УНИВЕРЗИТЕТ „ГОЦЕ ДЕЛЧЕВ" – ШТИП

ФАКУЛТЕТ ЗА ИНФОРМАТИКА

Катедра: Роботика и интелигентни системи

Стојанче Панов

ПЛАНИРАЊЕ НА ОПТИМАЛНА ТРАЕКТОРИЈА НА МОБИЛНИ РОБОТИ КОРИСТЕЈЌИ МУЗИЧКИ ИНСПИРИРАНИ АЛГОРИТАМИ

МАГИСТЕРСКИ ТРУД

Штип, септември, 2013 година

Комисија за оценка

Ментор:	Сашо Коцески Доцент, Универзитет ,,Гоце Делчев” - Штип, Факултет за Информатика
Член	Владо Гичев Вонреден професор, Универзитет ,,Гоце Делчев” - Штип, Факултет за Информатика
Член	Цвета Мартиновска Вонреден професор, Универзитет ,,Гоце Делчев” - Штип, Факултет за Информатика

Комисија за одбрана

Претседател:	Сашо Коцески Доцент, Универзитет ,,Гоце Делчев” - Штип, Факултет за Информатика
Член	Горан Девеџиќ Редовен професор, Универзитет во Крагуевац, Машински факултет во Крагуевац
Член	Цвета Мартиновска Вонреден професор, Универзитет ,,Гоце Делчев” - Штип, Факултет за Информатика

Научно поле:	Роботика и интелигентни системи
Научна област:	Оптимизациони алгоритами

Датум на одбрана: 9.9.2013

Датум на промоција: 9.9.2013

Благодарност

Би сакал да изразам особена благодарност до мојот ментор што ми ја даде можноста да работам на елабориранот алгоритам обработен во рамките на оваа магистерска теза. Им се заблагодарувам на професорите од лабораторијата при Универзитетот во Аквила, Италија, за помошта во експерименталната евалуација на новиот алгоритам.

Објавени трудови:

Panov, S., & Koceski, S. (2013). Harmony Search Based Algorithm for Mobile Robot Global Path Planning. In Embedded Computing (MECO), 2013 Mediterranean Conference on (pp. 168-171). IEEE.

Panov, S., & Koceski, S. (2013). Deterministic and metaheuristic approaches to solving Kakuro puzzles. In Embedded Computing (MECO), 2013 Mediterranean Conference on (pp. 227-230). IEEE.

Panov, S., & Koceski, S. (2013). Global path planning algorithm for mobile robots. In *ICEST*, (in print).

Panov, S., & Koceski, S. (2013). Solving Kakuro puzzle – comparison of deterministic approaches. In *ICEST*, (in print).

Panov, S., & Koceski, S. (2013). Solving Kakuro Puzzle using Self Adapting Harmony Search Metaheuristic Algorithm. In *International Journal of Engineering Practical Research*, (accepted).

Panov, S., & Koceski, S. (2013). Development of a novel Quad Harmony Search Algorithm for grid-based path finding. Advances in Electrical and Computer Engineering (submitted).

Panov, S., & Koceska, N. (2014). Global Path Planning in Grid-Based Environments Using Novel Metaheuristic Algorithm. In ICT Innovations 2013 (pp. 121-130). Springer International Publishing.

ПЛАНИРАЊЕ НА ОПТИМАЛНА ТРАЕКТОРИЈА НА МОБИЛНИ РОБОТИ КОРИСТЕЈЌИ МУЗИЧКИ ИНСПИРИРАНИ АЛГОРИТАМИ

Апстракт – Воведен е нов пристап кон проблемот на наоѓање на пат кај регуларно структурирани околини. Методот го користи Harmony Search (HS) алгоритамот, метаевристички алгоритам, за добивање на оптималното решение, додека алгоритамот на квад-стебла обезбеди голема можност за намалување на времето потребно за пресметка на решението. Овој Quad HS алгоритам ја користи декомпозицијата на слободниот простор со Quad-tree во околината за да ги означи слободните области и да ги третира како еден јазол, што во голема мера го подобрува извршувањето. Резултатите од Quad HS алгоритамот се споредени со други метаевристички алгоритами, т.е. колонија на мравки и генетскиот алгоритам и се покажа дека дава најдобри резултати во контекст на време и добивање на оптимален пат.

Клучни зборови - Евристички алгоритами, вештачка интелигенција, пресметковна интелигенција, оптимизација, наоѓање на пат.

OPTIMAL PATH PLANNING FOR MOBILE ROBOTS BY USING MUSIC-INSPIRED ALGORITHMS

Abstract - A novel approach to the problem of the grid-based pathfinding has been introduced. The method uses the Harmony Search (HS) algorithm, a metaheuristic algorithm, for obtaining the optimal solution, whereas the Quad-tree algorithm has offered a great opportunity for decreasing the time needed to compute the solution. This Quad HS algorithm uses the Quad-tree decomposition of free space in the grid to mark the free areas and treat them as a single node, which greatly improves the execution. The results of the Quad HS algorithm have been compared to other metaheuristic algorithms, i.e. the Ant Colony and the Genetic Algorithm and proved to obtain the best results in terms of time and giving the optimal path.

Key words - Heuristic algorithms, Artificial intelligence, Computational intelligence, Optimization, Path planning.

СОДРЖИНА

1. ВОВЕД 9
2. ПРЕГЛЕД НА ЛИТЕРАТУРАТА 18
2.1. Потреба од автономен интелигентен робот 18
2.1.1. Основни концепти 18
2.1.2. Склад на однесувања 22
2.1.3. Склад со пропозиции 22
2.1.4. Склад со предикати 23
2.2. Навигација 23
2.2.1. Барања за автономија 25
2.2.1.1. Термални барања 25
2.2.1.2. Енергија 25
2.2.1.3. Менаџирање на комуникација 25
2.2.1.4. Механички дизајн 26
2.2.2. Критериуми за задоволување на возилата за да бидат автономни и интелигентни 26
2.2.2.1. Интелигенција 26
2.2.2.2. Навигација 26
2.2.3. Планирање на пат 27
2.3. Алгоритами за планирање на пат 28
2.3.1. Пристапи за планирање на пат 28
2.3.2. Карактеристики на планирањето на пат 28
2.3.3. Еволуција на моделирањето на околината на роботот 29
2.3.4. Глобално планирање на пат 31
2.3.5. Локално планирање на пат 32
2.4. Основни концепти за метаевристичките алгоритами 32
2.4.1. Оптимизациони модели 33
2.4.2. Комплексност на проблеми 34
2.4.3. Метаевристички оптимизациони методи 36
2.4.4. Кога да се користат метаевристички алгоритами? 39
2.5. Претходни истражувања 41
3. ЦЕЛ НА ИСТРАЖУВАЊЕТО 48
4. МЕТОДИ НА ИСТРАЖУВАЧКАТА РАБОТА 49

4.1. Harmony Search алгоритам 49
4.1.1. Основни концепти на Harmony Search алгоритамот 49
4.1.2. Основна структура на Harmony Search алгоритамот 51
4.1.2.1. Дефинирање на проблемот 52
4.1.2.2. Поставување на параметрите на алгоритамот 53
4.1.2.3. Случајно штелување за иницијализација на меморијата 55
4.1.2.4. Импровизација на хармонија 56
4.1.2.5. Ажурирање на меморијата 60
4.1.2.6. Терминација 61
4.1.2.7. Каденца 61
4.2. Quad-tree податочна структура 61
4.3. Quad Harmony Search (QHS) алгоритам 64
5. РЕЗУЛТАТИ 71
5.1. Опис на основните компоненти на архитектурата на системот за тестирање 71
5.2. Резултати од испитување на статички околини 73
5.3. Резултати од испитување на динамички околини 83
6. ДИСКУСИЈА 88
7. ЗАКЛУЧОК 89
8. ДОДАТОК 90
8.1. Користени кратенки 90
8.2. Слики од роботот користен за евалуацијата на резултатите од Quad Harmony Search алгоритамот 91
9. КОРИСТЕНА ЛИТЕРАТУРА (REFERENCES) 93

1. ВОВЕД

Регуларно структурираните околини отсекогаш се користени за потребите на глобално планирање на пат. Проблемот на глобално планирање на пат, применет на автономната навигација кај роботите, може да се дефинира како наоѓање на можен или оптимален пат од почетна точка до предефинирана цел во реална околина, при што околината може да биде статичка или динамичка, избегнувајќи колизија со препреки на тој пат. Ова знаење коешто се дава на планерот е значајно за автономијата на мобилниот робот. Проблемот на планирање на глобален пат не наоѓа честа примена во реални апликации, но се има докажано дека е од огромно значење во фазата на планирање на роботот пред да го започне својот пат до дестинацискатa точка. Најголем дел од алгоритамите за глобално планирање на пат вклучуваат локални и глобални планери и користат сензорски податоци и претходно меморирано знаење за околината. Додека локалниот планер се занимава со мали пречки во околината, аглите на вртење и применување на конкретните потребни брзини, глобалниот планер се справува со големите пречки и навигација кон дефинираната цел. Ова имплицира дека додека глобалниот планер ја врши навигацијата, локалниот планер се занимава со подобрување на патеката избегнувајќи ги околните објекти во околината.

Регуларно структурираните мапи кои се користат за глобално планирање на пат кај мобилни роботи, често може да бидат интерпретирани како дводимензионални или тридимензионални, зависно од комплексноста на околината. Овие пристапи вклучуваат дискретизација на околината и наоѓање на пат без колизии меѓу две точки во околината, односно почетна и крајна позиција. За да се изведе ова, треба да се конструира поврзан граф, а потоа да се примени погоден алгоритам за наоѓање на оптималниот пат. За главните цели на решавање на овој проблем, многу метаевристички алгоритами се покажале како практични. Знаејќи дека проблемот на глобално планирање на пат е NP-комплетен проблем и метаевристичките алгоритами се меѓу најпогодните решенија на ваков тип на проблеми, ова е оправдан пристап.

Автономните роботи кои работат без човечки манипулирања се неопходни во полето на роботиката. Со цел да се изведат задачи, автономните роботи мора да бидат интелигентни и да бидат способни да одлучат да изведат нивна сопствена акција. Кога автономниот робот ќе донесе одлука да изврши одредена акција, потребно е да се изведе оптимално планирање зависно од неговата задача. Уште повеќе, неопходно е да се планира пат ослободен од колизии со минимизирање на цените како време, енергија и растојание. Кога автономниот робот се движи од почетна до крајна точка во неговата претходно дефинирана околина, потребно е да се планира оптимален или можен пат со избегнување на препреки на неговиот пат и да се задоволат некои критериуми на автономните побарувања, како што се на пример: топлина, енергија, време и безбедност. Според тоа, најважната и главна работа на планирањето на пат за автономни мобилни роботи е да се пронајде пат слободен од колизии [1].

Многу истражувања се објавени на оваа тема за планирање на пат кај автономни мобилни роботи. Планирањето на движењето е една од најважните задачи во интелигентната контрола на автономниот мобилен робот. Често ова е поделено на планирање на пат и планирање на траекторија. Планирањето на пат има за цел да генерира пат ослободен од колизии во околина со препреки и да го оптимизира со почитување на одредени критериуми [2,3]. Планирањето на траекторија има за цел да го определи движењето на мобилниот робот низ планираниот пат. Предложени се неколку пристапи за да се адресира проблемот на планирање на движење кај мобилен робот. Ако околината е познат статички терен и однапред се изведува генерирање на пат се нарекува офлајн алгоритам (анг. off-line). Велиме дека алгоритамот е онлајн (анг. on-line) ако е способен за продуцирање на нов пат во согласност со промените што настануваат во околината [1].

Роботските системи кои се способни за одреден степен на самостојност се крајната цел на автономните мобилни роботи и се барани во многу полиња [4,5,6,7,8,9]. Фокусот е на способноста за самостојно движење додека се обидуваат да ја имитираат биологијата. Навистина, биолошките модели се од

најголем интерес, бидејќи живите суштества се прототипи на автономни однесувања. Интелигентните автономни системи (ИАС) имаат многу возможни примени во голем број на различни области, од просторни истражувања до справување со материјали, и од индустриски задачи до помош за хендикепирани. Поточно, препознавањето, учењето, донесувањето на одлуки и способностите за дејствување ги конституираат основните проблеми на избегнувањето на препреки на ИАС. Три нивоа се потребни за препознавање, имено: непрецизно процесирање на податоци (добиено од сензорите), конструкција на база на знаење, и конструирање на мапа на околината. Со цел да се решат овие проблеми и да се надминат недостатоците на класичните пристапи поврзани со реално време, самостојност и интелигенција, тековните пристапи се базирани на хибридни интелигентни системи [1].

Дизајнерите на ИАС се во потрага по креирање на динамички системи што прават навигација и изведуваат однесувања со определена цел, како човекот во реални ситуации каде условите се напнати. Како и да е, комплексноста на околината е специфичен проблем за решавање, бидејќи околините можат да бидат непрецизни, нејасни, динамички и делумно структурирани или неструктурирани. Тогаш, ИАС мора да биде способен да ја разбере структурата на овие околини. За да се достигне целта без колизии, ИАС мора да бидат потпомогнати со препознавање, учење, донесување на одлуки и способности за дејствување [1].

Способноста да се добијат овие предности да се обработи и пренесе знаењето го сочинува клучот на специфичен тип на интелигенција. Изградбата на ваква интелигенција е досега човечка амбиција во дизајнот и развојот на интелигентни возила. Како и да е, мобилниот робот е погодна алатка за истражување на опционални проблеми од вештачка интелигенција поврзани со разбирање на светот и преземање на соодветна акција, како планирање на мисии, избегнување на пречки и обединување на податоци од повеќе извори [1].

Тековно истражување за ИАС потенцираше ветувачки резултати за понатамошни истражувања во мобилната роботика каде реалното време, автономијата и интелигенцијата имаат значително добиено повеќе на тежина отколку, на пример,

оптималноста и комплетноста. Многу навигациски пристапи ја заменија експлицитната репрезентација на знаење со имплицитна базирана на стекнување на интелигентни однесувања што му овозможуваат на роботот да биде во интеракција ефективно со неговата околина, така што треба да се ориентираат себеси, да ја истражат нивната околина автономно, да се повратат од можен пад и да изведе цело множество на задачи во реално време [1].

Роботско возило е интелигентна мобилна машина способна за автономни операции во структурирана и неструктурирана средина, способна за добивање на податоци од сензори (во комуникација со околината), размислување (планирање и резонирање), и извршување (движење и манипулирање). Според тоа, тековните напредоци во автономните побарувања, интелигентните компоненти, мулти-роботските системи и масивното паралелно пресметување ги направија ИАС прилично употребувани, посебно кај планетарните истражувања, рударството, и автопатите [10,11,12,13,14]. Но, тековните мобилни роботи во мала мера изведуваат препознавачки активности како интелигентно размислување, а ова е поради тоа што:

- Перцепцијата не ги задоволува потребните стандарди.
- Голем дел од интелигенцијата е поврзан со однесување специфично за определена задача и повеќе е поврзан со конкретни уреди и мисии отколку со мобилни роботи генерално.
- Голем дел од предизвиците на мобилните роботи бара интелигенција на потсвесно ниво [1].

Движењето на мобилните роботи во непозната средина каде има стационарни непознати препреки побарува постоење на алгоритами кои се способни за решавање на проблемите на планирање на пат и движење на овие роботи така што колизиите се избегнуваат. Со цел да се изврши посакуваното движење, мобилниот робот интелигентно навигира и избегнува препреки така што целта е достигната. Проблемот станува потежок кога параметрите кои го опишуваат моделот и/или работниот простор на роботот не се точно познати [1].

Најважниот клучен проблем во дизајнот на автономниот робот е навигацискиот процес, што е еден од највиталните аспекти на автономниот мобилен робот. Според тоа, просторот и неговата репрезентација играат важна улога во областа на движењето на интелигентен систем. Ова значење може да се појасни со следниве причини:

- Ја обезбедува неопходната информација за планирање на пат.
- Дава информација за следење на позицијата на роботот за време на изведувањето на планираниот пат.
- Важно е дека мобилниот робот ја има способноста да изгради и да користи модели на неговата околина што му овозможуваат да ја разбере структурата на околината. Неопходно е да се разберат наредбите, да се планира и да се изведат патеките [1].

Теоријата и праксата на интелигенцијата и роботските системи се тековно најмногу проучувани и ветувачки области во компјутерските науки и инженерството кои дефинитивно во иднина ќе играат значајна улога. Овие теории и апликации обезбедуваат извор кој ги поврзува сите полиња во кои интелигентната контрола игра доминантна улога. Когницијата, перцепцијата, акцијата и учењето се есенцијални компоненти на такви системи и нивната интеграција во реални системи со различно ниво на комплексност (од микроботи до општества на роботи) ќе помогне да се појасни вистинската природа на роботската интелигенција [1].

Роботот е ,,уред" што дава одговор на влез од сензорите со извршување на програм автоматски без интервенирање од човечка страна. Типично, роботот е потпомогнат со определена вештачка интелигенција така што ќе реагира на различни ситуации на коишто ќе наиде. Роботот се однесува на сите тела кои се моделирани геометриски и можат да се контролираат преку план за движење. Роботското возило е интелигентна мобилна машина способна за акции во структурирана и неструктурирана околина. Мора да биде способна за собирање на податоци од сензори, размислување и делување. Мобилниот робот е идеална

алатка за решавање и на други типови на проблеми од областа на вештачката интелигенција [1].

Целта на процесот на навигација кај мобилните роботи е да го движи роботот кон означено место во позната, непозната или делумно позната околина. Во најголем дел од практичните ситуации, мобилниот робот не може да го изведе директниот пат од почетната до крајната точка. На овој начин, техниките на планирање на пат во оваа ситуација е неопходно да бидат користени, додека поедноставните типови на мисија на планирање вклучуваат движење од почетната до крајната точка притоа минимизирајќи определена цена како време, можност за детекција или трошење на гориво [1].

Честопати, патот кој треба да го следи роботот се планира офлајн, што може да го води роботот до дестинацијата под претпоставка дека околината е совршено позната и стационарна и роботот нема да има тешкотии. Рани планери на пат се таков тип на офлајн планери и се погодни за такво планирање. Како и да е, ограничувањата на офлајн планирањето ги поведе истражувачите да го проучуваат онлајн планирањето, кое се потпира на знаење добиено од сензорите при комуникација со локалната околина за справување со непознати препреки како што роботот се движи низ околината [1].

Уште повеќе, кога роботот се движи во специфичен простор, неопходно е да се одбере најразумен пат за да се избегне колизија со препреки. Постојат неколку пристапи за планирање на пат кај мобилни роботи, чија примена зависи од конкретен проблем во апликација. На пример, реактивните методи базирани на однесување се добар избор за робусно избегнување на препреки. Планирањето на пат во просторна репрезентација често бара интеграција на неколку пристапи. Ова може да обезбеди ефикасна, прецизна и конзистентна навигација на мобилен робот [1].

Главна задача на планирањето на пат за единични мобилни роботи е да се пронајде пат слободен од колизии. Истражувањата во планирањето на пат водеа кон решавање на проблеми на репрезентирање на мапата од реалниот свет. Според тоа, овој проблем се смета за еден од предизвиците во полето на

мобилните роботи поради неговото директно влијание на едноставната и пресметковно ефикасна стратегија за планирање на пат. За областите на планирање на пат, за роботот е доволно да користи тополошка мапа која претставува различни области без детали, како кај канцелариските соби. Можноста да се користат тополошки мапи со различни нивоа на апстракција помага во заштеда на времето на процесирање. Статичниот аспект на тополошките мапи го овозможува креирањето на патеки без информација која е значајна во време на извршување. Креираниот распоред на движење, кој се заснова на тополошката мапа, не чува ништо друго освен објекти кои го заземаат патот. Во тој случај, не е возможно да се изведе тој распоред. За да се добие натамошна неопходна информација, распоредот мора да биде збогатен со користење на повеќе ажурни планови, како егоцентрични мапи [1].

Тополошкото планирање на пат е корисно за креирање на долги патеки, кои ја поддржуваат навигацијата за решавање на одредена задача. Според тоа, тие јазли кои репрезентираат на пример, слободен регион, се екстрахирани од тополошка мапа, кои поврзуваат почетна точка со крајна точка. Почетната точка е најчесто тековната позиција на роботот. За да се генерира патот, постојат неколку софистицирани и класични алгоритами кои се базирани на теорија на графови како алгоритамот на најкраток пат. За да се даде најдобрата поддршка за планирањето на пат, може да биде од помош да се користат различни нивоа на апстракција за тополошките мапи. На пример, ако роботот влезе во конкретна соба на вработен за да достави пошта, роботот мора да користи тополошка мапа што ги содржи вратите на една зграда и броевите на собите [1].

Тополошките мапи можат да се користат за решавање на апстрактни задачи, како на пример, за одење и собирање на објекти чии позиции не се точно познати, бидејќи локациите на објектите често се менуваат. Тополошките мапи се графови чии јазли претставуваат статички објекти како соби и врати. Конекциите помеѓу јазлите се релациите помеѓу објектите [1].

Една од специфичните карактеристики на мобилните роботи е комплексноста на нивната околина. Според тоа, еден од критичните проблеми за мобилните роботи

е планирањето на пат, кој сè уште широко се истражува. Според тоа, еден од клучните проблеми во дизајнот на автономните роботи е навигацијата. Навигацијата е науката на насочување на курсот на мобилниот робот како што роботот се движи низ околината. Следствено во секоја шема за навигација постои желба да се достигне целта без роботот да се изгуби или судри со било кој објект. Целта на навигацискиот систем на мобилни роботи е да се движи роботот кон означено место во позната, непозната, или делумно позната околина [1].

Системите што ја контролираат навигацијата кај мобилниот робот се засновани на неколку парадигми. Биолошки мотивираните апликации, на пример, го адаптираат вообичаеното однесување кај животните. Геометриските репрезентации користат геометриски елементи како правоаголници, многуаголници, и цилиндари за моделирање на околината. Исто така, постојат системи за мобилни роботи кои не користат репрезентација на нивната околина. Однесувањето на роботот овде се одредува од сензорски добиените податоци. Беа воведени и други пристапи кои користат икони за репрезентирање на околината. Доволно е за роботот да користи тополошка мапа која ќе ги претстави областите за навигација (слободни области, зафатени области со препреки). Значајно е за роботот да може да изгради и користи модели на неговата околина што му овозможуваат да ја разбере структурата на околината. Ова е неопходно за разбирање на наредби, планирање и извршување на патишта [1].

Многу истражувања кои се правени во ова поле користеа ,,граф на видливост" за да се постави конфигурациски простор што може да се мапира во граф со јазли помеѓу кои патувањето е возможно во права линија. Недостатокот на овој метод е временската комплексност. Наспроти тоа, некои истражувања се засновани на поделба на мапата на околината во правилна структурирана околина и доделување на цена на секое поле (,,квадратче"). Цената на патот е сумата на сите цени на полињата на правилно структурираната околина низ кои патеката поминува. Моделот на правилно структурирана околина е усвоен од многу автори, каде околината на роботот е поделена на многу квадратчиња и индицираше присуство или отсуство на објект во секое поле [2, 3]. Модел на ќелии, од друга

страна, е развиен од многу истражувачи каде светот за навигација е поделен во области со ќелии, од кои некои содржат препреки. Уште повеќе, нацрт моделите на репрезентација на мапа за згради се користени за разбирање на структурата на околината, да се избегнат препреки и да се најде можен пат за навигација. Овие истражувања се развиени со цел да се пронајде ефикасна автоматизирана стратегија за наоѓање на пат кај мобилни роботи која ќе работи во опишаната околина каде што се движи роботот [1].

2. ПРЕГЛЕД НА ЛИТЕРАТУРАТА

2.1. Потреба од автономен интелигентен робот

2.1.1. Основни концепти

Теоријата и праксата на интелигентни автономни системи (ИАС) се тековно најинтензивно проучуваните и ветувачките области во компјутерската наука и инженерство кои ќе имаат клучна улога во иднина. Овие теории и примени обезбедуваат основа за поврзување на сите полиња во кои интелигентната контрола игра значајна улога. Когницијата, перцепцијата, акцијата и учењето се значајни компоненти на такви системи и нивната употреба екстензивно се насочува кон креирање на апликации кои претставуваат предизвик (сервисни роботи, микро-роботи, био-роботи, чувари-роботи, магацински роботи). Многу традиционални работни машини кои веќе се користат, како на пример во агрикултура или рударството, минуваат низ промени за да можат да бидат управувани од далечина, дури и автономни. Автономното возење во одредени услови е тогаш реалистична цел во блиска иднина [1].

Индустриските роботи кои се користат за манипулирање на склад типично се состојат од една или две раце и контролер. Терминот контролер се користи на најмалку два различни начини. Во овој контекст, се однесува на компјутерскиот систем кој се користи за контрола на роботот, често познат како контролер на работната станица на роботот. Контролерот може да биде програмиран да го управува роботот на различни начини, а исто така е одговорен за мониторирањето на помошни сензори кои детектираат присуство, растојание, брзина, облик, тежина, или други својства на објектите. Роботите можат да бидат опремени со системи за визија, зависно од апликацијата за која се користени. Најчесто, индустриските роботи се стационарни и работата се пренесува до нив со носач на приколки, кои најчесто се нарекуваат автономни водени возила (АВВ). Автономните водени возила сè повеќе се користат во индустријата за транспорт на материјали. Најчесто, овие возила користат сензор за следење на жица на подот на фабриката. Некои системи вградуваат и рака монтирана на АВВ [1].

Програмабилноста на роботот обезбедува големи предности во однос на потполната автоматизација. Ако постојат многу модели или опции на продукт, програмабилноста дозволува варијациите да бидат справувани полесно. Ако моделите на продуктот се менуваат често, како во автомобилската индустрија, генерално помалку чини да се репрограмира робот отколку повторно да се направи потполна автоматизација. Работната станица на роботот може да биде програмирана да изведува неколку задачи во последователен редослед во однос на единечни чекори. Ова овозможува полесно да се сретнат променливостите во продуктите со додавање и отстранување на работни станици. Исто така, бидејќи роботите можат да бидат репрограмирани да прават различни задачи, често е возможно да се амортизира нивната прва цена низ неколку продукти. Роботите може исто така да потпомагаат многу апликации кои не се погодни за човечките способности. Овие вклучуваат манипулирање на мали и големи објекти како електронски делови и турбински сечила, соодветно. Друг тип на апликација е работа во невообичаени околини, како чисти соби, топилници, области со висока радијација, и вселената. Јапонија го водеше светот кон користење на роботите во производството. Двата сектори кои најмногу ги користат роботите се автомобилската и електронската индустрија. Интерес во движењето на нозе е стимулирано од примената во изминување на нерамни површини и без придружното истражување на непозната околина. Настрана од електронската мотивација, постојат многу неодговорени научни прашања за тоа како биолошкиот организам го продуцира исклучителното однесување на сензорните мотори што се набљудуваат. Конечно, желбата за симулирање на биолошкиот организам има одредено инстинктивно репродуктивно барање и ја нуди можноста за задоволување на нашата љубопитност во однос на тоа како да се дојде до она што сме [1].

