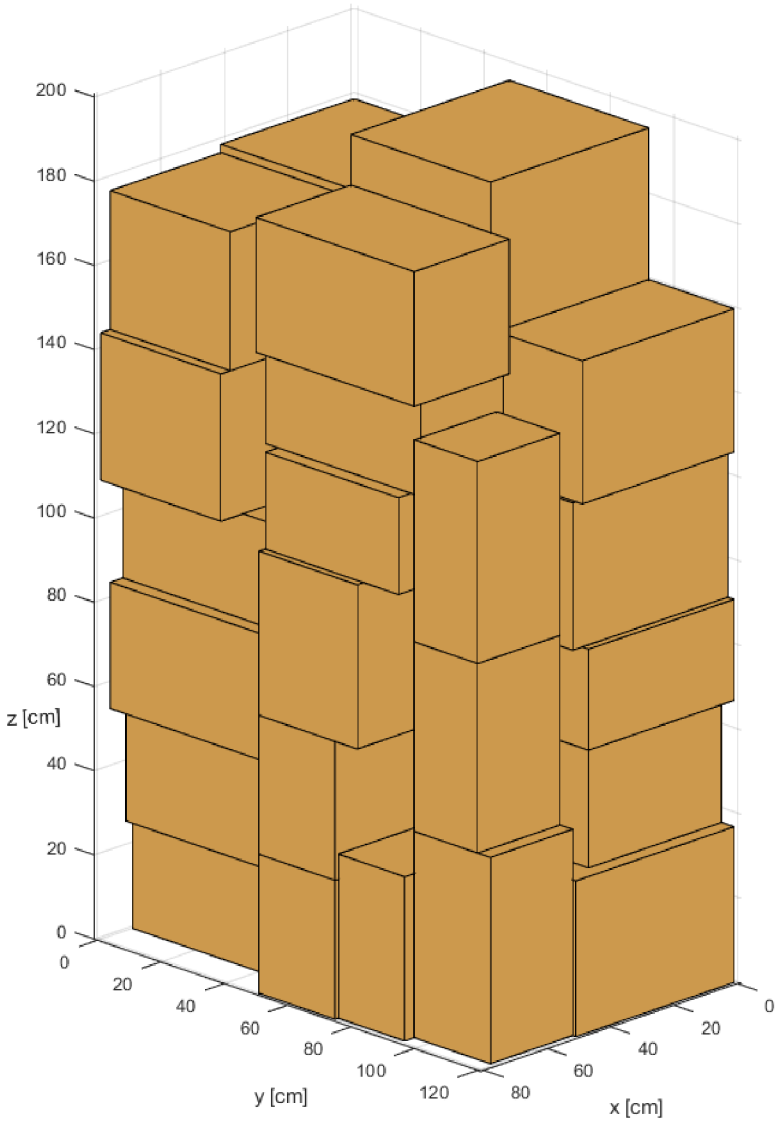
Výsledná paleta krabic rozměrů 10 - 50 cm