Роботско возило е интелигентна мобилна машина способна за автономни операции во структурирана и неструктурирана околина. Мора да биде способна за добивање на податоци од сензори (од неговата околина), размислување (планирање и резонирање), и дејствување (движење и манипулирање). Како и да е, мобилниот робот е соодветна алатка за истражување и на опционални

проблеми од вештачка интелигенција поврзани со разбирање на светот и изведување на соодветна акција, како планирање на мисии, одбегнување на препреки и собирање на податоци од повеќе извори. Денес, роботиката зафаќа специјално место во областа на интерактивните технологии. Комбинира софистицирани пресметки со богат влез од сензорите во физичко отелотворување што може да истражува реално и експресивно однесување во физичкиот свет. Во однос на ова, централното прашање кое ја интересира одредената истражувачка група е: ,,Која е соодветната прва улога за интелигентен човек - робот интеракција во дневната човечка околина?" Дојдено е времето да се разгледа ова прашање. Роботските технологии денес се доволно природни да овозможат интерактивни и компетентни роботски артефакти да бидат креирани. Проучувањето на човек - робот интеракцијата, кое дава плод во последниве години, покажува голема разновидност и во времетраењето на интеракцијата и улогите одиграни од човечките и роботските учесници. Во ситуации каде што човек дава краткотрајна потпомагачка интеракција на робот, истражување го има демонстрирано развојот на ефективните социјални односи. Дизајнот на антропоморфни роботи може да помогне во таков интерактивен експеримент со инстантно обезбедување на разбирливи социјални знаци за човечките субјекти. Технологијата го направи ова возможно со користење на напредни компјутерски контролни системи. Исто така, автомобилската индустрија има вложено многу напор во развивање на перцепциски и контролни системи за да го направат возилото побезбедно и полесно за управување [1].

За да се изведат сите задачи во различни околини, роботот мора да биде карактеризиран со построги ограничувања во однос на големиот волумен, потрошувачката на енергија, способностите за автономни реакции и комплексност на дизајнот. Поточно, за планетарните операции строгите ограничувања доаѓаат од достапната енергија и капацитетите за пренос на податоци, на пример, возилата се вообичаено дизајнирани како автономни единици со: пренос на податоци преку радио модеми до станиците за поддршка (сателити во орбитата или фиксни површински станици) и напојување од соларни ќелии, батерии или радиоизотопни термоелектрични генератори (за поголеми возила). Вообичаена

апликација на мобилниот робот е манипулацијата со објекти. Примерите вклучуваат операции на земање и сместување на објекти на фабричкиот под, сортирање на пакети и распределба. Некои истражувања се интересни според едноставноста на манипулацијата со објекти, т.е. туркањето. Туркањето е проблемот на менување на позицијата на објект со пренесување на контактна сила на него. Поради едноставност, ограничувањата на проблемот се сведуваат на менување на позицијата (во хоризонталната рамнина). Поранешен пристап кон роботското туркање беше имплементиран со цилиндрични роботи на две тркала опремени со сензори на допир кои имплементираат објектна реориентација и објектна транслација. Стратегијата беше наменета да користи два роботи за туркање на објект на дијагонално спротивните агли. Како резултат на ова туркање на товар надвор од центарот, кутијата ротира во едно место. Овој проблем е решен со цел да се детектираат и туркаат стационарни објекти во рамна средина со користење на сензорска мрежа вградена во средината и едноставен мобилен робот. Стационарните сензори се користат за детектирање на објекти што може да бидат туркани [1].

Роботот ќе биде спречен да се движи и свртува кон препреките со давање на информацијата за патеката помеѓу роботот и препреката во форма на сила. Оваа сила е слична на традиционалното потенцијално поле за планирање на пат кај мобилни роботи. Како и да е, околинските сили се различни од потенцијалните сили во одредени аспекти. Прво, нема фокусирање кон одредена цел зашто претпоставуваме дека крајната точка е непозната. Второ, само препреките во ,,релевантни" области (согласно со логичката позиција на интерфејсот) се разгледуваат, т.е. препреките кои се во насока спротивна на движењето на роботот не се релевантни. Во овој контекст, цел опсег на напредни интерфејси за управување со возила е испитуван од истражувачите. Овие трудови демонстрираат дека детекцијата и избегнувањето на препреки е подобрено со добри резултати [1].

Класичните системи за вештачка интелигенција претпоставуваат дека целото знаење е сместено во централна база на податоци од логички тврдења или друга

симболичка репрезентација и дека резонирањето се состои во голем дел од пребарување и секвенцијално ажурирање на таа база. Иако овој модел беше успешен за нематеријален систем на резонирање, истиот е проблематичен за роботи. Роботите се дистрибуирани системи: повеќе сензори, резонирање, и моторни контролни процеси се извршуваат паралелно, често на посебни процесори меѓусебно поврзани. Секоја од овие подготовки неопходно ја одржува својата посебна ограничена репрезентација на светот и задачата, побарувајќи од нив константно да се синхронизираат со централната база на знаење. Автоматизираните системи за резонирање се типично изградени на трансакционо-ориентирани модели на пресметување. Знаењето за светот е сместено во база на тврдења изразени во некој логички јазик, индексирани веројатно според имињата на предикатите. На пример, роботот треба да има рефлексивни складови кои му даваат на роботот пристап до неговата внатрешна состојба [1].

2.1.2. Склад на однесувања

Складот на однесувања содржи поврзувања помеѓу тагови и специфично роботско однесување. Секое однесување континуално споредува тагови со тагот на глобалниот повикувачки сигнал. Кога одредено однесување е детектирано, се активира. Активното однесување исто носи глобален движечки сигнал со бит-вектор на неговите тагови. Сигналот, според тоа, ги чува таговите на сите однесувања кои се во функција, дозволувајќи било кој дел од системот кој го мониторира сигналот да одреди дали однесувањето поврзано со одреден таг е во извршување [1].

2.1.3. Склад со пропозиции

Складот со пропозиции содржи поврзувања помеѓу таг и специфичен сигнал со бинарна вредност во системот. Складот генерира вистинит сигнал кој се состои од множество на сите тагови поврзани со пропозиции кои се тековно вистинити. Ова овозможува една компонента од системот да ,,пренесе" сигнал до друга компонента со негово врзување со таг кој е претходно дефиниран. Компонентата примач може потоа да го мониторира сигналот со испитување на соодветниот бит од вистинитиот сигнал [1].

2.1.4. Склад со предикати

Складот со предикати чува поврзувања помеѓу таг и унарен предикат. Складот со предикати генерира вектор на сигналот, индексиран според улога, чии елементи ги чува сите поврзани предикати – улога 0 во елемент 0, улога 1 во елемент 1, итн. Повторно, ова обезбедува олеснета интеракција за пренесување на сигнали помеѓу компонентите. Во овој контекст, може да се вклучи семантичка мрежа на пренесување на маркери. Јазол во мрежата може да биде поврзан со таг-улога и потоа да пропагира маркер низ линковите во мрежата за да се изведе повлекување и заклучок од долготрајната меморија [1].

Важно е да се разбере дека даден објект или концепт може да биде репрезентиран во неколку од овие складови симултано, со тоа што секој склад репрезентира различен аспект на објектот. Ова е поддржано со дозволување на елементи од различни складови да делат единичен регистриран таг. На пример, лексиконскиот влез во складот за зборот ,,prikazi", однесувањето PRIKAZI и семантичкиот јазол од мрежата кој репрезентира информација за однесувањето делат заеднички регистриран таг. Според тоа, кога парсерот поврзува ,,prikazi" со улога, однесувањето што може да го имплементира глаголот е автоматски поврзано со истата улога истовремено. Неколку трудови се презентирани во оваа област, при што многу истражувачи сметаат дека овој проблем може да произведе успешен систем за резонирање. Тие имаат дискутирано за алтернативна класа на архитектурно-тагирани системи базирани на однесување кои поддржуваат големо подмножество на способности на класична архитектура на вештачка интелигенција, вклучувајќи ограничено квантифицирано заклучување, поврзување кон напред и кон назад, едноставно одговарање на прашања на природен јазик и следење на команди, материјализирање и пресметковна рефлексија, овозможувајќи репрезентацијата на објектот да остане дистрибуирана низ повеќе сензорни и репрезентациони модули [1].

2.2. Навигација

Навигацијата е способност за самостојно движење. ИАС мора да биде способен да ги оствари овие задачи: избегнување на пречки и остварување на пат кон

нивната цел. Поточно, препознавањето, учењето, донесувањето на одлуки и акцијата конституираат основен проблем на навигацијата. Една од специфичните карактеристики на мобилниот робот е комплексноста на нивната околина. Според тоа, еден од критичните проблеми за мобилните роботи е планирањето на пат, кој е сè уште отворен за екстензивно истражување. Соодветно, еден од клучните проблеми во дизајнот на автономниот робот е навигацијата, за кој, планирањето на навигацијата е еден од највиталните аспекти на автономниот робот [1].

Неколку модели се применети за околината каде принципот на навигација е применет за изведување на планирање на пат. На пример, модел на правилно структурирана околина е прифатен од многу научници, каде околината на роботот е поделена на повеќе еднакви квадратчиња и индицираат присуство или отсуство на објект во секое од нив. Пречките се моделираат со делови на ,,sид", каде секој дел од ,,sидот" е права линија и репрезентирана од листа од неговите две крајни точки. Оваа репрезентација е конзистентна со репрезентацијата на познати објекти, истовремено потврдувајќи го фактот дека единствената парцијална информација за непозната препрека може да биде добиена преку земање на податоци од сензори на соодветна локација [1].

Покрај тоа, најважниот елемент на навигациониот систем на мобилниот робот е да го движи роботот до означено место во позната, непозната или делумно позната околина. Во најмногу практични ситуации мобилниот робот не може да изведе директна патека од почетната до крајната точка. Така, техниките за наоѓање на пат мора да се користат во оваа ситуација, и наједноставните видови на планирање на мисии вклучуваат движење од почетната до крајната точка минимизирајќи одредена цена, како време, можност за детекција, или потрошувачка на гориво. Кога роботот всушност ќе почне да патува низ планираниот пат, може да пронајде дека постојат совладливи препреки низ патот кои не се на мапата. Кога ова ќе се случи, роботот мора да ја исцрта препреката на мапата, и ако не е можно одбегнување со локално маневрирање, мора да испланира нов пат со модифицираната мапа и неговата тековна позиција станува

нова стартна локација. Соодветно, може да се подели задачата на движење на мобилниот робот во неговата околина во процес во два чекори:

1) Планирање на патеки кои се оптимални според одредени критериуми.

2) Контролирање на роботот за извршување на планираните патеки [1].

2.2.1. Барања за автономија

Неколку барања за автономија мора да бидат задоволени за добро да се извршат задачите на ИАВ. Ова е сумирано во следните секции.

2.2.1.1. Термални барања

За да се изведат задачи во различни околини како и во вселенските апликации, термалниот дизајн мора да биде земен во предвид, посебно кога температурата може значително да варира [2]. На амбиентни температури, достапната електронска опрема осетлива и ограничена според температурата мора да биде сместена во термално изолирани оддели [2]. Термалната околина на Марс го предизвикува термалниот контролен систем. На пример, во курсот на еден ден на Марс, температурата може да варира од 140K до 300K [1].

2.2.1.2. Енергија

За одреден период, каде ИАВ може да оперира автономно, еден многу ограничен ресурс за подводни и вселенски апликации е енергијата. Така, ИАС вообичаено носи енергетски систем на полнење, со соодветни достапни батерии [1].

2.2.1.3. Менаџирање на комуникација

Достапните компоненти на возилото и на површинската станица мора да бидат поврзани со двонасочен комуникациски линк. Како и кај подводните и вселенските апликации, систем за менаџирање со податоци вообичаено е потребен за пренесување на податоци од ИАС до земјен склад и процесирачки станици со двонасочен комуникациски линк. Навистина, системот за менаџирање со податоци мора да биде поделен помеѓу компонентите на возилото и површинската станица. Според тоа, возилото мора да биде поавтономно и интелигентно да ги изведе и

постигне овие задачи. Поради лимитираните ресурси и тежинските ограничувања, најголемото процесирање на податоците и капацитетите за складирање мора да бидат на површинската станица. Иако индивидуални возила може да имаат многу различни надворешни изгледи, различни механизми на движење и различни мисии или цели, многу од клучните пресметковни проблеми кои се вклучени се поврзани со земање податоци од сензори и сензорско просторно моделирање на податоците, како и резонирање [1].

2.2.1.4. Механички дизајн

Механичкиот дизајн на интелигентните автономни роботи (ИАР) е резултатот на пристап на интеграција земајќи во предвид неколку критериуми поврзани со перцепција, контрола и проблеми со планирање заедно со структурален дизајн, како и други механички побарувања [1].

2.2.2. Критериуми за задоволување на возилата за да бидат автономни и интелигентни

За да се евалуираат перформансите на ИАС кои се интелигентни и автономни возила, роботот мора да ги задоволи критериумите на интелигенција и навигација.

2.2.2.1. Интелигенција

Роботскиот систем способен за одреден степен на самостојност е примарната цел на интелигентните автономни возила. Така, роботот мора да ја оствари неговата задача со поголема автономност и интелигенција. Исто така, возилото реагира на непознати статички и динамички пречки со безбедносно побарување да не биде опасност за себеси или други објекти во околината. Доверливоста е земена во предвид во полето на роботиката; тоа е веројатноста дека побараната функција се извршува без пад во одреден период [14].

2.2.2.2. Навигација

Навигацијата е способноста за самостојно движење. Автономниот робот мора да биде способен да ги изведува следниве две задачи: да избегнува препреки и да

оствари пат кон целта. Поточно, препознавањето, учењето, донесувањето на одлуки и акцијата сочинуваат основен проблем на навигацијата [1].

2.2.3. Планирање на пат

Нека планирањето на пат се одвива во правилна средина и патеката помеѓу две локации се апроксимира со секвенца на соседни ќелии во структурираната околина соодветно на теренот. Должината A(α,) од ќелијата ,,α″ до неговата соседна ќелија ,,β″ е дефинирана со Евклидовото растојание од центарот на ќелијата ,,α″ до центарот на ќелијата ,,β″. Секоја ќелија во оваа околина може да биде во една од следниве состојби: слободна, зафатена или непозната. Ќелија е слободна ако е познато дека не содржи препреки, зафатена ако е познато дека се состои од една или повеќе препреки. Сите други ќелии се означени како непознати. Во средината, било која ќелија може да биде класифицирана според овие три состојби и да се потврдат ограничувањата во видливоста при навигацијата низ просторот [1].

Конфигурациската правилно структурирана средина е репрезентација на конфигурацискиот простор. Во конфигурациската средина почнувајќи од било која локација за да се стигне до друга, ќелиите припаѓаат на достиглив или недостиглив пат. Треба да се забележи дека како што множеството на достапни ќелии е подмножество на множеството од слободни конфигурациски ќелии, множеството на недостапни ќелии е подмножество од множеството на зафатени конфигурациски ќелии. Со селектирање на цел која лежи во достапниот простор, постои сигурност дека нема да биде во колизија и дека постои некој ,,возможен пат", така што целта ќе биде достигната во околината. Откако е одреден достапниот простор, алгоритамот работи и оперира на достапната околина [1].

Планирањето на пат е еден од најважните елементи за мобилниот робот [15]. Планирањето на пат е детерминирањето на патека што роботот мора да ја изведе низ одредени точки во околината до определена цел [16-19] и патеката е план од геометриски точки во даден простор низ кои роботот мора да помине [20]. Генерално, проблемот на планирање на пат е во врска со наоѓање на пат со поврзување на различни локации во дадена околина, како граф, лавиринт или

автопат. Планирањето на пат им дава можност на мобилните роботи да ги видат препреките и генерираат оптимален пат со нивно избегнување [15].

Генералниот проблем на планирање на пат за мобилни роботи е дефиниран како пребарување на пат кој роботот (со специфицирана геометрија) мора да го следи во опишана околина, со цел да се стигне до конкретна локација и ориентација, при дадена почетна позиција и ориентација. Знаејќи дека мобилниот робот не е точка во просторот, истиот мора да ја одреди коректната насока или да изведе соодветно движење за да се стигне до дестинацијата и ова е неговото т.н. планирање на маневрирање [15].

2.3. Алгоритами за планирање на пат

2.3.1. Пристапи за планирање на пат

Различни пристапи се воведени за имплементирање на планирање на пат кај мобилни роботи [21]. Овие пристапи зависат од околината, типот на сензорите, способностите на роботот итн. Истите постојано се подобруваат во подобри перформанси во контекст на време, растојание, цена и комплексност. Ал-Тахарва [22] на пример, го категоризираше планирањето на пат како оптимизационен проблем според дефиницијата дека, со даден мобилен робот и опис на околината, план е потребен помеѓу почетната и крајната точка за креирање на пат кој треба да биде слободен од колизии и да задоволува одредени оптимизациони критериуми, како најкраток пат. Оваа дефиниција е коректна ако целта на решавањето на проблемот на планирање на пат е наоѓање на најкраток пат зашто најмногу од најновите пристапи вклучуваат наоѓање на оптимален пат. Барањето на најкраткиот пат не гарантира дека времето ќе биде пократко, зашто некогаш најкраткиот пат има потреба од комплексен алгоритам кој ќе ја направи пресметката која може да зафати повеќе време [15].

2.3.2. Карактеристики на планирањето на пат

Планирањето на пат кај мобилни роботи има повеќе основни карактеристики согласно со околината, алгоритамот и комплетноста. Карактеристиките можат да бидат статички или динамички, локални или глобални и комплетни или

евристички. Статичното планирање на пат се однесува на околината која не содржи објекти или други препреки кои се движат покрај навигацискиот робот, додека динамичното планирање на пат се однесува на околина која содржи динамично движење и објекти кои се менуваат, како подвижни препреки. Во меѓувреме, локалното и глобалното планирање на пат зависи од алгоритамот каде информацијата за околината е дадена претходно или не е дадена на алгоритамот. Ако планирањето на пат е глобално, информацијата за околината е веќе позната врз основа на мапа, полиња, правилно структурирана околина итн. Од друга страна, ако планирањето на пат е локално, роботот нема информација за околината и роботот мора да ја скенира околината пред да донесе одлука да се движи одбегнувајќи препреки и да генерира планирање на траекторија кон целта [15].

Проблемот на навигација на мобилен робот може да биде поделен на три под-задачи, имено мапирање и моделирање на околината, планирање на пат и изминување на патот со избегнување на препреки [23]. Проблемите за навигација кај мобилните роботи не можат да бидат разложени на фиксни под-задачи зашто навигацискиот проблем варира зависно од користениот пристап да се реши проблемот. На пример, баг-алгоритамот го решава проблемот без потреба од мапа и модел на околината и само дава одговор на излезот од контактниот сензор [15].

2.3.3. Еволуција на моделирањето на околината на роботот

Од истражувачите на планирањето на пат (ПП) се предложени различни типови на познати статички модели на околината. Неколку имаат креирано модели на околината со следење на препреките (т.е. граф на видливост), ја имаат искористено областа на слободен простор (т.е. MAKLINK граф) и ги имаат искористено областите на слободен простор и препреки (т.е. декомпозиција на ќелии) за продуцирање на графот на поврзаност за ПП влезот [15].

Во раните 1980-ти, Родни Брукс вовел генерализирани конуси [24] кои го репрезентираат глобалниот слободен простор со конуси пред генерирање на оптималниот пат кон целта. Иако овој пристап докажано работи во едноставна

околина во неговото време, имал ограничувања кога се соочувал со комплексни околини. Имало потреба од повеќе време да пронајде пат поради комплексноста на процесот и за време од една година, друг алтернативен метод познат како пристап на ,,мапа на патот" (анг. roadmap) е воведен [15].

Пристапите на мапа на патот, познати како граф на видливост и Воронои дијаграм, се модели во категориите на пребарувачки граф техники [25]. Поврзаноста на графот на слободниот простор е генерирана пред ПП алгоритамите да почнат да наоѓаат пат. Иако овој пристап бил успешно имплементиран на неколку роботски ПП системи во едноставни околини, неговото ограничување било тоа што имал потреба од повеќе време да го креира Воронои дијаграмот бидејќи роботот има потреба од креирање на просторни точки во почетниот процес. За графот на видливост, јазлите се блиску со препреките и можноста за судир со препреките е висока [25]. Дополнително, исто така е комплексен пристап за роботски апликации во комплексни, екстремно несредени и променливи околини [15].

Пристапот на декомпозиција на ќелии, граф техника која е поефикасна, била воведена во раните 1990-ти. Овој пристап бил широко користен во роботските ПП системи и во статички и во динамички околини, бидејќи имплементацијата е полесна, точна и лесно се ажурира. Исто е една од најпопуларните репрезентации на околината. Неговото ограничување е дека работи побавно од други пристапи, посебно на постари машини кои имаат бавни процесори бидејќи има потреба од складирање на ќелиите. Иако на почетокот ова било проблем, тековната моќ на процесирање со новите генерации на компјутери што може да го реши овој проблем произведе нов интерес кон овој пристап. На пример, Галваски, кој моментално ги истражува неговите перформанси [26]. Денес, други пристапи на моделирање базирани на правилно структурирани околини се предложени, како квад-стебла (анг. Quadtree) и квад-стебла со рамки (анг. Framed Quadtree) [27-30], за да се зголеми точноста на пронајдениот пат. Дополнително, друг граф за моделирање на слободниот простор познат како MAKLINK граф [31] бил исто така воведен во 2000-та година. Моделирањето на околината се подобрува од година

во година. Споредено со постарите традиционални пристапи, како генерализирани конуси и мапа на патот, тековните пристапи за моделирање, како правилно структурирана околина, се побезбедни, прецизни, точни и повеќе прилагодливи за употреба во статички и динамички околини [15].

2.3.4. Глобално планирање на пат

Глобалното планирање на пат е планирање кое бара од роботот да се движи со претходно достапна информација за околината. Информацијата за околината прво се вчитува во програмот за планирање на пат кај роботот пред да се одреди патот од почетната до крајната точка. Во овој пристап, алгоритамот генерира комплетен пат од стартната до дестинацискатa точка пред роботот да почне да се движи [32]. Глобалното планирање на пат е процесот на внимателно одбирање на најдобриот начин за движење на роботот од стартната до крајната локација. Според тоа, за глобалното планирање на пат, одлуката за движење на роботот од почетна до крајна точка веќе е направена и тогаш роботот се пушта во одредена околина [15].

Еден од пораните модели за глобално планирање на пат што е темелно проучуван е проблемот на ,,придвижувачот на пијаното” (анг. Piano’s Mover) каде што целата информација се претпоставува дека е достапна во однос на геометријата, позициите на пречките и објектот кој се движи [33]. Во овој модел, целата комплексност на проблемот на планирање на пат е испитана, и бројни евристички и неевристички пристапи кои вклучуваат движење на цврсти или споени тела во дводимензионален или тридимензионален простор се земени во предвид [34]. Неколку вообичаени пристапи се користат во глобалното планирање на пат, како што се ,,мапа на патот” со користење на граф на видливост, Воронои граф и ,,силуети”, декомпозиција на ќелии преку точна декомпозиција, приближна декомпозиција и хиерархиска декомпозиција, како и нови модерни пристапи како генетски алгоритам [35-37], невронска мрежа [38] и оптимизација со колонија на мравки [39, 40].

2.3.5. Локално планирање на пат

Локалното планирање на пат е планирање кое побарува од роботот да се движи во непозната околина или динамичка околина каде алгоритамот кој се користи за планирање на пат ќе даде одговор на пречките и промените во околината. Локалното планирање на пат може исто така да биде дефинирано како избегнување на пречки во реално време со користење на сензорски базирана информација во согласност со непредвидливите мерења кои влијаат на зачувувањето на навигацијата на роботот [15,41].

Во локалното планирање на пат, нормално, роботот е воден по една линија од почетната до крајната точка и роботот ја следи линијата додека не наиде на препрека. Тогаш роботот изведува избегнување на препреката со отстапување од линијата и истовремено ажурирајќи некои важни информации, како ново растојание од тековната позиција до крајната точка, точка на изминување на препреката и сл. Во овој тип на планирање на пат, роботот мора секогаш да ја знае позицијата на крајната точка од тековната позиција за да постои одредена сигурност дека роботот може прецизно да стигне до целта [15].

Методот на потенцијални полиња [42] е еден од добро познатите техники на планирање на локален пат. Кај овој метод, роботот се третира како честичка која се движи под влијание на вештачкиот потенцијал произведен од конфигурацијата на целта и препреките. Вредноста на потенцијалната функција може да се разгледува како енергија и градиентот на потенцијалот е силата. Конфигурацијата на целта е атрактивен потенцијал, додека препреките се одбивен потенцијал [15].

2.4. Основни концепти за метаевристичките алгоритами

Пресметувањето на оптимални решенија е сè уште нерешено за многу оптимизациони проблеми од индустриско и научно значење. Во пракса, вообичаено се прифаќаат само ,,добри” решенија, кои се добиени со евристички или метаевристички алгоритами. Метаевристичките алгоритами претставуваат фамилија на оптимизациони техники на приближување кои стекнаа голема популарност во последните две декади. Тие се меѓу најмногу ветувачките и

успешни техники. Метаевристиките обезбедуваат ,,прифатливи" решенија во прифатливо реално време за решавање на тешки и комплексни проблеми во науката и инженерството. Ова го објаснува значајното зголемување на интересот во метаевристичкиот домен. За разлика од другите оптимизациони алгоритами, метаевристиките не гарантираат оптималност на добиените решенија. Тие не дефинираат колку се блиски добиените решенија до оптималните решенија [43].

Зборот ,,евристика" потекнува од старогрчкиот збор ,,heuriskein", што значи откривање на нови стратегии (правила) за решавање проблеми. Префиксот ,,мета", исто така старогрчки збор, значи ,,методологија од повисоко ниво." Терминот метаевристика за првпат е воведен од Ф. Гловер во трудот [44]. Метаевристичките методи на пребарување можат да бидат дефинирани како општи методологии (шаблони) од повисоко ниво што можат да бидат користени како водечки стратегии во дизајнирање на евристики за решавање на специфични оптимизациони проблеми [43].

2.4.1. Оптимизациони модели

Научниците, инженерите и менаџерите секогаш треба да донесуваат одлуки. Донесувањето на одлуки е присутно насекаде. Како што светот станува сè повеќе комплексен и компетитивен, донесувањето на одлуки мора да биде направено на рационален и оптимален начин. Донесувањето на одлуки се состои од следниве неколку чекори [43]:

- Формулирање на проблемот: во овој прв чекор, се идентификува проблемот за донесување на одлука. Потоа, се прави почетно тврдење од проблемот. Оваа формулација може да биде и непрецизна. Внатрешните и надворешните фактори, како и целите на проблемот се наведуваат. Повеќе донесувачи на одлуки можат да бидат вклучени во формулирањето на проблемот.
- Моделирање на проблемот: во овој важен чекор, се гради апстрактен математички модел за проблемот. Моделарот може да биде инспириран од слични модели од литературата. Ова ќе го редуцира проблемот на добро проучувани оптимизациони проблеми. Вообичаено, моделите што се

решаваат се поедноставувања на реалноста. Тие вклучуваат апроксимации и некогаш ги прескокнуваат процесите кои се комплексни за претставување во математичкиот модел. Овде може да се јави интересно прашање: зошто егзактно да се решаваат оптимизациони проблеми од реалниот живот кои по природа не се егзактни?

- Оптимизација на проблемот: откако проблемот е моделиран, процедурата на решавање генерира ,,добро" решение за проблемот. Решението може да биде оптимално или близу оптимално. Овде треба да се забележи дека се наоѓа решение за апстрактен модел на проблемот, а не за почетно формулираниот проблем од реалниот живот. Според тоа, перформансите на оптималното решение се покажуваат кога моделот е конкретен. Дизајнерот на алгоритамот може да користи постоечки алгоритами за слични проблеми или да се интегрира знаењето за оваа конкретна апликација во алгоритамот.
- Имплементирање на решение: добиеното решение се тестира практично од донесувачот на одлуки и се имплементира ако е ,,прифатливо." Некое практично знаење може да биде воведено во решението што треба да се имплементира. Ако решението е неприфатливо, моделот и/или оптимизациониот алгоритам мора да биде подобрен и процесот на донесување на одлуки се повторува [43].

2.4.2. Комплексност на проблеми

Комплексноста на даден проблем е еквивалентна на комплексноста на најдобриот алгоритам за решавање на тој проблем. Проблем е ,,решлив" (или лесен) ако постои алгоритам во полиномијално време за негово решавање. Проблем е нерешлив (или тежок) ако не постои алгоритам во полиномијално време за негово решавање [43].

Теоријата на комплексност на проблеми се занимава со донесување на одлуки. Проблем за донесување на одлука секогаш има одговор ,,да" или ,,не" [43].

Пример 1. Донесување на одлука за прост број. Популарен проблем за одлучување се состои во давање на одговор на следното прашање: дали даден

број Q е прост број? Ќе врати ,,да” ако бројот Q е прост број, а во спротивно се враќа одговор ,,не” [43].

Оптимизационен проблем може секогаш да биде редуциран на проблем за донесување на одлука [43].

Пример 2. Оптимизација наспроти проблем на донесување на одлука. Оптимизациониот проблем асоциран со проблемот на ,,трговецот што патува” (анг. traveling salesman problem) е: ,,пронајди го оптималниот Хамилтонов пат кој го оптимизира вкупното растојание,” додека проблемот на донесување на одлука е: ,,ако е даден број D, дали постои Хамилтонов пат со растојание помало или еднакво со D?” [43]

Важен аспект на теоријата на пресметување е да ги категоризира проблемите на класи за комплексност. Класа за комплексност го претставува множеството на сите проблеми што можат да бидат решени со одредена количина на ресурси за пресметување. Постојат две важни класи на проблеми: P и NP [43].

Класата на комплексност P го претставува множеството на сите проблеми за донесување на одлука што можат да бидат решени од детерминистичка машина во полиномијално време. Детерминистички проблем е полиномијален за проблем на донесување на одлука A ако неговата најлоша комплексност е поврзана со функција полином p(n) каде n ја претставува големината на влезната инстанца I. Оттука, класата P ја претставува фамилијата на проблеми каде постои познат алгоритам со полиномијално време за решавање на проблемот. Проблемите кои припаѓаат на класата P се релативно ,,лесни” за решавање [43].

Пример 3. Некои проблеми од класата P. Некои класични проблеми кои припаѓаат на класата P се стебло со минимално префрлување, проблеми на најкраток пат, мрежа со максимален тек, максимално поврзување на парови, и континуирани модели од линеарно програмирање [43].

Класата на комплексност NP го претставува множеството на сите проблеми за донесување на одлука кои можат да бидат решени од недетерминистички алгоритам во полиномијално време. Недетерминистички алгоритам содржи една

или повеќе точки за одбирање во кои повеќе различни текови се возможни без одредена спецификација за тоа која ќе биде одбрана. Се користат следните примитиви: ,,избор" што предлага решение (извор), ,,провери" што верификува во полиномијално време ако предлогот за решение дава позитивен или негативен одговор, ,,успех" кога алгоритамот одговара со ,,да" после проверката, и ,,неуспех" кога алгоритамот не одговара со ,,да." Тогаш, ако примитивата ,,избор" предлага решение што дава одговор ,,да" и изворот го има капацитетот за тоа, тогаш комплексноста за пресметување е полиномијална [43].

Прашањето дали P = NP е едно од најважните отворени прашања поради големото влијание што би го имало решението на теоријата на комплексност на пресметување. Очигледно, за секој проблем во P постои недетерминитички алгоритам за негово решавање. Според тоа, $P \subseteq NP$. Сепак, зависноста $P \subset NP$ е сè уште отворено прашање [43].

2.4.3. Метаевристички оптимизациони методи

Метаевристичките алгоритами решаваат проблеми од поголем обем со обезбедување на прифатливи решенија во разумно време. Не постои гаранција дека ќе бидат пронајдени оптимални или приближно оптимални решенија. Метаевристичките алгоритами имаат стекнато голема популарност во последните дваесет години. Нивната употреба во многу апликации ја покажуваат нивната ефикасност и ефективност за решавање на големи и комплексни проблеми. Примената на метаевристички проблеми спаѓа во голем број на области; некои од нив се [43]:

- Инженерски дизајн, тополошка оптимизација и структурна оптимизација во електрониката и VLSI, аеродинамика, динамика на флуиди, телекомуникации, автомобилски проблеми и роботика.
- Машинско учење и податочно рударење во биоинформатиката, пресметковната биологија и финансиите.
- Системско моделирање, симулација и идентификација во хемијата, физиката, и биологијата; контролно процесирање, сигнално процесирање и процесирање на слики.

- Планирање кај проблеми на рутирање, планирање кај роботи, проблеми на закажување и производство, логистика и транспорт, управување со ланец на снабдување, околини и сл.

Оптимизацијата е секаде; оптимизационите проблеми се често комплексни; според тоа и метаевристичките алгоритами се насекаде. Дури и во истражувачката заедница, бројот на сесии, работилници и конференции сè почесто се присутни метаевристичките алгоритами [43].

При дизајнирањето на метаевристички алгоритами, два контрадикторни критериуми мора да бидат земени во предвид: истражување на просторот на пребарување (диверзификација) и искористување на најдобро пронајдените решенија (интензификација). Регионите со добри решенија се одредени од добиените ,,добри" решенија. При интензификацијата, регионите со добри решенија се подобро истражуваат со цел да се пронајдат подобри решенија. При диверзификацијата, неистражуваните региони мора да бидат посетени за да постои сигурност дека сите региони на просторот за пребарување се подеднакво истражувани и дека пребарувањето не е намалено на само редуциран број на региони. Во овој простор на дизајнирање, алгоритамот за екстремно пребарување во контекст на истражување (соодветно искористување) е случајно пребарување (соодветно итеративно подобрување на локалното пребарување). Во случајното пребарување, во секоја итерација, се генерира случајно решение во просторот на пребарување. Не се користи меморија на пребарување. Кај основниот алгоритам на локално пребарување, во секоја итерација се селектира најдоброто соседно решение што го подобрува тековното решение [43].

Може да се користат повеќе класификациски критериуми за метаевристички алгоритами [43]:

- **Инспирирани од природата наспроти незасновани на природата:** Многу метаевристички алгоритами се инспирирани од природни процеси: еволуционите алгоритами и вештачките имунолошки системи од биологијата; колониите на мравки, колониите на пчели и оптимизација ја

множество на честички од интелигенцијата на маса од различни видови (социјални науки); како и симулирано жарење од физиката.

- **Базирани на меморија наспроти небазирани на меморија:** Некои метаевристички алгоритами не се базираат на меморија; ова значи дека ниедна информација што се екстрахира динамички не се користи за време на пребарувањето. Некои претставници на оваа класа се локално пребарување, GRASP и симулирано жарење. Другите метаевристички алгоритами користат меморија што содржи некоја информација која е добиена за време на пребарувањето. На пример, како краткотрајните и долготрајните мемории во табу пребарување.
- **Детерминистички наспроти стохастички:** Детерминистичкиот метаевристички алгоритам го решава оптимизациониот проблем со донесување на детерминистички одлуки (на пример, локално пребарување, табу пребарување). Кај стохастичките метаевристички алгоритами, некои случајни правила се применуваат за време на пребарувањето (на пример, симулирано жарење, еволуциони алгоритами). Кај детерминистичките алгоритами, користењето на истото почетно решение ќе води кон истото крајно решение, додека кај стохастичките метаевристички алгоритами, можат да се добијат различни крајни решенија од исто почетно решение. Оваа карактеристика мора да биде земена во предвид при евалуирање на метаевристичките алгоритами.
- **Пребарување базирано на популација наспроти пребарување базирано на едно решение:** Алгоритамите базирани на едно решение (на пример, локално пребарување, симулирано жарење) управуваат и трансформираат едно решение за време на пребарувањето, додека кај алгоритамите базирани на популации (на пример, множество на честички, еволуциони алгоритами) еволуира цела популација на решенија. Овие две фамилии имаат комплементарни карактеристики: метаевристичките алгоритами базирани на едно решение се ориентирани кон искористување; тие ја имаат моќта да го интензивираат пребарувањето во локалните региони. Метаевристичките алгоритами базирани на популации се

ориентирани кон истражување; тие дозволуваат подобра диверзификација во целиот простор на пребарување.

- **Итеративни наспроти алчни:** Кај итеративните алгоритами, се почнува со комплетно решение (или популација на решенија) и се трансформира во секоја итерација со користење на одредени оператори за пребарување. Алчните алгоритами почнуваат од празно решение и во секој чекор се доделува променлива на одлука за проблемот сè додека не се добие комплетно решение. Најголем дел од метаевристичките алгоритами се итеративни алгоритами.

2.4.4. Кога да се користат метаевристички алгоритами?

Комплексноста на даден проблем дава индикација на неговата тежина. Исто така е важно да се знае големината на влезните инстанци што алгоритамот треба да ги реши. И во случајот ако проблемот е NP - тежок, мали инстанци можат да бидат решени со егзактен пристап. Уште повеќе, структурата на инстанците игра многу важна улога. Некои инстанци со средна големина или големи инстанци со одредена структура можат оптимално да бидат решени со егзактни алгоритами. Конечно, потребното време за решавање на даден проблем е значаен проблем во селекцијата на оптимизациониот алгоритам [43].

Не е мудро да се користат метаевристичките алгоритами за решавање на проблеми каде што можат да се применат ефикасни егзактни алгоритами. Пример на оваа класа на проблеми е P класата на оптимизациони проблеми. Во случајот каде што тие егзактни алгоритами даваат ,,прифатливо" време на пребарување за постигнување на целта, метаевристичките алгоритами се бескорисни. На пример, не би требало да се користи метаевристички алгоритам за наоѓање на минимално стебло за префрлување. Постојат егзактни алгоритами со полиномијално време што ги решаваат овие проблеми [43].

Според тоа, за лесни оптимизациони проблеми, метаевристичките алгоритами се користат поретко. За жал, многу инженери и истражувачи решаваат оптимизациони проблеми кои имаат полиномијално време со метаевристички алгоритами. Затоа, првото нешто што треба да се направи при решавање на

проблем е анализа на неговата комплексност. Ако проблемот може да биде редуциран на класичен или веќе решен проблем во литературата, тогаш треба да се провери алгоритамот што најмногу одговара за решавање на проблемот. Инаку, ако постојат други слични проблеми, истата методологија мора да биде применета [43].

Дури за полиномијални проблеми, можно е степенот на функцијата полином кој ја претставува комплексноста на алгоритамот да е толку голем што реални инстанци на проблемот не можат да бидат решени во разумно време (на пример, комплексност од $O(n^{5000})$). Во контекст на комплексноста на проблемот, бараното време за извршување за решавање на проблемот е друг важен параметар кој треба да се земе во предвид. Навистина, дури и ако проблемот е полиномијален, потребата од користење на метаевристички алгоритами може да биде оправдана поради ограничувањата во реално време [43].

Многу комбинаторни оптимизациони проблеми припаѓаат на NP – тешката класа на проблеми. Оваа класа на проблеми, која е од големи димензии и со комплексни оптимизации, се јавува во многу области од индустриско значење: телекомуникација, пресметковна биологија, транспорт и логистика, планирање и производство, инженерски дизајн, итн. Уште повеќе, најголем дел од класичните оптимизациони проблеми се NP – тешки во нивната генерална формулација: алгоритамот на трговецот што патува, покривање на множества, управување со возила, партиционирање на граф, обојување на граф итн. [43,45]

За NP-тежок проблем каде познати егзактни алгоритами не можат да ги решат потребните инстанци (големина, структура) во бараното време на пребарување, оправдано е користењето на метаевристички алгоритами. За оваа класа на проблеми, егзактните алгоритами бараат (во најлошиот случај) експоненцијално време. Терминот на ,,потребно време” зависи од оптимизациониот проблем кој е предмет на интерес. За некои проблеми, ,,прифатливо” време може да е еквивалентно на неколку секунди, додека за други проблеми е еднакво на неколку месеци (проблеми на производство наспроти дизајн). Фактот дека проблемот не е во P класата не значи дека сите големи инстанци на проблемот се тешки или дека

најголемиот дел од нив се тешки. NP – комплетноста на проблем не имплицира ништо за комплексноста на конкретна класа на инстанци која треба да биде решена [43].

За нелинеарни континуирани оптимизации, треба да се применат метаевристички алгоритами, во случај доколку методи базирани на изводи како квази-Њутновиот метод или конјугираниот градиент се неуспешни поради неправилни простори за пребарување (на пример, неконтинуирани, нелинеарни, имаат недостатоци, шум, мултимодални се, не се глатки, или се неразделиви). Функцијата *f* мора да биде најмалку со просечна димензионалност (повеќе од три променливи). Овие својства се карактеристика на многу проблеми од реалноста. За лесни проблеми, како чисти конвексно квадратни функции, квази-Њутновиот метод е вообичаено побрз за фактор од околу 10 (во контекст на време на пресметување за да се добие целна вредност за функцијата на оптимизација) за разлика од најефикасните метаевристички алгоритами [43].

2.5. Претходни истражувања

Во последните неколку години објавени се многубројни истражувања за проблемот на глобално планирање на пат за мобилни роботи, со применување на различни типови на алгоритами и пристапи, со силно акцентирање на метаевристичките методи за решавање на планирање на пат во регуларно структурирани околини, кои широко се користат во голем број на апликации во областа на вештачката интелигенција. Таква ефикасна имплементација на планирање на пат во дводимензионална регуларно структурирана околина е презентирана во истражувањето приложено во [46], кое дава детално моделирање на околината со комплексни пречки, обезбедувајќи можно и приближно оптимално решение.

Постојат имплементации кои примениле модифицирани верзии на познати метаевристички алгоритами. Еден значаен дел од нив познатиот Ant Colony Optimization (ACO) алгоритам [47]. Постои релативно ново истражување што покажало дека алгоритамот на колонија на мравки може да биде доста практичен пристап за добивање на оптимални решенија каде целта се наоѓа во простор без

колизии [48]. Слично истражување, со користење на повеќе MATLAB експерименти, резултирало во прилично ефикасна модификација на ACO со подобра брзина, конвергенција и стабилност, и успешно бил применет на проблемот на глобално планирање на пат [49]. Друга имплементација го применува овој алгоритам на комплексни виртуелни сцени, детектирајќи колизии со технологијата на регуларно порамнети полиња (анг. axis aligned bounding boxes), а со применување на ограничувањата на околината, бил намален пребарувачкиот простор, што резултирало со подобрена ефикасност и практична примена [50]. Повеќекратен ACO систем, кој се состоел од повеќе колонии на мравки почнувајќи во иницијалната точка кои разменуваат различни типови на информација помеѓу себе во однос на откривање на можниот пат исто така се покажало како ефикасен пристап [51]. Хетерогениот ACO систем има дадено значајни модификации на традиционалниот ACO, во однос на функцијата на веројатност на премин (анг. Transition Probability Function), правилото на ажурирање на феромони (анг. Pheromone Update Rule), и презентирање на методот на вкрстување на патишта (анг. Path Crossover) [52]. Овој алгоритам бил понатаму подобрен во друго истражување, бидејќи тековната имплементација конвергирала кон решенијата пронајдени во пораните фази на алгоритамот, резултирајќи со конвергенција кон локално најдобри решенија. Овој подобрен пристап е познат под името билатерално кооперативно пребарување (анг. Bilateral Cooperative Exploration), кое го изведува пребарувањето два пати, со тоа што вториот пат почетната точка на пребарување е крајна и обратно [53]. Исто така, имплементирани се и хибридни пристапи, односно комбинација на ACO и генетскиот алгоритам (анг. genetic algorithm, GA), познат како smartPATH алгоритам [54]. Овој хибриден алгоритам вклучува подобрувања на ACO и модифициран оператор на вкрстување за GA делот од алгоритамот, избегнувајќи го афинитетот кон локален оптимум. Резултатите имаат покажано дека овој алгоритам е многу подобар од стандардниот ACO и Bellman-Ford методот [54]. Двонасочен ACO бил применет на проблемот на глобално планирање на пат во статична околина, така што постојат две маси на мравки кои се движат во спротивни насоки од почетната и крајната точка [55]. Евристичката информација

потоа била собрана од почетната точка, крајната точка и движењата на мравките [55]. Метод на апроксимација на дестинацискатa точка исто така резултирал со ефикасни резултати, кој се состоел од константно придвижување на почетната и крајната точка на лавиринтот една кон друга, па конвергенцијата резултирала со зголемена брзина [56]. Одреден вид на систем на колонија на мравки со евристика на потенцијално поле дал ефективна конструкција на планирањето на патот на роботот и овозможил успешно избегнување на препреките во околината [57]. Алгоритамот на клеточна колонија на мравки, кој се состоел од две колонии кои се движат според различни стратегии, ги оптимизирал патеките на мравките со користење на одредени клеточни правила, па мравките би можеле да прават скокови до регионот кој води до решението, резултирајќи во постабилен алгоритам [58]. Метод на ,,најдобра-најлоша" колонија на мравки значајно ги подобрил процесите на пребарување, додека максимален-минимален систем за ограничување на глобалниот интензитет на феромоните обезбедил применливи решенија на проблемот на планирање на пат [58].

Генетските алгоритами (GA) исто така имаат обезбедено ветувачки резултати во однос на проблемот на глобално планирање на пат кај мобилни роботи. Покажано е дека овој алгоритам, применет на робот способен да се движи во четири насоки, може да надмине многу недостатоци во однос на градиентните методи [59]. Паралелниот елитен генетски алгоритам (PEGA), кој се состои од два елитни генетски алгоритами со миграционен оператор, ја има подобрено разновидноста на истражување на решенијата, а резултатот од оваа техника понатаму беше подобрен со користење на кубична Б-крива, обезбедувајќи можен пат од почетната до дестинацискатa точка [60]. Постојат и хибридни пристапи кои го користат генетскиот алгоритам, имено комбинацијата со алгоритамот на симулирано жарење (анг. Simulated Annealing). Истражувањето приложено во [61] ја користи MAKLINK теоријата на графови за конструирање на простор без колизии, потоа алгоритамот на Dijkstra се користи за добивање на можен пат, па на крајот решението се наоѓа со применување на хибридниот алгоритам кој се состои од генетскиот алгоритам и алгоритамот на симулирано жарење. Слично истражување ги користи првите две фази од претходно опишаниот пристап, со

единствена модификација во третата фаза со користење на комбинација од генетскиот алгоритам и A* алгоритамот за пребарување, кој исто така обезбедил подобри резултати од алгоритамот на Dijkstra [62]. Хаотичниот генетски алгоритам (анг. chaotic genetic algorithm, CGA) вовел додавање на хаотичен оператор на генетскиот алгоритам, подобрувајќи ја значително конвергенцијата на алгоритамот и избегнувањето на локалните оптимуми се покажало како успешно [63]. Прецизни и приближно оптимални решенија исто така се добиени кога дводимензионалната интерпретација на патеката е трансформирана во еднодимензионална интерпретација и оптимизациските функции за добивање на пат и обезбедување на оптимално решение се вклопени во главната оптимизациска функција [64]. Релативно нов алгоритам, имено мултифрекфенциониот вибрационен GA алгоритам (mVGA) воведува концепт на случаен глобален и локален диверзитет. Во почетните фази на алгоритамот се употребуваат метод на кластерирање и Воронои дијаграм, а потоа алгоритамот се применува на планирање на пат на автономно возило во тридимензионална околина [65]. Комбинација на критериум на ,,планирање во секое време" и мултирезолуциски пребарувачки простори во генетскиот алгоритам резултирала со повеќе паралелни еволуции, разменувајќи информација помеѓу овие паралелни пребарувачки нишки за решенијата со ниска цена, подобрувајќи ја целокупната конвергенција [66]. Неодамна подобрен GA се состои од неколку оптимизации применети на овој алгоритам. Методот на постепено искачување (анг. hill-climbing) прво се користи за да се подобри мутацијата на алгоритамот. Потоа, алгоритамот на оптимизација на множество на честички (анг. particle swarm optimization) се користи да се забрза конвергенцијата, па на крајот се прави емулација со вклучено т.н. float-point кодирање во подобрениот GA [67]. Дуалната еволуција на популации со адаптација на пропорционален праг кога постои бинарен пат со фиксна должина ја подобрил способноста на алгоритамот да гравитира кон глобално оптималните решенија и ја зголемил брзината на конвергирање [68]. Воведувањето на адаптивен локален оператор за пребарување и применување на ортогонален метод на дизајнирање во процесот на иницијализација на популацијата и со вклучување на интергенерационен елитен механизам, исто така резултирало со забрзување во

споредба со традиционалниот GA [69]. Постојат и истражувања кои имплементираат хибрид од алгоритамот на Вештачки Потенцијални Полиња (анг. Artificial Potential Fields, APF) и GA, кој го користи APF методот за одредување на област без колизии и избегнување на препреките во околината, а потоа се применува оптимизацискатa функција на GA подобрена со помош на методот на најмали квадрати [70]. Оваа комбинација е применета на планирање на пат кај мобилни роботи во играње фудбал и обезбедила ефективни резултати во однос на наоѓање на оптималниот пат [70].

Алгоритамот на оптимизација на множество честички (анг. Particle Swarm Optimization, PSO) [69] е метаевтистички алгоритам кој потекнува од 1995 и оттогаш има доживеано многу модификации и варијации. PSO повеќе ја користи аналогијата на социјалната интеракција, отколку на индивидуалните когнитивни способности [72]. За целите на проблемот на глобално планирање на пат кај мобилни роботи, предложен е PSO со гарантирана конвергенција (анг. Guaranteed Convergence Particle Swarm Optimization) [73]. Во ова истражување, патеката помеѓу почетната и крајната точка се третира како честичка. Потоа, според локациите на препреките, се дефинира активен регион, каде се генерира почетната популација. Ова е само-адаптирачка техника која се покажала дека ги избегнува локалните оптимуми и дава ефективни и брзи резултати [73]. Едноставниот PSO алгоритам исто така е забрзан со имплементацијата на копија на клонови, вкрстување на клонови, хипер мутација и селекција на клонови, а исто така е воведен и поли-клон PSO (анг. polyclone PSO, PCPSO) алгоритам. Овие два методи се споредени со алгоритамот на PSO со инертна ширина, и PCPSO докажано дал ефективни и приближно оптимални резултати [74]. Постои и истражување кое користи подобрен Dijkstra алгоритам и PSO, каде што околината на роботот е конструирана со MAKLINK граф [75]. Овој метод продуцирал помалку можни патишта за истражување и PSO потоа се користел за истражување на оптималното решение од малкуте останати патишта [75]. PSO со кубични криви го дефинирал патот на роботот како низа од кубични криви, па потоа го воведува PSO за оптимизирање на параметрите на дадена крива за целите на наоѓање на оптимален пат. Резултатите имаат покажано дека овој алгоритам е практичен во

обезбедување на приближно-оптимален пат [76]. Планирањето на глобален пат исто така е решено со користење на PSO со квантум-однесување. Овде, мапата на роботот помеѓу почетната и крајната точка е репрезентирана со промена на координатниот систем. Алгоритамот покажал забрзана конвергенција без особени ограничувања во однос на обликот на препреките [77]. Во последните неколку години, објавени се и хибридни алгоритами кои го користат PSO алгоритамот во комбинација со други метаевристички алгоритами. Овие пристапи ја имаат докажано својата ефикасност преку повеќе експериментални истражувања. Еден од нив е хибридниот GA-PSO алгоритам, кој го користи генетскиот алгоритам како забрзување на постоечкиот PSO алгоритам. Овој алгоритам ги примени операторите за мутација и вкрстување на генерираните резултати од честичките. После наоѓање на можен пат, техниката на кубична Б-крива се користи за продуцирање на поглатко и подобро решение. Овој метод ги избегнува недостатоците на рана конвергенција, што е типичен проблем кога овие метаевристички алгоритами се користат посебно [78]. Исто така, објавен е и ортогонален PSO алгоритам, кој го вклучува ортогоналниот оператор на дизајнирање кон едноставниот PSO алгоритам со цел да се избегне конвергенцијата кон локален оптимум. Споредено со традиционалниот PSO алгоритам, овој пристап дал и поефективни резултати [79]. Кога моделот на патниот простор на мобилниот робот е трансформиран со декомпозиција во дводимензионална патека, парови на честички се способни за размена на информација за операторот на вкрстување. Ова довело до избегнување на конвергенцијата кон локален оптимум и давање на приближно оптимални решенија [80]. Неодамнешни пристапи го одведоа проблемот на планирање на пат во четири димензии, дефинирајќи го проблемот како проблем на калкулус на варијации (ПКВ). Потоа, решението на ПКВ се добива ефективно преку користење на PSO алгоритамот [81].