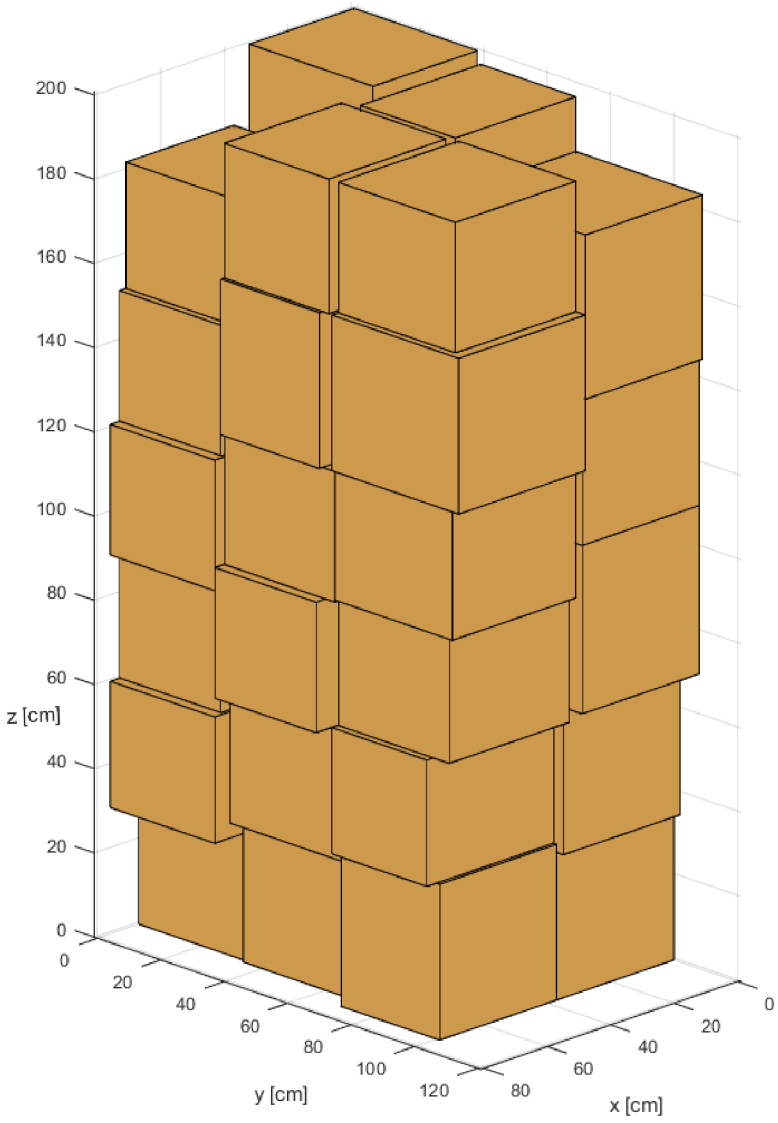
Výsledná paleta krabic rozměrů 20 - 30 cm



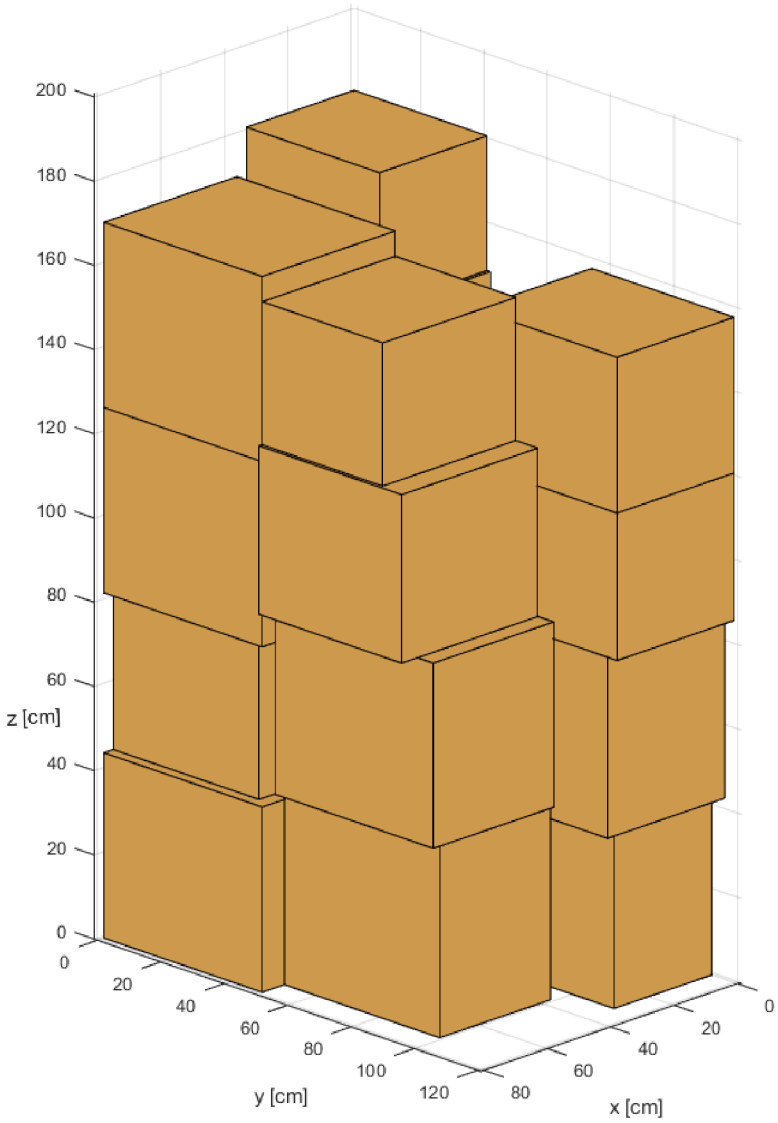
Výsledná paleta krabic rozměrů 20 - 40 cm