Алгоритамот на симулирано жарење (анг. simulated annealing, SA) е метаевристички алгоритам кој е инспириран од процесот на бавно ладење присутно во процесот на жарење [82], и е користен за многу истражувања поврзани со проблемот на глобално планирање на пат. Постојат неодамнешни

истражувања кои користат модифицирани верзии на овој алгоритам за овие цели. Еден таков пример е комбинацијата на конјугациониот дирекционен метод со симулирано жарење [83]. Овде, конјугациониот метод се користи за пребарување на локално оптимални решенија за секоја температура, па SA алгоритамот се користи за преместување од тој оптимум и потоа се ажурира вредноста на температурата. Овие чекори се повторуваат додека не се достигне глобалниот оптимум [83]. Метод што користи пристап на адаптивно соседство при SA алгоритамот го репрезентира патот на роботот како поли-крива, Безиерова крива и како интерполирана крива. Параметрите се ажурираат на секоја итерација со цел да се оствари забрзана конвергенција и да се зголеми бројот на прифатливи решенија [84]. Други истражувања го репрезентираат патот на роботот преку Воронои дијаграм. Конкретно истражување користи Воронои дијаграм за наоѓање на пат без колизии со користење на алгоритамот на Dijkstra, користејќи ги овие податоци за пресметување на најдобриот пат со примена на SA, задоволувајќи ги кинематичките ограничувања [85]. Резултатите имаат покажано дека овој пристап дава подобри резултати од традиционалниот SA алгоритам [85]. SA исто така е комбиниран со други метаевристички техники за решавање на проблемот на глобално планирање на пат кај мобилни роботи. Конкретни истражувања го користат GA како забрзување на SA алгоритамот [86, 87]. Некои истражувања тврдат дека овој хибриден пристап на GA и SA избегнува прерана конвергенција и дава подобри резултати [88]. На крај, постои истражување кое тврди дека SA алгоритамот, користен самостојно и спореден со другите метаевристички техники, е способен секогаш да најде решение и се докажал дека е практичен и ефективен, но споредено со табу пребарувањето, не секогаш е способен да го пронајде најкраткиот можен пат [89].

3. ЦЕЛ НА ИСТРАЖУВАЊЕТО

Целта на оваа магистерска тема е да презентира нов метаевристички пристап за решавање на проблемот на глобално планирање на пат кај мобилни роботи со користење на популарниот Harmony Search оптимизационен алгоритам. Како што имаат покажано резултатите, пребарувањето може да биде изведено во помалку итерации на алгоритамот со соодветниот избор на параметрите на алгоритамот и прецизна дефиниција на функцијата на оптимизација. За целите на ова истражување, се претпоставува дека околината на дејствување на роботот е дискретизирана во дводимензионална околина и истата може да биде статичка и динамичка. Ова знаење за работниот простор на роботот е дадено *a priori*, што значи дека процесот на планирање може да почне пред мобилниот робот да го започне својот кон својата дестинација. Тестовите беа извршени на автономен робот со три тркала и беа добиени прифатливи резултати во тестирањата во реална околина. Целта на ова исцрпно истражување е да даде алтернатива на постоечките метаевристички алгоритами кои се применуваат на овој проблем и да обезбеди поефикасен, поефективен и побрз начин за исполнување на целите на овој процес.

Релативно новиот Harmony Search алгоритам, чии основи се базираат на музичкиот процес на џез импровизација на музичарите, ги користи процесите на импровизирање на решенија од меморијата на музичарот и ги имплементира ефективно истите концепти. Со цел да се забрза процесот на оптимизација, се воведува и Quad-tree алгоритамот за декомпозиција на просторот на мобилниот робот на слободни региони, при што истите региони се третираат како посебни јазли. На тој начин, се гради листа на соседство која понатаму служи за оперирање на оптимизациониот Harmony Search алгоритам.

Овој нов алгоритам има за главна цел да овозможи подобра алтернатива за решавање на проблемот на глобално планирање на пат кај мобилни роботи и оваа цел е ефективно постигната. Во наредните поглавја е елаборирана алгоритамот кој се користи во ова истражување и направена е споредба на новиот

Quad Harmony Search алгоритам со други метаевристички алгоритами. Воедно, се изложени и резултатите од компарацијата на алгоритамите од различни аспекти.

4. МЕТОДИ НА ИСТРАЖУВАЧКАТА РАБОТА

4.1. Harmony Search алгоритам

4.1.1. Основни концепти на Harmony Search алгоритамот

За оптимизација, луѓето традиционално имаат користено алгоритами базирани на калкулус што даваат градиентна информација со цел да се пронајде вистинската насока кон оптималното решение. Како и да е, ако променливите се дискретни наместо непрекинати, не можат да бидат изведени. За да се надмине оваа ситуација, Harmony search (HS) алгоритамот користи нов стохастички извод [90, 91] кој ги користи искуствата на музичарите во џез импровизацијата и може да се примени на дискретни променливи. Наместо инклинациската информација на функцијата за оптимизација, стохастичкиот извод на HS обезбедува веројатност да биде селектирана за секоја вредност на променливата на одлука. На пример, ако променливата на одлука x_1 има три кандидат вредности {1, 2, 3}, парцијалниот стохастички извод на функцијата за оптимизација со однос на x_1 за секоја дискретна вредност ја дава веројатноста на селекција за секоја вредност како 20% за 1, 30% за 2, и 50% за 3. Додека кумулативната веројатност е 100%, веројатноста за секоја од вредностите се ажурира итерација по итерација. Пожелно е вредноста која е вклучена во оптималниот вектор на решението да има повисока веројатност да биде избрана како што се развива итерацискиот процес [90].

Со оваа информација за стохастичкиот извод, HS алгоритамот е применет на различни научни и инженерски оптимизациски проблеми кои вклучуваат [90, 92, 93]:

Апликации во реално време

- Компонирање на музика
- Судоку сложувалки
- Временски табели

- Планирање на патувања
- Логистика

Компјутерски науки

- Кластерирање на веб-страници
- Сумаризација на текст
- Интернет рутирање
- Визуелно следење
- Роботика

Електроинженерство

- Испраќање на енергетски системи
- Фото-електронска детекција
- Дизајн на системи за напојување
- Повеќе-нивовско инвертерско оптимизирање
- Мрежа на мобилни телефони

Градежништво

- Структурален дизајн
- Дизајн на водоводни мрежи
- Распоредување на прегради
- Калибрирање на модел на поплави
- Управување со подземните води
- Анализа на стабилноста на почвата
- Еколошка конзервација
- Рутирање на возила

Механичко инженерство

- Дизајнер на размена на топлина
- Дизајн на сателитски топлотни цевки
- Сидра на крајбрежна структура

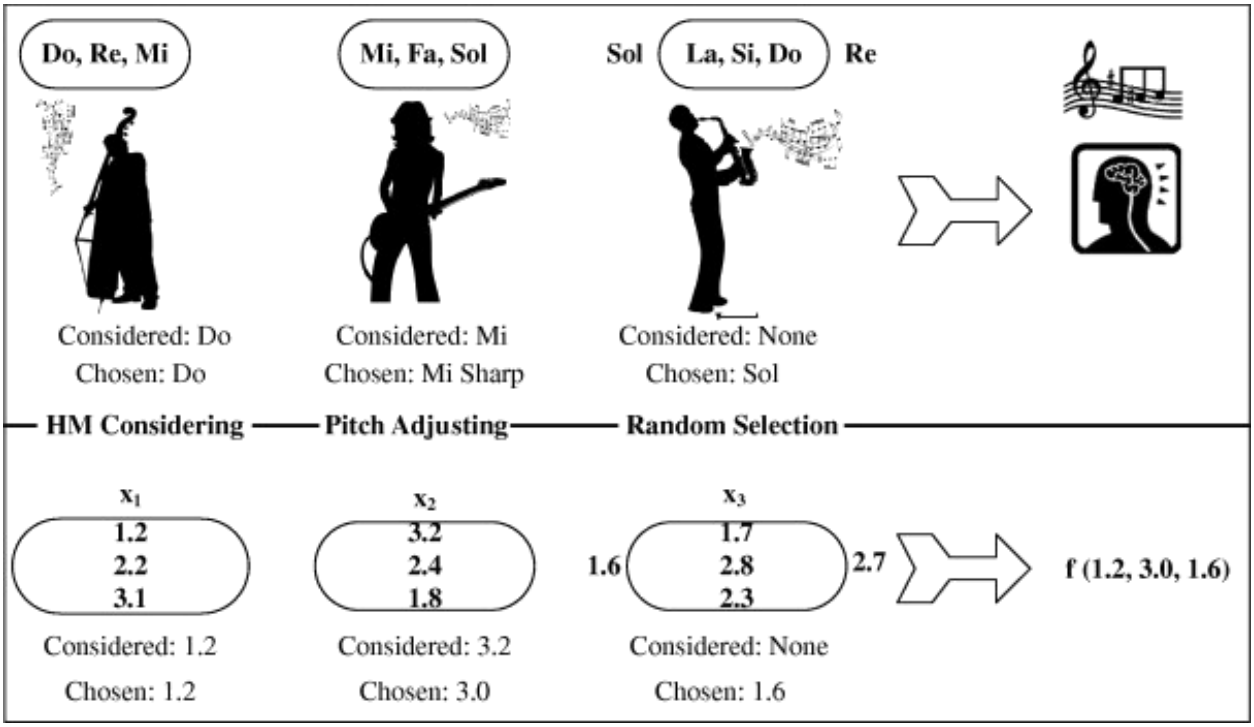
Биолошки и медицински апликации

- Предвидување на структура на RNA
- Помагала за слух
- Медицинска физика.

Како надополнување на гореспоменатите различни апликации, HS алгоритамот исто има различни алгоритамски структури што можат да бидат применети на толку многу различни проблеми. Според тоа, ова поглавје ја дава основната структура на HS алгоритамот, така што корисниците ќе можат лесно да го прилагодат алгоритамот на нивните оптимизациони проблеми [90].

4.1.2. Основна структура на Harmony Search алгоритамот

HS алгоритамот е оригинално инспириран од импровизацискиот процес на џез музичарите. Сл. 1 ја покажува аналогијата помеѓу импровизацијата и оптимизацијата [90].



Слика 1. Аналогија помеѓу импровизацијата и оптимизацијата.

Figure 1. Analogy between the improvisation and optimization.

Секој музичар соодветствува на една променлива на одлука; опсегот на фреквенции на музичкиот инструмент соодветствува на опсегот на вредности на

променливата на одлука; музичката хармонија во одредено време соодветствува на векторот на решение во конкретна итерација; и естетиката од публиката соодветствува на функцијата за оптимизација. Исто како што музичката хармонија се подобрува низ времето, векторот на решение се подобрува итерација по итерација [90].

Ова поглавје го претставува секој чекор од HS алгоритамот во детали, вклучувајќи: 1) дефинирање на проблемот, 2) поставување на параметрите на алгоритамот, 3) случајно штелување за иницијализација на меморијата, 4) импровизација на хармонија (случајна селекција, разгледување на меморијата, и модулирање на фреквенциите), 5) ажурирање на меморијата, 6) терминација и 7) каденца [90].

4.1.2.1. Дефинирање на проблемот

HS алгоритамот е креиран за решавање на оптимизациони проблеми. Според тоа, со цел да се примени HS, проблемите треба да бидат формулирани во оптимизационата околина, со оптимизациона функција и ограничувања [90]:

Оптимизирај (минимизирај или максимизирај) f(x) (1)

При што:

$$h_i(x) = 0; \quad i = 1,..., p \tag{2}$$

$$g_i(x) \geq 0; \quad i = 1,..., p \tag{3}$$

$$x_i \in X_i = \{x_i(1),..., x_i(k),..., x_i(K_i)\} \quad или \quad x_i^L \leq x_i \leq x_i^U \tag{4}$$

HS алгоритамот ја пребарува целата област на решенија со цел да го пронајде оптималниот вектор на решенија x = (x_1,…,x_n), кој ја оптимизира (минимизира или максимизира) функцијата во равенката (1). Ако проблемот содржи услови за еднаквост и/или нееднаквост, можат да бидат разгледувани како ограничувања во равенките (2) и (3). Ако променливата на одлука има дискретна вредност, множеството на кандидати вредности за променливата станува

$x_i \in X_i = \{x_i(1),..., x_i(k),..., x_i(K_i)\}$; и ако променливата има непрекинати вредности, множеството на кандидати вредности за променливата станува $x_i^L \le x_i \le x_i^U$ [90].

HS алгоритамот во основа ја разгледува само оптимизационата функција. Како и да е, ако генерираниот вектор на решение нарушува некое од ограничувањата, 1) алгоритамот го отфрла векторот или 2) го зема во предвид со додавање на одредена вредност за казнување кон вредноста на оптимизационата функција. Исто така, HS може да биде применет на проблеми со повеќе цели со конјугирање со Парето множеството [90].

4.1.2.2. Поставување на параметрите на алгоритамот

Откако е спремно дефинирањето на проблемот, параметрите на алгоритамот треба да бидат поставени на одредени вредности. HS содржи неколку параметри, како големина на хармониската меморија (анг. harmony memory size, HMS), на избирање вредност за променливата на одлучување од хармониската меморија (анг. harmony memory considering rate, HMCR), веројатност на модулирање на тонот (анг. pitch adjusting rate, PAR), максимална импровизација (анг. maximum improvisation, MI), и ширина на фрет (анг. fret width, FW) [90].

HMS е бројот на вектори на решенија кои симултано се разгледуваат во алгоритамот; HMCR е веројатност (0 ≤ HMCR ≤ 1) на избирање вредност за променливата на одлучување од хармониската меморија. Според тоа, (1-HMCR) е веројатност на избирање вредност за променливата на одлучување од сите можни вредности; PAR (0 ≤ RAR ≤ 1) е веројатноста каде HS ја променува вредноста што е оригинално одбрана од меморија. Според тоа, (1-PAR) е веројатноста каде HS ја чува оригиналната вредност која е добиена од меморија; MI е бројот на итерации. HS импровизира една хармонија (= вектор) во секоја итерација; и FW е произволна само за континуирана променлива, што порано беше позната под името ширина на опсег (анг. bandwidth, BW). За повеќе информација околу терминот, фрет е металната линија на вратот на жичан инструмент (како гитара), кој го дели вратот на фиксни сегменти (сл. 2), и секој фрет претставува еден полутон. Во контекст на HS алгоритамот, фретовите

значат произволни точки кои го делат вкупниот опсег на вредности на фиксни сегменти, и ширина на фрет (FW) е должината помеѓу два соседни фрета. Рамномерен FW се вообичаено користи за HS [90].

Слика 2. Фретови на вратот на гитарата.

Figure 2. Frets on the neck of the guitar.

На почетокот се користеа фиксни параметарски вредности. Како и да е, некои истражувачи имаат предложено променливи параметарски вредности. Махдави [94] предложи PAR да расте линеарно и FW да опаѓа експоненцијално во итерациите [90]:

$$PAR(I) = PAR_{\min} + (PAR_{\max} - PAR_{\min}) \times \frac{I}{MI} \tag{5}$$

$$FW(I) = FW_{\max} \exp\left[\ln(\frac{FW_{\min}}{FW_{\max}}) \frac{I}{MI} \right] \tag{6}$$

Мукопадхај [95] сугерираше FW да биде стандардната девијација на тековната популација кога HMCR е блиску до 1 [88].

$$FW(I) = \sigma(x_i) = \sqrt{\mathrm{var}(x_i)} \tag{7}$$

Гем [96] ги табелираше фиксните вредности за параметрите, како број на променливи, HMS, HMCR, PAR, и MI, после истражување на различни литератури. FW вообичаено се движи од 1% до 10% од вкупниот опсег на вредности [90].

Уште повеќе, некои истражувачи имаат предложено теории на адаптивни параметри кои му овозможуваат на HS автоматски да ги одбере најдобрите вредности за параметрите во секоја итерација [90, 93, 97].

4.1.2.3. Случајно штелување за иницијализација на меморијата

Откако проблемот е формулиран и вредностите на параметрите се коректно поставени, се изведува процес на случајно штелување [90].

На оркестрален концерт, откако обоата ќе ја отсвири нотата A (вообичаено A440), другите инструменти случајно свират ноти од опсегот на ноти кои можат да се отсвират. На истиот начин, HS алгоритамот иницијално импровизира многу случајни хармонии. Бројот на случајни хармонии треба да биде најмалку HMS. Како и да е, бројот може да биде поголем од HMS, како два пати или три пати повеќе од HMS [98]. Потоа, најдобрите HMS хармонии се одбрани како стартни вектори [90].

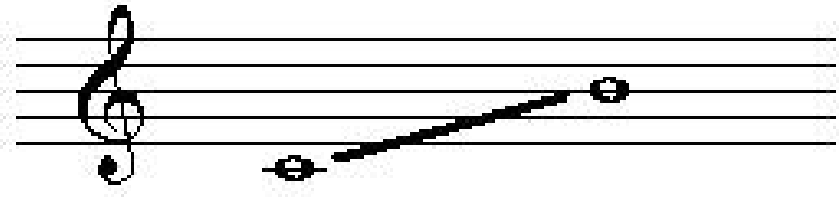
Хармониската меморија на музичарот (HM) може да биде претставена како матрица [90]:

$$HM = \left[\begin{array}{cccc|c} x_1^1 & x_2^1 & \cdots & x_n^1 & f(x^1) \\ x_1^2 & x_2^2 & \cdots & x_n^2 & f(x^2) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_1^{HMS} & x_2^{HMS} & \cdots & x_n^{HMS} & f(x^{HMS}) \end{array}\right] \quad (8)$$

Претходно, вредностите на функцијата на оптимизација беа сортирани во редослед ($f(x^1) \le f(x^2) \le \ldots \le f(x^{HMS})$) во HM, но тековната структура повеќе нема потреба од тоа [90].

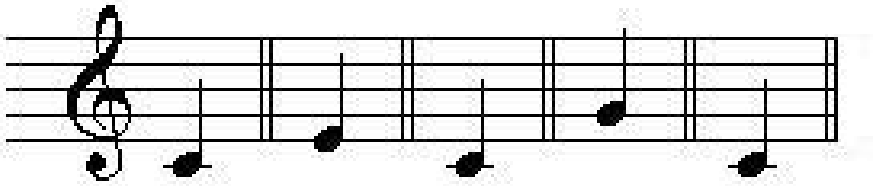
4.1.2.4. Импровизација на хармонија

Во џез импровизацијата, музичарот свири нота со нејзино случајно селектирање од вкупниот опсег кој може да се отсвири (сл. 3), од меморијата на музичарот (сл. 4), или со менување на нотата добиена од меморијата на музичарот (сл. 5). Аналогно, HS алгоритамот импровизира вредност со нејзино одбирање од вкупниот опсег на вредности или од HM, или со менување на вредноста што е оригинално добиена од HM [90].



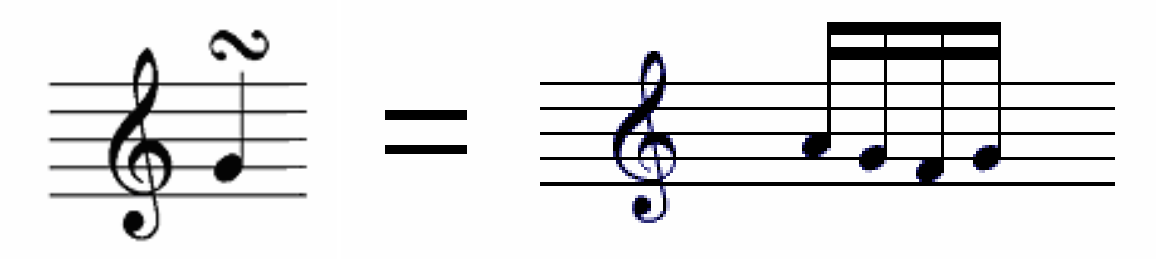
Слика 3. Тотален опсег на можни вредности.

Figure 3. Total range of possible values.



Слика 4. Множество на добри ноти во меморијата на музичарот.

Figure 4. A set of good notes in the memory of the musician.



Слика 5. Модифицирање на нотата добиена од меморијата на музичарот.

Figure 5. Modifying the note obtained from the musician’s memory.

Случајна селекција: Кога HS ја одредува вредноста x_i^{new} од новата хармонија $X^{new} = (x_1^{new}, \ldots, x_n^{new})$, случајно ја одбира било која од вредностите од тоталниот опсег на вредности $x_i \in X_i = \{x_i(1), ..., x_i(k), ..., x_i(K_i)\}$ или $x_i^L \le x_i \le x_i^U$ со веројатност од (1-HMCR). Случајната селекција исто така се користи за иницијализацијата на меморијата [90].

Разгледување на меморијата: Кога HS ја одредува вредноста x_i^{new}, случајно ја одбира било која вредност x_i^j од $HM = \{x_i^1, \ldots x_i^{HMS}\}$ со веројатност HMCR. Индексот *j* може да биде пресметан со користење на рамномерна распределба *U*(0,1) [90]:

$$j \leftarrow \text{int}(U(0,1) \cdot HMS) + 1 \qquad (9)$$

Како и да е, може да се користат различни распределби. На пример, ако се користи $[U(0,1)]^2$, HS повеќе го одбира пониското *j*. Ако вредностите на оптимизациската функција се сортирани по *j*, HS ќе се однесува слично на алгоритамот на оптимизација на множество честички [90].

Модулирање на фреквенциите: Откако вредноста x_i^{new} е случајно одбрана од HM во претходниот процес на разгледување на меморијата, може понатаму да биде променет во некој од соседните вредности со додавање на конкретна количина кон таа вредност, со веројатност од PAR. За дискретни променливи, ако $x_i(k) = x_i^{new}$, модулираните вредности стануваат $x_i(k+m)$ каде вообичаено m∈{−1, 1}; а за непрекинати променливи, модулираната вредност станува $x_i^{new} + \Delta$, каде вообичаено $\Delta = U(0,1) \cdot FW(i)$ [90].

Гореспоменатите три основни операции (случајна селекција, разгледување на меморијата и модулирање на фреквенциите) можат да се изразат на следниот начин [90]:

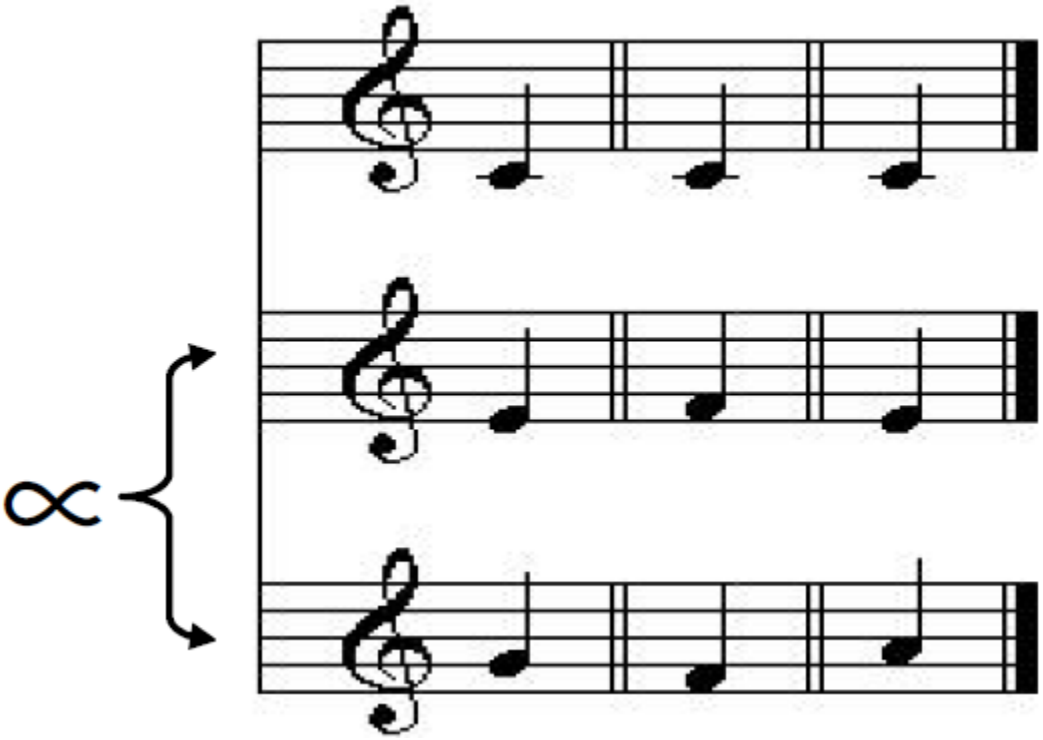
$$x_i^{new} = \begin{cases} \begin{cases} x_i \in \{x_i(1), \ldots, x_i(k), \ldots, x_i(K_i)\} \\ x_i \in [x_i^{lower}, x_i^{upper}] \end{cases} & \textit{веројатност} \quad (1 - HMCR) \\ x_i \in HM = \{x_i^1, \ldots, x_i^{HMS}\} & \textit{веројатност} \quad HMCR \cdot (1 - PAR) \\ \begin{cases} x_i(k+m) & \textit{ако} \quad x_i(k) \in HM \\ x_i + \Delta & \textit{ако} \quad x_i \in HM \end{cases} & \textit{веројатност} \quad HMCR \cdot PAR \end{cases} \tag{10}$$

Посебно за дискретни променливи, HS алгоритамот го има следниот стохастички парцијален извод кој се состои од три члена, како случајна селекција, разгледување на меморијата и модулирање на фреквенциите [90]:

$$\frac{\partial f}{\partial x_i} = \frac{1}{K_i} \cdot (1 - HMCR) + \frac{n(x_i(k))}{HMS} \cdot HMCR \cdot (1 - PAR) + \frac{n(x_i(k-m))}{HMS} \cdot HMCR \cdot PAR \tag{11}$$

Исто така, HS алгоритамот може да ја земе во предвид релацијата помеѓу променливите со користење на целосно разгледување, како што постои силна релација помеѓу конкретни музичари (сл. 6). Вредноста x_i^{new} може да се одреди врз основа на x_j^{new} ако двете имаат силна релација [99]:

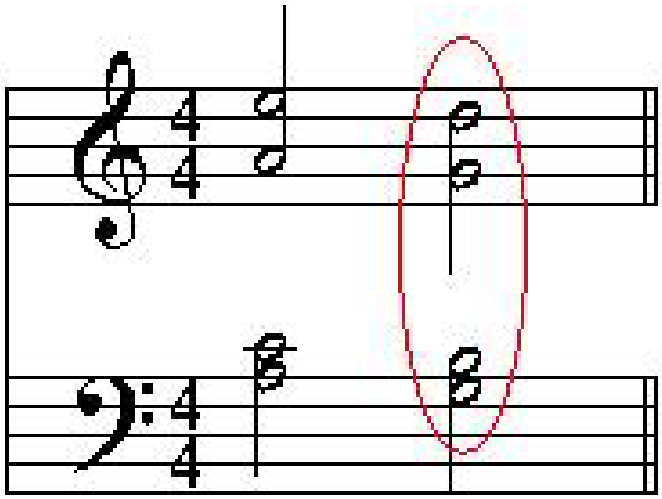
$$x_i^{new} \leftarrow fn(x_j^{new}) \quad \textit{каде} \quad \max_{i \neq j} \left\{ \left[Corr(x_i, x_j) \right]^2 \right\} \tag{12}$$



Слика 6. Поврзаност помеѓу одредени музичари.

Figure 6. Relationship between concrete musicians.

Ако новата импровизирана хармонија x^{new} нарушува некое од ограничувањата, HS го отфрла или го зачувува со додавање на казна кон вредноста на функцијата на оптимизација како што музичарите некогаш прифаќаат хармонија што нарушува некое правило (сл. 7) [90].



Слика 7. Хармонија што нарушува правило (паралелна петтина).

Figure 7. Rule-Violated Harmony (parallel fifth).

4.1.2.5. Ажурирање на меморијата

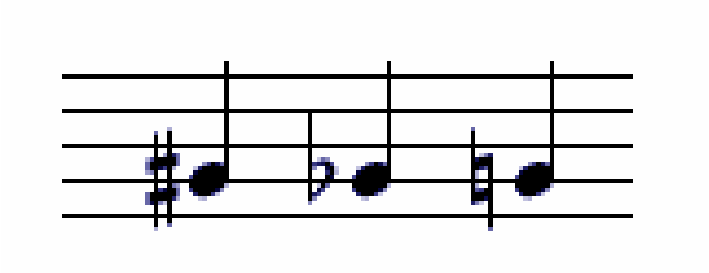
Ако новата хармонија x^{new} е подобра, во контекст на функцијата на оптимизација, отколку најлошата хармонија во HM, новата хармонија се вклучува во HM и најлошата хармонија се исклучува од HM [90]:

$$x^{new} \in HM \quad \wedge \quad x^{worst} \notin HM \qquad (13)$$

Како и да е, за да се добие различност на хармониите во HM, други хармонии (во контекст на најмала сличност) можат исто така да бидат разгледани. Исто така, максимален број на идентични хармонии во HM може да се земе во предвид со цел да се спречи прерана HM [90].

Ако новата хармонија x^{new} е најдобрата кога ќе се спореди со секоја друга хармонија во HM, новата хармонија може да вклучи нов процес наречен случајно извршување. Во музиката, предзнак е нота чија фреквенција не е член на скала и предзнакот ја повишува (#) или снижува (b) следната нота од нормалната фреквенција, како што е покажано на сл. 8. Слично, HS може понатаму да ја модулира фреквенцијата на секоја нота од новата хармонија ако е воопшто најдобрата хармонија, што може да доведе и до подобро решение [90]:

$$x_i^{new} \leftarrow \begin{cases} x_i(k \pm m), & x_i \text{ е дискретна променлива} \\ x_i + \Delta, & x_i \text{ е непрекината променлива} \end{cases}, \quad i = 1, \ldots, n \tag{14}$$



Слика 8. Предзнак на нотата сол.

Figure 8. Accidental for the note sol.

4.1.2.6. Терминација

Доколку HS ги задоволува критериумите за терминација (на пример, максимален број на итерации), пресметувањето се завршува. Инаку, HS повторно импровизира нова хармонија [90].

4.1.2.7. Каденца

Каденца е музичка секција која се појавува на крајот на изведба. Во контекст на HS алгоритамот, каденца претставува процес кој се појавува на крајот на HS пресметките. Во овој процес, HS ја враќа најдобрата хармонија присутна во HM [90].

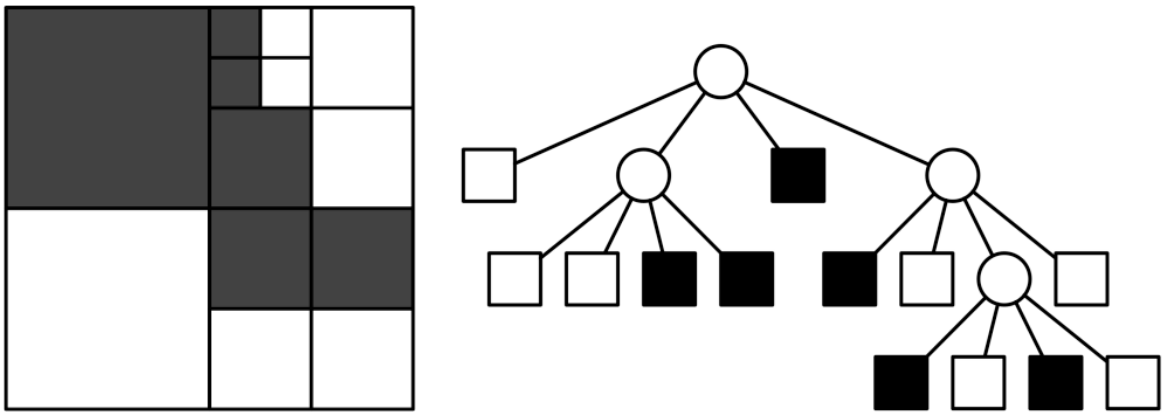
4.2. Quad-tree податочна структура

Quad-tree податочната структура е воведена од Рафаел Финкел и Џ. Л. Бентли во 1974 [100]. Во нивното истражување презентирале алгоритами кои се користат за конструирање и оптимизирање на квад-стебла и ја дискутирале комплексноста на двата пристапи: операции на вметнување и враќање. Денес, многу решенија во различни истражувачки области се базирани на нивната работа, некои од кои се презентирани во оваа секција [101].

Еден од примерите на користење на квад-стеблата е податочно рударење со слики кои користат квад-стебла како дел од машина за повлекување на податоци

што ефикасно може да ги спореди сликите добиени од сензорите. Во истражувањето приложено во [102] авторите опишуваат систем кој користи хиерархиска репрезентација за пресметување на спектрални и просторни својства на сликите. Доколку постојат две квад-стебла со просторна автокорелација од второ ниво, можно е да се искористат и двете својства (спектрални и просторни) својства на дадена слика на ефикасен начин. Друг пример поврзан со процесирање на слики е користењето на квад-стебла во визуелизацијата на мултирезолуциски слики складирани во магацин. Интерактивната визуелизација на складови од мултирезолуциски слики во 3Д [103] презентира пристап во кој секоја слика се состои хиерархиска пирамидална слика направена од мали делови на слика. Квад-стеблото овде е прилагодено да ги организира деловите и да ги репрезентира мултирезолуциските слики. Чен, Скерба и Ухран во истражувањето приложено во [104] пишува за пребарување на делови во околина со пречки, каде информацијата за пречките е обезбедена од надворешни сензори и патеката е пресметана при патувањето. Растојанието се пресметува со користењето на кружен бран на планирање на пат. Авторите предлагаат метод, кој користи техника на квад-стебла со рамки за репрезентирање на околината. Оваа техника ја комбинира точноста на планирањето на бат во регуларно структурирани околини и ефикасноста на квад-стеблата. Динамичкото партиционирање на системот и подоцнежната алокација на овие партиции за влезните задачи кои користат алгоритами базирани на алокација на процесори со квад-стебла се опишани во истражување напишано во 2005 година [105]. Квад-стебло се користи со цел да се репрезентира информацијата за алокацијата на под-јамки во поврзан систем од јамки. Како резултат, внатрешната фрагментација е минимизирана и се добива најголемиот возможен слободен под-систем. Оптимизацијата е друга област каде што се користат квад-стеблата. Истражување изложено во [106] претставува екстензија на традиционалната податочна структура на квад-стебло, што води кон заштеди во меморијата или просторот за складирање, како и во времето на извршување. Авторите прават широка споредба помеѓу овие две податочни структури и покажуваат значителни разлики. Покрај

горенаведените, постојат повеќе примени на овие податочни структури, а нови се истражуваат секоја година [101].



Слика 9. Црно-бела слика и нејзина репрезентација преку квад-стебло.

Figure 9. Black and white picture and her quad-tree representation.

Одредена слика може да биде зачувана во квад-стебло така што секој јазол има четири деца, секој од кои претставува квадрант (североисток, северозапад, југоисток и југозапад) на даден квадрат на соодветното ниво [107]. Секој квадрант се проверува дали сликата целосно го пополнува, делумно, или не го пополнува. Ако квадрантот е целосно пополнет, соодветниот јазол на квад-стеблото се смета за ,,црно". Ако квадрантот е делумно пополнет, соодветниот јазол се смета за ,,сив". На крај, доколку не е пополнет, соодветниот јазол се означува како ,,бел". Црните и белите јазли стануваат листови, додека сивите се делат на четири под-квадранти. Овој процес продолжува сè додека сите листови не се означени со ,,црно" или ,,бело", или не е достигнато одредено ниво кое се вика резолуција (или висина) на квад-стеблото (на сл. 9 висината r = 3). Постојат најмногу r нивоа на поделба. Подлабоки разгранувања се наведени во [108] и [109].

Варијанта, наречена линеарно квад-стебло е предложена во [110]. Секој лист на линеарното квад-стебло е претставено со подреден пар, *(n, l)*, каде *n* е неговата просторна адреса наречена локационен код и *l* е неговото ниво. Нивото на јазолот корен е 0, на неговите четири деца е 1, итн. Линеарно квад-стебло е листа на парови код/ниво на сите ,,црни" јазли.

Наоѓањето на соседниот јазол на конкретен лист на стеблото е фундаментална операција на многу алгоритами кои манипулираат податочни структури од типот на квад-стебла. Кај квад-стеблата, наоѓањето на соседи одзема *O(r)* време на пресметување за најлошиот случај [109]. Шрак [111] предложи алгоритам со константно време за пронаоѓање на соседни јазли со иста големина во линеарни квад-стебла. Неговиот алгоритам ги пресметува само локационите кодови на соседните јазли со иста големина, без одредување на нивното постоење. За да се пронајдат локационите кодови на соседни јазли со различна големина, потребно е време на пресметување пропорционално на разликата во нивоата на овие соседни јазли (односно најмногу *O(r)*), што општо е потребно за пребарување на листата на локациони кодови на даденото линеарно квад-стебло. Во [112], предложен е нов алгоритам за пронаоѓање на соседните јазли на даден лист во квад-стеблото, што побарува само *O(1)* (односно, константно) време на пресметување за најлошиот случај. Понатаму, алгоритамот не бара разгледување на постоењето или непостоењето на соседни јазли. Според тоа, не се потребни дополнителни проверки.

4.3. Quad Harmony Search (QHS) алгоритам

Имајќи во предвид дека растојанието помеѓу почетната и крајната точка треба да биде минимизирано со алгоритамот со помош на кој агентот за пребарување се извршува, проблемот на глобално планирање на пат кај регуларно структурирани околини може да биде дефиниран како проблем од линеарно програмирање [113]. Доколку е даден граф *(G, A)*, почетен јазол *s*, краен јазол *t*, и цена w_{ij} за секоја конекција *arc(i,j)* што постои во *A*, проблемот може да биде дефиниран на следниот начин:

$$\min \sum_{ij \in A} \omega_{ij} x_{ij} \tag{15}$$

каде $x \geq 0$ и за секое *i*:

$$\sum_i x_{ij} - \sum_j x_{ji} = \begin{cases} -1, & ако\ i = s \\ 1, & ако\ i = t \\ 0, & инаку \end{cases} \quad (16)$$

Во рамките на ова магистерска теза, се воведува нов пристап кон решавањето на проблемот на глобално планирање на пат во регуларно структурирани околини за мобилни роботи кој се базира на употребата на HS алгоритамот. Регуларно структурните околини кои се испитувани за овие цели имаат правоаголна форма, т.е. се работи за ќелии со квадратен облик со фиксна должина и висина, и мобилниот агент беше способен да се движи во четири насоки, означени како север (С), исток (И), југ (Ј) и запад (З). За оптимизационите цели на алгоритамот, се користи техника на редуцирање на просторот на пребарување со цел да се забрза процесот на наоѓање на решение и да се обезбеди подобра конвергенција во помал број на итерации и со помала големина на меморијата. Оваа техника широко се користи во проблеми од областа на геометријата, но за главните цели на овој труд, се докажа дека е од особено значење во контекст на временска и просторна комплексност. Се работи за гореспоменатиот алгоритам, т.е. алгоритамот на квад-стебла.

Алгоритамот на квад-стебла (КС) се заснова на техника на поделба која го дели просторот на пребарување на четири подпроблеми, а потоа се повикува себеси рекурзивно на овие подпроблеми. Алгоритамот сопира со рекурзивните повици кога ќе се задоволи одреден дефиниран критериум. За главните цели на ова истражување, КС техниката се користи на таков начин што означува (обојува) определени квадратчиња во просторот на пребарување и полињата кои се означени со истиот број (боја) се третираат како поединечен јазол. Техниката е прилагодена на ова истражување на следниот начин:

1. Процесирај ја на влез околината од интерес.

2. Ако големината на околината е со димензии 1x1, тогаш провери дали е празна. Доколку е празна, означи ја со број (боја), инаку означи ја со бројот -1.

3. Подели го просторот на пребарување на четири квадратни под-простори.

4. Провери дали секој од под-просторите содржи најмалку една препрека.

А) Ако ниеден од под-просторите нема препреки, означи ги сите со ист број (боја).

Б) Ако два соседни под-простори немаат препреки, означи ги со ист број (боја). За останатите два под-простори, повтори го процесот рекурзивно од чекор 2.

В) Ако А) и Б) не се исполнети, провери дали било кој од под-просторите содржи препреки. Оние што не содржат препреки се означуваат со ист број (боја). За останатите, процесот се повторува рекурзивно од чекор 2.

Извршувањето на овој алгоритам на пример околина и неговиот резултат се прикажани на слика 10. КС фазата на алгоритамот е прикажана преку дијаграм на активности на сл. 11, додека главната фаза на QHS алгоритамот е прикажана на сл. 12.

Откако околината е означена со соодветните броеви, се конструира редуциран граф на поврзаност со користење на поделбата од претходната фаза и се конструира листа на соседство со користење на соседните правоаголници (квадрати). Овој пристап е прилично ефикасен кога станува збор за пребарување во околини кои имаат многу слободни области и процентот на препреки во околината е релативно мал.

Откако е изградена листата на соседство, соодветните влезни податоци се спремни да се пренесат како влез на HS алгоритамот. Наредниот чекор е да се дефинира функцијата на оптимизација, која се користи за остварување на конвергенција кон посакувано и прифатливо решение. Во секоја итерација од евалуацијата, имплементиран е следниот процес:

1. Дефинирај променлива *prevRect* за следење на претходно посетениот јазол.

2. Постави го *prevRect* на почетниот јазол од векторот кандидат.

3. За секој член на векторот:

А) Ако тековниот јазол е дестинацискиот јазол, зголеми ја вредноста на функцијата за оптимизација за константна вредност (во овој случај еден) и заврши ја итерацијата.

Б) Декларирај променлива *now* и постави ја на вредноста на *prevRect*.

В) Од сите соседи на тековниот правоаголник (јазол), одбери го следниот правоаголник за испитување на следниот начин: остатокот од делењето на следната вредност во кандидат векторот со бројот на соседи на тековниот правоаголник. Овој број е индексот на следниот правоаголник (јазол) во листата на соседство.

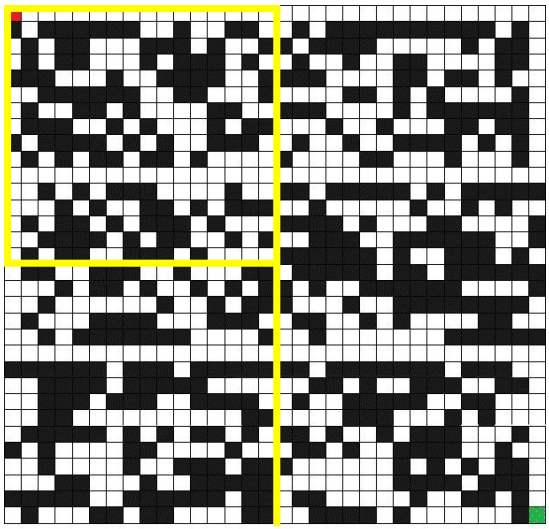
Г) Зголеми ја вредноста на функцијата за оптимизација за константна вредност (или еден) и доколку не е достигнат крајниот јазол, врати се на чекор 3.А).

Според тоа, со претходно опишаниот алгоритам, формулата за функцијата на оптимизација спремна за минимизирање може едноставно да биде дефинирана на следниот начин:

$$f(.) = \sum_{d \in D} PathCost(d) \tag{17}$$

каде d е насоката во која оди алгоритамот (јазол во околината), D е множеството на насоки (јазли) дадени со кандидат векторот кој е случајно генериран (со крајниот јазол како краен елемент), и *PathCost(d)* е предефинирана цена за изведување на насока на движење d и може да се дефинира како:

$$PathCost(d) = \begin{cases} 1, & d \text{ е пред крајниот јазол} \\ 0, & d \text{ е после крајниот јазол} \end{cases} \tag{18}$$



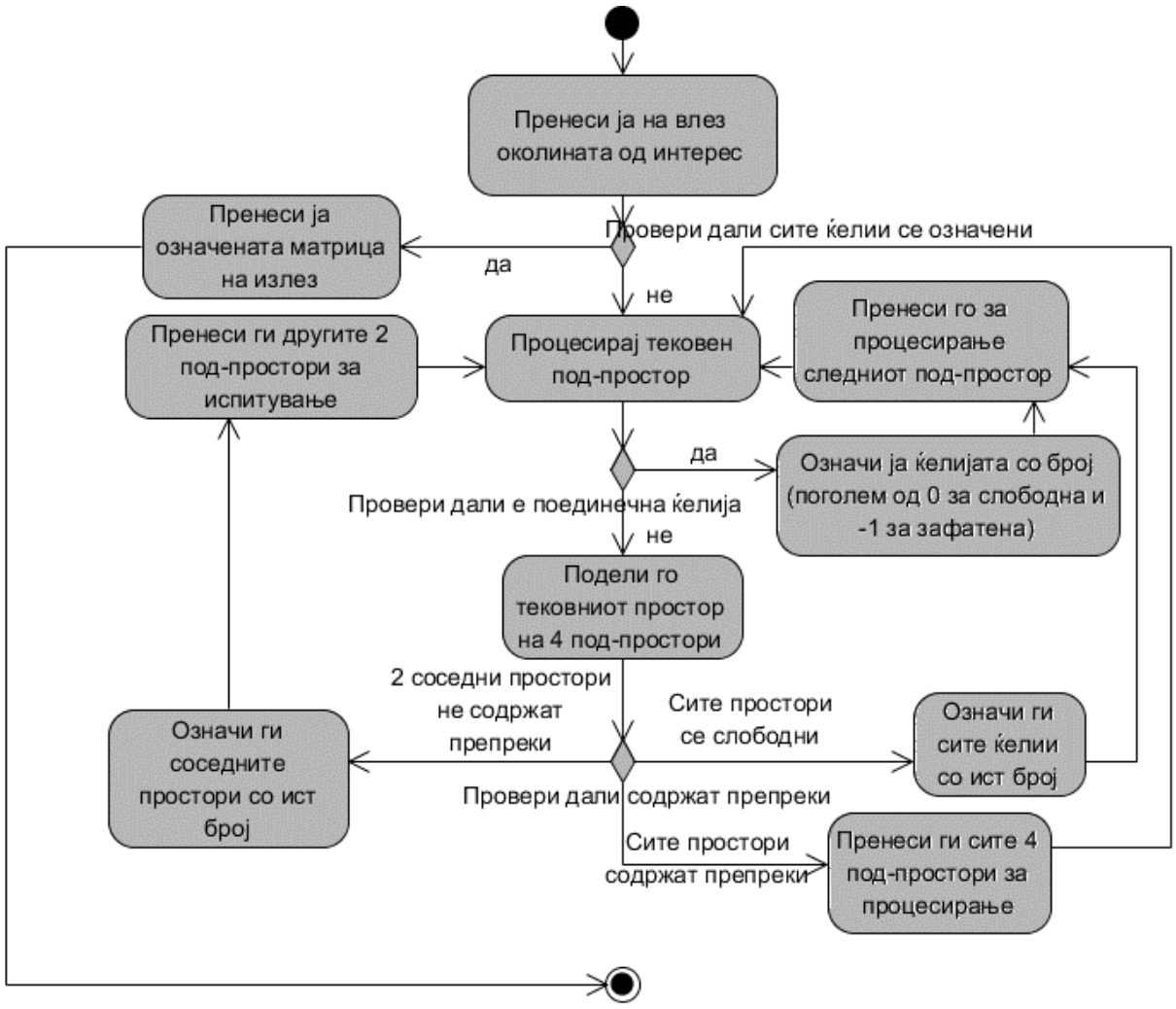
(а)

1	1	3	3	6	6	7	7	19	19	20	20	24	24	26	26
-1	2	-1	-1	-1	-1	-1	-1	19	19	-1	21	25	-1	-1	27
4	-1	5	-1	-1	8	9	9	-1	-1	-1	23	-1	28	29	-1
4	-1	5	-1	-1	8	9	9	-1	22	-1	-1	-1	28	-1	30
-1	-1	-1	11	14	14	-1	15	31	-1	-1	-1	-1	36	38	-1
10	10	-1	-1	-1	-1	-1	-1	31	32	-1	-1	37	36	38	-1
12	-1	13	13	-1	-1	17	-1	33	33	35	35	-1	-1	40	40
12	-1	-1	-1	16	16	-1	18	-1	34	35	35	-1	39	-1	-1
-1	41	-1	-1	-1	-1	48	-1	65	-1	66	66	-1	-1	-1	72
42	-1	43	-1	47	-1	-1	49	-1	-1	67	-1	71	71	73	72
44	44	45	45	50	50	52	52	68	68	70	70	74	74	76	76
44	44	-1	46	-1	51	-1	-1	-1	69	70	70	75	-1	76	76
53	53	55	-1	59	-1	61	61	-1	-1	77	-1	82	-1	-1	-1
54	-1	55	-1	-1	60	-1	62	-1	-1	-1	78	-1	83	84	84
56	56	-1	-1	-1	-1	64	-1	79	-1	-1	80	85	-1	87	-1
57	-1	58	-1	-1	63	64	-1	-1	-1	81	-1	85	86	87	88

(б)

Слика 10. Горниот лев квадрат означува почеток, а долниот десен квадрат дестинација. (а) Околината од интерес која се пренесува на влез. (б) Означената околина на горниот лев квадрант позициониран во горниот лев агол.

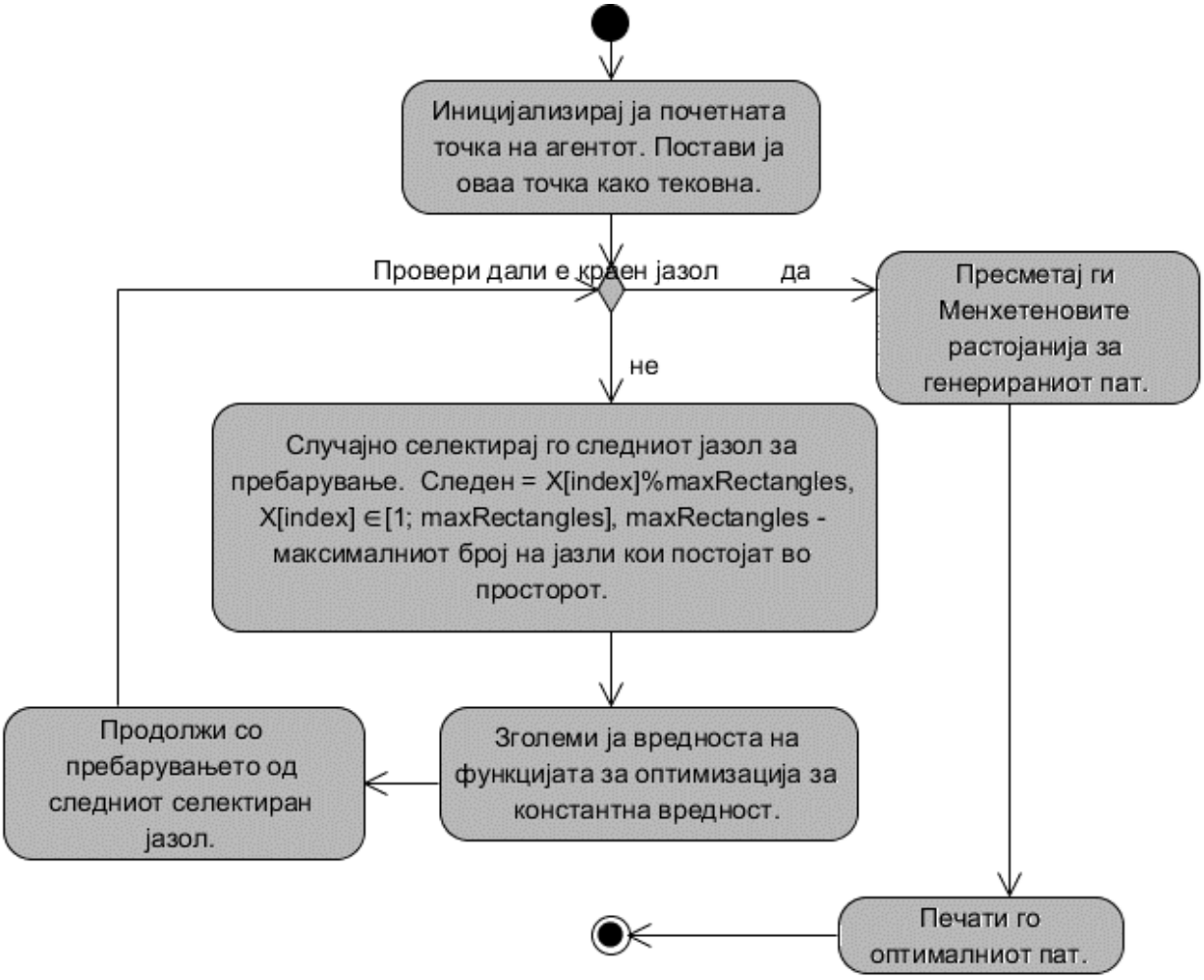
Figure 10. Upper left square indicates start, whereas lower right square the destination. (a) The grid of interest which is given as input. (b) The labeled grid of the upper left quadrant sub-space.