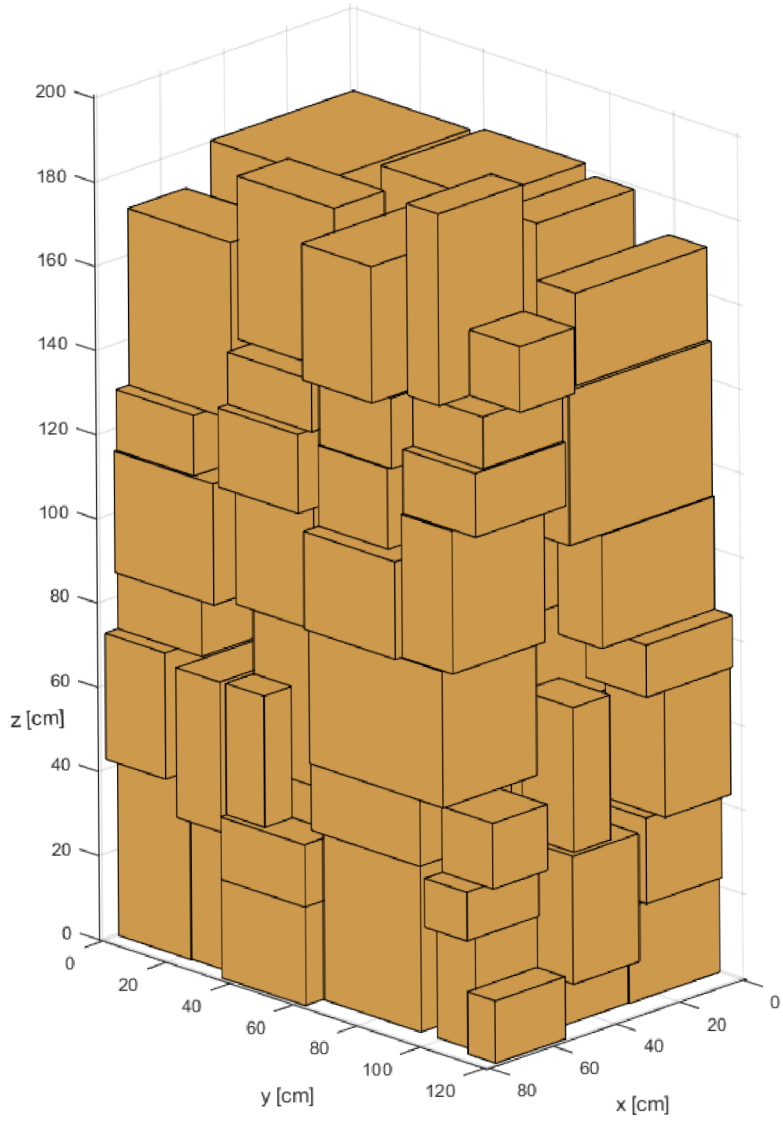
Výsledná paleta krabic rozměrů 20 - 50 cm



Výsledná paleta krabic rozměrů 30 - 40 cm



Výsledná paleta krabic rozměrů 30 - 50 cm



Výsledná paleta krabic datasetu OR 1 [25]