Слика 11. Дијаграм на активности за КС фазата на QHS алгоритамот.

Figure 11. Activity diagram of the QY stage of the QHS algorithm.

Кога е достигнат крајот на HS стадиумот, се конструира патеката од најдобриот кандидат вектор во меморијата на алгоритамот. Ова се прави со вчитување на позициите на полињата кои се наоѓаат во горниот лев агол на правоаголниците и димензиите на секој од правоаголниците претходно добиени со КС стадиумот на алгоритамот и со едноставно пресметување на минималното Менхетеново растојание помеѓу нив, што значително ги подобрува перформансите на алгоритамот.



Слика 12. Дијаграм на активности за главната фаза на QHS алгоритамот.

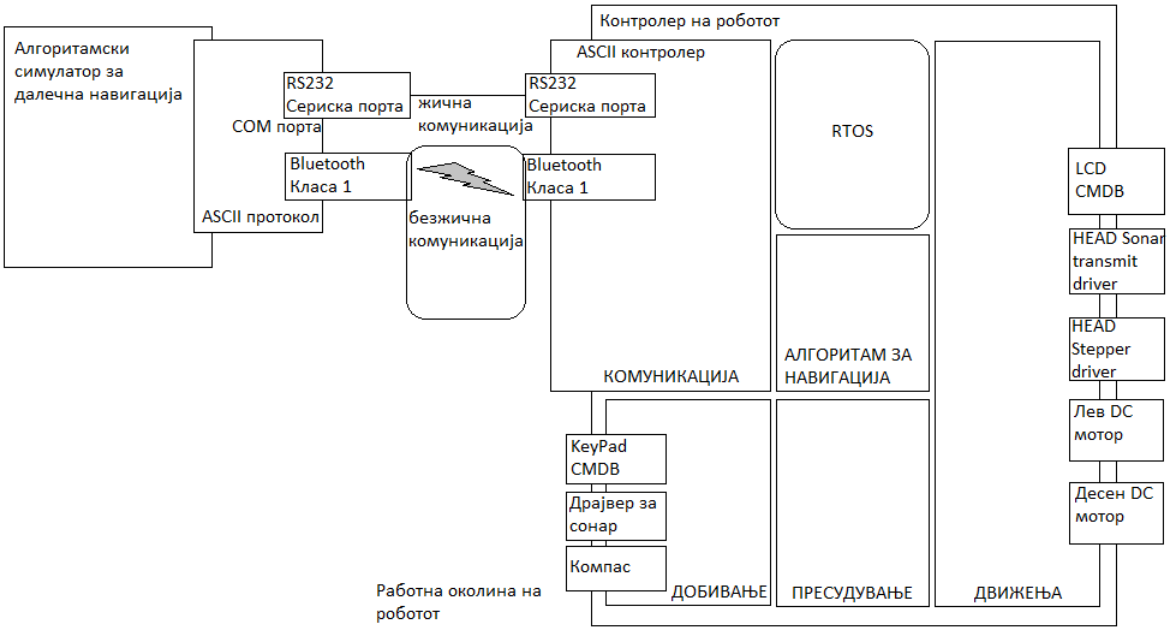
Figure 12. Activity diagram of the main stage of the QHS algorithm.

5. РЕЗУЛТАТИ

5.1. Опис на основните компоненти на архитектурата на системот за тестирање

Основната цел на експерименталната евалуација е да ја потврди способноста за генерирање на пат кој е оптимален и слободен од колизии на развиениот алгоритам. За експерименталната евалуација на развиениот алгоритам беше користена специфична лабораториска поставка за планирање на движењето.

За овие цели, покрај фазата на планирање на пат, системот исто така мора да биде способен да изведува локализација на роботот и контрола на движењето. Системот се состои од три главни компоненти: мобилен робот, контролна работна станица и камери монтирани на плафонот за локализација на роботот, пречките и целта. Архитектурата на развиениот тест систем е прикажан на сл. 13.



Слика 13. Архитектура на системот за тестирање.

Figure 1. Architecture of the testing system.

Мобилниот робот на три тркала користен во ова истражување е модифицирана верзија на ARobot [114]. Роботот содржи еден Basic Stamp II контролер од Paralax [115], и два копроцесори: PIC16F84 за контрола на моторот. Роботот ги има

следниве сензори: сонарен сензор, два светлосни сензори, температурен сензор, сензори на допир, PIR пасивен инфрацрвен детектор на движење, дигитален компас, R283-HOKUYO-LIDAR и конвертор на звучен излез. Роботот има 12 волтни DC подвижни мотори. Овие мотори се независно регулирани со користење на PWM контролирани H - мостови. Роботот, исто така, има оптички кодери што овозможуваат определување на брзината и позицијата на тркалата на роботот. Сите овие компоненти се сместени во мала алуминиумска конструкција со димензии: 10" x 10", 5" висина и капацитет на носивост од 1.36kg.

Мобилниот робот е поврзан со командна станица преку безжичен линк – со користење на RF двонасочни радио модули. Командната станица извршува програм што е одговорен за локализација на роботот, планирање на пат и планирање на контрола на движењето.

Со оглед на димензиите на работната станица и потребата од контрола во реално време, беше имплементиран систем базиран на повеќе камери за задачата на локализација. Работната околина беше виртуелно поделена на четири преклопувачки региони и секој од нив беше целосно покриен со веб камера монтирана на плафонот со поле на видливост од 90 степени (сл. 2), испраќајќи видеа во реално време со резолуција од 640x480 кон командната работна станица за понатамошно процесирање. На овој начин, сите преки (означени бела и зелена боја), целта (означена со сино) и роботот означен со QR код беа сликани со задоволувачка резолуција, па според тоа беа лесни за детекција. Задачата на процесирање на слики се изведуваше на контролната работна станица од модулот за локализација што го зема видеото како влез, ги детектира препреките и ја генерира мапата на регуларно структурираната околина. Препреките беа детектирани користејќи алгоритам базиран на филтрирање на бои и детекција на рабови со грешка на локализација помала од 2cm.

Откако е креирана, мапата на регуларно структурираната околина е испратена до модулот за планирање на пат кој го користи развиениот Quad Harmony Search алгоритам за да се пронајде патот од почетната до крајната точка. На крајот, со цел да се контролира мобилниот робот, контролниот програм кој се извршува на

работната станица испраќа директни команди до актуаторите на роботот за да се изведе конкретното движење.

5.2. Резултати од испитување на статички околини

Quad Harmony Search алгоритамот, како и алгоритамите со кои истиот е спореден, имено алгоритамот на колонија на мравки и генетскиот алгоритам, беше имплементиран во програмскиот јазик Јава и тестиран на компјутерска машина со Intel® Core i5 процесор со фреквенција од 2.53 GHz, 4GB RAM, и 64 битен Windows 7 оперативен систем. На почетокот, беше извршен едноставен тест на примерот прикажан на слика 10. Дадената околина беше тестирана со користење на три метаевристички алгоритами, имено: Quad Harmony Search, алгоритамот на колонија на мравки и генетскиот алгоритам. Како што покажуваат резултатите во Табела 1, може лесно да се воочи дека QHS доби најдобар резултат во споредба со другите алгоритами истовремено добивајќи оптимално решение, додека генетскиот алгоритам даде најслаби резултати во контекст на време и обезбедување на оптималното решение.

Табела 1. Време на извршување на QHS за примерот на сл. 1.

Table 1. Time of execution of QHS for the example on fig. 1.

Алгоритам/Algorithm	**Време (ms)/Time (ms)**
QHS	99
Ant Colony	10063
Genetic algorithm	172542

Во сите експерименти на евалуација користени се следните вредности за QHS параметрите: HMCR со вредност од 0.9, FW со вредност од 0.2, PAR со вредност од 0.4, големината на кандидат векторот е еднаков на бројот на означени правоаголници (јазли), големината на HM е поставена на 10 редови и бројот на итерации (критериумот за завршување) е поставен на максимум 50. За алгоритамот на колонија на мравки користени се следните параметри: број на мравки со вредност од 1000, број на итерации со вредност од 100, зголемување на феромоните θ со вредност од 1, намалување на феромоните φ со вредност од 0.01, степен на феромонот α со вредност од 2, степен на хевристиката β со

вредност од 16 и значајност на хевристиката γ со вредност од 0.1. И на крај, за генетскиот алгоритам користени се следните параметри: големина на популација со вредност 50, број на генерации со вредност од 2000, тип на вкрстување со две точки, веројатност на вкрстување со вредност од 0.5, тип на мутација со превртување на битови и веројатност на мутација со вредност од 0.005. Сите алгоритами се тестирани со помош на гореспоменатата архитектура за експериментирање.

Откако се направени овие едноставни тестови, беа испитани и други регуларно структурирани околини со димензии од 8x8, 16x16, 32x32, 64x64 и 128x128 и беше земен во предвид различен процент на препреки во околините, почнувајќи од 10% и завршувајќи со 90% со чекор од 10%. Овие тестови беа извршени со QHS, алгоритамот на колонија на мравки и генетскиот алгоритам и резултатите беа споредени за да се добијат валидни заклучоци. Резултатите од пребарувањата со користење на овие три алгоритами се дадени во милисекунди и се претставени на Табела 2, Табела 3 и Табела 4, соодветно. Истите се претставени преку графикони на сл. 14, сл. 15, сл. 16, сл. 17 и сл. 18 за димензии од 8x8, 16x16, 32x32, 64x64 и 128x128 соодветно.

Анализирајќи ги резултатите добиени од QHS алгоритамот може лесно да се заклучи дека при помал процент на препреки (10%-20%) се добива побрзо извршување, со оглед на фактот дека се користи КС алгоритамот за да се определат слободните области во графот на регуларно структурираната околина. Ист е случајот и со поголем процент на препреки (70%-90%). Како и да е, во опсегот помеѓу 30% и 60% покриеност на препреки добиваме поголемо време на извршување. Ова може да се објасни со поголемиот број на правоаголници (јазли) што има потреба да се испитаат, па според тоа добиваме поголема меморија (НМ), повеќе можности за испитување, и сите овие фактори водат до поголем период за пребарување на оптималното решение.

Кога овие резултати од QHS ќе се споредат со алгоритамот на колонија на мравки и генетскиот алгоритам, можат да се заклучат неколку значајни точки. Алгоритамот на колонија на мравки докажа дека секогаш го дава оптималното

решение, но имаше поголемо време на извршување отколку QHS во 100% од тест случаите (Табела 3). Со користење на генетскиот алгоритам, резултатите покажаа дека секогаш може да пронајде можно решение, но на испитаните тест случаи никогаш не го доби оптималното решение. Ова значи дека генетскиот алгоритам може лесно да заглави во локален оптимум и како последица на тоа да не го обезбеди оптималното решение. Исто така, во контекст на временско извршување споредено со QHS и алгоритамот на колонија на мравки, генетскиот алгоритам даде најслаби резултати.

Исто така, за сите тест случаи кои се користени за ова истражување, беше изведена и проверка на оптималноста на решенијата добиени од QHS. Оваа проверка беше направена со помош на познатиот Breadth First Search (BFS) алгоритам. Должините на оптималните патишта добиени со Breadth First Search и Quad Harmony Search алгоритамите беа исти за сите тест случаи, што може да се види од Табела 5.

Табела 2. Тестови за различни големини на околини и различен процент на покриеност со препреки со користење на Quad Harmony Search. Резултатите се прикажани во милисекунди.

Table 2. Tests for different grid sizes and different percentages of obstacles by using the Quad Harmony Search. The results are shown in milliseconds.

Процент на препреки/ Percentages of obstacles	**Димензии/Dimensions**				
	8x8	**16x16**	**32x32**	**64x64**	**128x128**
10%	13	22	55	370	5416
20%	15	12	66	663	4997
30%	14	25	86	927	9545
40%	14	26	60	727	8959
50%	15	19	55	114	957
60%	13	23	35	67	460
70%	12	24	20	46	401
80%	8	7	15	25	169
90%	7	14	13	23	129

Табела 3. Тестови за различни големини на околини и различен процент на покриеност со препреки со користење на алгоритамот на колонија на мравки. Резултатите се прикажани во милисекунди.

Table 3. Tests for different grid sizes and different percentages of obstacles by using the Ant Colony. The results are shown in milliseconds.

Процент на препреки/ Percentages of obstacles	**Димензии/Dimensions**				
	8x8	**16x16**	**32x32**	**64x64**	**128x128**
10%	786	2440	9501	16094	30903
20%	1102	2861	8896	20145	36472
30%	971	2007	7974	21538	33812
40%	598	2811	5441	18668	29171
50%	461	1702	5513	8640	22920
60%	349	1131	3215	7859	21174
70%	278	701	2701	7003	21854
80%	334	818	2150	6634	20801
90%	374	695	2585	6020	22563

Табела 4. Тестови за различни големини на околини и различен процент на покриеност со препреки со користење на генетскиот алгоритам. Резултатите се прикажани во милисекунди.

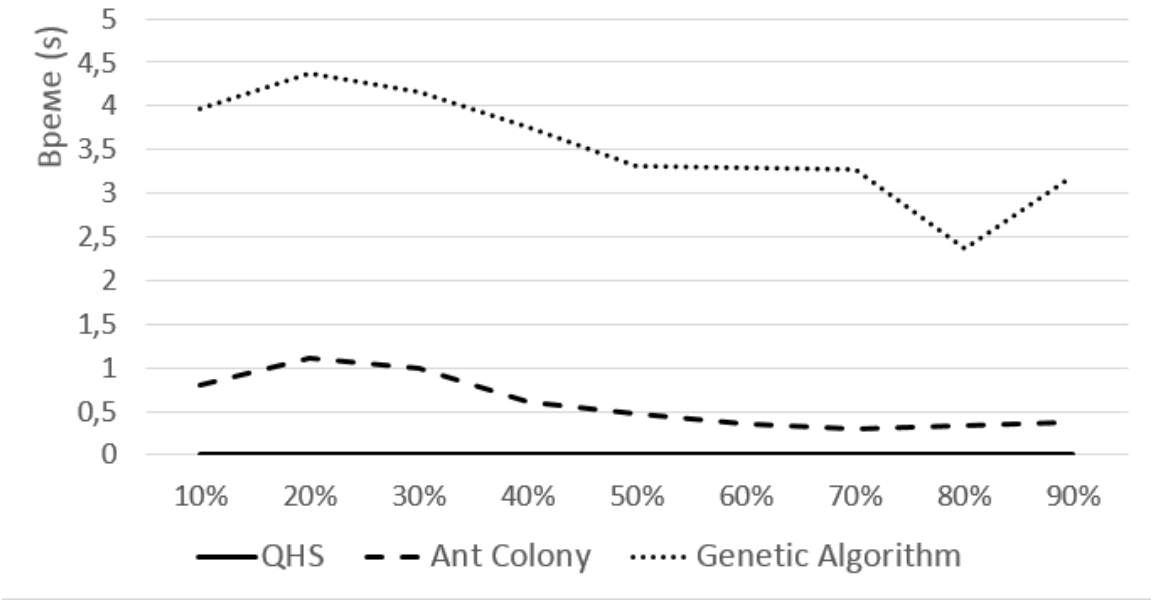
Table 4. Tests for different grid sizes and different percentages of obstacles by using the Genetic Algorithm. The results are shown in milliseconds.

Процент на препреки/ Percentages of obstacles	**Димензии/Dimensions**				
	8x8	**16x16**	**32x32**	**64x64**	**128x128**
10%	3171	9321	25540	1114136	4254514
20%	3260	6290	38588	576558	3695286
30%	3170	11200	33670	1607530	1242007
40%	3140	9080	31529	1619849	1513206
50%	2840	20040	41525	428831	1443848
60%	2930	20996	36724	279093	561349
70%	2990	13150	45172	267380	571141
80%	2030	13852	54988	514670	634857
90%	2820	15814	52294	268741	1616770

Табела 5. Должини на оптималните патишта добиени со користење на Breadth First Search и Quad Harmony Search алгоритамите.

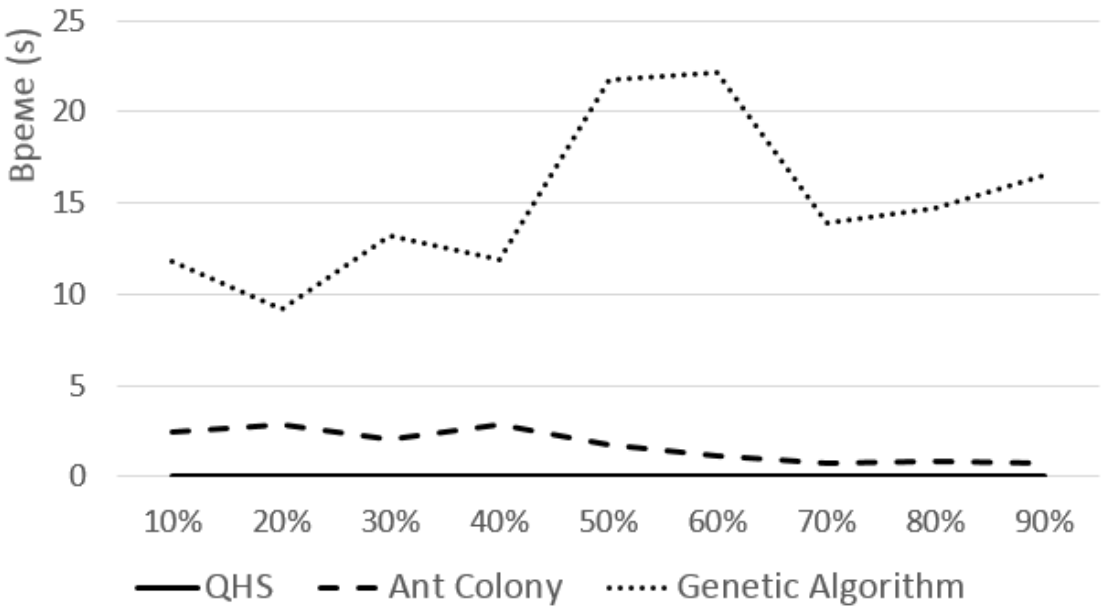
Table 5. Lengths of the optimal paths obtained by using Breadth First Search and Quad Harmony Search algorithms.

Процент на препреки/ Percentages of obstacles	**Димензии/Dimensions**				
	8x8	**16x16**	**32x32**	**64x64**	**128x128**
10%	14	30	66	133	74
20%	14	30	61	114	71
30%	16	28	66	100	89
40%	13	42	50	234	66
50%	9	23	51	38	53
60%	6	13	19	22	30
70%	3	7	9	11	47
80%	6	10	5	8	28
90%	5	7	7	4	59



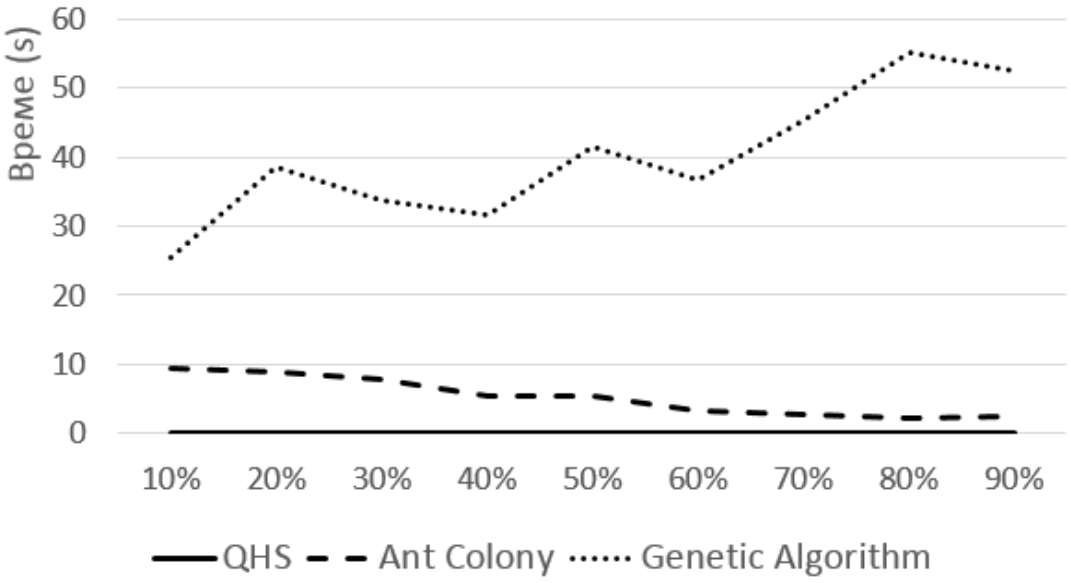
Слика 14. Споредба на QHS, Ant Colony и Genetic Algorithm за простори со димензии 8x8.

Figure 14. Comparison of QHS, Ant Colony and Genetic Algorithm for 8x8 grid sizes.



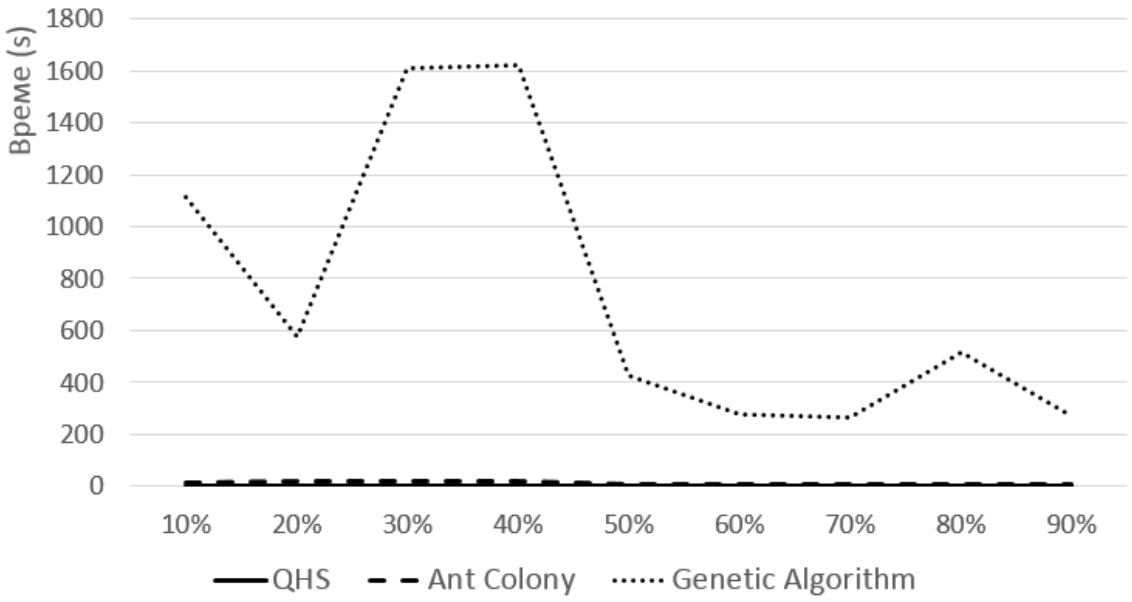
Слика 15. Споредба на QHS, Ant Colony и Genetic Algorithm за простори со димензии 16x16.

Figure 15. Comparison of QHS, Ant Colony and Genetic Algorithm for 16x16 grid sizes.



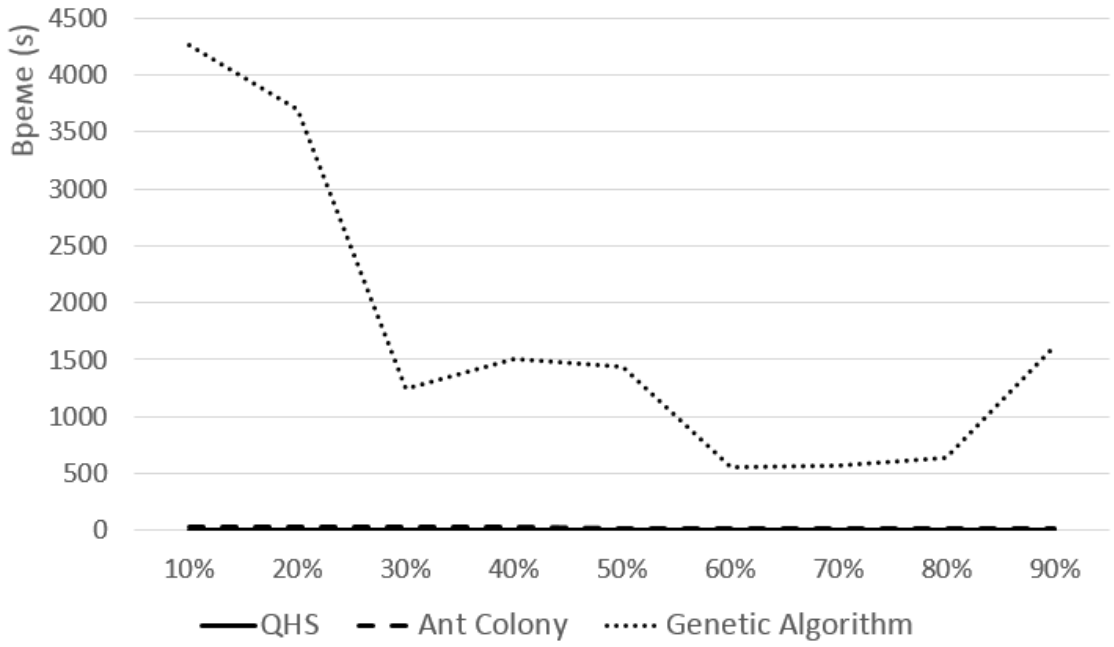
Слика 16. Споредба на QHS, Ant Colony и Genetic Algorithm за простори со димензии 32x32.

Figure 16. Comparison of QHS, Ant Colony and Genetic Algorithm for 32x32 grid sizes.



Слика 17. Споредба на QHS, Ant Colony и Genetic Algorithm за простори со димензии 64x64.

Figure 17. Comparison of QHS, Ant Colony and Genetic Algorithm for 64x64 grid sizes.



Слика 18. Споредба на QHS, Ant Colony и Genetic Algorithm за простори со димензии 128x128.

Figure 18. Comparison of QHS, Ant Colony and Genetic Algorithm for 128x128 grid sizes.

Воедно, беа тестирани и посебни типови на регуларно структурирани околини. Имено, станува збор за тип на околини во кои оптималната патека која треба да ја изведе агентот за пребарување има карактеристичен облик. Во овој случај се работи за облик на оптимална патека која ја има формата на македонската буква ,,П” (сл. 10). За целите на тестирање на алгоритамот, беа дизајнирани регуларно структурирани околини во кои покрај патеката во облик на буквата ,,П”, имаше дополнителни патишта, т.н. ,,мамки”, во кои се очекува метаевристичките алгоритами да заглават во извршување и да ги изведат и дополнителните околни патишта. Ова карактеристично однесување беше повторно тестирано со користење на трите горенаведени алгоритами, имено QHS, AC и GA. Резултатите од ова истражување се прикажани табеларно во милисекунди на Табела 6 и Табела 7 за димензии на околини од 16x16 и 32x32, соодветно. Истите се визуелно претставени и на графикони на сл. 19 и сл. 20 за димензии на околини 16x16 и 32x32, соодветно.

Табела 6. Резултати од испитувањето на околини во форма на буквата ,,П” за димензија на околини од 16x16.

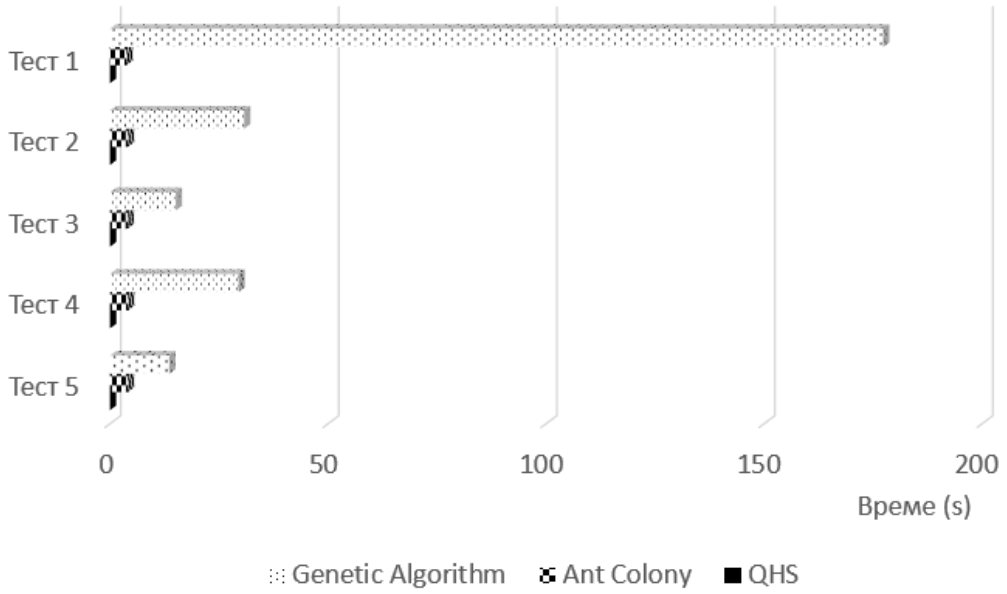
Table 6. Results from examination of grids in the form of letter ,,П” for 16x16 grid sizes.

Тест случаи/ Test cases	**Алгоритам/Algorithm**		
	Harmony Search	**Ant Colony**	**Genetic Algorithm**
Тест 1	23	3829	177323
Тест 2	17	4120	30801
Тест 3	34	4090	15106
Тест 4	34	4197	29522
Тест 5	19	4153	13613

Табела 7. Резултати од испитувањето на околини во форма на буквата ,,П” за димензија на околини од 32x32.

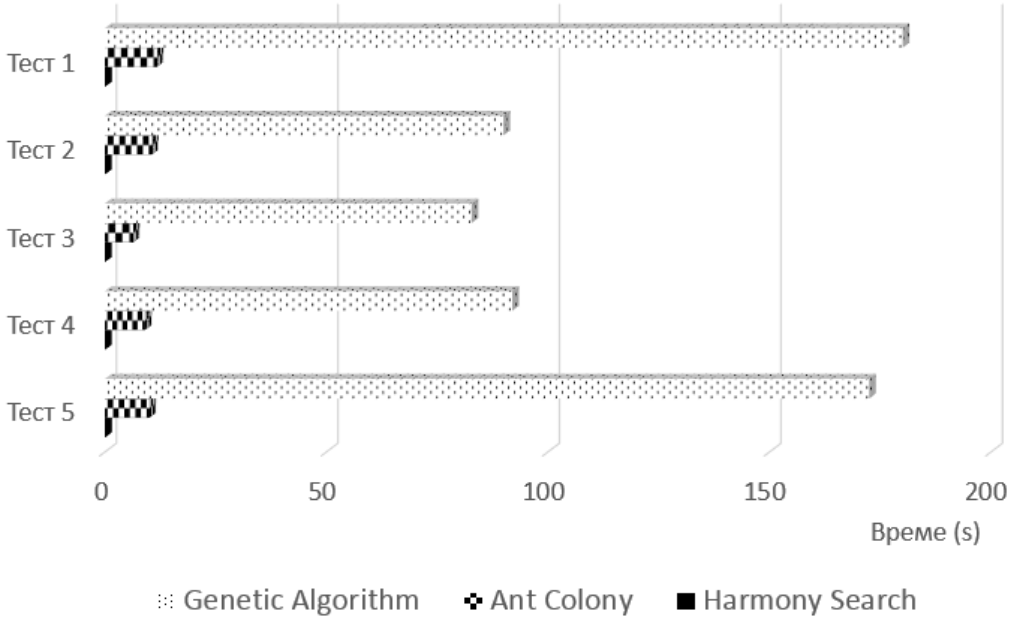
Table 7. Results from examination of grids in the form of letter ,,П” for 32x32 grid sizes.

Тест случаи/ Test cases	**Алгоритам/Algorithm**		
	Harmony Search	**Ant Colony**	**Genetic Algorithm**
Тест 1	105	11746	180097
Тест 2	98	10600	90000
Тест 3	73	6378	82815
Тест 4	93	9082	91982
Тест 5	99	10063	172542



Слика 19. Графички приказ на резултатите од Табела 6.

Figure 19. Graphical representation of the results shown in Table 6.



Слика 20. Графички приказ на резултатите од Табела 7.

Figure 20. Graphical representation of the results shown in Table 7.

5.3. Резултати од испитување на динамички околини

Quad Harmony Search алгоритамот беше тестиран и во рамките на динамичка околина. Имено, истите принципи на поделба на просторот на агентот и минимизирање на вредноста на функцијата на оптимизација беа применети. Меѓутоа, овде агентот има способност да прима податоци (на пример, преку сензори) од околината за моменталната состојба на полињата. На почетокот на агентот му е дадена мапа на околината која ја користи за своето првично пребарување. Потоа, агентот е способен во рамките на околната област на одредена далечина да види дали настанала промена во околината. Доколку не настанала промена во момент од движењето, се користат истите правила и принципи при оригиналното планирање. Во спротивно, се пресметува нова поделба на просторот, се гради нова листа на соседство на слободни региони и повторно се пресметува оптимален пат, при што во овој случај се зема како стартна позиција тековната позиција на роботот, а крајната точка останува иста. Во рамките на ова истражување, агентот беше способен да прими податоци за околината околу себе на далечина од 2 мерни единици во сите насоки, при што една мерна единица во овој случај е должината на една страна на едно поле во просторот на пребарување. Врз основа на горенаведените резултати, како регуларно структурирани околини за тестирање беа земени околините кои имаат процент на покриеност со препреки во опсег 30%-60%, кои се покажаа како најкритични и најкарактеристични случаи за тестирање. Беа тестирани димензии на околини од 8x8, 16x16, 32x32, 64x64 и 128x128. Алгоритамот QHS беше повторно спореден со перформансите на Ant Colony и Genetic Algorithm.

Формулата според која се пресметува времето на извршување на пребарувањето во динамичка околина е следната:

$$total = \sum_{s \in S} T(s) \tag{19}$$

Овде, *s* означува состојба на промена на околината, *S* е множеството на промени кои се случуваат при изведување на оптималниот пат, а *T(s)* е времето на извршување на алгоритамот во состојбата на промена *s*.

Резултатите од споредбата на QHS, AC и GA за динамични околини може да се видат на Табела 8, Табела 9, Табела 10, Табела 11 и Табела 12 за димензии на околини од 8x8, 16x16, 32x32, 64x64 и 128x128, соодветно. Истите визуелно се претставени на сл. 21-25.

Табела 8. Резултати од тестирање на динамички околини со димензии 8x8.

Table 8. Results from testing of dynamic environments of size 8x8.

Процент на препреки/ Percentage of obstacles	**Алгоритам/Algorithm**		
	QHS	**Ant Colony**	**Genetic Algorithm**
30%	29	1131	7515
40%	32	1076	13899
50%	34	846	12103
60%	31	887	14785

Табела 9. Резултати од тестирање на динамички околини со димензии 16x16.

Table 9. Results from testing of dynamic environments of size 16x16.

Процент на препреки/ Percentage of obstacles	**Алгоритам/Algorithm**		
	QHS	Ant Colony	Genetic Algorithm
30%	57	2042	20850
40%	64	1873	16526
50%	57	1440	11930
60%	49	1584	25571

Табела 10. Резултати од тестирање на динамички околини со димензии 32x32.

Table 10. Results from testing of dynamic environments of size 32x32.

Процент на препреки/ Percentage of obstacles	**Алгоритам/Algorithm**		
	QHS	Ant Colony	Genetic Algorithm
30%	127	5093	81020
40%	143	3879	60121
50%	129	2896	55558
60%	124	3337	51711

Табела 11. Резултати од тестирање на динамички околини со димензии 64x64.

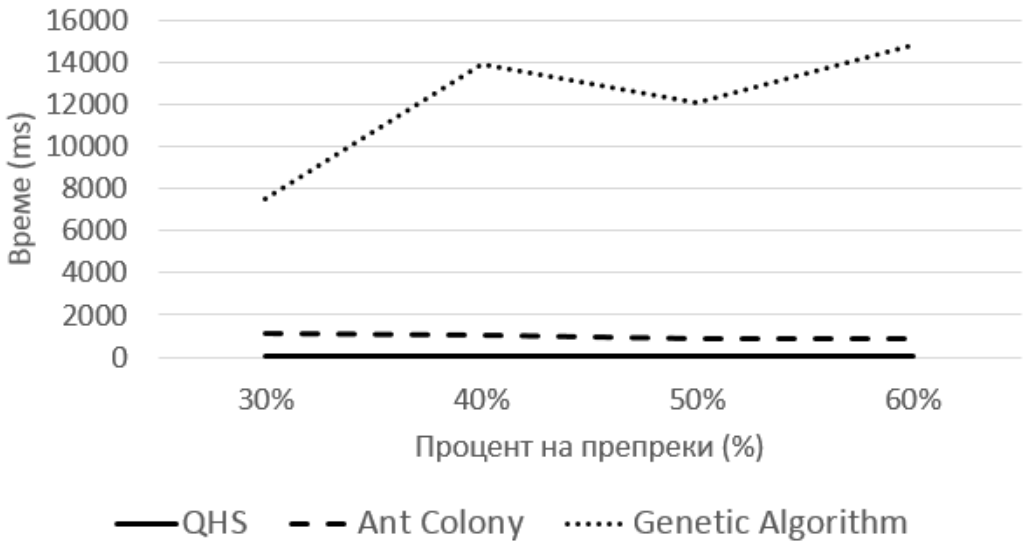
Table 11. Results from testing of dynamic environments of size 64x64.

Процент на препреки/ Percentage of obstacles	**Алгоритам/Algorithm**		
	QHS	Ant Colony	Genetic Algorithm
30%	701	12355	1320807
40%	822	12272	2395689
50%	451	7180	553800
60%	372	10071	789675

Табела 12. Резултати од тестирање на динамички околини со димензии 128x128.

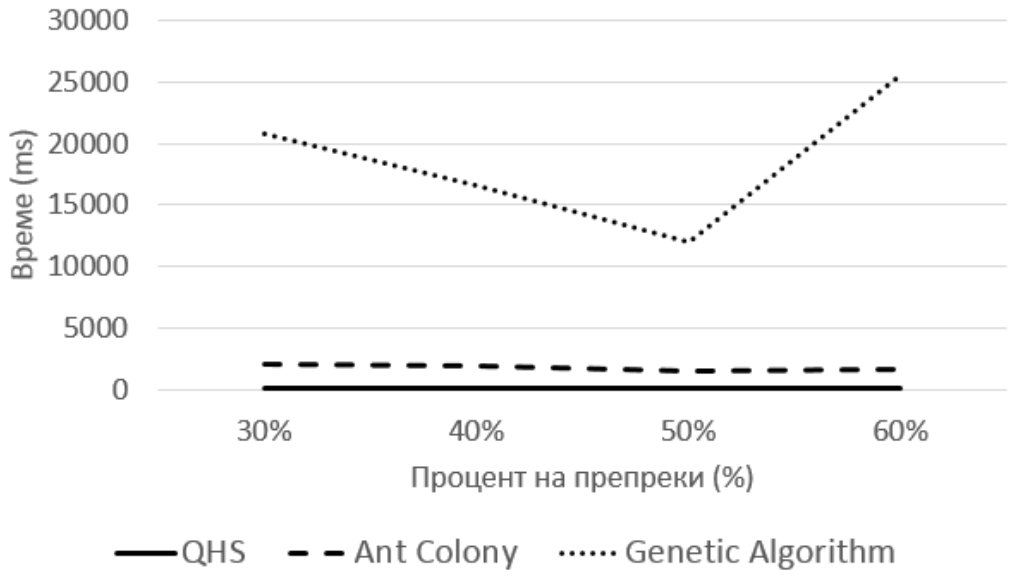
Table 12. Results from testing of dynamic environments of size 128x128.

Процент на препреки/ Percentage of obstacles	**Алгоритам/Algorithm**		
	QHS	Ant Colony	Genetic Algorithm
30%	6098	34506	7949800
40%	8914	33628	1949254
50%	3130	26775	1621883
60%	1517	40474	2748907



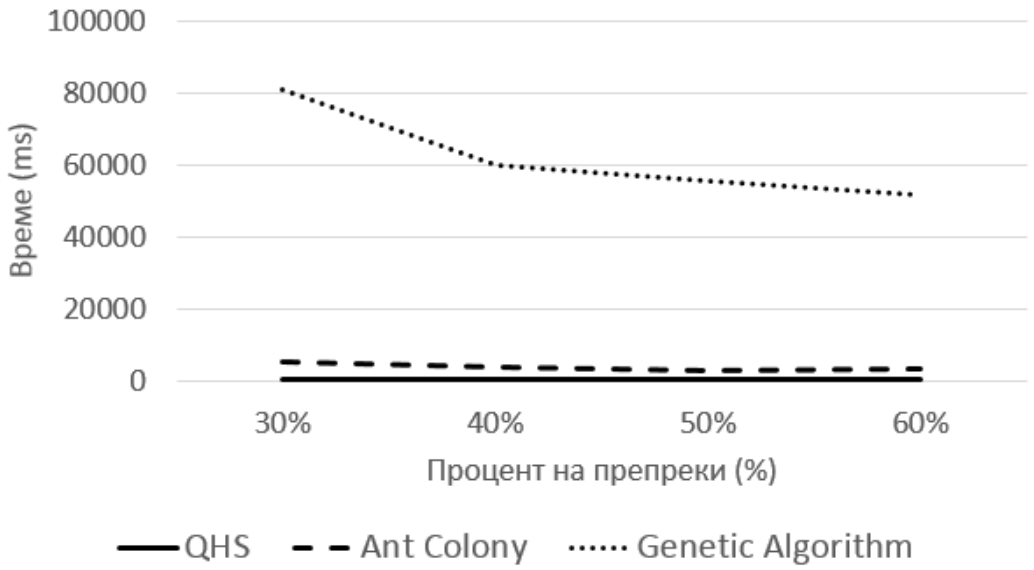
Слика 21. Споредба на QHS, AC и GA за динамички околини од димензии 8x8.

Figure 21. Comparison of QHS, AC and GA for dynamic environments of sizes 8x8.



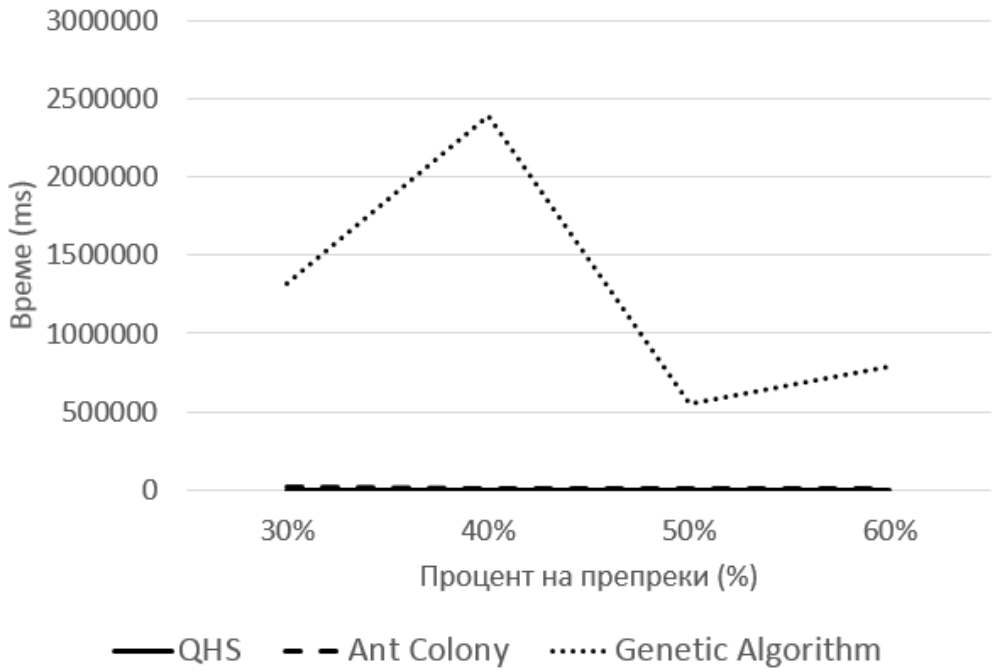
Слика 22. Споредба на QHS, AC и GA за динамички околини од димензии 16x16.

Figure 22. Comparison of QHS, AC and GA for dynamic environments of sizes 16x16.



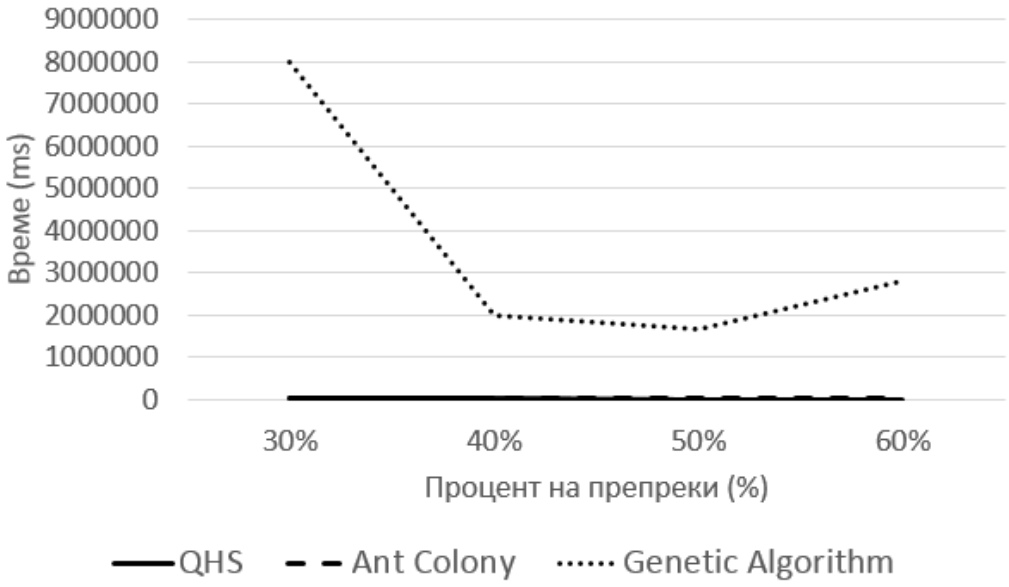
Слика 23. Споредба на QHS, AC и GA за динамички околини од димензии 32x32.

Figure 23. Comparison of QHS, AC and GA for dynamic environments of sizes 32x32.



Слика 24. Споредба на QHS, AC и GA за динамички околини од димензии 64x64.

Figure 24. Comparison of QHS, AC and GA for dynamic environments of sizes 64x64.



Слика 25. Споредба на QHS, AC и GA за динамички околини од димензии 128x128.

Figure 25. Comparison of QHS, AC and GA for dynamic environments of sizes 128x128.

6. ДИСКУСИЈА

Како што лесно може да се воочи од графиконот на сл. 19 за димензии на околини од 16x16, GA алгоритамот лесно заглавуваше во локални оптимуми, па според тоа и времето на извршување на алгоритамот беше најголемо во однос на другите алгоритами. Ова ги потврдува претходните резултати и лесно може да се заклучи дека GA алгоритамот е навистина несоодветен за примена на решавање на проблемот на глобално планирање на пат, пред сè поради неговата непредвидливост во однос на процентот на покриеност на околината со препреки.

За разлика од него, AC алгоритамот имаше потреба од многу помалку итерации и побрзо се снаоѓаше во однос на излегување од локален оптимум, па според тоа и времето на извршување беше многу пократко отколку она на GA алгоритамот.

На крајот, може лесно да се воочи дека на QHS алгоритамот му беше потребно најмалку време за извршување и добивање на глобалниот оптимум. Ова се должи, како што претходно беше потенцирано, на способноста за редуцирање на просторот на пребарување во QT фазата на алгоритамот, но и на оптимизационите способности на Harmony Search алгоритамот.

Исто така, од тестирањето на наведените алгоритами за динамички околини, резултатите јасно покажуваат дека QHS е многу пофлексибилен во однос на промените не околината и повторно му требаше многу помалку кумулативно време на извршување отколку Ant Colony и Genetic Algorithm.

Поради овие причини, на QHS алгоритамот му се потребни многу помалку итерации за извршување, што уште еднаш го потврдува фактот дека QHS е погоден алгоритам кој може да се примени на проблемот на глобално планирање на пат.

7. ЗАКЛУЧОК

Регуларно структурираните околини се од огромен интерес за истражувачите, зашто секоја околина од реалноста може да биде моделирана и дискретизирана со дводимензионален приказ, т.е. околина составена од подеднакво поделени региони кои содржат информација, како на пример дали дадено поле е слободно или не. При наоѓањето на пат на мобилен агент од почетна до крајна точка потребни се два типа на планери: глобален и локален планер. Додека глобалниот планер се занимава со наоѓањето на изводлив пат од стартната точка до дестинацијата, локалниот планер се занимава со локална навигација и избегнување на пречки. Знаејќи дека проблемот на глобално планирање на пат е NP – комплетен, т.е. не постои детерминистички алгоритам за негово решавање во реално прифатливо време, сѐ повеќе стануваат популарни метаевристичките алгоритами кои се применуваат на овој проблем.

Во рамките на оваа магистерска теза беше презентиран нов метаевристички пристап кон решавањето на проблемот на глобално планирање на пат, имено QHS алгоритамот. Овој алгоритам се заснова на способностите за поделба на просторот на пребарување на мобилниот робот со помош на Quad-tree алгоритамот и оптимизационите способности на популарниот Harmony Search алгоритам. Беа испитани различни големини на околини, како и околини со различен процент на зафатеност со препреки со цел да се испита ефикасноста на новосоздадениот QHS алгоритам. Експерименталните резултати беа изложени и елаборирани. Ова истражување даде цврсти аргументи дека новиот QHS алгоритам има многу подобри перформанси отколку другите метаевристички алгоритами применети на проблемот на глобално планирање на пат. Имено, QHS се покажа како подобар алгоритам од ACO и GA алгоритамите во контекст на време на извршување, брзината на конвергирање кон решенија и добивање на оптималност на решенијата. Според ова, овој алгоритам претставува одлична основа за натамошни истражувања на полето на примена на метаевристичките алгоритами за решавање на NP – комплетни проблеми, како што воедно беше

проблемот на глобално планирање на пат кај мобилни роботи објаснет во детали во рамките на овој труд.

8. ДОДАТОК

8.1. Користени кратенки

AC – Ant Colony

ACO – Ant Colony Optimization

APF – Artificial Potential Fields

CGA – Chaotic Genetic Algorithm

GA – Genetic Algorithm

HS – Harmony Search

PEGA – Parallel Elite Genetic Algorithm

PCPSO – Poly-Clone Particle Swarm Optimization

PSO – Particle Swarm Optimization

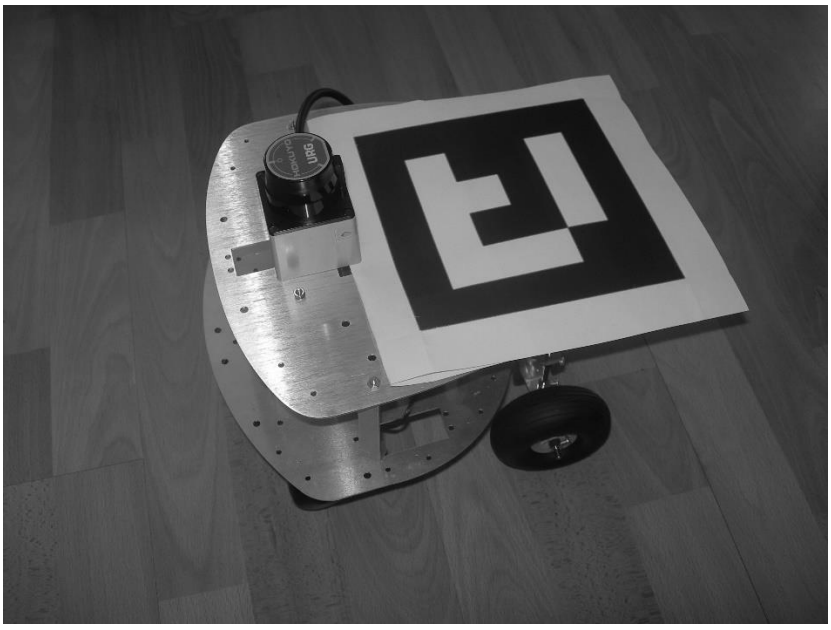
QHS – Quad Harmony Search

SA – Simulated Annealing

8.2. Слики од роботот користен за евалуацијата на резултатите од Quad Harmony Search алгоритамот

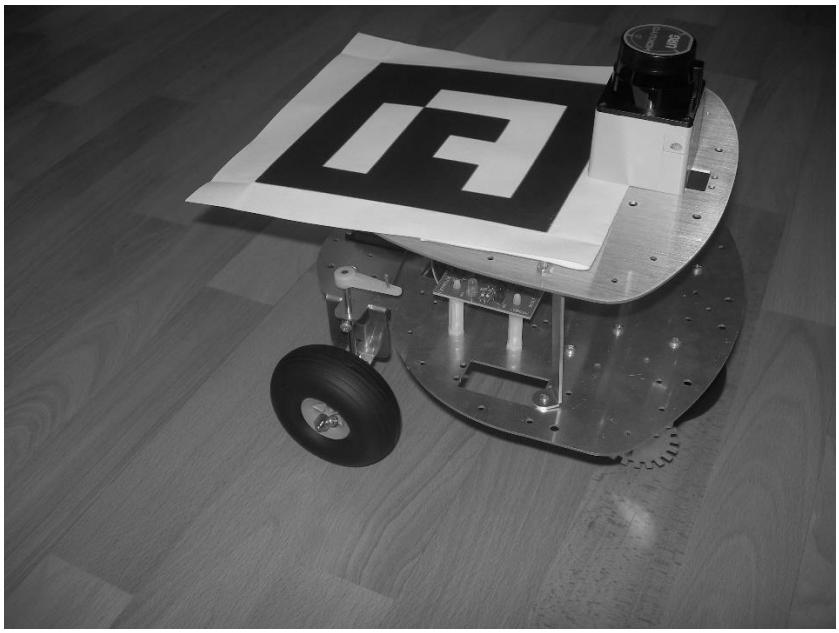
Слика 26. Работната околина на роботот.

Figure 26. Robot's working environment.



Слика 27. Роботот користен за евалуација (1).

Figure 27. The robot used for evaluation (1).



Слика 28. Роботот користен за евалуација (2).

Figure 28. The robot used for evaluation (2).

9. КОРИСТЕНА ЛИТЕРАТУРА (REFERENCES)

[1] Hachour, O. (2008). Path planning of Autonomous Mobile robot. International journal of systems applications, engineering & development, 2(4), 178-190.

[2] Howard, A., Matarić, M. J., & Sukhatme, G. S. (2002). An incremental self-deployment algorithm for mobile sensor networks. Autonomous Robots, 13(2), 113-126.

[3] Florczyk S. (2005). Robot Vision Video-based Indoor Exploration with Autonomous and Mobile Robots, WILEY-VCH Verlag GmbH & Co. KGaA, Weinheim.

[4] Gerkey, B. P., & Matarić, M. J. (2001). Principled communication for dynamic multi-robot task allocation. In Experimental Robotics VII (pp. 353-362). Springer Berlin Heidelberg.

[5] Saripalli, S., Sukhatme, G. S., & Montgomery, J. F. (2003). An experimental study of the autonomous helicopter landing problem. In Experimental Robotics VIII (pp. 466-475). Springer Berlin Heidelberg.

[6] Estrin, D., Culler, D., Pister, K., & Sukhatme, G. (2002). Connecting the physical world with pervasive networks. Pervasive Computing, IEEE, 1(1), 59-69.

[7] Willeke T., Kunz C., Nourbakhsh I. (2003). The Personal Rover Project : The comprehensive Design Of a domestic personal robot, Robotics and Autonomous Systems (4), Elsevier Science, pp.245-258.

[8] Moreno, L., Puente, E. A., & Salichs, M. A. (1993). World Modelling and Sensor Data Fusion in a Non Static Environment. Application to Mobile Robots. In IFAC SYMPOSIA SERIES (pp. 433-436). PERGAMON PRESS.

[9] Schilling, K., & Jungius, C. (1996). Mobile robots for planetary exploration. Control Engineering Practice, 4(4), 513-524.

[10] Hachour, O., & Mastorakis, N. (2004). IAV: A VHDL methodology for FPGA implementation. WSEAS Transactions on Circuits and Systems, 3(5), 1091-1096.

[11] Hachour, O., & Mastorakis, N. (2004, October). FPGA implementation of navigation approach. In WSEAS international multiconference 4th WSEAS robotics, distance learning and intelligent communication systems (ICRODIC 2004) (pp. 1-15).

[12] Hachour, O., & Mastorakis, N. (2004). Avoiding obstacles using FPGA-a new solution and application-. WSEAS Transactions on Systems, 3(9), 2827-2833.

[13] Hachour, O., & Mastorakis, N. (2001). Behaviour of intelligent autonomous ROBOTIC IAR. IASME transaction, 76-86.

[14] Hachour, O. (2004). N. Mastorakism Intelligent Control and planning of IAR, 3rd WSEAS International Multiconfrence on System Science and engineering. Copacabana Rio De Janeiro, Brazil.

[15] Buniyamin, N., Wan Ngah, W. A. J., Sariff, N., & Mohamad, Z. (2011). A simple local path planning algorithm for autonomous mobile robots. International journal of systems applications, engineering & development, 5(2), 151-159.

[16] Choset, H., & Pignon, P. (1998, January). Coverage path planning: The boustrophedon cellular decomposition. In Field and Service Robotics (pp. 203-209). Springer London.

[17] Sariff, N., & Buniyamin, N. (2006, June). An overview of autonomous mobile robot path planning algorithms. In Research and Development, 2006. SCOReD 2006. 4th Student Conference on (pp. 183-188). IEEE.

[18] Wan Ngah, W. A. J., Buniyamin, N., & Mohamad, Z. (2010). Point to Point Sensor Based Path Planning Algorithm for Mobile Robots. In 9th WSEAS International Conference on System Science and Simulation in Engineering (ICOSSSE'10).

[19] Vacariu, L., Roman, F., Timar, M., Stanciu, T., Banabic, R., & Cret, O. (2007, February). Mobile robot path-planning implementation in software and hardware. In Proceedings of the 6th WSEAS International Conference on Signal Processing, Robotics and Automation (pp. 140-145). World Scientific and Engineering Academy and Society (WSEAS).

[20] Haklıdır, M., Modelıng, T., & Arm, S. O. A. A. R. (2006). By Usıng Dymola. In 5th Int. Symp. on Intelligent Manufacturing Systems.

[21] Sariff, N., & Buniyamin, N. (2010, October). Ant colony system for robot path planning in global static environment. In 9th WSEAS International Conference on System Science and Simulation in Engineering (ICOSSSE'10).

[22] Ismail, A. T., Sheta, A., & Al-Weshah, M. (2008). A mobile robot path planning using genetic algorithm in static environment. Journal of Computer Science, 4(4), 341.

[23] Nieuwenhuisen, M., Steffens, R., & Behnke, S. (2012). Local multiresolution path planning in soccer games based on projected intentions. In RoboCup 2011: Robot Soccer World Cup XV (pp. 495-506). Springer Berlin Heidelberg.

[24] Brooks, R. A. (1983). Solving the find-path problem by good representation of free space. Systems, Man and Cybernetics, IEEE Transactions on, (2), 190-197.

[25] Shi, Y. (2001). Particle swarm optimization: developments, applications and resources. In Evolutionary Computation, 2001. Proceedings of the 2001 Congress on (Vol. 1, pp. 81-86). IEEE.

[26] Glavaski, D., Volf, M., & Bonkovic, M. (2009). Mobile robot path planning using exact cell decomposition and potential field methods. WSEAS Transactions on Circuits and Systems, 8(9), 789-800.

[27] Stentz, A. (1994, May). Optimal and efficient path planning for partially-known environments. In Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on (pp. 3310-3317). IEEE.

[28] Yahja, A., Singh, S., & Stentz, A. (2000). An efficient on-line path planner for outdoor mobile robots. Robotics and Autonomous systems, 32(2), 129-143.

[29] Yahja, A., Stentz, A., Singh, S., & Brumitt, B. L. (1998, May). Framed-quadtree path planning for mobile robots operating in sparse environments. In Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on (Vol. 1, pp. 650-655). IEEE.

[30] Singh, S., Simmons, R., Smith, T., Stentz, A., Verma, V., Yahja, A., & Schwehr, K. (2000). Recent progress in local and global traversability for planetary rovers. In Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on (Vol. 2, pp. 1194-1200). IEEE.

[31] Min, H. Q., Zhu, J. H., & Zheng, X. J. (2005, August). Obstacle avoidance with multi-objective optimization by PSO in dynamic environment. In Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on (Vol. 5, pp. 2950-2956). IEEE.

[32] Sedighi, K. H., Ashenayi, K., Manikas, T. W., Wainwright, R. L., & Tai, H. M. (2004, June). Autonomous local path planning for a mobile robot using a genetic algorithm. In Evolutionary Computation, 2004. CEC2004. Congress on (Vol. 2, pp. 1338-1345). IEEE.

[33] Schwartz, J. T., & Sharir, M. (1983). On the “piano movers” problem. II. General techniques for computing topological properties of real algebraic manifolds. Advances in applied Mathematics, 4(3), 298-351.

[34] Lumelsky, V., & Stepanov, A. (1986). Dynamic path planning for a mobile automaton with limited information on the environment. Automatic Control, IEEE Transactions on, 31(11), 1058-1063.

[35] Sariff, N. B., & Buniyamin, N. (2009, December). Comparative study of genetic algorithm and ant colony optimization algorithm performances for robot path planning in global static environments of different complexities. In Computational Intelligence in Robotics and Automation (CIRA), 2009 IEEE International Symposium on (pp. 132-137). IEEE.

[36] Xiong, L., Xiao-ping, F., Sheng, Y., & Heng, Z. (2004). A Novel Genetic Algorithm for Robot Path Planning in Environment Containing Large Numbers of Irregular Obstacles. Robot, 1, 15-22.

[37] Gao, X. G., Fu, X. W., & Chen, D. Q. (2005, August). A genetic-algorithm-based approach to UAV path planning problem. In Proceedings of the 5th WSEAS

international conference on Simulation, modelling and optimization (pp. 523-527). World Scientific and Engineering Academy and Society (WSEAS).

[38] Bin, N., Xiong, C., Liming, Z., & Wendong, X. (2005). Recurrent neural network for robot path planning. In Parallel and Distributed Computing: Applications and Technologies (pp. 188-191). Springer Berlin Heidelberg.

[39] Bell, J. E., & McMullen, P. R. (2004). Ant colony optimization techniques for the vehicle routing problem. Advanced Engineering Informatics, 18(1), 41-48.

[40] Sariff, N., & Buniyamin, N. (2009). Comparative Study of Genetic Algorithm and Ant Colony Optimization Algorithm in Global Static Environment of Different Complexities. In 2009 IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA 2009), Daejeon, Korea (pp. 132-137).

[41] Kumar, E. V., Aneja, M., & Deodhare, D. (2008). Solving a Path Planning Problem in a Partially Known Environment using a Swarm Algorithm. In IEEE International Symposium on Measurements and Control in Robotics.

[42] Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots. The international journal of robotics research, 5(1), 90-98.

[43] Talbi, E. G. (2009). Metaheuristics: from design to implementation (Vol. 74). John Wiley & Sons.

[44] Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. Computers & Operations Research, 13(5), 533-549.

[45] Gary, M. R., & Johnson, D. S. (1979). Computers and Intractability: A Guide to the Theory of NP-completeness.

[46] Lau, B., Sprunk, C., & Burgard, W. (2012). Efficient grid-based spatial representations for robot navigation in dynamic environments. Robotics and Autonomous Systems.

[47] Dorigo, M., & Stützle, T. (2010). Ant colony optimization: overview and recent advances. In Handbook of metaheuristics (pp. 227-263). Springer US.

[48] Chia, S. H., Su, K. L., Guo, J. H., & Chung, C. Y. (2010, December). Ant colony system based mobile robot path planning. In Genetic and Evolutionary Computing (ICGEC), 2010 Fourth International Conference on (pp. 210-213). IEEE.

[49] Han, Q., Wang, Q., Zhu, X., & Xu, J. (2011, April). Path planning of mobile robot based on improved ant colony algorithm. In Consumer Electronics, Communications and Networks (CECNet), 2011 International Conference on (pp. 531-533). IEEE.

[50] Wang, J. M., Yin, H., Xie, W. B., Qiu, G. Q., Xu, J. H., & Huang, Y. (2012). Research of Path Planning in Virtual Scenes Based on Ant Colony Algorithm and Axis Aligned Bounding Boxes Collision Detection Technology. In Advances in Automation and Robotics, Vol. 1 (pp. 615-622). Springer Berlin Heidelberg.

[51] Viet, N. H., Vien, N. A., Lee, S., & Chung, T. (2008, February). Obstacle avoidance path planning for mobile robot based on multi colony ant algorithm. In Advances in Computer-Human Interaction, 2008 First International Conference on (pp. 285-289). IEEE.

[52] Lee, J. W., & Lee, J. J. (2010, March). Novel Ant Colony Optimization algorithm with Path Crossover and heterogeneous ants for path planning. In Industrial Technology (ICIT), 2010 IEEE International Conference on (pp. 559-564). IEEE.

[53] Lee, J. W., Lee, D. H., & Lee, J. J. (2011, May). Global path planning using improved ant colony optimization algorithm through bilateral cooperative exploration. In Digital Ecosystems and Technologies Conference (DEST), 2011 Proceedings of the 5th IEEE International Conference on (pp. 109-113). IEEE.

[54] Chaari, I., Koubaa, A., Bennaceur, H., Trigui, S., & Al-Shalfan, K. (2012, June). smartPATH: A hybrid ACO-GA algorithm for robot path planning. In Evolutionary Computation (CEC), 2012 IEEE Congress on (pp. 1-8). IEEE.

[55] Xianlun, T. A. N. G., Guangdan, C. H. E. N., ZHANG, P., JIANG, B., & ZHANG, Y. (2012). Ant Colony Optimization Based on Maximum Selection Probability for Path Planning in Unknown Environment. Journal of Computational Information Systems, 8(24), 10325-10332.

[56] Wang, P. D., Tang, G. Y., Li, Y., & Yang, X. X. (2012, July). Ant colony algorithm using endpoint approximation for robot path planning. In Control Conference (CCC), 2012 31st Chinese (pp. 4960-4965). IEEE.

[57] Luo, D. L., & Wu, S. X. (2010). Ant colony optimization with potential field heuristic for robot path planning. Systems Engineering and Electronics, 32(6), 1277-1280.

[58] Wu, Y. F., Zhang, X. X., & Wu, J. Q. (2012). Using Cellular Ant Colony Algorithm for Path-Planning of Robots. Applied Mechanics and Materials, 182, 1776-1780.

[59] Zheng, Z., Shi-Rong, L., & Bo-Tao, Z. (2011, July). Global path planning of mobile robot based on improved ant colony algorithm. In Control Conference (CCC), 2011 30th Chinese (pp. 4083-4088). IEEE.

[60] Ismail, A. T., Sheta, A., & Al-Weshah, M. (2008). A mobile robot path planning using genetic algorithm in static environment. Journal of Computer Science, 4(4), 341.

[61] Tsai, C. C., Huang, H. C., & Chan, C. K. (2011). Parallel elite genetic algorithm and its application to global path planning for autonomous robot navigation. Industrial Electronics, IEEE Transactions on, 58(10), 4813-4821.

[62] Zeng, C., Zhang, Q., & Wei, X. (2011, January). Robotic Global Path-Planning Based Modified Genetic Algorithm and A* Algorithm. In Measuring Technology and Mechatronics Automation (ICMTMA), 2011 Third International Conference on (Vol. 3, pp. 167-170). IEEE.

[63] Gao, M., Xu, J., Tian, J., & Wu, H. (2008, October). Path planning for mobile robot based on chaos genetic algorithm. In Natural Computation, 2008. ICNC'08. Fourth International Conference on (Vol. 4, pp. 409-413). IEEE.

[64] Ghorbani, A., Shiry, S., & Nodehi, A. (2009, April). Using genetic algorithm for a mobile robot path planning. In Future Computer and Communication, 2009. ICFCC 2009. International Conference on (pp. 164-166). IEEE.

[65] Pehlivanoglu, Y. V. (2012). A new vibrational genetic algorithm enhanced with a Voronoi diagram for path planning of autonomous UAV. Aerospace Science and Technology, 16(1), 47-55.

[66] Lucas, D., & Crane, C. (2012, October). Development of a multi-resolution parallel genetic algorithm for autonomous robotic path planning. In Control, Automation and Systems (ICCAS), 2012 12th International Conference on (pp. 1002-1006). IEEE.

[67] LIU, C., YAN, X., LIU, C., & LI, G. (2010). Dynamic path planning for mobile robot based on improved genetic algorithm. Chinese Journal of Electronics, 19(2), 2010-2014.

[68] Hua, J. M. W. H. Z., & Xingzhe, X. (2011). APPLYING IMPROVED GENETIC ALGORITHM TO GLOBAL PATH PLANNING FOR MOBILE ROBOT. Computer Applications and Software, 8, 033.

[69] Yan, X. (2012). An Improved Robot Path Planning Algorithm. TELKOMNIKA (Telecommunication, Computing, Electronics and Control), 10(4), 629-636.

[70] Xu, X., Xie, J., & Xie, K. (2006). Path planning and obstacle-avoidance for soccer robot based on artificial potential field and genetic algorithm. In Intelligent Control and Automation, 2006. WCICA 2006. The Sixth World Congress on (Vol. 1, pp. 3494-3498). IEEE.

[71] Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization, IEEE International of first Conference on Neural Networks.

[72] Poli, R., Kennedy, J., & Blackwell, T. (2007). Particle swarm optimization. Swarm intelligence, 1(1), 33-57.

[73] Li, Q., Tang, Y., Wang, L., Zhang, C., & Yin, Y. (2010, August). A Specialized Particle Swarm Optimization for global path planning of mobile robots. In Advanced Computational Intelligence (IWACI), 2010 Third International Workshop on (pp. 271-276). IEEE.

[74] Shen, Y., & Yuan, M. (2010, May). A novel poly-clone particle swarm optimization algorithm and its application in mobile robot path planning. In Control and Decision Conference (CCDC), 2010 Chinese (pp. 2271-2276). IEEE.

[75] Chen, N. C., He, P., & Rui, X. M. (2010). Global Path Planning for Mobile Robot Based on Improved Dijkstra Algorithm and Particle Swarm Optimization. Applied Mechanics and Materials, 26, 909-912.

[76] Wu, X. X., Guo, B. L., & WANG, J. (2009). Mobile robot path planning algorithm based on particle swarm optimization of cubic splines. Robot, 31, 556-560.

[77] Liu, W. K. Z. H. Y., & Zhi-lei, C. H. A. I. (2010). Path Planning for Robots Based on Quantum-behaved Particle Swarm Optimization [J]. Microcomputer Information, 11, 066.

[78] Huang, H. C., & Tsai, C. C. (2011, September). Global path planning for autonomous robot navigation using hybrid metaheuristic GA-PSO algorithm. In SICE Annual Conference (SICE), 2011 Proceedings of (pp. 1338-1343). IEEE.

[79] Qian-zhi, M., & Xiu-juan, L. (2010, August). The application of hybrid orthogonal particle swarm optimization in robotic path planning. In Natural Computation (ICNC), 2010 Sixth International Conference on (Vol. 7, pp. 3536-3540). IEEE.

[80] Li, W., & Wang, G. Y. (2010, October). Application of improved PSO in mobile robotic path planning. In Intelligent Computing and Integrated Systems (ICISS), 2010 International Conference on (pp. 45-48). IEEE.

[81] Ma, Y., Zamirian, M., Yang, Y., Xu, Y., & Zhang, J. (2013). Path Planning for Mobile Objects in Four-Dimension Based on Particle Swarm Optimization Method with Penalty Function. Mathematical Problems in Engineering, 2013.

[82] Aarts, E. H., Korst, J. H., & van Laarhoven, P. J. (1997). Simulated annealing, Local Search in Combinatorial Optimization (Emile Aarts and Jan Karel Lenstra, eds.).

[83] Liang, Y. M., & Xu, L. H. (2010). Mobile robot global path planning based modified simulated annealing hybrid algorithm. Control and Decision, 25(2), 237-240.

[84] Tavares, R. S., Martins, T. C., & Tsuzuki, M. S. G. (2011). Simulated annealing with adaptive neighborhood: A case study in off-line robot path planning. Expert Systems with Applications, 38(4), 2951-2965.

[85] Ho, Y. J., & Liu, J. S. (2010, March). Simulated annealing based algorithm for smooth robot path planning with different kinematic constraints. In Proceedings of the 2010 ACM Symposium on Applied Computing (pp. 1277-1281). ACM.

[86] Liang, Y., & Xu, L. (2009, June). Global path planning for mobile robot based genetic algorithm and modified simulated annealing algorithm. In Proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation (pp. 303-308). ACM.

[87] DU, Z. Z., & LIU, G. D. (2009). Path Planning of Mobile Robot Based on Genetically Simulated Annealing Algorithm. Computer Simulation, 12, 036.

[88] Wang, H. B., Yang, W. J., & Wang, J. H. (2012). Research on Path Planning for Mobile Robot Based on Grid and Hybrid of GA/SA. Advanced Materials Research, 479, 1499-1503.

[89] Hussein, A., Mostafa, H., Badrel-din, M., Sultan, O., & Khamis, A. (2012, October). Metaheuristic optimization approach to mobile robot path planning. In Engineering and Technology (ICET), 2012 International Conference on (pp. 1-6). IEEE.

[90] Geem, Z. W. (2010). Recent advances in harmony search algorithm (Vol. 270). Springer.

[91] Geem, Z. W. (2008). Novel derivative of harmony search algorithm for discrete design variables. Applied Mathematics and Computation, 199(1), 223-230.

[92] Geem, Z. W. (Ed.). (2009). Music-inspired harmony search algorithm: theory and applications (Vol. 191). Springer.

[93] Geem, Z. W. (Ed.). (2009). Harmony search algorithms for structural design optimization (Vol. 239). Springer.

[94] Mahdavi, M., Fesanghary, M., & Damangir, E. (2007). An improved harmony search algorithm for solving optimization problems. Applied mathematics and computation, 188(2), 1567-1579.

[95] Mukhopadhyay, A., Roy, A., Das, S., & Abraham, A. (2008, November). Population-variance and explorative power of harmony search: an analysis. In Digital Information Management, 2008. ICDIM 2008. Third International Conference on (pp. 775-781). IEEE.

[96] Geem, Z. W. (2006). Optimal cost design of water distribution networks using harmony search. Engineering Optimization, 38(03), 259-277.

[97] Wang, C. M., & Huang, Y. F. (2010). Self-adaptive harmony search algorithm for optimization. Expert Systems with Applications, 37(4), 2826-2837.

[98] Degertekin, S. O. (2008). Optimum design of steel frames using harmony search algorithm. Structural and Multidisciplinary Optimization, 36(4), 393-401.

[99] Geem, Z. W. (2006, January). Improved harmony search from ensemble of music players. In Knowledge-Based Intelligent Information and Engineering Systems (pp. 86-93). Springer Berlin Heidelberg.

[100] Finkel, R. A., & Bentley, J. L. (1974). Quad trees a data structure for retrieval on composite keys. Acta informatica, 4(1), 1-9.

[101] Waresiak, B., & Skrzyński, P. (2011). Using quad tree as data storage for a terrain representation and a core for a path finding algorithm. Automatyka/Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie, 15, 681-691.

[102] Samal, A., Bhatia, S., Vadlamani, P., & Marx, D. (2009). Searching satellite imagery with integrated measures. Pattern Recognition, 42(11), 2502-2513.

[103] Trotts, I., Mikula, S., & Jones, E. G. (2007). Interactive visualization of multiresolution image stacks in 3D. NeuroImage, 35(3), 1038-1043.

[104] Chen, D. Z., Szczerba, R. J., & Uhran Jr, J. J. (1995, August). Planning conditional shortest paths through an unknown environment: A framed-quadtree approach. In Intelligent Robots and Systems 95.'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on (Vol. 3, pp. 33-38). IEEE.

[105] Gabrani, G., & Mulkar, T. (2005). A quad-tree based algorithm for processor allocation in 2D mesh-connected multicomputers. Computer Standards & Interfaces, 27(2), 133-147.

[106] Sun, M. (2006). A primogenitary linked quad tree data structure and its application to discrete multiple criteria optimization. Annals of Operations Research, 147(1), 87-107.

[107] Aizawa, K., Motomura, K., Kimura, S., Kadowaki, R., & Fan, J. (2008, March). Constant time neighbor finding in quadtrees: An experimental result. In Communications, Control and Signal Processing, 2008. ISCCSP 2008. 3rd International Symposium on (pp. 505-510). IEEE.

[108] Samet, H. (1990). The design and analysis of spatial data structures (Vol. 199). Reading, MA: Addison-Wesley.

[109] Samet, H. (1990). Applications of spatial data structures.

[110] Gargantini, I. (1982). An effective way to represent quadtrees. Communications of the ACM, 25(12), 905-910.

[111] Schrack, G. (1992). Finding neighbors of equal size in linear quadtrees and octrees in constant time. CVGIP: Image Understanding, 55(3), 221-230.

[112] Aizawa, K., & Tanaka, S. (2009). A constant-time algorithm for finding neighbors in quadtrees. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 31(7), 1178-1183.

[113] Schrijver, A. (1998). Theory of linear and integer programming. Wiley. com.

[114] ARobot Mobile Robot for Experimenters and Educators (2013), http://www.arrickrobotics.com/arobot.

[115] Parallax (2013), http://www.parallax.com.

Стојанче Панов

ПЛАНИРАЊЕ НА ОПТИМАЛНА ТРАЕКТОРИЈА НА МОБИЛНИ РОБОТИ КОРИСТЕЈЌИ МУЗИЧКИ ИНСПИРИРАНИ АЛГОРИТАМИ

Универзитет „Гоце Делчев“ – Штип